

Chapitre 10

Correction d'erreur en MQ

Dans la théorie classique la façon usuelle de compenser les effets du bruit sur le stockage ou la communication des données est d'introduire de la redondance de façon suffisamment structurée. C'est ce que l'on appelle la correction d'erreurs. La correction d'erreurs est efficace dans la mesure où l'information est digitale. En effet pour des signaux analogues les perturbations sont continues ce qui rend la correction d'erreurs est impraticable. On pourrait maintenant croire que dans le cas quantique la correction d'erreurs est impraticable (ou néanmoins difficile) car les états de l'espace d'Hilbert forment un continuum. D'autrepart le processus de mesure tend à détruire l'état quantique du système, ce qui n'est pas le cas dans le monde classique. Parfois des arguments ont été avancés pour dire que la correction d'erreur n'est pas possible pour des états quantiques, mais il s'est avéré (suite aux travaux de Shor) qu'il n'en est rien. La nature discrète de la MQ se manifeste dans la classification des perturbations possibles et on peut encore construire des codes correcteurs d'erreurs.

Dans ce chapitre nous donnons quelques constructions élémentaires de codes quantiques. Les idées générales seront développées en partant des codes les plus simples qui sont les codes répétitions quantiques. En concaténant deux codes de répétitions quantiques - relatifs à des bases différentes de l'espace de Hilbert - on obtient déjà un code intéressant ! Comme nous le verrons on peut développer des constructions assez générales en élaborant cette idée.

10.1 Bref rappel sur les codes linéaires classiques

Le code correcteur le plus simple que l'on puisse imaginer est le code de répétition. Supposons que l'on veuille transmettre un bit 0 ou 1 à travers un canal BSC qui renverse le bit avec probabilité $0 < p < \frac{1}{2}$. On pourrait utiliser le code de répétition

$$\begin{aligned} 0 &\longrightarrow 000 \\ 1 &\longrightarrow 111 \end{aligned}$$

Pour aucun ou un seul renversement on peut facilement détecter puis corriger l'erreur grâce à la règle de la majorité. Par exemple si 010 est reçu on en conclut que probablement le deuxième bit a été renversé et on décode en faisant l'opération $010 \rightarrow 000$. Ce faisant, peut-être que l'on fait une erreur de décodage car il se pourrait qu'en fait 111 était effectivement transmis et que les premiers et troisièmes bits aient été renversés par le canal. Mais la probabilité de cet événement est faible si p est assez petit. Sans répétition la probabilité d'erreur de décodage sera toujours égale à p alors qu'avec répétition elle est égale à $p^3 + 3p^2(1-p) = 3p^2 - 2p^3$ et est inférieure à p pour tout $0 < p < \frac{1}{2}$.

Définition. Un code linéaire est un sous-espace vectoriel de dimension k de \mathbb{F}_2^n . Le nombre de bits d'information est k , le nombre de mots de code est 2^k , leur longueur est n . Le rendement du code est défini par le rapport entre le nombre de bits d'information contenus dans un mot et sa longueur. On dit que les paramètres du code sont (n, k) . Il existe deux représentations fondamentales d'un code linéaire. L'une utilise la matrice génératrice et l'autre la matrice de parité.

Matrice génératrice. Un code binaire linéaire peut être décrit par une application linéaire

$$\begin{aligned} \mathbb{F}_2^k &\rightarrow \mathbb{F}_2^n \\ \vec{u} &\mapsto G\vec{u} = \vec{x} \end{aligned}$$

où $\vec{u} = \begin{pmatrix} u_1 \\ \vdots \\ u_k \end{pmatrix}$ est un vecteur colonne contenant k bits d'information et G une matrice (d'éléments 0 ou 1) $n \times k$. Le vecteur $G\vec{u}$ est le mot de code, vecteur binaire à n composantes. Toutes les opérations sont faites mod 2. Pour que l'image de \mathbb{F}_2^k dans \mathbb{F}_2^n soit un sous-espace vectoriel de dimension k

on prend une matrice G de rang k . C'est à dire que les k colonnes de G sont linéairement indépendantes. La matrice G s'appelle la matrice génératrice.

Par exemple, le code de répétition $(3, 1)$ correspond à la matrice 3×1

$$G = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

Matrice de Parité. Puisqu'un code linéaire est un sous-espace vectoriel de \mathbb{F}_2^n , il peut être vu comme le noyau d'une matrice H , c'est à dire l'ensemble des vecteurs solutions de

$$H\vec{x} = 0, \quad H \text{ de dimension } (n - k) \times n$$

Puisque l'on veut $\dim \ker(H) = k$ et que $\dim \ker(H) + \dim \text{rang}(H) = n$ on demande $\dim \text{rang}(H) = n - k$. C'est à dire que H doit posséder au moins $n - k$ lignes indépendantes. En général on prendra H de dimension $(n - k) \times n$ mais son choix n'est pas unique.

Relation entre les deux matrices. On observe que pour un code donné $HG = 0$; H et G sont duales. Pour construire H à partir de G on peut écrire $G = [\vec{g}_1, \dots, \vec{g}_k]$ où $\vec{g}_1, \dots, \vec{g}_k$ sont indépendants et prendre $n - k$ vecteurs orthogonaux $\vec{h}_1, \dots, \vec{h}_{n-k}$ au sous-espace engendré par $\vec{g}_1, \dots, \vec{g}_k$. Alors

$$H = \begin{bmatrix} \vec{h}_1^T \\ \vdots \\ \vec{h}_{n-k}^T \end{bmatrix}$$

Réciproquement pour construire G à partir de H on prend les $n - k$ vecteurs lignes indépendants de H et on construit k vecteurs orthogonaux (au sous-espace engendré par les lignes de H) qui forment les k colonnes de G .

Par exemple une matrice de parité pour le code de répétition est la matrice 2×3 égale à

$$H = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$$

Détection, correction d'erreurs et distance minimale. La matrice de parité permet de détecter certaines erreurs (mais pas toutes). Supposons que le mot \vec{x} soit transmis à travers un canal et que \vec{x}' soit reçu : $\vec{x}' = \vec{x} + \vec{e}$ où \vec{e} est un vecteur contenant des 1 aux composantes erronées et 0 ailleurs. On a :

$$H\vec{x}' = H\vec{x} + H\vec{e} = H\vec{e}$$

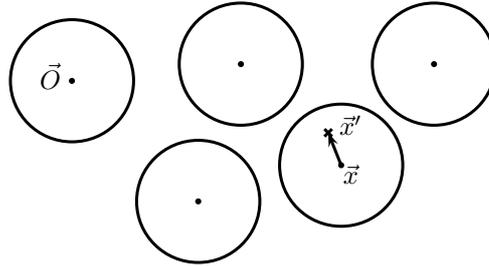
On appelle $H\vec{e}$ le syndrome. Si celui-ci est non-nul il indique que le mot reçu est erroné. Notez que s'il est nul il peut y avoir ou non erreur de transmission.

