



PRÉPASIMPLE

prepasimple.fr

HEC

Mathématiques 1

2004

Corrigé de mathématiques approfondies

Mathématiques approfondies — Prépa ECG

Annales de concours · Corrigé

Sur la transmission de messages

Partie I. L'opération Δ sur les parties d'un ensemble

- 1) a) Par définition, $A \Delta \emptyset = A$. De plus, chaque élément de A appartient aux deux ensembles A et E , tandis que chaque élément de $E \setminus A$ appartient seulement à E . Ainsi, $A \Delta E = E \setminus A$.

On en déduit $d(A, \emptyset) = \text{Card}(A)$ et $d(A, E) = n - \text{Card}(A)$.

- 1) b) En appliquant la définition de d à $A \Delta B$ et $\emptyset$, on obtient

$$d(A \Delta B, \emptyset) = \text{Card}((A \Delta B) \Delta \emptyset) = \text{Card}(A \Delta B) = d(A, B).$$

La propriété demandée est donc établie.

- 2) a) L'appartenance à $A \Delta B$ signifie qu'une seule des deux appartenances à A et à B est satisfaite. Le tableau de z_i est donc le suivant :

$x_i \backslash y_i$	0	1
0	0	1
1	1	0

Lorsque $x_i = y_i$, les deux quantités z_i et $|x_i - y_i|$ valent 0. Lorsque $x_i \neq y_i$, elles valent 1. Dans tous les cas, $z_i = |x_i - y_i|$.

En particulier, si A et B sont représentées par les listes x et y ,

$$d(A, B) = \sum_{i=1}^n |x_i - y_i|.$$

2) b) Méthode 1 — Inégalité triangulaire coordonnée par coordonnée

Notons x_i, y_i et t_i les indicatrices respectives des appartenances de e_i à A, B et C .

L'inégalité triangulaire réelle donne $|x_i - t_i| \leq |x_i - y_i| + |y_i - t_i|$.

En sommant les n inégalités et en utilisant la question précédente, on obtient

$$d(A, C) = \sum_{i=1}^n |x_i - t_i| \leq \sum_{i=1}^n |x_i - y_i| + \sum_{i=1}^n |y_i - t_i| = d(A, B) + d(B, C).$$

Méthode 2 — Inclusion d'ensembles

Si un élément appartient à $A \Delta C$, ses appartenances à A et à C sont différentes. Son appartenance à B coïncide alors avec celle à l'un des deux ensembles et diffère de celle à l'autre. Il appartient donc à $A \Delta B$ ou à $B \Delta C$.

Ainsi, $A \Delta C \subset (A \Delta B) \cup (B \Delta C)$. Pour deux ensembles finis F et G , $\text{Card}(F \cup G) \leq \text{Card}(F) + \text{Card}(G)$, puisqu'on peut compter deux fois les éléments communs. On en déduit

$$d(A, C) \leq \text{Card}((A \Delta B) \cup (B \Delta C)) \leq d(A, B) + d(B, C).$$

La première méthode se réutilise pour les distances définies par des sommes de valeurs absolues ; la seconde exprime directement que toute différence entre les extrémités impose une différence sur au moins l'une des deux étapes.

Partie II. Calcul binaire sur des colonnes

Question adaptée au programme actuel. Les opérations du sujet sont traitées comme des règles combinatoires sur des tableaux réels à coefficients 0 ou 1. Les termes « libre pour l'addition binaire » et « base binaire » ont exactement les définitions finies données dans

l'énoncé; on ne leur applique aucun théorème d'algèbre linéaire sur un autre ensemble de scalaires.

Pour alléger les preuves, notons $w(x) = d(x, O)$ le nombre de bits égaux à 1 d'une colonne x . Pour chaque $j \in \{1, \dots, n\}$, notons e_j la colonne dont le seul bit égal à 1 est en position j .

Tout code contient O : il est non vide, donc contient un élément x , et sa stabilité implique $x \oplus x = O \in \mathcal{C}$. Par stabilité répétée, il contient toutes les sommes binaires finies de ses éléments.

1) a) Question adaptée au programme actuel.

La combinaison dont les quatre coefficients sont nuls donne O , donc $\mathcal{C}$ est non vide. Si $u = \bigoplus_{j=1}^4 \varepsilon_j x_j$ et $v = \bigoplus_{j=1}^4 \delta_j x_j$, la distributivité et l'associativité donnent

$$u \oplus v = \bigoplus_{j=1}^4 (\varepsilon_j \oplus \delta_j) x_j \in \mathcal{C}.$$

L'ensemble $\mathcal{C}$ est donc un code.

On calcule $x_1 \oplus x_2 = {}^t(0, 0, 1, 1, 0) = x_3$. Toute combinaison des quatre colonnes se réécrit alors

$$(\varepsilon_1 \oplus \varepsilon_3) x_1 \oplus (\varepsilon_2 \oplus \varepsilon_3) x_2 \oplus \varepsilon_4 x_4.$$

Réciproquement, toute combinaison de x_1, x_2, x_4 est obtenue en prenant $\varepsilon_3 = 0$. Il suffit donc d'énumérer ces huit combinaisons :

Coefficients de x_1, x_2, x_4	Combinaison	Colonne, de haut en bas
000	O	00000
001	x_4	10000

Coefficients de x_1, x_2, x_4	Combinaison	Colonne, de haut en bas
010	x_2	11100
011	$x_2 \oplus x_4$	01100
100	x_1	11010
101	$x_1 \oplus x_4$	01010
110	$x_1 \oplus x_2 = x_3$	00110
111	$x_1 \oplus x_2 \oplus x_4$	10110

Les huit colonnes sont distinctes ; cette liste est exhaustive et $\text{Card}(\mathcal{C}) = 8$.

