Chapitre 1

Introduction au calcul matriciel et à ses applications

1.1 Notion de matrice

1.1.1 Introduction

Une matrice $(m \times n)$ est un ensemble d'éléments, les *coefficients* ou les *composantes*, rangés en tableau à m lignes et n colonnes de la forme

$$\begin{array}{ccc} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \dots & a_{mn} \end{array}$$

Notation On note une matrice par une lettre capitale et on la représente par un tableau à m lignes et n colonnes :

$$A = \begin{pmatrix} a_{11} & \dots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \dots & a_{mn} \end{pmatrix}$$

Les éléments a_{ij}, où i est l'indice de ligne et j celui de la colonne peuvent être pris dans divers ensembles comme par exemple :

- les entiers $\mathbb{N}$ pour l'optimisation entière par exemple
- les réels $\mathbb{R}$ ou les complexes $\mathbb{C}$ pour l'analyse
- les polynômes réels en λ : $\mathbb{R}[\lambda]$

- les fractions rationnelles en λ : $\mathbb{F}[\lambda]$
- les matrices de taille $(k \times l)$
- les corps finis quelconques $\mathbb{Z}/2\mathbb{Z}$ pour la cryptographie, les codes, en informatique théorique

Exemple Soit M matrice à coefficients dans $\mathbb{Z}/2\mathbb{Z}$. Par exemple

$$M = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

L'ensemble $\mathbb{Z}/2\mathbb{Z}$ est l'ensemble des restes modulo 2 muni de l'addition et de la multiplication modulo 2. C'est un corps servant à écrire les codes binaires.

Remarque Dans $\mathbb{Z}/2\mathbb{Z}$,
- $0 + 0 = 0$, $0 + 1 = 1$ et $1 + 1 = 0$
- $0 * 0 = 0$, $0 * 1 = 0$ et $1 * 1 = 1$.

Donc $\sum_{i=1}^{n} x_i = 1$ si on a un nombre impair de x_i égaux à 1.
Attention, ne pas confondre avec l'ensemble des booléens $B = 0',1'$ où $\sum_{i=1}^{n} x_i = 1$ dès que un des x_i est égal à 1.

1.1.2 Définition mathématique d'un tableau matriciel

Définition 1.1.1 *On appelle* matrice *à m lignes et n colonnes à coefficients dans un corps $\mathbb{K}$ une suite $(a_{ij})_{1 \leq i \leq m,\ \ 1 \leq j \leq n}$*
c'est à dire une application A :

$$[1,m] \to \mathbb{K} \qquad (i,j) \to A(i,j) = a_{ij}$$

L'ensemble des matrices à m lignes et n colonnes à coefficients dans $\mathbb{K}$ est noté $\mathcal{M}_{m,n}(\mathbb{K})$.

Quelques définitions et exemples
- les matrices carrées : $m = n$
- les matrices creuses :
  - la matrice nulle
  - la matrice identité
  - les matrices diagonales
  - les matrices tridiagonales
  - les matrices triangulaires supérieures ou inférieures

• les matrices élémentaires E_{ij} : ses coefficients sont nuls sauf celui placé sur la i-ème ligne et j-ème colonne. Ces matrices forment une base de $\mathcal{M}_{m,n}$.

• les matrices de Hessenberg supérieure ou inférieure où $a_{ij} = 0$ pour $i > j + 1$.
Ces matrices présentent un intérêt au point de vue du temps de traitement ou place de mémoire (cf Matlab qui exploite cette caractéristique).

– les matrices partitionnées de la forme $A = [A_{ij}]$ où les matrices A_{ij} sont des blocs de dimension $m_i \times n_j$ tels que $\sum m_i = m$ et $\sum n_j = n$. Le partitionnement permet, lorsqu'il est choisi judicieusement, d'accélérer les calculs ou de réduire les formules.

Exemple

$$A = \begin{pmatrix} 3 & 4 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} B & 0 \\ 0 & I_2 \end{pmatrix}$$

avec

$$B = \begin{pmatrix} 3 & 4 \\ 1 & 2 \end{pmatrix}$$

– Dans la plupart des cas, on considère des matrices à coefficients constants (donc des opérations interprétables en tant que transformation linéaire), le langage matriciel permet d'appéhender des objets plus généraux

• Soit f une fonction réelle de plusieurs variables de $\mathbb{R}^n$ dans $\mathbb{R}$ qui à $(x_1,..,x_n)$ associe $f(x_1,..x_n)$.
On définit le *gradient* de cette fonction comme une matrice ligne

$$\frac{\partial f}{\partial x} = [\frac{\partial f}{\partial x_1},..,\frac{\partial f}{\partial x_n}]$$

et sa *matrice Hessienne*

$$H_f = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}$$

• Soit f une fonction vectorielle de plusieurs variables de $\mathbb{R}^n$ dans $\mathbb{R}^m$ qui à $(x_1,..,x_n)$ associe $(f_1(x_1,..x_n),..,f_m(x_1,..x_n))$.

On définit la matrice jacobienne *Jacobienne*

$$J_f = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots & \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}$$

Application : en optimisation.

Cette écriture matricielle permet de traduire, de façon concise et claire, de nombreuses opérations que l'on peut rencontrer dans les Sciences de l'Ingénieur.
C'est un formalisme simple qui permet d'une part de généraliser des manipulations scalaires (le seul point délicat est le perte de commutativité) et d'autre part de pouvoir manipuler dans un seul langage scalaires, vecteurs, tableaux.

1.1.3 Une application

Exemple du problème d'affectation.

Soient M_1,M_2,M_3,M_4 4 machines à affecter aux tâches T_1,T_2,T_3,T_4. Soient a_{ij} les coûts d'affectation de M_i à T_j :

	T_1	T_2	T_3	T_4
M_1	8	3	1	5
M_2	11	7	1	6
M_3	7	8	6	8
M_4	11	6	4	9

Toute machine M_i doit effectuer une seule tâche $T_{\phi(i)}$.

Problème Trouver une affectation de coût minimum ie une bijection ϕ de {1,2,3,4} dans {1,2,3,4}.

Procédé naturel : On choisit M_i puis la tâche de moindre coût parmi celles qui restent.

Par exemple : $M_1 \to T_3$, $M_2 \to T_4$, $M_3 \to T_1$ et $M_4 \to T_2$. On obtient un coût total de $1 + 6 + 7 + 6 = 20$.
Est-ce bien l'affectation de coût minimum?

Si l'on veut faire une énumération exhaustive, il faut faire $4 * 3 * 2 * 1 = 4!$ tests. Ce qui devient vite impossible dès que n est grand.

1.2 Opérations élémentaires

1.2.1 Egalité

Les matrices A et B sont égales ssi elles ont même dimension et $\forall i, j,\ a_{ij} = b_{ij}$.

1.2.2 Somme

On définit la somme de 2 matrices A et B par la matrice $A + B$ de coefficients $(a_{ij} + b_{ij})$.

Propriétés
$\bullet A + B = B + A$
$\bullet A + 0 = 0 + A = A$
$\bullet -A = (-a_{ij}$
$\bullet -A + A = 0$

1.2.3 Multiplication par un scalaire

On définit la multiplication de A par un scalaire k par la matrice kA de terme général ka_{ij}.

Remarque Lorsque $\mathbb{K}$ est un corps, l'addition et la multiplication par un scalaire munissent l'ensemble $\mathcal{M}_{m,n}(\mathbb{K})$ d'une structure d'espace vectoriel de dimension $dim(\mathcal{M}_{m,n}(\mathbb{K})) = m \times n$ sur $\mathbb{K}$.

1.2.4 Transposition

La notion de transposition vient du produit scalaire (aussi appelé produit intérieur) de deux vecteurs de même dimension n : $(x,y) \rightarrow < x,y > = \sum_{i=1}^{n} x_i y_i$.
Soit $A \in \mathcal{M}_{mn}$. Si on calcule $< x,Ay >$ alors la matrice B, *transposée* de A est l'unique matrice telle que $< x,Ay > = < Bx,y >$. Ainsi la transposée de A est l'unique matrice $(n \times m)$ notée A^T dont le coefficient correspondant à la i-ème ligne et là j-ème colonne est a_{ji}.

Définition 1.2.1 *Lorsque $m = n$ et $A^T = A$, on dit que A est* symétrique

et $A^T = -A$ *on dit que* A *est* antisymétrique.

Lorsque $\mathbb{K} \subset \mathbb{C}$, on définit la matrice *adjointe* ou *transconjuguée* par $A^\star = \bar{A}^T$ (où $\bar{A} = (\bar{a_{ij}})_{1\leq i\leq n,\ 1\leq j\leq n}$ est la conjuguée).

Définition 1.2.2 *Lorsque* $m = n$ *et* $A^\star = A$*, on dit que* A *est* hermitienne *et* $A^\star = -A$ *on dit que* A *est* antihermitienne.

Exemple -Si $A = \begin{pmatrix} 8 & 3 & 1 \\ 1 & 7 & 1 \\ 7 & 8 & 6 \end{pmatrix}$ alors $A^T = \begin{pmatrix} 8 & 1 & 7 \\ 3 & 7 & 8 \\ 1 & 1 & 6 \end{pmatrix}$

-Si $B = \begin{pmatrix} 1 & 5 \\ 2 & 6 \\ 3 & 7 \\ 4 & 8 \end{pmatrix}$ alors $B^T = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{pmatrix}$

- les matrices symétriques carrées d'ordre 3 sont de la forme

$$\begin{pmatrix} a & d & f \\ d & b & e \\ f & e & c \end{pmatrix}$$

Propriétés
-$(A+B)^T = A^T + B^T$ et $(A+B)^\star = A^\star + B^\star$
-$(A^T)^T = A$ et $(A^\star)^\star = A$
-$(A^\star)^T = (A^T)^\star = \bar{A}$

1.2.5 Trace

On définit la *trace* d'une matrice A par $tr(A) = \sum_{i=1}^{\min(m,n)} a_{ii}$.

Propriétés
-$tr(\alpha A + \beta B) = \alpha tr(A) + \beta tr(B)$
-$tr(A^T) = tr(A)$
-$tr(A^\star) = tr\hat{(}A)$

1.2.6 Dérivation et intégration

On dérive et on intgre terme à terme.

1.2.7 Partionnement

C'est l'organisation de la matrice A en sous-matrices.

Exemple 1) organisation de A en colonnes :
$A = [A_1 ... A_n]$ où A_i est la i-ème colonne de A.

2) organisation de A en lignes :

$$A = \begin{pmatrix} \alpha_1^T \\ \vdots \\ \alpha_m^T \end{pmatrix}$$

où α_i^T est la i-ème ligne de A.

1.2.8 Construction de mineures

Une mineure de A est une matrice extraite de A de la forme :

$$M = \begin{pmatrix} a_{i_1 j_1} & \dots & a_{i_1 j_\nu} \\ \vdots & & \vdots \\ a_{i_\mu j_1} & \dots & a_{i_\mu j_\nu} \end{pmatrix}$$

avec $1 \leq i_1 < ... < i_\mu \leq m$ et $1 \leq j_1 < ... < j_\nu \leq n$.

Notation habituellement utilisée pour définir une matrice carrée (ie $\mu = \nu$) extraite de A :

$$M = A \begin{pmatrix} i_1 & i_2 & \dots & i_\mu \\ j_1 & j_2 & \dots & j_\mu \end{pmatrix}$$

Définition 1.2.3 *Lorsque $\mu = \nu$ et que $i_k = j_k$ $forall k$, on dit que la mineure est* centrée.
Si de plus, $i_\mu = i_1 + \mu - 1$, on dit que la mineure est principale.

Exemple Soit $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$. Alors $M = \begin{pmatrix} 1 & 2 & 3 \\ 7 & 8 & 9 \end{pmatrix}$ est une mineure de A.
$M = \begin{pmatrix} 1 & 3 \\ 7 & 9 \end{pmatrix}$ est centrée et $M = \begin{pmatrix} 5 & 6 \\ 8 & 9 \end{pmatrix}$ est principale.