Pour le canal BSC la probabilité d'avoir un vecteur d'erreur \vec{e} est $p^{|\vec{e}|}(1-p)^{n-|\vec{e}|}$ et il est naturel (pour p petit) de décoder \vec{x}' en déclarant que \vec{x} est le vecteur qui minimise $d(\vec{x}, \vec{x}') = |\vec{e}|$, la distance de Hamming entre \vec{x} et \vec{x}' . En effet \vec{x} est le vecteur transmis le plus probable (qui maximise $p^{|\vec{e}|}(1-p)^{n-|\vec{e}|}$) étant donné que \vec{x}' a été reçu (les vecteurs transmis sont choisis uniformément). C'est le décodeur dit de "distance minimale".

Un paramètre important qui caractérise la qualité d'un code est sa distance minimale $d = \min_{x, x' \in \mathcal{C}} d(\vec{x}, \vec{x}')$. Pour un code linéaire on a aussi $d = \min_{x \neq 0 \in \mathcal{C}} d(\vec{x}, 0)$. On parle alors de code (n, k, d) : longueur n , dimension k et distance minimale d .

Théorème : *Soit un code (n, k, d) . On peut toujours corriger jusqu'à t erreurs avec $2t + 1 \leq d$ grâce au décodeur de la distance minimale.*

Si on considère les boules de Hamming de rayon $\frac{d-1}{2}$ autour des mots de codes, celles-ci ne s'intersectent pas. De plus si le nombre d'erreur est $\leq \frac{d-1}{2}$ alors le mot reçu \vec{x}' est certainement dans une et une seule des boules. On déclare que le mot transmis est l'unique mot de code dans la même boule que \vec{x}' .



Théorème : *Pour tout code linéaire, le nombre minimal de colonnes de H qui sont linéairement indépendantes donne exactement d . En particulier toute collection de $d - 1$ colonnes sont linéairement indépendantes.*

Pour s'en convaincre il suffit de remarquer que si \vec{x} est un mot de code $H\vec{x} = 0$, ce qui signifie que les colonnes (de H) indexées par les composantes non nulles de \vec{x} , sont linéairement indépendantes.

Rappel sur les codes de Hamming. Ceux-ci sont commodément décrits par leur matrice de parité. Soit $r \geq 2$ et H la matrice de parité dont les colonnes sont tous les $2^r - 1$ vecteurs binaires non nuls de longueur r . On a $n = 2^r - 1$ et $n - k = r \Rightarrow k = 2^r - r - 1$. On obtient donc un code $(n, k) = (2^r - 1, 2^r - r - 1)$ de longueur $2^r - 1$ et dimension $2^r - r - 1$. Il faut vérifier que le rang de cette matrice est bien r : cela est vrai car parmi

les $2^r - 1$ vecteurs binaires non nuls, il y en a r indépendants qui engendrent tout \mathbb{F}_2^r . Pour les codes de Hamming on voit facilement que $d = 3$. En effet l'ensemble des colonnes est formé de *tous* les vecteurs binaires non-nuls : il y en a nécessairement trois qui sont linéairement dépendants, et de plus toute paire de colonnes distinctes sont différentes. Ainsi ces codes corrigent une erreur. Par exemple le code de Hamming $(7, 4, 3)$ possède la matrice (ici $r = 3$)

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

Supposons qu'une erreur soit produite dans le 3^{ième} bit. Alors $\vec{e} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$

et le syndrome vaut $H\vec{e} = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$ qui est égal à la 3^{ième} colonne de H . Le

syndrome nous montre automatiquement quel bit est erroné et il suffit de le renverser pour effectuer la correction d'erreur. On est assuré que l'erreur est correctement détectée et corrigée, *seulement* si elle est unique.

Exercice : Donnez la liste des syndromes possibles pour le code de répétition $(3, 1, 3)$ et vérifiez que sa connaissance permet de corriger une erreur unique.

10.2 Codes de Répétition Quantique

Par analogie avec le cas classique on peut considérer le code de répétition

$$\begin{aligned} |0\rangle &\rightarrow |000\rangle \\ |1\rangle &\rightarrow |111\rangle. \end{aligned}$$

que nous allons utiliser sur un *canal bit-flip*.

Quel est le procédé de détection et correction dans le cas quantique ? Il faut choisir correctement la base de mesure pour ne pas détruire irrémédiablement l'information contenue dans l'état sortant ! Ici l'espace d'Hilbert à la sortie du canal est de dimension $2^3 = 8$ et il faut une base à 8 dimensions.

Considérons les 4 projecteurs de dimension 2 chacun (donc 8 au total)

$$\begin{aligned} P_0 &= |000\rangle\langle 000| + |111\rangle\langle 111| \\ P_1 &= |100\rangle\langle 100| + |011\rangle\langle 011| \\ P_2 &= |010\rangle\langle 010| + |101\rangle\langle 101| \\ P_3 &= |001\rangle\langle 001| + |110\rangle\langle 110|. \end{aligned}$$

P_0 projette dans le sous-espace à 0 erreurs, P_1 dans le sous-espace à une erreur sur le premier bit, P_2 dans le sous-espace à une erreur sur le deuxième bit, P_3 dans le sous-espace à une erreur sur le 3^{ième} bit. L'entrée du canal est le ket $\alpha |0\rangle + \beta |1\rangle$ codé :

$$\alpha |000\rangle + \beta |111\rangle$$

Notez que ce code de répétition ne viole pas le "no-cloning theorem" car on répète uniquement des états de base orthonormés. Ici on ne répète pas l'état $\alpha |0\rangle + \beta |1\rangle$, ce qui violerait le no-cloning theorem. Si une seule erreur est produite, disons pour le deuxième bit, la sortie est

$$\alpha |010\rangle + \beta |101\rangle$$

Une mesure dans la base $\{P_0, P_1, P_2, P_3\}$ préserve l'état avec probabilité 1. En effet

$$P_2(\alpha |010\rangle + \beta |101\rangle) = \alpha |010\rangle + \beta |101\rangle, \quad P_i(\alpha |010\rangle + \beta |101\rangle) = 0, \quad i \neq 2.$$

Ainsi on détecte que l'erreur est dans le deuxième bit sans détruire l'état. La correction d'erreur se fait en appliquant l'opérateur unitaire $\mathbb{I} \otimes X \otimes \mathbb{I}$ qui renverse le deuxième bit.

$$\mathbb{I} \otimes X \otimes \mathbb{I}(\alpha |010\rangle + \beta |101\rangle) = \alpha |000\rangle + \beta |111\rangle.$$

Bien sur (comme dans le cas classique) on ne détecte pas correctement des renversements de deux ou trois bits.

