Examen de Techniques et Implémentation de Méthodes Numériques UE EMEAT1B1

Supports de cours autorisés
Téléphones et autres appareils électroniques interdits
Durée : 1h30 - 30 minutes supplémentaires pour tiers temps

Directives : Il est indispensable de traiter les questions en respectant les contraintes et exigences de l'énoncé et des questions. La notation tiendra compte de cela.

Prévision avec des chaines de Markov Une chaîne de Markov est une suite de variables aléatoires $(X_n, n \in \mathbb{N})$ qui permet de modéliser l'évolution dynamique d'un système aléatoire : X_n représente l'état du système à l'instant n.

La propriété fondamentale des chaînes de Markov, dite propriété de Markov, est que son évolution future ne dépend du passé qu'au travers de sa valeur actuelle. Autrement dit, conditionnellement à X_n, nous avons $(X_0, ..., X_n)$ et $(X_{n+k}, k \in \mathbb{N})$ qui sont indépendants.

Les applications des chaînes de Markov sont très nombreuses (réseaux, génétique des populations, mathématiques financières, gestion de stock, algorithmes stochastiques d'optimisation, simulation, etc.).

Pour prévoir l'état dans lequel se trouve un système, nous avons besoin d'un ensemble E qui énumère tous ses états. X_n prendra à chaque instant une valeur dans E.

À une chaîne de Markov, nous associons :

- un graphe orienté pondéré, dit graphe pondéré associé, dont les sommets sont les états et dont l'arête orientée reliant l'état i à l'état j est pondérée par la probabilité de passer de l'état i à l'état j.
- une matrice carrée P, dite matrice de transition associée, où le coefficient situé en ligne i et colonne j est égal à la probabilité de passer de l'état i à l'état j.

L'évolution d'une chaîne de Markov est conditionnée par sa matrice de transition P et par sa distribution initiale π_0. La distribution π_n est la distribution après n transitions, c'est-à-dire l'ensemble des probabilités d'être dans chacun des états du système à l'instant n. La distribution π_0 est donc l'ensemble des probabilités d'être dans chacun des états du système à l'instant initial. La somme des probabilités de chaque distribution doit être égale à 1.

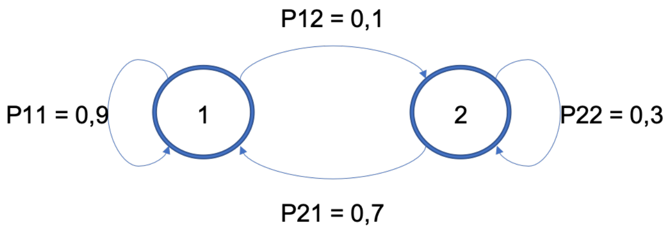
Parmi les propriétés, nous avons :

1. Pour tout entier naturel n, on a $\pi_{n+1} = \pi_n \times P$
2. Pour tout entier naturel n, on a $\pi_n = \pi_0 \times P^n$
3. Pour tous les états i et j et tout entier naturel $n \geq 1$, le coefficient de la ligne i et colonne j de la matrice P^n est la probabilité de passer de l'état i à l'état j en n transitions

Exemple Dans un système, au cours de sa vie opérationnelle, s'il fonctionnne correctement un certain jour alors il sera également en fonctionnement nominal le lendemain avec une probabilité de 0,9. Il a donc une probabilité de 0,1 de tomber en panne.

De plus, s'il est en panne un certain jour alors il sera encore en panne le lendemain avec une probabilité égale à 0,3. Il a donc une probabilité de 0,7 d'être réparé et de fonctionner correctement à nouveau.

On choisit au hasard une journée. X_n est la variable aléatoire qui prend la valeur 1 si le système fonctionne et 2 s'il est en panne après n jours.



Nous avons :

- l'ensemble des états $E = \{1, 2\}$
- $\pi_0 = (1 \quad 0)$ avec probabilité d'être dans état $1 = 1$ et probabilité d'être dans état $2 = 0$
- $P = \begin{pmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{pmatrix} = \begin{pmatrix} 0,9 & 0,1 \\ 0,7 & 0,3 \end{pmatrix}$
- $\pi_1 = \pi_0 \times P = (1\ 0) \cdot \begin{pmatrix} 0,9 & 0,1 \\ 0,7 & 0,3 \end{pmatrix} = (0,9 \quad 0,1)$ et on vérifie que $0,9 + 0,1 = 1$
- $\pi_2 = \pi_1 \times P = (0,9 \quad 0,1) \cdot \begin{pmatrix} 0,9 & 0,1 \\ 0,7 & 0,3 \end{pmatrix}$
  $\pi_2 = (0,81 + 0,07 \quad 0,09 + 0,03) = (0,88 \quad 0,12)$ et on vérifie que $0,88 + 0,12 = 1$
- si $\pi_n = (a_n \quad b_n)$ alors $\pi_{n+1} = \pi_n \times P = (a_n \cdot p_{11} + b_n \cdot p_{21} \quad a_n \cdot p_{12} + b_n \cdot p_{22})$
- comme $\pi_n = \pi_0 \times P^n$ alors on peut calculer directement $\pi_{n+1} = \pi_0 \times P^{n+1}$

