République algérienne démocratique et populaire

Ministère de l'enseignement supérieur et de la recherche scientifique

Université Blida 1

Institut d'aéronautique et des études spatiales

Département des études spatiales

Présenté pour l'obtention du diplôme de Master

Mémoire de fin des études

En : Aéronautique

Option : Télécommunications spatiales

Thème

Etude, simulation et analyse comparative des performances de codage turbo du canal pour les communications spatiales

Réalisé par :

Mehtougui Nawel

Dirigé par :

Mme Azine Houria

Mr Iftene Essedik

Promotion 2019

Mémoire

Thème de stage :

Etude, simulation et analyse comparative des performances de codage turbo du canal pour les communications spatiales

Table des matières

Chapitre 3 : Simulation et résultats

Table des figures

Table des figures

Table des figures

Liste des tableaux

Liste des Tableaux

Liste des symboles et abréviations

Liste des symboles

K	Longueur de contrainte d'un code convolutionnel
m	Mémoire d'un code convolutionnel
R	Taux de codage d'un code convolutionnel
n	Taille d'une séquence d'information binaire
d_{free}	Distance libre d'un code
E_b/N_0	Rapport signal sur bruit
P(B)	Probabilité d'erreur par bit
σ^2	Variance du bruit dans le canal
R_t	Taux de codage pour des codeurs montés parallèle
R_1	Taux de codage de code convolutionnel 1
R_2	Taux de codage de code convolutionnel 2
x_k^s	Symbole systématique à la sortie de l'encodeur turbo
x_k^{1p}	Premier symbole de parité à la sortie de l'encodeur turbo
x_k^{2p}	Deuxième symbole de parité à la sortie de l'encodeur turbo
$S_0(t)$	Signal BPSK lorsqu'on transmet un 0
$S_1(t)$	Signal BPSK lorsqu'on transmet un 1
f_p	Fréquence porteuse
T_b	Durée de transmission d'un bit
r_k^s	Symbole d'information systématique reçu à l'instant k
r_k^{1p}	Symbole de parité 1 reçu à l'instant k
r_k^{2p}	Symbole de parité 2 reçu à l'instant k
d_k	Bit d'information
d_{min}	Distance minimal
S_j	Etats internes possible du codeur
y	Bit d'entrée
x_0x_1	Séquence en sortie

Liste des abréviations

BBGA	Bruit Blanc Gaussien et Additif
BPSK	Binary Phase Shift Keying
ASK	Amplitude Shift Keying
FSK	Frequency Shift Keying
PSK	Phase Shift Keying
QAM	Quadrature Amplitude Modulation
BFSK	Binary Frequency Shift Keying
CRS	Code récursif et systématique
LLR	Logarithme du rapport de vraisemblance
APP	Probabilité à Posteriori
ML	Maximum de vraisemblance
MAP	Maximum A Posteriori
TEB	Taux d'Erreur Binaire
DEC1	Décodeur 1
DEC2	Décodeur 2
SNR	Signal to Noise Ratio
JD	Probabilité conjointe
FSM	Métrique d'état en avant
BSM	Métrique d'état en arrière
BM	Métrique de branche

Introduction générale

Dans tout système de communication numérique, on cherche à transmettre l'information provenant d'une source vers un récepteur à travers un canal de transmission.

La transmission numérique est très intéressant dans le sens où il permet l'utilisation de nombreuses techniques de manipulation de l'information [1]. Parmi celles-ci, nous pouvons penser à la compression de l'information, au cryptage de cette dernière et aussi aux codes correcteurs d'erreurs. Ce sont ces derniers qui font l'objet de beaucoup de recherches actuellement. Les perturbations intervenant sur le canal de transmission induisent des erreurs de transmission que le codage de canal s'efforce de combattre. L'objectif est alors d'assurer un taux d'erreur minimal. Le codage de canal est basé sur l'insertion parmi les éléments d'information d'éléments supplémentaires (la redondance) qui suivent une loi connue.

C'est le domaine spatial qui a permis le développement de ces codes correcteurs d'erreur. En effet, les premières utilisations de ceux-ci en communication devaient permettre de surmonter la perte de puissance lors de la propagation entre le transmetteur de l'engin spatial et le récepteur de la station terrestre [2]. Les engins envoyés dans l'espace, à l'origine se devaient d'être assez petits pour des raisons économiques, mais aussi des raisons d'énergie. Par conséquent, emporter une grande antenne, ou de grosses batteries, ou encore de grands panneaux solaires dans ces appareils était hors de question. Néanmoins, le bruit est toujours le même, donc les erreurs pouvaient facilement détériorer le signal. L'idée des codes correcteurs d'erreur est d'apporter un gain en puissance. En général, nous parlons de rapport signal sur bruit. Le gain de codage est la différence entre un rapport signal sur bruit requis pour atteindre une certaine performance d'erreur par bit (TEB) avec un système codé et requis pour atteindre la même performance d'erreur avec un système non codé.

Les codes convolutionnels sont parmi les plus utilisés dans les systèmes de communications en raison de leur propriété de continuité dans la transmission de l'information.

Les performances des codes turbo sont fonctions, non seulement des codes convolutionnels utilisés, mais aussi de l'algorithme de décodage et des entrelaceurs [3]. De nombreuses recherches ont portés sur les deux derniers points.

Ce mémoire est organisé comme suit. Le premier chapitre consiste en des généralités sur les systèmes de communication numérique y compris la modélisation d'un canal. Nous nous intéressons à ce que nous appelons la concaténation parallèle. Cette dernière est une méthode utilisée pour le codage de l'information que nous devons corriger à la réception, il s'agit d'utiliser deux codeurs en parallèle qui reçoivent la même séquence d'information, mais dans un ordre différent. Dans ce chapitre, nous introduisons tout d'abord les systèmes de communication. Nous définissons des différents paramètres pertinents aux transmissions numériques : la modulation BPSK et les différents types de canaux utilisés. Le codage convolutionnel qui est la base des codes turbo est aussi introduit de façon théorique. Nous en verrons deux types, à savoir le codage convolutionnel systématique et le codage récursif et systématique. A la fin de ce chapitre, la concaténation parallèle et le turbo codage des codes convolutionnels sont vue en détail, où nous présentons le principe de fonctionnement de la concaténation parallèle des codes convolutionnels séparés par des entrelaceurs.

Dans le deuxième chapitre, un des paramètres importants des codes turbo est présenté. En effet, sans l'algorithme de décodage, ces derniers ne sont plus d'excellents correcteurs d'erreur. Il existe de nombreux algorithmes de décodage et un des meilleurs pour les codes turbo s'avère être l'algorithme MAP [2]. En premier lieu, nous avons introduit le principe du maximum a posteriori qui est la base de l'algorithme MAP. En second lieu, nous avons présentés la version de l'algorithme MAP que nous avons adoptée dans notre travail. L'algorithme de décodage étant présenté, nous pouvons nous intéresser au décodage itératif. Enfin, l'entrelacement des codes turbo est introduit.

Par la suite, dans le troisième chapitre une analyse de la performance des codes turbo utilisant l'algorithme MAP a été simulée .L'étude est faite premièrement dans un canal BBGA et ensuite dans un canal de Rayleigh. Avant de conclure ce travail, nous comparons l'effet de l'utilisation de chaque type de ces deux canaux.

Chapitre I : Concept de base sur le système de communication par satellite

1.1Introduction

Le but d'un système de communication est de transmettre à distance des informations d'un émetteur à un ou plusieurs récepteurs au travers d'un canal de manière fiable.

Dans un système de transmission numérique, une suite finie de symboles représente l'information. Celle-ci est transmise sur le canal de transmission par un signal (la porteuse). Ce signal peut prendre une infinité de valeurs différentes et est ainsi soumis à différentes formes de perturbations et d'interférences, pouvant conduire à des erreurs d'interprétations du signal recueilli par le récepteur.

Le rôle des recherches en télécommunications est donc de s'assurer que le récepteur pourra recevoir le message émis par l'émetteur sans aucune erreur, par un dimensionnement judicieux du canal de transmission et par la mise en place des techniques le rendant plus robuste, autrement dit, il faut qu'il adapte le signal initial au canal envisagé, afin de transmettre l'information le plus fidèlement possible tout en optimisant l'utilisation du canal.

1.2Chaîne de transmission

Le schéma classique d'une chaîne de transmission est représenté sur la figure 1.1. En télécommunication, on utilise le modèle simplifié.

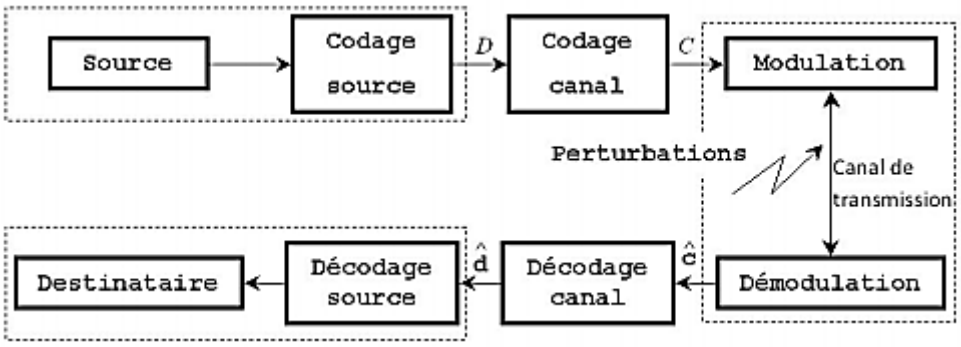

Fig. 1.1 - Diagramme d'une chaîne de transmission classique

1.2.1 La source

La source du message émet l'information sous la forme de symboles binaires.

1.2.2 Le codage source

Le codage source consiste à transformer le message de la source en une séquence d'information de façon à minimiser la taille du message en éliminant les redondances naturelles de l'information source (algorithme de compression) et à retrouver le message original à partir de la séquence de substitution (algorithme réversible). Il est à noter que les limites théoriques du codage source sont fixées par le premier théorème de Shannon [4, 5].

1.2.3 Le codage canal

Le codage canal a pour rôle de protéger l'information émise contre les perturbations du canal de transmission susceptible de modifier son contenu. Il s'agit donc de rajouter de la redondance de manière à détecter et éventuellement corriger les erreurs lors de la réception si la stratégie adoptée le permet.

L'information issue du codage source est transformée en séquence codée. Comme le décrit le théorème fondamental du codage canal, pour se rapprocher de la capacité du canal de transmission, il est nécessaire de coder l'information avant de la transmettre. Au niveau du récepteur, le décodage canal consiste dans un premier temps à détecter la présence d'erreurs dans l'information et puis dans un deuxième temps de les corriger. Les codes correcteurs d'erreurs ont été utilisés pour la détection et la correction des erreurs induites par le canal la transmission.

1.2.4 La modulation/démodulation

La modulation agit sur les paramètres d'un signal porteur afin de transmettre les données codées. Le signal porteur est une sinusoïde dont on peut faire varier l'amplitude, la fréquence ou la phase indépendamment (ASK (Amplitude Shift Keying), FSK (Frequency Shift Keying), PSK (Phase Shift Keying)) ou simultanément QAM (Quadrature Amplitude Modulation), en fonction de l'information à émettre. Le démodulateur joue le rôle inverse du modulateur et transforme donc le signal reçu en un train binaire.

1.2.4.1 Modulation BPSK

La fonction de modulation a pour objectif d'adapter le spectre de signal à émettre au canal de transmission. Dans le cas d'une modulation numérique on peut distinguer la modulation BPSK utilisée dans notre projet.

BPSK est la forme la plus simple du PSK. Elle utilise deux phases qui sont séparées de 180° ; on l'appelle également 2-PSK. Cette modulation est la plus robuste de toutes les PSK car il faut une grande déformation du signal pour que le démodulateur se trompe sur le symbole reçu. Cependant on ne peut moduler qu'un seul bit par symbole (voir la figure 1.2), ce qui est un inconvénient pour les applications qui nécessitent un débit binaire élevé.

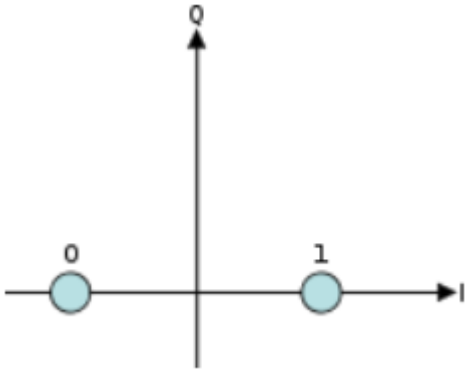

Fig. 1.2 - Diagramme de constellation pour BPSK

En BPSK, les phases opposées de la porteuse (0 et π) sont transmises toutes les secondes en considérant la durée temporelle d'un bit. Ces phases sont aussi représentées par $+1$ et -1. Ces derniers viennent des expressions $cos(w_p t)$ et $cos(w_p t + \pi)$. Ceci se représente sous forme de constellation à deux points. Chaque bit entrant dans le modulateur BPSK se retrouvera en l'un de ces deux points de cette constellation. Ceci est alors une modulation par phase.
Mathématiquement on peut exprimer un signal BPSK de la façon suivante :

$$s_0(t) = -\left(2\sqrt{\frac{E_b}{T_b}}\right)\cos 2\pi f_p t \qquad 0 \le t \le T_b$$
$$s_1(t) = +\left(2\sqrt{\frac{E_b}{T_b}}\right)\cos 2\pi f_p t \qquad 0 \le t \le T_b \tag{1.3}$$

Où :
$s_0(t)$: Signal BPSK lorsqu'on transmet un 0 ;
$s_1(t)$: Signal BPSK lorsqu'on transmet un 1 ;
T_b : Durée de transmission d'un bit ;
f_p: Fréquence porteuse ;
E_b: Energie par bit du signal transmis.

1.2.5 Le canal de transmission

Dans un système de communication, un canal de transmission s'occupe de l'acheminement de l'information d'un expéditeur (source) vers le destinataire (figure 1.1), on peut le considérer comme un milieu de propagation. Le canal est un véritable problème pour les transmissions de données.

Il existe une multitude de types de canaux. Dans le cadre de notre projet nous nous intéressons aux types de canaux suivants :

1.2.5.1 Canal avec Bruit Blanc, Gaussien et Additif (BBGA)

Le modèle du canal avec Bruit Blanc, Gaussien et Additif (BBGA) (voir figure 1.3) est le plus simple des modèles. Le signal reçu $r(t)$ est la résultante du signal $s(t)$ avec l'ajout du bruit $n(t)$ modélisé par une fonction de densité de probabilité gaussienne. Ce canal est d'écrit par l'équation suivante :

$$r(t) = s(t) + n(t) \tag{1.1}$$

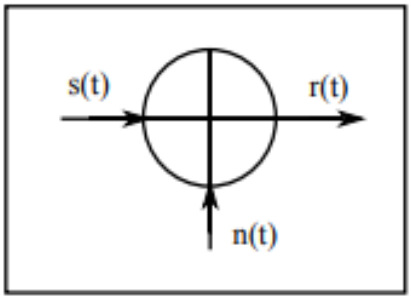

Fig. 1.3 - Modèle d'un canal BBGA [6]

1.2.5.2 Canal de Rayleigh

C'est un canal qui modélise à la fois un canal avec évanouissement qui prit en charge les évanouissements qui affectent le signal reçu et un BBGA (voir figure 1.4). Ce modèle est décrit par l'équation suivante :

$$r(t) = h(t,\tau) * s(t) + n(t) \tag{1.2}$$

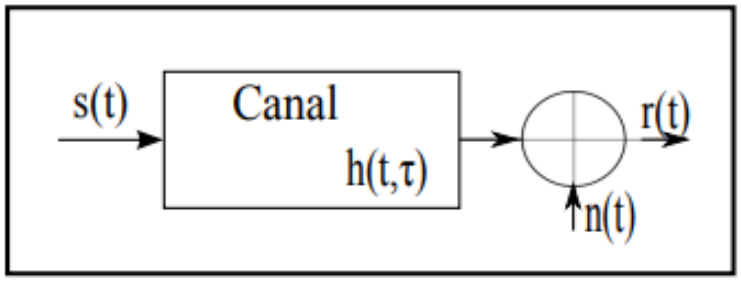

Fig. 1.4 - Modèle d'un canal de Rayleigh [6]

1.3 Types de codes

Les codes de canal appelés aussi codes correcteurs d'erreurs sont répartis en trois catégories: Les codes en blocs linéaires, les codes convolutifs et les codes concaténés ou les turbo codes.