Il est utile de d'écrire le procédé de correction d'erreur d'un autre point de vue. Considérons un appareil de mesure qui mesure les observables

$$Z_1 \otimes Z_2 \otimes \mathbb{I} \quad \text{et} \quad \mathbb{I} \otimes Z_2 \otimes Z_3.$$

Notons que la mesure simultanée de ces observables (avec un seul appareil de mesure) est possible car elles commutent. La mesure de chaque observable donne les résultats (± 1) et (± 1) . Ces mesures constituent l'analogie du

“syndrome classique” :

+1	et	+1	→	pas d’erreur
-1	et	+1	→	erreur dans le 1 ^{er} bit
-1	et	-1	→	erreur dans le 2 ^{ème} bit
+1	et	-1	→	erreur dans le 3 ^{ème} bit.

Une fois l’erreur détectée on la corrige en appliquant un opérateur unitaire :

pas d’erreur	→	$\mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I}$
1 ^{er} bit	→	$X_1 \otimes \mathbb{I} \otimes \mathbb{I}$
2 ^{ème} bit	→	$\mathbb{I} \otimes X_2 \otimes \mathbb{I}$
3 ^{ème} bit	→	$\mathbb{I} \otimes \mathbb{I} \otimes X_3$.

Toutes les opérations – encodage, transmission, mesure, correction – sont permises par la MQ (opérateurs unitaires ou mesures projectives).

Exercice Proposez un circuit pour réaliser l’opération d’encodage.

Ce code n’est pas efficace pour protéger l’état contre les erreurs (disons uniques) produites par un canal “phase-flip”. En effet un phase-flip (sur n’importe-quel bit) produit l’état

$$\alpha|000\rangle - \beta|111\rangle$$

à la sortie du canal. On voit facilement que le syndrome est (+1, +1) qui est interprété comme une absence d’erreur. Un code de répétition utile pour le canal “phase-flip” (mais du coup, inutile pour le bit-flip) est obtenu en travaillant dans la base $|+\rangle = \frac{|0\rangle+|1\rangle}{\sqrt{2}}$; $|-\rangle = \frac{|0\rangle-|1\rangle}{\sqrt{2}}$.

Encodage :

$$\begin{aligned} |0\rangle &\rightarrow |+\rangle \rightarrow |+++ \rangle \\ |1\rangle &\rightarrow |-\rangle \rightarrow |-- \rangle. \end{aligned}$$

Détection d’erreur :

Mesurer les observables (qui commutent)

$$X_1 \otimes X_2 \otimes \mathbb{I} \text{ et } \mathbb{I} \otimes X_2 \otimes X_3.$$

La mesure donne les syndromes :

+1	et	+1	→	pas d'erreur
-1	et	+1	→	erreur dans le 1 ^{er} bit
-1	et	-1	→	erreur dans le 2 ^{ème} bit
+1	et	-1	→	erreur dans le 3 ^{ème} bit.

Correction d'erreur :

Appliquer les opérateurs unitaire :

pas d'erreur	→	$\mathbb{I} \otimes \mathbb{I} \otimes \mathbb{I}$
1 ^{er} bit	→	$Z_1 \otimes \mathbb{I} \otimes \mathbb{I}$
2 ^{ème} bit	→	$\mathbb{I} \otimes Z_2 \otimes \mathbb{I}$
3 ^{ème} bit	→	$\mathbb{I} \otimes \mathbb{I} \otimes Z_3$.

Par exemple si on transmet $|0\rangle$, on code par $|+++ \rangle$ et si le canal produit $|++-\rangle$ on mesurera $(+1, -1)$ ce qui indique que le 3^{ème} bit a subi un phase-flip. On applique

$$\mathbb{I} \otimes \mathbb{I} \otimes Z_3 |++-\rangle = |+++ \rangle \rightarrow |0\rangle.$$

10.3 Le code de Shor

La situation du paragraphe précédent n'est pas encore satisfaisante car en MQ les deux types d'erreurs peuvent essentiellement survenir en même temps et nous aimerions un procédé qui les corrige toutes. La solution de ce problème a été présentée pour la première fois par Shor et elle se révèle être à la base des autres constructions plus élaborées qui ont suivies.

Si nous pouvons corriger à la fois les bits-flips et les phases-flips, alors nous pouvons corriger du même coup une très large classe d'erreurs à 1 qubit. En effet nous pourrions corriger toutes les erreurs produites par des canaux du type

$$\mathcal{E}(\varrho) = p_0\varrho + p_1X\varrho X + p_2Y\varrho Y + p_3Z\varrho Z$$

avec $p_0 + p_1 + p_2 + p_3 = 1$. Cette classe contient les canaux blit-flip, phase-flip, bit-phase-flip, dépolarisant, etc... Notons que cela n'inclut pas des erreurs sur l'amplitude des coefficients α et β .

L'idée principale du code de Shor est de "concaténer" les deux codes de répétition. La concaténation est en fait une méthode de la théorie classique du codage. Elle permet de construire des nouveaux codes en assemblant des codes élémentaires.

