MASTER 1 PHYSIQUE
PARCOURS RECHERCHE FONDAMENTALE

Projet

Étude de la dynamique du problème à trois corps restreint

ROLLO Valentin
valentin.rollo@etu.univ-grenoble-alpes.fr

ZIMNIAK Nathan
nathan.zimniak@etu.univ-grenoble-alpes.fr

SYSTÈMES DYNAMIQUES, CHAOS ET APPLICATIONS

Janvier 2021

Table des matières

1 Introduction

L'objectif de ce projet est d'étudier le problème à trois corps restreint en s'appuyant sur sa modélisation numérique. Ce problème physique est une simplification du problème à trois corps général bien connu : une des trois masses est supposée négligeable par rapport aux deux autres. Ainsi, les deux corps les plus massifs sont en orbite circulaire autour de leur centre de masse. La modélisation numérique du problème à trois corps est incontournable puisque ce problème est non soluble analytiquement (en réalité une solution a été trouvée par Sundman en 1909 mais elle est inexploitable en pratique).

Il s'agit dans un premier temps de mettre en évidence les équations du mouvement relatives au corps le plus léger (aussi appelé "troisième corps" dans la suite) puis de réaliser l'étude dynamique du problème : mise en évidence des points fixes et de la zone accessible par le troisième corps. Le code est présenté en annexe. Le langage informatique utilisé est Python.

2 Mise en équation du système dynamique

Il s'agit ici d'introduire les grandeurs adimensionnées ainsi que les équations du mouvement utilisées dans le code.

La distance entre les deux corps les plus massifs est normalisée à l'unité. En introduisant le paramètre de masse ν, rapport entre l'une des deux masses et la somme des deux masses, il est possible de placer notre repère de façon à ce que la position des deux corps soit simplifiée :

$$\nu = \frac{m_2}{m_1 + m_2}$$

$$\begin{cases} (x_1, y_1) = (-\nu, 0) \\ (x_2, y_2) = (1 - \nu, 0) \end{cases}$$

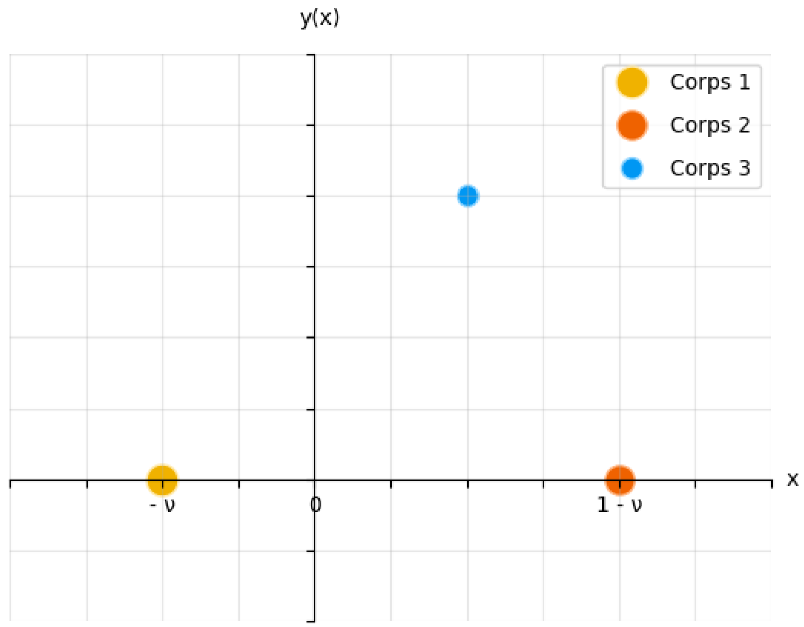

FIGURE 1 – Position des trois corps dans l'espace

Les distances entre les deux corps massifs et le troisième corps peuvent aussi être exprimées à partir du paramètre de masse :

$$\begin{cases} d_1 = \sqrt{(x+\nu)^2 + y^2} \\ d_2 = \sqrt{(x+\nu-1)^2 + y^2} \end{cases}$$

Remarque : Pour des raisons évidentes de clarté, les grandeurs relatives au troisième corps ne sont notées avec aucun indice.

La suite de l'étude est réalisée dans le référentiel tournant (x', y') :

$$\begin{cases} x'(t) = xcos(\omega t) - ysin(\omega t) \\ y'(t) = xsin(\omega t) + ycos(\omega t) \end{cases}$$

Les corps 1 et 2 sont fixes dans ce référentiel.

Dans le cas du problème à trois corps restreint, les grandeurs d'intérêt sont les grandeurs relatives au troisième corps :

Énergie cinétique $T = \frac{1}{2}\left[\dot{x}^2 + \dot{y}^2 + 2\omega(\dot{y}x - \dot{x}y) + \omega^2(x^2 - y^2)\right]$
Énergie potentielle $V = -Gm\left(\frac{m_1}{d_1} + \frac{m_2}{d_2}\right)$
Lagrangien $\mathcal{L} = \frac{1}{2}\left[\dot{x}^2 + \dot{y}^2 + 2\omega(\dot{y}x - \dot{x}y) + \omega^2(x^2 - y^2)\right] + Gm\left(\frac{m_1}{d_1} + \frac{m_2}{d_2}\right)$

Hamiltonien $\mathcal{H} = \dfrac{p_x^2 + p_y^2}{2m} + \omega(p_x y - p_y x) - \omega^2 m \left(\dfrac{1-\nu}{d_1} + \dfrac{\nu}{d_2}\right)$

Remarque : La constante gravitationnelle a pu être remplacée par la vitesse de rotation du système à partir de la trosième loi de Kepler : $\omega^2 = G(m_1 + m_2)$