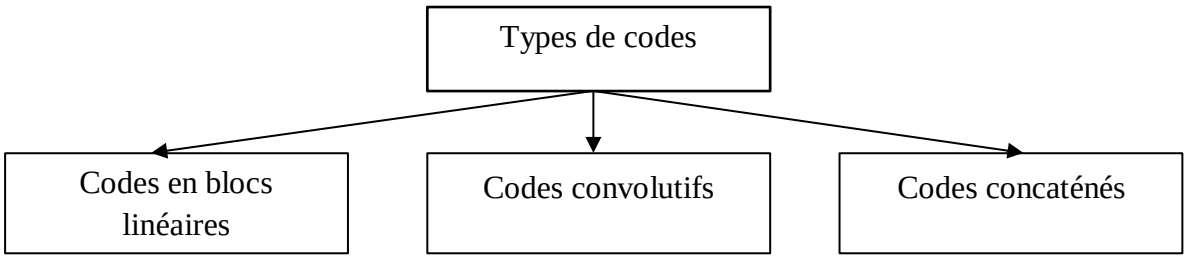

Fig. 1.5 – Type de codes correcteurs d'erreurs

1.4 Code convolutif

Les codes convolutifs forment une classe extrêmement souple et efficace de codes correcteurs d'erreur. Ce sont les codes les plus utilisés dans les communications fixes et mobiles. Les codes convolutifs ont les mêmes caractéristiques que les codes en bloc sauf qu'ils s'appliquent des cases mémoires.

1.4.1 Types de code convolutif

Il existe deux types de code convolutif : code convolutif systématique, et code convolutif récursif et systématique.

1.4.1.1 Code convolutif systématique

Un code convolutif est dit systématique si les symboles codés comportent une réplique exacte des bits d'information de l'entrée. Autrement dit, un codeur systématique de taux de codage R = 1/2, laisse passer directement un bit d'information de l'entrée à la sortie comme symbole systématique. Les autres symboles à la sortie de l'encodeur sont appelés symboles de parités.

1.4.1.2 Code convolutif récursif et systématique

Un code convolutif est dit récursif si la séquence passant dans les registres à décalages est « alimentée » par le contenu de ces registres [7]. Un exemple de codeur récursif et systématique est représenté en Figure 1.6.

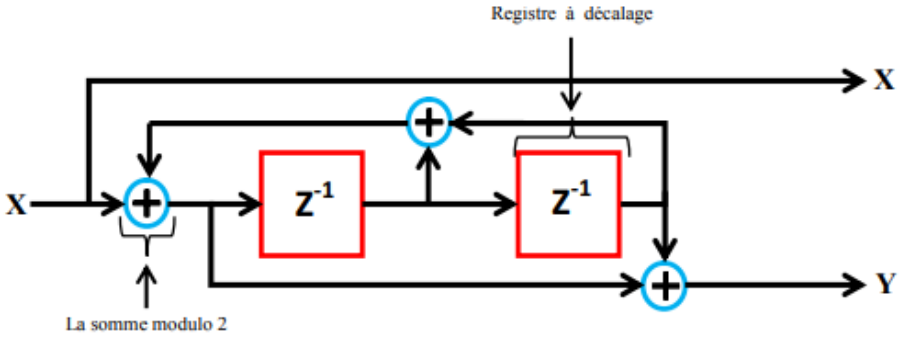

Fig. 1.6 - Exemple de codeur CRS [8]

On a constaté expérimentalement grâce aux travaux sur les codes turbo (et une équipe de chercheurs australiens s'attache à le démontrer) que seuls les codes CRS sont susceptibles d'atteindre la limite de Shannon.

1.4.2 Représentation des codes convolutifs

En pratique, un codeur convolutif peut être considéré comme une machine d'état et être représenté par un diagramme d'état, une structure en arbre ou une représentation en treillis.

a. Diagramme d'état

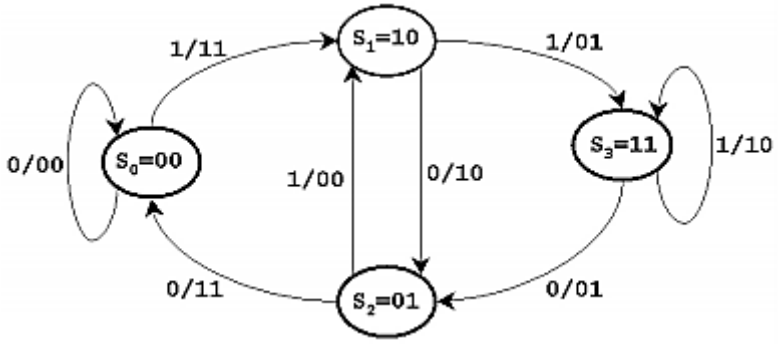

Fig. 1.7 - Exemple d'un diagramme d'état ($\boldsymbol{R = 1/2}$, $\boldsymbol{m = 2}$)

Le diagramme d'état représente les transitions possibles entre les états. Les valeurs des sorties du codeur sont indiquées sur chacune des transitions. Tous les états internes possibles du codeur sont représentés par des nœuds S_j. Pour un codeur de rendement $1/n$ possédant une mémoire de taille m, il existe 2^m états internes possibles. Chaque nœud est connecté à un autre via une branche et le passage se fait par une transition y/x_0x_1 , où y correspond au bit d'entrée et x_0x_1 représente la séquence correspondante en sortie.

b. Structure en arbre

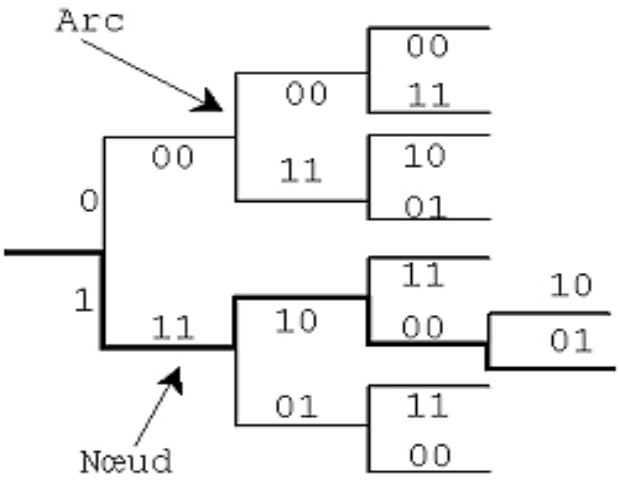

Fig. 1.8 - Exemple d'une structure en arbre ($\boldsymbol{R = 1/2}$, $\boldsymbol{m = 2}$)

Un arbre (fig. 1.8) est une structure partant d'un point appelé racine, il se compose d'arcs et de nœuds. Les arcs sont des traits verticaux dont le sens est déterminé par le bit d'information. Par convention, le 0 est représenté par un arc montant et le 1 par un arc descendant. Les nœuds sont des traits horizontaux indexés par les n sorties correspondant au bit d'entrée. Le chemin en gras sur la figure 1.7 correspond au mot de code 11100001 généré par la séquence d'entrée 1011.

c. Représentation en treillis

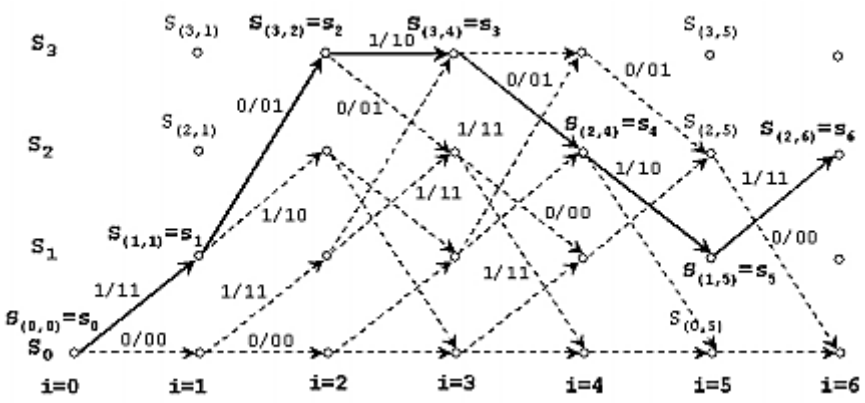

Fig. 1.9 - Exemple d'une représentation en treillis ($\boldsymbol{R = 1/2}$, $\boldsymbol{m = 2}$)

Contrairement aux deux précédentes, la représentation en treillis met en évidence le paramètre temporel. Chaque nœud $S_{(j,i)}$ correspond à un état particulier s_i du codeur à un instant i, où i représente l'indice du temps. Chaque branche est indexée par les bits qui se présentent en entrée et en sortie du codeur e/s. Chaque mot de code est associé à un chemin unique du treillis qu'on appelle ‹‹séquence d'état››.

1.5 Concaténation parallèle

Le but de la recherche dans le domaine de la théorie de l'information, est de trouver un système de contrôle d'erreur qui s'approchera le plus de la limite de Shannon. Pour cette raison, il faut trouver des codes ayant des bonnes propriétés de distance et déterminer leurs structures, et aussi il est nécessaire de développer un algorithme de décodage adéquat ayant la plus faible complexité [9]. La puissance de correction des erreurs d'un code correcteur dépend de la longueur de contrainte de l'encodeur, plus la longueur de contrainte est grande plus le code est puissant. En contrepartie, l'augmentation d'une telle longueur augmente la complexité du décodeur exponentiellement. Pour surmonter cette imperfection, on adopte l'usage de la concaténation des codes. Le codage concaténé utilise la concaténation de plusieurs codeurs simples [2].

Plusieurs approches ont été proposées, parmi ces solutions nous citons la notion de codage par la concaténation en série d'un code de Reed Solomon avec un code convolutionnel. Malheureusement tous les résultats des travaux qui ont été menés sur ce sujet n'ont pas réussi à s'approcher assez de la limite de Shannon [10]. En raison de ses

performances exceptionnelles pour des faibles rapports signal sur bruit, la concaténation parallèle des codes correcteurs d'erreurs a entraîné un intérêt remarquable.

En 1993 Berrou, Glavieux et Thitimajshima décrivent ce que nous appelons maintenant la concaténation parallèle [11]. En fait, cette dernière fut le point partant des codes turbo. Elle fut introduite avec la notion de décodage itératif. C'est à partir des codes convolutionnels que nous venons de décrire que les codes turbo ont été découverts. La concaténation est à la base de ces derniers.

1.6 Code turbo

Le code turbo repose sur un principe simple qui a souvent fait ses preuves : il est souvent plus avantageux de diviser un problème complexe en deux plus simples, mais dont l'efficacité combinée est souvent bien plus importante. Les codes turbo que nous considérons dans ce projet sont constitués de deux codeurs identiques de type récursifs et systématiques séparés par un entrelaceur. Nous avons représenté un codeur turbo classique à la figure 1.10.

La concaténation de deux codeurs récursifs et systématiques sert à générer deux codages de la même information. Il s'agit en fait de deux codeurs de structures équivalentes, c.à.d. qu'ils possèdent les mêmes polynômes générateurs. Tout l'intérêt réside en fait dans l'insertion d'un entrelaceur entre l'entrée du premier codeur et l'entrée du deuxième codeur [12].

Intéressons-nous maintenant au fonctionnement des codes turbo. Nous allons expliquer ces derniers à partir de la figure 1.10. Nous observons que ce codeur possède deux étages. Le premier étage correspond à la transmission de l'information d'entrée, c'est-à-dire que les bits sont transmis sans aucun changement. Ce que nous appelons le deuxième étage correspond à la génération des symboles de parité permettant la correction des erreurs. Cet étage produit deux symboles de parité pour chaque symbole d'information transmis.

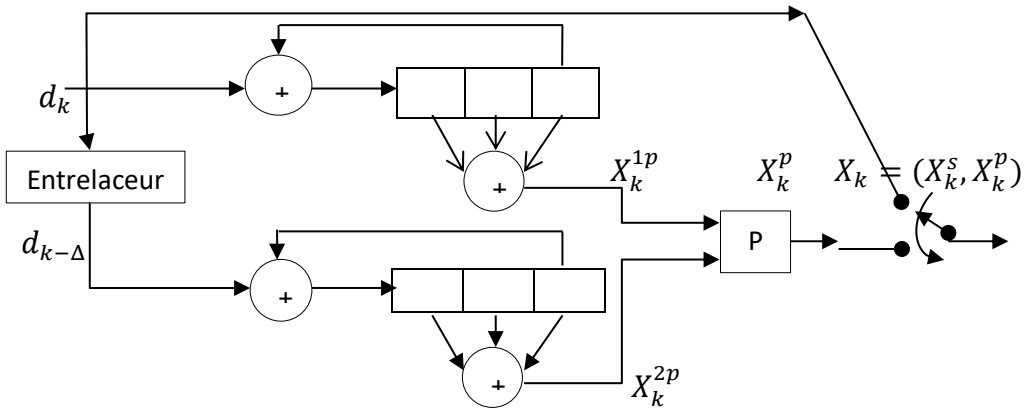

Fig.1.10 - Schéma d'un codeur turbo

Lorsqu'une séquence de symboles $\{d_k\}$ arrive au codeur, elle passe par deux étapes parallèles. La première correspond au premier codeur de l'étage supérieur. Cette étape est simplement le codage convolutionnel de cette séquence. Elle produit alors une séquence de symboles de parité $\{x_k^{1p}\}$.

La séquence d'entrée passe en parallèle par le codeur inférieur après avoir été entrelacée [10]. Ce deuxième codeur produit une séquence de symboles de parité $\{x_k^{2p}\}$. Une fois ces deux symboles de parité générés, ils peuvent être perforés ou non afin de produire la séquence ou le vecteur de parité $\{x_k^p\} = \{x_k^{1p}, x_k^{2p}\}$ qui sera multiplexé avec les symboles d'information $\{x_k^s\}$. Le but de la perforation est de supprimer certains symboles de parité afin de faire varier le taux de codage [3] [10]. Pour un bit en entrée, on obtient trois bits fournis en sortie du codeur. Cependant il existe en fait deux options pour le choix des bits à envoyer :

1- Soit on envoie systématiquement les trois bits de sortie en les multiplexant, c'est à dire qu'on transmet $[x_1^s, x_1^{1p}, x_1^{2p}, x_2^s, x_2^{1p}, x_2^{2p}]$. Le rendement du codeur est alors égal a $1/3$. Cela impose donc plus des données à transmettre mais assure aussi un décodage plus facile. On transmet dans ce cas 3N bits par séquence de N bits.

2- Soit on utilise une opération dite de "perforation" (Puncturing) qui consiste à n'envoyer qu'une fois sur deux les bits de parité (sorties des codeurs CRS) en alternant la sortie sélectionnée. On transmet alors $[x_1^s, x_1^{1p}, x_2^s, x_2^{2p}]$. Le rendement du codeur n'est alors que $1/2$, mais le décodage est moins efficace.

Si nous considérons la concaténation parallèle de deux codeurs systématiques dont les taux de codage sont $R_1 = k/n_1$ et $R_2 = k/n_2$, le taux global du codeur turbo est :

$$R_t = \frac{k}{n_1+n_2-k} = \frac{k}{\frac{k}{R_1}+\frac{k}{R_2}-k} \tag{1.4}$$

Où n_1 et n_2 présentent les nombres de bits à la sortie du codeur 1 et codeur 2.

La soustraction, au dénominateur, de est due au fait que les symboles systématiques ne sont transmis qu'une seule fois. Cette dernière équation s'écrit aussi :

$$\frac{1}{R_t} = \frac{1}{R_1} + \frac{1}{R_2} - 1 \tag{1.5}$$

Dans notre cas de la figure 1.8, les taux de codage R_1 et R_2 sont tous deux de $1/2$. Le taux global de notre codeur turbo sans perforation est alors $R_t = 1/3$.

Soulignons enfin le fait que dans l'algorithme de codage-décodage nous accordons beaucoup d'importance à l'état de la mémoire du codeur. En effet, pour pouvoir faire le décodage, il faut non seulement que la mémoire du codeur parte mais aussi revienne dans l'état (0,0) [1]. Si nous imposons l'état initial, l'état final dépend de la chaîne que nous donnons à l'entrée du codeur. Nous devons donc ajouter des bits à la chaîne de donnée, en fonction de l'état de la mémoire à laquelle elle conduit, pour forcer la mémoire à revenir à l'état (0,0).

