

Résolution du problème des n corps en mécanique céleste

Dirk Stratmann
Yohan Duarte - François Puel

10 décembre 2018

Résumé

Dans le cadre du projet de l'unitée d'enseignement de physique numérique. De la description d'un système à N corps à sa résolution numérique dans le but de l'implémenter dans un jeu vidéo.

Table des matières

1	Introduction	3
2	Petit cours d'histoire	3
3	Partie Théorique	3
3.1	Equation du problème à n corps	3
4	Partie Codage	4
4.1	Modélisation à l'aide d'Heingen	4
4.2	Méthode de Runge-Kutta	5
4.3	Le Jeu	5
5	Discussions sur les limites du modèle	7

1 Introduction

Le but premier de ce projet est d'appliquer la loi de la gravitation de Newton à un système planétaire : le système solaire, pour le modéliser. Le code doit intégrer numériquement les équations du problème à partir des positions et des vitesses initiales pour fournir la simulation sur les différentes orbites la plus précise possible.

Le second objectif est de faire un jeu en utilisant le code du problème à n corps.

2 Petit cours d'histoire

Avant de nous pencher sur le problème à n corps intéressons nous à un cas particulier, le problème à 3 corps, qui a longtemps occupé les mathématiciens.

En 1765, Euler résout le cas particulier où les trois corps sont alignés. En 1772, Lagrange résout le cas où les trois corps (dont un de masse négligeable) sont aux sommets d'un triangle équilatéral. On parle de points de Lagrange, c'est à dire qu'il existe des points d'équilibre pour le petit corps où toutes les forces se compensent.

En 1890, Henri Poincaré publie un article "Sur le problème des trois corps et les équations de la dynamique" dans la revue *Acta Mathematica* ce qui lui vaudra le prix de roi Oscar.

En 1909, Sundman démontre qu'il existe une solution exacte du problème à trois corps dont voici le théorème : *"Si les constantes des aires dans le mouvement des trois corps par rapport à leur centre commun de gravité ne sont pas toutes nulles, on peut trouver une variable τ telle que les coordonnées des corps, leurs distances mutuelles et le temps soient développables en séries convergentes suivant les puissances de τ , qui représentent le mouvement pour toutes les valeurs réelles du temps, et cela quels que soient les chocs qui se produisent entre les corps."* Il publiera un résumé de ses travaux sur le sujet dans la revue *Acta Mathematica* en 1913.

En 1997 Moore, puis en 2001 avec Chenciner et Montgomery résolvent le cas d'un système à 3 corps de masse égale parcourant un 8.

Contrairement au problème à 3 corps, le problème à n corps ne possède pas de solution exacte connue (pour l'instant tout du moins). Pour le résoudre il faut donc utiliser des méthodes de résolutions approchées. Il en existe deux, la méthode des perturbations et l'analyse numérique qui va être l'objet de ce petit rapport.

3 Partie Théorique

3.1 Équation du problème à n corps

On considère un repère galiléen $R(O, x, y, z)$ avec n corps C_i tel quel $1 \leq i \leq n$.

