# Rapport ON5 : Cartographie du Chaos déterministe dans un système mécanique multipolaire

NGUYEN Jacques | 22107195
RODAT Gautier | 22105257
Université Toulouse III - Paul Sabatier

Mots clés : Chaos, cartographie, potentiel, N corps

**Date : Mai 2024**

# Table des matières

# 1 Introduction

Dans le cadre de l'UE Projet numérique autour de la physique, nous étudions et réalisons la cartographie du chaos déterministe.
Certains systèmes mécaniques présentent une très grande sensibilité aux conditions initiales, on parle alors de chaos déterministe. Il est possible de cartographier, pour un système donné, les zones de stabilité ou l'état final du système est peu sensible aux conditions initiales et les zones chaotiques où il y a une extrême sensibilité aux conditions initiales.
La cartographie, nous permet de visualiser et d'analyser les comportements imprévisibles et complexes des systèmes déterministes. Ce rapport se concentre sur la mise en œuvre de différentes méthodes de résolution numérique permettant d'obtenir une cartographie du chaos déterministe.
Notre objectif est donc de réaliser une cartographie du chaos déterministe dans un système mécanique constitué de plusieurs pôles attractifs ainsi que d'interpréter les résultats obtenus pour mieux comprendre ses systèmes chaotiques.

Nous pourrons assimiler cette cartographie dans un cas concret dans la suite de ce rapport c'est-à-dire dans le cas du système solaire.

## 1.1 Initialisation des paramètres

Dans un premier temps, la vitesse initiale de notre point sera nulle. On travaillera dans un plan (de coordonnées x et y). Voici la liste des paramètres initialisés en début de code :

- Le nombre de pôles
- L'intensité des différents potentiels (notée k)
- La taille du cadre N (c'est-à-dire que notre cadre sera de dimension NxN)
- Les coordonnées des pôles (notées X et Y)
- Le rayon des pôles
- Les couleurs associées à chaque pôle
- Les coordonnées initiales de notre corps ponctuel (notées x et y)
- La composante en x et en y de la vitesse initiale de notre corps ponctuel (noteé vx et vy)
- Le pas de temps (noté dt)

## 1.2 Calcul de l'accélération

Calcul de l'accélération selon x et y : Le but est ici de résoudre les équations du mouvement de notre corps ponctuel. Les pôles attractifs ont un potentiel fixe (constant dans le temps) de la forme k/r, avec k l'intensité du potentiel. Nous avons un corps ponctuel en contact avec plusieurs champs de potentiels créés par les plots, ce corps ponctuel est donc soumis aux forces exercées par chaque pôle attractif. Comme la force dérive du potentiel, celle-ci s'exprime de la manière suivante :

$$\overrightarrow{F}_{\text{pôle}\to\text{corps}} = -\overrightarrow{\nabla V} = -\frac{k}{r^2}\overrightarrow{u}_{\text{pôle,corps}}$$

Avec,

- $\overrightarrow{F}_{\text{pôle}\to\text{corps}}$ : la force exercée par le pôle attractif sur notre corps ponctuel
- $r$ : la distance entre les deux corps
- $\overrightarrow{u}_{\text{pôle,corps}}$ : le vecteur unitaire dirigé du pôle attractif vers notre corps ponctuel et qui a pour expression, $\frac{\overrightarrow{PC}}{\|\overrightarrow{PC}\|}$ avec $\overrightarrow{PC}$ le vecteur non nul et $\|\overrightarrow{PC}\|$ sa norme qui est la distance entre les deux corps.
- $V$ : le potentiel

On obtient donc une formule pour notre accélération selon X et selon Y :

$$\begin{pmatrix} a_x = -k \times \frac{x-X}{((x-X)^2+(y-Y)^2)^{\frac{3}{2}}} \\ a_y = -k \times \frac{y-Y}{((x-X)^2+(y-Y)^2)^{\frac{3}{2}}} \end{pmatrix}$$

Avec

- x : l'abscisse de notre point ponctuel
- y : l'ordonnée de notre point ponctuel
- X : l'abscisse de notre pôle attractif
- Y : l'ordonnée de notre pôle attractif

L'accélération de notre corps ponctuel est donc la somme des forces exercées par les différents pôles attractifs sur notre objet.
Après avoir initialisé les paramètres et calculé les différentes valeurs de l'accélération, nous pouvons maintenant nous intéresser à la résolution des équations différentielles en utilisant différentes méthodes de résolution d'équation différentielles afin de pouvoir déterminer les trajectoires de notre corps ponctuel.

# 2 Méthodes de résolutions numériques

## 2.1 Méthode d'Euler

La première méthode de résolution numérique que nous utilisons pour déterminer la trajectoire de notre point est la méthode d'Euler.
Cette méthode est une technique permettant de résoudre numériquement des équations différentielles ordinaires (EDO) de la forme $y'(t) = f(y(t), t)$ avec une condition initiale.
La méthode d'Euler est d'ordre 1 et repose sur l'approximation locale de la solution d'une EDO par une tangente. Pour un petit pas de temps dt, la méthode d'Euler estime la solution au point suivant t+1=t+dt.
Autrement dit, cette méthode consiste à avancer pas à pas à partir d'un point initial en utilisant une estimation de la dérivée (la pente) à chaque étape pour extrapoler la valeur de la fonction au point suivant.

On rappelle que $f(t+dt) = f(t) + dt \times \frac{df}{dt}$

Ainsi on en déduit que :

$$v_x(t+dt) = v_x(t) + dt \times a_x(t)$$
$$v_y(t+dt) = v_y(t) + dt \times a_y(t)$$

Et,

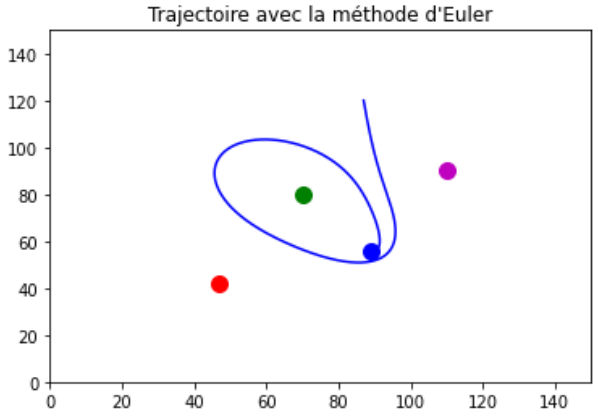
$$x(t+dt) = x(t) + dt \times v_x(t)$$
$$y(t+dt) = y(t) + dt \times v_y(t)$$