L'Hamiltonien obtenu permet d'obtenir les équations du mouvement du troisième corps. La mécanique Hamiltonienne donne les équations canoniques de Hamilton :

$$\begin{cases} \dfrac{dx}{dt} = \dfrac{\partial \mathcal{H}}{\partial p_x} \\ \dfrac{dy}{dt} = \dfrac{\partial \mathcal{H}}{\partial p_y} \\ \dfrac{dp_x}{dt} = -\dfrac{\partial \mathcal{H}}{\partial x} \\ \dfrac{dp_y}{dt} = -\dfrac{\partial \mathcal{H}}{\partial y} \end{cases}$$

Après dérivation, en travaillant avec les vitesses plutôt qu'avec les impulsions,

$$\boxed{\begin{cases} \dfrac{dx}{dt} = v_x + \omega y \\ \dfrac{dy}{dt} = v_y - \omega x \\ \dfrac{dv_x}{dt} = \omega v_y - \omega^2 \left[\dfrac{(1-\nu)(x+\nu)}{d_1^3} + \dfrac{\nu(x+\nu-1)}{d_2^3}\right] \\ \dfrac{dv_y}{dt} = -\omega v_x - \omega^2 \left[\dfrac{(1-\nu)y}{d_1^3} + \dfrac{\nu y}{d_2^3}\right] \end{cases}}$$

Ce sont ces équations qui, une fois intégrées, permettent d'obtenir la position et la vitesse du corps à chaque instant.

3 Étude du système dynamique

3.1 Points de Lagrange

Le système possède plusieurs points fixes appelés points de Lagrange. Ce sont les points pour lesquels la vitesse et l'accéleration du corps est nulle :

$$
\begin{cases}
L_1 : (x_{L_1}, y_{L_1}) = (1 - \left(\frac{1}{3}\nu\right)^{\frac{1}{3}}, 0) \\
L_2 : (x_{L_2}, y_{L_2}) = (1 + \left(\frac{1}{3}\nu\right)^{\frac{1}{3}}, 0) \\
L_3 : (x_{L_3}, y_{L_3}) = (-1 - \frac{5}{12}\nu, 0) \\
L_4 : (x_{L_4}, y_{L_4}) = (\frac{1}{2} - \nu, \frac{\sqrt{3}}{2}) \\
L_5 : (x_{L_5}, y_{L_5}) = (\frac{1}{2} - \nu, -\frac{\sqrt{3}}{2})
\end{cases}
$$

Remarque : Le calcul de ces points est en annexe.

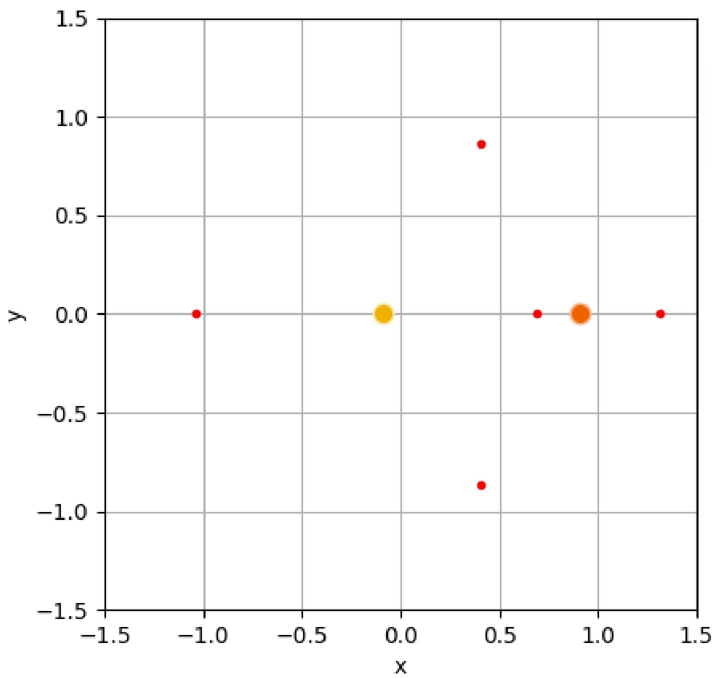

FIGURE 2 – Position des points de Lagrange dans l'espace

Il est maintenant possible de vérifier numériquement si ces points sont des points fixes : il faut placer le corps sur chacun des points de Lagrange avec comme vitesses initiales $v_x = -y$ et $v_y = x$ et observer son comportement. Si le corps reste immobile alors la position initiale est bien un point fixe.

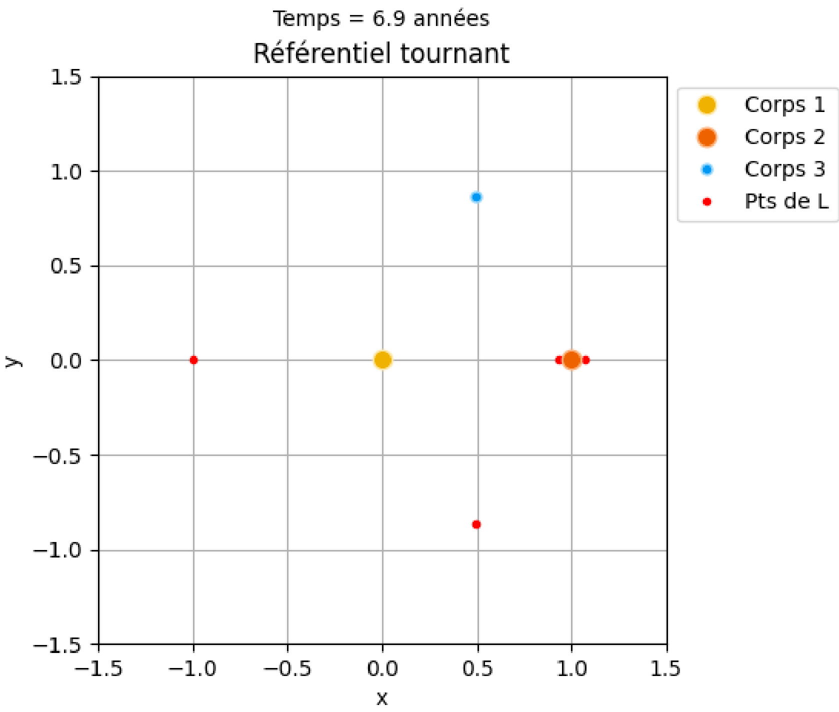

FIGURE 3 – Test du point fixe L_4

Le corps reste sur les points de Lagrange (ici L_4) après un certain temps : ce sont bien des points fixes.