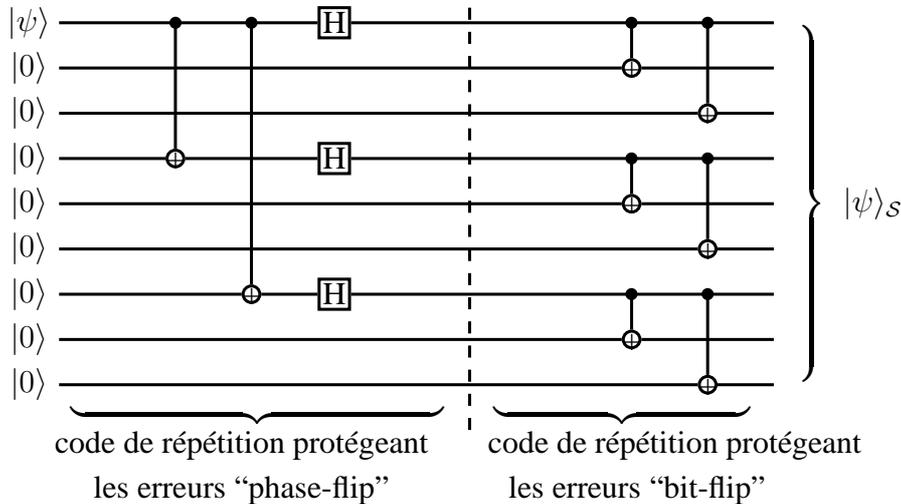
Les mots de code sont construits comme suit :

$$\begin{aligned}
 |0\rangle \rightarrow |+\rangle &\longrightarrow |+++ \rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle + |1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle + |1\rangle}{\sqrt{2}} \\
 &\longrightarrow \frac{|000\rangle + |111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle + |111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle + |111\rangle}{\sqrt{2}} \equiv |0_S\rangle. \\
 |1\rangle \rightarrow |-\rangle &\longrightarrow |--- \rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} \otimes \frac{|0\rangle - |1\rangle}{\sqrt{2}} \\
 &\longrightarrow \frac{|000\rangle - |111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle - |111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle - |111\rangle}{\sqrt{2}} \equiv |1_S\rangle.
 \end{aligned}$$

Ainsi chaque qubit est encodé par un état à 9 qubits. Le “rendement” de ce code est 1/9. Si nous appelons $|0\rangle_S$ et $|1\rangle_S$ les deux états à 9 qubits, un mot de code général est

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle \rightarrow |\psi_S\rangle = \alpha |0\rangle_S + \beta |1\rangle_S.$$

exercice : Montrez que le circuit suivant réalise l’opération de codage de façon unitaire :



Montrons maintenant que ce code est capable de protéger l’état contre n’importe quelle erreur sur 1 qubit engendrée par les opérateurs \mathbb{I} (pas d’erreur), X (bit-flip), Z (phase-flip) et Y (bit & phase flip). En d’autres termes nous voulons montrer que si la sortie du canal est $|\psi_S\rangle$, $X_1|\psi_S\rangle$, $Y_1|\psi_S\rangle$, $Z_1|\psi_S\rangle$ agit sur un seul des 9 qubits, il est possible de détecter l’erreur et de reconstruire $|\psi\rangle_S$ (et donc $|\psi\rangle$) grâce à des opérations permises en MQ.

Erreurs de type bit-flip. Considérons d'abord l'action de X tout seul sur un des trois premiers qubits (bit-flip). Comme avant, la mesure simultanée des observables Z_1Z_2 et Z_2Z_3 permet de décider si un des trois premiers qubit est renversé ou non. Ensuite il est facile de le corriger en appliquant X_i (si c'est $i = 1, 2, 3$ qui est renversé). Pour pouvoir traiter tous les qubits ce cette façon il faut faire une mesure des opérateurs suivants

$$Z_1Z_2 \text{ et } Z_2Z_3; \quad Z_4Z_5 \text{ et } Z_5Z_6; \quad Z_7Z_8 \text{ et } Z_8Z_9.$$

Tous ces opérateurs commutent et la mesure simultanée est possible.

Erreurs de type phase-flip. Considérons maintenant l'action de Z sur un seul des trois premiers qubits (phase-flip). Un phase-flip sur le deuxième qubit change les états de base en (l'état à la sortie du canal) $Z_2|\psi\rangle_S$,

$$\frac{|000\rangle - |111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle + |111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle + |111\rangle}{\sqrt{2}}$$

et

$$\frac{|000\rangle + |111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle - |111\rangle}{\sqrt{2}} \otimes \frac{|000\rangle - |111\rangle}{\sqrt{2}}$$

Cette erreur peut être détectée (sans détruire ces états) grâce à une mesure des (opérateurs qui commutent entre eux)

$$(X_1X_2X_3)(X_4X_5X_6) \quad \text{et} \quad (X_4X_5X_6)(X_7X_8X_9)$$

Par exemple pour l'erreur ci-dessus

$$(X_1X_2X_3)(X_4X_5X_6)(Z_2|\psi_S\rangle) = -(Z_2|\psi_S\rangle)$$

et

$$(X_4X_5X_6)(X_7X_8X_9)(Z_2|\psi_S\rangle) = +(Z_2|\psi_S\rangle)$$

On conclut que l'erreur phase-flip est dans un des trois premiers qubits. Notez que l'on ne peut pas affirmer lequel des trois est erroné. Mais cela ne nous empêche pas d'effectuer la correction en appliquant l'opérateur $Z_1Z_2Z_3$ (vérifiez !)

$$Z_1Z_2Z_3(Z_2|\psi_S\rangle) = |\psi_S\rangle.$$

Erreurs de type bit-phase flip. Illustrons maintenant la correction d'une erreur bit & phase-flip sur le 4^{ième} bit. L'état sortant du canal est

$$X_4Z_4|\psi_S\rangle.$$

D'abord on détecte les bit-flip comme suit :

$$\begin{aligned}
Z_1 Z_2 (X_4 Z_4 |\psi_S\rangle) &= X_4 Z_4 |\psi_S\rangle \\
Z_2 Z_3 (X_4 Z_4 |\psi_S\rangle) &= X_4 Z_4 |\psi_S\rangle \\
Z_4 Z_5 (X_4 Z_4 |\psi_S\rangle) &= -(X_4 Z_4 |\psi_S\rangle) \quad (\text{car } Z_4 X_4 = -X_4 Z_4) \\
Z_5 Z_6 (X_4 Z_4 |\psi_S\rangle) &= X_4 Z_4 |\psi_S\rangle \\
Z_7 Z_8 (X_4 Z_4 |\psi_S\rangle) &= X_4 Z_4 |\psi_S\rangle \\
Z_8 Z_9 (X_4 Z_4 |\psi_S\rangle) &= X_4 Z_4 |\psi_S\rangle
\end{aligned}$$

On conclut qu'il y a un bit-flip sur le 4^{ième} bit. Ensuite on détecte les phase-flips comme suit

$$\begin{aligned}
X_1 X_2 X_3 X_4 X_5 X_6 (X_4 Z_4 |\psi_S\rangle) &= -X_4 Z_4 |\psi_S\rangle \\
&\quad (\text{car } X_4 \text{ et } Z_4 \text{ anti-commutent}) \\
X_4 X_5 X_6 X_7 X_8 X_9 (X_4 Z_4 |\psi_S\rangle) &= -X_4 Z_4 |\psi_S\rangle
\end{aligned}$$

et on conclut qu'il y a aussi un phase-flip sur le quatrième qubit. L'état est corrigé en appliquant $(X_4 Z_4)$ sur la sortie du canal : $X_4 Z_4 (X_4 Z_4) |\psi_S\rangle = |\psi_S\rangle$.