1.7 Conclusion

Dans ce chapitre, nous avons définir le concept de base d'un système de communication par satellite en introduisant les éléments de ce système. Nous avons, par la suite, mentionné les deux types de canaux les plus utilisés dans les systèmes de communication et la modulation que nous avons utilisé dans notre système de codage. Enfin nous nous sommes intéressés aux codeurs convolutifs récursifs et systématique qui sont utilisés pour construire des codes turbo.

Nous avons ainsi décrit la partie codage de notre système. Il nous reste maintenant à parler du décodage et en grande partie de l'algorithme de décodage que nous utilisons pour les codes turbo.

CHAPITRE II : Le décodage turbo et l'algorithme MAP

2.1 Introduction

Le théorème de Shannon affirme qu'il existe au moins un codage de canal qui nous permet de transmettre de l'information sous forme d'une série de symboles discrets avec une probabilité d'erreur faible que l'on veut à condition que le débit d'information soit inférieur à la capacité de canal.

Après un long tâtonnement C. Berrou, A. Glavieux et P. Thitimajshima, en 1993 [11] ont trouvé un très bon code correcteur d'erreurs fonctionnant à moins de 0.5 dB de la limite de Shannon pour un canal à bruit gaussien [13]. Cette rupture technologique dans le domaine du codage de canal a tout d'abord surpris la communauté scientifique, mais les résultats ont été très rapidement confirmés. La solution proposée, qui consiste en une concaténation parallèle ou série de deux codes convolutifs, généralement récursifs systématiques identiques [14], au travers d'un entrelaceur, est connue sous le nom «code turbo».

Le but de l'entrelaceur dans un codeur turbo est de faire la permutation des données émises afin de casser les paquets d'erreur surviennent lors de la transmission, et ce pour rendre leur distribution plus uniforme, ce qui donne au turbo code convolutif une grande efficacité de corriger les erreurs [15].

Le décodage des codes turbo peut être réalisé à partir de deux décodeurs DEC1 et DEC2. Ce décodage est un décodeur qui utilise un algorithme Maximum A Posteriori MAP. Un tel décodage permet d'extraire de l'information sur chacun des décodeurs constitutifs et de l'échanger entre ces deux décodeurs. L'information extraire d'un décodeur est dite « extrinsèque » et est réinjectée à l'itération suivante dans un autre décodeur composant afin de bénéficier de la diversité de codage. Le processus pouvant se répéter plusieurs fois, on parle de décodage itératif ou de décodage turbo.

2.2 Le décodage itératif et l'algorithme MAP

Dans un système de communication numérique sur un canal bruité, un codeur est souvent utilisé à l'émetteur avant l'étape de modulation, afin de corriger les erreurs de transmission au récepteur. Lorsque la structure du code s'y prête, le processus de décodage peut être exécuté en plusieurs étapes ou itérations simples, d'où le nom de décodage itératif.

2.2.1 Principe du Maximum A Posteriori MAP

L'algorithme MAP consiste à minimiser la probabilité d'erreur par symbole ou par bit. Dans le cas du MAP on parlera de la règle du maximum a posteriori c.à.d. de prendre le maximum des probabilités a posteriori (APP) du symbole ou bit envoyé dans le canal.

Le calcul de l'APP se base sur l'observation de la séquence reçue, d'où la nomination « a posteriori ». Afin d'éviter la confusion des termes qui sont utilisés dans la littérature, nous

rappelons que la notion Algorithme MAP signifie que la règle de maximum a posteriori associée à un symbole a été appliquée. Nous pouvons alors parler de l'algorithme MAP symbole-par-symbole. La règle de maximum a posteriori peut être appliquée à un symbole ou une séquence. Dans le cas où cette règle est appliquée à une séquence (ou un mot de code par exemple) et que la distribution des mots de source est uniforme, ceci revient exactement à appliquer le principe de Vraisemblance Logarithmique (Maximum Likelihood ML). En effet, il suffit d'appliquer la règle de Bayes afin de prouver l'identité entre les deux règles. Avec le MAP à la sortie du décodeur, les bits décodés ne sont pas comparées à un seuil donné, les valeurs (des bits) obtenues ne sont pas modifiées (par large approximation ou quantification). Dans ce cas, nous pouvons parler d'une décision douce (Soft) en opposition à une décision dure où la valeur du bit à la sortie du décodeur est affectée soit à la valeur 0 soit à 1.

Soient les événements $A_i, i = 1,2, \dots, M$, l'ensemble des messages transmis dans un intervalle de temps donné, $P(A_i)$ représente la probabilité a priori de ces événements. Soit le signal reçu qui n'est qu'un des corrompu par du bruit. B est une variable aléatoire continue de densité de probabilité $p(B)$. L'expression $P(A_i|B)$ est l'APP de l'événement A_i sachant que le signal reçu B a été observé. Le détecteur MAP cherche à maximiser l'APP $P(A_i|B)$. Cette dernière s'exprime sous la forme [10].

$$P(A_i|B) = \frac{P(B|A_i)P(A_i)}{p(B)} \tag{2.1}$$

Où $P(B|A_i)$ est la fonction de densité de probabilité conditionnelle du vecteur observé B, et $P(A_i)$ est la probabilité a priori du $i^{ème}$ signal transmis. Le dénominateur de l'équation (3.4) est donné par

$$p(B) = \sum_{i=1}^{M} p(B|A_i)P(A_i) \tag{2.2}$$

Notons que $p(B|A_i)$ est appelée aussi la fonction de vraisemblance. Le critère de décision qui consiste à prendre le maximum de $p(B|A_i)$ parmi les M signaux est le principe du ML si les signaux sont équiprobables.

Si la distribution des signaux A_i est uniforme (c.à.d. les signaux A_i sont équiprobables), or $p(B)$ est indépendant du signal A_i transmis, donc maximiser $P(A_i|B)$ revient à maximiser l'expression $p(B|A_i)$. Donc le détecteur basé sur le principe MAP fera la même décision que celui basé sur le principe ML.

Dans notre étude les événements A_i appartiennent au champ de Galois $GF(2)$ dont les éléments sont $\{0,1\}$ où 0 est l'élément neutre sous l'addition $\oplus$, donc $M = 2$. Soit R la séquence reçue sur toutes les classes de signaux. Alors l'équation (2.1) devient

$$P(d_k = i|R) = \frac{P(R|d_k = i)P(d_k=i)}{p(R)} \tag{2.3}$$

La règle de décision MAP consiste à comparer les probabilités APP $P(d_k = 0|R)$ et $P(d_k = 1|R)$, ensuite à en prendre le maximum, ceci peut se résumer dans l'équation suivante :

$$P(d_k = 1|R) < P(d_k = 0|R) \implies H_0$$
$$P(d_k = 1|R) > P(d_k = 0|R) \implies H_1 \tag{2.4}$$

Dans l'équation (2.4) l'hypothèse H_1 signifie que le détecteur MAP assigne d_k à la valeur 1 dans le cas ou $P(d_k = 1|R)$ est supérieur à $P(d_k = 0|R)$. Dans le cas contraire le détecteur MAP choisit $d_k = 0$ ce qui correspond à l'hypothèse H_0.

La décision ferme optimale donne d_k au sens du MAP est alors la suivante :

$$d_k = \begin{cases} 0 & si \quad P(d_k = 0|R) > P(d_k = 1|R) \\ 1 & sinon \end{cases} \tag{2.5}$$

La décision souple est définie par le rapport de vraisemblance logarithmique LLR.

$$LLR(d_k) = \log\left[\frac{P(d_k=1|R)}{P(d_k=0|R)}\right] \tag{2.6}$$

En utilisant (2.3) la décision souple peut être décomposée en :

$$LLR(d_k) = \Lambda_c(R) + \Lambda_a(d_k) \tag{2.7}$$

Où $\Lambda_c(R) = \log\left[\frac{p(R|d_k=1)}{p(R|d_k=0)}\right]$ est le rapport de vraisemblance logarithmique du canal et $\Lambda_a(d_k) = \log\left[\frac{P(d_k=1)}{P(d_k=0)}\right]$ est le rapport de vraisemblance logarithmique a priori. Le critère de décision au sens du MAP peut alors se s'écrire :

$$d_k = \begin{cases} 1 & si \quad \Lambda_a(d_k) > 0 \\ 0 & sino \end{cases} \tag{2.8}$$

Il s'ensuit que le signe de la décision souple détermine la décision ferme.

Pour un système de transmission codé, nous pouvons montrer que si l'information a priori est distribuée de manière indépendante, la sortie souple d'un décodeur peut s'écrire sous la forme [16].

$$LLR(d_k) = \Lambda_c(R) + \Lambda_a(d_k) + \Lambda_e(d_k) = \Lambda_c(d_k) + \Psi(d_k) \tag{2.9}$$

Où $\Lambda_e(d_k)$ est le rapport de vraisemblance logarithmique représentant la connaissance acquise grâce au processus de décodage. La quantité $\Psi(d_k) = \Lambda_a(d_k) + \Lambda_e(d_k)$ est appelée information extrinsèque, indépendante de R.

2.2.2 Notations et règle de décision

Afin d'appliquer le principe MAP aux turbo codes, considérons un code récursif et systématique de taux de codage $R = 1/2$ (voir chapitre 1, figure 1.5). Nous allons adopter les notations suivantes :

- $d_k = i, i = 0{,}1$ le bit d'information présent à l'entrée du codeur à l'instant k ;
- $M = (K - 1)$ bits, est la mémoire du codeur ;
- $S_k = m, m = 0{,}1, \dots, 2^M$ est l'état du codeur à l'instant ;
- N est la taille de la séquence à coder (ou encore la longueur du bloc exprimée en bits) ;
- $(d_1, \dots, d_k, \dots, d_N) = (X_1^s, \dots, X_k^s, \dots, X_N^s)$ est la séquence à coder ;
- $(X_1^s, \dots, X_k^s, \dots, X_N^s)$ est appelée la séquence systématique ;
- $(X_1^p, \dots, X_k^p, \dots, X_N^p)$ est la séquence de parité à la sortie du codeur récursif et systématique ;
- $R_k = (r_k^s, r_k^p)$ est la version bruitée de (X_k^s, X_k^p) à l' instant k ;
- $R_1^N = (R_1, R_2, \dots, R_N)$ est la séquence reçue après passage dans un canal.

En supposant que la séquence reçue est R_1^N, l'expression du LLR devient :

$$LLR(d_k) = \log\left[\frac{P(d_k=1|R_1^N)}{P(d_k=0|R_1^N)}\right] \tag{2.10}$$

Le calcul de l'APP $P(d_k = i|R_1^N), i = 1{,}0$ s'avère complexe d'où l'idée d'introduire la probabilité conjointe (JD) définie par :

$$\lambda_k^i(m) = P(d_k = i, S_k = m|R_1^N) \tag{2.11}$$

Donc l'APP d'un bit décodé d_k, peut s'exprimer en fonction de la JD et est égale à :

$$P(d_k = i|R_1^N) = \sum_{m=0}^{2^M-1} \lambda_k^i(m) \tag{2.12}$$

Où $i = 0{,}1$ et la sommation est sur les 2^M états du codeur.

En remplaçant l'expression de la probabilité conjointe dans l'équation (2.10), nous obtenons :

$$LLR(d_k) = \log\left[\frac{\sum_{m=0}^{2^M-1} \lambda_k^1(m)}{\sum_{m=0}^{2^M-1} \lambda_k^0(m)}\right] \tag{2.13}$$

Le décodeur MAP peut faire la décision sur le bit décodé en comparant le $LLR(d_k)$ à un seuil égal à zéro.

$$Si\ LLR(d_k) \geq 0\ le\ bit\ décodé\ est\ 1 \tag{2.14}$$

$$Si\ LLR(d_k) < 0\ le\ bit\ décodé\ est\ 0 \tag{2.15}$$

Maintenant que nous avons introduit le principe de l'algorithme MAP, nous pouvons décrire la dernière version de l'algorithme que nous avons utilisée.

2.3 Algorithme MAP utilisé

Dans cette partie, nous allons introduire les nouvelles notions de la dernière version de l'algorithme MAP.

2.3.1 Expression de la probabilité conjointe

En notant que les événements R_k sont indépendants et en utilisant des relations les plus élémentaires en probabilité, nous pouvons développer l'expression de la probabilité conjointe (2.11) sous la forme suivante :

$$\lambda_k^i(m) = \frac{P(d_k=i,S_k=m,R_1^{k-1},R_1^N)}{P(R_1^N)} \tag{2.16}$$

De plus, nous savons, de règle générale que :

$$P(A|B) = \frac{P(B,A)}{P(B)} \tag{2.17}$$

D'où :

$$P\left(d_k = i, S_k = m, R_1^{k-1}, R_1^N\right) = P\left(R_1^{k-1}\middle| d_k = i, S_k = m, R_k^N\right) \times P(d_k = i, S_k = m, R_k^N) \tag{2.18}$$

On utilise maintenant (2.18) dans (2.16), on obtient :

$$\lambda_k^i(m) = \frac{P\left(R_1^{k-1}\middle| d_k = i, S_k = m, R_k^N\right) \times P\left(d_k = i, S_k = m, R_k^N\right)}{P(R_1^N)}$$

$$= \frac{P\left(R_1^{k-1}\middle| d_k = i, S_k = m, R_k^N\right) \times P(d_k = i, S_k = m, R_1^k, R_{k+1}^N)}{P(R_1^N)}$$

$$= \frac{P\left(R_1^{k-1}\middle| d_k = i, S_k = m, R_k^N\right) \times P\left(R_{k+1}^N\middle| d_k = i, S_k = m, R_1^k\right) \times P\left(d_k=i,S_k=m,R_1^k\right)}{P\left(R_1^N\right)} \tag{2.19}$$

Nous allons définir la métrique d'état en avant FSM par :

$$\alpha_k(m) = P(R_1^{k-1}|d_k = i, S_k = m, R_k^N) \tag{2.20}$$

Or les événements produits après l'instant n'influencent pas les événements qui les ont précédés, donc l'équation précédente se réduit à :

$$\alpha_k(m) = P(R_1^{k-1}|S_k = m) \tag{2.21}$$

De même nous définissons aussi la métrique d'état en arrière BSM comme :

$$\beta_k(m) = P(R_k^N|S_k = m) \tag{2.22}$$

Nous définissons la métrique de branche (BM) par :

$$\delta_k^i(m) = P(d_k = i, S_k = m, R_1^k) \tag{2.23}$$

En remplaçant les expressions des FSM, BSM et BM dans l'équation (2.19) :

$$\lambda_k^i(m) = \frac{\alpha_k(m) \times P\left(R_{k+1}^N \middle| d_k = i, S_k = m, R_1^k\right) \times \delta_k^i(m)}{P(R_1^N)}$$

$$= \frac{\alpha_k(m) \times P\left(R_{k+1}^N \middle| S_{k+1} = f(i,m)\right) \times \delta_k^i(m)}{P(R_1^N)}$$

$$= \frac{\alpha_k(m) \times \beta_{k+1}(f(i,m)) \times \delta_k^i(m)}{P(R_1^N)} \tag{2.24}$$

En connaissant l'état du codeur $S_k = m$ et le bit d'entrée $d_k = i$ à l'instant k, l'état successeur S_{k+1} peut se noter sous la forme $S_{k+1} = f(i,m)$. L'égalité précédente signifie que l'état précédent de S_{k+1} est l'état m et que le bit d'entrée à l'instant k est $d_k = i$.

Finalement, en remplaçant l'expression de la probabilité conjointe dans l'équation (2.13), nous obtenons :

$$LLR(d_k) = \log\left[\frac{\sum_{m=0}^{2^M-1} \alpha_k(m) \times \beta_{k+1}(f(i,m)) \times \delta_k^i(m)}{\sum_{m=0}^{2^M-1} \alpha_k(m) \times \beta_{k+1}(f(0,m)) \times \delta_k^0(m)}\right] \tag{2.25}$$

D'après l'équation (2.25), le LLR est en fonction des trois expressions définies précédemment: FSM, BSM et BM. Donc, il suffit de calculer ces trois expressions afin de fournir une valeur pour le LLR. Nous allons d'abord simplifier l'expression de la FSM ensuite celle de la BSM et finalement celle de la métrique de branche BM.