Il est important de noter que nos points fixes ne sont pas calculés numériquement mais sont calculés avec les expressions établies précédemment, c'est-à-dire dans l'hypothèse d'un paramètre de masse très petit pour les trois premiers points de Lagrange. Ainsi, lorsque le paramètre de masse est assez grand ($\simeq 0,3$), le corps ne reste pas sur ces trois points. Lorsque ce paramètre de masse est assez petit ($\simeq 0,0001$), le corps reste sur les deux premiers points au moins pendant un an et reste indéfiniment sur le troisième point. Pour se convaincre que ces points sont bien des points fixes, il est possible de faire varier le paramètre de masse et d'étudier l'écart relatif entre la position initiale du corps et sa position finale. Comme le montre le figure 4, le troisième point de Lagrange est bien un point fixe pour des valeurs du paramètre de masse inférieures à 0,04.

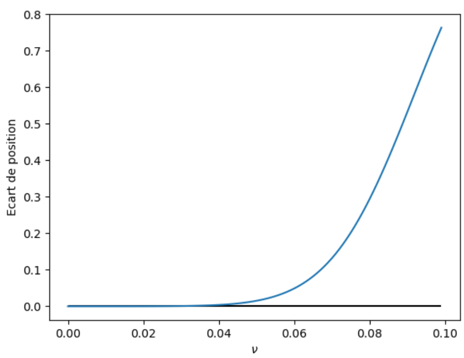

FIGURE 4 – Etude de la nature de L3 en fonction du paramètre de masse

Il reste maintenant à déterminer la stabilité de ces points. Il faut placer le corps sur le point de Lagrange, perturber la vitesse initiale (ici de 10^{-2}), et étudier la variation de la position du corps par rapport à sa position initiale. Le paramètre de masse choisi est très faible de façon à avoir la meilleur approximation possible sur les points de Lagrange. Pour les points L_4 et L_5, il y a une très faible variation de position : ces points fixes sont alors stables. Pour les points L_1, L_2 et L_3 il y a une très grande variation de position (de l'ordre de 100 %) : ces points fixes sont donc instables. Il est aussi possible de remarquer que le premier point de Lagrange est beaucoup plus instable que les deux autres car l'écart de position relatif diverge plus rapidement.