On retrouve les matrices extraites dans la formule d'inversion d'une matrice carrée par exemple.

1.2.9 Retour au problème d'affectation

Soit $A = \begin{pmatrix} 8 & 3 & 1 & 5 \\ 11 & 7 & 1 & 6 \\ 7 & 8 & 6 & 8 \\ 11 & 6 & 4 & 9 \end{pmatrix}$.

Le coût minimum d'affectation pour T_1 est 7, pour T_2 3, pour T_3 1 et pour T_4 5. Décomposons alors la matrice A en la somme d'une matrice colonnes constantes et d'un reste. On obtient

$$A = \begin{pmatrix} 7 & 3 & 1 & 5 \\ 7 & 3 & 1 & 5 \\ 7 & 3 & 1 & 5 \\ 7 & 3 & 1 & 5 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 & 0 \\ 4 & 4 & 0 & 1 \\ 0 & 5 & 5 & 3 \\ 4 & 3 & 3 & 4 \end{pmatrix}$$

On fait de même avec les lignes du reste :

$$A = \begin{pmatrix} 7 & 3 & 1 & 5 \\ 7 & 3 & 1 & 5 \\ 7 & 3 & 1 & 5 \\ 7 & 3 & 1 & 5 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 3 & 3 & 3 & 3 \end{pmatrix} + \begin{pmatrix} 1 & 0 & 0 & 0 \\ 4 & 4 & 0 & 1 \\ 0 & 5 & 5 & 3 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

On a donc une décomposition de A en la somme de 3 matrices : une colonnes-constante, une lignes-constante et une matrice reste

$$A = C + L + R$$

et le coût d'une affectation par rapport à A est

$$C/A = C/C + C/L + C/R \geq C/C + C/L = 16 + 3 = 19$$

En affectant M_1 à T_4, M_2 à T_3, M_3 à T_1 et M_4 à T_2, le coût du reste est 0 et on a donc l'affectation de coût minimal ($C/A = 19$).

Chapitre 2

Notions sur les codes

2.1 Introduction

2.1.1 Erreur de transmission

Lorsque l'on veut communiquer des informations, il faut les coder et pour cela on utilise des sons, des lettres, des chiffres...En informatique et dans les télécommunications, le codage se fait habituellement à l'aide de 0 et 1 transmis sous forme de courants électriques, d'ondes radios, de signaux lumineux...
Malheureusement, quand la transmission se prolonge des erreurs finissent toujours par apparaitre (risque de parasitage dus à l'imperfection de tout système de transport d'informations).
Sachant que nous ne pourrons éviter ces déformations de message que nous allons transmettre, on peut se poser plusieurs questions:

- Comment peut-on coder le message de sorte que l'on puisse détecter si le message reçu est correct i.e. s'il n'y a pas eu d'erreur lors de la transmission?
- Dans le cas où on a détecté des erreurs, peut-on trouver un codage qui permette de rectifier?

Nous allons décrire dans ce qui suit différentes méthodes pour répondre à ces questions. Les codages remplissant ces fonctions s'appellent les *codes correcteurs*

2.1.2 Probabilité d'erreur

On travaillera à partir de maintenant avec des bits.
Un dispositif permettant de transmettre des bits s'appelle un *canal binaire*. Il comporte un risque d'erreur qui se mesure par une probabilité. On définit cette probabilité en considérant l'expérience (transmission d'un bit) comme un phénomène aléatoire qui comporte deux issues:
-ou bien le bit est bien transmis,
-ou bien il ne l'est pas.
Soit p la probabilité d'erreur. Donc un bit est bien transmis avec la probabilité $q = 1 - p$.
En général, p est très petit. C'est ce que l'on espère en tout cas. On supposera aussi que p a la même valeur pour la transmission d'un 0 ou d'un 1. On dit alors que le canal binaire est *symétrique*.
On suppose que le canal est *sans mémoire* i.e. la transmission de chaque bit est indépendante des autres.

Théorème 2.1.1 *Lorsqu'on transmet n bits sur un canal binaire sans mémoire*
1) la probabilité que toutes les erreurs soient commises sur u bits désignés à l'avance est $p^u q^{n-u}$
2) la probabilité que le nombre de bits mal transmis soit r est $C_n^r p^r q^{n-r}$.

2.1.3 Exemples

Test de parité avec un bit de parité

0	1	0	1	1	0	0	1

On rajoute un bit de parité au message : 0 si la somme des bits est paire, 1 sinon.
On détecte donc un nombre impair d'erreurs. Ce codage permet de détecter toute configuration d'une erreur mais pas de savoir sur quel bit l'erreur s'est produite. On ne peut donc pas corriger.

Exemple Bits par blocs de 3.
001 est codé en 0011
110 est codé en 1100...
Si on reçoit 1011, on a la certitude qu'il y a une ou trois erreurs. Mais on ne sait pas où.
Si on reçoit 1111, le message sempble correct, toutefois on n'est pas certain

que ce soit celui qui a été transmis car 2 ou 4 erreurs ont pu se produire. Dans le doute on considère que c'est le bon. Quel risque prend-on en agissant ainsi? On a
-$\mathbb{P}$(message bien transmis) $= q^4$
-$\mathbb{P}$(message mal transmis) $= 1 - q^4$
-$\mathbb{P}$(message contienne 2 erreurs) $= C_4^2 p^2 q^2 = 6p^2q^2$
-$\mathbb{P}$(message contienne 4 erreurs) $= p^4$
Ainsi la probabilité que le receveur se trompe en agissant ainsi est $6p^2q^2+p^4$. En assimilant probabilité et proportion, nous voyons que parmi les messages transmis, la proportion de ceux erronés qui ne sont pas détectés est

$$\frac{6p^2q^2 + p^4}{1 - q^4} \sim 1{,}5p$$

Et si $p = 10\%$ cette proportion est égale à environ 14,1%.

Conclusion: En transmettant des blocs d'information sans bits de parité, la proportion de messages mal transmis qui ne sont pas détectés est de 100%. Donc le codage précédent améliore la détection d'erreurs. Par contre, il ne permet pas la correction.

Codage par répétition

Ici on transmet les bits 1 par 1 et pour donner une allure caractéristique aux messages on triple chaque bit. Donc 1 est codé en 111 et 0 en 000.
Si le destinataire reçoit deux 0 et un 1 ou deux 1 et un 0, alors il a la certitude que des bits sont faux i.e. mal transmis. Au contraire s'il reçoit 3 fois le même bit, le message lui semble correct et il l'accepte pour bon. Quel risque prend-il en agissant ainsi? On a
-$\mathbb{P}$(message reçu soit faux) $= 1 - q^3$
-$\mathbb{P}$(destinataire ne s'en aperçoive pas) $= p^3$
Donc la proportion de messages faux non détectés est

$$\frac{p^3}{1 - q^3}$$

Possibilité de correction :
• Si les 3 bits sont identiques, le receveur garde le message et la probabilité pour qu'il se trompe est p^3.
• S'il reçoit 011 ou 101 ou 110, il sait que le message est faux. Le message le plus vraisemblable étant 111, il le corrige. En agissant ainsi, il se trompe

quand 2 bits ont été mal transmis ce qui arrive avec la probabilité $3p^2q$. Ainsi la probabilité qu'après correction le message soit encore faux est $p^3 + 3p^2q << p$.

Conclusion

Plusieurs idées se dégagent de ces deux exemples :
a) pour transmettre un bloc de bits, on le code en lui ajoutant quelques bits choisis de façon à lui donner une allure caractéristique.
b) quand le destinataire reçoit un message qui n'a pas cette allure caractéristique il est certain que le message est faux. Mais il ne sait pas où est l'erreur. Quand un message ressemble plus que tous les autres valable au message reçu alors il corrige en le remplaçant par ce message. Mais il peut faire des erreurs. D'autre part, si plusieurs messages valables sont en concurrence pour la correction, il fait un choix mais une fois encore il peut se tromper.
c) quand le message reçu présente toutes les caractéristiques d'un message valable, il l'accepte comme vrai mais peut-être à tort.
d) on calcule la probabilité qu'un message reste faux après correction. Plus cette probabilité est faible, meilleur est le code. Mais attention à ne pas trop ralentir la vitesse de transmission.

2.2 Codes en blocs

2.2.1 Mot de code

Une technique utilisée pour transmettre des bits est le codage en blocs. On regroupe les bits en blocs de même longueur et on code les blocs avant de les transmettre.

Définition 2.2.1 *On appelle* bloc *de dimension k toute suite de k bits. Il y en a 2^k.*
Pour transmettre un bloc, on lui adjoint r bits supplémentaires appelés bits de contrôle.
La suite de $n = k+r$ bits ainsi obtenue est appelée mot de code. *L'ensemble des mots de code est le* code.
Le rendement *est défini par $\frac{k}{n}$ (pour l'exemple 1, $rt = 3/4$ et pour l'exemple 2, $rt = 1/3$).*
Un message *est une suite de $n = k + r$ bits. Il y en a 2^n. La* longueur

du message est n. L'ensemble des messages est $\mathcal{B}^n$ *où* $\mathcal{B}^1 = \{0,1\}$, $\mathcal{B}^2 = \{00,01,10,11\}$...

Les mots de code sont donc des messages particuliers.

Représentation géométrique des mots de code Diagramme de Hasse

Pour l'exemple 2, $k = 1$, $r = 2$ et $n = 3$. Le code est $\{000,111\}$ et l'ensemble des messages est $\{000,001,010,100,110,101,011,111\}$ de cardinal $2^n = 8$.

2.2.2 Distance et poids de Hamming

Distance de Hamming

On définit la notion de distance afin de doner un sens à celle de proximité.

Définition 2.2.2 *Soient a et b deux messages. On appelle* distance de Hamming *entre a et b le nombre de fois où les symboles occupant la même position dans a et b diffèrent.*

Elle est notée $d(a,b)$.

Exemple
$d(001,100) = 2$
$d(brebis,gradin) = 4$.

Propriétés La distance de Hamming vérifie les propriétés d'une distance :
1) positivité : $d(a,b) \geq 0$
2) symétrie : $d(a,b) = d(b,a)$
3) annulation : $d(a,b) = 0 \Leftrightarrow a = b$
4) inégalité triangulaire : $d(a,b) + d(b,c) \geq d(a,c)$

Représentation géométrique $d(a,b)$ représente le nombre d'arêtes qu'il faut longer pour se rendre de a à b par le plus court chemin possible dans

le diagramme de Hasse.

Définition 2.2.3 *La* distance minimale du code *est la plus petite distance que l'on peut trouver entre tous les couples de mots de code différents.*

Poids de Hamming

Définition 2.2.4 *Le* poids de Hamming *est le nombre de symboles différents de 0.*

Il est noté $p(a)$.

Exemple $p(23040) = 3$

Propriété $d(a,b) = p(a - b)$
où $a - b$ représente la différence entre les vecteurs a et b composante par composante.

Exemple
Si $a = [23983]$ et $b = [21884]$, alors $a - b = [0210 - 1]$, $d(a,b) = 3$ et $p(a - b) = 3$.

Cas particulier du binaire
En binaire, "$+ = - = \oplus$" et par conséquent,

$$d(a,b) = p(a,b)$$

Définition 2.2.5 *Le* poids minimal du code *est le plus petit poids non nul que l'on peut trouver parmi les mots de code.*

2.2.3 Détection et correction des erreurs

Notation On notera

- un mot quelconque du code avec la lettre c et les bits qui le composent par les c_i : $c = [c_1..c_n]$
- une configuration d'erreur e : $e = [e_1..e_n]$
- un mot reçu $c^\star$: $c^\star = [c_1^\star..c_n^\star]$

Ainsi $c^\star = c + e$

Etude de cas

Si $d_{min} = 1$, on ne peut rien détecter.

