Activité numérique

python

COMPÉTENCES :
REA Utiliser un modèle
APP Schématiser une situation

Orbite de Mercure

Dans cette activité, on se propose de vérifier la première loi de Kepler en utilisant un simulateur d'orbite créé à l'aide d'un langage de programmation.

DOC 1 Simulateur d'orbite

Le programme modélise l'orbite de Mercure dont la position est repérée à intervalle de temps constant. Il est adaptable à d'autres planètes du système solaire.

N correspond au nombre de positions mesurées.

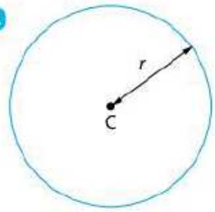
PYTHON

```
import matplotlib.pyplot as plt
import math as m
import scipy.optimize as op
#déclaration des listes
t,M,u,theta,R,X,Y = [],[],[],[],[],[],[]
# Données d'astronomie
T_rev = 0.240 #période de révolution (an)
e = 0.206 # excentricité
a = 0.387# demi grand axe en (UA)
N = 40 # Nombre de position
# résolution des équations de Kepler
for i in range(N):
    t.append(i*T_rev /N)
    M.append(2*m.pi/T_rev *t[i])
    u.append(float(op.fsolve (lambda x:x-e*m.
  sin(x)-M[i],0) ))
# Calcul des coordonnées
    theta.append(2*m.atan((m.sqrt((1+e)/
  (1-e))*m.tan(u[i]/2))))
    R.append(a*(1-e**2)/(1+e*m.cos(theta[i])))
    X.append(R[i]*m.cos(theta[i]))
    Y.append(R[i]*m.sin(theta[i]))
#affichage de l'orbite
plt.grid(True)
plt.xlabel("distance (U.A)")
plt.ylabel("distance (U.A)")
plt.axis('equal')
plt.plot(X,Y,"bo")
plt.plot(0,0,"go")
plt.show()
```

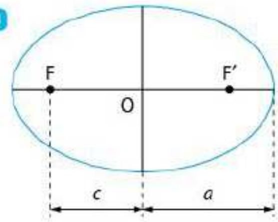
DOC 2 Caractéristiques d'une orbite elliptique

Les orbites des planètes autour du Soleil ne sont pas des cercles, mais des ellipses. Un cercle est caractérisé par son centre C et son rayon r (A), tandis qu'une ellipse est caractérisée par son centre O, son demi-grand axe a, ses deux foyers F et F', et son **excentricité** $e = c/a$, où $c = OF = OF'$ (B). Le Soleil occupe l'un des deux foyers de cette ellipse. La position de la planète au plus proche du Soleil est appelée **périhélie** et l'**aphélie** le point le plus lointain.

A



B



DOC 3 Données planétaires

Planète	a (U.A)	T (an)	e
Mercure	0,387	0,240	0,206
Vénus	0,723	0,615	0,007
Terre	1,00	1,00	0,017
Mars	1,52	1,88	0,093
Jupiter	5,20	11,9	0,048
Saturne	9,54	29,4	0,054
Uranus	19,2	84,0	0,046
Neptune	30,0	165	0,010

U.A : l'unité astronomique, elle correspond à la distance moyenne entre la Terre et le soleil soit 150 millions de km.

EXPLOITATION ET ANALYSE

1 Lancer le programme pour la planète Mercure.

a. Dans quel référentiel considéré galiléen est représentée son orbite ?

b. À quelle distance du Soleil se trouve l'aphélie de Mercure ? son périhélie ?

c. En utilisant la représentation obtenue par la simulation, déterminer la vitesse à l'aphélie et périhélie de Mercure.

2 En faisant évoluer le programme, on peut identifier les planètes dont l'orbite est proche d'un cercle. Pour cela, il suffit de changer les données (valeur demi-grand axe a, excentricité, période de révolution) des différentes planètes dans le code. On peut aussi modifier le nombre N de positions affichées.

a. Dans ce cas, de quelle valeur est proche l'excentricité e ?

b. Donner les noms des planètes concernées.

c. Que peut-on dire de la vitesse de ces planètes ?

CONCLUSION

3 Représenter sur l'annexe quelques vecteurs vitesses et accélérations de la planète étudiée :

– dans le cas de Mercure ;

– dans le cas d'une planète ayant une orbite quasi circulaire.

Je réussis si...