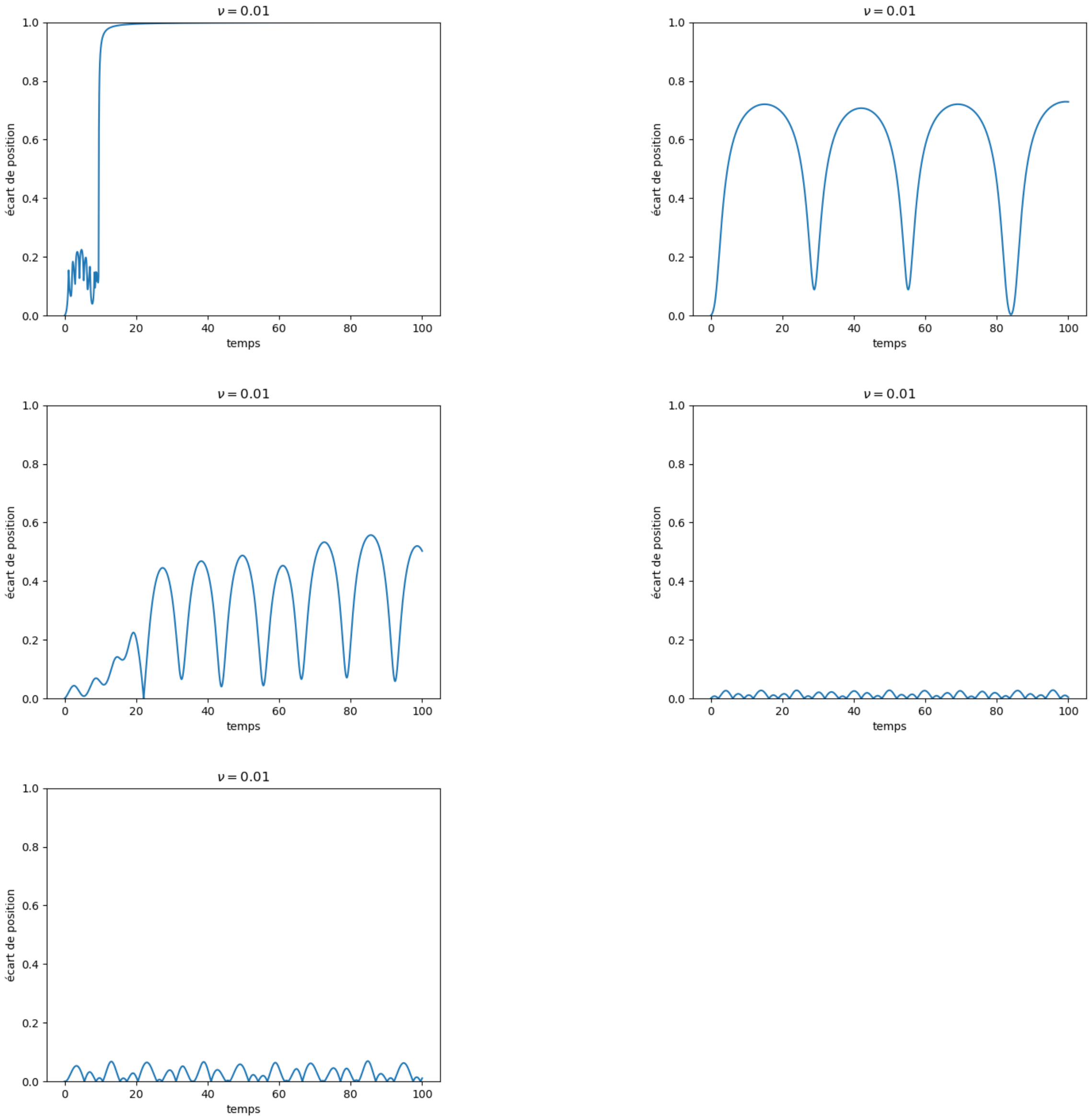

FIGURE 5 – Variation de la position du 3ème corps aux points L_1, L_2, L_3, L_4, L_5

3.2 Zones accessibles

Certaines zones sont accessibles et d'autres sont inaccessibles par le troisième corps. Les zones accessibles correspondent aux points pour lesquels son énergie cinétique est positive :

$$\mathcal{A} = \{(x, y) | T = E - V(x, y) \geq 0\}$$

Le système étant conservatif, l'énergie est égale à l'Hamiltonien :

$$\begin{aligned} E &= \mathcal{H} \\ &= \frac{p_x^2 + p_y^2}{2m} + \omega(p_x y - p_y x) - \omega^2 m \left(\frac{1-\nu}{d_1} + \frac{\nu}{d_2} \right) \end{aligned}$$

Il est alors possible d'identifier l'énergie cinétique et potentielle du système (en normalisant la masse et la vitesse de rotation) :

$$\begin{cases} T = \dfrac{1}{2} \left[(p_x + y)^2 + (p_y - x)^2 \right] \\ V(x, y) = -\dfrac{1}{2}(x^2 + y^2) - \left(\dfrac{1-\nu}{d_1} + \dfrac{\nu}{d_2} \right) \end{cases}$$

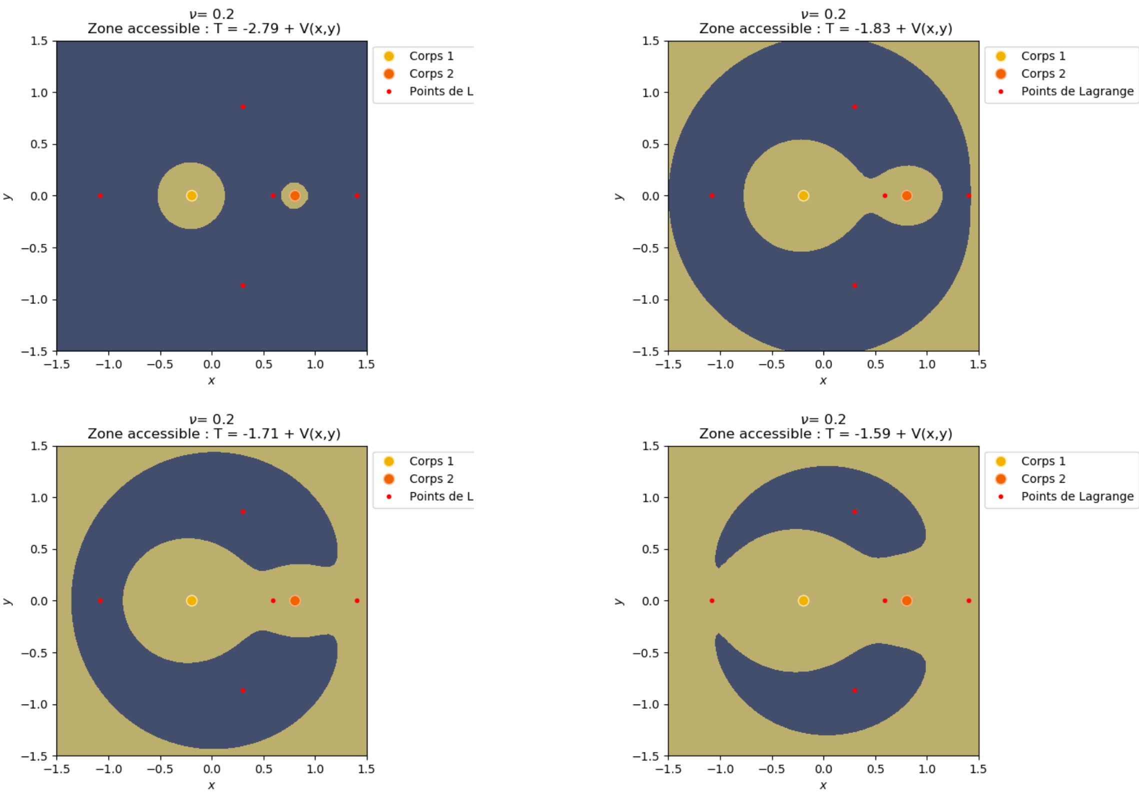

FIGURE 6 – Zone accessible (zone claire) par le troisième corps pour $\nu = 0.2$

Plus l'énergie cinétique augmente et plus il y a de zones accessibles. Les bifurcations, c'est-à-dire les endroits pour lesquels les zones inaccessibles se brisent, se passent aux points fixes du système.

