TD numérique Tipe n°3 – Mouvements à force centrale

Description du TD

Le but du TD est d'étudier les trajectoires obtenues pour différentes forces centrales, typiquement l'interaction gravitationnelle. On pourra se contenter du l'appel au solveur, ou bien tester encore les capacités de la méthode d'Euler pour comparer (dans une situation connue, donc avec la gravitation).

Dans tout le TD, on ramène les constantes à 1 (constante gravitationnelle, masse d'un astre central, longueur à vide et constante de raideur d'un ressort, etc.).

Exigibles numériques

- Des TD précédents :
  - appel au solveur Python pour un système d'équations
  - récupération des données
  - tracé d'une courbe légendée en fonction du temps
- Nouveautés : voir l'aide sur ces deux derniers points en fin de document.
  - Affichage multi graphique
  - Création d'un graphique en coordonnées polaires

Théorie

Q1. Rappeler la définition polaire de la constante des aires C, et en déduire son expression en fonction des conditions initiales r_0, v_0, α, où α est l'angle entre le vecteur position et le vecteur vitesse, à l'état initial.

Q2. Rappeler dans le cas de la gravitation (ou obtenir à nouveau : voir le cours) le système des équations différentielles polaires du mouvement, obtenues avec la RFD, où l'on fait intervenir C.

L'écrire en fonction de champ gravitationnel $g(r)$ tel que la force gravitationnelle soit définie par $\vec{F}_g = -m\,g(r)\vec{e}_r$, m étant la masse du système.

On remarquera le signe choisi ici, de telle sorte que g soit positif.

Q3. Le vecteur argument `Y` de la fonction fondamentale du système d'équations `F:(Y,t)→Y'` envoyée au solveur Python `odeint` peut être choisi comme $Y = (r, \mathring{r}, \theta)$.

En effet, donner l'expression de `Y'`, c'est-à-dire de chacune de ses coordonnées, en fonction de celles du vecteur `Y`, de C et de g.

Q4. Quelle est l'expression de la norme v de la vitesse en fonction de $r, \mathring{r}$ et C ?

Implémentation

On souhaite obtenir un graphique multiple contenant, sur un petit nombre de périodes :

- l'allure de la trajectoire
- la fonction r du temps
- la fonction θ du temps
- la fonction v du temps

Q5. Définir la fonction de r appelée `champGrav`, qui correspond au codage du champ gravitationnel g.

Q6. Définir la variable globale `champEtud` égale pour l'instant à `champGrav`

On modifiera par la suite cette variable globale à cet endroit du code : une nouvelle exécution devra donner le résultat correct en changeant simplement le champ (nom de fonction) donc la force étudiée.

Q7. Coder la fonction fondamentale de résolution `F` en appelant évidemment `champEtud`

Q8. Définir comme variables globales : le nombre de dates souhaitées (quelques centaines typiquement), une date maximale (un peu au hasard : comme tout est ramené à 1, de l'ordre de quelques unités), puis la liste de dates correspondant à ce choix, à construire.

Q9. Définir des conditions initiales pratiques (cf **Q1**) quelconques comme variables globales (un angle en degré est plus parlant, mais attention alors à la conversion nécessaire...).
En déduire l'expression du vecteur `Y(0)` (on peut ou non définir une variable pour lui).

On testera ultérieurement différentes conditions initiales de vitesse (comme les constantes sont prises égales à l'unité, $r_0=1$ ou proche conviendra toujours), et on ajustera la date finale en fonction des premiers résultats obtenus.

On peut prendre au début un angle α de $90°=\pi/2$, mais ce n'est pas général : il faudra tester d'autres valeurs.

Q10. Appeler la résolution du système d'équations, en conservant le résultat dans une variable globale au nom intelligemment choisi (compréhensible).

Vérifier dans la console son contenu (structure de listes, longueur) avant de se lancer dans les graphiques, et récupérer les données sous forme de quelques listes comme au TD précédent.

Q11. Tester d'abord un graphique simple du type $r(t)$ ou $\theta(t)$.

Q12. Implémenter le multi graphique souhaité.

On change maintenant la nature de la force.

Q13. Définir le champ harmonique `champHarm`, de telle sorte que la force soit proportionnelle et opposée au vecteur position (attention au signe choisi à la **Q2**), ce qui correspond à la force exercée par un ressort linéaire de longueur à vide nulle.

Tester en modifiant simplement la variable globale `champEtud`

(On sera peut être amené à modifier la date finale pour obtenir des résultats satisfaisants du point de vue esthétique.)

Q14. Définir le champ `champRessort`, associé à la force de rappel élastique d'un ressort, avec une longueur à vide égale à 1 (on pourra alors choisir éventuellement $r(0)\neq 1$). Tester.

Q15. Définir le champ `champ1SurR3`, associé à une force de rappel proportionnelle à $\frac{1}{r^3}$.

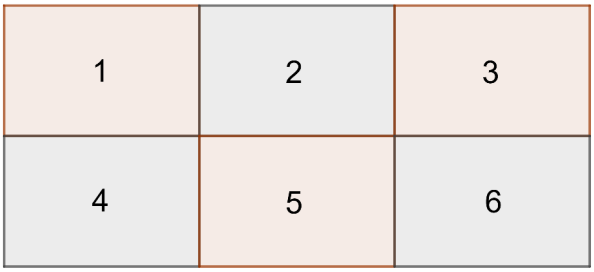
Tester.

Aide pour les graphiques

1. Afficher plusieurs graphes dans la fenêtre

Une solution simple existe, lorsqu'on souhaite faire un simple pavage régulier (ou tableau) de plusieurs graphiques.

Par exemple, pour 2 lignes et 3 colonnes :



On crée chaque graphique, numéroté comme ci-contre, l'un après l'autre, grâce à la commande `plt.subplot`

Premier graphique : `plt.subplot(231)`, puis toutes les instructions habituelles (`plt.plot`, `plt.hlines`, etc.), mais sans appeler `plt.show`

Dans « 231 », le 2 signifie « 2 lignes », le 3 « 3 colonnes », et le 1 que l'on commence à créer le graphique n°1.

Deuxième graphique : Initialisation avec `plt.subplot(232)`, etc.

Après la construction du sixième graphique : `plt.show()`

Si l'un des `subplot` n'est pas utilisé, aucun problème : il y aura une case vide.

2. Tracer une courbe en coordonnées polaires

`plt.polar(`*`θlist`*`,`*`rlist`*`)` remplace `plt.plot(`*`tlist`*`,x`*`list`*`)`, et il faut alors modifier l'argument optionnel `polar` à l'appel du `subplot` correspondant à ce graphique en polaires : `plt.subplot(`*`3digitIntStr`*`,polar=True)`.