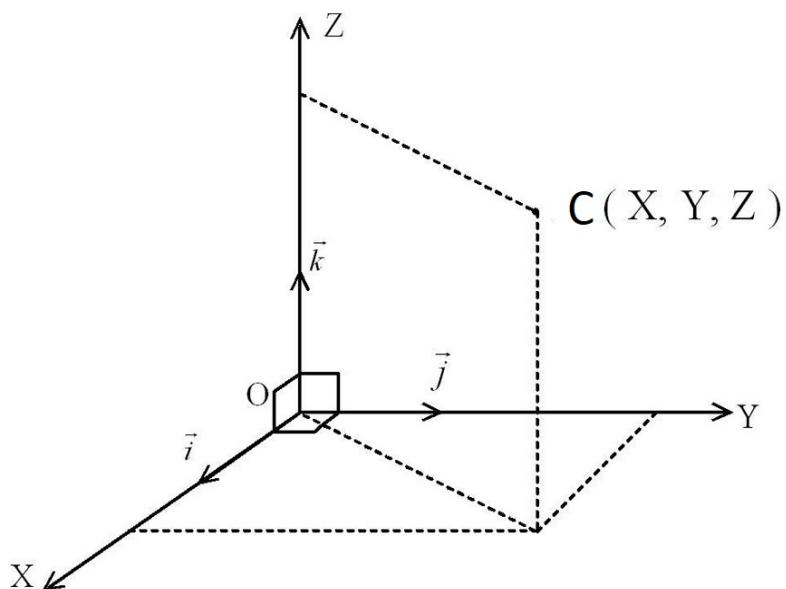


FIGURE 1 –

La force à laquelle est soumis un corps C_j et une somme des forces gravitationnelles exercées par tous les autres corps du système. Elle s'écrit comme :

$$\vec{F}_j = \sum_{i=1, i \neq j}^N Gm_i m_j \frac{\overrightarrow{C_i C_j}}{|C_i C_j|^3}$$

On introduit ensuite notre expression de la force dans l'équation de la deuxième loi de newton :

$$\sum \vec{F}_{ext} = m \vec{a} \quad (1)$$

$$\vec{F}_j = m_j \frac{d^2 \overrightarrow{OC_j}}{dt^2} \quad (2)$$

$$\sum_{i=1, i \neq j}^N Gm_i \frac{\overrightarrow{C_i C_j}}{|C_i C_j|^3} = \frac{d^2 \overrightarrow{OC_j}}{dt^2} \quad (3)$$

Nous avons donc une expression de l'accélération à chaque instant d'un corps C_j . En intégrant cette expression nous pouvons ainsi obtenir successivement la vitesse et la position du corps en question. C'est à cet instant que le programme entre en jeu.

4 Partie Codage

Les codes de cette partie sont dans le dossier "Headers".

On se place dans le repère du centre de masse du système. Notre code peut simuler un système à 3 dimension, mais notre but final étant de faire un jeu en 2D, nous allons donc nous limiter à 2 dimension.

4.1 Modélisation à l'aide d'Heingen

Notre système est modélisé comme une matrice $S(m,n)$ avec n le nombre de corps dans le système et m deux fois le nombre dimension spatiale :

$$S = \begin{bmatrix} X_1 & X_2 & \dots & X_n \\ Y_1 & Y_2 & \dots & Y_n \\ \vdots & \vdots & \dots & \vdots \\ V_{X1} & V_{X2} & \dots & V_{Xn} \\ V_{Y1} & V_{Y2} & \dots & V_{Yn} \\ \vdots & \vdots & \dots & \vdots \end{bmatrix}$$

et un vecteur des masses du système :

$$M = \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_n \end{bmatrix}$$

On peut donc définir dans le cadre de la résolution du système à n corps la dérivé de notre système par apport au temps, soit $\frac{dS}{dt}$. On a donc

$$\frac{dS}{dt} = \begin{bmatrix} V_{X1} & V_{X2} & \dots & V_{Xn} \\ V_{Y1} & V_{Y2} & \dots & V_{Yn} \\ \vdots & \vdots & \dots & \vdots \\ \vec{u}_x \cdot \vec{f}(S, M) & \vec{u}_x \cdot \vec{f}(S, M) & \dots & \vec{u}_x \cdot \vec{f}(S, M) \\ \vec{u}_y \cdot \vec{f}(S, M) & \vec{u}_y \cdot \vec{f}(S, M) & \dots & \vec{u}_y \cdot \vec{f}(S, M) \\ \vdots & \vdots & \dots & \vdots \end{bmatrix}$$

avec (cf l'équation (3)) :

$$\vec{f}(S, M) = \sum_{i=1, i \neq j}^N Gm_i \frac{\overrightarrow{C_i C_j}}{|C_i C_j|^3}$$

