Informatique S6 - CC4

Le but de ces exercices est de construire une hiérarchie de systèmes dynamiques et leurs schémas en temps associés.

Exercice 1 (**EDO générales** (8.75 points)).

On commence par implémenter une classe générale d'équations différentielles du premier ordre. Il s'agit de systèmes dynamiques de la forme :

$$\frac{d\mathbf{X}(t)}{dt} = \mathbf{F}(\mathbf{X}(t))$$

avec $\mathbf{X}$ *un vecteur de taille* N_c*. Le flot de l'équation est la fonction* $\mathbf{F}(\mathbf{X})$*. A* $t = 0$*, on suppose que*

$$\mathbf{X}(t = 0) = \mathbf{X}_0$$

L'équation sera résolue par un schéma en temps de la forme générale :

$$\mathbf{X}^{n+1} = G(\mathbf{X}^n)$$

Le but est de stocker la suite en temps et de résoudre les équations.

On souhaite construire une classe ***EDO*** *qui contient comme attributs :* N_c *la taille du vecteur* $\mathbf{X}$*, un "vector"* ***de*** *"valarray" qui contiendra la suite en temps. Chaque élément* i *du "vector" sera un "valarray" qui correspond à* $\mathbf{X}_i$*. Elle contiendra un pointeur de fonction "flot" qui correspond au flot de l'équation.*

Question (**flot** (0.5 points)).
Expliquer quel est le type du/des paramètres de la fonction "flot" ainsi que le type de sortie (utile pour déclarer le pointeur).

Question (**classe** (2.25 points)).
Ecrire la classe, les constructeurs par défaut, copie et un prenant comme paramètre : une taille N_c *et une valeur d'initialisation* $\mathbf{X}_0$ *("valarray"), le destructeur et "=".*

Question (**explication** (0.75 points)).
Expliquer pourquoi l'utilisation de "valarray" peut simplifier l'écriture de certaines des fonctions précédentes par rapport aux exemples de cours utilisant des tableaux dynamiques (comme dans la classe "polynome").

Question (**flot** (0.75 points)).
Ecrire un mutateur pour l'attribut "flot", redéfinissez l'opérateur "[]" qui renvoie le "valarray" $\mathbf{X}_i$

Question (**flot** (0.75 points)).
Ecrire une fonction virtuelle "compute_flot" qui prenant un $\mathbf{X}$ *renvoit le résultat de l'application du flot :* $\mathbf{F}(\mathbf{X})$*.*

Question (**schéma en temps** (1.5 points)).
Ecrire une fonction virtuelle "solve_dt(double dt)" qui résout un cycle en temps avec le schéma :

$$\mathbf{X}^{n+1} = \mathbf{X}^{n} + \mathrm{dt}\,\mathbf{F}(\mathbf{X}^{n})$$

et qui stocke le nouvel itéré dans le "vector" contenant la suite en temps.

Pour cela, utilisez de préférence, les fonctionnalité de la classe "vector" et la fonction précédentes. Expliquez l'avantage de "valarray" pour écrire cette fonction.

Question (**Boucle en temps** (1.25 points)).
Ecrire une fonction "loop_time" qui n'appartient à aucune classe, qui prend en entrée : un pointeur sur une EDO, un temps final Tf (double), un dt (double). Cette fonction implémentera l'algorithme suivant :

- *time =0*
- *tant que time < Tf :*
  - *on appelle "solve_dt(dt)" à partir de l'EDO donnée en paramètre.*
  - *time = time +dt*

Question (**Exemple** (1.0 points)).
Résolvez en temps l'exemple : $N_c = 1$, $\mathbf{X} = x$ *et le flot* $\mathbf{F}(\mathbf{X}) = e^{-2x}$. *La condition initiale, le pas de temps et le temps final sont à choisir à votre guise.*

Exercice 2 (**Systèmes de Poisson** (7.5 points)).

Un système de Poisson est une EDO particulière de la forme

$$\frac{d\mathbf{X}(t)}{dt} = J\big(\mathbf{X}(t)\big)\nabla H\big(\mathbf{X}(t)\big)$$

avec $H(\mathbf{X}(t))$ *une fonction appelée* ***Hamiltonien*** *de* $\mathbb{R}^{N_c}$ *dans* $\mathbb{R}$ *et* $J(\mathbf{X}(t))$ *une matrice antisymétrique satifaisant des propriétés qu'on ne détaillera pas dans ce travail. La forme particulière de ces équations permet de montrer que l'****Hamitonien*** *est une quantité conservée :*

$$\frac{H(\mathbf{X}(t))}{dt} = 0$$

Remarque *: Il s'agit donc d'une classe d'EDO ou on définit les équations de mouvements à l'aide de la quantité invariante. Préserver au maximum cette propriété en temps est important quoi que difficile pour les systèmes de Poisson.*

