XI. Algorithme de Gauss-Jordan

1 Opérations élémentaires sur les lignes d'une matrice

Définition 1. *On définit trois types d'*opérations élémentaires *sur les lignes d'une matrice :*
- *échange des lignes i et j : $L_i \leftrightarrow L_j$.*
- *multiplication de la ligne i par $\lambda \neq 0$: $L_i \leftarrow \lambda L_i$.*
- *ajout à la ligne i de la ligne $j \neq i$ multipliée par λ : $L_i \leftarrow L_i + \lambda L_j$.*

Exercice 1. *On représente une matrice par un tableau à deux dimensions.*
- *Écrire une fonction* ***echange_lignes*** *de paramètres M, i et j qui échange les valeurs des tableaux M_i et M_j. (on procédera à l'échange élément par élément)*
- *Écrire une fonction* ***multiplication_ligne*** *de paramètres M, i et λ qui multiplie chacun des éléments du tableau M_i par λ.*
- *Écrire une fonction* ***ajout_ligne*** *de paramètres M, i, j et λ qui ajoute à chaque élément du tableau M_i le produit par λ de l'élément de même indice du tableau M_j.*

2 Algorithme de Gauss-Jordan

Exercice 2. *Effectuer successivement les opérations élémentaires suivantes correspondant à l'algorithme de Gauss-Jordan sur la matrice* $M = \begin{pmatrix} 2 & 4 & -8 & 10 \\ -2 & -4 & 8 & 10 \\ 3 & 9 & -3 & 6 \\ 1 & 4 & 2 & -2 \end{pmatrix}$.

- *Colonne* 1
  - *choix du pivot :* m_{11}
  - *remontée de la ligne du pivot :*
  - *unitarisation du pivot :* $L_1 \leftarrow \frac{1}{2}L_1$
  - *élimination :* $\begin{array}{lcl} L_2 & \leftarrow & L_2 + 2L_1 \\ L_3 & \leftarrow & L_3 - 3L_1 \\ L_4 & \leftarrow & L_4 - L_1 \end{array}$
- *Colonne* 2
  - *choix du pivot :* m_{31}
  - *remontée de la ligne du pivot :* $L_2 \leftrightarrow L_3$
  - *unitarisation du pivot :* $L_2 \leftarrow \frac{1}{3}L_2$
  - *élimination :* $\begin{array}{lcl} L_1 & \leftarrow & L_1 - 2L_2 \\ L_4 & \leftarrow & L_4 - 2L_2 \end{array}$
- *Colonne* 3
  - *choix du pivot :*
  - *remontée de la ligne du pivot :*
  - *unitarisation du pivot :*
  - *élimination :*
- *Colonne* 4
  - *choix du pivot :* m_{44}
  - *remontée de la ligne du pivot :*
  - *unitarisation du pivot :* $L_3 \leftarrow \frac{1}{20}L_3$
  - *élimination :* $\begin{array}{lcl} L_1 & \leftarrow & L_1 - 11L_3 \\ L_2 & \leftarrow & L_2 + 3L_3 \\ L_4 & \leftarrow & L_4 + L_3 \end{array}$

Exercice 3.

1. *Écrire une fonction* ***choix_pivot*** *de paramètres M, i et j qui retourne l'indice de ligne du premier pivot non nul dans la colonne d'indice j du tableau à deux dimensions M à partir de l'indice de ligne i.* *(on retournera la valeur -1 si aucun pivot n'est trouvé)*
2. *En déduire une fonction* ***gauss_jordan*** *de paramètre M, qui retourne le tableau obtenu par exécution de l'algorithme de Gauss-Jordan sur le tableau à deux dimensions M.*

Exercice 4. *On applique l'algorithme de Gauss-Jordan à une matrice de taille $n \times n$, majorer le nombre de multiplications et en déduire la classe de complexité temporelle de l'algorithme de Gauss-Jordan.*

3 Calcul de l'inverse d'une matrice

Exercice 5. *Appliquer l'algorithme de Gauss-Jordan sur la matrice carrée* $M = \begin{pmatrix} 1 & 0 & 5 \\ 2 & 1 & 6 \\ 3 & 4 & 0 \end{pmatrix}$ *et effectuer conjointement les opérations élémentaires sur la matrice* $I_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

Vérifier que la matrice ainsi obtenue est l'inverse de la matrice M.