Si $d_{min} = 2$,

- si la probabilité qu'il y ait trois erreurs est non négligeable alors on ne peut rien détecter.
- si $\mathbb{P}(2erreurs)$ est non négligeable alors on ne peut rien détecter.
  Dans ces deux cas le code est inutilisable.
- si $\mathbb{P}(2erreurs)$ est négligeable alors le code peut détecter une erreur. Par contre, on ne pourra pas corriger.
  ex : soit le code $\mathcal{C} = \{[000],[101],[110],[011]\}$. Si on reçoit $c^\star = [011]$, on le garde, on considère qu'il n'y a pas eu d'erreur. Par contre, si on reçoit $c^\star = [001]$, on détecte une erreur mais on ne sait pas corriger.
- si $\mathbb{P}(1erreur)$ est négligeable alors le code est inutile.

Si $d_{min} = 3$,

- si la probabilité qu'il y ait trois erreurs est négligeable alors on peut faire 0, 1 ou 2 erreurs lors de la transmission.
  ex : soit le code $\mathcal{C} = \{[000],[111]\}$. Si on reçoit $c^\star = [011]$, on détecte une erreur et alors $c = [111]$, ou deux erreurs et alors $c = [000]$. On ne sait pas corriger.
- si $\mathbb{P}(2erreurs)$ est négligeable alors on peut faire une erreur.

On remarque donc que la correction et la détection dépendent bien de la distance minimale du code. Voici le théorème fondamental du codage précisant ce lien.

Théorème 2.2.1 *Théorème de Hamming.*
1) pour que le codage détecte tous les messages faux dont le nombre d'erreurs est compris entre 1 et N, il faut et il suffit que la distance minimale d_{min} soit strictement supérieure à N.
2) pour que tout message ayant N erreurs ou moins soit correctement corrigé, il faut et il suffit que d_{min} soit strictement supérieure à N.

Remarque On a l'habitude de noter t le plus grand entier inférieur ou égal à $\frac{d-1}{2}$. Tout message faux ayant t erreurs ou moins est correctement corrigé

et il existe des messages faux ayant plus de t erreurs qui sont mal corrigés. C'est pourquoi t est appelé le *nombre d'erreurs mal corrigées du code.*

2.3 Les codes linéaires

Dès que l'on désire construire un code de longueur n vérifiant certaines propriétés de détection et de correction, on est amené à choisir un sous-ensemble de $\mathbb{K}^n$ de points convenablement espacés, points qui devront être faciles à repérer, où l'on note $\mathbb{K}$ le corps : l'alphabet. Pour les codes binaires, $\mathbb{K} = \{0,1\}$.
Une idée est de prendre pour $\mathcal{C}$ un sous-espace vectoriel de $\mathbb{K}^n$. Un tel code est dit *linéaire.*
On dira qu'un code linéaire est de *type* $[n,k,d]$ s'il est de longueur n, de dimension k et de distance minimale d.

Exemple Les codes des exemples précédents sont bien linéaires.
Le code de parité est de type [4,3,2] et celui à répétition de type [3,1,3].

A partir de maintenant, nous n'étudierons plus que les codes linéaires et binaires.

Propriétés $d_{min} = p_{min}$
La somme modulo 2 de 2 mots du code est encore un mot du code.
La plus courte distance qui sépare un mot a du code à tous les autres mots du code est la même pour tout a.
$d_{min} = d_{min}$(0,tout mot).

Remarque Coder un bloc c'est lui associer un message. Un codage peut donc s'interpréter comme une application de $\mathbb{K}^k$ dans $\mathbb{K}^n$. On appellera cette application l'*application de codage.*

Remarquons que le codage est linéaire si l'application f de codage est linéaire ie si $f(a \oplus b) = f(a) \oplus f(b) \quad \forall a,b$.

On se place en binaire. Posons $b_i = [000..1..00]$ pour tout $i \in [1,k]$, où le 1 est placé en i-ème position. Les b_i forment une base de $(\mathbb{Z}/2\mathbb{Z})^k$.
Alors tout bloc $\alpha_1\alpha_2\alpha_3..\alpha_k$ s'écrit

$$\alpha_1\alpha_2\alpha_3..\alpha_k = \alpha_1 b_1 \oplus \alpha_2 b_2 \oplus \alpha_3 b_3 .. \oplus \alpha_k b_k$$

La propriété fondamentale des codages linéaires est qu'il suffit de connaitre les mots de code $f(\alpha_1)$,$f(\alpha_2)$,$f(\alpha_3)$,..,$f(\alpha_k)$ pour connaitre le mot de code associé à n'importe quel bloc puisque

$$\begin{aligned} f(\alpha_1\alpha_2\alpha_3..\alpha_k) &= f(\alpha_1 b_1 \oplus \alpha_2 b_2 \oplus \alpha_3 b_3.. \oplus \alpha_k b_k) \\ &= f(\alpha_1 b_1) \oplus f(\alpha_2 b_2) \oplus f(\alpha_3 b_3) \oplus .. \oplus f(\alpha_k b_k) \\ &= f(\alpha_1) b_1 \oplus f(\alpha_2) b_2 \oplus f(\alpha_3) b_3 \oplus .. \oplus f(\alpha_k) b_k \quad (2.1) \end{aligned}$$

Pour définir un code linéaire $\mathcal{C}$ de type [n,k,d], il suffit donc de se donner une famille de k vecteurs linéairement indépendants de $(\mathbb{Z}/2\mathbb{Z})^n$

$$(f(b_1),f(b_2),..,f(b_k)) = (g_1,g_2,..,g_k)$$

que l'on consigne horizontalement dans une matrice G $k \times n$ dite *matrice génératrice du code.*

Exemple 1) pour le code de parité
$f(b_1) = f(100) = 1001$,$f(b_2) = f(010) = 0101$,$f(b_3) = f(001) = 0011$ et donc

$$G = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix}$$

Les trois premières colonnes donnent les bits d'information tandis que la dernière $P = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$ donne les bits de contrôle.

2) pour le code à répétition
$f(b_1) = f(1) = 111$ et donc

$$G = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}$$

Ici c'est la première colonne qui donne les bits d'information tandis que ce sont les deux dernières qui donnent les bits de contrôle : $P = \begin{pmatrix} 1 & 1 \end{pmatrix}$.

Remarque On a ainsi $G^T = [f]_{\textbf{dans les bases canoniques}}$.
Connaissant la matrice génératrice G d'un code linéaire, il est facile de calculer le mot de code associé à un bloc b. En effet, d'après (**??**), il suffit de faire la somme modulo 2 des lignes de G qui correspondent aux bits égaux à 1 dans b.

Exemple Pour le codage de parité, le mot de code associé à 101 est 1010. On obtient le même résultat en ajoutant la 1ère et la 3ème ligne de G ou en faisant

$$f(101) = f(100 \oplus 001) = f(100) \oplus f(001) = 1001 \oplus 0011 = 1010$$

Chapitre 3

Combinaison linéaire et produit matriciel

3.1 Produit matriciel

3.1.1 Vecteurs et combinaisons linéaires

Définition 3.1.1 *1) Soit E un espace vectoriel sur le corps $\mathbb{K}$. Soient λ_1,..,$\lambda_p \in \mathbb{K}$ et x_1,..,$x_p \in E$. Alors on dit que $\sum_{i=1}^{p} \lambda_i x_i$ est* combinaison linéaire *(CL) des x_i.*
2) Une famille de vecteurs $\{x_1,..,x_p\}$ d'un ev E est dite génératrice *si $E = Vect\{x_1..x_p\}$ ie*

$$\forall x \in E \quad \exists \quad \lambda_1,..,\lambda_p \in \mathbb{K} \quad / \quad x = \sum_{i=1}^{p} \lambda_i x_i$$

On dit que tout x de E se décompose *sur les vecteurs x_i ou encore que tout x de E est combinaison linéaire des vecteurs x_i.*

ex : dans $\mathbb{R}^2$, $\{(1,0),(0,1)\}$ est une famille génératrice.

Un espace vectoriel est de dimension finie *s'il existe une famille génératrice finie.*
3) Une famille $\{x_1,..,x_p\}$ est libre si

$$\lambda_1 x_1 + .. + \lambda_p x_p = 0 \Rightarrow \lambda_1 = .. = \lambda_p = 0$$

On dit que les vecteurs sont linéairement indépendants. *Si la famille n'est pas libre, on dit qu'elle est* liée.

ex : $x_1 = (1,2,1)$, $x_2 = (-1,3,1)$ et $x_3 = (-1,13,5)$ sont liés car $2x_1 + 3x_2 = x_3$.
4) On appelle une base *de E toute famille libre et génératrice.*

Proposition 3.1.1 *Une famille $\{x_1,..,x_p\}$ est une base de E ssi tout x de E se décompose de manière unique sur les vecteurs x_i.*

Théorème 3.1.1 *Théorème d'existence de base*
Soit $E \neq \{0\}$ un espace vectoriel de dimension finie. Alors
1) de toute famille génératrice on peut extraire une base.
2) toute famille libre peut être complétée en une base.

Propriété-Définition Toutes les bases ont même nombre d'éléments. On appelle ce nombre la *dimension* de E et on le note $dim_{\mathbb{K}}(E)$.

Conséquence Soit E un ev de dimension n. Alors
1) Toute famille génératrice ayant n éléments est libre.
2) Toute famille libre ayant n éléments est génératrice.

Cas particuliers : 1) cas de deux vecteurs v_1 et v_2 :

- ou bien ils sont libres
- ou bien $v_1 = \lambda v_2$

2) dans $\mathbb{R}^3$, si v_1 et v_2 sont deux vecteurs,

$$\text{ils sont libres} \quad \Leftrightarrow v_1 \wedge v_2 = 0$$

3) Si $\mathbb{K} = \mathbb{Z}_2$, les CL sont $\sum_{i=1}^{p} \lambda_i x_i = \sum_{\sigma} x_i$ où σ est le support de la suite des λ_i ie $\sigma = \{i/\lambda_i \neq 0\} = \{i/\lambda_i = 1\}$

3.1.2 Le produit matriciel

Définition 3.1.2 *Le* produit $C = AB$ *de deux matrices $A \in \mathcal{M}_{n,p}(\mathbb{K})$ et $B \in \mathcal{M}_{p,m}(\mathbb{K})$ est une matrice de $\mathcal{M}_{n,m}(\mathbb{K})$ de terme général*

$$c_{ij} = \sum_{k=1}^{p} a_{ik} b_{kj}$$

CL des a_{ik} et b_{kj}.

Remarque AB n'a de sens et n'est défini que si le nombre de colonnes de A est égal au nombre de lignes de B. Par exemple, AB peut exister alors que BA non (dès que $m \neq n$).

Propriétés

- associativité : $A(BC) = (AB)C$
- distributivité par rapport à la loi $+$: $(A+B)C = AC + BC$ et $A(B+C) = AB + AC$
- **attention** non commutativité : $AB \neq BA$ en général
- existence de diviseurs de 0 : $AB = 0$ n'implique pas nécéssairement que $A = 0$ ou $B = 0$.
  Conséquence : $AB = AC$ n'implique pas que $B = C$

Définition 3.1.3

- *Si* $AA^T = A^T A$, *on dit que* A *est* normale.
- *Si* $AA^* = A^* A$, *on dit que* A *est* normale complexe.
- *Si* $\exists k > 1,\ A^k = A$, *on dit que* A *est* idempotente.
- *Si* $\exists k > 1,\ A^k = 0$, *on dit que* A *est* nilpotente.