1) b) Question adaptée au programme actuel.

Pour deux colonnes fixées a_1, a_2 , il existe seulement quatre couples de bits $(\varepsilon_1, \varepsilon_2)$. Leurs combinaisons binaires sont donc au plus les quatre colonnes O, a_1, a_2 et $a_1 \oplus a_2$.

Comme $\mathcal{C}$ contient huit colonnes distinctes, aucune famille de deux éléments ne peut engendrer $\mathcal{C}$ par addition binaire. La réponse est négative.

1) c) Question adaptée au programme actuel.

Supposons $\varepsilon_1 x_1 \oplus \varepsilon_2 x_2 \oplus \varepsilon_4 x_4 = O$. La troisième coordonnée vaut ε_2 , car les troisièmes coordonnées de x_1 et x_4 sont nulles. Ainsi, $\varepsilon_2 = 0$.

La quatrième coordonnée vaut ε_1 , donc $\varepsilon_1 = 0$. La première coordonnée devient alors ε_4 , donc $\varepsilon_4 = 0$.

La famille (x_1, x_2, x_4) est donc libre pour l'addition binaire. Avec la question 1) a), elle est une base binaire de $\mathcal{C}$.

2) a) Pour chaque indice i , le bit $x_i \oplus y_i$ vaut 1 exactement lorsque $x_i \neq y_i$. Le nombre de coordonnées différentes entre x et y est donc le nombre de coordonnées égales à 1 dans $x \oplus y$.

Par conséquent, $d(x, y) = w(x \oplus y) = d(x \oplus y, O)$.

- 2) b) Pour chaque indice i , si $x_i \neq z_i$, au moins l'une des relations $x_i \neq y_i$ ou $y_i \neq z_i$ est vraie. En effet, si elles étaient toutes deux fausses, on aurait $x_i = y_i = z_i$.

En comptant les indices concernés, on obtient $d(x, z) \leq d(x, y) + d(y, z)$. Il s'agit exactement de la propriété de la partie I appliquée aux listes binaires ; la distance est aussi symétrique et $d(x, y) = 0$ équivaut à $x = y$.

- 3) a) Question adaptée au programme actuel.

Comme $\mathcal{C} \neq \{O\}$, il existe $u \in \mathcal{C} \setminus \{O\}$. La famille à un élément (u) est libre pour l'addition binaire : le seul coefficient non nul possible est 1, et $1u = u \neq O$.

Toute famille libre a des éléments non nuls et distincts. Un élément nul donnerait une relation avec un seul coefficient égal à 1 ; deux éléments égaux donneraient une relation avec deux coefficients égaux à 1. Comme $\mathcal{C} \subset K^n$ est fini, une telle famille a au plus $\text{Card}(\mathcal{C}) - 1$ éléments.

L'ensemble des cardinaux possibles des familles libres non vides de $\mathcal{C}$ est donc une partie non vide d'un ensemble fini d'entiers. Il admet un maximum q . Choisissons une famille libre $(u_1, \dots, u_q)$ de ce cardinal.

Supposons qu'une colonne $v \in \mathcal{C}$ ne soit pas une combinaison binaire des u_j . Une relation

$$\bigoplus_{j=1}^q \varepsilon_j u_j \oplus \delta v = O$$

a nécessairement tous ses coefficients nuls : si $\delta = 0$, cela résulte de la liberté des u_j ; si $\delta = 1$, l'addition de v aux deux membres donnerait $v = \bigoplus_{j=1}^q \varepsilon_j u_j$, contrairement au choix de v .

La famille $(u_1, \dots, u_q, v)$ serait donc libre, ce qui contredit la maximalité de q . Tout élément du code est ainsi une combinaison binaire des u_j . Cette famille est une base binaire de $\mathcal{C}$.

3) b) Question adaptée au programme actuel.

L'existence d'une écriture est incluse dans la définition d'une base binaire. Pour l'unicité, supposons

$$\bigoplus_{j=1}^q \varepsilon_j u_j = \bigoplus_{j=1}^q \delta_j u_j.$$

En additionnant les deux membres par $\oplus$, on obtient $\bigoplus_{j=1}^q (\varepsilon_j \oplus \delta_j) u_j = O$. La liberté impose $\varepsilon_j \oplus \delta_j = 0$ pour tout j , donc $\varepsilon_j = \delta_j$.

L'application qui associe à une liste de q bits sa combinaison binaire est donc une bijection de K^q sur $\mathcal{C}$. Puisqu'il y a deux choix indépendants pour chacun des q bits, $\text{Card}(\mathcal{C}) = 2^q$.

Pour $q = 0$, il existe une seule liste vide, qui donne O . La même formule fournit $\text{Card}(\{O\}) = 2^0 = 1$.

