TD: matrices creuses

(*) très facile (1-4 lignes), (**) moyen, (***) des subtilités
On rappelle les trois formats principaux de matrices creuses.

COO format triplet ou coordonnées. On stocke des triplets valeur, colonne, ligne. Pratique pour construire une matrice creuse. Sinon, inefficace.

CSR format ligne compressée. Il utilise trois tableaux: le tableau `data` des valeurs de taille n_{nz}, le tableau `indices` de taille n_{nz} qui contient la colonne de chaque valeur, le tableau `indptr` de taille nombre de lignes+1, qui stocke le numéro du premier élément dans sa ligne.

CSC format colonne compressée. Même principe que CSR, mais par colonne.

Exercice 1 : Opérations sur matrices CSR

1. (*) Somme des lignes: Ecrire une fonction sumRow(A,i) pour sommer tous les coefficients de la ligne i d'une matrice A codée en CSR.
2. (*) ou (**) Chercher la valeur d'un élément: Ecrire une fonction at(A,i,j) qui retourne la valeur du coefficient A_{ij} (A CSR).
3. (**) Produit matrice CSR / vecteur: Ecrire la fonction produit(A,v), qui réalise le produit $A\mathbf{v}$, où A est une matrice CSR, $\mathbf{v}$ un vecteur de taille n.
4. (*) Coût de stockage: Dans l'exemple du système masse-ressort, quel serait le coût de stockage de la matrice A pleine ? Même question si elle est stockée sous forme CSR ? Si nous avons 10000 ressorts, que les nombres à virgule flottante sont codés en 64 bits (double) et que les indices sont codés par des entiers 32 bits, évaluez le coût mémoire réel de ces deux codages de matrice.
5. (*) Calcul de la norme ∞ d'une matrice CSR. Quel est la complexité de cette fonction ? Serait-ce plus compliqué de calculer la norme 1 sur ce type de matrice ?

Exercice 2 : Systèmes triangulaires avec matrice creuse
On suppose que L est une matrice triangulaire inférieure $n \times n$ représentée creuse en format CSR, avec n_{nz} coefficients non nuls.

1. (*) Ecrivez la fonction isLowerTriangular(L) qui retourne vrai si et seulement si la matrice L est bien triangulaire inférieure. Quelle est sa complexité en fonction de n et/ou n_{nz} ?
2. (**) Ecrivez la fonction solveLowerTriangular(L, b) qui retourne le vecteur solution $\mathbf{x}$ au système $L\mathbf{x} = \mathbf{b}$.

Exercice 3 : Conversion vers ou de matrice CSR

1. (***) Conversion COO vers CSR: Ecrire une fonction tocsr(A) qui convertit la matrice A de format COO en une matrice au format CSR.

Exercice 4 : Opérations sur matrices CSC

1. (**) Peut-on écrire un algorithme de produit matrice/vecteur aussi efficace entre une matrice stockée en mode CSC (colonne compressée) et un vecteur ? Ecrivez son pseudo-code et donnez sa complexité.

2. (**) Produit matrice CSR avec matrice CSC.

3. (**) Conversion matrice CSR vers COO, puis COO vers CSC.

Take-away message

- La plupart des problèmes d'algèbre linéaire que l'on doit résoudre en pratique comportent des matrices creuses, où peu de coefficients par ligne/colonne sont différents de zéro.
- Les représentations CSR (compressed row format) et CSC (compressed column format) sont des représentations efficaces de matrices creuses, avec un coût mémoire proportionnel au nombre de coefficients non nuls.
- Beaucoup d'opérations matrice/matrice et matrice/vecteur peuvent être codés efficacement avec ces formats.
- Il faut adapter les algorithmes de résolution de systèmes linéaires à ces formats, avec des balayage par ligne ou colonne suivant que le format est CSR ou CSC.