2.3.2 Expression de la métrique d'état en avant FSM

Nous pouvons remarquer que la FSM à l'instant k dépend de la séquence reçue R_1^k qui précède l'instant k (ou au plus jusqu'à l'instant k). En utilisant la règle de Bayes, nous pouvons exprimer la métrique d'état en avant sous la forme :

$$\alpha_k(m) = P(R_1^{k-1}|S_k = m) \tag{2.26}$$

$$\alpha_k(m) = \sum_{i=0}^{2^{M-1}} \sum_{j=0}^{1} P(d_{k-1} = j, S_{k-1} = i, R_1^{k-1}|S_k = m) \tag{2.27}$$

$$\alpha_k(m) = \sum_{i=0}^{2^{M-1}} \sum_{j=0}^{1} \begin{matrix} P\left(R_1^{k-2}\middle|S_k = m, d_{k-1} = j, S_{k-1} = i, R_{k-1}\right) \\ \times P(d_{k-1} = j, S_{k-1} = i, R_{k-1}|S_k = m) \end{matrix} \tag{2.28}$$

Sachant que le passage de l'état $S_{k-1} = m'$ a été provoqué par le bit $d_{k-1} = j$, nous pouvons compacter ces informations sous la forme $S_{k-1} = b(j,m)$. Ceci signifie que l'état S_{k-1} est l'état qui précède l'état $S_k = m$ sachant que le bit d'entrée au codeur à l'instant $(k-1)$ est égal j.

$$\begin{aligned} \alpha_k(m) &= \sum_{i=0}^{2^{M-1}} \sum_{j=0}^{1} P\left(R_1^{k-2}|S_{k-1} = b(j,m)\right) P(d_{k-1} = j, S_{k-1} = b(j,m), R_{k-1}) \\ &= \sum_{j=0}^{1} \alpha_{k-1}(b(j,m)) \delta_{k-1}^{j}(b(j,m)) \end{aligned} \tag{2.29}$$

Nous pouvons déduire que la métrique d'état à l'instant k s'exprime en fonction de la FSM à l'instant $k-1$ d'où la nomination « en avant ». En effet, il faut connaître la valeur initiale à l'instant $k = 0$ afin d'avancer aux instants suivants.
La métrique d'état en avant peut être schématisée sous forme graphique comme à la figure 2.1.

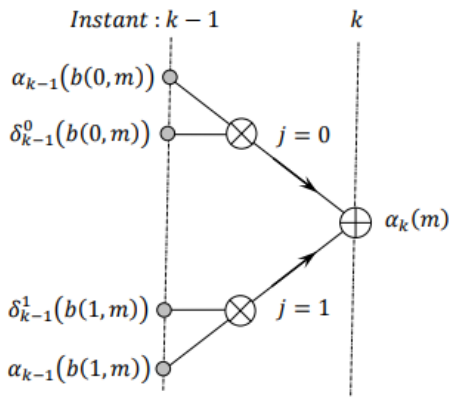

Fig. 2.1 - Représentation graphique de la métrique d'état en avant

2.3.3 Expression de la métrique d'état en arrière BSM

Contrairement à la FSM, la métrique d'état en arrière a l'instant dépend de la séquence reçue après cet instant k (ou à partir de l'instant k) et jusqu'à recevoir la totalité de la séquence (instant N). En utilisant encore la règle de Bayes, nous pouvons exprimer la métrique d'état en arrière BSM sous la forme :

$$\beta_k(m) = P(R_k^N | S_k = m) \tag{2.30}$$

$$\beta_k(m) = \sum_{i=0}^{2^M-1} \sum_{j=0}^{1} P\left(d_k = j, S_{k+1} = i, R_k^N | S_k = m\right) \tag{2.31}$$

$$\beta_k(m) = \sum_{i=0}^{2^M-1} \sum_{j=0}^{1} \begin{matrix} P\left(R_{k+1}^N | S_k = m, d_k = j, S_{k+1} = i, R_k\right) \\ \times P(d_k = j, S_{k+1} = i, R_k | S_k = m) \end{matrix} \tag{2.32}$$

Comme pour la FSM, nous pouvons définir une fonction $S_{k+1} = f(j, m)$ de façon à simplifier cette expression. Il vient alors que :

$$\beta_k(m) = \sum_{j=0}^{1} P\left(R_{k+1}^N | S_k = f(j,m)\right) P(d_k = j, S_k = m, R_k) \tag{2.33}$$

$$\beta_k(m) = \sum_{j=0}^{1} \delta_k^j(m) \beta_{k+1}(f(j,m)) \tag{2.34}$$

Nous pouvons déduire que la BSM à l' instant k s'exprime en fonction de la BSM à l'instant $k+1$ d'où la nomination « en arrière ». En effet, il faut connaitre la valeur initiale à l' instant $k = N$ afin de reculer dans le treillis et de calculer les BSM précédentes. La métrique d'état en arrière peut être aussi schématisée sous forme graphique à la figure 2.2.

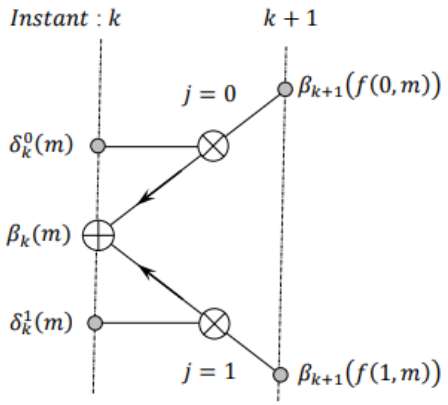

Fig. 2.2 - Représentation graphique de la métrique d'état en arrière

2.3.4 Calcul de la métrique de branche BM

Comme nous l'avons indiqué précédemment le calcul de la métrique de branche BM est nécessaire afin de calculer le logarithme du rapport de vraisemblance. D'après l'expression

de δ_k^i (équation (2.23)), la BM à l'instant k dépend seulement de l'instant même et non plus des instants précédents ou suivants d'où la nomination métrique de branche.

$$\delta_k^i(m) = P(d_k = i, S_k = m, R_1^k) \tag{2.35}$$

Donc, toujours grâce à la règle de Bayes :

$$\delta_k^i(m) = P(R_k | d_k = i, S_k = m) \times P(d_k = i, S_k = m) \tag{2.36}$$

$$\delta_k^i(m) = P(R_k | d_k = i, S_k = m) \times P(S_k = m | d_k = i) \times P(d_k = i) \tag{2.37}$$

En remplaçant $R_k = (r_k^s, r_k^p)$ dans l'équation précédente nous obtenons :

$$\begin{aligned}\delta_k^i(m) = P(r_k^s | d_k = i, S_k = m) \times P\left(r_k^p | d_k = i, S_k = m\right) \\ \times P(S_k = m | d_k = i) \times P(d_k = i)\end{aligned} \tag{2.38}$$

Les deux premiers termes de la métrique de branche dépendent des symboles recueillis à la sortie du canal. Donc le type canal utilisé affecte l'expression de la BM.

2.4 Principe du décodeur itératif MAP

Le décodage des codes turbo peut être réalisé à partir de deux décodeurs DEC1 et DEC2 associés selon le principe représenté sur la figure 2.3. Les symboles de redondance bruités $R_k^{(2)}$, $R_k^{(3)}$ sont envoyés vers les décodeurs DEC1 et DEC2 respectivement lorsque la redondance est produit par les codeurs CRS1 et CRS2 respectivement. Le décodeur DEC1 reçoit des symboles bruités $R_k^{(1)}$, $R_k^{(2)}$ issus du démodulateur et produit une décision relative à chaque symbole d_k. Un entrelaceur (Π), place entre les deux décodeurs élémentaires permet d'éclater les paquets d'erreurs produit par le décodeur DEC1. Le décodeur DEC1 associe à chaque symbole décodé d_k une mesure de fiabilité sous forme du logarithme de son rapport de vraisemblance $LLR_1(d_k)$, et DEC2 associe au même symbole d_k une autre mesure de fiabilité $LLR_2(d_k)$. L'information extrinsèque $\Psi_1(d_k)$ extraire du décodeur DEC1 est réinjectée à l'itération suivante dans le décodeur DEC2 afin de bénéficier de la diversité de codage, et l'information extrinsèque $\Psi_2(d_k)$ extraire du décodeur DEC2 passe par le désentrelaceur puis elle est réinjectée dans le décodeur DEC1. Le processus pouvant se répéter plusieurs fois (ce qui justifie l'appellation décodage itératif).

Après un nombre précisé des itérations, une décision ferme du décodeur DEC2 (ou DEC1) donne la séquence estimée $\hat{d}$ de la séquence originale d en basant sur la relation (2.8).

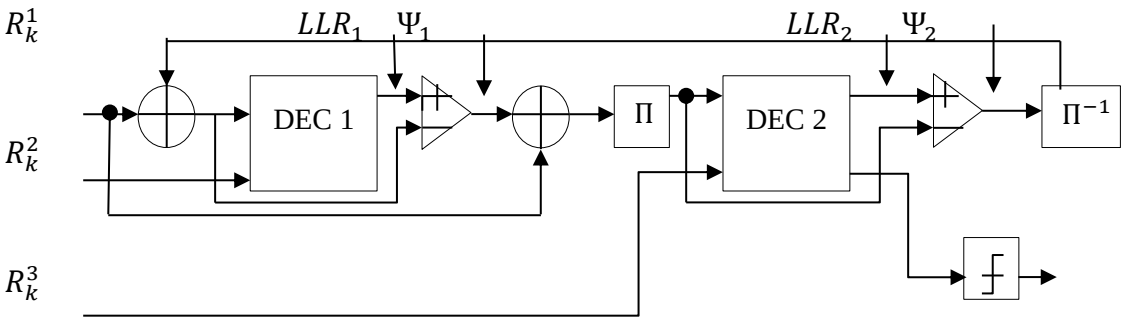

Fig. 2.3 -Schéma bloc d'un décodeur itératif.

2.5 Entrelacement

L'entrelacement consiste à permuter une séquence de bit de manière à ce que deux symboles proches à l'origine soient le plus éloignés possibles l'un de l'autre. Cela permet en particulier de transformer une erreur portant sur des bits regroupés en une erreur répartie sur l'ensemble de la séquence.

2.5.1 Avantage d'utilisation des entrelaceurs

Dans la théorie des codes correcteurs d'erreurs, plusieurs propriétés associées avec les mots de code ont été définies. Les deux propriétés importantes sont la distance de Hamming et le poids de Hamming. La distance de Hamming entre deux mots codés est le nombre total de positions, où les deux mots codes sont différents. Le poids de Hamming d'un mot de code est défini comme étant le nombre total des positions par lesquelles il est différent par rapport au vecteur nul. L'une des propriétés qui définissent la qualité d'un code, est appelée la distance minimale d_{min}. Elle est définie comme la plus petite distance entre les mots de code distincts, et donne une mesure de la qualité de code pour détecter et corriger les erreurs.

Un code à distance minimale d_{min} peut détecter jusqu'à $(d_{min} - 1)$ erreurs et peut corriger jusqu'à t erreurs, t vérifie la relation suivante :

$$d_{min} \geq 2t + 1 \text{ où } t = floor\left(\frac{d_{min}-1}{2}\right) \tag{2.38}$$

Considérons un code qui peut corriger jusqu'à erreurs. Nous prenons un exemple de transmission de données simplifiée sur un canal bruyant comme le montre la figure 2.4. Nous voyons que certaines données sont perdues lors de la transmission parce que deux erreurs apparus dans ces mots de code. Comme le décodeur peut corriger une seule erreur dans chaque mot de code ainsi ces deux mots de code ont fini comme non décodé. Une solution pour améliorer la fiabilité de transmission est d'augmenter la puissance de transmission, mais cette solution est couteuse et difficile à réaliser dans le domaine spatial.

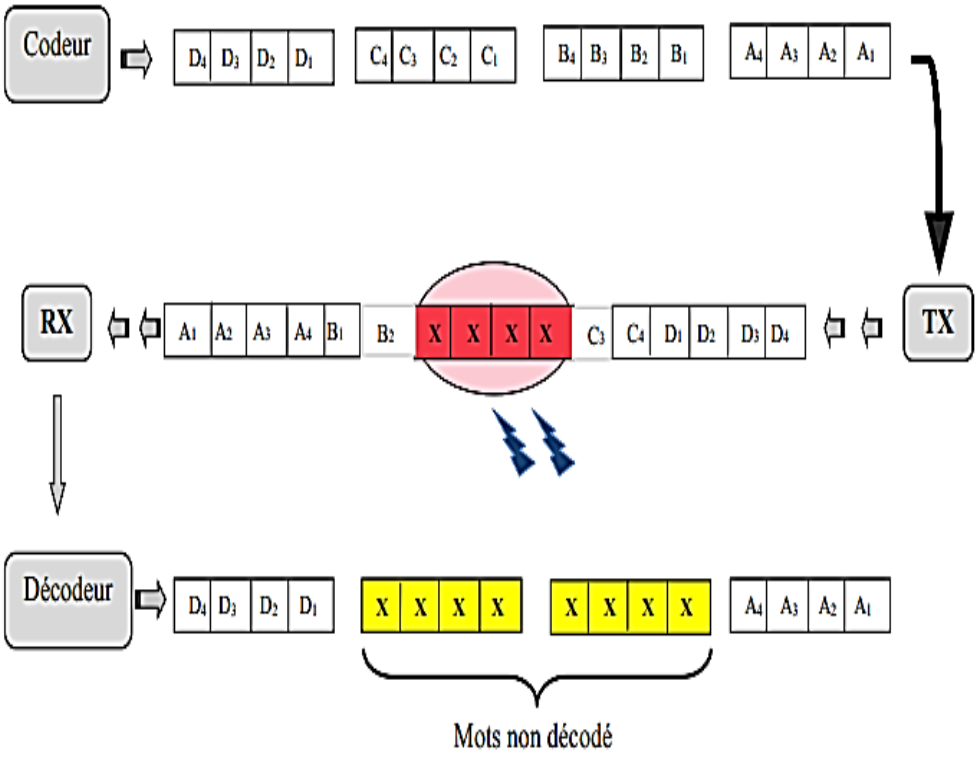

Fig. 2.4 - Modèle simplifiée de transmission et de réception sur un canal bruité [17]

Une autre façon pour améliorer la fiabilité de transmission contre le bruit de canal est d'intoduire un entrelaceur entre le codeur et l'émetteur. Vice versa un désentrelaceur doit être incorporé entre le récepteur et le décodeur comme représenté sur la figure 2.5.

Les bits transmis sont les mêmes, mais commandés d'une manière spéciale. Cette réorganisation des bits est appelé permutation des données.

Considérant même effet du bruit, comme dans la figure 2.4, les données reçues sont corrompues sous la forme d'un paquet d'erreurs. Dans ce cas, au lieu de fournir directement les données reçues vers le décodeur, il est tout d'abord transmis au désentrelaceur. Le réarrangement effectué par le désentrelaceur donne l'avantage que chacun des mots de code a au maximum une erreur.

Puisque le décodeur est capable de corriger un mot de code avec une erreur, par consquent, tous les mots de code sont récupérés et décodés correctement. L'entrelaceur fait le réarrangement de chaque mot de code sur un domaine spatial plus large qui donne une robustesse au même code contre les erreurs fatales induites par le bruit sur le canal de transmission.