- **Je sais repérer les orbites elliptiques et leurs caractéristiques.**
- **Je sais mettre en évidence la variation de vitesse sur une orbite elliptique.**

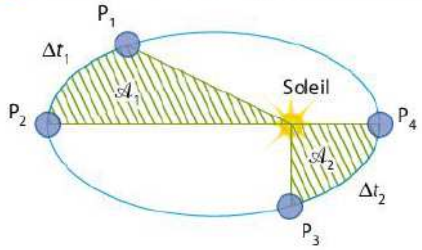
Activité numérique

COMPÉTENCES :
RÉA Mettre en œuvre les étapes d'une démarche
VAL Confronter un modèle à des résultats expérimentaux

La comète de Halley

La deuxième loi de Kepler est complexe à vérifier par l'expérience, aussi l'outil informatique permet de modéliser le calcul de l'aire balayée par une trajectoire comme dans l'étude de la comète de Halley.

DOC 1 Deuxième loi de Kepler

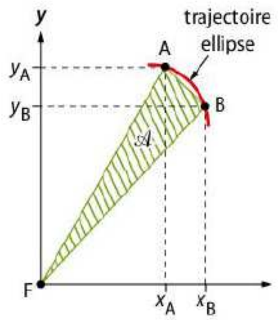


On appelle $\mathscr{A}_1$ et $\mathscr{A}_2$ des aires balayées par le rayon vecteur partant du Soleil vers la planète durant les intervalles de temps Δt_1 et Δt_2.
D'après la 2e loi de Kepler (loi des aires) : si les durées de parcours sont identiques $\Delta t_1 = \Delta t_2$ alors $\mathscr{A}_1 = \mathscr{A}_2$.

DOC 2 Principe de calcul d'une aire balayée

Pour calculer l'aire balayée, on estime l'aire d'un triangle. Cette approximation reste correcte si les instants considérés sont proches.
L'aire du triangle $\mathscr{A}$ est calculée en utilisant le demi-périmètre :

$$p = \frac{AF + AB + FB}{2}$$



et la formule de Héron telle que :

$$\mathscr{A} = \sqrt{p(p - AF)(p - AB)(p - FB)}$$

avec AF, AB et FB les longueurs du triangle.

DOC 3 Programme de calcul des aires balayées

La portion de programme du calcul des aires balayées est à insérer dans les lignes de codes du programme de l'activité précédente (qui permet de représenter à l'aide d'un langage de programmation l'orbite d'une planète), juste avant la partie correspondant l'affichage de l'orbite. Les intervalles de temps sont modifiables en changeant les dates de mesures.

```
# Aires des triangles
# t1,t2 dates première aire et t2, t3 date de la seconde aire
t1,t2 = 0,2
t3,t4 = 10,12
#calcul de l'aire balayée entre t0 et t1
Delta_t1 =t2-t1
AIRE1,AIRE2 = 0,0
i1,i2 = 0,0
for i1 in range(Delta_t1):
  long1 = m.sqrt((X[t1+i1])**2+(Y[t1+i1])**2)
  long2 = m.sqrt((X[t2+i1])**2+(Y[t2+i1])**2)
  long3 = m.sqrt((X[t2+i1]-X[t1+i1])**2+(Y[t2+i1]-Y[t1+i1])**2)
  S_1 = 1/2*(long1+long2+long3)
  AIRE1 = m.sqrt(S_1*(S_1-long1)*(S_1-long2)*(S_1-long3))+AIRE1
#calcul de l'aire balayée entre t2 et t3
Delta_t2 =t4-t3
for i2 in range(Delta_t2):
  long1b = m.sqrt((X[t3+i2])**2+(Y[t3+i2])**2)
  long2b = m.sqrt((X[t4+i2])**2+(Y[t4+i2])**2)
  long3b =m.sqrt((X[t4+i2]-X[t3+i2])**2+(Y[t4+i2]-Y[t3+i2])**2)
  S_1b = 1/2*(long1b+long2b+long3b)
  AIRE2 = m.sqrt(S_1b*(S_1b-long1b)*(S_1b-long2b)*(S_1b-long3b))
  +AIRE2

print('aire balayée entre t1 et t2 --> ' + str(AIRE1))
print('aire balayée entre t3 et t4 --> ' + str(AIRE2))
```

DOC 4 La comète de Halley

La comète de Halley possède une période de révolution de 76,09 années autour du Soleil, son excentricité est de 0,967 et la valeur du demi-grand axe est de 17,83 UA.

EXPLOITATION ET ANALYSE

1 Repérer dans le programme les parties concernant : le calcul de la longueur de chaque segment, le calcul du demi-périmètre et la formule de Héron.