Exercice 6. *Écrire une fonction* ***inverse*** *de paramètre M qui retourne l'inverse de la matrice M dans le cas où celle-ci est inversible.*

4 Résolution d'un système linéaire

Exercice 7. *Résoudre le système* $\left\{ \begin{array}{rcrcrcr} x & + & 2y & - & z & = & 2 \\ x & + & y & - & 2z & = & -3 \\ 2x & + & y & - & z & = & 1 \end{array} \right.$ *en appliquant l'algorithme de Gauss-Jordan.*

Exercice 8. *Écrire une fonction* ***resolution_systeme*** *de paramètres A et V qui retourne la solution du système linéaire $AU = V$ d'inconnue U dans le cas où celui-ci admet une unique solution.*

Exercices supplémentaires

Exercice 9. *Modifier la fonction* ***choix_pivot*** *de l'exercice 3 afin de retourner le pivot de valeur absolue maximale dans la colonne d'indice j du tableau à deux dimensions M à partir de l'indice de ligne i.* *(ce choix est souvent effectué pour des raisons de stabilité numérique)*

Exercice 10. *Écrire une fonction* ***test_inversibilite*** *de paramètre M permettant de déterminer si la matrice carrée M est inversible.*

Exercice 11. *Écrire une fonction* ***test_compatibilite*** *de paramètres A et V permettant de déterminer si le système linéaire $AU = V$ d'inconnue U est compatible.*

Exercice 12. *Écrire une fonction* ***noyau*** *de paramètre M permettant de déterminer le noyau de la matrice M.*

Réponses

1)

- **Fonction**: echange_lignes(M, i, j)
  Action: échange des valeurs des tableaux M_i et M_j
  Début
  Pour k **allant de** 0 **à** longueur$(M_i) - 1$ **faire**
  $m \leftarrow M_{ik}$
  $M_{ik} \leftarrow M_{jk}$
  $M_{jk} \leftarrow m$
  FinPour
  Fin

- **Fonction**: multiplication_ligne(M, i, λ)
  Action: multiplication de chacun des éléments du tableau M_i par λ
  Début
  Pour k **allant de** 0 **à** longueur$(M_i) - 1$ **faire**
  $M_{ik} \leftarrow \lambda M_{ik}$
  FinPour
  Fin

- **Fonction**: ajout_ligne(M, i, j, λ)
  Action: ajout à chaque élément du tableau M_i du produit de l'élément de même indice du tableau M_j par λ
  Début
  Pour k **allant de** 0 **à** longueur$(M_i) - 1$ **faire**
  $M_{ik} \leftarrow M_{ik} + \lambda M_{jk}$
  FinPour
  Fin

2)

$$\begin{pmatrix} 2 & 4 & -8 & 10 \\ -2 & -4 & 8 & 10 \\ 3 & 9 & -3 & 6 \\ 1 & 4 & 2 & -2 \end{pmatrix}$$

$$L_1 \leftarrow \tfrac{1}{2}L_1 \quad \begin{pmatrix} 1 & 2 & -4 & 5 \\ -2 & -4 & 8 & 10 \\ 3 & 9 & -3 & 6 \\ 1 & 4 & 2 & -2 \end{pmatrix}$$

$$L_2 \leftarrow L_2 + 2L_1 \quad \begin{pmatrix} 1 & 2 & -4 & 5 \\ 0 & 0 & 0 & 20 \\ 3 & 9 & -3 & 6 \\ 1 & 4 & 2 & -2 \end{pmatrix}$$

$$L_3 \leftarrow L_3 - 3L_1 \quad \begin{pmatrix} 1 & 2 & -4 & 5 \\ 0 & 0 & 0 & 20 \\ 0 & 3 & 9 & -9 \\ 1 & 4 & 2 & -2 \end{pmatrix}$$

$$L_4 \leftarrow L_4 - L_1 \quad \begin{pmatrix} 1 & 2 & -4 & 5 \\ 0 & 0 & 0 & 20 \\ 0 & 3 & 9 & -9 \\ 0 & 2 & 6 & -7 \end{pmatrix}$$

$$L_2 \leftrightarrow L_3 \quad \begin{pmatrix} 1 & 2 & -4 & 5 \\ 0 & 3 & 9 & -9 \\ 0 & 0 & 0 & 20 \\ 0 & 2 & 6 & -7 \end{pmatrix}$$

$$L_2 \leftarrow \tfrac{1}{3}L_2 \quad \begin{pmatrix} 1 & 2 & -4 & 5 \\ 0 & 1 & 3 & -3 \\ 0 & 0 & 0 & 20 \\ 0 & 2 & 6 & -7 \end{pmatrix}$$

$$L_1 \leftarrow L_1 - 2L_2 \quad \begin{pmatrix} 1 & 0 & -10 & 11 \\ 0 & 1 & 3 & -3 \\ 0 & 0 & 0 & 20 \\ 0 & 2 & 6 & -7 \end{pmatrix}$$