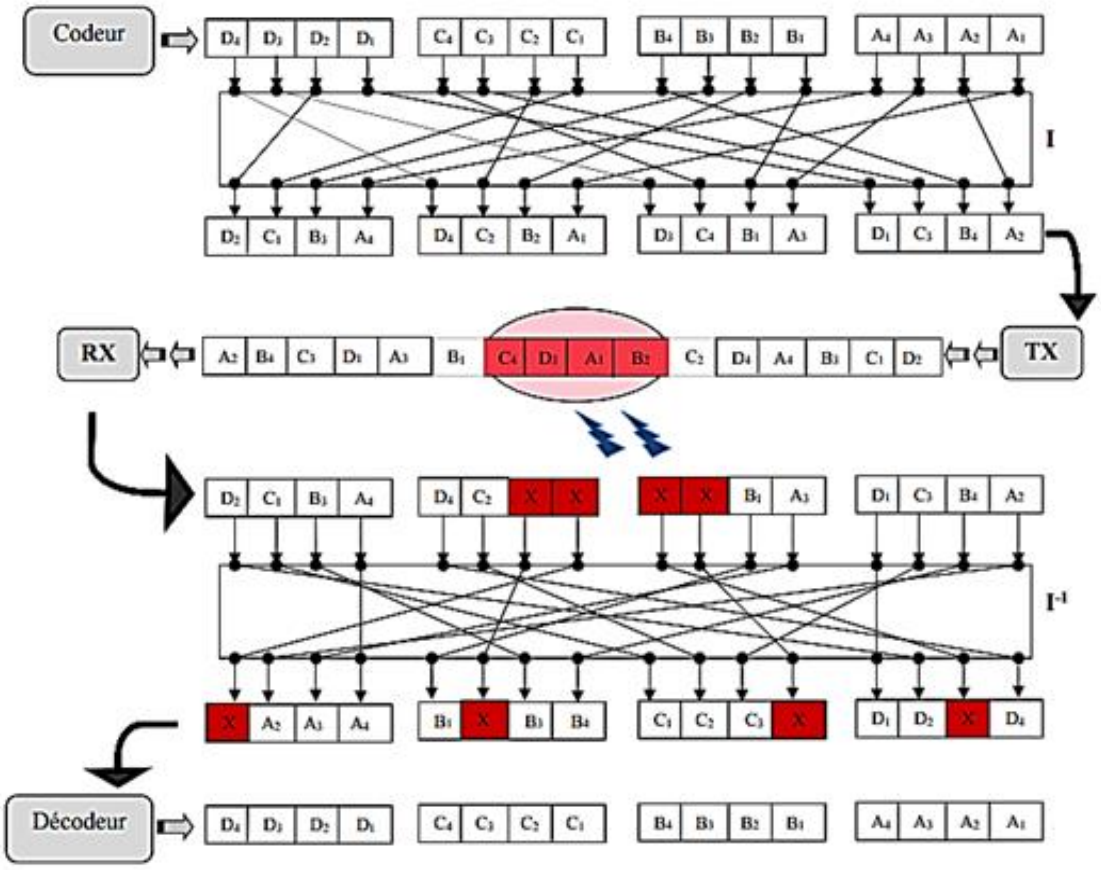

Fig. 2.5 - Modèle simplifiée de transmission et de réception sur un canal bruité
avec entrelaceur et désentrelaceur [17]

2.5.2 Types d'entrelacement

La taille et le type d'entrelacement choisis sont deux facteurs liés aux performances d'un turbo code. Le schéma synoptique ci-dessous nous donne les différents types d'entrelacement :

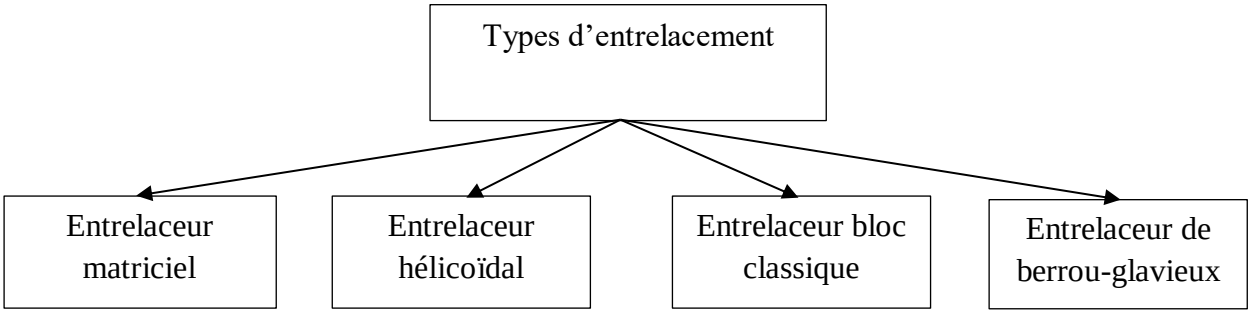

Fig. 2.6 – Type d'entrelacement

2.5.2.1 Entrelaceur matriciel

Un entrelaceur matriciel est un entrelaceur de période $P = N.M, P$ représente la taille de la matrice d'entrelacement. Ce type d'entrelaceur se caractérise par un processus dans lequel les données sont écrites ligne par ligne dans une matrice et ensuite ces données sont lues colonne par colonne.

Nous allons essayer à l'aide d'un exemple d'illustrer le fonctionnement d'entrelaceur en matrice.

Supposons qu'une séquence de 9 symboles (N = 3, M= 3, P = 9) soit reçue à l'entrée de l'entrelaceur. Nous allons d'abord placer ces 9 symboles dans une matrice (voir figure (2.6)) dans un ordre croissant des nombres indiques dans les cases (c.à.d. le premier symbole reçu est placé dans la case 0, le deuxième dans la case 1 et ainsi de suite). Les symboles sont réécrits à la sortie de l'entrelaceur en matrice dans l'ordre suivant :

Sortie de l'entrelaceur : 0 3 6 1 4 7 2 6 8

0	1	2
3	4	5
6	7	8

Fig. 2.7 - Entrelaceur en matrice de taille 3x3

2.5.2.2 Entrelaceur hélicoïdal

L'entrelaceur hélicoïdal est un autre entrelaceur ligne-colonne modifié qui écrit des données ligne par ligne dans les mémoires et les lit en diagonale à partir de l'entrée en haut à gauche de l'entrelaceur en utilisant la permutation suivante :

$$\pi(t) = i_r C + j_r \tag{2.39}$$

$$i_r = C \times R - 1 - t(mod\ R) \tag{2.40}$$

$$j_r = t(mod\ C) \tag{2.41}$$

Où R et C sont premiers et représentent le nombre de ligne et colonne de l'entrelaceur, respectivement. La figure 2.7 présente un exemple d'entrelacement Hélicoïdal.

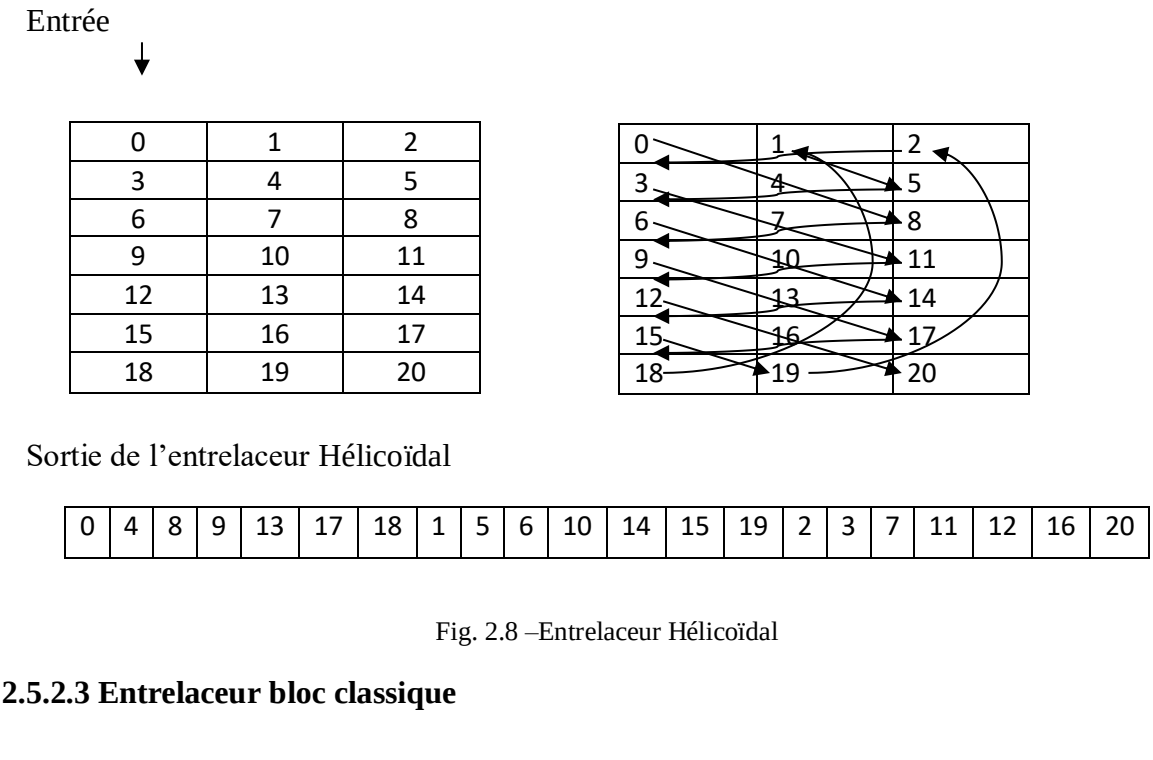

Fig. 2.8 –Entrelaceur Hélicoïdal

2.5.2.3 Entrelaceur bloc classique

Dans la plupart des cas, l'entrelaceur et le désentrelaceur sont implémentés séparément puisqu'ils sont différents. Par conséquent, le besoin en espace mémoire risque d'être énorme surtout pour des entrelaceurs de grandes tailles. Ce problème peut être résolu par l'utilisation d'un entrelaceur bloc classique.

En effet, l'entrelaceur bloc classique échange les positions des symboles d'une séquence deux à deux sans répétition. Autrement dit, pour deux symboles A et B de positions initiales respectives I et J, si A à la position J après entrelacement alors nécessairement B devra occuper celle de I. Un exemple illustrant le principe de l'entrelaceur bloc classique est présenté à la figure 2.8.

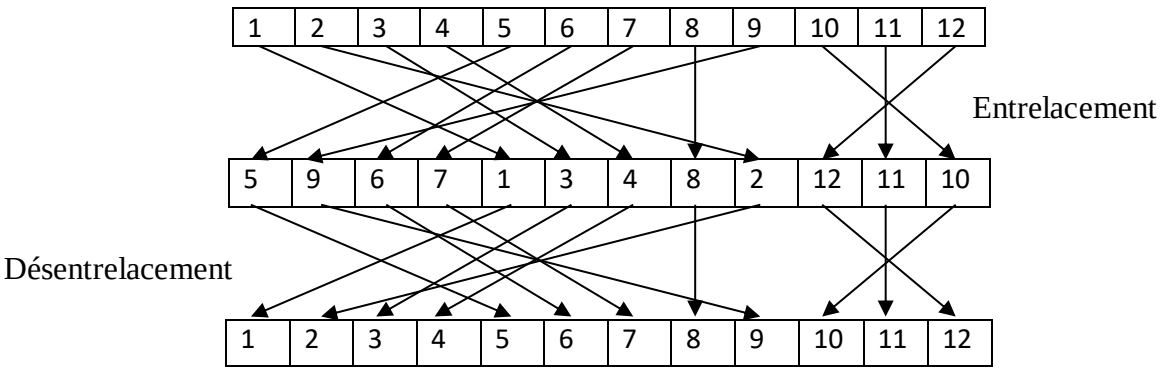

Fig. 2.9 – Entrelaceur et désentrelaceur bloc classique

2.5.2.4 Entrelaceur de berrou-glavieux

Afin d'améliorer les performances de l'entrelaceur ligne-colonne dans les applications des codes turbo, Berrou et Glavieux conçu un entrelaceur, qui améliore le modèle de permutation. La taille K de l'entrelaceur est choisie comme étant une puissance de 2 et le nombre de lignes est choisi égal au nombre de colonnes ($K = w \times w$). Les permutations sont obtenues en appliquant les formules suivantes

$$i_r = \left(\frac{w}{2} + 1\right)(i + j)(mod\ w) \tag{2.42}$$

$$\xi = (i + j)(mod\ 8) \tag{2.43}$$

$$j_r = [P(\xi)(j + 1)] - 1(mod\ w) \tag{2.44}$$

ξ	P
0	17
1	37
2	19
3	29
4	41
5	23
6	13
7	7

Tableau 2.1 – La fonction pseudo-aléatoire (ξ)

Où i et j sont la ligne et la colonne d'écriture d'un symbole et i_r et j_r donnent la position de lecture. En outre, $P(\xi)$ est un nombre premier avec w, qui est déterminée par les valeurs présentées dans le tableau 2.1.

2.5.2.5 Entrelaceur aléatoire

Les entrelaceurs aléatoires sont les plus performants pour les codes turbo dans le cas des blocs d'information de tailles moyenne et grande [18]. Leurs inconvénients par rapport aux entrelaceurs cités précédemment résident dans leur complexité.

Le concept fondamental d'un entrelaceur aléatoire est simple mais sa réalisation en pratique est plus complexe que celle des entrelaceurs bloc classique et hélicoïdal. Une fois que les symboles d'un bloc sont introduits dans un entrelaceur aléatoire, les symboles en sortie sont choisis d'une façon aléatoire de telle manière qu'il ne faut pas répéter le même symbole déjà sélectionné.

Comme la sélection est aléatoire, il sera impossible de connaître les positions des symboles à la sortie d'entrelaceur. Par conséquent, il serait utile de garder une table de correspondance entre les anciennes et les nouvelles positions des symboles entrelacés afin de pouvoir désentrelacer ces derniers.

Ce type d'entrelaceurs donne de très bons résultats pour les turbo codes [19], parce qu'il est classé parmi les entrelaceurs optimaux [20].

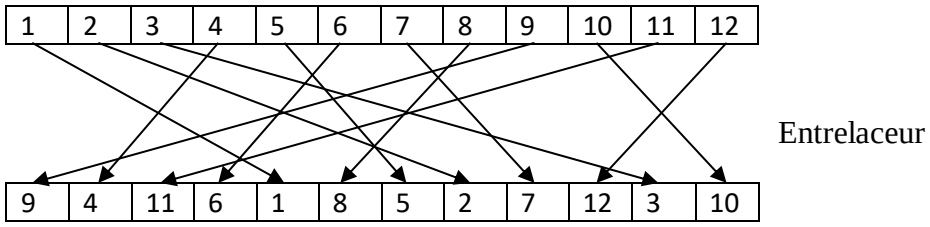

Fig. 2.10 – Entrelaceur aléatoire.

2.6 Conclusion

Dans ce chapitre, nous avons donc étudié deux notions très importantes des Turbo codes. Dans un premier temps, la notion de décodage itératif a été introduite, pour ensuite nous concentrer sur le décodeur turbo. Par la suite, nous avons pu nous concentrer sur un des éléments importants des Codes Turbo, à savoir l'entrelaceur. Ce dernier est un mélangeur de bits qui permet de décorder l'information, vue du décodage turbo. Nous avons par la fin illustrée les différentes méthodes d'entrelacement.

Par conséquent, le prochain chapitre est consacré à l'évaluation des performances des codes turbo.

Chapitre 3 : Simulations et résultats

3.1 Introduction

La simulation est un outil utilisé par le chercheur, l'ingénieur, le militaire etc… pour étudier les réponses d'un système sans réaliser l'expérience sur le système réel.

Lorsque l'outil de simulation utilise un ordinateur on parle de simulation numérique. Les chercheurs, les ingénieurs, les militaires et bien d'autres professionnels se posent souvent la question : quel est le résultat que j'obtiens si j'exerce telle action sur un système ? Le moyen le plus simple serait de tenter l'expérience, c'est-à-dire d'exercer l'action souhaitée sur le système en cause pour pouvoir observer ou mesurer le résultat. Dans de nombreux cas l'expérience est irréalisable, trop chère ou contraire à l'éthique. On a alors recours à la simulation : rechercher un modèle qui réagit d'une manière semblable à celui que l'on veut étudier et qui permettra de déduire les résultats.

On appelle modèle un système, analogique ou numérique, dont le comportement vis-à-vis d'un phénomène est similaire à celui du système à étudier. Les sorties sont les éléments que l'on veut étudier. Les entrées, paramètres et contraintes sont les éléments dont la variation influe sur le comportement du modèle ; on appelle entrée ceux qui sont commandés par l'expérimentateur, paramètres ceux que l'opérateur choisit de fixer et contraintes ceux qui dépendent d'éléments extérieurs. On appelle simulation l'ensemble constitué par un modèle, les ordres d'entrée, les paramètres et contraintes, et les résultats obtenus. Les équations sont des simulations numériques. Aujourd'hui ce terme s'applique essentiellement aux modèles et simulations réalisés sur ordinateur.

3.2 Méthodologie de simulation

Le code turbo est le seul type de code correcteur d'erreur qui se rapproche assez près de la limite de Shannon. Cependant, son comportement dépend de beaucoup de paramètres. Dans ce chapitre, nous allons présenter les étapes de conception d'une chaine de communication à l'aide du logiciel Simulink/Matlab.

Le système conçu est caractérisé par la possibilité de modifier : la longueur de contrainte de code CRS, la taille d'entrelaceur et le nombre d'itération de décodage itératif.