3) c) Question adaptée au programme actuel.

Si deux bases binaires ont respectivement q et s éléments, la question précédente donne $2^q = \text{Card}(\mathcal{C}) = 2^s$. La suite des puissances de 2 est strictement croissante sur les entiers naturels, donc $q = s$.

Toutes les bases binaires d'un même code ont ainsi le même cardinal.

3) d) Question adaptée au programme actuel.

Considérons l'ensemble $\mathcal{D} = \left\{ \bigoplus_{j=1}^q \varepsilon_j v_j : (\varepsilon_1, \dots, \varepsilon_q) \in K^q \right\}$. La stabilité du code donne $\mathcal{D} \subset \mathcal{C}$.

La liberté de $(v_1, \dots, v_q)$ assure, par le même argument d'unicité qu'en 3) b), que les 2^q listes de coefficients produisent 2^q colonnes distinctes. Ainsi, $\text{Card}(\mathcal{D}) = 2^q = \text{Card}(\mathcal{C})$.

Deux ensembles finis de même cardinal, dont l'un est inclus dans l'autre, sont égaux. Donc $\mathcal{D} = \mathcal{C}$, et la famille $(v_1, \dots, v_q)$ est une base binaire.

- 4) a) Le produit de Q par la colonne nulle vaut O , donc $\mathcal{C}_Q$ est non vide. Si $x, y \in \mathcal{C}_Q$, la distributivité donne

$$Q \odot (x \oplus y) = (Q \odot x) \oplus (Q \odot y) = O \oplus O = O.$$

Ainsi, $x \oplus y \in \mathcal{C}_Q$, et $\mathcal{C}_Q$ est un code.

- 4) b) Pour chaque $j \in \{1, \dots, p\}$, choisissons une position i_j où la colonne de Q est la j -ième colonne de I_p . Ces positions sont distinctes, car les colonnes de I_p sont distinctes.

Construisons une permutation σ telle que $\sigma(j) = i_j$ pour $1 \leq j \leq p$, puis complétons avec les autres positions dans un ordre quelconque. La matrice obtenue en rangeant les colonnes de Q dans l'ordre $\sigma(1), \dots, \sigma(n)$ est $(I_p \ P)$, où P contient les $n - p$ colonnes restantes.

Si q_j désigne la j -ième colonne de Q , alors, pour tout $x \in K^n$,

$$Q \odot x = \bigoplus_{j=1}^n x_j q_j = \bigoplus_{j=1}^n x_{\sigma(j)} q_{\sigma(j)} = (I_p \ P) \odot {}^t(x_{\sigma(1)}, \dots, x_{\sigma(n)}).$$

La seconde égalité réordonne simplement une liste de termes, ce qui est autorisé par la commutativité et l'associativité. Le même choix de σ convient à toutes les colonnes x .

- 4) c) Question adaptée au programme actuel. La convention de la base vide permet de traiter également $p = n$.

Méthode 1 — Paramétrisation des solutions

Dans l'ordre des coordonnées de la question précédente, écrivons une colonne sous la forme ${}^t(a, b)$, avec $a \in K^p$ et $b \in K^{n-p}$. La condition du code devient $a \oplus (P \odot b) = O$, soit $a = P \odot b$, puisque chaque bit est son propre opposé pour $\oplus$.

Chaque liste b détermine donc une unique colonne du code, et toutes les colonnes du code sont obtenues ainsi. Il y a 2^{n-p} listes possibles, donc $\text{Card}(\mathcal{C}_Q) = 2^{n-p}$.

Si $n > p$, le code n'est pas réduit à $\{O\}$ et la question 3) a) lui assure une base binaire. Si cette base a q éléments, la question 3) b) donne $2^q = 2^{n-p}$, donc $q = n - p$.

Si $n = p$, la liste b est vide et la seule solution est $a = O$. Le code est $\{O\}$ et sa base vide a $0 = n - p$ éléments. Dans tous les cas, une base binaire de $\mathcal{C}_Q$ a $n - p$ éléments.

Méthode 2 — Dénombrement par valeurs du produit

Pour toute colonne $s \in K^p$, il existe une colonne $t_s \in K^n$ telle que $Q \odot t_s = s$: il suffit de placer les bits de s aux positions des colonnes de I_p et de mettre les autres bits à 0.

L'ensemble $F_s = \{x : Q \odot x = s\}$ est en bijection avec $\mathcal{C}_Q$ par $x \mapsto x \oplus t_s$. En effet, $Q \odot (x \oplus t_s) = s \oplus s = O$; inversement, pour $v \in \mathcal{C}_Q$, $v \oplus t_s$ appartient à F_s . Ces deux opérations sont inverses l'une de l'autre.

Les 2^p ensembles F_s sont disjoints et forment une partition de K^n . Ils ont tous le même cardinal $\text{Card}(\mathcal{C}_Q)$. Ainsi,

$$2^n = 2^p \text{Card}(\mathcal{C}_Q), \quad \text{Card}(\mathcal{C}_Q) = 2^{n-p}.$$

L'argument de la première méthode fournit alors une base binaire de cardinal $n - p$, y compris la base vide si $n = p$.