Rappels Pour le produit matriciel d'une matrice carrée P, nous avons :

$P^2 = P \times P$
$P^3 = P \times P \times P = P \times P^2 = P^2 \times P$
$P^n = P \times P^n = P^n \times P$

Objectifs Nous désirons disposer d'un programme capable de pouvoir :

— saisir une distribution π_0 de taille $1 \times T$
— saisir la matrice P de taille $T \times T$
— afficher une matrice
— calculer la puissance n de la matrice carrée P
— calculer $\pi_{n+1} = \pi_n \times P$, de taille $1 \times T$
— calculer $\pi_n = \pi_0 \times P^n$, de taille $1 \times T$

L'utilisateur devra saisir la taille T.

Les matrices distributions et la matrice de transition seront donc créés dynamiquement.

Même si une distribution π_n ressemble à un vecteur ligne, il s'agit d'une matrice 1 ligne et T colonnes.

Questions Il faut écrire en langage C différents sous-programmes dans le fichier `bibliotheque.c` à partir des spécifications données ci-dessous, permettant de réaliser les opérations nécessaires.

Vos réponses devront respecter les exigences et contraintes définies dans l'énoncé et les questions.

Vous devrez respecter : les constantes, les types des variables, le nombre et le type des paramètres des fonctions, les allocations, etc.

Le fichier `bibliotheque.c` est à télécharger sur Moodle avec 2 autres fichiers :
`bibliotheque.h` et `modele-examen-TIMN-21-22-session1.c`

Une fois que les fonctions ont été codées, il faut **impérativement** déposer le fichier `bibliotheque.c` sur l'espace Moodle dans la partie réponse au test de Examen1.

Les fichiers `bibliotheque.h` et `modele-examen-TIMN-21-22-session1.c` ne doivent pas être modifiés!!!

Pour compiler et tester votre code, les commandes sont indiquées au début du fichier `modele-examen-TIMN-21-22-session1.c`

Les spécifications des sous-programmes à écrire sont les suivantes :

1. `void Afficher_matrice(int nb_lignes,int nb_colonnes, float ** matrice)`
   Ce sous-programme affiche une matrice quelconque de nombre réels à l'aide de ses dimensions. Il permet donc d'afficher les matrices de distribution π_n et la matrice de transition P.
2. `void Saisir_distribution_init(int taille, float ** matrice_pi)`
   Ce sous-programme remplit la matrice de distribution π_0 de dimension $1 \times taille$. L'utilisateur remplit la matrice manuellement. La matrice est déjà dimensionnée et donc l'allocation dynamique est déjà faite.
3. `void Saisir_matrice_transition(int taille, float ** matrice_P)`
   Ce sous-programme remplit la matrice de transition P carrée de dimension $taille \times taille$. L'utilisateur remplit la matrice manuellement. La matrice est déjà dimensionnée et donc l'allocation dynamique est déjà faite.
4. ```
   void Calculer_PIn(int taille, float ** Pi_mat, float ** P_mat,
                     float ** Pi_n_mat)
   ```
   Ce sous-programme effectue le produit matriciel de deux matrices : une matrice de distribution π_n et une matrice de transition P^n. Cette fonction est générique pour le calcul de n'importe quel π_n.
   Si `Pi_mat`$=\pi_0$ et `P_mat`$=P$ alors elle calculera $\pi_1 = \pi_0 \times P$
   Si `Pi_mat`$=\pi_n$ et `P_mat`$=P$ alors elle calculera $\pi_{n+1} = \pi_n \times P$
   Si `Pi_mat`$=\pi_0$ et `P_mat`$=P^n$ alors elle calculera $\pi_n = \pi_0 \times P^n$
5. ```
   void Produit_Pn(int taille, float **P1_mat,  float **P2_mat,
                   float **Pn_mat)
   ```
   Ce sous-programme effectue le produit matriciel de deux matrices carrées à l'aide des matrices d'origine `P1_mat` et `P2_mat`, de leur taille, et renvoie le produit dans `Pn_mat`.
   Si `P1_mat`$=P$ et `P2_mat`$=P^2$ alors `Pn_mat`$=P^3$. Un appel itératif de cette focntion permettra de calculer P^n.