Question (**classe** (2.75 points)).
Ecrire une classe "Poisson_System" héritant de la classe "EDO". Elle contiendra comme attributs spécifiques : 3 pointeurs de fonction. 1 pour la Hamiltonien H*, 1 pour le gradient du Hamiltonien* ∇H*, 1 pour une fonction qui prendra en entrée deux vecteurs* X*,* Y *("valarray"), et qui renverra le produit le vecteur* $J(X)Y$*. Justifier les signatures (sortie et paramètres d'entrée) des trois pointeurs de fonctions. Ecrire les constructeurs (copie, défaut et un prenant comme paramètres : la taille du système et* $\mathbf{X}_0$*)* ***en utilisant les constructeurs de la classe mère*** *et le destructeur.*

Question (**mutateurs** (1.0 points)). *Ecrire les trois mutateurs pour les pointeurs de fonction et le "=".*

Question (**flot** (0.75 points)). *Réécrire la fonction compute_flot. Ici le flot est calculé par la formule*

$$J(\mathbf{X})\nabla H(\mathbf{X})$$

Noter que l'attribut flot de la classe mère EDO n'est pas utilisé et n'a pas besoin d'être défini pour les éléments de la classe "Poisson_System". La fonction compute_flot utilise uniquement les attributs de la classe fille "Poisson_System".

Question (**exemple** (1.25 points)). *Coder l'exemple de Lokta-Voltera*

$$H(x,y) = -x + \log(x) - \frac{4}{3}y + \frac{2}{3}\log(y), \quad \nabla H(x,y) = \begin{pmatrix} -1 + \frac{1}{x} \\ -\frac{4}{3} + \frac{2}{3}\frac{1}{y} \end{pmatrix}$$

$$J(x,y) = \begin{pmatrix} 0 & xy \\ -xy & 0 \end{pmatrix}$$

Pour cela, on définira les fonctions, on créera un objet "Poisson_system", on initialisera les données avec le cosntructeur et les mutateurs et on appelera la fonction "loop_time" de l'exercice 1.

Question (**explication** (1.75 points)). *Expliquer le fonctionnement des appels des fonctions "solve_dt" et "compute_flot" dans les 2 cas : le pointeur *edo pointe sur objet EDO (exo 1) ou sur un objet Poisson_System (question précédente). Quel mécanisme détaillé dans le cours est utilisé ?*

Exercice 3 (**Systèmes Hamiltonien** (6.0 points)).

Le système Hamiltonien sont un cas particulier de système de Poisson, utilisé pour les systèmes de particules en physique, pour lequel :

$$\mathbf{X} = \begin{pmatrix} \mathbf{q} \\ \mathbf{p} \end{pmatrix}$$

avec $\mathbf{q}$ *le vecteur des positions des* m *particules et* $\mathbf{p}$ *le vecteur des impulsions (en gros des vitesses) des* m *particules. Dans ce cas, la matrice* $J(\mathbf{X})$ *est constante :*

$$J(\mathbf{X}) = J = \begin{pmatrix} 0 & I_m \\ -I_m & 0 \end{pmatrix}$$

avec I_m *la matrice identité de taille* m*. On a donc pour les systèmes Hamiltonien* $N_c = 2m$*.*

Question (**Classe** (2.25 points)).
*Ecrire un **template de classe** "Halmitonien_System" héritant de la classe "Poisson_system" avec* m *comme paramètre du template. Ecrire les constructeurs (copie, défaut et un prenant comme paramètre* $\mathbf{X}_0$*), le destructeur. Il faudra **utilisé les constructeurs de la classe mère** et ne pas oublier de remplir la taille globale du système.*

Question (**flot** (1.0 points)).
Ecrire la fonction "compute_flot" qui calcule :

$$J\nabla H(\mathbf{x})$$

Question (**Verlet** (1.0)). *Pour les sytèmes Hamiltonien il existe un schéma en temps assez facile permettant de préserver au mieux* $\frac{d}{dt}H(\mathbf{X}(t))$*. Le code est donné en commentaire. Mettez le dans une redéfinition de la fonction solve_dt pour la classe "Halmitonien_System". Expliquez à quoi sert la fonction "slice" utilisée plusieurs fois.*

Question (**exemple** (1.75 points)).
Coder l'exemple de 2 oscillateurs Harmoniques découplés

$$H(q_1, q_2, p_1, p_2) = \frac{1}{2m}(p_1^2 + p_2^2) + \frac{1}{2}\omega(q_1^2 + q_2^2), \quad \nabla H(q_1, q_2, p_1, p_2) = \begin{pmatrix} \omega q_1 \\ \omega q_2 \\ \frac{p_1}{m} \\ \frac{p_2}{m} \end{pmatrix}$$

Pour cela on définira ces deux fonctions, on créera un objet "Hamiltonien_system", on initialisera les données avec le constructeur et les mutateurs et on appelera la fonction "loop_time" de l'exercice 1. Expliquer quelle fonction "solve_dt" est appelée et pourquoi.