4.2 Méthode de Runge-Kutta

Nous l'avons implémenté avec Heingen

```
1  /* on importe nos Lib*/
2  #include <iostream>      //Permet de gérer les input output (io)
3  #include <fstream>       //Permet de gérer des fichier (f)
4  #include <cmath>          //Importe quelques fonctions mathématiques
5  #include <cstdlib>        //Permet d'utiliser la fonction rand()
6  #include "Eigen/Dense"    //Permet d'utiliser des matrices
7
8  using namespace std;
9  using namespace Eigen;
10
11 void Sum(MatrixXd& Som, MatrixXd& A, MatrixXd& B,double x)
12 {
13     /*Sum: Prend trois matrices en entrées et "retourne" La somme des deux dernières
14     avec le second pondéré par un scalaire x*/
15     Som = A+B*x;
16 } //Sum
17
18 void RK4(MatrixXd& S0, VectorXd& M, double dt,void F(MatrixXd& ,MatrixXd& , VectorXd&))
19 {
20     /*RK4: Prend en entrée la matrice du système, le vecteur des masses,
21     le pas de temps, et la fonction qui retourne la dérivée de ton système*/
22     double Dt = dt,t = 0;
23     int Nbc,Nbl, D;
24     Nbc = S0.cols();
25     Nbl = S0.rows();
26
27     MatrixXd dS0(Nbl,Nbc);
28     MatrixXd k1(Nbl,Nbc);
29     MatrixXd k2(Nbl,Nbc);
30     MatrixXd k3(Nbl,Nbc);
31     MatrixXd k4(Nbl,Nbc);
32     MatrixXd S(Nbl,Nbc);
33
34     // on fait un pas de temps adaptatif
35     F(dS0, S0, M);
36     D = rint(dS0.row(2).norm()+dS0.row(3).norm());
37     dt = Dt/(1+D);
38
39     while (t < Dt)
40     {
41         F(dS0, S0, M);
42         k1 = dS0*dt; //Calcul de k1
43         Sum(S, S0, k1,1/2);
44         F(dS0, S, M);
45         k2 = dS0*dt;//Calcul de k2
46         Sum(S, S0, k2,1/2);
47         F(dS0, S, M);
48         k3 = dS0*dt;//Calcul de k3
49         Sum(S, S0, k3,1/2);
50         F(dS0, S, M);
51         k4 = dS0*dt;//Calcul de k4
52
53         S0 += (k1+2*k2+2*k3+k4)/6;//Calcul de S0 au temps t+dt
54
55         t += dt;
56     } //while
57 } //RK4
```

FIGURE 2 – RK4.hpp

4.3 Le Jeu

Le jeu nommé 2001 A Space Odyssey ou 2001ASO (en référence à l'œuvre de Stanley Kubrick et d'Arthur C. Clarke). Le jeu utilise le code du problème à n corps pour mettre en scène le discovery one que l'on peut guider dans le système solaire en modifiant la vitesse de celui-ci en le propulsant. Toute la partie graphique est faite grâce à la librairie SFML

Les contrôles sont expliqués dans les crédits du jeu. Deux modes jouables sont proposée.



5 Discussions sur les limites du modèle

Tout les graphes, code et scripts gnuplot de cette partie sont dans le dossier "Test".

Nous considérons un système à deux corps , tout deux ayant la masse équivalentes a celle du soleil, avec des vitesses et des distances équivalentes au système Soleil-Jupiter. Nous regardons son évolution sur un intervalle de temps donné. (Nous avons choisi de telles conditions pour pousser à bout la simulation)