Correction d'erreurs générales à 1 qubit Les résultats obtenus jusqu'ici peuvent être obtenus aussi d'une façon plus unifiée qui montre leur généralité. Considérons la matrice densité décrivant la sortie d'un canal quelconque :

$$|\psi_S\rangle\langle\psi_S| \rightarrow \mathcal{E}(|\psi_S\rangle\langle\psi_S|)$$

Une mesure des observables $Z_1 Z_2, Z_2 Z_3, Z_4 Z_5, Z_5 Z_6, Z_7 Z_8, Z_8 Z_9$ et $X_1 X_2 X_3 X_4 X_5 X_6, X_4 X_5 X_6 X_7 X_8 X_9$ va projeter l'état $\mathcal{E}(|\psi_S\rangle\langle\psi_S|)$ sur un des sous-espaces propres de ces observables. Les valeurs propres sont toutes égales à ± 1 . Si le bruit est faible et agit essentiellement sur un qubit au plus, le syndrome sera avec une grande probabilité du type discuté dans les paragraphes précédents, c'est à dire correspondant à un bit-flip, phase-flip ou bit & phase flip. On peut l'identifier puis corriger l'erreur. Il existe une faible probabilité (si le bruit est faible) que l'erreur affecte plus qu'un qubit (comme dans le cas classique) et alors la correction d'erreur peut être erronée.

Pour résumer, le code de Shor est de longueur 9, sa dimension est 2, et il corrige 1 erreur. Plus précisément les mots de code sont de type $\alpha |0\rangle_S + \beta |1\rangle_S$, donc vivent dans un sous-espace de Hilbert de dimension 2^1 , de l'espace à 9 qu-bits, qui est lui est de dimension 2^9 . On dit que les paramètres de ce code quantique sont $[9, 1, 1]$ (longueur 9 et $\dim \mathcal{H} = 2^9$; $\dim \mathcal{C} = 2^1$; corrige une erreur).

10.4 Formalisme des stabilisateurs

Ce paragraphe peut être sauté en première lecture.

La construction du code de Shor permet de dégager une structure algébrique qui est à la base de la construction de codes correcteurs d'erreurs plus généraux. Nous avons vu dans le paragraphe précédent que :

1. Les états (mots) $|\Psi\rangle$ du code de Shor sont les états propres associés à la valeur propre égale 1 des observables $Z_1Z_2, Z_2Z_3, Z_4Z_5, Z_5Z_6, Z_7Z_8, Z_8Z_9$ et $X_1X_2X_3, X_4X_5X_6, X_4X_5X_6, X_7X_8X_9$. Ces opérateurs sont les *stabilisateurs* du code.
2. Ces opérateurs commutent entre eux, ce qui permet d'en faire une mesure simultanée.
3. Si $|\Psi\rangle \rightarrow |\Psi'\rangle$ est corrompu par un bit, phase ou bit-phase flip, le nouvel état est encore un vecteur propre des opérateurs stabilisateurs¹.
4. Le syndrome est donné par la liste des v.p des opérateurs stabilisateurs dans l'état $|\Psi'\rangle$.

Remarque : Les stabilisateurs sont les analogues des lignes de la matrice de parité. Chaque équation aux valeurs propres satisfaite par un état du code est l'analogie d'une contrainte de parité. Les valeurs propres des stabilisateurs sont l'analogie du syndrome. Notons qu'il y a en quelque sorte deux types de contraintes de parité : une contrainte de type "z" et une contrainte de type "x". Il existe aussi deux "parties" dans le syndrome : l'une ayant trait au bit-flip et l'autre au phase-flip.

Pour généraliser cette construction il convient d'introduire le groupe de Pauli \mathcal{P}^n . Ce groupe est formé par l'ensemble de tous les produits (tensoriels) des opérateurs de Pauli X_j, Y_j, Z_j et $1, -1, i, -i$ pour $j = 1, \dots, n$. Il est facile de voir que deux éléments de ce groupe commutent ou anti-commutent. Un sous-groupe $\mathcal{S} \subset \mathcal{P}^n$ dont tous les éléments commutent est appelé groupe stabilisateur. On peut toujours trouver un ensemble de générateurs indépendants pour $\mathcal{S} : \{O_1, \dots, O_m\}$. Le sous-espace de Hilbert $\mathcal{C} \subset (\mathbb{C}^2)^{\otimes n}$ des états $|\Psi\rangle$ tels que

$$O_j|\Psi\rangle = 1 \cdot |\Psi\rangle, \quad j = 1, \dots, m$$

est un code de longueur n et dimension 2^{n-m} . Notez que $|\Psi\rangle$ est "stable" sous l'action de \mathcal{S} . Les observables $\{O_1, \dots, O_m\}$ sont simultanément mesurables puisqu'elles commutent entre elles. Toute erreur (multiple) de type bit, phase,

1. Si il est corrompu par une combinaison linéaire de ces erreurs alors la mesure des observables projète sur un état propre, et nous ramène à un de ces types d'erreurs discrètes.