Voici comment nous avons codé la fonction associée à la méthode d'Euler :

```
#Calcul des coordonnées x et y ainsi que la vitesse selon x et selon y en utilisant la méthode d'Euler
def MethodeEuler(x,y,vx,vy,dt,Nb_poles):
    acceleration=Acceleration(x,y,X,Y,k,Nb_poles)
    acc_x=acceleration[0]
    acc_y=acceleration[1]

    vx_plus_dt=vx+dt*acc_x
    vy_plus_dt=vy+dt*acc_y

    x_plus_dt=x+dt*vx
    y_plus_dt=y+dt*vy

    return (x_plus_dt,y_plus_dt,vx_plus_dt,vy_plus_dt)
```

Afin d'obtenir la trajectoire de notre corps ponctuel, nous devons créer une fonction qui nous retourne les trajectoires selon x et selon y.
Pour cela, il faut appliquer une condition d'arrêt : cette condition s'applique si la distance entre notre corps ponctuel et le pôle attractif est inférieure au rayon du pôle. En effet, si cette distance est inférieure au rayon du pôle, alors on considère que notre corps ponctuel a atterri sur le pôle en question.

Voici le tracé de la trajectoire avec la méthode d'Euler pour une position initiale donnée : x=87 et y=120




## 2.2 Méthode d'Euler à pas variable

La deuxième méthode de résolution numérique que nous allons utiliser dans ce projet est la méthode d'Euler avec un pas de temps variable. Cette méthode est différente des méthodes « classiques » utilisées pour résoudre des équations différentielles ordinaires.

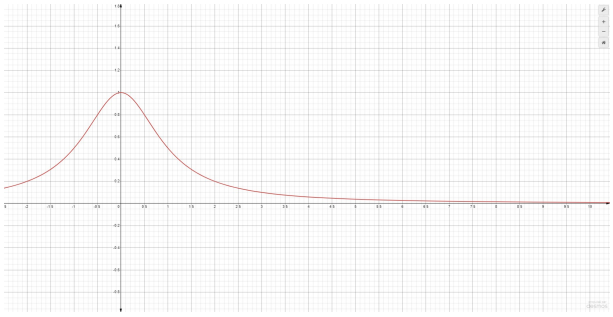
Dans notre étude, comme la vitesse peut varier rapidement en très peu de temps, le pas de temps défini peut ne pas être adapté au calcul et renvoyer des solutions fausses (avec une marge d'erreur assez grande).
Pour pallier ce problème, nous allons diminuer le pas de temps en fonction de la variation de la vitesse de notre corps ponctuel. En effet, le pas de temps dt va diminuer lorsque la vitesse varie rapidement et augmenter le pas de temps lorsque la vitesse varie lentement. Il faut également penser à borner notre pas de temps, cela signifie que nous devons nous assurer que notre pas de temps de soit ni trop grande ni trop petite (ne pas tendre vers 0 trop rapidement).

Voici comment nous allons procéder pour faire varier notre pas de temps dt en fonction de la variation de vitesse de notre corps ponctuel.
On calcule la différence de vitesse entre le temps t+dt et la vitesse au temps t : Si cette différence est trop grande, alors nous faisons diminuer le pas de temps dt.
Pour cela, nous calculons le rapport qu'on notera A entre la différence de vitesse calculée auparavant et la norme de la vitesse au temps dt :

$$A = \frac{\sqrt{(v_x(t+dt))^2 + (v_y(t+dt))^2} - \sqrt{(v_x(t))^2 + (v_y(t))^2}}{\sqrt{(v_x(t))^2 + (v_y(t))^2}}$$

Ainsi, notre pas de temps dt diminuera en suivant la fonction $\frac{1}{A^2+1}$

Voici le tracé de la courbe $f(t) = \frac{1}{x^2+1}$ pour avoir un aperçu de l'évolution de notre pas de temps dt. Cette fonction a été choisie pour faire diminuer notre pas de temps dt car elle tend lentement vers 0 :



Par ailleurs, si la norme de la vitesse au temps t est nulle, c'est-à-dire $\sqrt{(v_x(t))^2 + (v_y(t))^2} = 0$ alors on applique la méthode d'Euler. Il n'est pas nécessaire dans ce cas-là de faire varier le pas de temps dt.

Voici comment nous avons codé la fonction associée à la méthode d'Euler avec un pas de temps variable :

```
#On calcul les coordonnées de x et de y et la vitesse selon x et selon y en utilisant la méthode d'Euler à pas variable
def MethodeEuler_pas_variable(x,y,vx,vy,dt,dtmax):
    acceleration=Acceleration(x,y,X,Y,k,Nb_poles)
    acc_x=acceleration[0]
    acc_y=acceleration[1]
    dtmin=10**-8
    eps=10**-7

    vx_plus_dt=vx+dt*acc_x
    vy_plus_dt=vy+dt*acc_y

    if sqrt(vx**2+vy**2)==0:
        x_plus_dt=x+vx*dt
        y_plus_dt=y+vy*dt
        return(x_plus_dt,y_plus_dt,vx_plus_dt,vy_plus_dt,dt)

    Delta_v=sqrt(vx_plus_dt**2+vy_plus_dt**2)-sqrt(vx**2+vy**2)
    A=Delta_v/sqrt(eps+vx**2+vy**2) #On ajoute un terme epsilon très petit pour que le terme sous la racine ne soit pas nulle
    dt=dtmin+(dtmax-dtmin)*(1/(A**2+1)) #On vait varier notre dt avec une fonction de type (1/1+x**2)

    x_plus_dt=x+dt*vx
    y_plus_dt=y+dt*vy

    return (x_plus_dt,y_plus_dt,vx_plus_dt,vy_plus_dt,dt)
```

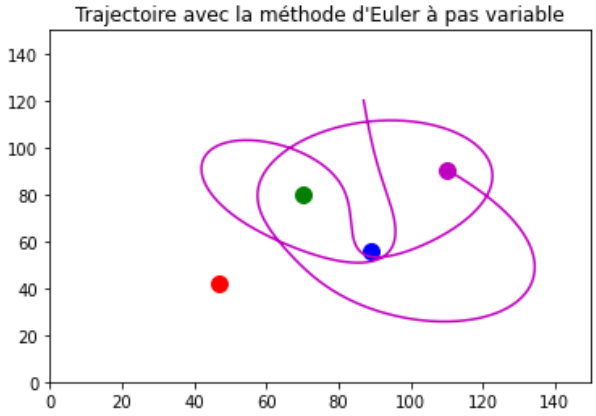
Nous avons également créé une fonction qui nous retourne les trajectoires selon x et selon y avec cette méthode (Cf. code)

