1 Systèmes linéaires

1.1 rappels

Résoudre avec Maple le système linéaire suivant : $\left\{\begin{array}{rcrcrcl} 4x & + & 5y & - & 2z & = & 4 \\ x & + & y & + & z & = & 1 \\ 2x & + & 2y & + & z & = & 3 \end{array}\right.$

1.2 “à la main”

Comment résoudre le système précédent sans Maple ?
La solution que vous avez vue en cours consiste à appliquer l'algorithme du pivot de Gauss. Pour notre exemple, les deux premières opérations sont $L_2 \leftarrow L_2 - \frac{1}{4} * L_1$ et $L_3 \leftarrow L_3 - \frac{1}{2} * L_1$.

On obtient alors le nouveau système : $\left\{\begin{array}{rcrcrcl} 4x & + & 5y & - & 2z & = & 4 \\ & - & \frac{1}{4}y & + & \frac{3}{2}z & = & 0 \\ & - & \frac{1}{2}y & + & 2z & = & 1 \end{array}\right.$

On poursuit ainsi avec l'opération $L_3 \leftarrow L_3 - 2 * L_2$. On obtient finalement le système triangulaire suivant : $\left\{\begin{array}{rcrcrcl} 4x & + & 5y & - & 2z & = & 4 \\ & - & \frac{1}{4}y & + & \frac{3}{2}z & = & 0 \\ & & & - & z & = & 1 \end{array}\right.$

La résolution d'un système triangulaire est “facile”, en utilisant une remontée en cascade. Pour cet exemple, on déduit d'abord $z = -1$, puis en injectant dans la seconde ligne $y = -6$, et enfin $x = 8$.
Le coefficient 4 de x dans la première équation, puis le coefficient $\frac{1}{4}$ de y dans la deuxième ont servis de “pivots”.

Résolvez sur une feuille de cette façon le système suivant $\left\{\begin{array}{rcrcrcl} a & & & + & c & = & 2 \\ 4a & + & b & + & 2c & = & 10 \\ a & + & 3b & - & 4c & = & 0 \end{array}\right.$

2 Préliminaires

2.1 Théorie

À un système linéaire S sont associés une matrice A et un vecteur B. Résoudre le système revient alors à trouver les vecteurs X tels que $AX = B$.
L'algorithme de Gauss consiste à trianguler la matrice A (ici uniquement avec des opérations sur les lignes), afin d'obtenir, après avoir appliqué les mêmes opérations sur B, une nouvelle équation $TX = C$, que la forme triangulaire de T permet de résoudre plus vite.
Remarque : A et T ayant même rang, cet algorithme permet également de calculer le rang d'une matrice.

2.2 Écrivez les procédures suivantes :

Vous pouvez utiliser dans la suite `coldim`, `rowdim`, et bien sûr `evalm` (après avoir chargé `linalg`...).
`Echange_lignes(A,i,k)` prend en arguments une matrice A et deux entiers distincts i et k, et renvoie la matrice obtenue à partir de A en échangeant les lignes i et k.
`Retranche_ligne(A,i,k,`λ`)` prend en arguments une matrice A, deux entiers distincts i et k et un réel λ, et renvoie la matrice obtenue à partir de A en soustrayant à la k-ème ligne λ fois la i-ème.

`Cherche_pivot(A,i,j)` prend en arguments une matrice A de taille $n \times m$ et deux entiers i et j, et renvoie les coordonnées sous la forme d'une liste de taille 2 du premier élément non nul de la sous matrice $A[i..n, j..m]$ lue de haut en bas, puis de gauche à droite. Si un tel élément n'existe pas, la procédure retournera la liste vide.

3 Algorithme de Gauss

L'idée de l'algorithme de Gauss est de choisir un pivot, un élément non nul de la sous-matrice restant à traiter (ici la procédure `Cherche_pivot` retourne le premier trouvé dans un certain ordre) et de ne garder qu'une ligne le concernant : on annule sur sa colonne toutes les autres lignes (de la sous-matrice).
Si A_{ij} est le pivot choisi, déterminez la valeur de λ_k permettant d'annuler la ligne k sur la colonne j à l'aide de la ligne i : on souhaite avoir $(L_k - \lambda_k * L_i)[j] = 0$.
Écrivez alors la procédure `Vide_colonne(A,i,j)` qui prend en argument une matrice A, deux entiers i et j et retourne la matrice B obtenue en appliquant l'étape précédente à toutes les lignes situées en dessous de la i-ème, de sorte à avoir $B[\ell, j] = 0$ pour tout $\ell \geq i + 1$.

Présentation de l'algorithme :

```
Entrée : A
Variables locales : m,n,i,j,p,M
n := nb de lignes de A
m := nb de colonnes de A
M := copie de A
i,j :=1
Tant que i<n+1 et j<m+1
   p :=Cherche_pivot(M,i,j)                  le pivot a pour coordonnées (p[1],p[2])
   Si p est vide                             dans ce cas, la sous-matrice restante est nulle : on a donc fini
     alors Renvoyer M et fin                 rappel : RETURN met fin à la procédure
   FinSi
   M :=Echange_lignes(M,i,p[1])              on remonte la ligne du pivot en position i
   M :=Vide_colonne(M,i,p[2])                on annule le reste de la colonne
   i :=i+1                                   on passe à la ligne suivante
   j :=p[2]+1                                et à la colonne suivante
Fin Tant que
Renvoyer M
Fin Procédure
```

4 Applications *(on se place dans le cas $n \times n$ inversible)*

4.1 Résolution d'un système

On se replace dans le cadre des systèmes. L'algorithme précédent nous donne donc $AX = B \iff TX = C$. Démontrez alors que la formule suivante est correcte : $\forall\, i,\ x_i = \frac{1}{T_{ii}}(c_i - \sum_{j>i} T_{ij}.x_j)$. Reprenez alors l'algorithme précédent de sorte à ce qu'il prenne également B en argument, et résolve ainsi le système $AX = B$.

4.2 Calcul de l'inverse

L'algorithme de Gauss tel que nous l'avons présenté plus haut retourne une matrice triangulaire. On peut facilement à partir de là en obtenir une diagonale. Ajoutez les opérations nécessaires.
Enfin, en multipliant chaque ligne par un coefficient approprié, on peut obtenir la matrice identité. Si on applique les mêmes opérations sur Id_n, on obtient A^{-1}. Pourquoi ? Reprenez la procédure pour qu'elle modifie en même temps A et Id_n et retourne ainsi A^{-1}.