Trayectorias gravitacionales: Explorando el problema de los dos cuerpos con métodos computacionales

Laura Sofía Sierra Sánchez
Juan Sebastian Novoa Ortiz

1. Descripción del Problema

Este proyecto aborda el problema de dos cuerpos de la mecánica celeste, que busca predecir el movimiento de un cuerpo bajo la influencia gravitacional de otro. La dinámica del sistema se describe mediante la siguiente Ecuación Diferencial Ordinaria (EDO) vectorial:

$$\frac{d^2\vec{r}}{dt^2} = -G(M_1 + M_2)\frac{\vec{r}}{|\vec{r}|^3}$$

donde $\vec{r}$ es el vector de posición relativa, G es la constante de gravitación universal, y M_1 y M_2 son las masas de los cuerpos. Dado que esta ecuación no tiene una solución analítica simple para la posición en función del tiempo, $\vec{r}(t)$, es necesario resolverla numéricamente.

Relevancia: La solución a este problema es fundamental en física y astronomía, siendo la base para comprender la estabilidad de sistemas solares, el movimiento de estrellas binarias y el diseño de trayectorias para satélites y misiones espaciales.

Objetivo General: Desarrollar un programa en Python para simular la trayectoria de un cuerpo celeste en un sistema de dos cuerpos, calculando su posición, velocidad y aceleración a lo largo del tiempo mediante la integración numérica de las ecuaciones del movimiento.

Objetivos Específicos:

- Implementar la EDO del problema en una función de Python.
- Utilizar `scipy.integrate.solve_ivp` para integrar numéricamente el sistema a partir de condiciones iniciales dadas.
- Validar la simulación verificando la conservación de la energía total y el momento angular.

- Visualizar los resultados, incluyendo la órbita y las variables de estado y conservación.

2. Concepto(s) a Aplicar

- **Ecuaciones Diferenciales Ordinarias (EDOs):** Es el concepto central, ya que la física del problema se formula como una EDO que se debe resolver mediante integración numérica.
- **Matrices (Álgebra Lineal):** El estado del sistema (posición y velocidad) se representará y manipulará como vectores (arrays de NumPy), lo cual es esencial para una implementación eficiente.
- **Interpolación:** Se utilizará como herramienta de post-procesamiento para analizar los resultados del integrador en instantes de tiempo específicos.

3. Metodología (Enfoque Computacional)

El proyecto se desarrollará siguiendo un plan estructurado:

1. **Configuración:** Importar librerías (`NumPy`, `SciPy`, `Matplotlib`) y definir constantes físicas y parámetros de la simulación.
2. **Condiciones Iniciales:** Establecer el vector de estado inicial del sistema $\vec{S}_0 = [x_0, y_0, z_0, v_{x0}, v_{y0}, v_{z0}]$.
3. **Implementación de la EDO:** Crear una función de Python que calcule la derivada del vector de estado, $d\vec{S}/dt$.
4. **Integración Numérica:** Emplear `scipy.integrate.solve_ivp` para resolver la EDO.
5. **Análisis y Validación:** A partir de la solución, calcular en cada paso la energía total y el momento angular para verificar su conservación.

$$\text{Energía Total: } E = \frac{1}{2}M_2|\vec{v}|^2 - \frac{GM_1M_2}{|\vec{r}|}$$

$$\text{Momento Angular: } |\vec{L}| = |\vec{r} \times (M_2\vec{v})|$$

4. Resultados Esperados

Se espera obtener los arrays con la solución numérica (posición y velocidad en función del tiempo), así como los cálculos de la energía y momento angular para la validación. Estos resultados se presentarán mediante un conjunto de gráficas claras y concisas:

- Una gráfica 2D de la trayectoria orbital, que debe ser una elipse cerrada.
- Gráficas de las componentes de velocidad y aceleración en función del tiempo.
- Gráficas de validación que muestren la conservación de la energía y el momento angular, esperando líneas prácticamente horizontales.

5. Referencias

1. Taylor, J. R. (2005). *Classical Mechanics*. University Science Books.
2. Newman, M. (2013). *Computational Physics*. CreateSpace Independent Publishing Platform.
3. Giordano, N. J., & Nakanishi, H. (2005). *Computational Physics*. Pearson/Prentice Hall.