Cette méthode d'Euler avec un pas de temps variable présente de nombreux avantages :

— Elle diminue le temps de calcul

— Elle augmente la précision et donc réduit la marge d'erreur

Voici le tracé de la trajectoire avec la méthode d'Euler avec un pas de temps variable pour une position initiale donnée : x=87 et y=120




## 2.3 Méthode de Runge-Kutta à l'ordre 4

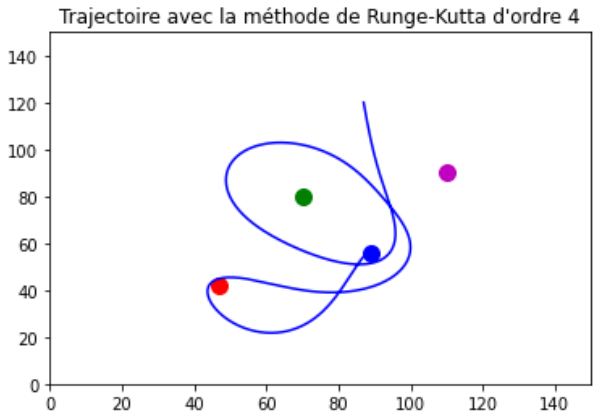
La troisième méthode de résolution numérique est la méthode de Runge Kutta à l'ordre 4. Cette méthode est beaucoup plus précise que celles utilisés jusqu'à maintenant. En effet, elle améliore la précision des approximations successives en utilisant une combinaison de plusieurs évaluations de la fonction étudié. Elle utilise la méthode de Simpson.

Voici comment nous avons procédé pour construire la fonction associée à cette méthode :
On calcule la pente au début de l'intervalle noté k1 (1er coefficient) pour faire un pas avec la méthode d'Euler jusqu'au milieu de l'intervalle t+dt/2 et on obtient une première approximation de k2 (2ème coefficient) qui est la pente du milieu.
Après on réitère le pas, mais maintenant avec la pente k2 afin d'obtenir une approximation plus précise de k3 (3ème coefficient) qui est également la pente au milieu de l'intervalle.
Enfin on utilise k3 pour aller à la fin de l'intervalle qui est t+dt et on utilise l'approximation de k4 (4ème coefficient) pour calculer la pente à la fin d'intervalle.
Ainsi avec nos 4 coefficients, nous pouvons approximer le point suivant de notre position et de notre vitesse.

Ici nos équations différentielles sont couplées, voici comment nous avons codé la fonction associée à la méthode de Runge-Kutta à l'ordre 4 :

```python
def RK4(x,y,vx,vy,X,Y,k,Nb_plots,dt):
    acceleration=Acceleration(x,y,X,Y,k,Nb_poles)
    acc_x=acceleration[0]
    acc_y=acceleration[1]

    kx1=dt*vx #Calcul du premier coefficient pour x et y
    ky1=dt*vy

    jx1=dt*acc_x #Calcul du premier coefficient pour vx et vy
    jy1=dt*acc_y

    acceleration=Acceleration(x+kx1/2,y+ky1/2,X,Y,k,Nb_poles) #L'accélération est recalculée en fonction des premiers coefficients
    acc_x=acceleration[0]
    acc_y=acceleration[1]

    kx2=dt*(vx+jx1/2)
    ky2=dt*(vy+jy1/2)

    jx2=dt*acc_x
    jy2=dt*acc_y

    acceleration=Acceleration(x+kx2/2,y+ky2/2,X,Y,k,Nb_poles)
    acc_x=acceleration[0]
    acc_y=acceleration[1]

    kx3=dt*(vx+jx2/2)
    ky3=dt*(vy+jy2/2)

    jx3=dt*acc_x
    jy3=dt*acc_y

    acceleration=Acceleration(x+kx3,y+ky3,X,Y,k,Nb_poles)
    acc_x=acceleration[0]
    acc_y=acceleration[1]

    kx4=dt*(vx+jx3)
    ky4=dt*(vy+jy3)

    jx4=dt*acc_x
    jy4=dt*acc_y

    x_plus_dt = x + 1/6*(kx1 + 2*kx2 + 2*kx3 + kx4)
    y_plus_dt = y + 1/6*(ky1 + 2*ky2 + 2*ky3 + ky4)

    vx_plus_dt = vx + 1/6*(jx1 + 2*jx2 + 2*jx3 + jx4)
    vy_plus_dt = vy + 1/6*(jy1 + 2*jy2 + 2*jy3 + jy4)

    return(x_plus_dt,y_plus_dt,vx_plus_dt,vy_plus_dt)
```

Voici le tracé de la trajectoire avec la méthode de Runge-Kutta à l'ordre 4 pour une position initiale donnée : x=87 et y=120

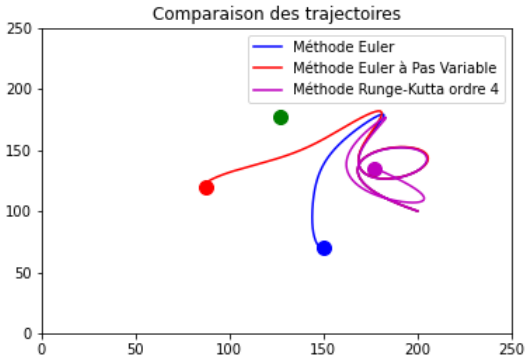



## 2.4 Comparaison des différentes méthodes

Après avoir codé les différentes fonctions pour chaque méthode, nous pouvons tester et comparer les différentes trajectoires de notre point.

Par exemple, nous avons pris une position initiale de notre point dont les coordonnées sont : x=200 et y=100. Nous constatons immédiatement qu'en fonction de la méthode utilisée, notre point n'atterrit pas sur le même plot.

