TP n° 13 : Méthodes d'Euler et de Verlet

Modéliser les interactions physiques entre un grand nombre de constituants mène à l'écriture de systèmes d'équations différentielles pour lesquels, en dehors de quelques situations particulières, il n'existe aucune solution analytique. Les problèmes de dynamique gravitationnelle et de dynamique moléculaire en sont deux exemples. Afin d'analyser le comportement temporel de tels systèmes, l'informatique peut apporter une aide substantielle en permettant leur simulation numérique. L'objet de ce sujet, composé de trois parties, est l'étude de solutions algorithmiques en vue de simuler une dynamique gravitationnelle afin, par exemple, de prédire une éclipse ou le passage d'une comète.

Les programmes doivent être écrits en langage Python. Les candidats peuvent utiliser librement les fonctions de la bibliothèque `math`. *La lisibilité des codes produits est un élément important d'appréciation.*

A. Quelques fonctions utilitaires

1. Donner la valeur des expressions Python suivantes :

   1.a) `[1, 2, 3] + [4, 5, 6]`

   1.b) `2 * [1, 2, 3]`

2. Écrire une fonction `smul` à deux paramètres, un nombre entier `n` et une liste de nombres `L`, qui renvoie une nouvelle liste constituée des éléments de la liste `L` multipliés par le nombre `n`. Par exemple `smul(2, [1,2,3])` renvoie `[2,4,6]`.
3. Écrire une fonction `vsom` qui prend en paramètres deux listes de nombres `L1` et `L2` de même longueur et qui renvoie une nouvelle liste constituée de la somme terme à terme de ces deux listes. Par exemple `vsom([1,2,3], [4,5,6])` renvoie `[5,7,9]`.
4. Écrire une fonction `vdif` qui prend en paramètres deux listes de nombres `L1` et `L2` de même longueur et qui renvoie une nouvelle liste constituée de la différence terme à terme de ces deux listes (la première moins la deuxième). Par exemple `vdif([1,2,3], [4,5,6])` renvoie `[-3,-3,-3]`.

Ces trois fonctions pourront être librement réutilisées dans la suite du TP, notamment la partie C.

B. Étude de schémas numériques

Soient y une fonction deux fois dérivable sur $\mathbb{R}$ avec y'' continue et $t_{\min}$ et $t_{\max}$ deux réels tels que $t_{\min} < t_{\max}$. On note I l'intervalle $[t_{\min}\,;t_{\max}]$. On s'intéresse à une équation différentielle du second ordre de la forme :

$$\forall t \in I, \quad y''(t) = f(y(t)), \tag{1}$$

où f est une fonction donnée continue sur $\mathbb{R}$. De nombreux systèmes physiques peuvent être décrits par une équation de ce type.
On suppose connues les valeurs $y_0 = y(t_{\min})$ et $z_0 = y'(t_{\min})$.

5. *Mise en forme du problème*

   Pour résoudre numériquement l'équation différentielle (1), on introduit la fonction $z : I \to \mathbb{R}$ définie par $\forall t \in I$, $z(t) = y'(t)$.

   5.a) Montrer que l'équation (1) peut se mettre sous la forme du système suivant :

$$(S) : \begin{cases} z = y' \\ z' = f(y(t)) \end{cases} .$$

5.b) Soit n un entier strictement supérieur à 1 et $J_n = [\![0\,;n-1]\!]$. On pose $h = \dfrac{t_{\max} - t_{\min}}{n-1}$ et $\forall i \in J_n$, $t_i = t_{\min} + ih$.

Montrer que, pour tout entier $i \in [\![0\,;n-2]\!]$,

$$y(t_{i+1}) = y(t_i) + \int_{t_i}^{t_{i+1}} z(t)\,\mathrm{d}t \qquad \text{et} \qquad z(t_{i+1}) = z(t_i) + \int_{t_i}^{t_{i+1}} f(y(t))\,\mathrm{d}t. \tag{2}$$

La suite du problème exploite les notations introduites dans cette partie et présente deux méthodes numériques dans lesquelles les intégrales précédentes sont remplacées par une valeur approchée.

6. *Schéma d'Euler explicite*

Dans le schéma d'Euler explicite, chaque terme sous le signe intégrale est remplacé par sa valeur prise en la borne inférieure, *i.e.* $\forall i \in [\![0\,;n-2]\!]$, $\forall t \in [t_i\,;t_{i+1}]$, $z(t) \approx z(t_i)$ et $y(t) \approx y(t_i)$.

6.a) Dans ce schéma, montrer que les équations (2) permettent de définir deux suites $(y_i)_{i\in J_n}$ et $(z_i)_{i\in J_n}$, où y_i et z_i sont des valeurs approchées de $y(t_i)$ et $z(t_i)$, vérifiant

$$\forall i \in [\![0\,;n-2]\!], \quad \begin{cases} y_{i+1} = y_i + hz_i \\ z_{i+1} = z_i + hf(y_i) \end{cases}.$$

6.b) Écrire une fonction `euler` qui reçoit en argument une fonction f, les valeurs y_0 et z_0, le pas h et le nombre de points n, et qui renvoie deux listes correspondant aux valeurs associées aux suites $(y_i)_{i\in J_n}$ et $(z_i)_{i\in J_n}$.