$$L_4 \leftarrow L_4 - 2L_2 \quad \begin{pmatrix} 1 & 0 & -10 & 11 \\ 0 & 1 & 3 & -3 \\ 0 & 0 & 0 & 20 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

$$L_3 \leftarrow \tfrac{1}{20}L_3 \quad \begin{pmatrix} 1 & 0 & -10 & 11 \\ 0 & 1 & 3 & -3 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

$$L_1 \leftarrow L_1 - 11L_3 \quad \begin{pmatrix} 1 & 0 & -10 & 0 \\ 0 & 1 & 3 & -3 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

$$L_2 \leftarrow L_2 + 3L_3 \quad \begin{pmatrix} 1 & 0 & -10 & 0 \\ 0 & 1 & 3 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \end{pmatrix}$$

$$L_4 \leftarrow L_4 + L_3 \quad \begin{pmatrix} 1 & 0 & -10 & 0 \\ 0 & 1 & 3 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

3) •

Fonction: choix_pivot(M, i, j)
Action: retourne l'indice du premier pivot non nul dans la colonne d'indice j du tableau à deux dimensions M à partir de l'indice de ligne i

```
Début
    k ← i
    TantQue k < longueur(M) et M_kj = 0 faire
        k ← k + 1
    FinTantQue
    Si k = longueur(M) alors
        Renvoyer −1
    sinon
        Renvoyer k
    FinSi
Fin
```

•

Fonction: gauss_jordan(M)
Action: retourne le tableau N obtenu par exécution de l'algorithme de Gauss-Jordan sur le tableau à deux dimensions M

```
Début
    N : tableau de taille longueur(M), longueur(M_0)
    Pour i allant de 0 à longueur(M) − 1 faire
        Pour j allant de 0 à longueur(M_i) − 1 faire
            N_ij ← M_ij
        FinPour
    FinPour
    i ← 0
    j ← 0
    TantQue i < longueur(N) et j < longueur(N_0) faire
        r ← choix_pivot(N, i, j)
        Si r = −1 alors
            j ← j + 1
        sinon
            Si r ≠ i alors
                echange_lignes(N, i, r)
            FinSi
            Si N_ij ≠ 1 alors
                multiplication_ligne(N, i, 1/N_ij)
            FinSi
            Pour k allant de 0 à longueur(N) − 1 faire
                Si k ≠ i et N_kj ≠ 0 alors
                    ajout_ligne(N, k, i, −N_kj)
                FinSi
            FinPour
            i ← i + 1
            j ← j + 1
        FinSi
    FinTantQue
    Renvoyer N
Fin
```

4) Pour l'unitarisation d'une ligne ainsi que pour une élimination, le nombre de multiplications est n. On en déduit que le nombre de multiplications de l'algorithme est majoré par $n(n+(n-1)n) = n^3$, la classe de complexité temporelle de l'algorithme de Gauss-Jordan est donc $O(n^3)$.

5)

$$
\begin{array}{rcl}
 & & \begin{pmatrix} 1 & 0 & 5 \\ 2 & 1 & 6 \\ 3 & 4 & 0 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\
L_2 & \leftarrow & L_2 - 2L_1 \quad \begin{pmatrix} 1 & 0 & 5 \\ 0 & 1 & -4 \\ 3 & 4 & 0 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \\
L_3 & \leftarrow & L_3 - 3L_1 \quad \begin{pmatrix} 1 & 0 & 5 \\ 0 & 1 & -4 \\ 0 & 4 & -15 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{pmatrix} \\
L_3 & \leftarrow & L_3 - 4L_2 \quad \begin{pmatrix} 1 & 0 & 5 \\ 0 & 1 & -4 \\ 0 & 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 5 & -4 & 1 \end{pmatrix} \\
L_1 & \leftarrow & L_1 - 5L_3 \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -4 \\ 0 & 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} -24 & 20 & -5 \\ -2 & 1 & 0 \\ 5 & -4 & 1 \end{pmatrix} \\
L_2 & \leftarrow & L_2 + 4L_3 \quad \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} -24 & 20 & -5 \\ 18 & -15 & 4 \\ 5 & -4 & 1 \end{pmatrix}
\end{array}
$$