bit-phase ou combinaison linéaires de telles erreurs suivie par une mesure donne un état corrompu du type

$$E_k|\Psi\rangle = |\Psi'\rangle$$

où E_k est un élément de \mathcal{P}^n . Cet état est encore vecteur propre des stabilisateurs, et donc une mesure ne va pas le détruire. De plus le résultat de la mesure donne les valeurs propres $\lambda_1 = \pm 1, \dots, \lambda_m = \pm 1$. Cette liste de v.p est le "syndrome". Si "l'erreur" E_k anti-commute avec un générateur O_j alors on a

$$O_j|\Psi'\rangle = O_jE_k|\Psi\rangle = -E_kO_j|\Psi\rangle = -E_k|\Psi\rangle = -|\Psi'\rangle$$

c.a.d $\lambda_j = -1$. Pour un code donné, on peut établir un table de "correction" avec les syndromes associés à chaque type d'erreur. Une fois l'erreur détectée et distinguée, la correction d'erreur se fait par (opération unitaire)

$$|\Psi'\rangle \rightarrow E_k|\Psi'\rangle = E_k^2|\Psi\rangle = |\Psi\rangle$$

Notons que dans les produits d'opérateurs de Pauli il est toujours possible d'éliminer Y_j au profit de produits X_jZ_j . Ensuite on peut toujours écrire un opérateur de Pauli comme un produit des Z multiplié par un produit des X . Par exemple

$$X_1I_2Y_3Z_4Y_5I_6 = -(X_1I_2X_3I_4X_5I_6)(I_1I_2Z_3Z_4Z_5I_6)$$

On peut "coder" les produits à droite par la "ligne"

$$(101010 \mid 001110)$$

En appliquant ce procédé à tous les stabilisateurs O_j du code, on obtient une matrice de la forme

$$(A_1 \mid A_2)$$

appelée parfois "matrice de parité quantique". Chaque ligne de la matrice permet de reconstruire une contrainte $O_j|\Psi\rangle = |\Psi\rangle$ satisfaite par les états du code. On peut montrer que la condition de commutation des $\{O_1, \dots, O_m\}$ est équivalente a

$$A_1A_2^T + A_2A_1^T = 0$$

(les lignes doivent être orthogonales par rapport à un produit scalaire symplectique). Un opérateur d'erreur E peut s'écrire de façon similaire comme un "vecteur d'erreur" $(e_1 \mid e_2)$. Le syndrome devient

$$A_1\vec{e}_1 + A_2\vec{e}_2 = \vec{s}$$

La correction d'erreur quantique est alors ramenée à la correction d'erreur classique (c.a.d à la résolution d'un système d'équations linéaires). Ce dernier point est bienvenu dans la mesure où la théorie classique du codage est très bien développée.

Dans le cas du code Shor on a une situation un peu particulière :

$$(A_1 | A_2) = \left[\begin{array}{c|c} 110000000 & 000000000 \\ 011000000 & 000000000 \\ 000110000 & 000000000 \\ 000011000 & 000000000 \\ 000000110 & 000000000 \\ 000000011 & 000000000 \\ 000000000 & 111111000 \\ 000000000 & 000111111 \end{array} \right]$$

Les codes dont la matrice de parité quantique possède une structure par bloc s'appellent des codes de Calderbank-Shor-Steane. En général

$$(A_1 | A_2) = \left[\begin{array}{c|c} H_1 & 0 \\ 0 & G_2^T \end{array} \right]$$

avec la contrainte d'orthogonalité

$$H_1 G_2 = 0$$

On peut interpréter H_1 comme la matrice de parité d'un code classique \mathcal{C}_1 , G_2 la matrice génératrice d'un code classique \mathcal{C}_2 . Ici, G_2^T est la matrice de parité² de \mathcal{C}_2^\perp . La contrainte d'orthogonalité signifie que $\mathcal{C}_2 \subset \mathcal{C}_1$. Pour le décodage on doit résoudre des équations du type $H_1 \vec{e}_1 = \vec{s}_1$ et $G_2^T \vec{e}_2 = \vec{s}_2$ (ou bien $H_2^\perp = 0$).

Nous étudions cette construction plus en détail dans la section suivante. Il deviendra clair que le code \mathcal{C}_1 protège contre les erreurs de type bit-flip et \mathcal{C}_2^\perp protège contre les erreurs de type phase-flip. L'utilisation simultanée des deux codes permet de protéger contre toutes combinaison de ces erreurs.

10.5 Codes de Calderbank-Shor-Steane

Ce paragraphe peut être lu indépendememnt du précédent.

Ces codes sont une classe particulière de codes de type stabilisateurs, dont

2. Cette matrice est appelée H_2^\perp dans la prochaine section.

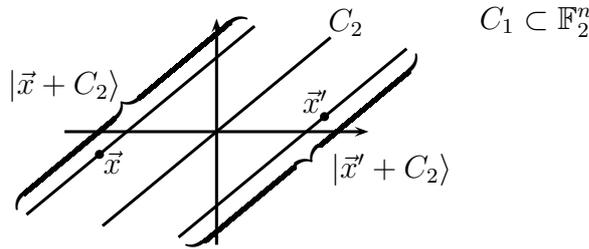
les propriétés principales sont aisées à analyser. Cette classe de codes utilise deux codes classiques : l'un pour corriger les bit-flips et l'autre pour corriger les phases-flips.

Soit C_1 et C_2 deux codes linéaires classiques tels que $C_2 \subset C_1$ de paramètres (n, k_1) et (n, k_2) . Nous rappelons que C_1 est un sous-espace vectoriel de \mathbb{F}_2^n de dimension k_1 et C_2 un sous-espace vectoriel de C_1 de dimension $k_2 \leq k_1$. On suppose aussi que C_1 et C_2^\perp corrigent t erreurs (par exemple leur distance minimale est $d = 2t + 1$). Le code dual C_2^\perp est le sous-espace vectoriel orthogonal à C_2 dans \mathbb{F}_2^n et sa dimension est donc $n - k_2$.

En fait C_2 est un sous-groupe de C_1 et on peut considérer l'ensemble des classes d'équivalences C_1/C_2 . Soit \vec{x} un mot de C_1 , la classe d'équivalence de \vec{x} est l'ensemble des mots de la forme $\{\vec{x} + \vec{y}; \vec{y} \in C_2\}$. C'est donc l'hyperplan parallèle à C_2 passant par $\vec{x} \in C_1$. Posons

$$|\vec{x} + C_2\rangle = \frac{1}{\sqrt{|C_2|}} \sum_{\vec{y} \in C_2} |\vec{x} + \vec{y}\rangle$$

Ce ket ne dépend que de la classe d'équivalence associée à \vec{x} , et non pas du représentant spécial choisi.



