DS n° 03 – DS03

- Faire tous les exercices dans un même fichier NomPrenom.py à sauvegarder,
- mettre en commentaire l'exercice et la question traités (ex : # Exercice 1),
- ne pas oublier pas de commenter ce qui est fait dans votre code (ex : # Je crée une fonction pour calculer la racine d'un nombre),
- il est possible de demander un déblocage pour une question, mais celle-ci sera notée 0,
- il faut vérifier avant de partir que le code peut s'exécuter et qu'il affiche les résultats que vous attendez.

Générer la trajectoire d'une sonde de la Terre vers Mars

Le 18 février 2021 à 20 h 44, la sonde Perseverance, partie de la Terre le 30 juillet 2020, a atterri sur Mars.

Après avoir été propulsée par une fusée pour s'extraire de l'attraction terrestre, elle entre sur une orbite autour du Soleil le long de laquelle elle se rapproche progressivement de l'orbite de Mars, jusqu'à l'intercepter.

L'objectif de cette épreuve est de déterminer la trajectoire (simplifiée) de la sonde à partir de l'étude des orbites de la Terre et de Mars.

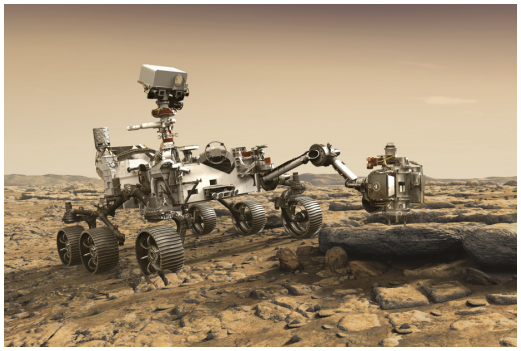

FIGURE 1 – Image virtuelle du robot Perseverance sur Mars.

Tracé et découpage des trajectoires de la Terre et de Mars.

Les paramètres (géométriques ou numériques) d'une ellipse utiles pour cette épreuve sont :

- la longueur du grand rayon (ou demi-grand axe), notée a,
- la longueur du petit rayon (ou demi-petit axe), notée b,
- la distance séparant le centre de l'ellipse et un des foyers (F), notée c,
- l'excentricité de l'ellipse (strictement comprise entre 0 et 1), notée $e = \frac{c}{a}$,

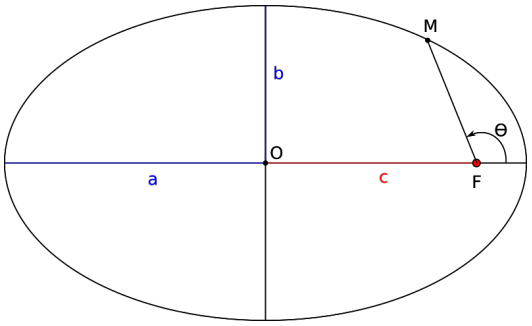

FIGURE 2 – Représentation mathématique d'une ellipse.

1 Mettre en en-tête du fichier les éléments suivants :

- `import matplotlib.pyplot as plt`
- `import numpy as np`

Soit M un point de l'ellipse, le rayon FM peut être calculé en fonction de l'angle θ en utilisant la formule suivante :

$$r(\theta) = \frac{a \cdot (1-e^2)}{1+e \cdot cos(\theta)}$$

2 Écrire la fonction `rayon(a,e,theta)` qui renvoie la valeur de $r(\theta)$.

Solution 1.

```
def rayon(a,e,theta):
    return a*(1-e**2)/(1+e*np.cos(theta))
```

On donne pour la trajectoire de la Terre :

— $a_{terre} = 1{,}49 \times 10^{11}$ m (`aterre` dans le script),

— $e_{terre} = 0{,}016$ (`eterre` dans le script),

— La révolution s'effectuant autour du Soleil, le point F, sera appelé S dans la suite.

Aide Python :

— Il est possible sous Python de tracer une courbe en coordonnées polaires en utilisant la fonction `plt.polar(theta,r)`,

— On générera les valeurs de θ à l'aide de la fonction `theta=np.arange(0,2*np.pi,0.001)`.

3 Écrire le code permettant de tracer la trajectoire de la Terre et faire apparaître la figure 3.

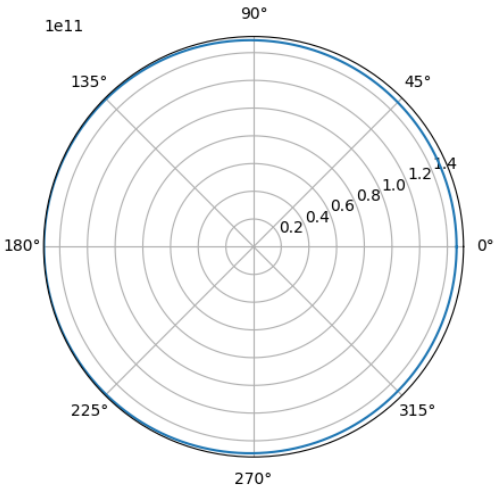

FIGURE 3 – Tracé de la trajectoire de la Terre autour du Soleil sous Python.