2 Vérifier pour plusieurs planètes (en utilisant les données planétaires du **doc. 3** de l'activité 2) et différentes dates, que la 2e loi de Kepler est vérifiée.

3 **a.** Pour une comète comme celle de Halley, les variations de vitesses observées sur son orbite sont-elles compatibles avec la 2e loi de Kepler ?
b. Justifier que la mesure des aires balayées est fortement imprécise dans le cas de la comète de Halley. Montrer, en justifiant, que l'on peut résoudre ce problème en modifiant certains paramètres du programme.

CONCLUSION

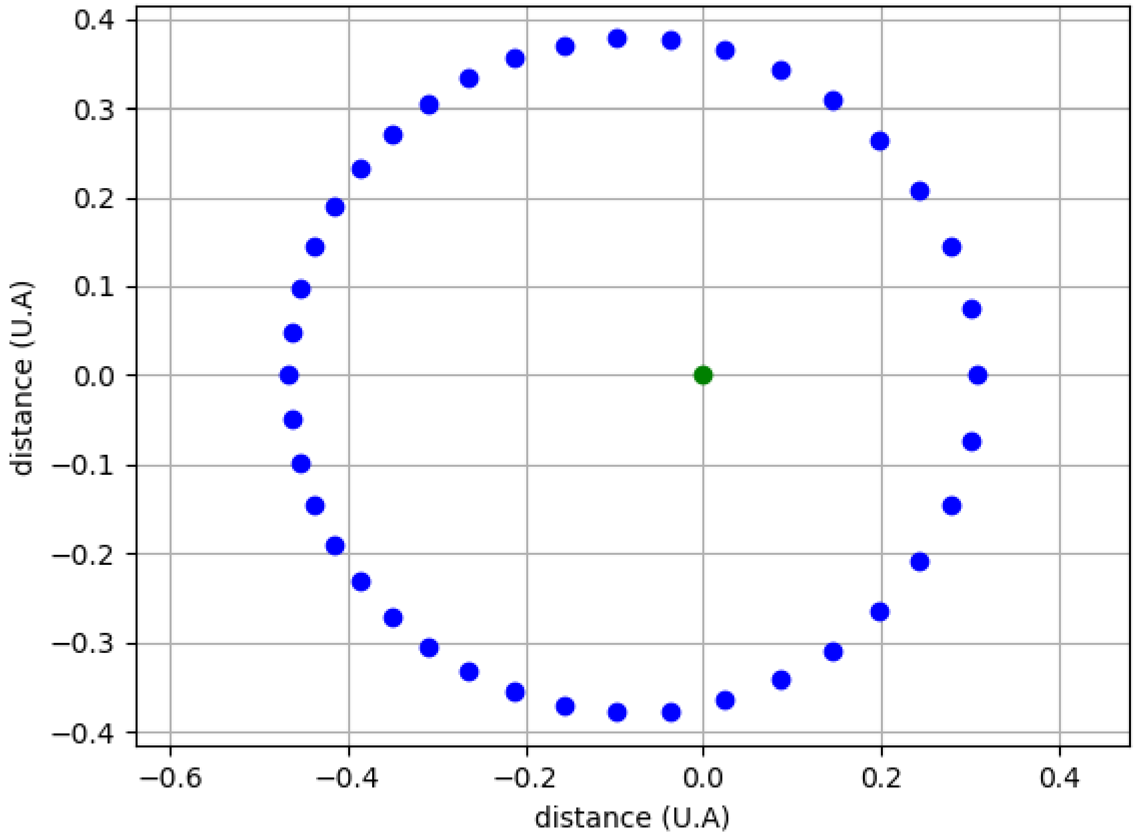
4 La deuxième loi de Kepler est-elle applicable à tous les astres ?