Le ket $|\vec{x} + C_2\rangle$ est la superposition cohérente de tous les états dans la classe d'équivalence $\vec{x} + C_2$. Pour deux classes d'équivalence différentes $\vec{x} + C_2$ et $\vec{x}' + C_2$ (c'est à dire que $\vec{x} - \vec{x}' \notin C_2$) les kets associés sont perpendiculaires :

$$\langle \vec{x} + C_2 | \vec{x}' + C_2 \rangle = 0$$

Il y a $\frac{|C_1|}{|C_2|} = 2^{k_1 - k_2}$ classes d'équivalence et donc $2^{k_1 - k_2}$ vecteurs orthogonaux.

Le code $\text{CSS}(C_1, C_2)$ est le sous-espace d'Hilbert engendré par ces $2^{k_1 - k_2}$ vecteurs orthogonaux. C'est un sous-espace de l'espace d'Hilbert à n qubits $\mathbb{C}^2 \otimes \mathbb{C}^2 \otimes \dots \otimes \mathbb{C}^2$ lequel est de dimension 2^n . Les paramètres du code sont $[n, k_1 - k_2]$; où $\dim \mathcal{H} = n$ et $\dim \text{CSS}(C_1, C_2) = 2^{k_1 - k_2}$. Nous allons montrer que si C_1 et C_2^\perp corrigent t erreurs alors $\text{CSS}(C_1, C_2)$ corrige t bit et phase

flips.

Il suffit de montrer ces propriétés pour les vecteurs de base du code. Les mêmes arguments entraînent qu'elles sont vraies pour tout état de superposition du code.

Soit \vec{e}_1 et \vec{e}_2 les vecteurs possédants des 1 et 0 avec la coordonnée 1 aux t endroits des bit et/ou phase flips. Le vecteur corrompu par le bruit est :

$$|\psi\rangle = \frac{1}{\sqrt{|C_2|}} \sum_{\vec{y} \in C_2} (-1)^{(\vec{x}+\vec{y}) \cdot \vec{e}_2} |\vec{x} + \vec{y} + \vec{e}_1\rangle$$

Pour effectuer la correction d'erreur il faut faire des opérations permises par la MQ, et qui reconstituent $|\vec{x} + C_2\rangle$. Comme il faudra calculer et stocker les "syndromes" du bit et phase flip on ajoute des bits auxiliaires de stockage et on travaille avec l'état

$$|\psi\rangle \otimes |0_n\rangle \otimes |0_n\rangle = \frac{1}{\sqrt{|C_2|}} \sum_{\vec{y} \in C_2} (-1)^{(\vec{x}+\vec{y}) \cdot \vec{e}_2} |\vec{x} + \vec{y} + \vec{e}_1\rangle \otimes |0_n\rangle \otimes |0_n\rangle$$

Correction du bit-flip

On calcule les syndromes $H_1(\vec{x} + \vec{y} + \vec{e}_1) = H_1\vec{e}_1$ (grâce à la matrice de parité du code C_1). Ce calcul peut être implémenté de façon unitaire :

$$U_1|\vec{x} + \vec{y} + \vec{e}_1\rangle \otimes |0_n\rangle \otimes |0_n\rangle = |\vec{x} + \vec{y} + \vec{e}_1\rangle \otimes |H_1\vec{e}_1\rangle \otimes |0_n\rangle$$

si bien que

$$U_1|\psi\rangle \otimes |0_n\rangle \otimes |0_n\rangle = \frac{1}{\sqrt{|C_2|}} \sum_{\vec{y} \in C_2} (-1)^{(\vec{x}+\vec{y}) \cdot \vec{e}_2} |\vec{x} + \vec{y} + \vec{e}_1\rangle \otimes |H_1\vec{e}_1\rangle \otimes |0_n\rangle$$

Une mesure du second registre dans la base computationnelle donne $|H_1\vec{e}_1\rangle$ avec probabilité 1 (car le ket $|H_1\vec{e}_1\rangle$ peut être factorisé dans la dernière expression, et c'est un vecteur de la base computationnelle). Donc on connaît le syndrome $H_1\vec{e}_1$ et si $|\vec{e}_1| \leq t$ on peut corriger les erreurs grâce à un algorithme classique (pour le code C_1). Cela permet d'appliquer l'opérateur unitaire

$$\prod_{\text{composantes non nulles de } \vec{e}_1} X_i \equiv U_{1,\text{corr}}$$

pour corriger l'état quantique. On obtient

$$U_{1,\text{corr}}U_1|\psi\rangle \otimes |0_n\rangle \otimes |0_n\rangle = \frac{1}{\sqrt{|C_2|}} \sum_{\vec{y} \in C_2} (-1)^{(\vec{x}+\vec{y}) \cdot \vec{e}_2} |\vec{x} + \vec{y}\rangle \otimes |H_1\vec{e}_1\rangle \otimes |0_n\rangle$$

Correction du phase-flip

D'abord, pour passer dans une base plus naturelle, on applique les portes de Hadamard $H^{\otimes n}$ ce qui donne

$$\begin{aligned}
& H^{\otimes n} U_{1,\text{corr}} U_1 |\psi\rangle \otimes |0_n\rangle \otimes |0_n\rangle \\
&= \frac{1}{2^{n/2}} \frac{1}{\sqrt{|C_2|}} \sum_{\vec{y} \in C_2} (-1)^{(\vec{x}+\vec{y}) \cdot \vec{e}_2} \sum_{\vec{z}} (-1)^{\vec{z} \cdot (\vec{x}+\vec{y})} |\vec{z}\rangle \otimes |H_1 \vec{e}_1\rangle \otimes |0_n\rangle \\
&= \frac{1}{2^{n/2}} \frac{1}{\sqrt{|C_2|}} \sum_{\vec{z}} \sum_{\vec{y} \in C_2} (-1)^{(\vec{z}+\vec{e}_2) \cdot (\vec{x}+\vec{y})} |\vec{z}\rangle \otimes |H_1 \vec{e}_1\rangle \otimes |0_n\rangle \\
&= \frac{1}{2^{n/2}} \frac{1}{\sqrt{|C_2|}} \sum_{\vec{z}} \sum_{\vec{y} \in C_2} (-1)^{\vec{z} \cdot (\vec{x}+\vec{y})} |\vec{z} + \vec{e}_2\rangle \otimes |H_1 \vec{e}_1\rangle \otimes |0_n\rangle
\end{aligned}$$