Solution 2.

```
aterre=1.49*10**11
eterre=0.016
theta=np.arange(0,2*np.pi,0.001)
plt.polar(theta,rayon(aterre,eterre,theta))
plt.show()
```

On peut calculer la période (en secondes) de révolution d'un astre autour du Soleil à l'aide de la loi de Kepler suivante : $P = \sqrt{\frac{4 \cdot a^3 \cdot \pi^2}{G \cdot (m_{astre} + m_{soleil})}}$

On donne pour effectuer les calculs :

— $G = 6{,}67 \times 10^{-11}\,\mathrm{N\,m^2\,kg^{-2}}$,

— $m_{soleil} = 1{,}97 \times 10^{30}$ kg (`msoleil` dans le script),

— $m_{terre} = 6 \times 10^{24}$ kg (`mterre` dans le script),

4 Créer une fonction `periode(a,m)` qui renvoie la période de révolution (en jours) d'un astre de masse m et de demi-grand axe a. Afficher le résultat numérique correspondant à celle de la Terre. Cette valeur était-elle prévisible ? (Donner la réponse à l'aide d'un `print("Réponse: ")`).

Solution 3.

```
G=6.67*10**(-11)
msoleil=1.97*10**30
mterre=6*10**24
def periode(a,m):
    P=np.sqrt(4*a**3*np.pi**2/(G*(mterre+msoleil)))
    return P/(3600*24)

P=periode(aterre,mterre)
print(P)
print("On trouve environ 365 jours soit une année, durée de révolution autour du soleil. Le rés
```

Deuxième loi de Kepler

La deuxième loi de Kepler ou « loi des aires » dit que le rayon-vecteur reliant une planète au Soleil balaie des aires égales en des durées égales. Le rayon-vecteur est un segment reliant le Soleil, situé à un des foyers de l'ellipse, à la planète située sur l'ellipse.

Cela signifie que sur la figure 4, l'astre, dont l'orbite est tracée en noir, mettra autant de temps pour passer du point P_1 au point P_2 que de A_1 à A_2, car les deux portions d'ellipse SP_1P_2 et SA_1A_2 ont la même aire.

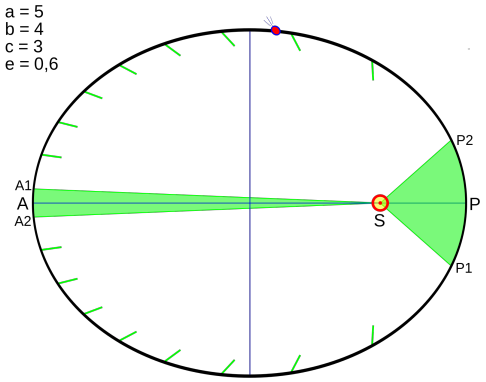

FIGURE 4 – Construction illustrant la deuxième loi de Kepler.

Dans un premier temps, nous allons discrétiser la trajectoire en la décomposant en un ensemble de segments M_iM_{i+1}.
L'aire de la section d'ellipse SM_iM_{i+1} est définie par la formule suivante :
$aire(i) = \frac{1}{2} \cdot r_{i+1} \cdot r_i \cdot \sin(\theta_{i+1} - \theta_i)$, avec (θ_i,r_i) les coordonnées polaires du point M_i par rapport à S.

5 Écrire la fonction `calcul_aire(a,e,i)` qui renvoie l'aire de la section d'ellipse SM_iM_{i+1} à partir des paramètres a et e de l'ellipse et de l'index i du point M_i. Afficher l'aire de l'élément d'index $i = 0$ de l'ellipse de trajectoire de la Terre.

Solution 4.

```
def calcul_aire(a,e,i):
    return 0.5*rayon(a,e,theta[i+1])*rayon(a,e,theta[i])*np.sin(theta[i+1]-theta[i])
print(calcul_aire(aterre,eterre,0))
```

6 Créer une fonction `calcul_aire_ellipse(a,e)` qui à partir de la fonction `calcul_aire(a,e,i)` renvoie l'aire totale de l'ellipse. En déduire l'aire totale de l'ellipse de la trajectoire de la Terre autour du Soleil.

Solution 5.

```
def calcul_aire_ellipse(a,e):
    aire_t=0
    for i in range(len(theta)-1):
        aire_t+=calcul_aire(a,e,i)
    return aire_t
```

```
aire_ellipse=calcul_aire_ellipse(aterre,eterre)
print(aire_ellipse)
```

7 Créer la fonction `calcul_aire_mois(aire_ellipse,P)` qui à partir du calcul de l'aire de l'ellipse et de la période de révolution P détermine l'aire parcourue en un mois (on comptera 30 jours par mois).