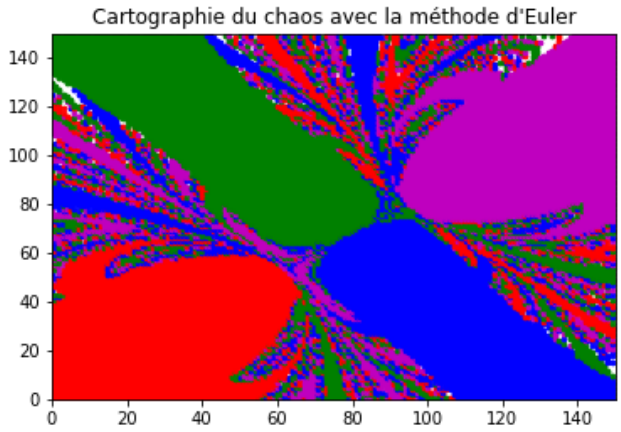
Voici le graphique montrant les différentes trajectoires en fonction de la méthode utilisée :



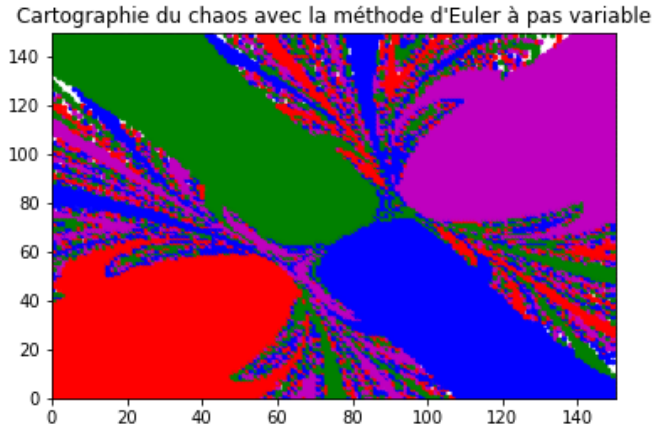

Ainsi, pour certaines positions initiales, le choix de la méthode par rapport à une autre peut avoir des changements significatifs sur la trajectoire de notre point et sur les coordonnées d'arrivée de notre point.

# 3 Cartographie du chaos

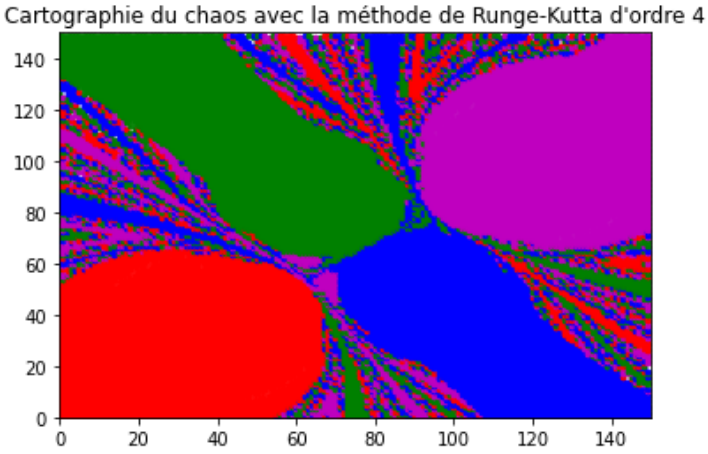
Voici comment nous allons cartographier les zones de stabilité ainsi que les zones chaotiques : Nous attribuons une couleur à chaque pôle attractif, ces pôles seront représentés sur la cartographie.
Si la trajectoire de notre point dépasse une distance maximale de notre cadre, la couleur blanche sera appliquée à la position initiale de notre point.
Si notre point se met en orbite ou atterrit sur l'un des pôles alors on affectera la couleur du pôle d'arrivé à la position de départ de notre point.
Pour obtenir une cartographie dans laquelle tous les points de la carte sont exploités, nous effectuerons un balayage point par point de notre cadre. Par exemple, si la taille de notre cadre est de 200 x 200 alors la trajectoire sera calculée pour 40 000 points.
Cela se traduit dans le code python par une double boucle, l'une pour les lignes, l'autre pour les colonnes.
Voici la cartographie du chaos effectuée avec les 3 méthodes de résolution numérique : Euler, Euler avec pas de temps variable, Runge-Kutta à l'ordre 4. Cette cartographie a été effectuée pour des pôles disposés de manière aléatoire et ayant des valeurs de k différentes : k=[7000(rouge),7900(vert),8200(bleu),7500(magenta)]




**Cartographie du chaos avec la méthode d'Euler**




**Cartographie du chaos avec la méthode d'Euler avec un pas de temps variable**




**Cartographie du chaos avec la méthode de Runge-Kutta à l'ordre 4**

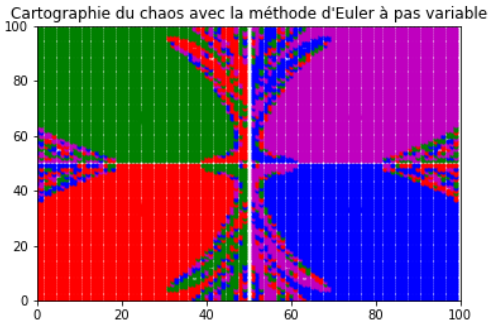
Les coordonnées des pôles sont :

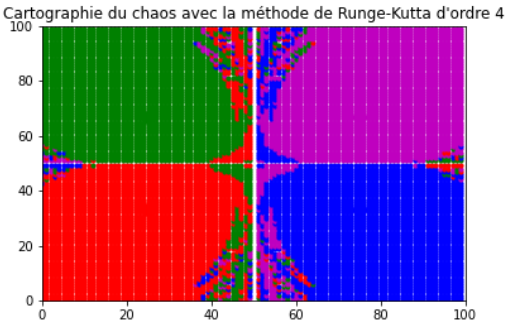
— Pôle rouge : (X,Y)=(47,42)

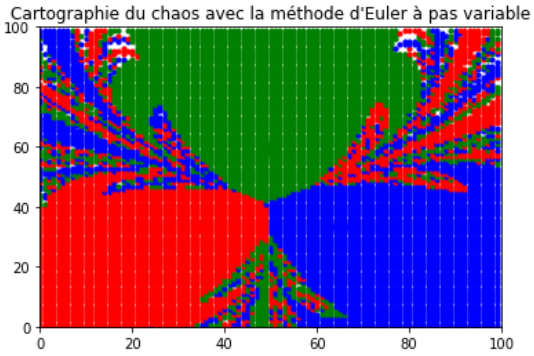
— Pôle vert : (70,80)
— Pôle bleu : (89,56)
— Pôle magenta : (110,90)

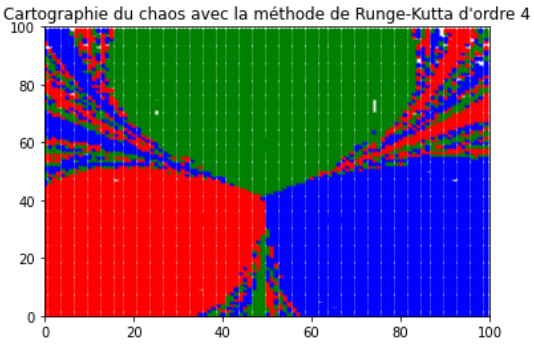
Après avoir cartographié les chaos déterministe en utilisant les 3 méthodes présentés auparavant, nous pouvons directement constater une différence significative des zones de stabilité et des zones chaotiques entre la méthode d'Euler pas variable et la méthode de Runge-Kutta à l'ordre 4 :
La méthode de Runge-Kutta à l'ordre 4 nous donne une visualisation plus précise de la cartographie du chaos. En effet, on constate que les zones de stabilité sont plus détaillés et complètes que celles données par la méthode d'Euler et Euler avec un pas variable. On observe bien les zones chaotiques où toutes les couleurs sont mélangés.