La paramétrisation décrit explicitement les mots du code. Le second raisonnement explique pourquoi chaque valeur du produit $Q \odot x$ correspond au même nombre de mots.

4) d) Question adaptée au programme actuel.

Posons $s = n - p$. Pour une colonne $a \in K^s$ et une colonne $b \in K^p$, la condition $Q \odot {}^t(a, b) = O$ s'écrit $(B \odot a) \oplus b = O$, donc $b = B \odot a$.

Supposons d'abord $s \geq 1$. Notons g_j la j -ième colonne de $\begin{pmatrix} I_s \\ B \end{pmatrix}$. Les s premières coordonnées de g_j forment la j -ième colonne de I_s , et les p dernières la j -ième colonne de B .

Le calcul précédent montre que $g_j \in \mathcal{C}_Q$. Pour tout $x \in \mathcal{C}_Q$, en notant a ses s premières coordonnées, on a

$$x = \begin{pmatrix} a \\ B \odot a \end{pmatrix} = \bigoplus_{j=1}^s a_j g_j.$$

Ces colonnes engendrent donc tout le code. Enfin, si $\bigoplus_{j=1}^s \varepsilon_j g_j = O$, ses s premières coordonnées sont précisément $\varepsilon_1, \dots, \varepsilon_s$; elles sont donc toutes nulles. La famille est libre et constitue une base binaire.

Si $s = 0$, $Q = I_p$ et le code est $\{O\}$; la matrice affichée n'a aucune colonne et la famille vide est bien sa base binaire.

4) e) Question adaptée au programme actuel.

Notons $q_1, \dots, q_n$ les colonnes de Q . Pour toute colonne binaire x , définissons son support $S(x) = \{j : x_j = 1\}$. Par définition du produit binaire,

$$Q \odot x = \bigoplus_{j \in S(x)} q_j, \quad w(x) = \text{Card}(S(x)).$$

L'existence d'une famille liée de r colonnes à des positions distinctes implique $r \leq n$. L'hypothèse de liberté de toutes les familles de $r - 1$ colonnes implique aussi celle de toute famille de moins de $r - 1$ colonnes : on peut compléter les positions choisies à $r - 1$ positions distinctes, et toute relation sur la petite famille se prolongerait en une relation sur la grande en ajoutant des coefficients nuls.

Si $x \in \mathcal{C}_Q \setminus \{O\}$ avait un poids inférieur à r , la relation $\bigoplus_{j \in S(x)} q_j = O$ serait une relation non triviale sur une famille de moins de r colonnes. Elle contredirait la liberté qui vient d'être établie. Ainsi, $w(x) \geq r$ pour tout mot non nul du code.

Choisissons maintenant des positions distinctes $j_1, \dots, j_r$ donnant une famille liée. Il existe des bits $\varepsilon_1, \dots, \varepsilon_r$, non tous nuls, tels que $\bigoplus_{k=1}^r \varepsilon_k q_{j_k} = O$. Aucun coefficient ne peut être nul, car sinon les coefficients égaux à 1 donneraient une relation non triviale sur moins de r colonnes. Tous les ε_k valent donc 1.

La colonne x dont le support est $\{j_1, \dots, j_r\}$ appartient à $\mathcal{C}_Q$ et a exactement le poids r . L'ensemble des poids non nuls est donc non vide, et son minimum est r :

$$\min\{d(x, O) : x \in \mathcal{C}_Q \setminus \{O\}\} = r.$$

Partie III. Un code correcteur d'erreurs

1) Question adaptée au programme actuel.

Les p colonnes de I_p sont non nulles et distinctes, donc apparaissent parmi les colonnes de H . La question II.4) s'applique avec $Q = H$ et $n = 2^p - 1$.

Pour $p \geq 2$, $2^p - 1 > p$. Cette inégalité est vraie pour $p = 2$, et si $2^p - 1 > p$, alors $2^{p+1} - 1 = 2(2^p - 1) + 1 > 2p + 1 > p + 1$. Le code n'est donc pas réduit à $\{O\}$.

Il contient 2^{n-p} éléments et chacune de ses bases binaires a $n - p = 2^p - p - 1$ éléments.

2) Question adaptée au programme actuel.

Deux colonnes distinctes non nulles a et b de H forment une famille libre pour l'addition binaire. En effet, une relation non triviale à deux coefficients aurait l'une des formes $a = O$, $b = O$ ou $a \oplus b = O$. Les deux premières sont exclues par la non-nullité; la troisième équivaut à $a = b$.

Comme $p \geq 2$, choisissons deux colonnes distinctes a, b . La colonne $c = a \oplus b$ est non nulle et différente de a et de b : les égalités contraires donneraient respectivement $a = b$,

$b = O$ ou $a = O$. Elle figure donc à une troisième position de H . La relation $a \oplus b \oplus c = O$ montre que ces trois colonnes sont liées.

La question II.4) e), avec $r = 3$, donne $\min\{w(x) : x \in \mathcal{C}_H \setminus \{O\}\} = 3$.

Pour $u, v \in \mathcal{C}_H$ distincts, la colonne $u \oplus v$ appartient au code et est non nulle. Par II.2) a), $d(u, v) = w(u \oplus v) \geq 3$.