Propriétés Lorsque les produits sont compatibles

- $(AB)^T = B^T A^T$
- $trace(AB) = trace(BA)$
- $trace(AA^T) = trace(A^T A) = \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}^2$

L'extension aux matrices partitionnées se fait sans difficulté :

$$C = AB = [C_{ij}] = [\sum_{k=1}^{\nu} A_{ik} B_{kj}]$$

où ν est le nombre de colonnes de A et de lignes de B.

Remarque Il existe d'autres produits matriciels.
1) terme à terme :

$$A_{np} * B_{pm} = [c_{ij}]_{1 \leq i \leq n, 1 \leq j \leq m} = [a_{ij} b_{ij}]_{1 \leq i \leq n, 1 \leq j \leq m} \in \mathcal{M}_{n,m}$$

2) produit de Kronecker (toujours défini)

$$AoB = \begin{pmatrix} a_{11}B & a_{12}B & \dots & a_{1p}B \\ \vdots & & & \vdots \\ a_{n1}B & a_{n2}B & \dots & a_{np}B \end{pmatrix}$$

si $A \in \mathcal{M}_{n,p}$ et $B \in \mathcal{M}_{r,m}$.
Il est souvent utilisé dans les équations de régression et généralise le produit d'une matrice par un scalaire.

Exemple Il est souvent utilisé pour transformer une équation linéaire matricielle, par exemple $AX = B$ avec $A \in \mathcal{M}_{m,n}$, $X \in \mathcal{M}_{n,p}$ et $B \in \mathcal{M}_{m,p}$, en un système linéaire.
En effet, si $X = [X_1..X_p]$ où les X_i sont les colonnes de X et si $B = [B_1..B_p]$ où les B_i sont les colonnes de B,
posons $x = \begin{pmatrix} X_1 \\ \vdots \\ X_2 \end{pmatrix}$ et $b = \begin{pmatrix} B_1 \\ \vdots \\ B_2 \end{pmatrix}$.
Le système est équivalent à $(IoA)x = b$.

3.1.3 Produit d'une matrice par un vecteur et applications

On définit différents produits.

1) Le produit usuel du type

$$Y = AX = \sum x_j A_j = \begin{pmatrix} \sum x_i a_{1i} \\ \vdots \\ \sum x_i a_{ni} \end{pmatrix}$$

CL des vecteurs colonnes de A.

2) La prémultiplication du type

$$XA = \sum x_j \tilde{A}_j = \begin{pmatrix} \sum x_i a_{i1} & \dots & \sum x_i a_{in} \end{pmatrix}$$

CL des vecteurs lignes de A. Ici le vecteur X, ligne, est placé devant.

3) Cas particulier où A est un vecteur ligne ou colonne. Soient x et y deux vecteurs colonnes.
On définit le *produit intérieur* par

$$x^T y = \begin{pmatrix} x_1 & \dots & x_n \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix} = \sum x_i y_i = \quad \text{nombre}$$

et le *produit extérieur* par

$$xy^T = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \begin{pmatrix} y_1 & \dots & y_n \end{pmatrix} = \begin{pmatrix} x_1y_1 & x_1y_2 & \dots & x_1y_n \\ \vdots & \vdots & & \vdots \\ x_ny_1 & x_ny_2 & \dots & x_ny_n \end{pmatrix} = \text{ matrice}$$

Application aux codes correcteurs

On a vu que pour calculer le mot de code associé à un bloc, il suffit de faire la somme modulo 2 des lignes de G qui correspondent aux bits égaux à 1 dans le bloc. Interprétation matricielle de ce calcul : dans la base canonique la matrice colonne qui représente un bloc b s'obtient en écrivant verticalement les bits de b. Ce qui revient à considérer b comme une matrice ligne pui à prendre sa transposée.
La matrice de l'application linéaire f est G^T. Donc si $f(b)$ est le mot de code associé à b alors $G^T b^T = f(b)^T$ ie

$$bG = f(b)$$

Exemple 1 (101) a pour image (101) $\begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix} = (1010)$.

Le sous-espace vectoriel orthogonal $\mathcal{C}^\perp = \{X \in E / XY^T = 0 \forall Y \in \mathcal{C}\}$ de ce code dans le dual de $\mathbb{Z}_2^n$ est un sous-espace de dimension $n-k$. Soit $(h_1,..,h_{n-k})$ une base de cet espace et soit H la matrice $(n-k) \times n$ des coordonnées dans la base canonique.
Si $c = (c_1..c_n) \in \mathbb{Z}_2^n$, alors

$$c \in \mathcal{C} \Leftrightarrow \forall i = 1..n-k,\ \ ch_i = 0 \Leftrightarrow Hc^T = 0$$

La matrice H est appelée *matrice de contrôle du code* $\mathcal{C}$. Et ces relations les *relations de code.*

Retour à l'exemple 1 $G = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{pmatrix} = (1010)$. On peut prendre par exemple $H = (1111)$.

Théorème 3.1.2 *Si* $G = \begin{pmatrix} I_k & | & A_{k,r} \end{pmatrix}$ *alors* $H = \begin{pmatrix} -A_{k,r}^T & | & I_r \end{pmatrix}$ *convient.*

Remarque Dans le cas des codes binaires, il suffit de prendre $H = \left(\begin{array}{c|c} A_{k,r}^T & I_r \end{array} \right)$

Retour à l'exemple 2 $G = \left(\begin{array}{ccc} 1 & 1 & 1 \end{array} \right)$ et on prend $H = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$

Exemple 3 Si $G = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{pmatrix}$, on prend $H = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \end{pmatrix}$

Soit c^* le mot reçu. $Hc^{*T} = \begin{cases} 0 & \text{si } c^* \in \mathcal{C} \\ \neq 0 & \text{sinon} \end{cases}$

Dans le cas où $Hc^{*T} \neq 0$ ce vecteur est appelé le *syndrome* s. On peut écrire le message reçu comme la somme du mot de code et d'un vecteur d'erreur : $c^* = c + e$. Alors

$$s = Hc^{*T} = Hc^T + He^T = He^T$$

Le syndrome caractérise une maladie (erreur) mais ne dépend pas du malade (du mot émis).

Il faut donc résoudre le système $Hc^{*T} = He^T$ pour estimer l'erreur. Le problème est que le système est indéterminé : il y a trop d'inconnues.

Si l'on suppose qu'il n'y a qu'un seule erreur, alors $e = (000..010..0000)$ où le 1 indique la place du bit erroné. Donc le syndrome recopie la colonne de H correspondant à l'erreur.
Cas général : le syndrome sera une CL des colonnes de H correspondant aux erreurs.

Retour à l'exemple 3 Si on reçoit $c^* = (11111)$,

$$s = Hc^{*T} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

$s \neq 0$ donc on a au moins une erreur et s est CL des vecteurs colonnes de H correspondant aux erreurs.
• ou bien il y a une erreur en 3ème position
• ou bien il y a deux erreurs qui sont sur les 1ère et 2ème positions ou 1ère et dernière positions ou 2ème et 4ème positions ou 4ème et dernière positions
• ...
Si on suppose que l'on fait au plus une erreur alors on voit que la 3ème colonne est recopiée donc l'erreur est en 3ème position dans c^*. Donc $c = (11011)$.

Plus généralement, le syndrome s a r composantes, il rend compte de 2^r situations.
Pour corriger une erreur,
• ou bien 1ère position
• ou bien 2ème position
• ...
• ou bien nème position
• ou bien pas d'erreurs
On a donc $n+1$ situations.

CN pour corriger une erreur :

$$2^r \geq n+1 \Leftrightarrow n \leq 2^r - 1$$

C'est une condition incluse dans celle déjà vue.

Définition 3.1.4 *Quand il y a égalité, on a des codes optimaux : codes de Hamming.*

Retour à l'exemple 1 $n = 4$, $r = 1$ donc $n + 1 = 5$ et $2^r = 2$.

Retour à l'exemple 2 $n = 3$, $r = 2$ donc $n + 1 = 4 = 2^r = 2$. C'est un code optimal.

La répercution sur la matrice H est la suivante : les colonnes de H doivent être non nulles et différentes.

Proposition 3.1.2 *Si les colonnes de H sont toutes différentes et non nulles alors on sait corriger et détecter une erreur au moins et si Y est le mot codé, Y' le mot reçu et e l'erreur, $s(Y') = HY'^T = HY^T + He^T = s(e) = H_i$ et alors $Y = Y' - e_i$.*

Question : jusqu'où peut-on aller dans la vérification d'un code? combien d'erreurs va-t-on pouvoir corriger?

Théorème 3.1.3 *Capacité de correction d'un code.*
Si dans la matrice H d'un code, tout système de $2e$ vecteurs colonnes est indépendant, alors le code est capable de corriger e erreurs au moins.

3.2 L'inversion

3.2.1 Dfinition et propriétés

La notion de produit permet de définir l'inverse d'une matrice et comme le produit matriciel n'est pas commutatif, il y a lieu de distinguer l'inverse à droite et l'inverse à gauche.
Soit $A \in \mathcal{M}_{m,n}$. On définit :

- l'*inverse à droite* comme la matrice $A^{-d} \in \mathcal{M}_{n,m}$ telle que $AA^{-d} = I_m$
- l'*inverse à gauche* comme la matrice $A^{-g} \in \mathcal{M}_{n,m}$ telle que $A^{-g}A = I_n$

Si $m \neq n$, les inverses à doite et à gauche ne peuvent simultanément exister et dans le cas où $m = n$, si A^{-d} et A^{-g} existent alors elles sont égales. Dans ce cas, on note $A^{-d} = A^{-g} = A^{-1}$.
On verra qu'en fait l'existence de l'une implique celle de l'autre.

Définition 3.2.1 *Une matrice carrée est* régulière *si elle est inversible ie si elle admet une inverse. Sinon, on dit qu'elle est* singulière.

Proposition 3.2.1 *Si elle existe, l'inverse est unique.*

Remarque On peut construire une inverse généralisée pour les matrices qui ne sont pas inversibles, qui présente des propriétés interessantes pour la résolution de systèmes linéaires.

Définition 3.2.2
- *Si* $A^2 = I$*, on dit que* A *est* involutive.
- *Si* $AA^T = A^TA = I$*, on dit que* A *est* orthogonale *et* $A^{-1} = A^T$.
- *Si* $AA^* = A^*A = I$*, on dit que* A *est* unitaire *et* $A^{-1} = A^*$.

Propriétés lorsque A^{-1} et B^{-1} existent, on a les propriétés suivantes :
• $(AB)^{-1} = B^{-1}A^{-1}$,
• $(A^{-1})^T = (A^T)^{-1}$, donc A^T inversible $\Leftrightarrow$ A inversible,
• $(A^{-1})^* = (A^*)^{-1}$.

Remarque les matrices carrées inversibles forment un groupe multiplicatif appelé le *groupe linéaire*

3.2.2 Calcul pratique de l'inverse

Lorsqu'une matrice carrée est régulière, comment calculer son inverse?

Méthode 1 : inverser le système linéaire $AX = Y$ en $X = A^{-1}Y$.

Exemple Si $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$, alors $A^{-1} = \frac{1}{ad-bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$

Méthode 2 : méthode de Gauss.

Méthode 3 : formule donnée par le déterminant et la transposée de la comatrice :

$$A^{-1} = \frac{1}{det(A)}\text{com}(A)^T = \frac{1}{det(A)}(\text{matrice des cofacteurs de } A)^T$$

C'est une formule complexe, difficile à mettre en pratique en général.

Méthode 4 : par récurrence. On ramène la calcul de l'inverse d'une matrice $n \times n$ à celui d'une matrice $n-1 \times n-1$, ceci grâce à l'inverse des matrices partitionnées.
Si $A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}$ est régulière avec A_{11} et A_{22} sont des blocs carrés, partitionnons l'inverse de A : $A^{-1} = \begin{pmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{pmatrix}$ où les X_{ij} ont des tailles identiques à celles des A_{ij}.
Les produits $AA^{-1} = A^{-1}A = I$ conduisent à un système que l'on résoud ensuite.