Nous allons analyser les performances de codage turbo sur deux types de canaux qui sont :

- Canal BBGA
- Canal de Rayleigh

Et aussi nous présentons :

- les effets de la taille d'entrelaceur aléatoire,
- l'influence de nombre d'itération,
- l'influence de la longueur de contrainte de code
- l'effet du canal de transmission sur la performance des codes turbo.

Nous effectuerons ensuite une comparaison des résultats de simulations obtenus. Nous utiliserons dans notre système, la modulation BPSK pour convertir les bits en sortie du codeur sous forme des symboles.

Tous les résultats de simulations présentés sont basés sur le décodage itératif en utilisant l'algorithme MAP.

3.2.1 Codes convolutifs utilisés

Nous avons utilisé dans notre simulation deux codes CRS [21] illustrés dans les figures 3.1. Le CRS de la figure 3.1.a contient quatre états et de rendement 1/2. Le code turbo parallèle associé à ce CRS est alors à quatre états et de rendement de 1/3 (Figure 3.2.a). Le CRS de la figure 3.1.b contient huit états et de rendement 1/2. Le code turbo parallèle associé à ce CRS est alors à huit états et de rendement 1/3 (Figure 3.2.b).

3.2.2 Entrelaceur utilisé

Le type d'entrelacement le plus performant à utiliser dans les codes turbo est celui d'un entrelaceur aléatoire [18] (voir chapitre 2).

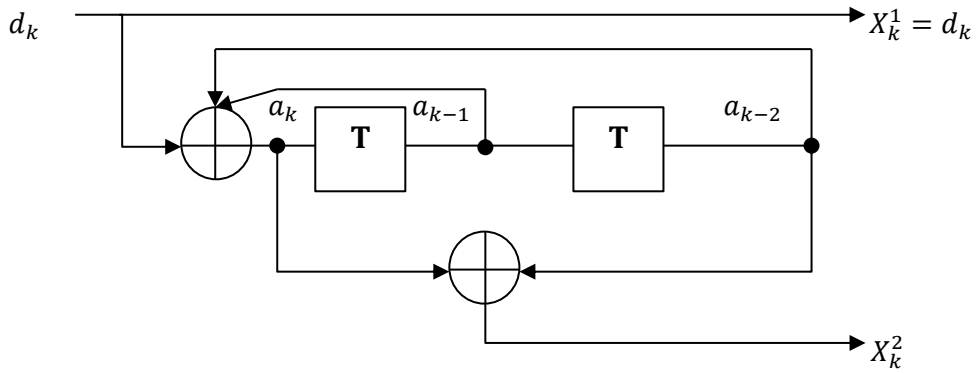

Fig. 3.1.a Code convolutif récursif systématique (5,7)

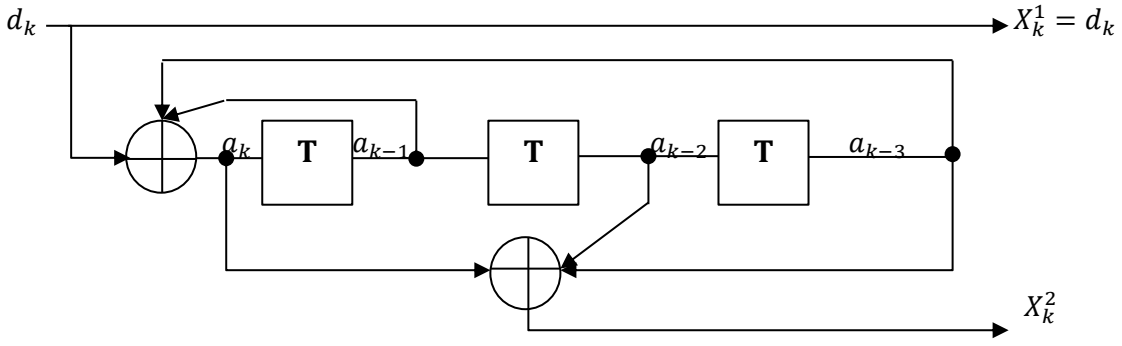

Fig. 3.1.b Code convolutif récursif systématique (13,15)

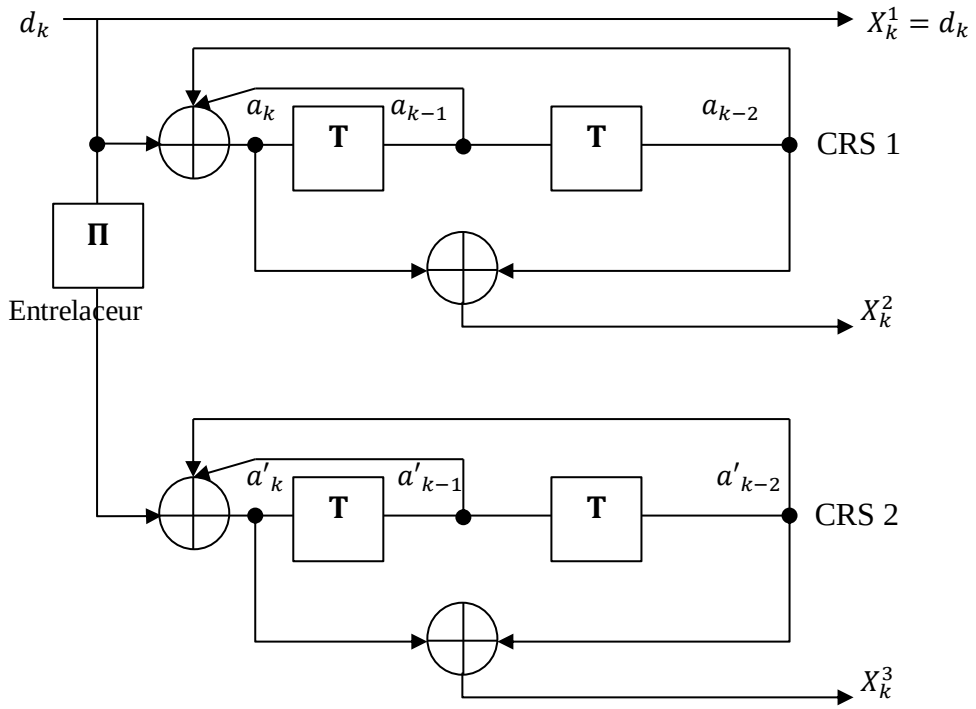

Fig. 3.2.a Code turbo (5,7)

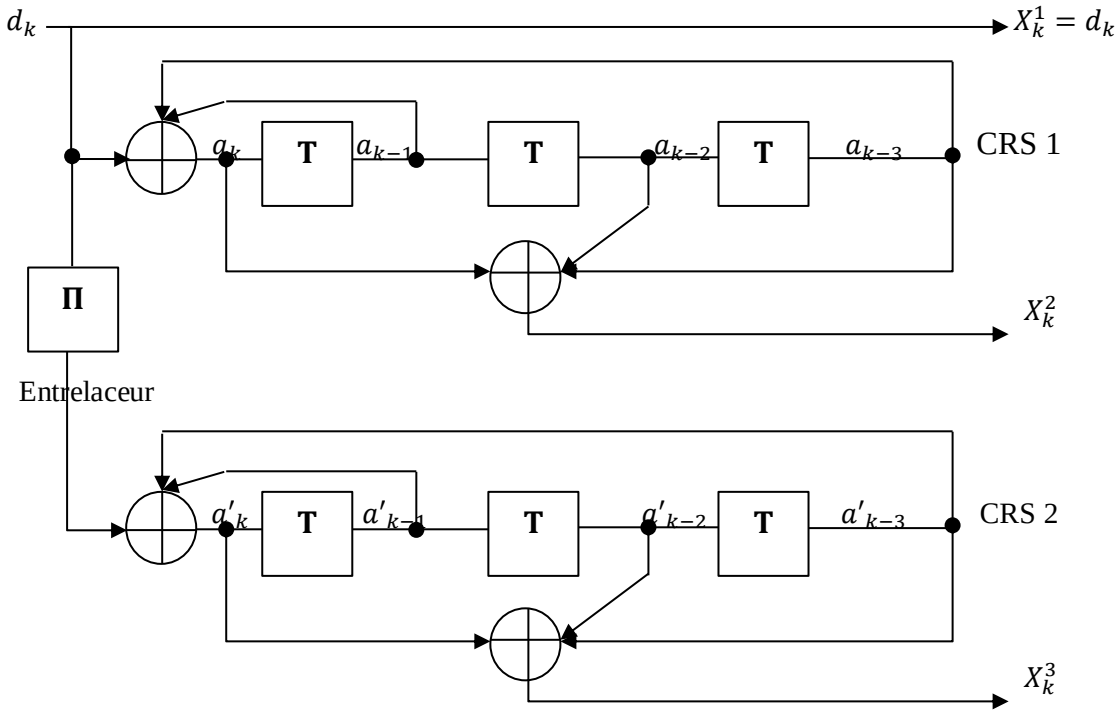

Fig. 3.2.b Code turbo (13,15)

3.3 Procédure de la simulation

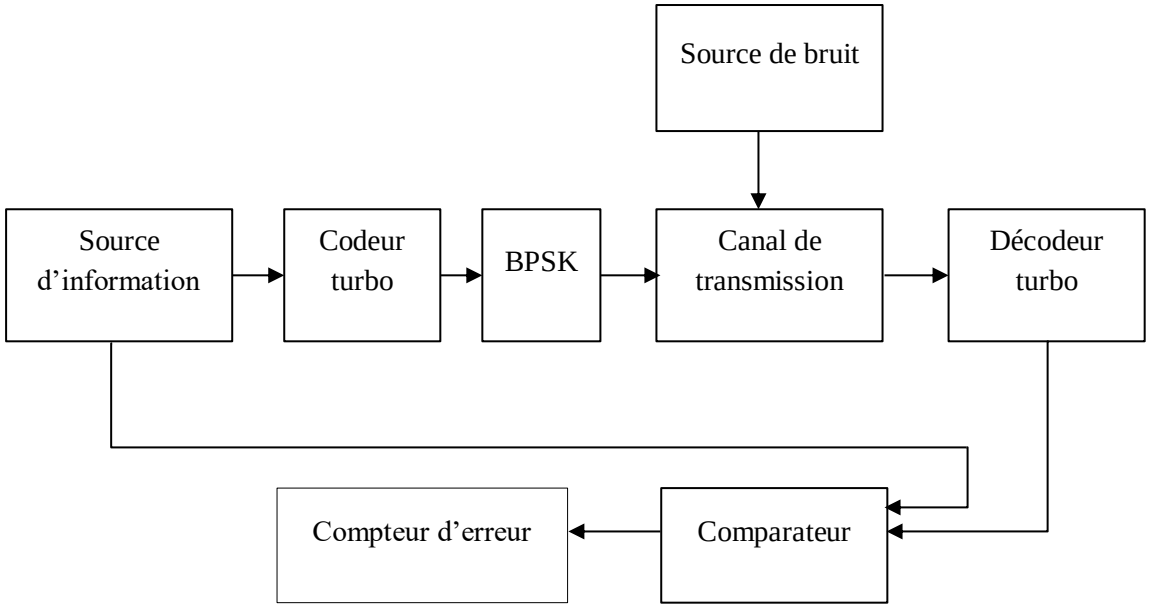

Fig. 3.3 – Modèle de simulation

1. Générer aléatoirement les bits d'information.
2. Encoder les bits d'information par le code turbo, à chaque étape de simulation nous changeons le code CRS utilisé pour construire le code turbo.
3. Utiliser la modulation BPSK pour convertir la donnée à la sortie de code turbo, en signaux complexes.
4. Introduire un bruit pour simuler les erreurs sur le canal de transmission. En première temps nous supposons que les signaux sont transmis sur un canal BBGA puis sur un canal de rayleigh.
5. A la réception, les opérations inverses sont effectuées pour démoduler et décoder la séquence des symboles reçues.
6. Comptez le nombre des bits erronés en comparant la séquence des bits décodés avec la séquence des bits transmis.
7. Calculer le Taux d'Erreurs Binaire (TEB).

Le système proposé doit améliorer le Taux d'Erreur Binaire, donc la simulation et l'évaluation de TEB pour différente valeur de E_b/N_0 est nécessaire. La simulation est initialisée avec une valeur de $E_b/N_0 = 0\ dB$. La valeur de $E_b/N_0\ dB$ est augmentée après chaque simulation par $0{,}5\ dB$.

La simulation se fait en exécutant le modèle Matlab/Simulink pour chaque valeur de $E_b/N_0\ dB$ et en calculons la valeur de TEB. Ensuite les valeurs de TEB correspondant aux

différentes simulations sont stockées dans l'espace de travail de MATLAB, puis les graphes représentant TEB en fonction $E_b/N_0\ dB$ sont tracés.

3.4 Les résultats de la simulation sur le canal BBGA

La figure 3.4 présente le modèle de simulation utilisé dans l'étude de code turbo sur un canal BBGA.

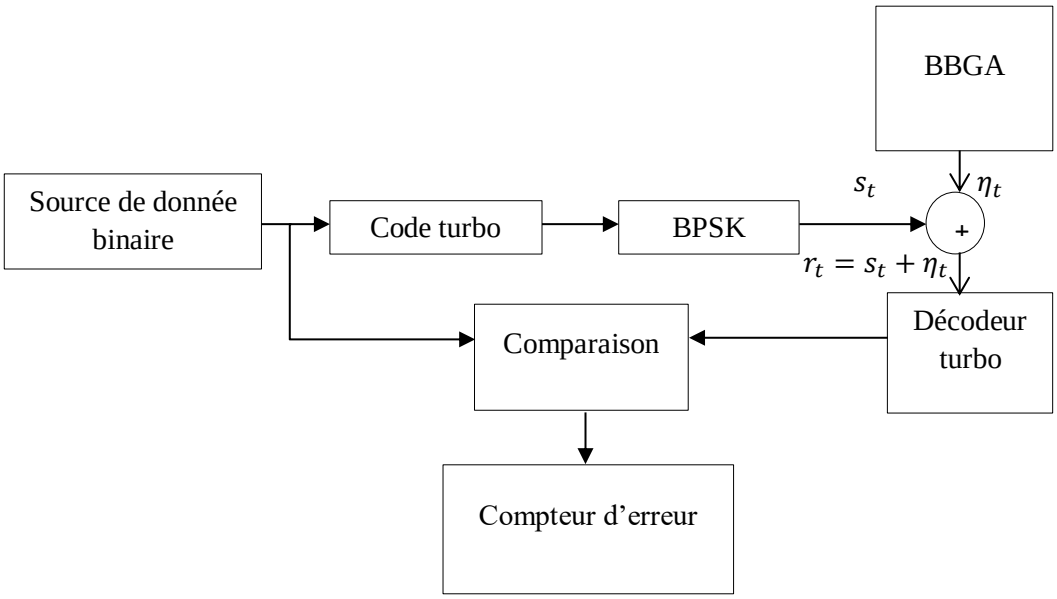

Fig. 3.4 – Modèle de simulation pour le canal BBGA

3.4.1 Effet de la taille de l'entrelaceur sur les performances des codes turbo

En utilisant les performances de taux d'erreur binaire en fonction du rapport E_b/N_0, Il est possible de comparer les performances d'un entrelaceur aléatoire pour plusieurs tailles d'entrelacement N. E_b détermine l'énergie émise par bit et N_0 mesure la densité spectrale de puissance de bruit. Les résultats de simulation de Taux d'Erreur Binaire (TEB) sont donnés par les figures 3.5 et 3.6 pour trois longueurs d'entrelacement : 64 bits, 256 bits et 1024 bits.

