Préparation à l'Agrégation

PYTHON – EXERCICES

Année Universitaire 2023–2024

Intervenant : L. Le Guillou (Sorbonne Université / LPNHE)

1. Échauffement : Conjecture de Syracuse

La "conjecture de Syracuse" (aussi connue comme la conjecture de Collatz, d'Ulam, etc) prédit que pour tout entier $n \geqslant 1$, lorsqu'on applique l'algorithme suivant :

— si n est pair, on le divise par 2 ;

— si n est impair, on le multiplie par 3 et on ajoute 1.

de manière itérative, on arrive, après un certain nombre d'itérations, au nombre 1. *Cette conjecture n'a encore jamais été ni démontrée, ni falsifiée par la découverte d'un contre-exemple. De nombreux mathématiciens se sont penchés sur la question et une littérature abondante y est consacrée.*

1.1 — Écrivez un court programme Python qui teste la conjecture pour un entier n donné que vous choisirez.

1.2 — Transformez le programme précédent en une fonction `syracuse(n)` qui teste la conjecture de Syracuse pour un nombre entier n donné, et qui retourne le nombre d'itérations nécessaires pour atteindre 1 pour l'entier n (le "temps de vol" du nombre choisi).

1.3 — Modifiez la fonction précédente pour qu'elle retourne la "trajectoire" complète jusqu'à 1, sous la forme d'une liste.

1.4 — Tracez la trajectoire, puis le temps de vol de tous les entiers inférieurs à 100.

2. Trajectoire d'un obus

On étude le mouvement d'un objet (par exemple, un obus) lancé à $t = 0$ à une vitesse v_0 selon un angle θ par rapport à l'horizontale. Les positions successives ont été enregistrées dans le fichier `positions.txt`.

2.1 — Lisez le fichier `positions.txt` (fonction `numpy.loadtxt`). Tracez l'évolution de l'abscisse x en fonction du temps ; tracez de même l'évolution de la hauteur z en fonction de t ; enfin, tracez la trajectoire $z(x)$ de l'obus.

2.2 — Déterminez la hauteur maximale atteinte. A quel instant cela se produit-il ? (On utilisera les fonctions `max` et `argmax`).

Le mouvement est parabolique entre les instants $t_1 = 0\,\mathrm{s}$ et $t_2 = 115\,\mathrm{s}$.

2.3 — En utilisant les fonctionnalités de `numpy`, sélectionnez les mesures dans cet intervalle de temps. Tracez en rouge les points correspondants sur le graphe $z = f(t)$, et en noir les points qui sont hors de cet intervalle.

2.4 — Dans l'intervalle de temps $[t_1 : t_2]$, ajustez un polynôme de degré 1 sur les mesures de $x(t)$; ajustez de même un polynôme de degré 2 sur les mesures de $z(t)$.

2.5 — Déduisez-en la valeur de la vitesse initiale v_0, et celle de l'angle de tir ; vérifiez que vous retrouvez la valeur de g.

2.6 — En considérant les mesures prises après le tir (lorsque l'objet est retombé), estimez la dispersion sur les mesures de hauteur, et déduisez-en une estimation de l'erreur de mesure de hauteur.

3. Etude du pendule

On considère un pendule de masse m et de longueur ℓ, qui oscille dans le champ de pesanteur terrestre g avec une amplitude angulaire $\pm\theta_0$.

Pour les applications numériques, on prendra $m = 100\,\text{g}$, $\ell = 70$ cm et $g = 9.806$ m $\cdot$ s^{-2}.

Modèle : formule de Borda

3.1 — Dans la limite des petites oscillations, la période du pendule est donnée par $T_0 = 2\pi\sqrt{\frac{\ell}{g}}$. Calculez numériquement T_0.

La formule de Borda nous dit que, à l'ordre 2, la période du pendule dépend légèrement de l'amplitude θ_0, selon :

$$\frac{T}{T_0} = 1 + \frac{\theta_0^2}{16}$$

3.2 — Tracez la formule de Borda pour θ_0 entre 0 et $\pi/2$

Analyse de données expérimentales

Vous venez de faire des mesures de la période du pendule pour différents angles. Pour cela, vous avez mesuré à l'aide d'un chronomètre la durée totale de N oscillations. Voici vos mesures :

θ_0 [^{o}]	N	Durée [s]
10	20	37.70
20	15	28.28
30	15	28.46
40	20	38.71
50	25	49.50
15	20	37.64

3.3 — Saisissez ces données dans un tableur (e.g. libreoffice), sauvegardez les dans un fichier csv et importez les données avec Python (par exemple avec `numpy.loadtxt`, ou encore avec le module `csv`).

3.4 — Tracez la periode en fonction de l'amplitude θ_0.

3.5 — On estime que l'incertitude dans la mesure de T est d'environ $200\,\mathrm{ms}$. Tracez les points avec des barres d'erreurs (fonction `errorbar`).

3.6 — À l'aide de la fonction `curve_fit` du module `scipy.optimize`, ajustez les données par la fonction $T(\theta) = T_0(1+\beta\theta^2)$. Pour cela il faudra commencer par écrire une fonction `periode(theta, T_0, beta)`.

- Tracez les données avec l'ajustement.
- Quelle est la longueur du pendule ?
- Quelle est l'incertitude sur cette longueur ?

Intégration numérique

Dans le cas général, on peut démontrer que la période du pendule est donnée par :

$$\frac{T}{T_0} = \frac{2}{\pi}\int_0^{\pi/2} \frac{d\phi}{\sqrt{1-k^2\sin^2\phi}}$$

où $k = \sin\frac{\theta_0}{2}$. Calculez T pour $\theta_0 = \frac{\pi}{4}$. On utilisera la fonction `quad` du package `scipy.integrate`.

3.7 — Tracez avec des points la valeurs de $T(\theta_0)/T_0$ pour une dizaine de points entre 0 et $\pi/2$.

3.8 — Tracez la formule de Borda sur le même graphe pour comparer.

3.9 — Vérifiez numériquement le coefficient $\frac{1}{16}$ de la formule de Borda.

Résolution numérique de l'équation différentielle

3.10 — En utilisant la fonction `solve_ivp` du package `scipy.integrate`, résoudre l'équation différentielle associée au pendule :

$$\ddot{\theta} = -g\sin\theta.$$

On prendra comme condition initiale $\theta(t=0) = \frac{3\pi}{4}$ et $\dot{\theta}(t=0) = 0$.

3.11 — Tracez l'évolution au cours du temps.

3.12 — Vérifiez graphiquement que la période correspond à celle calculée précédemment.