6)

```
Fonction: inverse(M)
Action: retourne l'inverse de la matrice M dans le cas où celle-ci est inversible
Début
    N : tableau de taille longueur(M), longueur(M)
    P : tableau de taille longueur(M), longueur(M)
    Pour i allant de 0 à longueur(M) − 1 faire
        Pour j allant de 0 à longueur(M) − 1 faire
            N_ij ← M_ij
            Si i = j alors
                P_ij = 1
            sinon
                P_ij = 0
            FinSi
        FinPour
    FinPour
    Pour j allant de 0 à longueur(N) − 1 faire
        r ← choix_pivot(N, j, j)
        Si r ≠ j alors
            echange_lignes(N, j, r)
            echange_lignes(P, j, r)
        FinSi
        Si N_jj ≠ 1 alors
            p ← 1/N_jj
            multiplication_ligne(N, j, p)
            multiplication_ligne(P, j, p)
        FinSi
        Pour i allant de 0 à longueur(N) − 1 faire
            Si i ≠ j et N_ij ≠ 0 alors
                l ← −N_ij
                ajout_ligne(N, i, j, l)
                ajout_ligne(P, i, j, l)
            FinSi
        FinPour
    FinPour
    Renvoyer P
Fin
```

7)

$$\begin{cases} x + 2y - z = 2 \\ x + y - 2z = -3 \\ 2x + y - z = 1 \end{cases}$$

$$L_2 \leftarrow L_2 - L_1 \quad \begin{cases} x + 2y - z = 2 \\ -y - z = -5 \\ 2x + y - z = 1 \end{cases}$$

$$L_3 \leftarrow L_3 - 2L_1 \quad \begin{cases} x + 2y - z = 2 \\ -y - z = -5 \\ -3y + z = -3 \end{cases}$$

$$L_2 \leftarrow -L_2 \quad \begin{cases} x + 2y - z = 2 \\ y + z = 5 \\ -3y + z = -3 \end{cases}$$

$$L_1 \leftarrow L_1 - 2L_2 \quad \begin{cases} x - 3z = -8 \\ y + z = 5 \\ -3y + z = -3 \end{cases}$$

$$L_3 \leftarrow L_3 + 3L_2 \quad \begin{cases} x - 3z = -8 \\ y + z = 5 \\ 4z = 12 \end{cases}$$

$$L_3 \leftarrow \frac{1}{4}L_3 \quad \begin{cases} x - 3z = -8 \\ y + z = 5 \\ z = 3 \end{cases}$$

$$L_1 \leftarrow L_1 + 3L_3 \quad \begin{cases} x = 1 \\ y + z = 5 \\ z = 3 \end{cases}$$

$$L_2 \leftarrow L_2 - L_3 \quad \begin{cases} x = 1 \\ y = 2 \\ z = 3 \end{cases}$$

8)

```
Fonction: resolution_systeme(A, V)
Action: retourne la solution du système linéaire AU = V d'inconnue U dans le cas où celui-ci
        admet une unique solution.
Début
  N : tableau de taille longueur(A), longueur(A)
  P : tableau de taille longueur(V), 1
  U : tableau de taille longueur(V)
  Pour i allant de 0 à longueur(A) − 1 faire
    P_i0 ← V_i
    Pour j allant de 0 à longueur(A) − 1 faire
      N_ij ← A_ij
    FinPour
  FinPour
  Pour j allant de 0 à longueur(N) − 1 faire
    r ← choix_pivot(N, j, j)
    Si r ≠ j alors
      echange_lignes(N, j, r)
      echange_lignes(P, j, r)
    FinSi
    Si N_jj ≠ 1 alors
      p ← 1/N_jj
      multiplication_ligne(N, j, p)
      multiplication_ligne(P, j, p)
    FinSi
    Pour i allant de 0 à longueur(N) − 1 faire
      Si i ≠ j et N_ij ≠ 0 alors
        l ← −N_ij
        ajout_ligne(N, i, j, l)
        ajout_ligne(P, i, j, l)
      FinSi
    FinPour
  FinPour
  Pour i allant de 0 à longueur(V) − 1 faire
    U_i ← P_i0
  FinPour
  Renvoyer U
Fin
```

9)

Fonction: choix_pivot(M, i, j)
Action: retourne l'indice du pivot de valeur absolue maximale dans la colonne d'indice j du tableau à deux dimensions M à partir de l'indice de ligne i

```
Début
  p ← i
  Pour k allant de i + 1 à longueur(M) − 1 faire
    Si |M_kj| > |M_pj| alors
      p ← k
    FinSi
  FinPour
  Si M_pj = 0 alors
    Renvoyer −1
  sinon
    Renvoyer p
  FinSi
Fin
```

10)