Comme précédemment, ces remarques ne sont valables que pour de petites valeurs du paramètre de masse. Pour un paramètre de masse trop grand (0.2 par exemple), les bifurcations se passe bien aux points L_3, L_4 et L_5, semblent aussi se passer en L_2 mais ne se passe pas exactement en L_1. Ainsi, en prenant une valeur assez faible du paramètre de masse, l'erreur sur la bifurcation en L_1 est quasiment corrigée comme le montre la figure 7 :

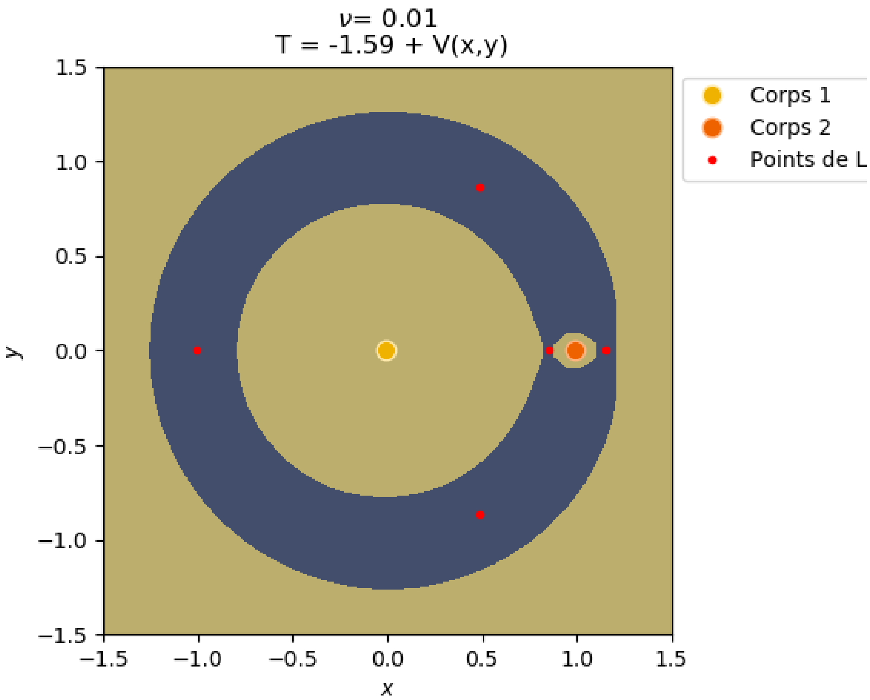

FIGURE 7 – Bifurcation au point L_1 pour $\nu = 0.01$

4 Conclusion

A l'aide des connaissances acquises en programmation il a donc été possible d'étudier un problème physique non soluble analytiquement. Nous avons pu mettre en évidence la trajectoire d'un corps soumis à l'attraction gravitationnelle de deux autres corps, ainsi que différents points (les points de Lagrange) où ce 3ème corps, de masse négligeable par rapport au deux autres, reste immobile. De plus, l'analyse de la zone accessible par le 3ème corps ainsi que l'étude des points fixes a permis de mettre en évidence la limite de l'hypothèse faite sur le paramètre de masse ν.

Le problème à trois corps est un problème physique qui se prête très bien à la programmation scientifique. Problème incontournable, il nous a permis d'approfondir nos connaissances aussi bien en Physique (astronomie, mécanique Hamiltonienne etc.) qu'en programmation (méthodes d'intégration, animation d'une figure etc.). Ces connaissances sont d'une importance primordiale car le monde de la recherche s'appuie quotidiennement sur les simulations numériques.

Il aurait sûrement été possible d'améliorer le code par ajout d'une interface graphique qui permettrait à l'utilisateur d'interagir avec le programme (pour changer le référentiel et les conditions initiales par exemple). La fonction ODEint (bibliothèque Scipy.integrate) aurait aussi pu être utilisée pour intégrer les fonctions mais elle ne nous aurait rien appris sur la méthode d'intégration car elle agit comme une "boîte noire" : il nous semblait plus important d'utiliser la méthode RK4 afin de mieux comprendre le fonctionnement de la résolution numérique d'équations différentielles. Enfin, une analyse plus complète aurait demandé le calcul numérique des points de Lagrange et la représentation des sections de Poincaré pour caractériser plus en détail le chaos du système.

5 Bibliographie

F. Faure - Le problème à trois corps restreint, 2018. https ://www-fourier.ujf-grenoble.fr/
S. Rondi - Le problème à trois corps, 2002. http ://www.astrosurf.com/
J. Roussel - Méthodes de Runge-Kutta, 2016. https ://femto-physique.fr/

6 Annexes

6.1 Calcul des points fixes

Le système possède plusieurs points fixes appelés points de Lagrange. Ce sont les points pour lesquels la vitesse et l'accéleration du corps est nulle :

$$\begin{cases} \left.\dfrac{dx}{dt}\right|_{x^*,y^*} = 0 \\ \left.\dfrac{dy}{dt}\right|_{x^*,y^*} = 0 \\ \left.\dfrac{dv_x}{dt}\right|_{x^*,y^*} = 0 \\ \left.\dfrac{dv_y}{dt}\right|_{x^*,y^*} = 0 \end{cases}$$

Ce système d'équations permet l'obtention de deux équations sur les points fixes :

$$\begin{cases} x^* = \dfrac{(1-\nu)(x^*+\nu)}{d_1^3} + \dfrac{\nu(x^*+\nu-1)}{d_2^3} \\ y^* = \dfrac{(1-\nu)y^*}{d_1^3} + \dfrac{\nu y^*}{d_2^3} \end{cases}$$