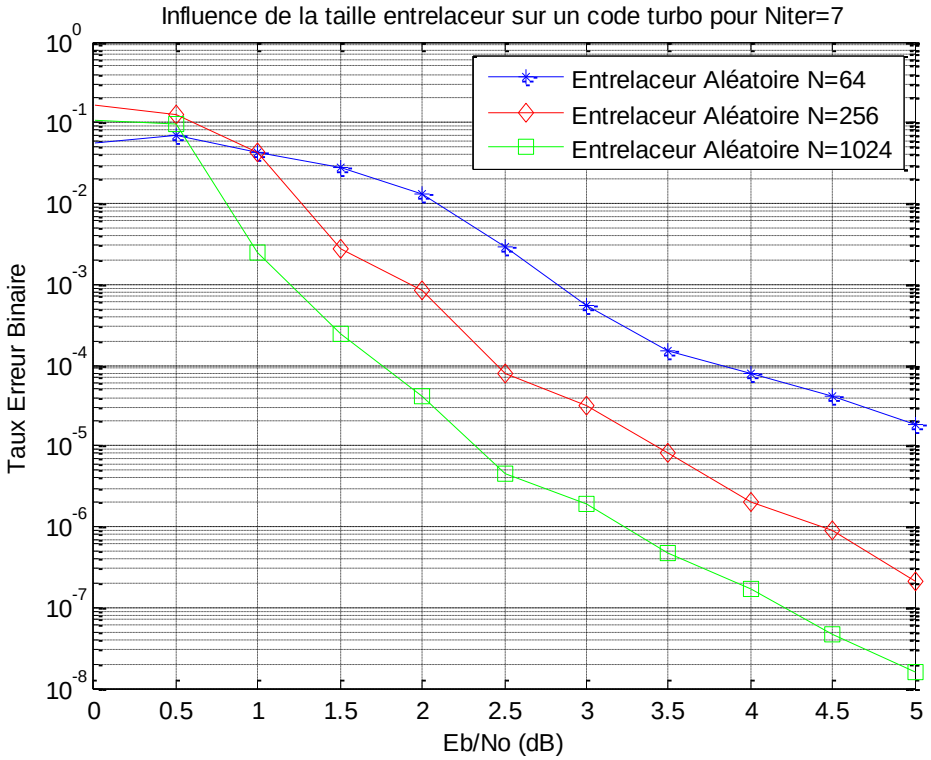

Fig. 3.5 - Influence de la taille de l'entrelaceur aléatoire sur les performances des codes turbo dans un canal BBGA avec K=3, G= (1,7/5) et R=1/3 Niter=5

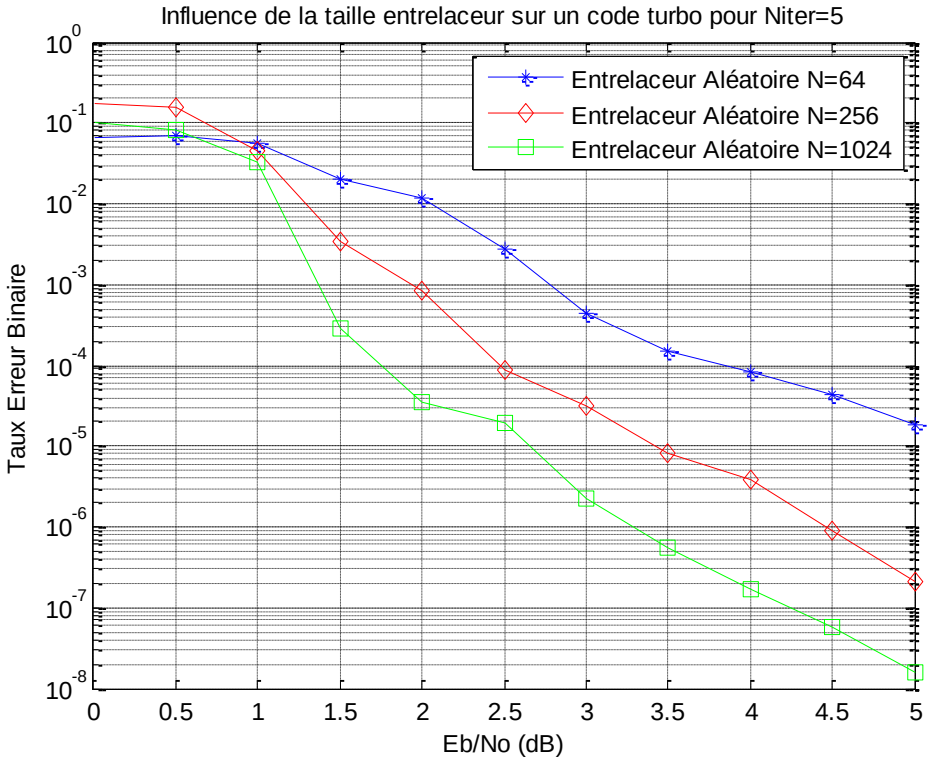

Fig. 3.6 - Influence de la taille de l'entrelaceur aléatoire sur les performances des codes turbo dans un canal BBGA avec K=3, G= (1,7/5) et R=1/3 Niter=7

Interprétation

La taille d'entrelaceur est un paramètre très critique pour améliorer la performance des codes turbo. Elle affecte les propriétés de la distance libre du code utilisé. Si on augmente la taille d'entrelaceur, la corrélation entre les bits adjacents entrelacés sera diminuée, donc les performances de décodage augmentent. Les résultats de la simulation vérifient cette conclusion.

Les résultats des simulations montrent bien qu'une augmentation de la taille des entrelaceurs engendre une amélioration de la performance de taux d'erreur binaire. On constate aussi que les performances des différents nombre d'itération évoluent de la même manière.

En comparant les résultats des simulations présentées sur les figures ci-dessus, on remarque clairement que l'efficacité énergétique des entrelaceurs augmente en augmentent la taille d'entrelaceur. Donc pour atteindre le même TEB, nous avons besoin de moins de puissance de transmission à l'émetteur en utilisant des entrelaceurs avec N élevé.

Si on prend l'exemple où le nombre d'itération Niter=7 (figure 3.6) :

La taille d'entrelaceur aléatoire	(E_b/N_0) dB Requise
N= 64 bits	4.75 dB
N= 256 bits	3.15 dB
N= 1024 bits	2.15 dB

Tableau 3.1 - $\boldsymbol{E_b/N_0}$ requis pour assurer un TEB = 2.10^{-5}

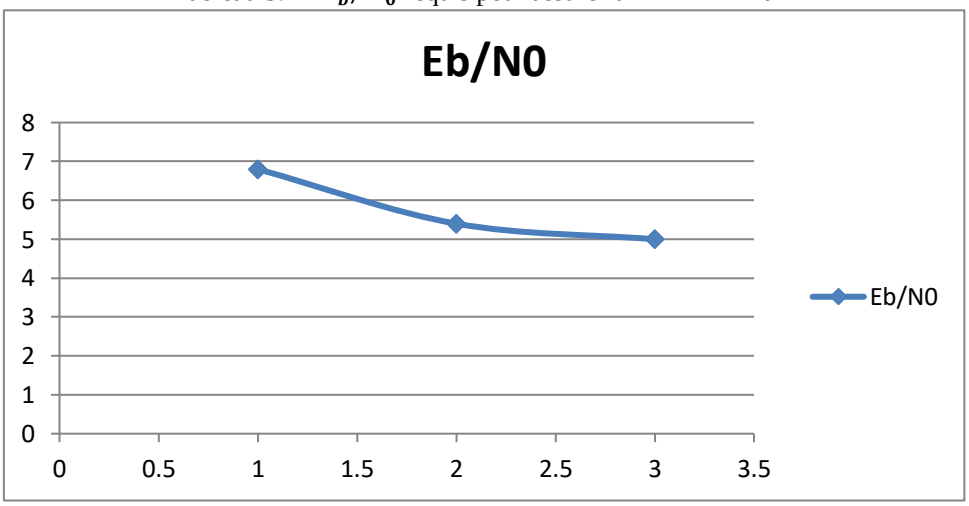

Fig. 3.7 –Effet de la taille d'entrelacement sur $\boldsymbol{E_b/N_0}$ **pour assure** un TEB = 2.10^{-5}

Les résultats de tableau 3.1 et la figure 3.7 montrent que pour des faibles rapports signal sur bruit, la taille de l'entrelaceur joue un rôle très important pour diminuer la probabilité

d'erreur. On peut conclure que plus la taille d'entrelacement est élevée plus le code turbo se rapproche de la limite de Shannon.

3.4.2 Effet du nombre d'itérations sur les performances des codes turbo

Considérons le même un code turbo de taux de codage R=1/3 avec un vecteur polynôme générateur G= (1,7/5) et l'entrelaceur aléatoire. Pour mesurer l'impact de nombre d'itération sur les performances des codes turbo, nous faisons varier le nombre d'itération de 1, 3, 5, ensuit 7.

Les figures 3.8 à 3.10, permettent d'évaluer l'influence de nombre d'itération sur la performance TEB de code turbo avec un entrelaceur aléatoire. Les résultats obtenus, montrent parfaitement que l'augmentation de nombre d'itération améliore les performances du codage turbo.

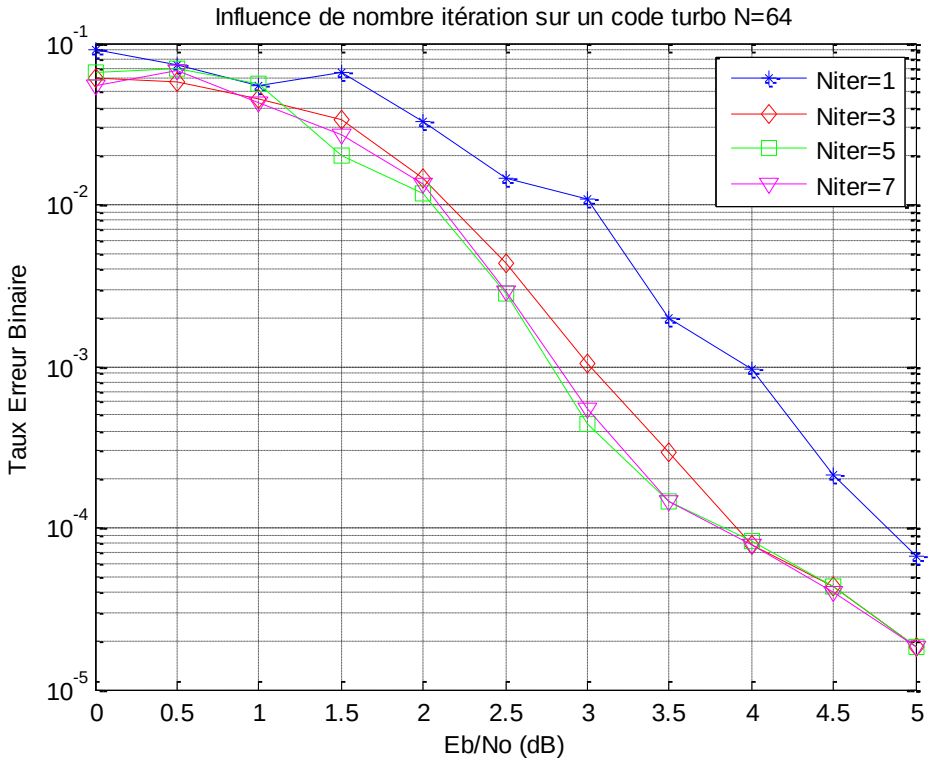

Fig. 3.8 - Influence de nombre d'itération sur les performances des codes turbo dans un canal BBGA avec K=3, G= (1,7/5), R=1/3 et la taille d'entrelaceur aléatoire N= 64 bits

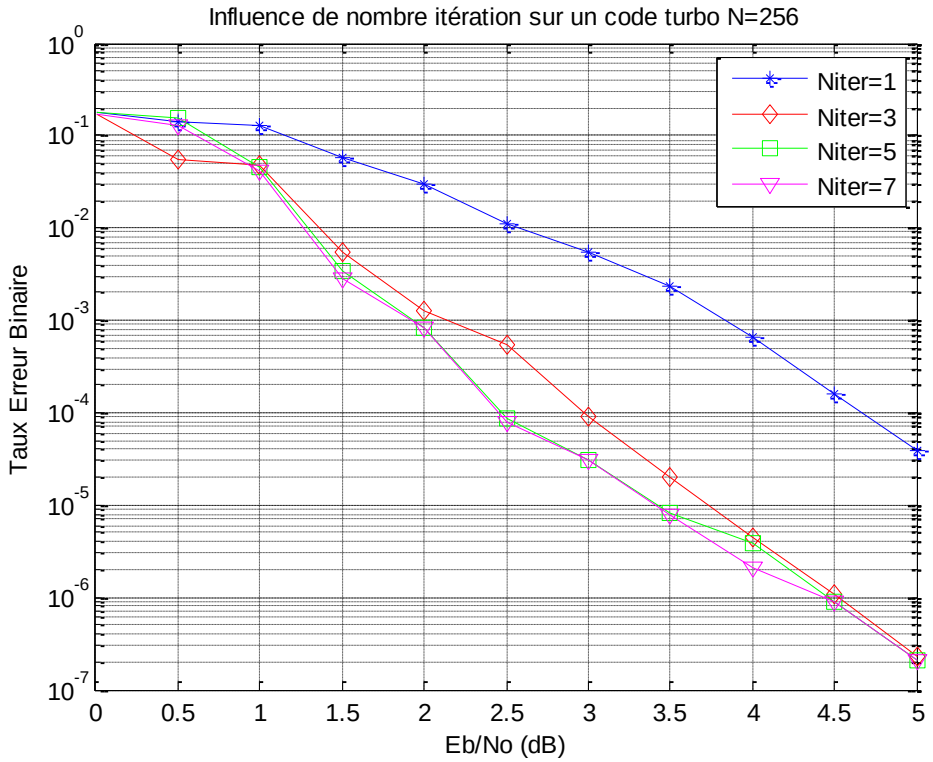

Fig. 3.9 – Influence de nombre d'itération sur les performances des codes turbo dans un canal BBGA avec K=3, G= (1,7/5), R=1/3 et la taille d'entrelaceur aléatoire N= 256 bits

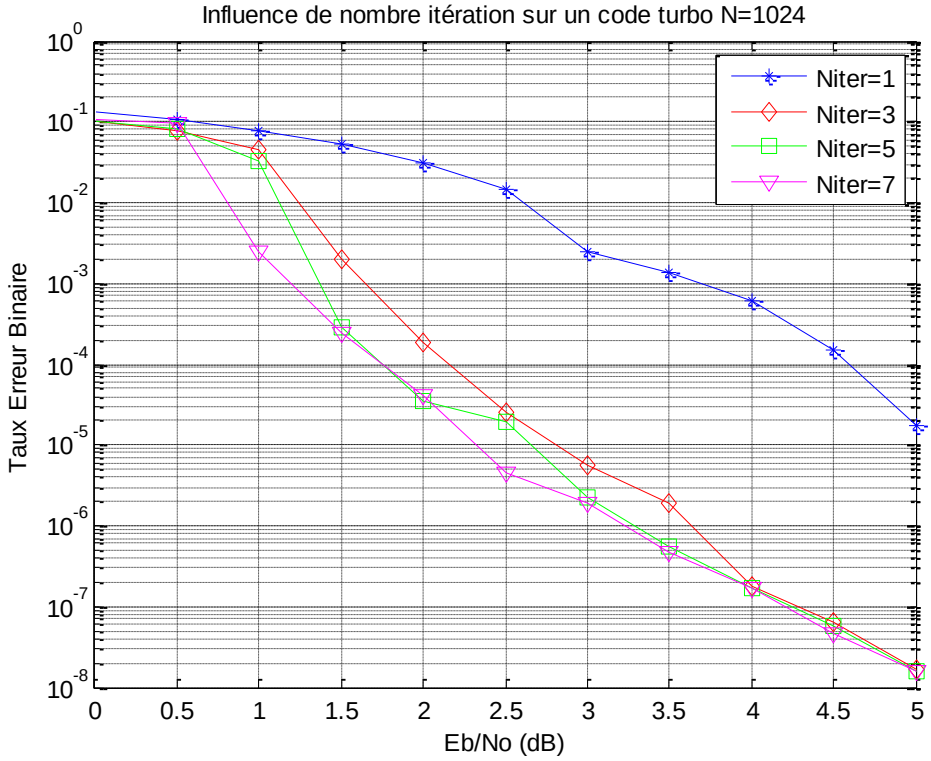

Fig. 3.10 - Influence de nombre d'itération sur les performances des codes turbo dans un canal BBGA avec K=3, G= (1,7/5), R=1/3 et la taille d'entrelaceur aléatoire N= 1024 bits

Interprétation

Les figures aussi montrent que pour quelque taille, qu'on ne devrait pas augmenter indéfiniment le nombre d'itération pour avoir des meilleurs résultats. En effet, après la cinquième itération, augmenter le nombre d'itérations ne donne pas un gain important sur les performances, mais par contre le système devient complexe.