Réciproquement, une colonne du code de poids 3 et la colonne nulle ont une distance égale à 3. La borne est atteinte et

$$\min\{d(u, v) : u, v \in \mathcal{C}_H, u \neq v\} = 3.$$

- 3) a) Une colonne à distance au plus 1 de v est soit v , soit une colonne obtenue en changeant exactement un de ses bits. Il y a n positions possibles ; les colonnes ainsi obtenues sont distinctes et différentes de v .

Ainsi, $B_v = \{v\} \cup \{v \oplus e_j : 1 \leq j \leq n\}$ et $\text{Card}(B_v) = n + 1 = 2^p$.

- 3) b) Supposons qu'une colonne u appartienne à $B_v \cap B_w$. Alors $d(v, u) \leq 1$ et $d(u, w) \leq 1$. L'inégalité triangulaire donne $d(v, w) \leq 2$.

Si v et w sont deux mots distincts du code, cette conclusion contredit la distance minimale 3 de la question 2). Par conséquent, $B_v \cap B_w = \emptyset$.

- 3) c) Question adaptée au programme actuel.

Méthode 1 — Dénombrement des boules disjointes

Il y a 2^{n-p} mots dans le code, et pour chacun d'eux la boule B_v possède 2^p éléments. Ces boules sont deux à deux disjointes d'après 3) b), donc

$$\text{Card} \left(\bigcup_{v \in \mathcal{C}_H} B_v \right) = 2^{n-p} 2^p = 2^n.$$

Cette réunion est incluse dans K^n , qui a lui aussi 2^n éléments. L'égalité des cardinaux finis donne $\bigcup_{v \in \mathcal{C}_H} B_v = K^n$.

Méthode 2 — Construction à partir du produit $H \odot u$

Fixons une colonne quelconque $u \in K^n$ et notons $s = H \odot u$.

Si $s = O$, $u \in \mathcal{C}_H$ et $u \in B_u$. Si $s \neq O$, il existe une unique position j où la colonne h_j de H est égale à s , puisque H contient exactement une fois chaque colonne binaire non nulle de longueur p .

Posons $v = u \oplus e_j$. Par distributivité,

$$H \odot v = (H \odot u) \oplus (H \odot e_j) = s \oplus h_j = O.$$

Ainsi, $v \in \mathcal{C}_H$ et $d(u, v) = 1$, donc $u \in B_v$. Toute colonne appartient à la réunion, ce qui établit la même égalité.

Le dénombrement montre que les boules occupent exactement tout l'ensemble des colonnes ; la seconde preuve fournit directement la colonne à choisir pour corriger une erreur.

4) a) La question 3) c) assure l'existence d'un mot $v \in \mathcal{C}_H$ tel que $d(z, v) \leq 1$. Comme z n'appartient pas au code, on ne peut pas avoir $z = v$, donc $d(z, v) = 1$.

Si deux mots du code convenaient, leurs boules de rayon 1 contiendraient toutes deux z , contrairement à 3) b). Ce mot est donc unique et définit $\Phi(z)$.

- 4) b) Notons $s = H \odot z$. Comme $z \notin \mathcal{C}_H$, $s \neq O$. Il existe une unique colonne h_j de H égale à s .

Les colonnes de poids 1 sont précisément $e_1, \dots, e_n$, et $H \odot e_i = h_i$. Il existe donc un seul e de poids 1 satisfaisant $H \odot e = H \odot z$: c'est e_j .

Le produit de $z \oplus e_j$ par H vaut $s \oplus s = O$, et sa distance à z vaut 1. Par l'unicité de 4) a), $\Phi(z) = z \oplus e$.

La colonne s est appelée le syndrome reçu : elle indique ici la position du bit à changer lorsqu'il y a une seule erreur.

- 5) a) Question adaptée au programme actuel.

Les trois dernières colonnes de H_1 forment un bloc terminal I_3 aux positions 5, 6, 7. Plus précisément,

$$H_1 = (B \ I_3), \quad B = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix}.$$

La matrice dont les colonnes sont c_1, c_2, c_3, c_4 est exactement $\begin{pmatrix} I_4 \\ B \end{pmatrix}$. La question II.4) d)

assure donc que cette famille est une base binaire de $\mathcal{C}_{H_1}$.

On peut aussi vérifier directement le mécanisme : pour une liste de quatre bits (a_1, a_2, a_3, a_4) , la combinaison $\bigoplus_{j=1}^4 a_j c_j$ a pour quatre premières coordonnées ces mêmes bits, et pour trois dernières

$$(a_1 \oplus a_3 \oplus a_4, a_1 \oplus a_2 \oplus a_4, a_1 \oplus a_2 \oplus a_3).$$

Ces trois bits annulent respectivement les trois sommes calculées par les lignes de H_1 .

Chaque liste de quatre bits fournit un mot du code, et ses quatre premières coordonnées permettent de retrouver la liste de manière unique.

- 5) b) Les bits égaux à 1 de $y^* = {}^t(0, 1, 0, 0, 1, 1, 0)$ sont aux positions 2, 5, 6. On calcule donc

$$H_1 \odot y^* = h_2 \oplus h_5 \oplus h_6 = \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} \oplus \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \oplus \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}.$$

Cette colonne est la troisième colonne h_3 de H_1 . Sous l'hypothèse d'une seule erreur, la question 4) b) indique qu'il faut changer le troisième bit.