Je réussis si…

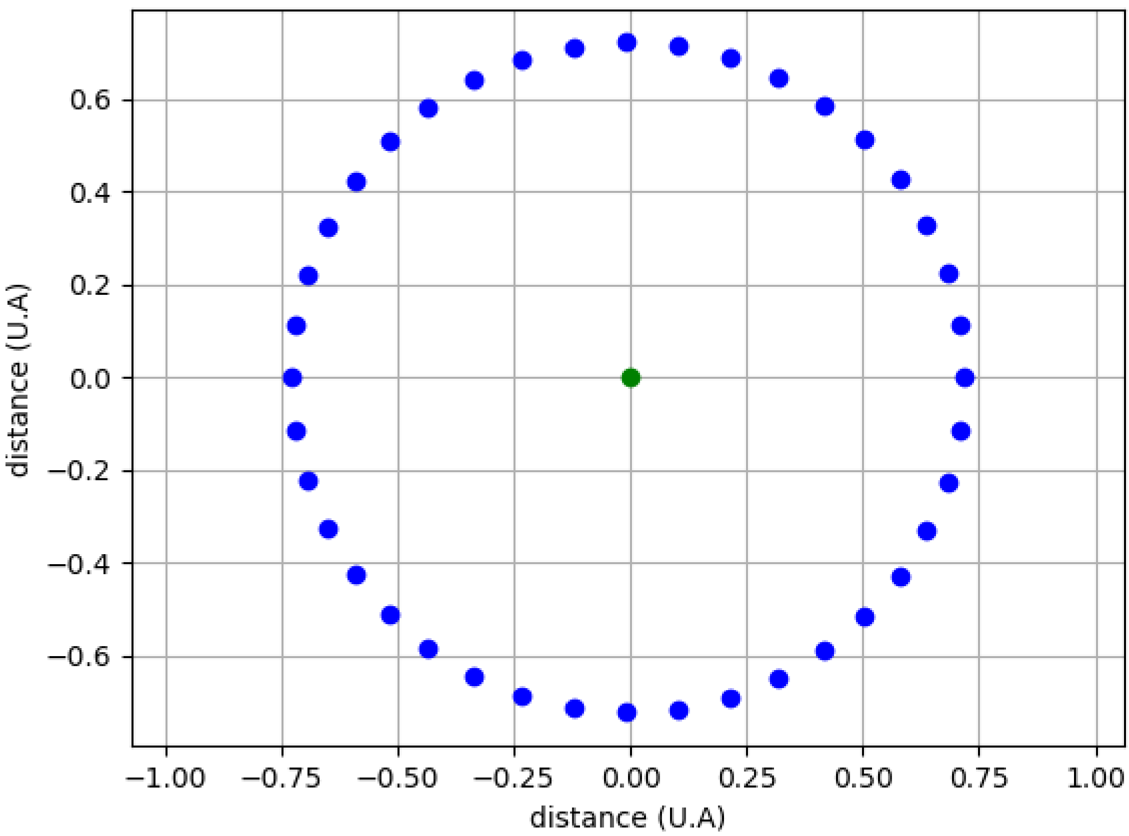
- **Je sais repérer les expressions mathématiques dans le code.**
- **Je sais vérifier la deuxième loi de Kepler à l'aide du programme.**
- **Je sais reconnaître les limites de l'approximation.**

ANNEXE :

Orbite de Mercure autour du Soleil



Orbite de Venus autour du Soleil



Activité numérique

python

COMPÉTENCES :
REA Utiliser un modèle
ANA/REA Faire des prévisions à l'aide d'un modèle

La masse de Jupiter

La loi des périodes ou troisième loi de Kepler s'applique non seulement au système solaire mais aussi aux systèmes satellitaires.

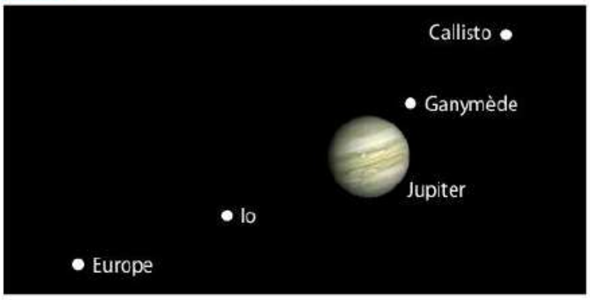
DOC 1 Expression de la 3e loi de Kepler

Après l'énoncé de la loi des périodes par Johannes Kepler, Isaac Newton la démontre et l'écrit ainsi :

$$\frac{T^2}{a^3} = \frac{4\pi^2}{G \cdot M} = \text{constante.}$$

Où T est la période de révolution de l'astre sur son orbite, a la valeur du demi-grand axe de l'orbite, G est la constante de gravitation universelle : $G = 6{,}67 \times 10^{-11}\ \text{m}^3 \cdot \text{kg}^{-1} \cdot \text{s}^{-2}$ et M la masse, en kg, de l'astre attracteur (le Soleil dans le cas du système solaire).