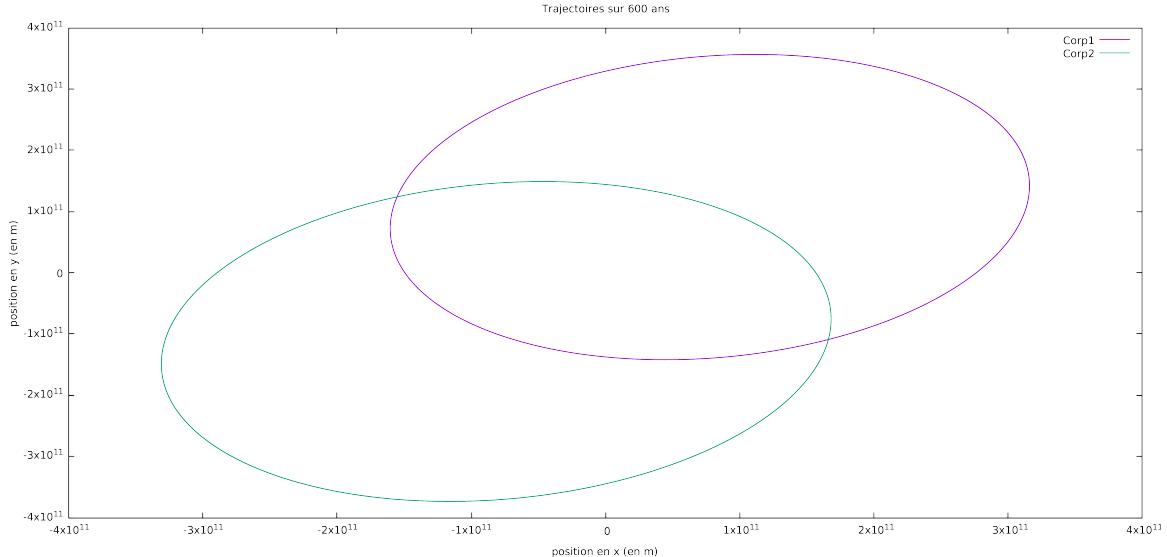


FIGURE 3 – Orbite des deux corps

La figure 4 nous montre l'évolution de l'énergie cinétique, l'énergie potentiel et de l'énergie totale dans le temps. En ordonnée nous avons l'énergie (Joule) et en abscisse le temps (ans).