Le mot transmis était donc $y = y^* \oplus e_3 = {}^t(0, 1, 1, 0, 1, 1, 0)$. Il appartient bien au code, puisque son syndrome vaut $h_3 \oplus h_3 = O$.

Comme les quatre premières coordonnées de c_1, c_2, c_3, c_4 forment I_4 , les quatre premières coordonnées de y sont les quatre coefficients transmis. Ainsi,

$$(\eta_1, \eta_2, \eta_3, \eta_4) = (0, 1, 1, 0).$$

On peut vérifier l'encodage : $c_2 \oplus c_3 = {}^t(0, 1, 1, 0, 1, 1, 0) = y$.

- 5) c) Supposons que les deux erreurs se trouvent à deux positions distinctes i et j . La colonne reçue est $z^* = z \oplus e_i \oplus e_j$, avec $H_1 \odot z = O$. Son syndrome est donc $s = h_i \oplus h_j$.

Les colonnes de H_1 étant distinctes, $h_i \oplus h_j \neq O$. Le syndrome non nul détecte donc la présence d'erreurs. Il ne permet cependant pas de conclure qu'il y a exactement deux erreurs si ce nombre n'est pas connu à l'avance : le même syndrome correspondrait aussi à une erreur unique à la position de la colonne égale à s .

Même en sachant qu'il y a exactement deux erreurs, on ne peut pas déterminer les deux positions. En effet, fixons un syndrome non nul s . Parmi les sept colonnes non nulles de longueur 3, excluons la colonne égale à s . Pour chacune des six colonnes restantes a , la colonne $a \oplus s$ est non nulle, différente de a et différente de s . Elle apparaît parmi ces six colonnes, et $a \oplus (a \oplus s) = s$.

Les six colonnes se regroupent donc en trois paires distinctes, chacune ayant pour somme binaire s . Par exemple, pour $s = {}^t(1, 1, 1) = h_1$, les paires de positions sont $\{2, 5\}$, $\{3, 6\}$ et $\{4, 7\}$, car

$$h_2 \oplus h_5 = h_3 \oplus h_6 = h_4 \oplus h_7 = h_1.$$

Plus généralement, chacune des trois paires compatibles avec le syndrome donne un mot transmis possible. Si $\{a, b\}$ est l'une de ces paires, $\tilde{z} = z^* \oplus e_a \oplus e_b$ vérifie $H_1 \odot \tilde{z} = s \oplus h_a \oplus h_b = O$ et diffère de z^* en exactement deux positions.

Les trois mots ainsi obtenus sont distincts, puisque leurs ensembles de deux positions modifiées sont distincts. La donnée de la colonne reçue et du syndrome ne suffit donc pas à choisir entre eux : deux erreurs sont détectables mais ne sont pas corrigeables de façon unique par ce code.

Partie IV. Distinguere falsum vero

1) Question adaptée au programme actuel.

L'écriture binaire des entiers $1, \dots, 2^p - 1$ donne exactement les colonnes binaires non nulles de longueur p , sans répétition. La matrice H_2 satisfait donc les hypothèses de la partie III.

Si l'unique erreur est en position j , on a $y^* = y \oplus e_j$. Le mot y , construit à partir d'une base binaire du code, appartient à $\mathcal{C}_{H_2}$, donc

$$H_2 \odot y^* = (H_2 \odot y) \oplus (H_2 \odot e_j) = O \oplus h_j = h_j.$$

La i -ième coordonnée du syndrome vaut donc $x_i = \varepsilon_{ij}$. Par définition de la j -ième colonne de H_2 ,

$$k = \sum_{i=1}^p x_i 2^{i-1} = \sum_{i=1}^p \varepsilon_{ij} 2^{i-1} = j.$$

L'erreur est exactement à la composante numéro k . Le nombre k appartient à $\{1, \dots, n\}$ sous l'hypothèse d'une erreur unique. En l'absence d'erreur, le syndrome serait nul et $k = 0$; il ne faudrait alors modifier aucune composante.

2) a) Question adaptée au programme actuel.

Les trois bits des entiers de 1 à 7, rangés du poids faible au poids fort, donnent les colonnes suivantes :

Numéro de colonne	Bit de poids 1	Bit de poids 2	Bit de poids 4
1	1	0	0
2	0	1	0
3	1	1	0
4	0	0	1
5	1	0	1
6	0	1	1
7	1	1	1

Ainsi,

$$H_2 = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}.$$

Méthode 1 — Appartenance, liberté et dénombrement

Les supports des quatre colonnes proposées sont respectivement $\{1, 6, 7\}$, $\{2, 5, 7\}$, $\{3, 5, 6\}$ et $\{4, 5, 6, 7\}$. Les sommes binaires des colonnes correspondantes de H_2 valent :

$$h_1 \oplus h_6 \oplus h_7 = {}^t(1, 0, 0) \oplus {}^t(0, 1, 1) \oplus {}^t(1, 1, 1) = O,$$

$$h_2 \oplus h_5 \oplus h_7 = {}^t(0, 1, 0) \oplus {}^t(1, 0, 1) \oplus {}^t(1, 1, 1) = O,$$

$$h_3 \oplus h_5 \oplus h_6 = {}^t(1, 1, 0) \oplus {}^t(1, 0, 1) \oplus {}^t(0, 1, 1) = O,$$

$$h_4 \oplus h_5 \oplus h_6 \oplus h_7 = {}^t(0, 0, 1) \oplus {}^t(1, 0, 1) \oplus {}^t(0, 1, 1) \oplus {}^t(1, 1, 1) = O.$$

Donc chaque d_j appartient au code. Les quatre premières coordonnées des d_j forment les colonnes de I_4 , donc toute relation binaire nulle entre eux impose successivement que ses quatre coefficients soient nuls. La famille est libre.