Voici quelques cartographies du chaos déterministe pour des systèmes ayant une certaine géométrie :



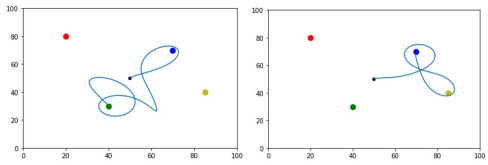




# 4 Vitesse initiale non nulle

Dans cette partie, nous assimilons notre point à une comète et les plots à des planètes. L'objectif est de voir comment la vitesse initiale d'une comète influence son comportement chaotique quand elle se déplace dans un champ gravitationnel créé par quatre planètes de différentes couleurs. On a étudié comment la norme de la vitesse initiale et l'angle de cette vitesse affectent la trajectoire de la comète dans ce champ.

## 4.1 Vitesse aléatoire

Dans cette partie, on a fixé la position de départ de la comète et on a fait varier sa vitesse initiale de manière aléatoire, en modifiant les composantes vx et vy du vecteur vitesse, en changeant donc sa direction et sa norme. Cela nous a permis de tester différentes conditions initiales et de voir comment elles affectent le système.



**Trajectoires de la comète pour des vitesses initiales aléatoires**

Les images ci-dessus montrent la trajectoire de la comète pour 2 vitesses initiales différentes avec la méthode d'Euler à pas variable, ayant la même origine.
On peut alors voir que, pour 2 vitesses différentes, la comète n'atterrit pas sur la même planète (sur la verte puis la bleue), bien qu'il parte du même point.
Décomposons alors le problème en deux parties.
Analysons, d'une part, l'effet de la norme de cette vitesse sur le système, et, d'autre part, l'influence de son angle.

## 4.2 Norme de la vitesse initiale

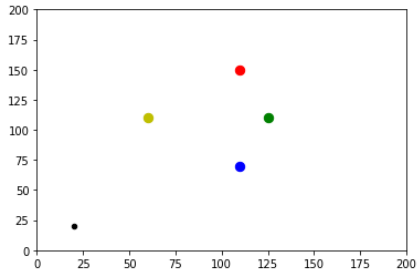
Tout d'abord, voyons quelle est l'influence de la norme de la vitesse sur le comportement chaotique ou non du système.
Pour cela, voyons sur quelle planète la comète atterrit (lié à l'importance du chaos) selon la norme de sa vitesse initiale. Cette dernière part du même endroit pour chaque norme testée et sa vitesse initiale forme un angle de $\frac{\pi}{4}$

Pour ce faire, nous allons balayer les normes de vitesse possibles, et, pour chacune d'entre elles, tester notre programme pour des valeurs proches de chaque norme. Afin de quantifier le chaos, nous récupérons la donnée de la position finale de la comète dans une liste qui va compter le nombre de fois que l'astéroïde atterrit sur chaque planète ou sort du cadre.
Suite à cela, nous mesurons l'importance du chaos grâce à l'écart type des éléments du compteur pour chaque norme testée. Cela nous permet alors d'avoir une idée de l'importance du chaos car celui-ci est inversement proportionnel à cet écart type.

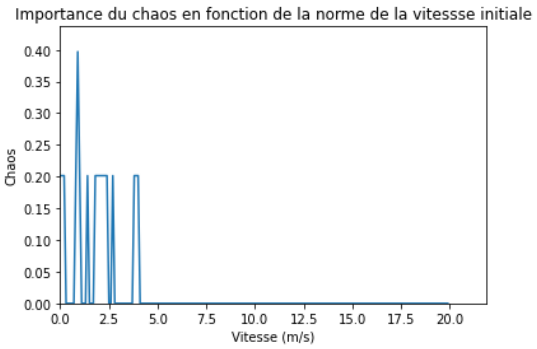
```
# mesure du chaos en fonction de la vitesse en mesurant l'écart type proche d'une certaine vitesse
# => plus la vitesse est grande, plus le système est déterministe

for j in range(Nv):  # Nv normes de vitesses

    x = 20  # position initiale
    y = 20
    v[j] = j / 10  # +0.1  pour chaque vitesse testée
    vx = v[j] / sqrt(2)
    vy = v[j] / sqrt(2)
    com = np.zeros(5)  # compte nombre de fois chaque cas
    moy = Nt / len(com)  # moyenne pour écart type


    for m in range(Nt):  # Nt vitesses proches => Au total : Nv*Nt vitesses
        TEPV = Trajectoire_Euler_Pas_Variable(x, y, vx, vy, X, Y, dtmax, dt, Nb_poles, Rayon, Couleurs)  # donne com grâce à Euler variable
        vx += pas / sqrt(2)  # on ajoute au maximum : Nt*pas/sqrt2 = 1/10*sqrt(2)
        vy += pas / sqrt(2)

    # print(v[j], com)

    for i in range(len(com)):

        chaos[j] += ((com[i] - moy)**2) / Nt  # variance

    chaos[j] = -sqrt(chaos[j])  # chaos inversement proportionnel à l'écart type

for i in range(len(chaos)):

    chaos[i] += -np.min(chaos)  # échelle entre 0 et chaos max
```

**Code utilisant cette méthode**

Testons cette méthode pour deux configurations de notre champ gravitationnel. Dans un premier temps, testons le programme pour un champ ne comportant pas de symétrie. Testons alors cette méthode avec un très grand nombre d'itérations afin de mesurer l'importance du chaos en fonction de la norme de la vitesse initiale de la comète.

