Université Lyon 1 Claude Bernard
Polytech Lyon
Année 2024/2025

Problème N-corps

Auteur:
Romain Urbaniak

Encadrant:
Noélie Di-Cesare

Contents

1 Introduction

Le problème à N corps est un problème fondamental en mécanique céleste qui consiste à déterminer l'évolution dans le temps du mouvement de N corps sous l'influence de leurs interactions gravitationnelles. Contrairement au problème à deux corps, pour lequel une solution exacte existe, le problème à N corps ne peut pas être résolu analytiquement à cause de la complexité des équations du mouvement.

Ce problème est très important pour comprendre des phénomènes astrophysiques comme la dynamique des systèmes planétaires, la formation de galaxies, et les interactions entre étoiles.Le but de ce TP est donc de mettre en place

2 Objectifs du projet :

L'objectif de ce TP est de développer un programme en C++ pour calculer les positions des planètes du système solaire, puis d'utiliser Python pour analyser et traiter les données obtenues.

3 Code en language C++

Le code en C++ mis en place dans ce TP simule le mouvement de plusieurs corps sous l'influence de la gravité en résolvant les équations différentielles associées. On va implémenter plusieurs classes permettant de décrire chaque corps :

3.1 Classe coordonées

Cette classe, dotée de deux attributs de type `double`, `x` et `y`, représente la position d'un point dans l'espace. Elle intègre également plusieurs méthodes permettant d'effectuer des opérations fondamentales, telles que la translation d'un point ou le calcul de la distance entre deux points.

3.2 Classe Vecteur

La classe `vect` est une classe qui hérite de la classe `coordonnées`, elle permet de représenter et manipuler des vecteurs dans un espace 2 dimensions. Cette classe inclut des méthodes pour calculer la norme et l'angle d'un vecteur, effectuer des rotations. De plus, elle surcharge plusieurs opérateurs arithmétiques, tels que l'addition entre vecteurs, la multiplication par un scalaire et la division par un nombre, facilitant ainsi les manipulations vectorielles.

3.3 Classe Point matériel

La classe `pointmat` permet de représenter un point matériel en mouvement. Nous utilisons plusieurs attributs tels qu'un vecteur vitesse (`v`) de type vecteur, une masse (`m`) de type double et une position (`p`) de type coordonées. Principalement, cette classe, en utilisant des méthodes et des surcharges des 2 classes précédentes permet l'intégration numérique de la position et de la vitesse en fonction du temps (`integre_position` et `integre_vitesse`). De plus, nous avons implémenté une méthodes force qui calcul l'intéraction gravitationelle entre 2 corps célestes par la formule d'intéraction suivante :

$$F = G\frac{m_1 m_2}{r^2} \tag{1}$$

3.4 Résolution

Grâce à ces 3 classes, on peut modéliser chaque corps par leur position dans l'espace, leur masse et leur vitesse. Partant de ces 3 caractéristiques, on peut modéliser le mouvement ou trajectoire de plusieurs planètes sur une période définit préalablement. Voici la charnière du code implémenté :

- **Initialisation des planètes :** On inscrit le nom des corps que nous souhaitons étudier ainsi que leur position initiale, leur masse et leur vecteur vitesse initiale.
- **Ouverture d'un fichier texte :** On ouvre un fichier texte pour y écrire les positions en x et y à chaque pas de temps.

- **Boucle de simulation :** On calcule les forces gravitationelles entre chaque planètes puis on mets à jour la vitesse et la position à l'aide des méthodes `set_position` et `set_vitesse` en utilisant le principe fondamentale de la dynamique à chaque pas de temps.
- A chaque itérations, les positions en x et y de chaque planètes sont écrites dans le fichier text ouvert préalablement.

4 Traitement des données en Python

Une fois les données simulées en C++ et exportées sur le fichier texte, on utilise le langage Python pour traiter ces résultats.Ce langage permet une bonne utilisation des données pour tracer les trajectoires de chaque planètes.

- **Récupération des valeurs :** On récupère les valeurs du fichier texte puis on les insères dans des tableaux de manière à avoir 2 tableaux par planètes.
- **Tracé des trajectoires :** On peut ensuite tracer les trajectoires des planètes dans le plan (x,y) en fonction du temps de simulation.

5 Visualisation des résultats

MAintenant que nous avons tout les outils nécessaires pour tracer des trajectoires de corps céléstes, nous pouvons étudier différents systèmes.