La partie III.1) donne un cardinal de base égal à $7 - 3 = 4$. La question II.3) d) montre alors que cette famille libre de quatre éléments est une base binaire de $\mathcal{C}_{H_2}$.

Méthode 2 — Résolution des contraintes de parité

Pour $u = {}^t(u_1, \dots, u_7)$, le produit $H_2 \odot u = O$ équivaut aux trois égalités

$$u_1 \oplus u_3 \oplus u_5 \oplus u_7 = 0, \quad u_2 \oplus u_3 \oplus u_6 \oplus u_7 = 0, \quad u_4 \oplus u_5 \oplus u_6 \oplus u_7 = 0.$$

Les deux premières donnent $u_5 = u_1 \oplus u_3 \oplus u_7$ et $u_6 = u_2 \oplus u_3 \oplus u_7$. En les remplaçant dans la troisième, chaque bit répété deux fois s'annule et on obtient

$$u_7 = u_1 \oplus u_2 \oplus u_4.$$

Il s'ensuit

$$u_5 = u_2 \oplus u_3 \oplus u_4, \quad u_6 = u_1 \oplus u_3 \oplus u_4.$$

Toute liste des quatre premiers bits fournit donc une unique solution, donnée par

$$u = \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_2 \oplus u_3 \oplus u_4 \\ u_1 \oplus u_3 \oplus u_4 \\ u_1 \oplus u_2 \oplus u_4 \end{pmatrix} = u_1 d_1 \oplus u_2 d_2 \oplus u_3 d_3 \oplus u_4 d_4.$$

Cette égalité prouve que les d_j engendrent le code ; la lecture des quatre premières coordonnées assure leur liberté et l'unicité des coefficients. Ils constituent donc une base binaire.

La première méthode permet de vérifier rapidement une famille proposée. La seconde construit explicitement l'encodage à partir des contraintes de parité.

- 2) b) Le message de longueur 21 se découpe, dans l'ordre de transmission, en trois colonnes de longueur 7 :

$$1110110, \quad 1000011, \quad 0010111.$$

Pour chaque bloc reçu v , on calcule le syndrome $s = H_2 \odot v$, puis $k = s_1 + 2s_2 + 4s_3$. Sous l'hypothèse d'au plus une erreur, $k = 0$ signifie qu'aucune erreur ne s'est produite ; sinon, on change le bit en position k .

Pour le premier bloc $v = {}^t(1, 1, 1, 0, 1, 1, 0)$, les trois sommes entières calculées par les lignes de H_2 sont 3, 3 et 2. Le syndrome est donc ${}^t(1, 1, 0)$, et $k = 1 + 2 = 3$. Le bloc corrigé est ${}^t(1, 1, 0, 0, 1, 1, 0)$.

Pour le deuxième bloc $v = {}^t(1, 0, 0, 0, 0, 1, 1)$, les sommes sont 2, 2 et 2. Le syndrome est nul, donc le bloc est conservé.

Pour le troisième bloc $v = {}^t(0, 0, 1, 0, 1, 1, 1)$, les sommes sont 3, 3 et 3. Le syndrome est ${}^t(1, 1, 1)$, et $k = 1 + 2 + 4 = 7$. Le bloc corrigé est ${}^t(0, 0, 1, 0, 1, 1, 0)$.

Les coefficients de l'encodage sont les quatre premières coordonnées de chaque bloc corrigé, puisque les quatre premières lignes de la matrice $(d_1 \ d_2 \ d_3 \ d_4)$ forment I_4 .

Bloc reçu	Syndrome	Position corrigée	Bloc corrigé	Quatre coefficients	Chiffre
1110110	110	3	1100110	1100	3
1000011	000	Aucune	1000011	1000	1

Bloc reçu	Syndrome	Position corrigée	Bloc corrigé	Quatre coefficients	Chiffre
0010111	111	7	0010110	0010	4

Attention à l'ordre des bits : les coefficients sont rangés du poids faible au poids fort. Les valeurs des chiffres sont donc

$$x = 1 + 2 \times 1 + 4 \times 0 + 8 \times 0 = 3,$$

$$y = 1 + 2 \times 0 + 4 \times 0 + 8 \times 0 = 1,$$

$$z = 0 + 2 \times 0 + 4 \times 1 + 8 \times 0 = 4.$$

Les chiffres décodés sont exactement 3, 1, 4 sous l'hypothèse d'au plus une erreur par bloc. Ils suggèrent que Primus et Secundus découvrent π , dont les premiers chiffres sont 3, 14. Trois chiffres significatifs ne suffisent cependant pas à identifier une constante avec certitude, ni à déterminer la position de sa virgule ; la conclusion sur π est l'interprétation probable demandée, et non une conséquence unique du décodage.