Exemple 1 Soit $M_{n+1,n+1} = \begin{pmatrix} A_{n,n} & b_n \\ c_n^T & a_n \end{pmatrix}$ avec a_n scalaire et b_n et c_n vecteurs colonnes.
Si $a_n \neq 0$, $M_{n+1,n+1} = \begin{pmatrix} X & -X\frac{b_n}{a_n} \\ -\frac{c_n^T}{a_n}X & \frac{1}{a_n^2}(a_n + c_n^T X b_n) \end{pmatrix}$
avec $X = (A - \frac{b_n c_n^T}{a_n})^{-1}$.

Exemple 2 Si $T_{n,n} = \begin{pmatrix} T_{n-1,n-1} & 0 \\ c_n^T & t_n \end{pmatrix}$, alors $T_{n,n}^{-1} = \begin{pmatrix} T_{n-1,n-1}^{-1} & 0 \\ \frac{-c_n^T}{t_n}T_{n-1,n-1}^{-1} & \frac{1}{t_n} \end{pmatrix}$

Méthode 5 : Lemme d'inversion matricielle.
Soi M une matrice régulière telle que $M = A + BCD$ où A et C sont

régulières.
Alors $M^{-1} = (A + BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} + DA^{-1}B)^{-1}DA^{-1}$.

Remarque Si B et D sont respectivement des matrices ligne et colonne et C un scalaire non nul, $M^{-1} = (A+BCD)^{-1} = A^{-1} - \frac{C}{1+CDA^{-1}B}A^{-1}BDA^{-1}$.

Exemple $M = \begin{pmatrix} 2 & -1 & 1 \\ 2 & -1 & 2 \\ 3 & -3 & 2 \end{pmatrix}$
Posons $A = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & O \\ 0 & 0 & -1 \end{pmatrix}$, $B = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$, $C = 1$ et $D = \begin{pmatrix} 1 & -1 & -1 \end{pmatrix}$.
Alors $M^{-1} = \frac{1}{3}\begin{pmatrix} 4 & -1 & -1 \\ 2 & 1 & -2 \\ -3 & 3 & 0 \end{pmatrix}$

3.3 Introduction à la cryptographie

Soit un message M à retransmettre entre deux personnes. L'objet de la cryptographie est de fournir des moyens afin de rendre illisible le message pendant son transport : c'est en effet pendant le transport qu'un message est le plus facilement intercepté : à des fins militaires, industrielles, financières... C'est un problème crucial aujourd'hui à cause de la mise en place de réseaux informatiques à grande échelle, comme l'Internet.
La cryptographie date de plusieurs siècles puisqu'elle était déjà utilisée chez les Romains.

Emetteur	$\longrightarrow^{M}$	Récepteur
$\downarrow$		$\downarrow$
Codage $\mathcal{C}$ fct C	$\longrightarrow^{\mathcal{C}(M)}$	Décodage fct $D = C^{-1}$

But : fournir un algorithme pour rendre illisible le message pendant son transport et un algorithme de décryptage.
$\Rightarrow$ course poursuite entre les fournisseurs d'algorithmes et les crypto-analystes.

3.3.1 Méthode de cryptage mono-alphabétique

On se place dans $\mathbb{Z}/N\mathbb{Z}$ où N=nombre de symboles différents utilisés (ex : $N = 26$ si on prend l'alphabet comme support).

Fonction la plus simple : la fonction affine

$$\mathcal{C} : x \to ax + b \quad [N]$$

C'est une bijection ssi $a \wedge N + 1$. Ici on code lettre par lettre.

Exemple OUI correspond à 15-21-9 dans $\mathbb{Z}/26\mathbb{Z}$ si on associe 1 à A, 2 à B... Si $a = 3$ et $b = 12$, ce dernier est codé en 5-22-13, ce qui donne EVM. La fonction de décryptage est alors $\mathcal{C}^{-1} = 9X + 22 \quad [26]$.

Le problème est que chaque lettre est codée de la même façon. Donc la plus fréquente après décodage est la plus fréquente avant décodage. Donc on peut déterminer l'algorithme.

3.3.2 Méthode de cryptage par blocs

On regroupe les lettres de M par paquets de k caractères. Chaque paquet correspond à un k-uplet de nombre de $[0,N-1]$: $(x_1,..,x_k)$. On considère ce paquet comme un vecteur de $(\mathbb{Z}/N\mathbb{Z})^k$. On construit alors la fonction de codage à l'aide d'une matrice carrée A de dimension k à coefficients dans $\mathbb{Z}/N\mathbb{Z}$ et le codage du k-ulpet sera tout simplement AX.

Exemple soit $k = 2$ et $A = \begin{pmatrix} 2 & 5 \\ 21 & 6 \end{pmatrix}$

On veut coder 'ANNA' $\leftrightarrow$ 'AN' et 'NA'. Le premier correspond à $X_1 = (1\ 14)^T$, il sera codé en $AX_1 = (20\ 1)^T$ et le second à $X_2 = (14\ 1)^T$ et il sera codé en $AX_1 = (7\ 14)^T$.

Finalement, 'ANNA' est codé en 'TAGN'.

On voit bien que chaque lettre est codée différemment. Le problème est qu'une lettre en i-ème position sera toujours codée de la même façon. Un autre problème concerne le fonction de décodage. Existe-t-elle ie la fonction de codage est-elle une bijection?

Proposition 3.3.1 *Une matrice carrée A à coefficients dans $\mathbb{Z}/N\mathbb{Z}$ est inversible ssi $det(A)$ est inversible dans $\mathbb{Z}/N\mathbb{Z}$.*

3.3.3 Méthode RSA

Elle est meilleure que les autres et est considérée comme une des plus fiables. De plus, elle ne nécessite pas de tenir secret la fonction de cryptage. Voilà pourquoi on parle de clé (=fonction de cryptage) publique. Seules les informations concernant le décryptage doivent être secrètes. Seul le récepteur possède un secret : le fonction de décryptage.

Dans un groupe de personnes devant communiquer, on distribue un annuaire de cryptage à utiliser pour expédier un message à une personne donnée.

Emetteurs		Récepteurs
	$\longrightarrow_{\mathcal{C}_1}$	$R_1\ \mathcal{C}_1^{-1}$
	$\longrightarrow_{\mathcal{C}_2}$	$R_2\ \mathcal{C}_2^{-1}$
	$\longrightarrow_{\mathcal{C}_3}$	$R_3\ \mathcal{C}_3^{-1}$

.....

Seul le récepteur connait $\mathcal{C}^{-1}$!

Chapitre 4

Notions sur les corps finis et les polynômes irréductibles

4.1 Définitions

4.1.1 Les corps finis

Définition 4.1.1 *Un*anneau $(A, +, *)$ *est un ensemble muni de 2 lois de composition internes* $+$ *et* $*$ *tel que*

($A1$) $(A,+)$ *est un groupe abélien (i.e. commutatif)*

($A2$) *la loi* $*$ *est associative :*

($A3$) *la loi* $*$ *est distributive à gauche et à droite par rapport à la loi* $+$ *:*

$\forall (x,y,z) \in A^3, x*(y+z) = x*y + x*z,$

$\forall (x,y,z) \in A^3, (x+y)*z = x*z + y*z,$

($A4$) *la loi* $*$ *a un élément neutre :* $1 = 1_A$

Définition 4.1.2 *Un* corps *est un anneau commutatif dans lequel tout élément non nul est inversible ie* $\forall x \in \mathbb{K},\ \exists y \in \mathbb{K},\ xy = 1.$

Proposition 4.1.1 *On montre que le cardinal d'un corps fini est une puissance d'un nombre premier* p*.*

Donc on a deux types de corps finis :

- ceux de cardinal un nombre premier p
- ceux de cardinal une puissance d'un nombre premie p : $q = p^k$

4.1.2 Les polynômes irréductibles

Définition 4.1.3 *Soit* A *un anneau. Un polynôme* P *de* $A[X]$ *est* irréductible *dans* $A[X]$ *ssi*

• *son degré est supérieur ou égal à 1,*
• *ses seuls diviseurs dans $A[X]$ sont les polynômes uP où u appartient à $U(A)$, les éléments inversibles de A.*

Remarque Cela signifie que l'on ne peut pas écrire P sous la forme $P(X) = Q(X)R(X)$ avec $deg(Q) < deg(P)$ et $deg(R) < deg(P)$ ie on ne peut pas *scinder* P.

Proposition 4.1.2 *Irréductibilité des polynômes*
1) Tout polynôme de degré 1 est irréductible.
2) Tout polynôme irréductible de degré ¿1 n'a pas de racine dans $\mathbb{K}$.
3) La réciproque de 2) est fausse : $Q = (X^2+1)^2$ n'a pas de racine dans $\mathbb{Q}$ mais Q est réductible dans $\mathbb{Q}[X]$.
4) Toutefois, elle est vraie pour les polynômes de degrés 2 et 3 : les polynômes irréductibles dans $\mathbb{K}[X]$ de degrés 2 et 3 sont exactement ceux qui n'ont pas de racine dans $\mathbb{K}$.

Théorème 4.1.1 *Critère d'Esenstein*
Soit $P = a_nX^n + .. + a_0$ un polynôme sur $\mathbb{K}$. Soit p un élément irréductible de $\mathbb{K}$ tel que
1) p ne divise pas a_n
2) p divise tous les a_i pour $i = 1..n-1$
3) p^2 ne divise pas a_0
Alors P est irréductible dans $\mathbb{K}[X]$.

4.2 Structure des corps finis

4.2.1 Corps de Galois de base, cardinal=p

Ils sont faciles à construire.

Théorème 4.2.1 *Pour tout nombre premier p, il existe à un andomorphisme près un seul corps de cardinal p. C'est le corps des restes modulo p.*

Plus précisément, c'est $(\mathbb{Z}/p\mathbb{Z}, +, *)$ le corps muni de l'addition et de la multiplication modulo p. Alors $\mathbb{Z}/p\mathbb{Z}$ s'écrit $\mathbb{Z}/p\mathbb{Z} = \{0,1,2,..,p-1\}$. Ici "$p = 0$".

Exemple 1) $\mathbb{Z}/3\mathbb{Z} = \{entiers\ relatifs\ modulo\ 3\}$ On a trois classes de représentants : 0,1 ou 2.
Construction des tables :

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

*	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

2) $\mathbb{Z}/7\mathbb{Z} = \{entiers\ relatifs\ modulo\ 7\}$. Etude des puissances successives de chaque élément non nul.

On peut représenter $\mathbb{Z}/7\mathbb{Z}$ par $\mathbb{Z}/7\mathbb{Z} = \{0,1,2,3,4,5,6\}$.

Puissances de 1 :
$1^0 = 1$, $1^1 = 1$, $1^2 = 1$, $1^3 = 1$, $1^4 = 1$, $1^5 = 1$, $1^6 = 1$

Puissances de 2 :
$2^0 = 1$, $2^1 = 2$, $2^2 = 4$, $2^3 = 1$

Puissances de 3 :
$3^0 = 1$, $3^1 = 3$, $3^2 = 2$, $3^3 = 6$, $3^4 = 4$, $3^5 = 5$, $3^6 = 1$

Puissances de 4 :
$4^0 = 1$, $4^1 = 4$, $4^2 = 1$

Puissances de 5 :
$5^0 = 1$, $5^1 = 5$, $5^2 = 3$, $5^3 = 6$, $5^4 = 2$, $5^5 = 4$, $5^6 = 1$

Puissances de 6 :
$6^0 = 1$, $6^1 = 6$, $6^2 = 1$

donc $\mathbb{Z}/7\mathbb{Z} = \{0,3^0,3^1,3^2,3^3,3^4,3^5\}$. On dit que 3 est un *élément primitif* ou encore *générateur* de $\mathbb{Z}/7\mathbb{Z}$. De même, 5 est un générateur.