DOC 2 Jupiter et ses satellites



Jupiter est la planète la plus grande et massive du système solaire ; sa masse est de $1{,}898 \times 10^{27}$ kg. Jupiter possède de nombreux satellites ; quelques-uns de ses satellites ainsi que leurs caractéristiques (demi-grand axe, excentricité et période de révolution) sont donnés ci-dessous.

Planètes	a (× 10³ km)	e	T (jour)
Amalthée	181	0,0031	0,498
Thébé	221	0,0177	0,674
Io	421	0,0041	1,769
Europe	671	0,0094	3,551
Ganymède	1 070	0,0011	7,155
Callisto	1 882	0,0074	16,689

DOC 3 Régression linéaire

La régression linéaire est une méthode statistique permettant de trouver l'équation d'une fonction affine à partir d'un nuage de points issu de mesures. Le modèle est validé, si le coefficient de corrélation est proche de 1 ou –1, 0,999... est une valeur convenable.

En langage Python, l'instruction (`stat.linregress()`) permet de trouver le coefficient directeur, l'ordonnée à l'origine et le coefficient de corrélation, valeurs stockées dans un tableau de trois valeurs.

DOC 4 Programme « vérification de la 3e loi de Kepler »

Le programme suivant permet de vérifier la 3e loi de Kepler pour le système solaire. Il est adaptable à d'autres systèmes notamment les systèmes satellitaires.

```
import matplotlib.pyplot as plt
import scipy.stats as stat
# listes des demi-grands axes a en U.A et périodes
  des planètes en années
LIST_a = [0.39,0.72,1,1.5,5.2,9.5,19.2,30]
LIST_T = [0.24,0.615,1,1.88,11.9,29.4,84,165.0]
LIST_P = ['Mercure','Venus','Terre','Mars',
  'Jupiter','Saturne','Uranus','Neptune']
# a au cube et T au carré
for i in range(0, len(LIST_a)):
    LIST_a[i] = (LIST_a[i]*150e9)**3
    LIST_T[i] = (LIST_T[i]*365*24*3600)**2
# régression linéaire
regression = stat.linregress(LIST_a,LIST_T)
pente =regression[0]
print('pente --> '+str(pente))
ordorigine =regression[1]
print('ordonnée à l origine --> '+str(ordorigine))
coeffcorel = regression[2]
print('coefficient de corrélation -->
  '+str(coeffcorel))
#affichage point et droite de regression
a_3_max = LIST_a[len(LIST_a)-1]
T_2_max =pente *a_3_max + ordorigine
plt.grid(True)
plt.xlabel(' a au cube (m3)')
plt.ylabel('periode au carré (s2)')
plt.scatter(LIST_a,LIST_T,s = 100,c ='red',
  marker ='+')
for i in range (0,len(LIST_a)):
  plt.text(LIST_a[i],LIST_T[i], LIST_P[i],fontsize =8)
plt.plot ([0,a_3_max],[ordorigine,T_2_max],c ='blue')
plt.show()
```

QUESTIONS PRÉLIMINAIRES

1 Lancer le programme pour le système solaire.

2 Tester la 3e loi de Kepler en vérifiant la proportionnalité entre T^2 et a^3.

PROBLÈME À RÉSOUDRE

3 **a.** Adapter le programme aux satellites de Jupiter.

b. Mettre en œuvre une stratégie pour déterminer la masse de Jupiter.

c. Comparer la valeur de la masse trouvée par cette stratégie à celle donnée dans le **document 2**.

Je réussis si...

- **je sais vérifier la troisième loi de Kepler pour le système solaire.**
- **Je sais adapter le programme pour le système satellitaire de Jupiter**
- **Je sais déterminer la valeur de la masse de Jupiter.**

Correction activité 1 :

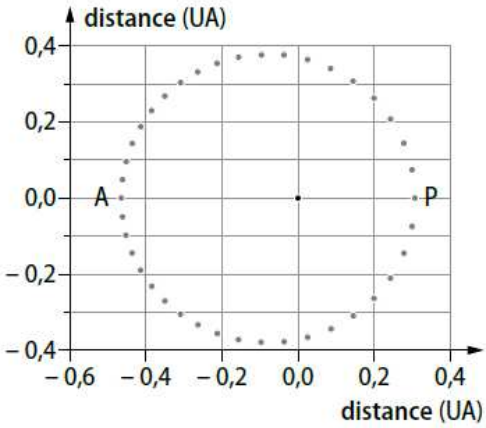
Orbite de Mercure

Commentaires pédagogiques et compléments expérimentaux

Dans cette activité, on exploite l'outil numérique pour vérifier la première loi de Kepler. Cette activité utilise un programme Python qui génère les positions d'une planète à intervalles constants. Ce programme peut être modifié pour obtenir toutes les orbites des planètes possibles. Il permet également de tracer une trajectoire de comète, mais aussi d'exoplanète. Il faut connaître les paramètres des astres : période de révolution, excentricité et valeur du demi-grand axe. On peut aussi augmenter ou diminuer le nombre de positions sur l'orbite.