7. *Schéma de Verlet*

Le physicien français Loup Verlet a proposé en 1967 un schéma numérique d'intégration d'une équation de la forme (1) dans lequel, en notant $f_i = f(y_i)$ et $f_{i+1} = f(y_{i+1})$, les relations de récurrence s'écrivent

$$y_{i+1} = y_i + hz_i + \frac{h^2}{2} f_i \qquad \text{et} \qquad z_{i+1} = z_i + \frac{h}{2}(f_i + f_{i+1}). \tag{3}$$

Écrire une fonction `verlet` qui prend en argument les mêmes paramètres que la fonction `euler` de la question 6.b et qui renvoie deux listes correspondant aux valeurs associées aux suites $(y_i)_{i\in J_n}$ et $(z_i)_{i\in J_n}$ définies par les relations (3).

Le schéma numérique de Verlet a l'avantage :
- d'être plus précise que la méthode d'Euler dans le sens où l'erreur commise à pas égal est plus faible ;
- d'avoir de bonnes propriétés de conservation de l'énergie pour les systèmes conservatifs (voir le sujet original pour plus de précisions).

C. Problème à N corps

On s'intéresse à présent à la dynamique d'un système de N corps massifs en interaction gravitationnelle. Dans la suite, les corps considérés sont assimilés à des points matériels P_j de masses m_j où $j \in [\![0\,;N-1]\!]$, $N \geqslant 2$ étant un entier donné. Le mouvement de ces points est étudié dans un référentiel galiléen muni d'une base orthonormée. L'interaction entre deux corps j et k est modélisée par la force gravitationnelle. L'action exercée par le corps k sur le corps j est décrite par la force

$$\overrightarrow{F_{k/j}} = G\frac{m_j m_k}{r_{jk}^3}\overrightarrow{P_jP_k}$$

où r_{jk} est la distance séparant les corps j et k ($r_{jk} = \left\|\overrightarrow{P_jP_k}\right\|$) et $G = 6{,}67 \times 10^{-11}\ \mathrm{N\cdot m^2\cdot kg^{-2}}$ est la constante de gravitation universelle.

À tout instant t_i avec $i \in J_n$, chaque corps de masse m_j est repéré par ses coordonnées cartésiennes (x_{ij}, y_{ij}, z_{ij}) et les composantes de son vecteur vitesse (v_{xij}, v_{yij}, v_{zij}) dans le référentiel de référence.
Trois listes sont utilisées pour représenter ce système :

– `masse` conserve les masses de chaque corps : `masse[j]` $= m_j$;
– `position` contient les positions successives de chaque corps : `position[i][j]` $= [x_{ij}, y_{ij}, z_{ij}]$;
– `vitesse` mémorise les vitesses successives de chaque corps : `vitesse[i][j]` $= [v_{xij}, v_{yij}, v_{zij}]$.

L'objet de la suite du problème est de construire ces deux dernières listes **en mettant en œuvre l'algorithme de Verlet** défini par les équations (3).

8. *Les forces en jeu dans ce problème*

8.a) Exprimer la force $\overrightarrow{F_j}$ exercée sur le corps P_j par l'ensemble des autres corps P_k, avec $k \neq j$.

8.b) Écrire une fonction `distance(p1, p2)` qui prend en paramètres les positions, sous forme de liste des trois coordonnées cartésiennes en mètres, de deux corps 1 et 2, et qui renvoie un flottant représentant la distance entre ces deux corps.

8.c) Écrire une fonction `force2(m1, p1, m2, p2)` qui prend en paramètres les masses (`m1` et `m2` en kilogrammes) et les positions (`p1` et `p2`, sous forme de liste des trois coordonnées cartésiennes en mètres) de deux corps 1 et 2 et qui renvoie la valeur de la force exercée par le corps 2 sur le corps 1, sous la forme d'une liste à trois éléments représentant les composantes de la force dans la base de référence, en newtons.

8.d) Écrire une fonction `forceN(j, masse, pos)` qui prend en paramètres l'indice `j` d'un corps, la liste `masse` des masses des N corps du système étudié ainsi que la liste `pos` de leurs positions et qui renvoie $\overrightarrow{F_j}$, la force exercée par tous les autres corps sur le corps `j`, sous la forme d'une liste de ses trois composantes cartésiennes.

9. *Approche numérique*

9.a) Expliciter la structure et la signification de `position[i]` et `vitesse[i]`.

9.b) Justifier que $\overrightarrow{a_j} = \dfrac{1}{m_j}\overrightarrow{F_j}$ où $\overrightarrow{a_j}$ est l'accélération du corps de masse m_j et $\overrightarrow{F_j}$ est défini à la question 8.a.