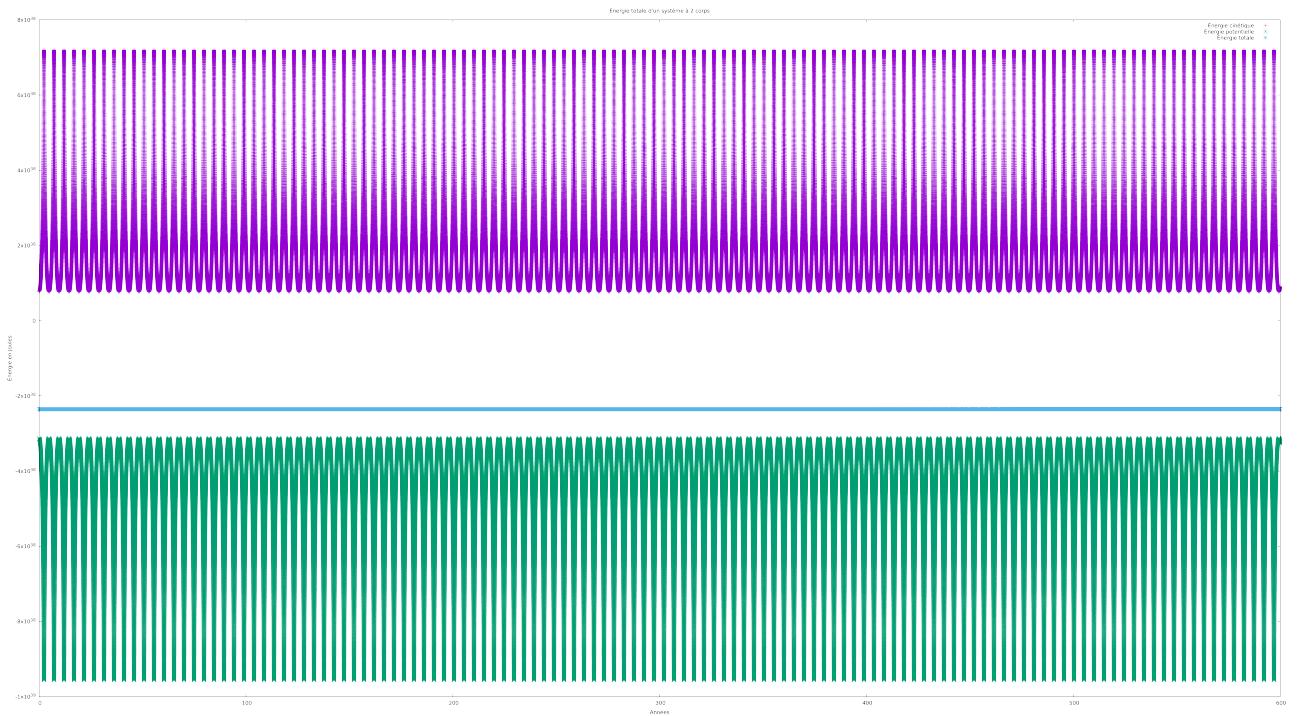


FIGURE 4 – Evolution de l'énergie cinétique (violet), potentielle (vert) et totale (bleu) en fonction du temps.

Jusqu'ici rien d'anormal mais si nous regardons plus en détail la dérive de l'énergie totale du système (en E_0 l'énergie à $t = 0$), en fonction du temps (ans) nous avons donc une énergie totale qui augmente .

En effet si l'on regarde plus près les trajectoires de deux corps nous pouvons voir que le système dérive petit à petit.

Nous avons donc un système qui commence à diverger au bout d'un siècle de $2 \times 10^{-5} E_0$. Notre modèle est donc précis jusqu'à un certain point. La cause première est, comme dit plus haut (partie 2), que la méthode utilisée pour résoudre ce système bien qu'efficace est une méthode de résolution approchée.