› Exploitation et analyse

1. a. On obtient la représentation suivante :



On observe que la position du Soleil n'est pas placée au centre de l'orbite. Il s'agit bien d'une ellipse. Le référentiel d'étude est le référentiel héliocentrique.

b. L'aphélie (point A sur la figure ci-dessus) se situe à une distance du Soleil de :

0,467 U.A. = 7,00 × 10^7 km

Le périhélie (point P sur la figure ci-dessus) se situe à une distance du Soleil de :

0,307 U.A. = 4,61 × 10^7 km

c. Dans le programme, on affiche $N = 40$ positions, soit une position tous les $\frac{0,240}{40} = 0,0060$ an, soit $0,006 \times 365,25 = 2,2$ jours. Il y a donc un point tous les 2,2 jours. En prenant le point avant et le point après, on détermine la vitesse aux points A et P.

Au périhélie :

$$v_P = \frac{(0,074 + 0,074) \times 150 \times 10^6}{2 \times 2,2 \times 24 \times 3\,600} = 59 \text{ km} \cdot \text{s}^{-1}.$$

À l'aphélie : $v_A = \frac{(0,049 \times 2) \times 150 \times 10^6}{2 \times 2,2 \times 24 \times 3\,600} = 39 \text{ km} \cdot \text{s}^{-1}$.

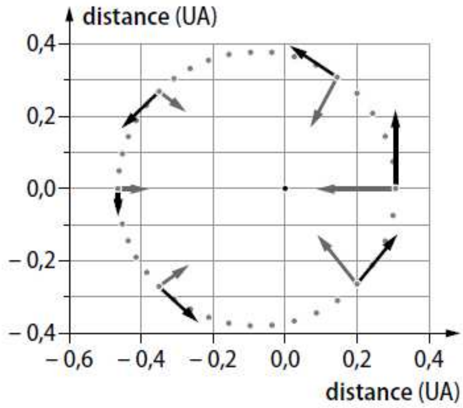
2. a. Plus l'orbite est proche d'un cercle, plus la valeur de e est proche de 0.

b. En utilisant le programme, on voit que les planètes Venus, Neptune et même la Terre ont des orbites proches d'un cercle. En effet, le Soleil est placé quasiment au centre de leurs orbites.

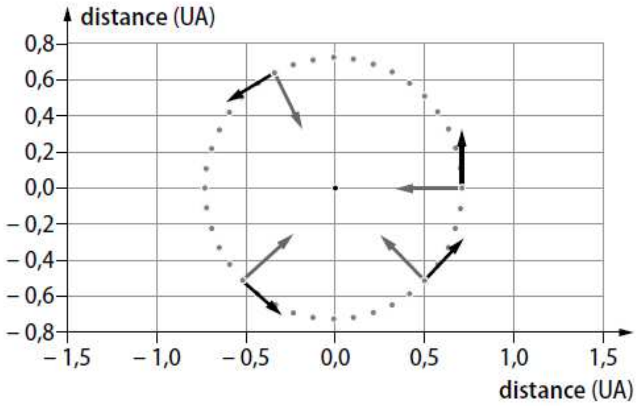
c. On note que les espaces entre les différentes positions à des intervalles de temps identiques sont les mêmes. La vitesse peut donc être considérée comme quasiment constante.

› Conclusion

3. Représentation de quelques vecteurs vitesse et accélération dans le cas de Mercure :



Représentation de quelques vecteurs vitesse et accélération dans le cas de Venus, qui a l'orbite la plus circulaire du système solaire :



La comète de Halley

Commentaires pédagogiques et compléments expérimentaux

On utilise le programme de l'activité précédente et on ajoute une partie de code permettant le calcul d'une aire approchée. On vérifie ainsi la deuxième loi de Kepler. Cette loi est complexe à vérifier numériquement de façon précise : il est nécessaire d'appliquer certaines précautions. Il faut ainsi prendre un nombre de points important si l'orbite de l'astre est très elliptique.