Proposition 4.2.1 *On montre que dans un corps fini, il existe toujours un élément primitif.*

4.2.2 Corps de Galois de card p^k avec $k > 1$

Théorème 4.2.2 *Pout tout nombre entier premier p et tout entier $k > 1$, il existe, à un isomorphisme près, un unique corps de cardinal $q = p^k$.*

Notation $GF(q) = GF(p^k) = F_q = F_p^k$ GF signifie Galois Field : corps de Galois.

Plus précisément c'est le corps $\mathbb{Z}/p\mathbb{Z}[X]$ quotienté par l'idéal engendré par un polynôme irréductible sur $\mathbb{Z}/p\mathbb{Z}$ de degré k.

Remarque Ici on travaille avec les polynômes de $\mathbb{Z}/p\mathbb{Z}[X]$ modulo un polynôme irréductible sur $\mathbb{Z}/p\mathbb{Z}$ de degré k, comme on travaillait précédemment avec les entiers modulo p.

Principe de construction : $GF(p^k)$ est une extension de $GF(p) = \mathbb{Z}/p\mathbb{Z}$ de même que $\mathbb{C}$ est une extension de $\mathbb{R}$.

Principe de l'extension de $\mathbb{R}$ à $\mathbb{C}$: Soit $P(X) = X^2 + 1 \in \mathbb{R}[X]$, c'est un polynôme irréductible dans $\mathbb{R}[X]$.
On pose $\mathbb{C} = \mathbb{R}[X]/(X^2 + 1)$ ensemble des classes des polynômes à coefficients dans $\mathbb{R}$ modulo $X^2 + 1$. Alors

$$\mathbb{C} = \{a + ib \text{ avec } a,b \in \mathbb{R}\}$$

avec i classe de X modulo $X^2 + 1$ et i vérifie $i^2 = -1$.

Exemple Construction de $GF(8)$

Ici $p = 2$ et $k = 3$, $GF(8) = GF(2^3)$ et le corps de base est $GF(2) = \mathbb{Z}/2\mathbb{Z}$. On cherche un polynôme irréductible de degré 3.

polynômes irréductibles de degré 3 sur $\mathbb{Z}/2\mathbb{Z}$	racines
$x^3 + x^2 + x + 1$	1
$x^3 + x^2 + x$	0
$x^3 + x + 1$	-
$x^3 + x^2 + 1$	-
$x^3 + x^2$	0,1
$x^3 + x$	0,1
$x^3 + 1$	1
x^3	0

Les seuls candidats possibles sont x^3+x+1 et x^3+x^2+1 car ils n'ont pas de racine. De plus, ils sont de degré 3 donc c'est une condition suffisante et ces deux polynômes sont bien irréductibles.
On choisit x^3+x+1 comme modulo. Le rest de la division euclidienne d'un polynôme par x^3+x+1 est de degré ¡3. Donc les seuls restes possibles sont les polynômes de degré ≤ 2 sur $\mathbb{Z}/2\mathbb{Z}$. Ils sont de la forme ax^2+bx+c :

restes	codage	restes	codage
0	000	x^2	100
1	001	x^2+1	101
x	010	x^2+x	110
$x+1$	011	x^2+x+1	111

Donc de la même façon que pour $\mathbb{Z}/p\mathbb{Z}$ on peut expliciter les éléments de $\mathbb{Z}/2\mathbb{Z}[x]/(x^3+x+1)$:

$$\mathbb{Z}/2\mathbb{Z}/(X^3+X+1) = \{0,1,x,x+1,x^2,x^2+1,x^2+x,x^2+x+1\}$$

Soit α une racine de x^3+x+1. α est la classe de x modulo x^3+x+1. Alors $\alpha^3+\alpha+1=0$ ie $\alpha^3=-\alpha-1=\alpha+1$.

Etdude des puissances successives de α :

Puissances de α	Elément	Log de base α	Codage
α^0	1	0	001
α^1	α	1	010
α^2	α^2	2	100
α^3	α+1	3	011
α^4	α^2+α	4	110
α^5	α^2+α+1	5	111
α^6	α^2+1	6	101
α^7	1	7	001

Donc $\forall k$, $\alpha^k=\alpha^l$ avec $l=0,1,..6$. On peut écrire

$$GF(8) = \{0,\alpha^0,\alpha^1,\alpha^2,\alpha^3,\alpha^4,\alpha^5,\alpha^6\}$$

α est un élément primitif.

Polynôme minimal

Définition 4.2.1 *Le* polynôme minimal *d'un élément β est le polynôme irréductible non nul de plus faible degré dont β est racine.*

Exemple x est le polynôme minimal de 0, $x+1$ celui de 1, x^3+x+1 celui de α.

Théorème 4.2.3 *Si β est racine d'un polynôme dans $\mathbb{Z}/p\mathbb{Z}$ alors β^p aussi.*

Cas particuliers : en binaire si P est un polynôme annulateur de β, alors β^2, β^4... sont aussi racines de P.

Définition 4.2.2 *Les racines d'un polynôme minimal sont les* éléments conjugués.

Exemple Recherche du polynôme minimal de α^3
Si α^3 est racine alors α^6, $\alpha^{12}=\alpha^5$, $\alpha^{24}=\alpha^{10}=\alpha^3$ aussi. Donc le polynôme minimal de α^3 est $(x+\alpha^3)(x+\alpha^5)(x+\alpha^6)=x^3+x^2+1$.

Donc nous avons vu que tout corps fini peut se représenter comme l'ensemble des puissances successives de son élément primitif (non nécessairement unique) : ie si $q=p^k$ et α est un élément primitif

$$\mathbb{K}=\{\alpha^0,\alpha^1,\alpha^2,..,\alpha^{q-2}\}\cup\{0\}\approx(\mathbb{Z}/p\mathbb{Z}[X])/(P)$$

où P est un polynôme irréductible de degré k.

On remarque que $\forall x\in\mathbb{K}^\star$, $\exists!\ u\in[0,q-1[$, $x=\alpha^u$. On dit que u est le *logarithme discret* de x en base α. La recherche de cet exposant constitue le problème du logarithme discret.

Comment se déroulent les calculs?

• La multiplication de deux éléments de $\mathbb{K}$ est extrêmement simple, elle se ramène à un addition des exposants dans $\mathbb{Z}/(q-1)\mathbb{Z}$
• L'addition par contre pose problème. Une idée simple est de passer par la représentation polynomiale. Pour chaque α^d, le polynôme Q de $\mathbb{Z}/p\mathbb{Z}[X]$ de degré $<k$ tel que $\alpha^d=Q(\alpha)$ s'obtient facilement en prenant le reste de la division euclidienne de X^d par P.

Par contre la manipulation inverse est difficile dans le cas général. Pour pallier la difficulté, on peut avoir recours lorsque le cardinal de $\mathbb{K}$ le permet à une table qui à chaque α^d associe sa représentation polynomiale.

Exemple On veut calculer $a = \alpha^3 + \alpha^4$ dans l'exemple précédent. Or en utilisant la table, $\alpha^3 = \alpha + 1$ et $\alpha^4 = \alpha^2 + \alpha$ donc $a = \alpha + 1 + \alpha^2 + \alpha = \alpha^2 + 1$ et en utilisant la table dans l'autre sens on obtient $a = \alpha^6$.

D'où l'intérêt du calcul du logarithme discret.

Une application : Echange de clés publiques et logarithme discret

Les méthodes à clés publiques sont considérées comme inviolables : l'usage de paramètres de codage (clés) de grande dimension interdit à un espion un décodage illicite. Cependant, les temps de calculs pour le codage et le décodage sont relativement longs. Donc on contourne le problème en changeant régulièrement les clés (on a donc des nombres plus petits que l'on change plus fréquemment).
Une clé est un nombre que l'on veut garder secret entre deux correspondants. Comment deux personnes peuvent elles choisir un nombre commun secret de manière sûre?

Soient X et Y deux partenaires. Ils conviennent publiquement d'un corps fini $\mathbb{K}$ et d'un élément primitif α. Puis chacun d'eux choisit un nombre, respectivement a_X et a_Y, qu'ils ne communiquent à personne. Ils calculent ensuite chacun de leur côté α^{a_X} et α^{a_Y} qu'ils échangent publiquement.

Remarque Notons que c'est ici qu'intervient pour la sécurité du procédé le fait que le problème du logarithme discret est trs difficile car on suppose qu'un espion en prenant connaissance de ce dernier échange est incapable de retrouver a_X et a_Y..

Nos deux partenaires n'ont plus qu'une chose à faire :

- X élève α^{a_Y} à la puissnace a_X(nombre qu'il est le seul à connaitre)
- de même Y élève α^{a_X} à la puissnace a_Y

Ils possèdent tous les deux en secret le nombre $\alpha^{a_X a_Y}$, la clé dontils ont besoin pour échanger plus rapidement par la suite. Ils choisissent un nombre r premier avec $n = \alpha^{a_X a_Y}$ et prennent pour fonction de codage $x \to x^r$. Tout le monde peut coder par contre le décodage est donné par $x \to x^u$ avec

$ru + nv = 1$ et reste secret.

Remarque D'autres algorithmes de cryptographie reposent sur ce problème du logarithme discret. On l'utilise par exemple pour réaliser des signatures de documents informatiques.

Chapitre 5

Les applications linéaires et les matrices associées

5.1 Les applications linéaires et les matrices associées

Définition 5.1.1 *Soient E et F deux espaces vectoriels sur $\mathbb{K}$ de dimension n et p. Soit $f : E \to F$. On dit que f est* linéaire *si*

$$\forall \lambda_1,..,\lambda_m \in \mathbb{K},\ \ \forall x_1,..,x_m \in E,\ \ f(\sum_{i=1}^{m} \lambda_i x_i) = \sum_{i=1}^{m} \lambda_i f(x_i)$$

Notation On note $\mathcal{L}(E,E')$ l'ensemble des applications linéaires. C'est un espace vectoriel.

Remarque f est donc complètement déterminée lorsqu'on connait $f(\mathcal{B}) = (f(e_1),..,f(e_n))$ si $\mathcal{B} = (e_1,..e_n)$ est une base de E.

Soient $e_1,..,e_n$ et $f_1,..,f_p$ des bases respectivement de E et F.
Alors $\exists A = (a_{ij}) \in \mathcal{M}_{p,n}$ telle que

$$\begin{cases} f(e_1) = a_{11}f_1 + a_{21}f_2 + ... + a_{p1}f_p \\ .... \\ f(e_n) = a_{1n}f_1 + a_{2n}f_2 + ... + a_{pn}f_p \end{cases}$$

ie $f(e_i) = \sum_{j=1}^{p} a_{ji} f_j$.

$$\text{On note que } A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ \vdots & \vdots & & \vdots \\ a_{p1} & a_{p2} & \dots & a_{pn} \end{pmatrix}$$
$$\begin{matrix} \uparrow & & \uparrow \\ f(e_1) & \dots & f(e_n) \end{matrix}$$

Proposition 5.1.1 *$Y = f(X)$ s'obtient alors en calculant le produit $AX = Y$ où A est la matrice de f dans les bases $(e_1,...,e_n)$ et $(f_1,...,f_n)$.*

5.2 Image et noyau

Définition 5.2.1 *Soit $f : E \to E'$ une application linéaire. Soient F et F' des sous-espaces vectoriels de E et E' respectivement.*
Alors

- $f(F) = \{f(x) \ / \ x \in F\}$ *est l'*image directe *de F par f.*
- $f^{-1}(F') = \{x \in E \ / \ f(x) \in F'\}$ *est l'*image réciproque *de F' par f.*

Ce sont des sous-espaces vectoriels.

Cas particuliers :
1) $f(E)$ est un sev de E' appelé *image* de f et noté $Im(f)$. Sa dimension est appelée *rang* de f et est notée $rg(f)$.
2) $f^{-1}(0)$ est un sev de E appelé *noyau* de f et noté $Ker(f)$.

