
TD N° 3 : Programmation dynamique

Exercice 1. (Économie)

Soit un consommateur qui possède une richesse initiale x_0. À chaque instant $t \in [0, T-1]$, il consomme c_t, mais il ne consomme pas à la dernière période, indexée par T. Ce qui n'est pas consommé à chaque étape est épargné avec un taux d'intérêt de 0%. Le programme du consommateur s'écrit :

$$\max_{c_0,\dots,c_{T-1}} \sum_{t=0}^{T-1} c_t^\alpha + x_T^\alpha$$
$$\text{s.c.} \quad x_{t+1} = x_t - c_t$$

pour une constante $\alpha > 0$

1) Proposer une interprétation pour le modèle ainsi que pour le paramètre $\alpha \in]0, 1[$.
2) Donner la valeur de $V(T-1, x)$ puis $V(T-2, x)$ pour tout x.
3) Décrire la stratégie optimale pour le cas $T = 2$, *i.e.,* donner combien vaut la solution : c_0^*, c_1^*, c_2^*.
4) Comment le modèle changerait-il si l'on avait un taux d'intérêt $\delta > 0$?

Exercice 2. (Multiplication en chaîne de matrice) On considère l'évaluation d'une matrice $M = M_1 \times M_2 \times \dots M_n$. On rappelle que multiplier une matrice $p \times r$ par une matrice $r \times q$ nécessite $p \times q \times r$ produits entre scalaires.

1) Considérons 4 matrices $A : 20 \times 5$, $B : 5 \times 100$, $C : 100 \times 8$, $D : 8 \times 30$. On souhaite calculer le produit $ABCD$. En fonction de l'ordre des opérations, le nombre de produits peut varier. Déterminer le nombre de produits pour calculer $ABCD$, si on utilise les parenthèses comme suit : $((AB)C)D$ ou $(A(BC))D$
2) L'objectif est de concevoir un algorithme qui permet de minimiser le nombre de produits scalaires pour évaluer M. Nous noterons que la matrice M_i est de dimension $d_{i_1} \times d_i$. Définissons le nombre minimal de produits scalaires nécessaires pour évaluer le produit des matrices $M_i \times M_{i+1} \dots M_{j-1} \times M_j$ par $c(i, j)$. Écrire une formule de récurrence pour calculer $c(i, j)$, en utilisant la programmation dynamique.

Exercice 3. (Fibonacci) Proposer un algorithme qui donne le calcul du n^{e} terme de la suite de Fibonacci, dont le nombre d'additions est exactement $2n$ (et qui peut stocker seulement deux nombre sentiers).