Solution 6.

```
def calcul_aire_mois(aire_ellipse,P):
    return aire_ellipse/(P/30)
aire_mois=calcul_aire_mois(aire_ellipse,P)
print(aire_mois)
```

8 Créer une fonction `gen_liste_points(a,e,aire_mois)` qui renvoie une liste contenant les θ_i correspondant à la position de l'astre à chaque début de mois. Les entrées de la fonction sont les caractéristiques de l'orbite (a et e), ainsi que l'aire de la section d'orbite parcourue en un mois calculée grâce à la fonction `calcul_aire_mois(aire_ellipse,P)`. Afficher le résultat pour la Terre.
Indices :

- Une solution consiste à intégrer la surface en partant de $i = 0$, jusqu'à obtenir la valeur `aire_mois` puis stocker la valeur de θ à ce moment-là, et continuer jusqu'à ce que toute la surface soit parcourue,
- On trouvera `[0.533, 1.062, 1.583, 2.096, 2.602, 3.104, 3.605, 4.11, 4.622, 5.142, 5.67, 6.203, 6.283]`.

Solution 7.

```
def gen_liste_points(a,e,aire_mois):
    aire_int=0
    i=0
    liste_points=[]
    while i < len(theta)-1:
        while aire_int<aire_mois and i < len(theta)-1:
            aire_int+=calcul_aire(a,e,i)
            i+=1
        aire_int=0
        liste_points.append(theta[i])
    return liste_points
print(gen_liste_points(aterre,eterre,aire_mois))
```

Si vous n'avez pas réussi les questions précédentes, il est nécessaire de demander le code aux enseignants pour continuer.
Le script donné ici permet de tracer les trajectoires de la Terre et de Mars.

```
a=[1.49*10**11,2.27944*10**11]
e=[0.016,0.093]
m=[6*10**24,6.418*10**23]

fig = plt.figure()
ax = fig.add_subplot(111, projection='polar')

for planete in range(2):
    P=periode(a[planete],m[planete])
    print(P)
    aire_ellipse=calcul_aire_ellipse(a[planete],e[planete])
    aire_mois=calcul_aire_mois(aire_ellipse,P)
    liste_points=gen_liste_points(a[planete],e[planete],aire_mois)
    plt.polar(theta,rayon(a[planete],e[planete],theta))
    ax.scatter(liste_points, rayon(a[planete],e[planete],liste_points))

plt.show()
```

9 Recopier le code précédent afin de faire apparaître les trajectoires.

Tracé de la trajectoire de la sonde

Pour continuer l'exercice, il est plus simple de négliger les excentricités des trajectoires de la Terre et de Mars.

10 Modifier le code précédent en remplaçant la ligne `e=[0.016,0.093]` par `e=[0,0]`.
On donne les valeurs suivantes pour déterminer la trajectoire de la sonde :

- $a_{sonde} = \frac{a_{terre}+a_{mars}}{2}$ (`asonde` dans le script),
- $e_{sonde} = \frac{a_{mars}-a_{terre}}{a_{terre}+a_{mars}}$ (`esonde` dans le script).

11 En s'inspirant du code donné pour la question 9, tracer la trajectoire et les points de passage mensuels de la sonde.

Solution 8.

```
a=[1.49*10**11,2.27944*10**11]
e=[0,0]
m=[6*10**24,6.418*10**23]

fig = plt.figure()
ax = fig.add_subplot(111, projection='polar')

for planete in range(2):
    P=periode(a[planete],m[planete])
    aire_ellipse=calcul_aire_ellipse(a[planete],e[planete])
    aire_mois=calcul_aire_mois(aire_ellipse,P)
    liste_points=gen_liste_points(a[planete],e[planete],aire_mois)
    plt.polar(theta,rayon(a[planete],e[planete],theta))
    ax.scatter(liste_points, rayon(a[planete],e[planete],liste_points))

asonde=(a[0]+a[1])/2
esonde=(a[1]-a[0])/(a[0]+a[1])

P=periode(a[0],m[0])
aire_ellipse=calcul_aire_ellipse(a[0],e[0])
aire_mois=calcul_aire_mois(aire_ellipse,P)
liste_points=gen_liste_points(asonde,esonde,aire_mois)
plt.polar(theta,rayon(asonde,esonde,theta))
ax.scatter(liste_points, rayon(asonde,esonde,liste_points))

plt.show()
```

12 A l'aide d'une lecture sur le graphique, déterminer la durée du trajet de la sonde. (Donner la réponse à l'aide d'un `print("Réponse: ")`).

Solution 9.

```
print("On trouve une durée entre 7 et 8 mois.")
print("Cela correspond approximativement aux dates données dans l'énoncé.")
```