

PROGRAMMER EN PYTHON

FICHE N°9 : CONSTANCE DES AIRES ET MERCURE

L'étude du mouvement de Mercure, menée dans la fiche n°9 pour la classe de seconde et dans la fiche n°4 pour la classe de première, est complétée pour illustrer la propriété de la « constante des aires ».

Capacité numérique mise en œuvre

Utiliser un langage de programmation pour étudier la relation approchée entre la variation du vecteur vitesse d'un système modélisé par un point matériel entre deux instants voisins et la somme des forces appliquées sur celui-ci.

Le script débute comme celui de la fiche n°4 pour la classe de première.

Dans un deuxième temps, le script calcule, pour tous les points de mesure, la « constante » des aires, obtenue grâce à son expression en coordonnées cartésiennes : $C = x \cdot v_y - y \cdot v_x$.

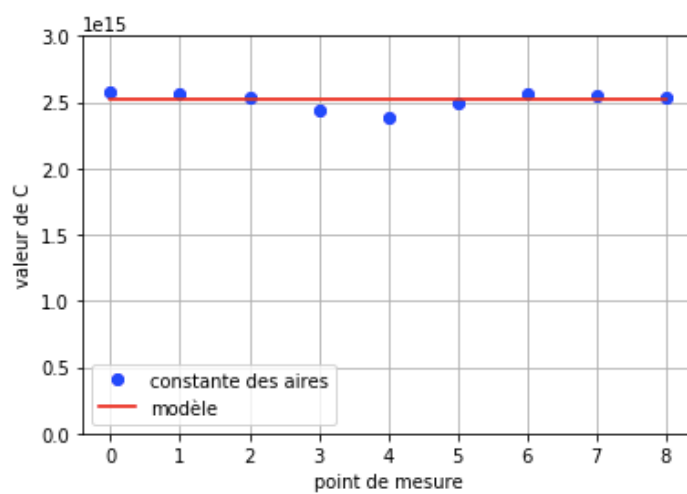
Comme dans la fiche n°4 pour la classe première, il faut prendre garde à bien faire correspondre les valeurs de distance et de vitesse calculée (il faut ainsi décaler la liste des distances).

On peut ensuite faire modéliser la fonction précédente par une constante.

```
dx=x[:-1]*1.5*10**11
dy=y[:-1]*1.5*10**11

vyl=1.5*10**11*np.array(vy)/(24*3600)
vxl=1.5*10**11*np.array(vx)/(24*3600)
dxn=dx[1:]
dyn=dy[1:]

Cs=(dxn*vyl-dyn*vxl)
n=np.arange(len(t)-2)
plt.plot(n,Cs,'bo',label='constante des aires')
plt.grid()
plt.xlabel('point de mesure')
plt.ylabel('valeur de C')
mod=np.polyfit(n,Cs,0)
Cmods=float(mod[0])
print('la constante des aires est de l ordre de ', "%.1e"%Cmods)
Cmodgs=0*n+mod[0]
plt.plot(n,Cmodgs,'r-',label="modèle")
plt.ylim(0,3*10**15)
plt.legend()
```



Conclusion

On illustre bien ici la constance des aires balayées.