› Exploitation et analyse

1. • Partie concernant le calcul de la longueur de chaque segment :

```
long1=m.sqrt((X[t1+i1])**2+(Y[t1+i1])**2)
long2=m.sqrt((X[t2+i1])**2+(Y[t2+i1])**2)
long3=m.sqrt((X[t2+i1]-
X[t1+i1])**2+(Y[t2+i1]-Y[t1+i1])**2)
```

• Partie concernant le calcul du demi-périmètre :

```
S_1=1/2*(long1+long2+long3)
```

• La formule de Héron est appliquée à la ligne de programme suivante :

```
AIRE1=m.sqrt(S_1*(S_1-long1)*(S_1-long2)*(S_1
-long3))+AIRE1
```

On notera que l'on additionne l'aire calculée par la méthode de Héron et l'aire précédemment calculée.

2. Dans l'ensemble des mesures présentées ci-après, les intervalles de temps ont été pris pour des positions identiques, entre les positions 0 et 5 pour l'aire entre t0 et t1, puis entre 50 et 55 pour l'aire entre t2 et t3.

L'unité des axes étant en UA, l'aire est donc en UA^2.

• Pour Mercure, en prenant 100 positions et deux intervalles de temps, on obtient :
Aire balayée entre t0 et t1 : 0,11155433605408169
Aire balayée entre t2 et t3 : 0,11401120231050339

• Pour Venus, dans les mêmes conditions :
Aire balayée entre t0 et t1 : 0,40368477611327347
Aire balayée entre t2 et t3 : 0,403817028432449

• Pour la Terre, dans les mêmes conditions :
Aire balayée entre t0 et t1 : 0,7717888982012697
Aire balayée entre t2 et t3 : 0,7730346099152757

• Pour Mars, dans les mêmes conditions :
Aire balayée entre t0 et t1 : 1,7688090548920724
Aire balayée entre t2 et t3 : 1,783820599781274

• Pour Jupiter, dans les mêmes conditions :
Aire balayée entre t0 et t1 : 20,813598293568564
Aire balayée entre t2 et t3 : 20,909093196386607

• Pour Saturne, dans les mêmes conditions :
Aire balayée entre t0 et t1 : 70,0091389292504
Aire balayée entre t2 et t3 : 70,3711799264318

• Pour Uranus, dans les mêmes conditions :
Aire balayée entre t0 et t1 : 283,8133754601152
Aire balayée entre t2 et t3 : 285,0605526305586

• Pour Neptune dans les mêmes conditions :
Aire balayée entre t0 et t1 : 694,9179063309426
Aire balayée entre t2 et t3 : 695,5771493475195

On remarque que les aires balayées sont très proches pour des durées égales. La deuxième loi de Kepler est vérifiée.

3. a. La deuxième loi est compatible car, pour des temps identiques, les aires doivent être égales, ce qui signifie que la vitesse de la comète lorsqu'elle est proche du Soleil est beaucoup plus importante que lorsqu'elle est plus éloignée, car la comète doit alors parcourir une distance plus grande pour balayer une aire identique.

b. Lorsqu'on fait une simulation avec l'orbite de la comète de Halley, on trouve des valeurs qui sont très éloignées. La trajectoire étant fortement excentrée, certains points de l'orbite peuvent se trouver très éloignés les uns des autres, ce qui entraîne une grande erreur dans le calcul des aires :
Aire balayée entre t0 et t1 : `7.040573468436815`
Aire balayée entre t2 et t3 : `12.696028451700366`

Pour améliorer ce calcul numérique, on augmente le nombre de points.

• Pour 1 000 positions :
Aire balayée entre t0 et t1 : `0.8304342674325653`
Aire balayée entre t2 et t3 : `1.0160632419184221`

• Pour 10 000 positions :
Aire balayée entre t0 et t1 : `0.10106850471309686`
Aire balayée entre t2 et t3 : `0.1015652295612906`

• Pour 100 000 positions :
Aire balayée entre t0 et t1 : `0.010177471791621585`
Aire balayée entre t2 et t3 : `0.010177486936778179`

On voit que le nombre de points permet d'améliorer les résultats.

› Conclusion

4. On voit que la deuxième loi de Kepler s'applique à toutes les planètes, et même à la comète de Halley, dès lors que où leur trajectoire est une orbite elliptique.