Il y a trois solutions dans le cas où l'ordonnée est nulle : elles correspondent aux trois premiers points de Lagrange :

$$\begin{cases} x^* = \dfrac{(1-\nu)(x^*+\nu)}{|x^*+\nu|^3} + \dfrac{\nu(x^*+\nu-1)}{|x^*+\nu-1|^3} \\ y^* = \dfrac{(1-\nu)y^*}{d_1^3} + \dfrac{\nu y^*}{d_2^3} \end{cases}$$

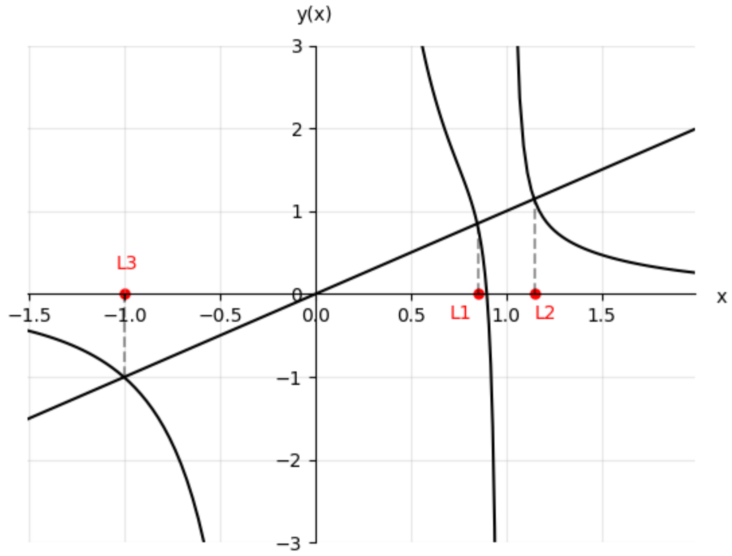

FIGURE 8 – Résolution graphique des trois premiers points de Lagrange, cas où $\nu = 0.001$

Il est possible de trouver analytiquement les coordonnées des ces points en supposant que le paramètre de masse très petit :

Pour L1,

$$\begin{cases} x_{L_1} \in \quad]-\nu, \quad 1-\nu[\\ \nu << 1 \\ y_{L_1} = 0 \end{cases}$$

$$\Leftrightarrow \begin{cases} x_{L_1} = \dfrac{(1-\nu)}{(x_{L_1}+\nu)^2} - \dfrac{\nu}{(x_{L_1}+\nu-1)^2} \\ x_{L_1} = 1+\epsilon \quad , \quad \epsilon << 1 \\ y_{L_1} = 0 \end{cases}$$

$$\Leftrightarrow \boxed{\begin{cases} x_{L_1} = 1 - \left(\dfrac{1}{3}\nu\right)^{\frac{1}{3}} + o(\nu^2) \\ y_{L_1} = 0 \end{cases}}$$

Pour L2,

$$\begin{cases} x_{L_2} > 1-\nu \\ \nu << 1 \\ y_{L_2} = 0 \end{cases}$$

$$\Leftrightarrow \begin{cases} x_{L_2} = \dfrac{(1-\nu)}{(x_{L_2}+\nu)^2} + \dfrac{\nu}{(x_{L_2}+\nu-1)^2} \\ x_{L_2} = 1+\epsilon \quad , \quad \epsilon << 1 \\ y_{L_2} = 0 \end{cases}$$

$$\Leftrightarrow \boxed{\begin{cases} x_{L_2} = 1 + \left(\dfrac{1}{3}\nu\right)^{\frac{1}{3}} + o(\nu^2) \\ y_{L_2} = 0 \end{cases}}$$

Pour L3,

$$\begin{cases} x_{L_2} < -\nu \\ \nu << 1 \\ y_{L_3} = 0 \end{cases}$$

$$\Leftrightarrow \begin{cases} x_{L_3} = -\dfrac{(1-\nu)}{(x_{L_3}+\nu)^2} - \dfrac{\nu}{(x_{L_3}+\nu-1)^2} \\ x_{L_3} = -1+\epsilon \quad , \quad \epsilon << 1 \\ y_{L_3} = 0 \end{cases}$$

$$\Leftrightarrow \boxed{\begin{cases} x_{L_3} = -1 - \dfrac{5}{12}\nu + o(\nu^2) \\ y_{L_3} = 0 \end{cases}}$$

Il y a deux solutions dans le cas où l'ordonnée est non-nulle : elles correspondent aux deux autres points de Lagrange :

$$\begin{cases} x^* = \dfrac{(1-\nu)(x^*+\nu)}{d_1^3} + \dfrac{\nu(x^*+\nu-1)}{d_2^3} \\ 1 = \dfrac{(1-\nu)}{d_1^3} + \dfrac{\nu}{d_2^3} \end{cases}$$

$$\Leftrightarrow \begin{cases} x^* = \left(1 - \dfrac{\nu}{d_2^3}\right)(x^*+\nu) + \dfrac{\nu(x^*+\nu-1)}{d_2^3} \\ \dfrac{(1-\nu)}{d_1^3} = 1 - \dfrac{\nu}{d_2^3} \end{cases}$$

$$\Leftrightarrow \begin{cases} (x^*+\nu-1)^2 + {y^*}^2 = 1 \\ (x^*+\nu)^2 + {y^*}^2 = 1 \end{cases}$$

Pour L4,

$$\boxed{\begin{cases} x_{L_4} = \dfrac{1}{2} - \nu \\ y_{L_4} = \dfrac{\sqrt{3}}{2} \end{cases}}$$

Pour L5,

$$\boxed{\begin{cases} x_{L_5} = \dfrac{1}{2} - \nu \\ y_{L_5} = -\dfrac{\sqrt{3}}{2} \end{cases}}$$

