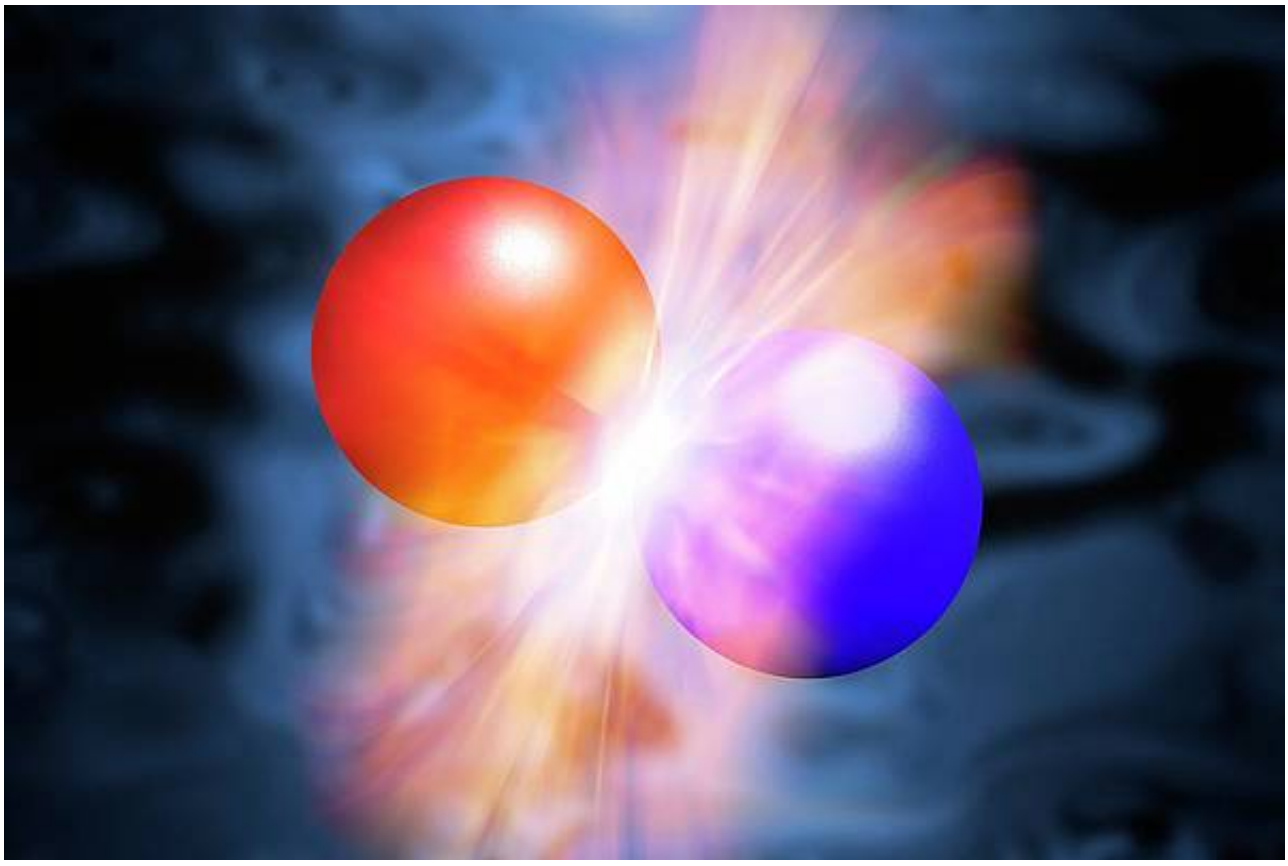
TÉCNICO LISBOA

Let's make some noisR

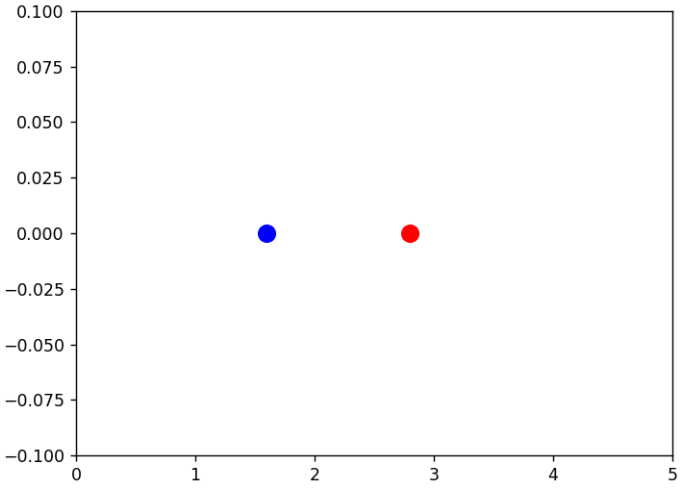
Particle Collision

Vasco Pires & Duarte Gonçalves



1 Introdução

Neste projeto vais recriar uma colisão entre duas partículas (p1 e p2) tendo em conta a conservação do seu momento e as suas massas (m1 e m2) no seguinte cenário:



2 Imports

Antes de escreveres o teu código é importante fazeres os seguintes imports para poderes usar as bibliotecas necessárias:

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation
```

3 Funções

Para o código em si, vais desenvolver a função:

```
def simulate_collision(initial_pos1, initial_pos2, initial_velocity1,
initial_velocity2, mass1, mass2, num_frames, box_size):
```

onde os argumentos são:

1. initial_pos1: posição inicial de p1
2. initial_pos2: posição inicial de p2
3. initial_velocity1: velocidade inicial de p1
4. initial_velocity2: velocidade inicial de p2
5. mass1: massa de p1
6. mass2: massa de p2
7. num_frames: número de frames da simulação (ignorem esta parte)
8. box_size: tamanho da caixa

Nesta função tens de ter em conta o momento linear de cada partícula tendo em conta a sua velocidade e massa.

Atenção: As colisões irão ocorrer não só entre as bolas mas também com as paredes da caixa.

4 Animação

Para poderes ver a animação do teu código, isto é, o movimento das bolas, pedimos que cries dois vetores:

1. ball1_x: onde irás guardar as posições de p1 em cada frame
2. ball2_x: onde irás guardas as posições de p2 em cada frame

Podes colocar a seguinte função no teu código (que te permitirá ver a animação):

```
def create_animation(positions1, positions2, box_size):
    num_frames = len(positions1)