5.1 Système TERRE-LUNE

Nous commençons par modéliser un système simple TERRE-LUNE pour vérifier et valider nos implémentations en C++ et python. Nous modélisons donc les 2 corps célestes sur un temps total de 27 jours pour que la Lune fasse un tour complet de la Terre, avec un pas de temps de 1 heure en centrant la simulation sur la TERRE. Nous obtenons la trajectoire de la lune suivante :

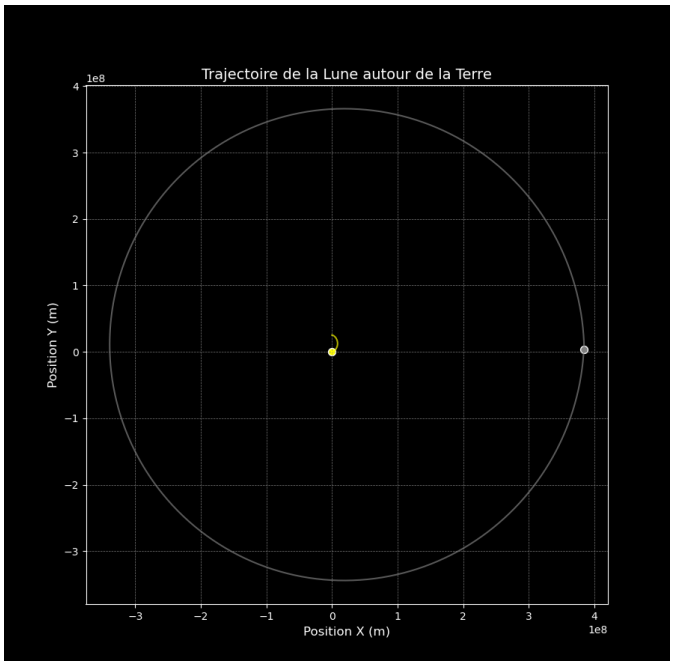

Figure 1: Trajectoire de la Lune autour de la Terre

On observe une trajectoire circulaire plutôt qu'elliptique pour la Lune autour de la Terre. Cela s'explique par le fait que, dans le système solaire, les forces gravitationnelles exercées par toutes les planètes agissent sur chacune d'elles. Ce que nous ne modélisons pas dans ce cas. Mais, la Lune fait bien un tour de la Terre en 27 jours. Tout

compte fait, ces résultats confirment la validité de nos méthodes et de nos codes. Nous pouvons ainsi envisager d'étudier l'entièreté du système solaire.

5.2 Système solaire

Maintenant que nos codes fonctionnent bien, nous pouvons résoudre le problème à N corps pour le système solaire centré autour du soleil. On trace les trajectoires sur 165 ans avec un pas de temps d'un jour. On obtient les trajectoires suivantes :

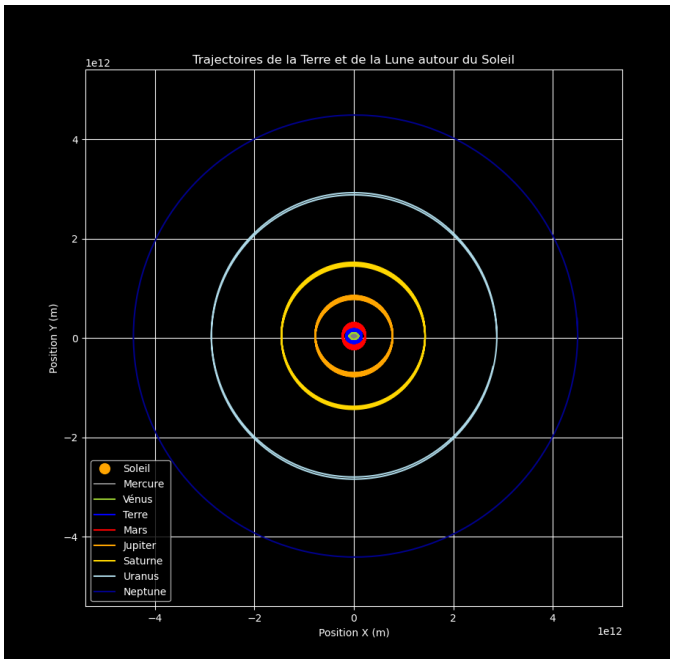

Figure 2: Système solaire

Pour étoffer l'étude,on peut aussi visualiser une animation en temps réelle des trajectoires des planètes autour du soleil. Pour cela, on mets en place un QR code permettant l'accès à la vidéo de l'animation. On fait l'animation sur un temps total de 165 ans pour visualiser toutes les trajectoires complètes.

Ce qr code permet de visualiser l'animation complète des trajectoires des planètes du système solaire.

6 Conclusion

A travers ce TP, nous avons pu mettre en place une résolution du problème à N corps. Notamment en utilisant une charnière en language C++ permettant de calculer la position des planètes à chaque instant. Puis nous avons traiter les données obtenues en python pour les exploiter.

La démarche de ce TP nous a permis de développer d'avantage nos connaissances en programmation sur C++ et python, mais aussi d'obtenir un résultat sur un problème concret.