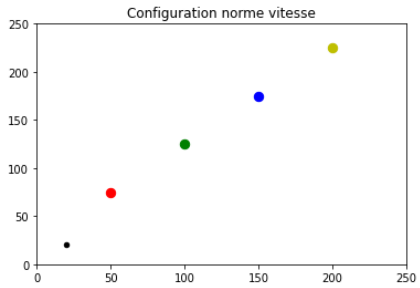


**Configuration n°1 du champ gravitationnel**

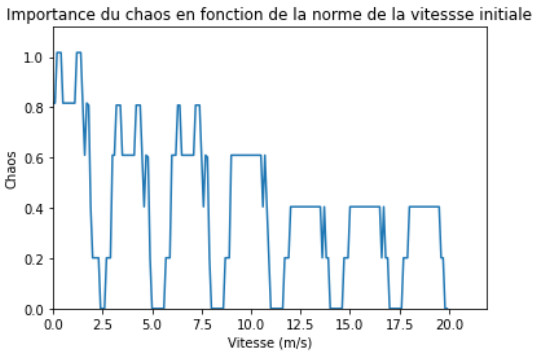



On peut alors s'apercevoir qu'à grande vitesse initiale, l'avenir de la comète sera la même (crash sur une certaine planète ou sortie), et que donc le système n'est pas chaotique.
Au contraire, pour des plus faibles vitesses, l'avenir de la comète est beaucoup plus sensible à une petite variation de norme de vitesse, et le système est, par conséquent, plus chaotique. En effet, c'est le cas pour certaines normes de vitesse comprises entre 0 et 4 m/s. De plus, c'est pour une norme de vitesse initiale d'environ 1 m/s que le système est le plus chaotique.

Testons désormais le programme pour un champ comportant une symétrie axiale, car les 4 planètes ont les mêmes potentiels attractifs et rayons.




**Configuration n°2 du champ gravitationnel**




On peut alors s'apercevoir que l'importance du caractère chaotique du système varie de façon quasi périodique (de période = 2.7 m/s), due à la symétrie du problème.
De plus, comme précédemment, plus la vitesse initiale de la comète est importante, moins le système est chaotique. Enfin, le fait que les motifs ne se répètent pas parfaitement semble être lié à l'imprécision de la méthode d'Euler à pas variable.

En plus de cela, dans notre cas, les valeurs de vitesse telles que le chaos est absent correspondent au cas où la comète sort du cadre pour toutes les valeurs proches de la vitesse testée. Cela est dû au fait que, pour ces vitesses, le système n'est jamais trop

proche d'une planète mais se situe entre deux planètes, qui exercent alors des forces de mêmes normes mais de sens opposés.
De ce fait, la comète s'éloigne des planètes, et, ne subissant presque plus de force attractive, sort du cadre.
Au contraire, pour certaines vitesses, la comète se retrouve proche d'une planète et a donc plus de chances d'atterrir sur cette dernière.

## 4.3 Étude de l'angle de la vitesse initiale

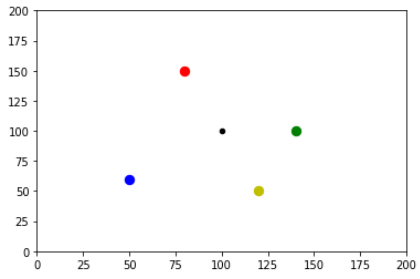
Voyons maintenant l'influence de l'angle de la vitesse initiale sur le comportement chaotique ou non du système.
Pour cela, observons sur quelle planète la comète atterrit (lié à l'importance du chaos) selon l'angle de sa vitesse initiale.
Cette dernière part du même endroit pour chaque angle testé, et sa vitesse initiale a une norme fixe et forme un angle de phi avec l'axe des x.

Pour ce faire, nous balayerons les angles de vitesses possibles et, pour chacun d'eux, testerons notre programme pour des valeurs proches de chaque angle. Afin de quantifier le chaos, nous utilisons la même méthode que précédemment.

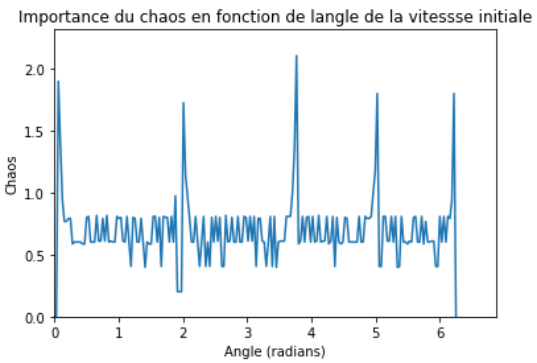
```
# mesure du chaos en fonction de la vitesse en mesurant l'écart type proche d'une certaine vitesse
# => plus la vitesse est grande, plus le système est déterministe

for j in range(Nangle):  # Nangle angles de vitesse

    x = 100
    y = 100
    phi[j] = j * 2 * pi / Nangle  # de 0 à 2*pi
    vx = R * np.cos(phi[j])  # entre -R et R
    vy = R * np.sin(phi[j])
    com = np.zeros(5)  # compte nombre de fois chaque cas
    moy = Nt / len(com)  # moyenne pour écart type


    for m in range(Nt):  # Nt angles proches => Nangle*Nt angles
        TEPV = Trajectoire_Euler_Pas_Variable(x, y, vx, vy, X, Y, dtmax, dt, Nb_poles, Rayon, Couleurs)  # donne com

        vx += R * np.cos(phi[j]) / 10 * Nt  # on teste angles proches pour chaque angle
        vy += R * np.sin(phi[j]) / 10 * Nt  # on ajoute au max R * np.sin(phi[j]) / 10

    # print('angle num', j, ',', 'angle :', phi[j], 'compteur', com)

    for i in range(len(com)):

        chaos[j] += ((com[i] - moy)**2) / Nt  # variance

    chaos[j] = -sqrt(chaos[j])  # chaos inversement proportionnel à l'écart type


for i in range(len(chaos)):
    chaos[i] += -np.min(chaos)  # échelle entre 0 et chaos max
```

**Code utilisant cette méthode**

Testons cette méthode pour deux configurations de notre champ gravitationnel.
Dans un premier temps, testons le programme pour un champ ne comportant pas de symétrie.

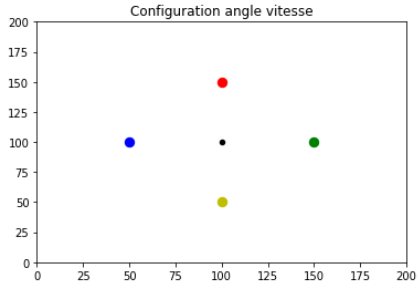


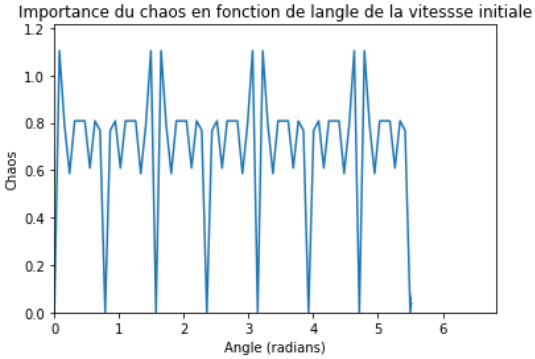

**Configuration n°3 du champ gravitationnel**



On peut alors s'apercevoir que l'importance du chaos varie très peu pour la plupart des angles.
Au contraire, pour certaines valeurs, le système est très peu chaotique ou très chaotique où l'avenir de la comète est beaucoup plus sensible à une petite variation d'angle de vitesse.