Proposition 5.2.1 *Soit $f \in \mathcal{L}(E,E')$. Alors*

- *f est injective $\Leftrightarrow Ker(f) = \{0\}$*
- *f est surjective $\Leftrightarrow Im(f) = E'$*

Exemple Soit $D : \mathbb{R}[x] \in \mathbb{R}[x]$ qui à P associe P'.
Alors $Ker(D) = \{$polynômes constants$\}$ et $Im(D) = \mathbb{R}[x]$.

Proposition 5.2.2 *Soi $g : G \to E$ et $f : E \to F$.*
1) $Ker(fog) \supset Ker(g)$
2) $Im(fog) \subset Im(f)$
3) fog injective $\Rightarrow g$ injective
4) fog surjective $\Rightarrow f$ surjective

Théorème 5.2.1 *Le théorème du rang.*
Soient E et E' deux ev de dimensions finies et soit $f : E \to E'$ une application linéaire. Alors on a

$$dim(E) = rg(f) + dim(Ker(f))$$

Corollaire 5.2.1 *Soit $f \in \mathcal{L}(E,E')$ avec E et E' deux ev de même dimension finie.*
Alors f bijective $\Leftrightarrow$ f injective $\Leftrightarrow$ f surjective

Contrexemple D est surjective mais pas injective. On est en dimension infinie!

5.3 Rang d'une application linéaire-Rang d'une matrice

Définition 5.3.1 *1) Soit $\{v_1,..,v_p\}$ une famille de vecteurs. On appelle* rang de la famille *la dimension de l'espace engendré par les vecteurs v_i.*
2) Soit $A \in \mathcal{M}_{p,n}(\mathbb{K})$, $A = [C_1..C_n]$ si les C_i sont les vecteurs colonnes de A. On appelle rang de la matrice *A le rang de la famille des vecteurs colonnes de A :*

$$rg(A) = rg[C_1...C_n] = dimVect\{C_1,...,C_n\}$$

Proposition 5.3.1 *Soit $f \in \mathcal{L}(E,F)$ avec E et F deux ev de dimensions finies. Soient $(e_1,..,e_n)$ et $(f_1,..,f_p)$ deux bases de E et F respectivement.*
Soit $A = Mat(f)_{(e_i,f_j)}$. On a alors $rg(A) = rg(f)$.
En particulier, deux matrices qui représentent la même application linéaire dans des bases différentes ont même rang.

Propriétés $\forall A$, $rg(A) = rg(A^T)$
Le rang d'une matrice est aussi égal au rang des vecteurs lignes. Donc $0 \leq rg(A) \leq \min(n,p)$.

Exemple $A = \begin{pmatrix} 1 & -1 & 3 & 5 & 1 \\ 2 & 0 & -1 & 3 & 1 \\ 3 & -1 & 2 & 8 & 2 \end{pmatrix}$
On remarque que $L_3 = L_1 + L_2$ donc $rg(A) \leq 2$ et en fait $rg(A) = 2$ car L_1 et L_2 sont indépendantes.

Définition 5.3.2 *Si $r = \min(n,p)$, on dit que A est de* rang plein.

Propriétés

- $rg(A) + rg(B) - n \leq rg(AB) \leq \min(rg(A), rg(B))$
- $rg(A + B) \leq rg(A) + rg(B)$
- A régulière $\Rightarrow rg(AB) = rg(B)$ et $rg(BA) = rg(B)$
- $rg(AA^T) = rg(A^TA) = rg(A)$
- En général, $rg(AB) \neq rg(BA)$

Théorème 5.3.1 *Si A est une matrice carrée ayant une inverse à gauche, alors elle a une inverse à droite.*

Caractérisation des matrices de rang plein : Soit $G \in \mathcal{M}_{m,n}(\mathbb{K})$ avec $m \leq n$.

$$\begin{array}{lll} G \text{ de rang plein} & \Leftrightarrow & rg(G) = m \\ & \Leftrightarrow & \exists m \text{ colonnes de } G \text{ formant un système libre} \\ & \Leftrightarrow & \exists m \text{ lignes de } G \text{ formant un système libre} \\ & \Leftrightarrow & \text{après permutation eventuelle des colonnes} \\ & & G \sim (R \;\; A) \text{ où } R \text{ est une matrice } m \times m \text{ régulière} \end{array}$$

Cas des matrices carrées :

$$\begin{array}{lll} A \text{ de rang plein} & \Leftrightarrow & A \text{ est régulière} \\ & \Leftrightarrow & A \text{ est régulière} \end{array}$$

5.4 Cas particulier : Les formes linéaires

Définition 5.4.1 *f est une* forme linéaire *si f est une application linéaire de E dans $\mathbb{K}$ considéré comme un espace vectoriel de dimension 1 sur $\mathbb{K}$.*

Conséquence $f(E)$ est de dimension 1 ou 0 et donc $Ker(f)$ est de dimension $n - 1$ ou n.

Description Si $c_i = f(e_i)$ où $(e_1,..,e_n)$ est une base de E et si $x = \sum_{i=1}^{n} x_i e_i$, alors

$$f(x) = \sum_{i=1}^{n} c_i f(e_i) = \sum_{i=1}^{n} x_i c_i = x^T c = c^T x = < x, c >$$

Proposition 5.4.1 *Quand les c_i ne sont pas tous nuls, $Ker(f)$ est un espace de dimension $n-1$. Par définition, un sev de dimension $n-1$ est un* hyperplan*, c'est le plus gros sev strictement contenu dans E.*

Remarque Quand on a r formes linéaires indépendantes f_1,..,f_r alors on montre par récurrence que $\cap_{i=1}^{n} Ker(f_i)$ est de dimension $n-r$.

5.5 Application aux sous-espaces vectoriels duaux

On définit le *produit scalaire* entre deux vecteurs x et y par

$$< x,y >= \sum_{i=1}^{n} x_i y_i x^T y = y^T x$$

si $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ et $y = \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$.

Soit V un sev de E. On désigne par $V^\perp$ = *orthogonal* de V l'ensemble des vecteurs y orthogonaux à tout vecteur de V pour le produit scalaire ie

$$V^\perp = \{y \in E \ / \ \forall x \in V, \quad < x,y >= 0\}$$

Exemple cf définition de la matrice de contrôle (ses lignes forment une base de l'orthogonal du code.

Théorème 5.5.1 *Dimension de l'orthogonal*
Si E est de dimension n et V est de dimension r alors $V^\perp$ est de dimension $n-r$ dans E.

Remarque Dans le cas $\mathbb{K} = \mathbb{R}$ ou $\mathbb{C}$, $< x,x >= 0 \Rightarrow x = 0$ et $V^\perp$=sev supplémentaire de V.
En général c'est faux, on peut très bien avoir $V \cap V^\perp \neq \{0\}$.

Chapitre 6

Notions sur la théorie des graphes

6.1 Introduction

6.1.1 Deux exemples

Nous allons présenter deux exemples concrets pour lesquels la résolution peut se faire à l'aide de la théorie des graphes.

Exemple 1 Le problème du cheminement.

Il s'agit de choisir un itinéraire de longueur minimale pour aller de la ville A à la ville B. On suppose donnée une carte routière à laquelle on associe un "graphe" $G = (X,U)$ où X l'ensemble des "sommets" est en bijection avec l'ensemble des carrefours de la carte. Deux sommets x et y sont joints par un "arc", segment orienté élément de U, si on peut se rendre du carrefour correspondant à x à celui correspondant à y par un tronçon direct de route. En général, le graphe d'une carte routière est symétrique ie $(xy) \in U \Rightarrow (yx) \in U$. Il n'en est pas de même pour une carte de ville (penser aux sens interdits).
A chaque arc, on associe sa "longueur" $d(u)$ et on appelle "réseau" $R = (X,U,d)$ le graphe muni de d.
Reformulation du pb de plus court chemin : Etant donné deux sommets A et B d'un réseau $R = (X,U,d)$, trouver une séquence d'arcs à la queue leu-leu dont l'extrémité initiale du premier arc est A et l'extrémité terminale du dernier arc est B de longueur minimale (la longueur d'un chemin est la

somme des longueurs des arcs).
Le graphe (X,U) étant fini, le nombre de chemins de A à B ne repassant pas par le même sommet est fini lui aussi.

Exemple 2 Le problème du voyageur de commerce.

Un voyageur de commerce ayant n villes à visiter souhaite établir une tournée qui lui permette de visiter une et une seule fois chaque ville pour finalement revenir à son point de départ, ceci en minimisant la distance parcourue.
On a donc un réseau $R = (X,U,d)$ et on cherche un circuit passant une fois et une seule par tous les sommets de longueur totale minimale.
Façon équivalente de définir le problème : on définit une matrice $n \times n$ dont l'élément $m_{i,j}$ de la i-ème ligne et de la j-ème colonne est égal à

- la distance entre les villes i et j si il existe un moyen d'y aller directement
- $+\infty$ sinon

But : résoudre $\min_p \sum_{i=1}^{n} m_{i,p(i)}$ où p est une permutation sans cycle de n objets.

6.1.2 Remarque

Les graphes s'introduisent de façon très naturelle comme support de modélisation de ces deux exemples. Mais les graphes sont utilisés également dans des modèles pour lesquels leur rôle est moins évident.

6.2 Les graphes

6.2.1 Définitions

Définition 6.2.1 *Un* graphe *est la donnée de*

- *deux ensembles* $X = \{sommets\}$ *et* $U = \{arcs\}$,
- *une application* $IT : U \to X \times X$ *qui à tout arc* u *fait correspondre* $IT(u) = (x,y)$, x *et* y *extrémités initiale et terminale respectivement de* u.

Exemple $X = \{1,2,3,4\}$, $U = \{a,b,c,d,e,f,g\}$
$IT(a) = (1,2)$, $IT(b) = (2,1)$, $IT(c) = (2,2)$, $IT(d) = (2,3)$, $IT(e) = (2,3)$, $IT(f) = (4,32)$ et $IT(g) = (4,1)$.
Représentation :

Notation $G = (X,U)$ ou $G = (X,U,IT)$.

Définition 6.2.2 – *Un arc $u = (xy)$ est une* boucle *si ses extrémités x et y sont confondues (**ex** : c).*

– *Soit $u = (xy)$ un arc. On dit que*
  - *x et y sont* adjacents *à l'arc u,*
  - *x et y sont deux* sommets adjacents,
  - *u est* adjacent *à x et y,*
  - *deux arcs sont* adjacents *s'ils sont adjacents à un même sommet.*

– *Un graphe G est dit* simple *s'il ne possède pas deux arcs ayant même extrémité initiale et même extrémité terminale.*
__rem__ : lorsque le concept de graphe est pris comme fondamental, un graphe général (ie non simple) peut être considéré comme un graphe simple valué $\hat{G} = (\hat{X},\hat{U},m)$ où $m : \hat{U} \to \mathbb{N}$ associe à chaque paire de sommets (x,y) la "multiplicité" de l'arc.

– *Un* chemin *de x à y dans G est une séquence d'arcs de G tels que*
  - *l'extrémité initiale du premier arc de la séquence est x,*
  - *l'extrémité terminale du dernier arc de la séquence est y,*
  - *l'extrémité initiale de chacun des autres arcs coincide avec l'extrémité terminale du précédent.*

– *Un chemin est* simple *s'il n'y a pas de répétitions d'arcs et* élémentaire *s'il n'y a pas de répétitions de sommets (**ex** : a,b,a,d est non simple, (a,d,e,b) est simple mais pas élémentaire et (a,b) est élémentaire).*

– *Un* circuit *est un chemin non vide faisant un aller-retour sur un sommet.*

6.2.2 Représentation matricielle des graphes

Matrice d'adjacence

Définition 6.2.3 *Si $G = (X,U,IT)$ est un graphe comportant n sommets, on lui associe $A = Adj(G) = (a_{ij})_{1\leq i\leq n, 1\leq j\leq n}$ où a_{ij} est le nombre d'arcs de x_i à x_j si $X = \{x_1..x_n\}$ A est la* matrice d'adjacence *de G.*

Exemple La matrice d'adjacence du graphe de l'exemple précédent est

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

Théorème 6.2.1 *Théorème du nombre d'arcs.*
Si $A = Adj(G)$ alors le nombre de chemins en k arcs de x_i à x_j est donné par le terme général a_{ij}^k de la matrice A^k (A à la puissance k).