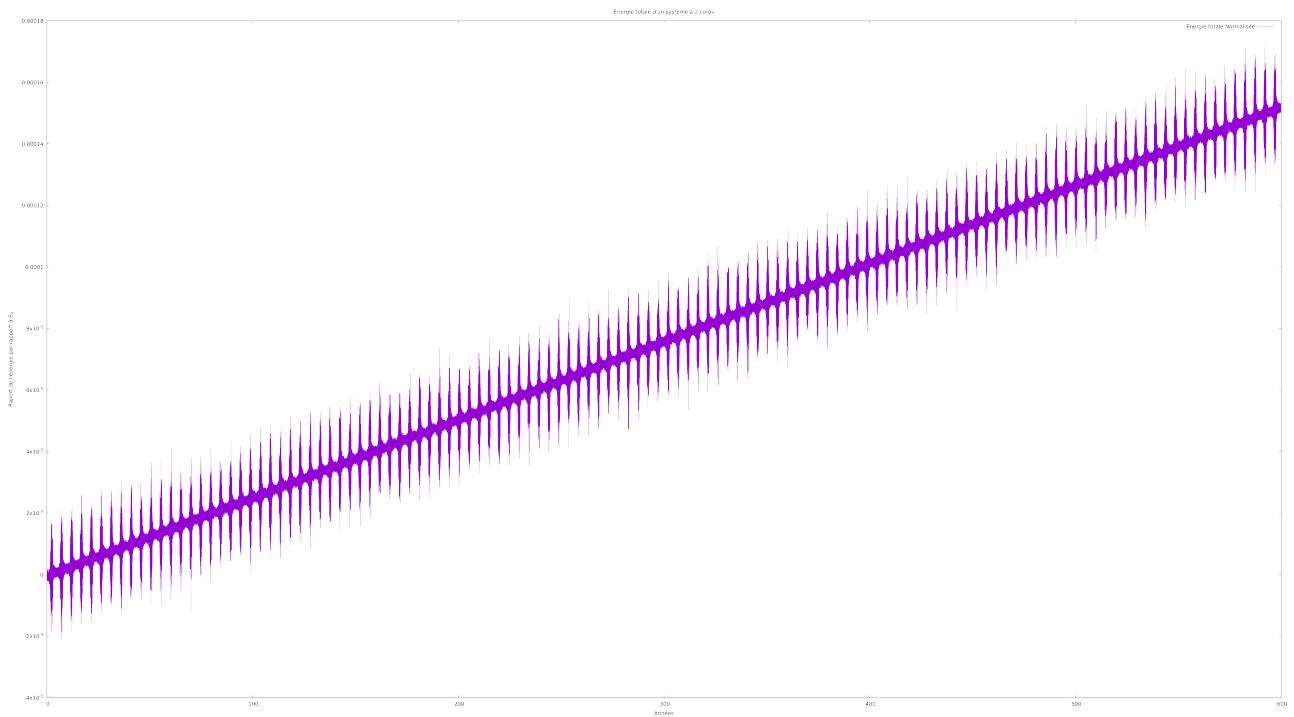


FIGURE 5 – Evolution de l'énergie totale en fonction du temps.

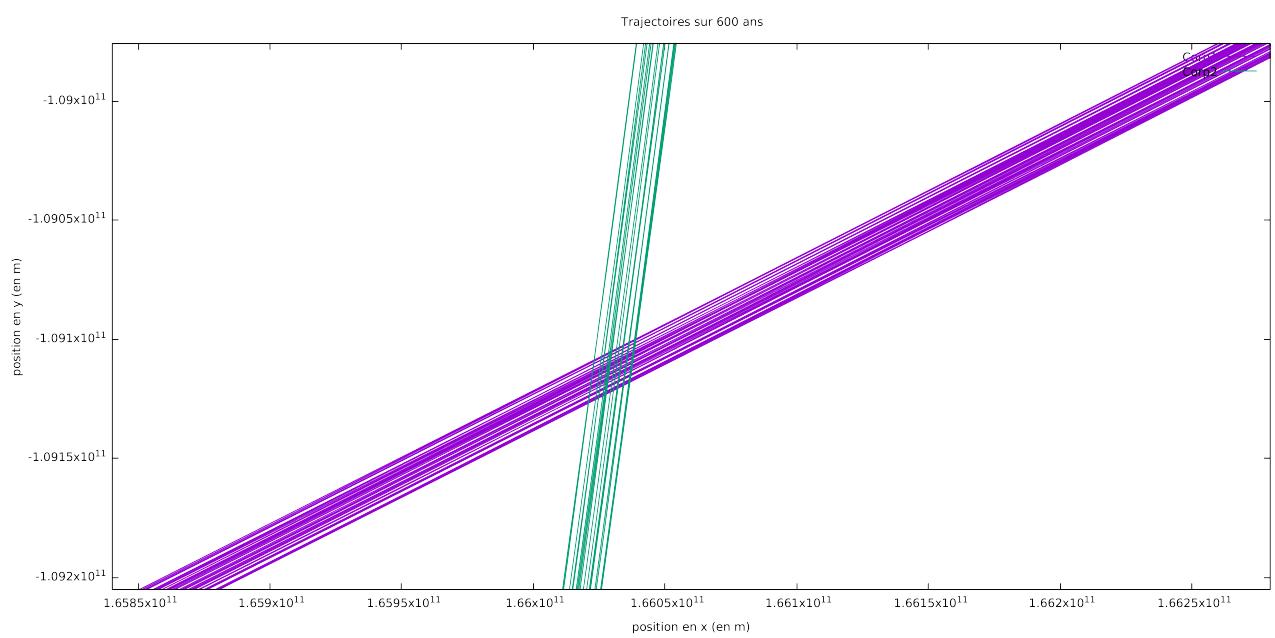


FIGURE 6 – Zoom sur la divergence des orbites des deux corps.