Testons désormais le programme pour un champ comportant une symétrie centrale, car les 4 planètes ont les mêmes potentiels attractifs et les mêmes rayons.



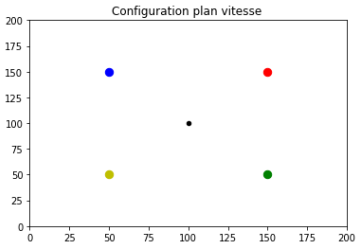

**Configuration n°4 du champ gravitationnel**

On peut alors s'apercevoir que l'importance du caractère chaotique du système varie de façon périodique (de période $T = \frac{\pi}{2}$rad), due à la symétrie du problème.

Le fait que les motifs se répètent parfaitement est dû au fait que les 4 planètes forment un carré, dont le centre est la position initiale du système. En effet, le champ gravitationnel du problème est invariant par rotation d'un angle de $\frac{\pi}{4}$.
De plus, pour des valeurs exactement égalent à un multiple de $\frac{\pi}{4}$ la comète atterrit sur la planète située exactement en face d'elle ou alors se déplace entre 2 planètes, qui ont alors des forces qui se compensent, et sort du cadre.
Cependant, le système est très chaotique pour des valeurs proches d'un multiple de $\frac{\pi}{2}$, car le système est attiré par la planète qu'il a en face de lui, mais pas assez, qui dévie donc sa trajectoire et l'expulse soit hors du cadre soit sur une autre planète.

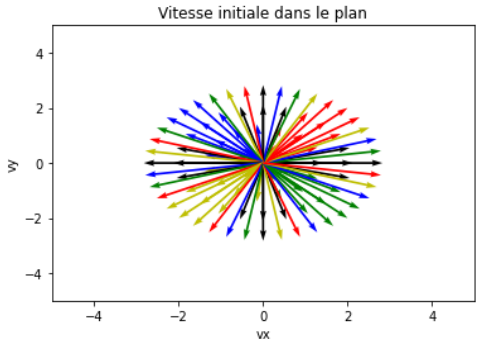
## 4.4 Vitesse initiale dans le plan

Après avoir mesuré l'effet de la norme et de l'angle de la vitesse initiale, intéressons-nous aux vecteurs vitesse dans leur plan.
Pour ce faire, pour toutes les normes de la vitesse, nous allons balayer tous les angles initiaux possibles et peindre le vecteur de la vitesse initiale en fonction de l'avenir de l'astéroïde.

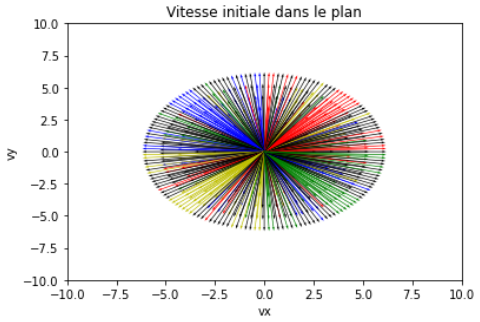



**Configuration n°5 du champ gravitationnel**

Traçons ces vecteurs vitesses pour différentes normes et angles et attribuons-leur une couleur correspondant à la position finale de la comète.




Augmentons la précision en balayant beaucoup plus d'angles et de normes.




Nous pouvons alors remarquer que les vecteurs ont des axes de symétrie formés par les axes de vx et vy.
Parlons des effets de cet angle initial. On distingue 3 effets :

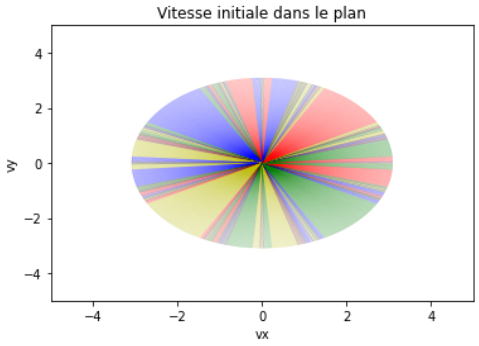
— On s'aperçoit alors que pour des angles proches des multiples impairs de $\frac{\pi}{4}$, la comète atterrit toujours sur la planète se situant en face d'elle.

— De plus, pour des angles proches de multiples pairs de $\frac{\pi}{4}$, la comète atterrit sur une des 2 planètes qui se situent en face d'elle.

— Enfin, pour des angles situés entre les multiples de $\frac{\pi}{4}$, il est très difficile de prévoir l'avenir de l'astéroïde car ce dernier est très sensible à une petite variation de l'angle initial.

Parlons maintenant des effets de cette norme initiale.
On remarque que la norme a moins d'influence sur le système que l'angle.
Cependant, plus la norme de la vitesse est élevée, plus la probabilité que le système sorte du cadre est grande. En effet, si celui-ci n'a pas une vitesse initiale orientée vers une des 4 planètes, la comète a une vitesse trop grande et cette dernière est alors moins sensible aux forces des planètes. En plus de cela, les forces gravitationnelles exercées par les 2 planètes, étant de sens opposés, font que la comète n'atterrit pas sur une de ces planètes.

Exécutons désormais le code pour un très grand nombre d'angles initiaux, en fixant la norme initiale.




Ce graphique nous permet finalement de mieux distinguer les zones correspondant à chaque cas. En effet, nous repérons mieux les zones déterministes (de couleurs rouges, jaunes, vertes et bleues) et les zones chaotiques (mélange de ces couleurs et de la couleur noire).

## 4.5 Résumé

En somme, cette étude montre l'importance des conditions initiales, telles que la norme et l'angle de la vitesse, dans le comportement chaotique des objets dans un champ gravitationnel.
En effet, d'une part, nous avons constaté que des vitesses initiales plus élevées rendent les trajectoires plus prévisibles, tandis que des vitesses plus faibles augmentent le caractère chaotique du système.
En plus de cela, la symétrie du problème peut rendre périodique l'importance du comportement chaotique de la comète selon sa norme initiale.
D'autre part, l'angle de la vitesse initiale a également une influence significative, pouvant créer des périodes de comportement chaotique autour de valeurs spécifiques liées à la symétrie du problème.