<u>Correction activité 2 :</u>

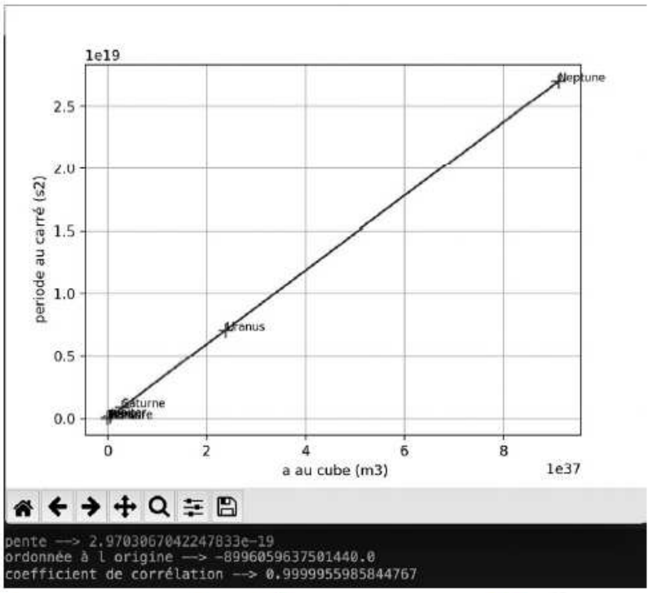
La masse de Jupiter

Commentaires pédagogiques

Cette activité propose de vérifier la troisième loi de Kepler à travers un programme Python pour le système solaire. Ce programme est adaptable à tous les types de systèmes satellitaires, mais aussi d'étoile avec exoplanètes, comme on cherche à le montrer à travers cette activité. Une partie « régression linéaire » permet de montrer la validité numérique du modèle.

› Questions préliminaires

1. On obtient le résultat suivant :



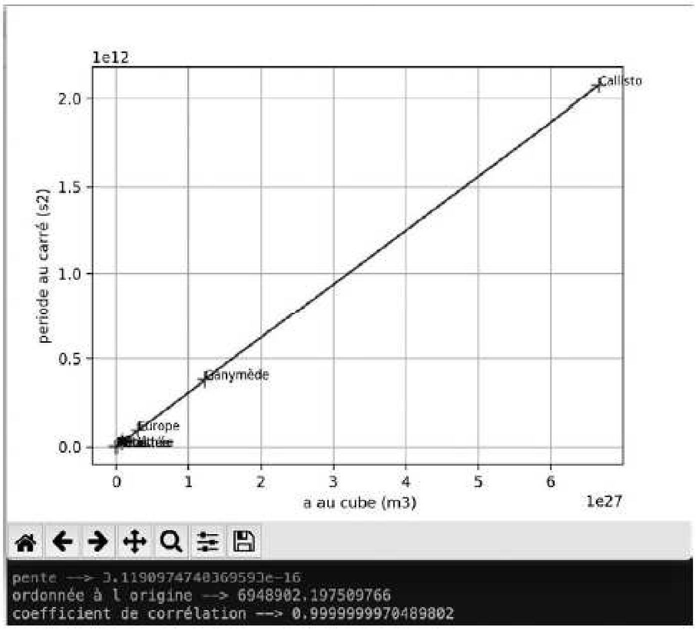
2. On s'aperçoit que T^2 est proportionnel à a^3.
$T^2 = 2{,}970 \times 10^{-19} \times a^3$. Le modèle est valide car le coefficient de corrélation est très proche de 1.

› Problème à résoudre

3. a. Voici la portion du programme à modifier :

```
LIST_a=[181,221,421,671,1070,1882]
LIST_T=[0.498,0.674,1.769,3.551,7.155,16.689]
LIST_P=['Amalthée','Thébé','Io','Eu-
rope','Ganymède','Callisto']
# a au cube et T au carré
for i in range(0, len(LIST_a)):
  LIST_a[i]=(LIST_a[i]*1e6)**3
  LIST_T[i]=(LIST_T[i]*24*3600)**2
```

Voici le graphe de T^2 en fonction de a^3 du système satellitaire de Jupiter :



b. On a : $\dfrac{T^2}{a^3} = \dfrac{4\pi^2}{G \cdot M} = \text{constante}.$

Donc $\dfrac{4\pi^2}{G \cdot M} = 3{,}119 \times 10^{-16}\ s^2 \cdot m^{-3}$

Ainsi $M = \dfrac{4\pi^2}{6{,}67 \times 10^{-11} \times 3{,}119 \times 10^{-16}}$
$= 1{,}90 \times 10^{27}$ kg

c. La masse trouvée est très proche la valeur indiquée dans le document 2.