Fonction: test_inversibilite(M)
Action: teste si la matrice carrée M est inversible

```
Début
  N : tableau de taille longueur(M), longueur(M_0)
  Pour i allant de 0 à longueur(M) − 1 faire
    Pour j allant de 0 à longueur(M_i) − 1 faire
      N_ij ← M_ij
    FinPour
  FinPour
  i ← 0
  j ← 0
  r ← 0
  TantQue i < longueur(N) et j < longueur(N_0) et r ≠ −1 faire
    r ← choix_pivot(N, i, j)
    Si r = −1 alors
      j ← j + 1
    sinon
      Si r ≠ i alors
        echange_lignes(N, i, r)
      FinSi
      Si N_ij ≠ 1 alors
        multiplication_ligne(N, i, 1/N_ij)
      FinSi
      Pour k allant de i + 1 à longueur(N) − 1 faire
        Si N_kj ≠ 0 alors
          ajout_ligne(N, k, i, −N_kj)
        FinSi
      FinPour
      i ← i + 1
      j ← j + 1
    FinSi
  FinTantQue
  Si r = −1 alors
    Renvoyer Faux
  sinon
    Renvoyer Vrai
  FinSi
Fin
```

11)

Fonction: test_compatibilite(A, V)
Action: teste si le système linéaire $AU = V$ d'inconnue U est compatible
Début
  N **: tableau de taille** longueur(A), longueur(A_0)
  P **: tableau de taille** longueur(V), 1
  Pour i **allant de** 0 **à** longueur(A) − 1 **faire**
    $P_{i0} \leftarrow V_i$
    Pour j **allant de** 0 **à** longueur(A_i) − 1 **faire**
      $N_{ij} \leftarrow A_{ij}$
    FinPour
  FinPour
  $i \leftarrow 0$
  $j \leftarrow 0$
  $n \leftarrow 0$
  TantQue $i <$ longueur(N) *et* $j <$ longueur(N_0) **faire**
    $r \leftarrow$ choix_pivot(N, i, j)
    Si $r = -1$ **alors**
      $j \leftarrow j + 1$
    sinon
      $n \leftarrow n + 1$
      Si $r \neq i$ **alors**
        echange_lignes(N, i, r)
      FinSi
      Si $N_{ij} \neq 1$ **alors**
        multiplication_ligne($N, i, \frac{1}{N_{ij}}$)
      FinSi
      Pour k **allant de** $i + 1$ **à** longueur(N) − 1 **faire**
        Si $N_{kj} \neq 0$ **alors**
          ajout_ligne($N, k, i, -N_{kj}$)
        FinSi
      FinPour
      $i \leftarrow i + 1$
      $j \leftarrow j + 1$
    FinSi
  FinTantQue
  $b \leftarrow$ Vrai
  $i \leftarrow n$
  TantQue b *et* $i <$ longueur(P) **faire**
    Si $P_{i0} \neq 0$ **alors**
      $b \leftarrow$ Faux
    sinon
      $i \leftarrow i + 1$
    FinSi
  FinTantQue
  Renvoyer b
Fin

12)

```
Fonction: noyau(M)
Action: retourne le noyau de la matrice M
Début
    N : tableau de taille longueur(M), longueur(M_0)
    Pour i allant de 0 à longueur(M) − 1 faire
        Pour j allant de 0 à longueur(M_i) − 1 faire
            N_ij ← M_ij
        FinPour
    FinPour
    i ← 0
    j ← 0
    l ← [ ]
    TantQue i < longueur(N) et j < longueur(N_0) faire
        r ← choix_pivot(N, i, j)
        Si r = −1 alors
            Ajout de j à la liste l
            j ← j + 1
        sinon
            Si r ≠ i alors
                echange_lignes(N, i, r)
            FinSi
            Si N_ij ≠ 1 alors
                multiplication_ligne(N, i, 1/N_ij)
            FinSi
            Pour k allant de 0 à longueur(N) − 1 faire
                Si k ≠ i et N_kj ≠ 0 alors
                    ajout_ligne(N, k, i, −N_kj)
                FinSi
            FinPour
            i ← i + 1
            j ← j + 1
        FinSi
    FinTantQue
    L : tableau de taille longueur(N_0), longueur(l)
    k = 0
    Pour i allant de 0 à longueur(L) − 1 faire
        Si i ∈ l alors
            Pour j allant de 0 à longueur(L_i) − 1 faire
                Si i = l_j alors
                    L_ij ← 1
                sinon
                    L_ij ← 0
                FinSi
            FinPour
        sinon
            Pour j allant de 0 à longueur(L_i) − 1 faire
                L_ij ← −N_{k l_j}
            FinPour
            k ← k + 1
        FinSi
    FinPour
    Renvoyer L
Fin
```