6.2 Code

```python
import numpy as np
import matplotlib.pyplot as plt
from matplotlib import animation





## RÉSOLUTION DU PROBLÈME À TROIS CORPS RESTREINT PAR MÉTHODE RK4





## Choix du référentiel (tournant par défaut, galiléen si "G")
```

```python

referentiel = "G"


## Constantes du système

m1, m2 = 2e30, 2e27     # Masse des corps du système (kg)
v = m2/(m1+m2)          # Paramètre de masse
w = 1                   # Vitesse de rotation

## Calcul de la position des deux corps les plus massifs

P1 = [-v, 0]
P2 = [1-v, 0]


## Calcul de la position des points de Lagrange

L1 = [(1-(v/3)**(1/3)), 0]
L2 = [(1+(v/3)**(1/3)), 0]
L3 = [(-1-5*v/12),0]
L4 = [(1/2-v), np.sqrt(3)/2]
L5 = [(1/2-v), -np.sqrt(3)/2]





## Fonction qui retourne un tableau avec les fonctions à intégrer

def TroisCorps(Y, t):
    # Attribution d'une colonne du tableau Y à chaque variable du système
    x, y, vx, vy = Y
    # Calcul des distances entre le troisième corps et les deux autres corps
    d1 = np.sqrt((x+v)**2 + y**2)
    d2 = np.sqrt((x+v-1)**2 + y**2)
    # Fonctions à intégrer
    dxdt = vx + w*y
    dydt = vy - w*x
    dvxdt = w*vy - (w**2)*((1-v)*(x+v)/d1**3 + v*(x+v-1)/d2**3)
    dvydt = -w*vx - (w**2)*((1-v)*(y)/d1**3 + v*(y)/d2**3)
    return np.array([dxdt, dydt, dvxdt, dvydt])





## Conditions initiales du système
```

```
ti = 0                        # Temps initial (années)
tf = 10                       # Temps final (années)
n = 250                       # Nombre de pas
dt = (tf-ti)/n                # Temps d'un pas
T = np.linspace(ti, tf, n)  # Création de la liste des valeurs de temps

x0, y0 = L4                   # Positions initiales
vx0, vy0 = -y0, x0            # Vitesses initiales
Y0 = (x0, y0, vx0, vy0)       # Création de la liste des positions et des vitesses
                              # initiales




## Calcul des positions et des vitesses par méthode RK4

def RK4(f, Y0, T):
    # Création du tableau contenant les positions et les vitesses
    Y = np.zeros([len(T),len(Y0)])
    # Initialisation des variables de position et de vitesse
    Y[0,:] = Y0
    # Calcul des nouvelles valeurs de positions et de vitesses à chaque temps
    for k in range(1, len(T)):
        # Calcul des fonctions de la méthode RK4
        k1 = dt * f(Y[k-1], T[k-1])
        k2 = dt * f(Y[k-1]+0.5*k1, T[k-1] + dt/2)
        k3 = dt * f(Y[k-1]+0.5*k2, T[k-1] + dt/2)
        k4 = dt * f(Y[k-1]+k3, T[k-1] + dt)
        # Ajout des nouvelles valeurs de positions et de vitesses à la liste
        Y[k,:] = Y[k-1] + (k1+2*k2+2*k3+k4)/6
    return Y

Y = RK4(TroisCorps, Y0, T)  # Création du tableau des solutions
x, y, vx, vy = Y.T            # Attribution d'une ligne de la transposée du tableau Y
                              # à chaque variable du système pour la lecture des
                              # solutions





## Animation 2D

fig = plt.figure()
ax = plt.axes()
```

```python
# Fonction qui retourne la position à chaque frame

def Animation(i):
    t = "Temps = " + str(round(T[i],1)) + " années"
    temps.set_text(t)
    if referentiel == "G":
        # Position des corps
        ligne1.set_data(P1[0]*np.cos(T[i]), P1[0]*np.sin(T[i]))
        ligne2.set_data(P2[0]*np.cos(T[i]), P2[0]*np.sin(T[i]))
        ligne3.set_data(x[i]*np.cos(T[i]) - y[i]*np.sin(T[i]),
                        x[i]*np.sin(T[i]) + y[i]*np.cos(T[i]))
        # Position des points de Lagrange
        ligneL1.set_data(L1[0]*np.cos(T[i]), L1[0]*np.sin(T[i]))
        ligneL2.set_data(L2[0]*np.cos(T[i]), L2[0]*np.sin(T[i]))
        ligneL3.set_data(L3[0]*np.cos(T[i]), L3[0]*np.sin(T[i]))
        ligneL4.set_data(L4[0]*np.cos(T[i]) - L4[1]*np.sin(T[i]),
                         L4[0]*np.sin(T[i]) + L4[1]*np.cos(T[i]))
        ligneL5.set_data(L5[0]*np.cos(T[i]) - L5[1]*np.sin(T[i]),
                         L5[0]*np.sin(T[i]) + L5[1]*np.cos(T[i]))
        return (ligne1, ligne2, ligne3, ligneL1, ligneL2, ligneL3, ligneL4, ligneL5)
    else:
        # Position du troisième corps
        ligne3.set_data(x[i], y[i])
        return ligne3


# Plot la position

if referentiel == "G":
    titre = "Référentiel Galiléen"
    # Position des corps
    ligne1, = ax.plot([], [], "o", label = "Corps 1", color = "#FFEAAE",
                      markersize = 10, markevery = 10000,
                      markerfacecolor = "#F0B200", zorder = 3)
    ligne2, = ax.plot([], [], "o", label = "Corps 2", color = "#FFB581",
                      markersize = 8, markevery = 10000,
                      markerfacecolor = "#EF6200", zorder = 3)
    ligne3, = ax.plot([], [], "o", label = "Corps 3", color = "#96D7FF",