2) c) Code adapté en Python. Question adaptée au programme actuel.

Le programme calcule les trois parités, transforme le syndrome en numéro de position, corrige éventuellement un bit, puis lit les quatre premiers bits comme les coefficients de poids 1, 2, 4, 8. Les positions mathématiques de 1 à 7 correspondent aux indices Python de 0 à 6 : un numéro k impose donc de modifier `v[k - 1]`.

Le paramètre `dtype=int` de `np.array` ou `np.zeros` indique que les coefficients du tableau sont des entiers. Le produit `np.dot(H2, v)` est d'abord un produit matriciel ordinaire ; l'opération `% 2` transforme ses trois coefficients en parités. La chaîne utilisée

ci-dessous doit comporter exactement trois blocs de sept caractères, chacun égal à 0 ou à 1.

Python

```
import numpy as np

H2 = np.array([
    [1, 0, 1, 0, 1, 0, 1],
    [0, 1, 1, 0, 0, 1, 1],
    [0, 0, 0, 1, 1, 1, 1]
], dtype=int)

def decode_colonne(recue):
    # On travaille sur une copie de la colonne reçue.
    v = np.array(recue, dtype=int)
    syndrome = np.dot(H2, v) % 2
    k = int(syndrome[0] + 2 * syndrome[1] + 4 * syndrome[2])
    if k != 0:
        v[k - 1] = 1 - v[k - 1]
    chiffre = int(v[0] + 2 * v[1] + 4 * v[2] + 8 * v[3])
    return chiffre, v, k

def decode_message(message):
    chiffres = np.zeros(3, dtype=int)
    positions = np.zeros(3, dtype=int)
    corriges = np.zeros((3, 7), dtype=int)
    for j in range(3):
        recue = np.zeros(7, dtype=int)
        for i in range(7):
            recue[i] = int(message[7 * j + i])
        chiffre, v, k = decode_colonne(recue)
        chiffres[j] = chiffre
```

```
positions[j] = k
corriges[j, :] = v
return chiffres, corriges, positions

message = "111011010000110010111"
chiffres, corriges, positions = decode_message(message)
print(chiffres)
print(corriges)
print(positions)
```

L'exécution donne les trois sorties suivantes : les chiffres sont $[3, 1, 4]$, les lignes corrigées sont $[1, 1, 0, 0, 1, 1, 0]$, $[1, 0, 0, 0, 0, 1, 1]$ et $[0, 0, 1, 0, 1, 1, 0]$, et les positions corrigées sont $[3, 0, 7]$.

La boucle extérieure parcourt les trois blocs. La boucle intérieure extrait les sept bits du bloc courant. Le numéro 0 dans la dernière sortie signifie qu'aucun bit n'a été modifié dans le deuxième bloc.

Le code a été exécuté sur le message fourni. Il a également été vérifié pour chacune des 16 listes de quatre bits, d'abord sans erreur, puis avec une erreur à chacune des sept positions, soit 128 situations. Dans chaque cas, le mot et les quatre coefficients initiaux ont été retrouvés. Ce contrôle confirme l'implémentation ; la garantie mathématique provient des preuves précédentes et de l'hypothèse d'au plus une erreur par bloc.

Si deux erreurs ou davantage sont possibles, le programme peut produire un mot différent de celui envoyé ; la question III.5) c) explique pourquoi le syndrome ne permet pas de corriger alors de manière unique. Les valeurs de 10 à 15 peuvent correspondre à des listes de quatre bits, mais ne sont pas des chiffres décimaux : elles ne peuvent pas apparaître ici si l'encodage porte sur un chiffre de 0 à 9 et si l'hypothèse sur les erreurs est satisfaite.

Relevé des adaptations

II, préambule ; II.1) a) à c), II.3) a) à d), II.4) c) à e), III.1), III.2), III.3) c), III.5) a), IV.1) et IV.2) a) : les opérations sur $\{0, 1\}$ sont définies comme des calculs de parité sur des tableaux réels. Les notions de liberté et de base sont reformulées en notions combinatoires de liberté pour l'addition binaire et de base binaire. Les démonstrations utilisent des relations finies et du dénombrement, sans théorie des espaces vectoriels sur un autre ensemble de scalaires. Dans l'ensemble des parties II à IV, les symboles $\oplus$ et $\odot$ distinguent explicitement ces opérations de l'addition réelle et du produit matriciel réel ; les données binaires sont conservées.

II.4) b) à d) : le cas autorisé $p = n$ est précisé par des blocs vides et par la convention selon laquelle la famille vide est une base binaire de $\{O\}$. Cela permet de conserver les demandes de cardinalité et de base sans exclure un cas de l'énoncé.

II.4) e) : les colonnes d'une famille sont prises à des positions distinctes, afin de préciser le sens de la recherche de la plus petite relation non triviale entre colonnes.

IV.2) c) : la demande de programme est convertie en Python. La syntaxe du reste de division par 2 et de la conversion en entier est fournie dans le sujet ; le corrigé donne et explique un programme complet, exécuté sur le message et sur les 128 situations de test de l'encodage de quatre bits avec au plus une erreur.