Matrice booléenne d'adjacence

Définition 6.2.4 *Si $G = (X,U,IT)$ est un graphe comportant n sommets, on lui associe $B = Bool(G) = (b_{ij})_{1 \leq i \leq n, 1 \leq j \leq n}$ où*

$$b_{ij} = \begin{cases} \dot{1} & \text{s'il existe } u/u = (x_i x_j) \\ \dot{0} & \text{sinon} \end{cases}$$

B est la matrice booléenne d'adjacence *de G.*

Remarque Attention en booléens $\dot{1} + \dot{1} = \dot{1}$, ne pas confondre avec $\mathbb{Z}_2$.

Théorème 6.2.2 *Théorème d'existence de chemins.*
Si $B = Bool(G)$ alors il existe un chemin en k arcs de x_i à x_j ssi $b_{ij}^k = \dot{1}$ où $B^k = (b_{ij}^k)_{i,j}$.

6.3 Classification topologique des sommets et réduction

6.3.1 Relation de précédance

Soit $G = (X,U,IT)$ un graphe donné.

Définition 6.3.1 *La relation de précédance de X est définie par $x \leq y$ pour deux sommets x et y ssi il existe un chemin de x à y.*

Cette relation est -réflexive : par convention $x \leq x$ (chemin vide)
-transitive : $x \leq y$ et $y \leq z$ implique $x \leq z$.

On associe à la relation de précédance une matrice booléenne $P = p_{ij}$ où

$p_{ij} = \dot{1}$ ssi $x_i \leq x_j$.

Théorème 6.3.1 *Soit B la matrice booléenne d'adjacence associée à G. Alors*

- *la suite $(I+B)^k$ est stationnaire à partir d'un certain rang r ie*

$$\exists r \quad \textit{minimal,} \quad (I+B)^r = (I+B)^{r+l} \quad \forall l$$

- *sa valeur stationnaire $(I+B)^\infty = (I+B)^r$ n'est autre que la matrice de la relation de précédance.*

Exemple $P = (I+B)^\infty = \begin{pmatrix} \dot{1} & \dot{1} & \dot{1} & \dot{1} & \dot{1} \\ .. & \dot{1} & \dot{1} & \dot{1} & \dot{1} \\ .. & \dot{1} & \dot{1} & \dot{1} & \dot{1} \\ .. & . & . & \dot{1} & . \\ .. & \dot{1} & \dot{1} & \dot{1} & \dot{1} \end{pmatrix}$

Remarque Dans l'exemple précédent, $x_2 \leq x_4$ et $x_4 \leq x_2$ mais $x_2 \neq x_4$. Ce n'est donc pas une relation d'ordre.

6.3.2 Relation de forte connexité et graphe réduit

Un ingénieur du service technique d'une ville propose pour la zone centrale le plan de sens uniques suivant. Avant d'adopter ce plan, il convient d'examiner s'il autorise la circulation des automobilistes ie s'il ne rend pas certains points inaccessibles.
Pour vérifier cette propriété, on associe le graphe de la figure au plan et on introduit la relation de forte connexité :

Définition 6.3.2 *On dit que x_i et x_j deux sommets sont reliés par une relation* de forte connexité $x_i \mathcal{C} x_j$ *si $\{x_i \leq x_j$ et $x_j \leq x_i\}$ ou si $\{x_i = x_j\}$.*

Exemple dans l'exemple, x_2, x_3, x_5 sont reliés deux à deux.

Remarque C'est bien une relation d'équivalence.

Propriétés Sa matrice booléenne est de terme général $p_{ij}p_{ji}$.

Définition 6.3.3 *Les classes d'équivalence de cette relation sont appelées les composantes fortement connexes. Un graphe est dit* fortement connexe *s'il n'a qu'une composante.*

Retour au problème : pour le résoudre, il faut donc voir si le graphe de la figure est fortement connexe.

Définition 6.3.4 *Le* graphe réduit *GR d'un graphe est le graphe dont l'ensemble des sommets est l'ensemble des classes d'équivalence pour $\mathcal{C}$ avec pour arc d'une classe $\mathcal{C}_1$ à $\mathcal{C}_2$ tous les arcs du graphe initial reliant un sommet x_1 de $\mathcal{C}_1$ à x_2 de $\mathcal{C}_2$.*

Exemple $\mathcal{C}_1 = \{x_1\}$, $\mathcal{C}_2 = \{x_2,x_3,x_5\}$ et $\mathcal{C}_3 = \{x_4\}$.

Théorème 6.3.2 *Tout graphe réduit est sans circuit.*

Classification par niveaux

On sait répartir les sommets de tout graphe Γ sans circuit en *niveaux* au sens suivant :

Théorème 6.3.3 *Si Γ est sans circuit, alors il existe une partition ordonnée des ses sommets, soit $N_0 \cup N_1 .. \cup N_l = X$ telle que tout arc va vers un niveau strictement plus grand : si $u = (xy)$ avec $x \in N_i$ et $y \in N_j$ alors $i < j$.*

Idée de preuve :

$N_0 = \{x_0 / x_0$ est sans prédécesseur$\}$

$N_1 = \{x_1 / x_1$ est sans prédécesseur autre que sommet de $N_0\}$...

Retour à l'exemple : $N_0 = \mathcal{C}_1$, $N_1 = \mathcal{C}_2$ et $N_2 = \mathcal{C}_3$.

Remarque La matrice de précédance donne les niveaux par l'étude des lignes identiques.

Retour au problème de plan de circulation

Composantes fortement connexes :
- $\mathcal{C}_1 = \{x_1,x_2,x_3,x_4,x_5,x_9,x_{10},x_{15},x_{16},x_{17},x_{18},x_{19}\}$,
- $\mathcal{C}_2 = \{x_6,x_7,x_{12},x_{13},x_{14},x_8\}$,
- $\mathcal{C}_3 = \{x_{11}\}$.

Classification par niveaux :
- $N_0 = \mathcal{C}_2$,
- $N_1 = \mathcal{C}_3$,
- $N_2 = \mathcal{C}_1$.

6.3.3 Application à la réduction de matrices

Soit

$$A = \begin{pmatrix} 1 & 3 & 2 & 0 \\ 0 & 5 & 0 & 6 \\ 3 & 0 & 4 & 0 \\ 0 & 7 & 0 & 3 \end{pmatrix}$$

A toute matrice $An \times n$ on associe un graphe par rapport à {1,2,..,n} avec un arc de i à j ssi $a_{ij} \neq 0$.

Représentation sous forme de graphe :

Composantes fortement connexes :
-$\mathcal{C}_1 = \{x_1 x_3\}$,
-$\mathcal{C}_2 = \{x_2, x_4\}$.

Classification par niveaux :
-$N_0 = \mathcal{C}_1$,
-$N_1 = \mathcal{C}_2$.

Application à la réduction de matrices
Cela permet de dire qu'il existe une matrice de permutation P telle que

$$P^{-1}AP = \left(\begin{array}{cc|cc} \times & \times & \times & \times \\ \times & \times & \times & \times \\ - & - & - & - \\ 0 & 0 & \times & \times \\ 0 & 0 & \times & \times \end{array}\right)$$

Calcul de la matrice de permutation : changement de base
$e'_1 = e_1$, $e'_2 = e_3$, $e'_3 = e_2$ et $e'_4 = e_4$. D'où

$$P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rappel : $P^{-1} = P^t$ pour les matrices de permutation.

Alors

$$P^{-1}AP = P^tAP \left(\begin{array}{cc|cc} 1 & 2 & 3 & 0 \\ 3 & 4 & 0 & 0 \\ - & - & - & - \\ 0 & 0 & 5 & 6 \\ 0 & 0 & 7 & 3 \end{array} \right)$$

Plus généralement, les niveaux permettent de trouver une matrice de permutation telle que P^tAP s'écrit sous forme réduite.

Chapitre 7

Compléments sur les codes

Problème de l'amélioration des codes.

Soit $H = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$
la matrice de vérification d'un code $\mathcal{C}$.

Deux vecteurs colonnes quelconques sont différents donc linéairement indépendants. Ainsi on sait corriger une erreur.
Par contre, $H_3 = H_1 + H_2 = H_5 + H_6 = H_4 + H_7$. Donc s'il y a deux erreurs (par exemple, $e = [1100000]$ ou $e = [0000110]$), on ne la détecte pas et on confond avec une erreur sur le 3ème bit.

Pour mieux détecter les erreurs, on peut passer dans le cas des codes binaires à :

- la moitié paire
- l'extension paire

7.1 La moitié paire

Théorème 7.1.1 *Théorème de la moitié paire.*
Soit $\mathcal{C}$ un code binaire de distance minimale d impaire et de cardinal e.
Alors l'ensemble $\mathcal{C}^+$ des mots pairs de $\mathcal{C}$ contient $\frac{e}{2}$ mots et a pour distance $d^+ = d + 1$.

Par conséquent, $\mathcal{C}$ corrige $\frac{d-1}{2}$ erreurs alors que $\mathcal{C}^+$ en détecte $\frac{d-1}{2}+1$.

Preuve: $\mathcal{C}^+ = \{[c_1..c_n] \in \mathcal{C} \text{ t.q. } \sum_{i-1}^{n} c_i = 0 \text{ dans } \mathbb{Z}/2\mathbb{Z}\}$.
Ainsi $\mathcal{C}^+$ est le noyau d'une forme linéaire. C'est donc un hyperplan, il est donc de dimension k-1 si $\dim(\mathcal{C}) = k$.
Alors $\mathcal{C} \approx (\mathbb{Z}/2\mathbb{Z})^k$ et $\text{card}(\mathcal{C}) = 2^k$.
De même, $\mathcal{C}^+ \approx (\mathbb{Z}/2\mathbb{Z})^{k-1}$ et $\text{card}(\mathcal{C}^+) = 2^{k-1} = \frac{\text{card}(\mathcal{C})}{2}$.

D'autre part, $\mathcal{C}^+ \subset \mathcal{C}$ donc $d^+ \geq d$. Mais d^+ est un nombre pair (puisque dans $\mathcal{C}^+$, on n'a plus que des mots pairs) et comme d est impaire, $d^+ \geq d+1$.
En fait, on vérifie que $d^+ = d+1$.

7.2 L'extension paire

Théorème 7.2.1 *Théorème de l'extension paire.*
Soit $\mathcal{C}$ un code binaire de distance minimale d impaire et de cardinal e. Soit $\hat{\mathcal{C}} = \{\hat{c} = [c_1..c_n c_{n+1}]$ t.q. $\hat{H}\hat{c}^T\}$ où H est la matrice de vérification du code $\mathcal{C}$ et

$$\hat{H} = \begin{pmatrix} & & & | & 0 \\ & H & & | & \vdots \\ & & & | & 0 \\ - & - & - & - & - \\ 1 & \dots & 1 & | & 1 \end{pmatrix}$$

Alors $\hat{\mathcal{C}}$ contient e mots et a pour distance $d^+ = d+1$.

Par conséquent, $\mathcal{C}$ corrige $\frac{d-1}{2}$ erreurs alors que $\mathcal{C}^+$ en détecte $\frac{d-1}{2}+1$.

Preuve: même genre que la précédente. A faire en exercice.