On note que

$$\sum_{\vec{y} \in C_2} (-1)^{\vec{z} \cdot \vec{y}} = \begin{cases} |C_2| & \text{si } \vec{z} \in C_2^\perp, \\ 0 & \text{sinon} \end{cases}$$

ce qui donne l'état :

$$= \frac{1}{|C_2^\perp|} \sum_{\vec{z} \in C_2^\perp} (-1)^{\vec{z} \cdot \vec{x}} |\vec{z} + \vec{e}_2\rangle \otimes |H_1 \vec{e}_1\rangle \otimes |0_n\rangle.$$

On implémente maintenant le calcul du syndrome de C_2^\perp de façon unitaire en stockant le résultat dans le dernier registre

$$\begin{aligned}
& U_2 |\vec{z} + \vec{e}_2\rangle \otimes |H_1 \vec{e}_1\rangle \otimes |0_n\rangle \\
&= |\vec{z} + \vec{e}_2\rangle \otimes |H_1 \vec{e}_1\rangle \otimes |H_2^\perp (\vec{z} + \vec{e}_2)\rangle \\
&= |\vec{z} + \vec{e}_2\rangle \otimes |H_1 \vec{e}_1\rangle \otimes |H_2^\perp \vec{e}_2\rangle.
\end{aligned}$$

où ici H_2^\perp est la matrice de parité de C_2^\perp . L'état obtenu est :

$$\begin{aligned}
& U_2 H^{\otimes n} \tilde{U}_{1,\text{corr}} U_1 |\psi\rangle \otimes |0_n\rangle \otimes |0_n\rangle \\
&= \frac{1}{\sqrt{|C_2^\perp|}} \sum_{\vec{z} \in C_2^\perp} (-1)^{\vec{z} \cdot \vec{x}} |\vec{z} + \vec{e}_2\rangle \otimes |H_1 \vec{e}_1\rangle \otimes |H_2^\perp \vec{e}_2\rangle.
\end{aligned}$$

La mesure du dernier registre (dans la base computationnelle) projetée sur $|H_2^\perp \vec{e}_2\rangle$ lui-même, avec probabilité 1. Donc $H_2^\perp \vec{e}_2$ est connu et \vec{e}_2 peut être calculé si C_2^\perp corrige t erreurs (on suppose $|\vec{e}_2| \leq t$). Cela permet alors d'appliquer l'opérateur unitaire

$$U_{2,\text{corr}} \equiv \prod_{\text{comp non nulles de } \vec{e}_2} X_i$$

pour trouver l'état :

$$\begin{aligned} & U_{2,\text{corr}} U_2 H^{\otimes n} \tilde{U}_1 U_1 |\psi\rangle \otimes |0_n\rangle \otimes |0_n\rangle \\ &= \frac{1}{\sqrt{|C_2^\perp|}} \sum_{\vec{z} \in C_2^\perp} (-1)^{\vec{z} \cdot \vec{x}} |\vec{z}\rangle \otimes |H_1 \vec{e}_1\rangle \otimes |H_2^\perp \vec{e}_2\rangle. \end{aligned}$$

La dernière étape consiste maintenant à appliquer une fois de plus $H^{\otimes n}$ pour revenir dans la base initiale. Cela donne

$$\begin{aligned} & H^{\otimes n} U_{2,\text{corr}} U_2 H^{\otimes n} U_{1,\text{corr}} U_1 |\psi\rangle \otimes |0_n\rangle \otimes |0_n\rangle \\ &= \frac{1}{\sqrt{|C_2^\perp|}} \frac{1}{2^{n/2}} \sum_{\vec{z} \in C_2^\perp} \sum_{\vec{y}} (-1)^{\vec{z} \cdot (\vec{x} + \vec{y})} |\vec{y}\rangle \otimes |H_1 \vec{e}_1\rangle \otimes |H_2^\perp \vec{e}_2\rangle \\ &= \frac{1}{\sqrt{|C_2^\perp|}} \frac{1}{2^{n/2}} \sum_{\vec{z} \in C_2^\perp} \sum_{\vec{y}} (-1)^{\vec{z} \cdot \vec{y}} |\vec{y} + \vec{x}\rangle \otimes |H_1 \vec{e}_1\rangle \otimes |H_2^\perp \vec{e}_2\rangle. \end{aligned}$$

Maintenant on note (comme avant) :

$$\sum_{\vec{z} \in C_2^\perp} (-1)^{\vec{z} \cdot \vec{y}} = \begin{cases} |C_2^\perp| & \text{si } \vec{y} \in (C_2^\perp)^\perp = C_2, \\ 0 & \text{sinon.} \end{cases}$$

L'état final obtenu est donc :

$$\left(\frac{1}{\sqrt{|C_2|}} \sum_{\vec{y} \in C_2} |\vec{x} + \vec{y}\rangle \right) \otimes |H_1 \vec{e}_1\rangle \otimes |H_2^\perp \vec{e}_2\rangle.$$

Nous voyons que le mot de code initial a été reconstruit dans le premier registre !

Exemple : Le code de Steane

Le code de Steane est un code $\text{CSS}(C_1, C_2)$ avec $C_1 = \text{Hamming}(7, 4)$ et $C_2 = C_1^\perp$. On peut facilement montrer (exercice) que $C_1^\perp \subset C_1$; le code de Hamming contient son propre dual.

Le code $\text{CSS}(C_1, C_2)$ possède 7 qubits, est de dimension 2 et corrige une erreur. Les mots de codes sont (voir exercices)

$$\begin{aligned} |0\rangle_{\text{Steane}} = \frac{1}{\sqrt{8}} \{ & |0000000\rangle + |1001101\rangle + |0101011\rangle + |0010111\rangle \\ & + |0111100\rangle + |1011010\rangle + |1100110\rangle + |1110001\rangle \} \end{aligned}$$

et

$$\begin{aligned} |1\rangle_{\text{Steane}} = \frac{1}{\sqrt{8}} \{ & |1111111\rangle + |0110010\rangle + |1010100\rangle + |1101000\rangle \\ & + |1000011\rangle + |0100101\rangle + |0011001\rangle + |0001110\rangle \}. \end{aligned}$$

Le mot $|0\rangle_{\text{Steane}}$ correspond à la classe d'équivalence de $(0000000) \in C_1$ et le deuxième mot $|1\rangle_{\text{Steane}}$ correspond à la classe d'équivalence de $(1111111) \in C_1$.