TD 4 : NumPy et matplotlib

Exercice 4.1 : Manipulation des tableaux

(a) Créer un tableau de nombres flottants 1.5, 2, 2.5 … 10.5, 11.

(b) Réarranger dans un tableau 4×5 avec la fonction `numpy.reshape()`.

(c) De ce tableau, extraire les colonnes aux indices impairs.

(d) Extraire les lignes à partir de l'indice 2.

(e) De la dernière ligne, extraire les éléments aux indices pairs à partir de 2.

Exercice 4.2 : Calcul matriciel avec NumPy (développement multipolaire)

On considère une distribution de n charges électriques q_i localisées aux points $\vec{r}_i = (x_i, y_i, z_i)$. À grandes distances $r \gg \max_i |\vec{r}_i|$ on peut approximer le potentiel électrostatique par les premiers termes du *développement multipolaire* :

$$\Phi(\vec{r}) = \frac{1}{4\pi\epsilon_0}\left(\frac{q}{r} + \frac{\vec{r}\cdot\vec{p}}{r^3} + \frac{1}{2}\frac{\vec{r}\cdot Q\cdot\vec{r}}{r^5} + \dots\right) \tag{1}$$

où $r = |\vec{r}|$, la *charge totale* est

$$q = \sum_{i=1}^{n} q_i$$

le *vecteur dipolaire* est

$$\vec{p} = \sum_{i=1}^{n} q_i\,\vec{r}_i$$

et le *tenseur quadrupolaire* est

$$Q_{kl} = \sum_{i=1}^{n} q_i\left(3\,r_{ik}r_{il} - \vec{r}_i{}^2\,\delta_{kl}\right), \qquad \text{avec } r_{ik} = \vec{r}_i\cdot\vec{e}_k \text{ et } \delta_{kl} = \begin{cases} 1 & k = l \\ 0 & k \neq l \end{cases}.$$

(a) Réaliser trois fonctions `charge(qi)`, `dipole(qi, ri)` et `quadrupole(qi, ri)` qui calculent la charge totale, le moment dipolaire et le moment quadrupolaire pour un ensemble de charges q_i et points $\vec{r}_i$ donnés. Se servir des tableaux NumPy pour représenter les quantités vectorielles et matricielles.

(b) Réaliser une fonction `Phi(qi, ri, r)` qui calcule $\Phi(\vec{r})$ dans l'approximation (1). (On a $\frac{1}{4\pi\epsilon_0} = 8.99\cdot 10^9$ et $e = 1.60\cdot 10^{-19}$ en unités du SI.)

(c) On regarde maintenant le cas concret de charges et positions suivantes :

q_i [e]	x_i [nm]	y_i [nm]	z_i [nm]
+1	4	−2	1
+1	2	1	−1
−1	−3	2	0
−1	0	0	0

Réaliser un programme qui calcule $\Phi(\vec{r})$ dans l'approximation (1) pour $\vec{r} = r\,\vec{e}_x$, où r = 1 nm, 10 nm, 100 nm. Comparer avec le potentiel exact donné par la loi de Coulomb,

$$\Phi(\vec{r}) = \frac{1}{4\pi\epsilon_0}\sum_{i=1}^{n}\frac{q_i}{|\vec{r} - \vec{r}_i|}\,. \tag{2}$$

Indications : La plus simple méthode de construire Q est de partir avec un tableau de zéros, puis d'initialiser ses éléments avec une triple boucle `for` (même s'il y a des méthodes plus élégantes qui exploitent mieux les capacités de NumPy.) Pour le calcul de (2) il convient d'implémenter une fonction `norme` pour calculer la norme d'un vecteur (ou d'utiliser la fonction pré-définie `numpy.linalg.norm()`).

Exercice 4.3 : Tracer une fonction

Pour la distribution de charges d'exercice 4.2 (c), poser $y = z = 0$ et tracer les contributions à $\Phi(\vec{r})$ des trois termes dans éq. (1) pour r entre 3 nm et 10 nm. Dans le même graphique inclure aussi le potentiel exact (2).

Exercice 4.4 : Importer et visualiser des données

Dans le fichier `647_Global_Temperature_Data_File.txt` vous trouvez les déviations ΔT de la température moyenne globale par rapport à T_0, où T_0 est la température moyenne entre 1951 et 1980. Première colonne : année, deuxième colonne : ΔT en °C, troisième colonne : $\overline{\Delta T}$ = (ΔT moyennée sur 5 ans).

(Source : `https://climate.nasa.gov/vital-signs/global-temperature/`)

(a) Tracer ΔT et $\overline{\Delta T}$ en fonction du temps.

(b) Faire un histogramme de ΔT.

Exercice 4.5 : Diagramme de bifurcation

Le comportement asymptotique de la *suite logistique*, qui est récursivement définie par

$$x_0 = \frac{1}{2}\,, \quad x_{n+1} = r\,x_n(1 - x_n)$$

dépend du paramètre réel r. Plus précisément, pour $0 \leq r \leq 3$ la suite converge ; pour $3 < r \lesssim 3.6$ elle se compose d'un nombre fini de sous-suites convergentes ; pour $r \gtrsim 3.6$ le comportement est largement chaotique, à part certains intervalles qui donnent lieu aux structures plus régulières. L'objectif de cet exercice est d'étudier la suite logistique numériquement par son *diagramme de bifurcation* ou *diagramme de Feigenbaum*.

(a) Pour 1000 valeurs de r entre 2.5 et 4, calculer les x_n pour $0 \leq n \leq 200$. Enregistrer ces 201 000 nombres dans un fichier `logistique.dat`.

(b) Avec un deuxième programme, importer les données de `logistique.dat` et tracer x_{100}, x_{101}, $x_{102} \ldots x_{200}$ en fonction de r entre $r = 2.5$ et $r = 4$, tout dans le même graphique. Pour l'affichage utiliser des marqueurs pixel ', ' dans l'appel à `matplotlib.pyplot.plot()`.

Exercice 4.6 : Graphiques des fonctions de deux variables

La fonction d'onde de l'état $|n = 4,\, \ell = 3,\, m = 1\rangle$ de l'atome d'hydrogène est donnée en coordonnées cartésiennes par

$$\psi_{431}(x, y, z) = N\ \exp(-r/4)\frac{(x + i\,y)\,z\,(7\,z^2 - 3\,r^2)}{r}$$

où $r = \sqrt{x^2 + y^2 + z^2}$, N est une constante de normalisation (on prendra $N = 1$), et toutes les quantités sont en unités du rayon de Bohr a_0.

Visualiser la densité de probabilité $||\psi_{431}||^2$ dans le plan (x, z) pour x et z entre $-20\,a_0$ et $20\,a_0$

(a) par des courbes de niveau,

(b) par une carte de chaleur.