De plus, pour certains angles, l'astéroïde peut avoir un comportement très chaotique car la direction initiale du système peut faire converger celui-ci vers une autre planète. Enfin, l'analyse des vecteurs vitesse dans leur plan révèle des zones de chaotiques et des régions déterministes, selon l'angle initial, dépendant de la symétrie des planètes. En plus de cela, ces zones sont plus déterministes quand la norme de la vitesse initiale est plus grande.

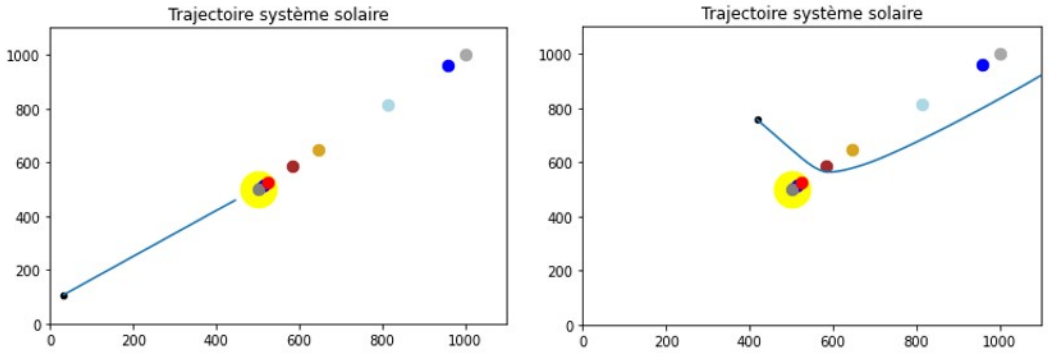
Enfin, ces observations, bien qu'elles proviennent d'un modèle simplifié d'un champ gravitationnel, montrent les principes fondamentaux des systèmes dynamiques (lois de Newton) et l'importance de la vitesse initiale du système.

# 5 Application au système solaire

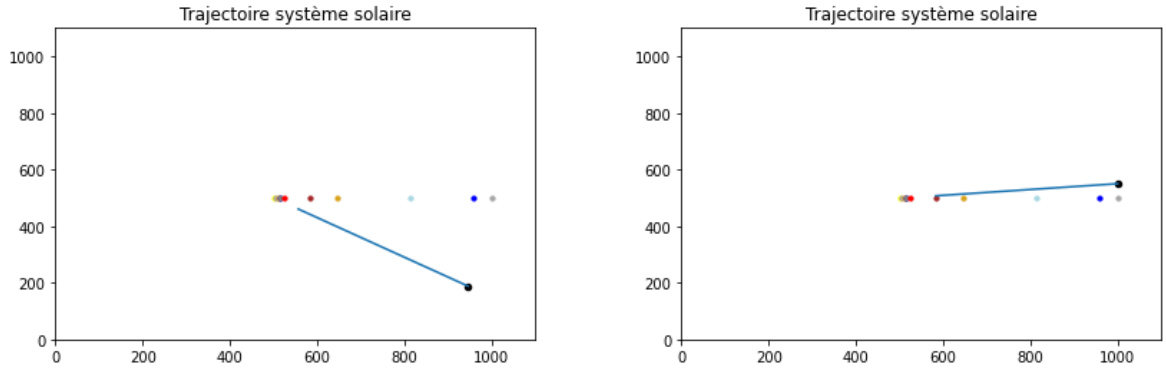
Appliquons maintenant ce programme à un champ gravitationnel étant plus proche de notre système solaire.
Plaçons le Soleil au centre et adaptons le rayon et le potentiel de chaque astre considéré du système solaire (Mercure, Vénus, Terre, Lune, Mars, Jupiter, Saturne, Uranus, Neptune). De plus, les 8 planètes et la Lune sont positionnées autour de ce dernier comme si elles

suivaient leur orbite.

On considère que la comète a une très grande vitesse par rapport aux planètes, qui sont considérées comme immobiles. On néglige, de plus, l'influence des astres entre eux. Voyons alors quel est le mouvement de notre système dans ce plan pour plusieurs configurations et positions initiales.
Testons notre code pour des planètes alignées, ayant des tailles proportionnelles à leur masse sur le graphe ou faisant la même taille que le Soleil.
Nous alignons les planètes afin de simplifier le programme, car les planètes peuvent se situer tout autour du Soleil en formant une ellipse.




**Trajectoires de la comète**




On s'aperçoit que la comète n'atterrit pas sur le Soleil seulement si elle est située très proche d'une planète ayant un grand potentiel attractif.
En effet, quand l'astéroïde est très proche d'une grosse planète du système solaire (Jupiter ou Saturne), il atterrit sur celle-ci ou est dévié et sort du cadre.
Dans tous les autres cas, l'astéroïde atterrit sur le Soleil car son potentiel attractif est bien plus important que celui de n'importe quelle planète de son système stellaire.

Finalement, le Soleil constitue une barrière protégeant la Terre et les autres planètes de la majorité des astéroïdes entrant dans le système solaire.
En appliquant cette analyse à un modèle simplifié du système solaire, nous avons constaté que le Soleil, grâce à son fort potentiel attractif, protège les planètes en attirant la comète.

# 6 Conclusion

En conclusion, la cartographie du chaos déterministe dans un système mécanique constitue une approche très intéressante pour l'étude des systèmes dynamiques.

À travers ce rapport, nous avons implémenté diverses méthodes numériques pour analyser et visualiser les comportements chaotiques dans un système mécanique constitué de plusieurs pôles attractifs. L'utilisation de méthodes numériques telles que la méthode d'Euler à pas variable et Runge-Kutta d'ordre 4 nous a permis de simuler et de comprendre les trajectoires complexes de ce système.
Les résultats obtenus montrent clairement comment les petites variations des conditions initiales peuvent conduire à des comportements radicalement différents. Cela illustre bien la nature intrinsèquement imprévisible des systèmes chaotiques.

Enfin, il faut également noter que ce projet à été réalisé en utilisant le langage Pyhton qui n'est pas un langage compilé. Cela se traduit donc par des temps de calculs extrêmement longs (la cartographie du chaos en utilisant Runge-Kutta à l'ordre 4 pour un cadre 150 x 150 prend environ 3h !) et la taille du programme n'est pas très compacte. Il est donc préférable d'effectuer ce projet en langage C pour des questions de temps de calcul. Néanmoins, ce projet est bien évidemment réalisable en langage Python comme le montre notre rapport.