    fig, ax = plt.subplots()
    ax.set_xlim(0, box_size)
    ax.set_ylim(-0.1, 0.1)

    ball1, = ax.plot(positions1[0], 0, 'bo', markersize=10)
    ball2, = ax.plot(positions2[0], 0, 'ro', markersize=10)

    def update(frame):
        ball1.set_xdata(positions1[frame])
        ball2.set_xdata(positions2[frame])
        return ball1, ball2

    ani = FuncAnimation(fig, update, frames=num_frames, blit=True)
    plt.show()

    plt.close(fig)
```

No fim da tua função "simulate_collision" coloca a seguinte linha de código:

```
create_animation(ball1_x, ball2_x, box_size)
```

para chamares a função que mostrámos em cima.

5 Exemplo

Podes pegar no exemplo seguinte e completá-lo:

```python
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.animation import FuncAnimation

def simulate_collision(initial_pos1, initial_pos2, initial_velocity1, initial_velocity2, mass1, mass2, num_frames, box_size):
    ball1_x = [initial_pos1]
    ball2_x = [initial_pos2]

    # ... o teu código

    create_animation(ball1_x, ball2_x, box_size)

def create_animation(positions1, positions2, box_size):
    # colocar o código da função dada no enunciado

    return # ignorem esta linha

# Definam as vossas variáveis isto é apenas um exemplo
initial_pos1 = 1
initial_pos2 = 4
initial_velocity1 = 0.1
initial_velocity2 = -0.1
mass1 = 1.0
mass2 = 1.5
num_frames = 100
box_size = 5

#chamem no final a função que simula a colisão para porem o vosso programa a correr
simulate_collision(initial_pos1, initial_pos2, initial_velocity1, initial_velocity2, mass1, mass2, num_frames, box_size)
```

Foi apenas dada uma imagem do código para treinarem implementar por vocês ;)

Boa sorte e saudações noisirianas para todos !