INSTITUTO LATINO AMERICANO DE CIÊNCIAS DA VIDA E DA NATUREZA (ILACVN)

Curso: Engenharia Física

Disciplina: Física Matemática

Professor: LUCIANO CALHEIROS LAPAS

Problema 1 - Vetores

Estudantes: Edrice Basil e Wallace Pannace Palma

Foz do Iguaçu, 05 de Setembro de 2022.

Problema

1- Assumindo um sistema geral tridimensional com 'n' partículas carregas, considere:
-cada partícula pode ter uma posição inicial e velocidade inicial
-Não há resistência ao movimento destas partículas
-Todas interações menos a elétrica são irrelevantes!
-Assuma que as partículas possuem massa e são adimensionais (pontuais)
Análise como este sistema se desenvolve com o tempo (partindo de t =0).
2- O que ocorrerá se as partículas estiverem presas em uma esfera de raio r?
3- Construa um modelo que demonstre o fenômeno.

Solução

Disponível no repositório (github):
https://github.com/Wallace-p-p/N-charged-particles-in-a-sphere-with-mplot3d

1- A interação será descrita a partir das forças elétricas, utilizamos a lei de Coulomb:

$$F = k \cdot \frac{q1.q2}{d^2}$$

Onde a força resultante sobre uma partícula é a soma de cada força que atua sobre ela pela interação das outras partículas do sistema:

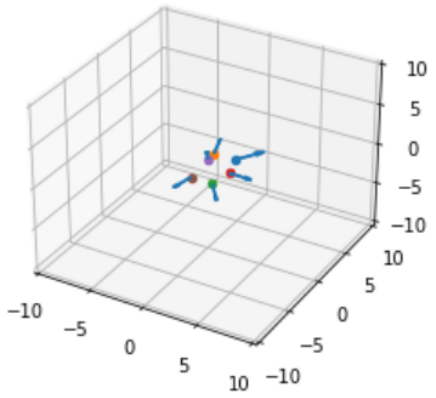
```
#Eletric force F = F1+F2+...
elF= []
k0 = 8.98755 * 10**9
for j in range(len(p)):
    forces=[]
    for i in range(len(p)):
        if(i != j):
            radiusx = p[j][1][0] - p[i][1][0]
            radiusy = p[j][1][1] - p[i][1][1]
            radiusz = p[j][1][2] - p[i][1][2]
            radiusValue = (radiusx*radiusx + radiusy*radiusy + radiusz*radiusz)**(1/2)
            forceVector = [radiusx/radiusValue, radiusy/radiusValue, radiusz/radiusValue]
            forceValue = ( k0*p[j][0]*p[i][0] )/(radiusValue**2)
            forceVector[0] = forceVector[0]*forceValue
            forceVector[1] = forceVector[1]*forceValue
            forceVector[2] = forceVector[2]*forceValue
            forces.append(forceVector)
    force = [0.,0.,0.]
    for k in range(len(forces)):
        force[0] = force[0]+ forces[k][0]
        force[1] = force[1]+forces[k][1]
        force[2] = force[2]+forces[k][2]
    elF.append(force)
```

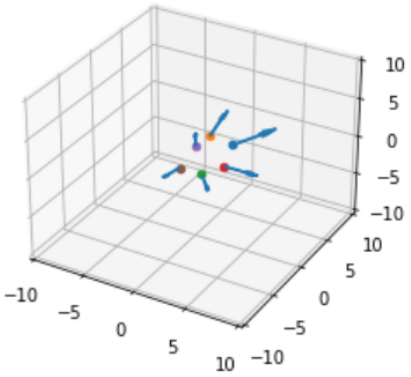
A partir disto, dados as condições iniciais de cada partícula podemos definir para onde vai usando: $S = S_0 + V_0 t + \frac{at^2}{2}$ e $v^2 = v_0^2 + 2 \cdot a \cdot \Delta s$.

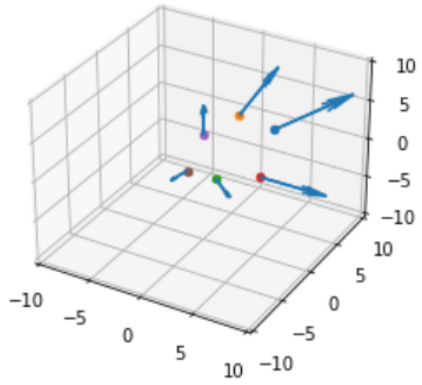
```
#New values for the particles
for i in range(len(p)):
    for j in range(3):
        px = p[i][1][j] + p[i][2][j]*t + (a[i][j]*(t**2))/2
        if(px - p[i][1][j]<0):
            p[i][2][j] = -((p[i][2][j])**2 + 2*a[i][j]*(px - p[i][1][j]))**(1/2)
        else:
            p[i][2][j] = ((p[i][2][j])**2 + 2*a[i][j]*(px - p[i][1][j]))**(1/2)

        p[i][1][j] = px
    mod = (p[i][1][0]**2 + p[i][1][1]**2 + p[i][1][2]**2)**(1/2)
    if( mod > 10):
        for uu in range(3):
            p[i][1][uu] = p[i][1][uu]*10/mod
```

Com isso montamos um algoritmo iterativo para a cada instante de tempo determinado gerar um gráfico 3D da posição de cada partícula com os seus vetores de velocidade e aceleração:





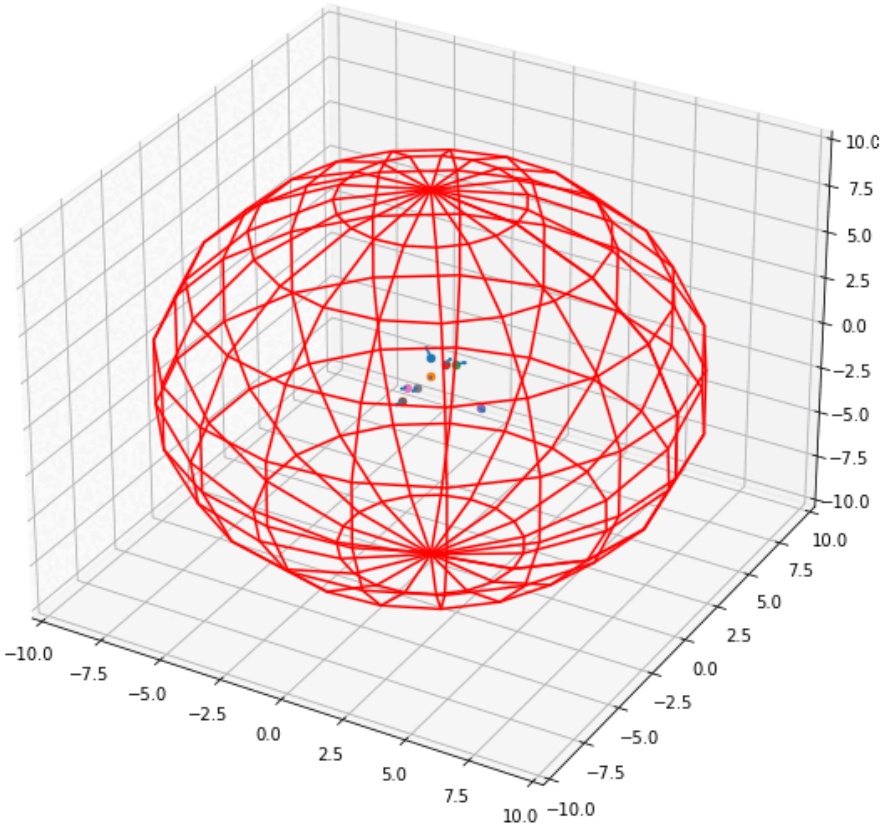


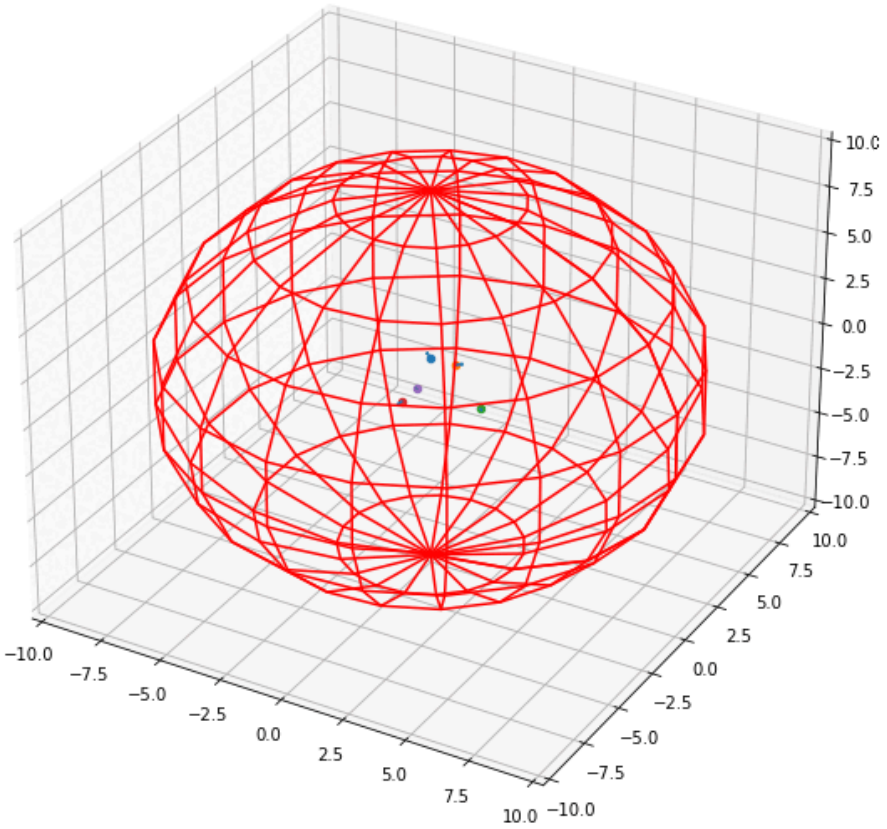
2- se estiverem presas em uma esfera de raio 10:

```
#New values for the particles
for i in range(len(p)):
    for j in range(3):
        px = p[i][1][j] + p[i][2][j]*t + (a[i][j]*(t**2))/2
        if(px - p[i][1][j]<0):
            p[i][2][j] = -((p[i][2][j])**2 + 2*a[i][j]*(px - p[i][1][j]))**(1/2)
        else:
            p[i][2][j] = ((p[i][2][j])**2 + 2*a[i][j]*(px - p[i][1][j]))**(1/2)

        p[i][1][j] = px
    mod = (p[i][1][0]**2 + p[i][1][1]**2 + p[i][1][2]**2)**(1/2)
    if( mod > 10):
        for uu in range(3):
            p[i][1][uu] = p[i][1][uu]*10/mod
t= t+0.1
imgid=imgid+1
```

Obtemos o seguinte resultado:




10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0
10.0
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0