                      markersize = 5, markevery = 10000,
                      markerfacecolor = "#0097F3", zorder = 3)
    # Position des points de Lagrange
    ligneL1, = ax.plot([], [], "o", label = "Pts de L", color = "#FF0000",
                       markersize = 3, markevery = 10000,
                       markerfacecolor = "#FF0000", zorder = 2)
    ligneL2, = ax.plot([], [], "o", color = "#FF0000", markersize = 3,
                       markevery = 10000, markerfacecolor = "#FF0000", zorder = 2)
    ligneL3, = ax.plot([], [], "o", color = "#FF0000", markersize = 3,
                       markevery = 10000, markerfacecolor = "#FF0000", zorder = 2)
```

```
    ligneL4, = ax.plot([], [], "o", color = "#FF0000", markersize = 3,
                        markevery = 10000, markerfacecolor = "#FF0000", zorder = 2)
    ligneL5, = ax.plot([], [], "o", color = "#FF0000", markersize = 3,
                        markevery = 10000, markerfacecolor = "#FF0000", zorder = 2)
else:
    titre = "Référentiel tournant"
    # Position des corps
    ax.plot(P1[0], 0, "o", label = "Corps 1", markersize = 9,
            markerfacecolor = "#F0B200", markeredgecolor = "#FFEAAE", zorder = 3)
    ax.plot(P2[0], 0, "o", label = "Corps 2", markersize = 9,
            markerfacecolor = "#EF6200", markeredgecolor = "#FFB581", zorder = 3)
    ligne3, = ax.plot([], [], "o", label = "Corps 3", color = "#96D7FF",
              markersize = 5, markevery = 10000, markerfacecolor = "#0097F3",
              zorder = 3)
    # Position des points de Lagrange
    ax.plot(L1[0], 0, "o", label = "Pts de L", markersize = 3,
            markerfacecolor = "#FF0000", markeredgecolor = "#FF0000", zorder = 2)
    ax.plot(L2[0], 0, "o", markersize = 3, markerfacecolor = "#FF0000",
            markeredgecolor = "#FF0000", zorder = 2)
    ax.plot(L3[0], 0, "o", markersize = 3, markerfacecolor = "#FF0000",
            markeredgecolor = "#FF0000", zorder = 2)
    ax.plot(L4[0], L4[1], "o", markersize = 3, markerfacecolor = "#FF0000",
            markeredgecolor = "#FF0000", zorder = 2)
    ax.plot(L5[0], L5[1], "o", markersize = 3, markerfacecolor = "#FF0000",
            markeredgecolor = "#FF0000", zorder = 2)


# Plot le temps passé

temps = ax.text(0, 1.8, "", horizontalalignment = "center",
                verticalalignment = "center")


# Fonction d'animation

anim = animation.FuncAnimation(fig, Animation, frames = len(T), interval = 30,
                               blit = False)


# Paramètre du graphe

ax.axis("square")
ax.set_xlim(-1.5, 1.5)
ax.set_ylim(-1.5, 1.5)
plt.xlabel("x")
plt.ylabel("y")

plt.title(titre)
```

```
plt.legend(bbox_to_anchor = (1, 1), loc = "upper left")
plt.grid()

plt.show()




# Définition de la fonction correspondant à la zone accessible

def ZA(E):
    x = np.linspace(-1.5, 1.5, 50)
    y = np.linspace(-1.5, 1.5, 50)[:,np.newaxis]
    # Calcul des distances entre le troisième corps et les deux autres corps
    d1 = np.sqrt((x+v)**2 + y**2)
    d2 = np.sqrt((x+v-1)**2 + y**2)
    # Calcul de l'énergie cinétique du troisième corps
    T = E + 0.5*(x**2 + y**2) + ((1-v)/d1) + v/d2
    # Création des lignes de niveaux
    ldn = np.linspace(-200, 200, 3)
    plt.contourf(T, ldn, extent = (-1.5, 1.5, -1.5, 1.5), cmap = "cividis")
    #Affichage des corps et points de Lagrange
    plt.plot(P1[0], P1[1],'o', label = "Corps 1", markersize = 9,
             markerfacecolor = "#F0B200", markeredgecolor = "#FFEAAE")
    plt.plot(P2[0], P2[1],'o', label = "Corps 2", markersize = 9,
             markerfacecolor = "#EF6200", markeredgecolor = "#FFB581")
    plt.plot(L1[0], L1[1],'o', label = "Points de Lagrange", markersize = 3,
             markerfacecolor = "#FF0000", markeredgecolor = "#FF0000")
    plt.plot(L2[0], L2[1],'o', markersize = 3, markerfacecolor = "#FF0000",
             markeredgecolor = "#FF0000")
    plt.plot(L3[0], L3[1],'o', markersize = 3, markerfacecolor = "#FF0000",
             markeredgecolor = "#FF0000")
    plt.plot(L4[0], L4[1],'o', markersize = 3, markerfacecolor = "#FF0000",
             markeredgecolor = "#FF0000")
    plt.plot(L5[0], L5[1],'o', markersize = 3, markerfacecolor = "#FF0000",
             markeredgecolor = "#FF0000")
    plt.legend(bbox_to_anchor = (1, 1), loc = "upper left")
    # Paramètres du graphe
    plt.axis("square")
    plt.title(rf'$\nu$= {v}' + f"\n T = {E:4.2f} + V(x,y)")
    plt.xlabel('$x$')
    plt.ylabel('$y$')





def anim_ZA():
```

```
        E = np.arange(-3, 1, 0.03)  # Choix arbitraire d'une plage d'énergies
        for Energie in E:
                plt.clf()
                ZA(Energie)
                plt.pause(0.03)

anim_ZA()
```