Documentação Problema de Três Corpos
Introdução à Computação em Física

Alexânia Maria Batista Soares
Ana Luisa Rodrigues Ferreira
André Luiz Soares Martins

Universidade Federal de Minas Gerais

1 Introdução

Em 1687, Isaac Newton escreveu, em seu livro *Principia I*, a lei da gravitação universal. Esta descreve como dois corpos interagem entre si devido à força de atração entre as massas. Seu objetivo era explicar as leis de Kepler, que, por sua vez, tinha como objetivo, ao escrever suas leis, explicar matematicamente as observações do astrônomo Tycho Brahe[1].

Durante muitos anos, o foco da astronomia era o sistema solar. Como a massa do Sol é muito maior que a soma das massas de todos os planetas que o orbitam, o centro de massa do sistema solar está dentro do Sol e, devido a isso, o sistema solar pode ser considerado um sistema de dois corpos[2].

Contudo, frequentemente, tem havido a necessidade de analisar como mais de dois corpos com massas parecidas interagem entre si. Esse conhecimento é importante para colocar satélites artificiais em órbita, para evitar colisão de corpos celestes com a Terra, para entender sistemas solares com estralas binárias etc. Inclusive na mecânica quântica há inúmeras situações em que vários corpos interagem de forma inesperada[3], porém estes não podem ser explicados pela mecânica newtoniana, que é a que vai ser usada neste projeto.

Apesar de todos os esforços das áreas da matemática e da física, o sistema de vários corpos continua sendo uma incógnita e o comportamento dos corpos nesse sistema é muito imprevisível. Essa situação ficou conhecida na astrofísica como problema de n corpos, em que n se refere a um número maior que três[4].

Nesse projeto, visa-se analisar como três corpos interagem entre si, levando em consideração infinitas combinações de condições iniciais. Exemplos de parâmetros que podem interferir completamente nas órbitas dos corpos são: a massa dos corpos, suas velocidades e posições iniciais. Para que isso seja possível, implementar-se-á um código em Python que gera uma animação, que será melhor explicado a seguir.

2 Objetivos

Demonstrar graficamente órbitas de três objetos corpos astronômicos massivos, interagindo entre si, utilizando a lei da gravitação newtoniana e mecânica clássica.

3 Metodologia teórica

Da lei da gravitação universal de Newton, mostrada na equação 1, é possível encontrar o sistema de equações ordinárias acopladas[5] com as seguintes equações,

$$m_i \frac{d^2 \overrightarrow{r}}{dt^2} = \frac{G m_i m_j}{r_i^3} \overrightarrow{r}_i \tag{1}$$

$$\frac{d^2 \overrightarrow{r_1}}{dt^2} = G m_2 \frac{r_1 - r_2}{|r_1 - r_2|^3} + G m_3 \frac{r_1 - r_3}{|r_1 - r_3|^3} \tag{2}$$

$$\frac{d^2 \overrightarrow{r_2}}{dt^2} = G m_1 \frac{r_2 - r_1}{|r_2 - r_1|^3} + G m_3 \frac{r_2 - r_3}{|r_2 - r_3|^3} \tag{3}$$

$$\frac{d^2 \overrightarrow{r_3}}{dt^2} = G m_1 \frac{r_3 - r_1}{|r_3 - r_1|^3} + G m_2 \frac{r_3 - r_2}{|r_3 - r_2|^3} \tag{4}$$

pelas quais determinam as posições de cada um dos três corpos. Estas dependem diretamente da força gravitacional entre os corpos e da distancia entre eles, que são analisadas em pares.

Assim, com a informação a segunda derivada a posição de cada corpo, é possível prosseguir utilizando-se de dois métodos:

1. Resolver a equação diferencial de segunda ordem utilizando métodos numéricos disponíveis na biblioteca *SciPy*, com auxílio da função $scipy.integrate.odeint()$. A função *odeint* retorna o valor numérico da equação diferencial de segunda ordem, sendo necessários alguns parâmetros iniciais: a posição inicial, a velocidade inicial (que é a primeira derivada da posição), uma função que abriga as equações da equação diferencial de segunda ordem, onde serão resolvidas e um malha de tempo.

2. Resolver a equação diferencial usando um método de aproximação que foi anteriormente usado para animar objetos usando a biblioteca *VPython*.Os valores da posição inicial, velocidade inicial e massa são pré-determinados, podendo calcular também o momento de cada corpo. Para calcular as posições de um corpo ao longo do tempo é primeiro necessário encontrar a força resultante, que é multiplicação da massa de cada corpo por cada segunda derivada referente a ele, apresentada anteriormente. Logo em seguida, calcula-se o momento do corpo, com base no momento inicial, na força resultante calculada e um intervalo de tempo (*dt*). Com o valor do momento do corpo, é possível encontrar a sua posição. Tudo isso é concatenado dentro de um laço que itera sobre esses cálculo por um certo período de tempo pré-determinado. Um exemplo do algoritmo usado está disposto pelas equações à seguir:

$$F_{res} = \frac{Gm_j m_k}{r_i^3} \overrightarrow{r}_i \tag{5}$$

$$p_{i+1} = p_i + F_{res} \cdot dt \tag{6}$$

$$r_{i+1} = r_i + (\frac{p_{i+1}}{m_j}) \cdot dt \tag{7}$$

Para ambos métodos, que são numéricos, é necessário que não haja uma grande discrepância entre a ordem dos dados trabalhados para que a convergência seja satisfatória. Assim, é feita uma mudança nos dados com base em valores conhecidos de algumas estrelas e planetas. O valor da constante da gravitação universal é modificado. a partir da terceira lei de Kepler, para que fique em função de: massa do sol, distância média entre entre as duas estrelas escolhidas como parâmetro, Alpha Centauri A e Alpha Centauri B, velocidade orbital média da Terra e o período orbital médio entre as duas estrelas em torno do centro de massa do seu sistema. No primeiro método, também, é calculada uma constante, a partir da equação de posição do movimento uniforme ($s = s_0 + v \cdot t$) para modificar os valores da velocidade em função do tempo, para que possa ser utilizada no método de resolução da equação de segunda ordem, já que a velocidade é a derivada da posição. Assim, essa constante multiplica a velocidade a cada iteração. A massa dos corpos é deixada na unidade de medida de massas solares e a posição inicial e velocidade inicial são colocadas em metros e metros por segundo, respectivamente, porém com valores bem pequenos, já que estão proporcionais aos valores de referência determinados anteriormente. Ao final da resolução das equações diferencias, as posições de cada corpo são armazenadas em vetores para que sejam *plotadas* com auxílio da biblioteca *Matplotlib*. São feitos *plots* relacionando suas posições no x em função do tempo, uma visão do plano x e y da trajetória e uma visão de três dimensões na trajetória, apresentando suas posições no x, y e z em função do tempo.

4 Instruções para uso do código

No repositório do GitHub de link `https://github.com/alexaniamaria/icf-problema-tres-corpos`, é possível encontrar cinco arquivos: um arquivo *.ipynb* que apresenta o primeiro método explicado na seção anterior, chamado de *problema-de-três-corpos-odeint.ipynb*; um arquivo *.ipynb* que apresenta o segundo método explicado na seção anterior, chamado de *problema-de-três-corpos-euler-vpython.ipynb*; um arquivo *README.md* em branco; um arquivo *.gif* que demonstra um resultado do segundo método; esta documentação. Ambos arquivos possuem os mesmo valores iniciais, permitindo que o usuário escolha entre a opção 1 de velocidade inicial ou a opção 2 de velocidade inicial. A posição inicial é fixa para uma melhor demonstração das trajetórias. Se o usuário desejar mudar os dados iniciais manualmente para ver o resultado, é de extrema importância analisar se as velocidade estão fazendo os corpos se aproximarem ou distanciarem, porque, se os vetores da posição inicial e velocidade inicial forem opostos, não haverá interação aparente. Também, se as velocidades iniciais forem valores muito altos, os corpos escapam de seus campos gravitacionais rapidamente, não havendo interação aparente (é aconselhado que utilize valores próximos entre $-0.01\, m/s$ e $0.01\, m/s$ para as velocidade iniciais). Para uma melhor visualização das interações e trajetórias de cada corpo, ao final do código é possível baixar um arquivo *.gif* que mostra as posições em um período de tempo, deixando a opção *anima.save()* não comentada. Se preferir por uma visualização dentro do próprio código, deve-se comentar a opção anterior e deixar a linha que inicia com *HTML* não comentada. Para animar o gráfico dentro do código ou baixá-lo, é levado cerca de dois minutos.

OBS.: É necessário reiniciar o kernel toda vez que for executar o código, pelo fato de os valores serem concatenados numa lista, ao invés de serem resetados. Dessa maneira, para uma melhor performance do código em seu computador e para que não acumulem 1 milhão de pontos para serem plotados, por favor, reinicie o kernel.

References

[1] PIRES, Antônio S. T. **Evolução das ideais da física**. São Paulo, editora Livraria da Física, 2008.

[2] RANGEL, Ana Paula da Cunha de Melo Silva. **O movimento de dois corpos**. 2001. Dissertação de mestrado - Faculdade de Ciências da Universidade do Porto. Porto, 2001.

[3] GRIFFITH, David Jeffrey. **Introduction to quantum mechanics**. 1ª edição. Nova Jersey: Prentice Hall, 1995.

[4] FIGUEREDO, Elysandra; CASTRO, Antonio S. de. Um Problema de Três Corpos Analiticamente Solúvel. **Revista Brasileira de Ensino de Física**, vol. 23, no. 3, Setembro, 2001.

[5] SOUZA, Jefferson Wherculis da Silva. **Mecânica Celeste:** uma análise dinâmica do problema restrito de três corpos para astro do sistema solar. Trabalho de Conclusão de Curso - Instituto Federal do Piauí, 2018.

[6] ROMANAZZI, Giuseppe. **Métodos de Alta Ordem:** Taylor, Runge-Kutta, Euler aperfeiçoado. Unicamp, 2020. Disponível em: `https://www.ime.unicamp.br/~roman/courses/MS211/1s2020/problemavalorinicial3.pdf`. Acesso em 4 de maio de 2023.

[7] DESHMUKH, Gaurav. Modelling the Three Body Problem in Classical Mechanics using Python. **Towards Data Science**. Disponível em: `https://towardsdatascience.com/modelling-the-three-body-problem-in-classical-mechanics-using-python-9dc270ad7767`. Acesso em 1 de maio de 2023.

[8] API Reference - scipy.integrate.odeint. SciPy Documentation. Disponível em: `https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.odeint.html`. Acesso em 2 de maio de 2023.

[9] Gravitation constant. Wikipedia. Disponível em: `https://en.wikipedia.org/wiki/Gravitational_constant`. Acesso em 3 de julho de 2023.

[10] Alpha Centauri. Wikipedia. Disponível em: `https://pt.wikipedia.org/wiki/Alpha_Centauri`. Acesso em 3 de julho de 2023.

[11] Translação da Terra. Wikipedia. Disponível em: `https://pt.wikipedia.org/wiki/Transla%C3%A7%C3%A3o_da_Terra`. Acesso em 3 de julho de 2023.