9.c) On considère la fonction `fct` suivante où `masse` est la liste des masses des N corps du système étudié (en kilogrammes), `pos` la liste de leurs positions (en mètres) à l'instant t_i, `vit` celle de leurs vitesses (en mètres par seconde) au même instant et h le pas d'intégration (en secondes).

```
1  def fct(masse, pos, vit, h):
2      N = len(masse)
3      L = []
4      for j in range(N):
5          next_pj = smul(0, pos[j])
6          mj, pj, vj = masse[j], pos[j], vit[j]
7          force = forceN(j, masse, pos)
8          for k in range(3):
9              next_pj[k] = pj[k] + h*vj[k] + h**2/2*force[k]/mj
10         L.append(next_pj)
11     return L
```

i. Quelle est le type et que représente la variable `force` définie à la ligne 7 ?

ii. Que représente le nombre 3 de la ligne 8 ?

iii. Expliquer précisément le contenu de la ligne 9.

iv. Que renvoie cette fonction ? À quoi sert cette fonction dans le cadre du problème posé ?

9.d) On considère maintenant la fonction `etat_suiv` ci-dessous qui prend les mêmes paramètres que la fonction `fct` définie à la question 9.c.

```
1  def etat_suiv(masse, pos, vit, h):
2      N = len(masse)
3      next_pos = fct(masse, pos, vit, h):
4      next_vit = []
```

```
5       for j in range(N):
6           next_vj = ...
7           mj, vj = masse[j], vit[j]
8           fi = ...
9           fiplus1 = ...
10          for k in range(3):
11              next_vj[k] = ...
12          next_vit.append(next_vj)
13      return next_pos, next_vit
```

i. Compléter la ligne 6 afin que la variable `next_vj` soit initialisée comme le vecteur nul de même taille que `vit[j]`.

ii. Compléter les lignes 8 et 9 afin que les variables `fi` et `fiplus1` contiennent respectivement la valeur de l'accélération du corps de masse `masse[j]` à l'instant t_i et à l'instant t_{i+1}.

iii. Compléter la ligne 11 de façon qu'à la fin de la boucle sur `k` la variable `next_vj` contienne la valeur de la vitesse du corps de masse `masse[j]` à l'instant t_{i+1} en suivant l'algorithme de Verlet.

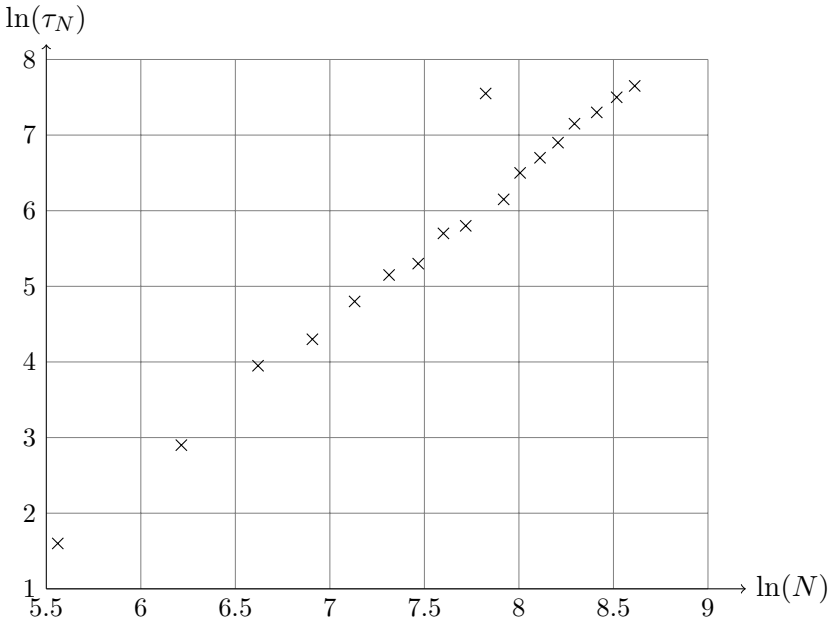
iv. Que renvoie la fonction `etat_suiv(masse, pos, vit, h)` ?

10. *Complexité*

10.a) Estimer la complexité temporelle des fonctions `force2` (question 8.c) et `forceN` (question 8.d) en fonction du nombre N de corps.

10.b) Estimer la complexité temporelle de la fonction `etat_suiv` sous la forme $O(N^\alpha)$ où N est le nombre de corps.

10.c) En notant τ_N la durée des calculs pour un nombre N de corps, la mise en œuvre de la fonction `etat_suiv` a donné le résultat graphique ci-dessous où on a porté $\ln(N)$ en abscisse et $\ln(\tau_N)$ en ordonnée.



i. Quelle relation simple peut-on établir entre $\ln(\tau_N)$ et $\ln(N)$ à partir de cette figure ?

ii. En déduire la valeur de τ_N en fonction de N.

iii. Ces résultats numériques corroborent-il le résultat théorique obtenu à la question 10.b ?

Ce TP est une adaptation du sujet de Centrale 2015 intitulé « Autour de la dynamique gravitationnelle ».