Prenant l'exemple de la taille d'entrelacement N= 256 (figure 3.9), une probabilité d'erreur par bit TEB= 10^{-5} dB est atteinte pour un rapport signal sur bruit E_b/N_0 = 3.45 dB après 5 itération de décodage. Le passage de 5 à 7 itérations apporte un gain négligeable, pour une complexité de décodage doublée.

3.4.3 Effet de la longueur de contrainte sur les performances des codes turbo

Pour montrer l'effet de la longueur de contrainte sur les performances de codage turbo, nous avons utilisé deux codes turbo : le premier avec une longueur de contrainte k = 3 et un polynôme générateur (7,5) et le deuxième avec une longueur de contrainte k = 4 et un polynôme générateur (13,15).

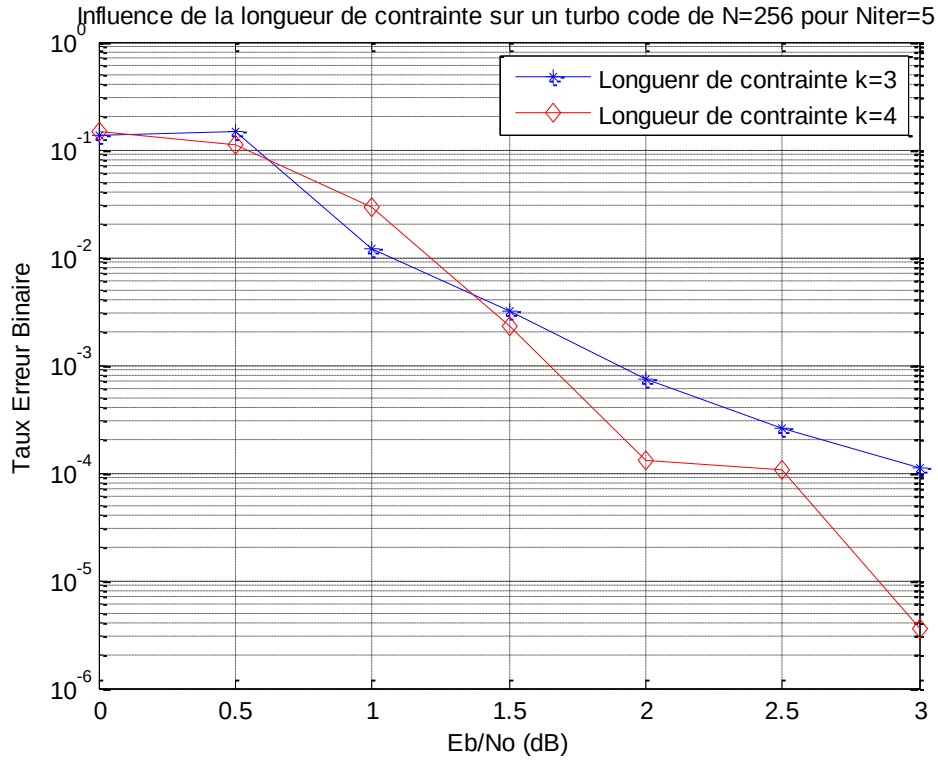

Fig. 3.11 - Influence de la longueur de contrainte sur les performances des codes turbo dans un canal BBGA avec R=1/3, la taille d'entrelaceur aléatoire N= 256 bits et un nombre d'itération= 5

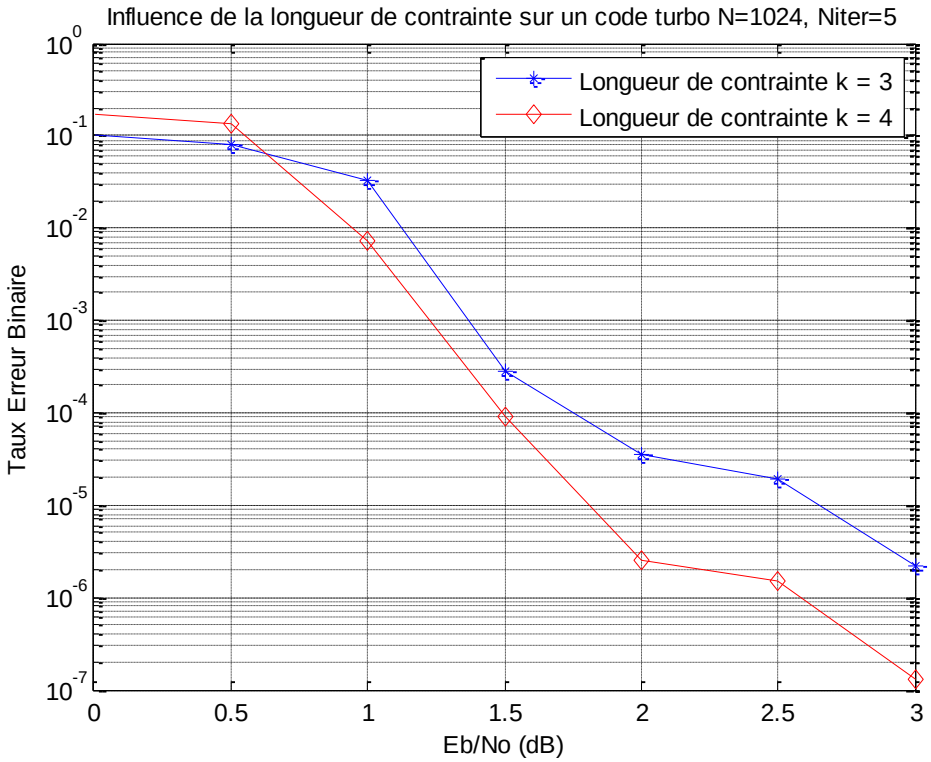

Fig. 3.12 - Influence de la longueur de contrainte sur les performances des codes turbo dans un canal BBGA avec R=1/3, la taille d'entrelaceur aléatoire N= 1024 bits et un nombre d'itération= 5

Interprétation

Les figures 3.11 et 3.12 montrent que pour un taux de codage $1/3$, pour de faibles rapports signal sur bruit, il semble que l'utilisation de code turbo avec une longueur k = 4 n'apporte pas une grande amélioration par rapport au code turbo avec une longueur de contrainte k = 3. Des gains faibles sont constatés. Cela n'est donc pas nécessaire de compliquer le processus de codage. Toutefois, nous notons un certain gain pour de plus grands rapports E_b/N_0.

Les figures présentent aussi une comparaison. Prenant l'exemple de la taille d'entrelacement N=256 (figure 3.11), un gain est constaté pour un rapport signal sur bruit $E_b/N_0 = 1.5\ dB$, par contre pour une taille d'entrelacement N=1024 (figure 3.12), le gain est constaté pour $E_b/N_0 = 0.65\ dB$. Donc nous pouvons affirmer que pour des tailles d'entrelacement faible, les codes turbo ont des résultats qui ne présentent pas une grande amélioration par rapport aux résultats pour des tailles d'entrelacement grandes.

3.5 Les résultats de la simulation sur le canal de Rayleigh

Le modèle du canal de Rayleigh utilisé dans les simulations que nous allons analyser est présenté au figure 3.13. Les performances en termes de probabilité d'erreur par bit du code turbo dans le canal de Rayleigh sont obtenues par simulations en utilisant un code turbo avec un taux de codage 1/3, une longueur de contrainte de valeur 3, et un vecteur générateur G = (7,5). Sauf pour la simulation de l'effet de la longueur de contrainte en utilisant un turbo code

turbo avec un taux de codage 1/3, une longueur de contrainte de valeur 4, et un vecteur générateur G=(13,15).

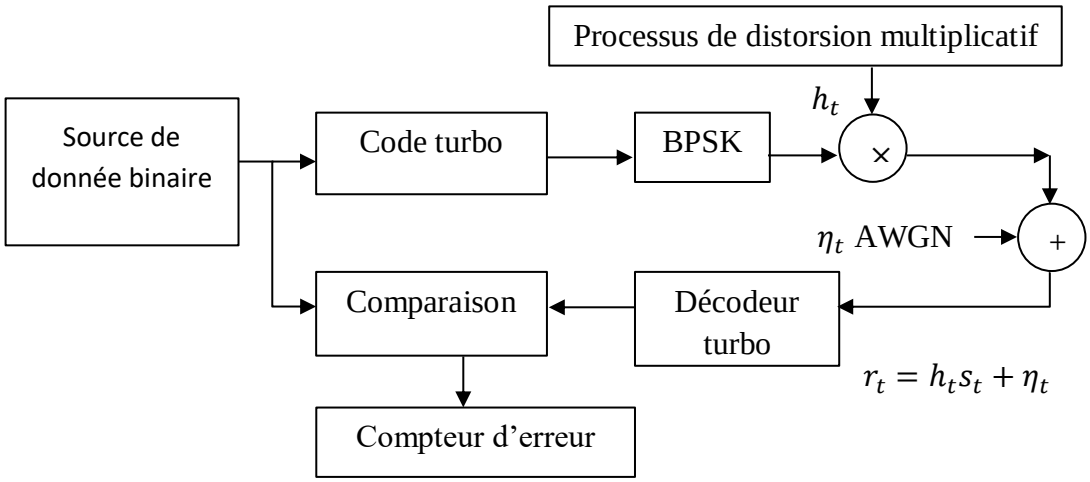

Fig. 3.13 -Modèle de simulation pour le canal de Rayleigh

3.5.1 Effet de la taille de l'entrelaceur sur les performances des codes turbo

Pour mesurer l'effet de la taille de l'entrelaceur aléatoire, nous fixons le nombre d'itération nous faisons varier la taille d'entrelacement. Nous utilisons d'abord 5 itérations puis 7 itérations.

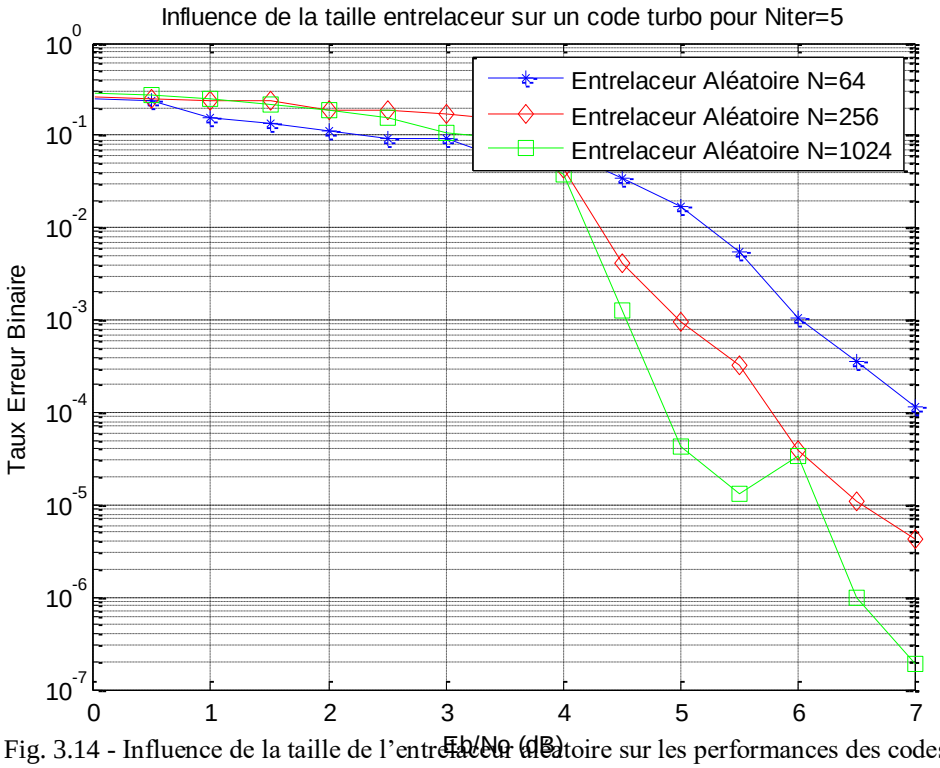

Fig. 3.14 - Influence de la taille de l'entrelaceur aléatoire sur les performances des codes turbo dans un canal de Rayleigh avec K=3, G= (1,7/5), R=1/3et Niter=5

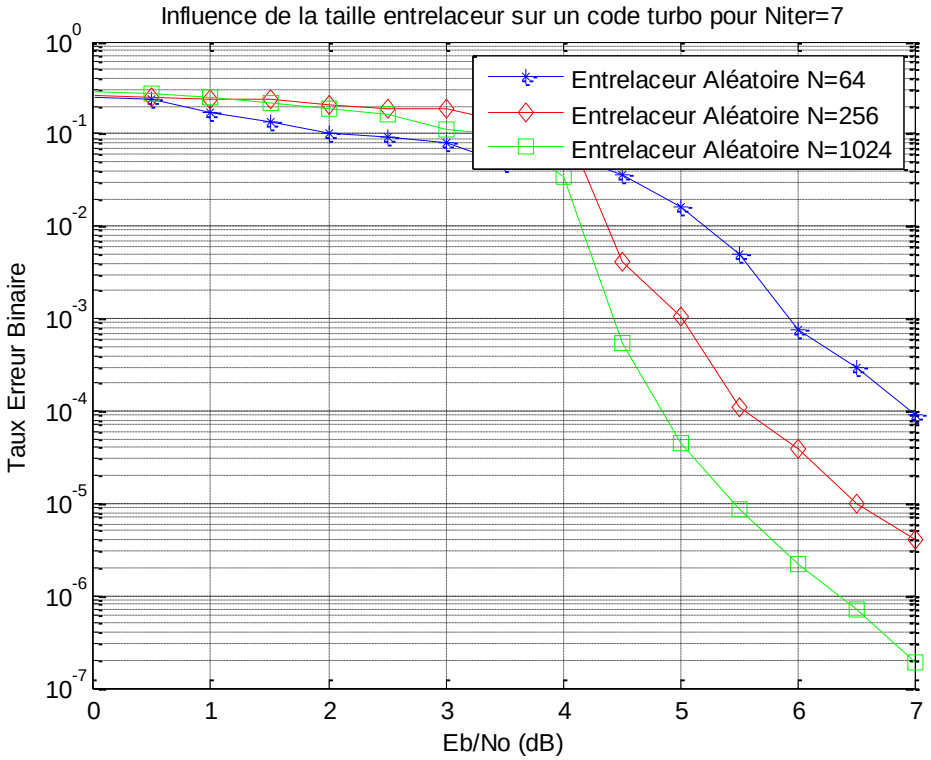

Fig. 3.15 - Influence de la taille de l'entrelaceur aléatoire sur les performances des codes turbo dans un canal de Rayleigh avec K=3, G= (1,7/5), R=1/3 et Niter=7

Interprétation

Les figures 3.14 et 3.15 présentent les performances du système de transmission sur un canal de Rayleigh en utilisant un entrelaceur aléatoire avec un codage de canal de type turbo, un taux de codage égale à 1/3, le décodage est fait en utilisant l'algorithme de MAP avec un nombre d'itération égale à 5 (figure 3.14) et un nombre d'itération égale à 7 (figure 3.15). Par rapport aux résultats obtenus sur le canal BBGA, on voit que les performances se trouvent dégradées. En effet, les performances obtenues avec le canal gaussien sont meilleures que celles obtenues avec le canal Rayleigh et ceci est dû à la présence d'un bruit plus importante sur le canal de Rayleigh.

La probabilité d'erreur est tracée pour trois longueurs d'entrelacement : 64 bits, 256 bits et 1024 bits en fonction de E_b/N_0. Les remarques citées dans le cas du canal BBGA pour un taux de codage R = 1/3 et K = 3 s'appliquent aussi dans le cas de Rayleigh. D'après les résultats obtenus, on constate que pour conserver une probabilité d'erreur par bit constante lorsque la taille d'entrelaceur diminue, il faut augmenter le rapport E_b/N_0.

Les résultats de simulation donnée par les figures 3.14 et 3.15 montrent aussi que lorsque la taille de l'entrelaceur augmente, les performances en termes de TEB s'améliorant. Ainsi, quand le rapport signal sur bruit est faible (entre 0 et 4 dB), l'influence de la taille d'entrelacement sur les performances de code turbo est faible. Cette influence devient importante pour des rapports signal sur bruit fort (supérieur à 4 dB).

Prenant l'exemple de le nombre d'itération égale à 7 (figure 3.15), pour N=256 bits, le rapport E_b/N_0 pour assurer une probabilité d'erreur $TEB = 10^{-5}$ est 6.5 dB, tandis qu'en utilisant une longueur =1024 bits, nous avons besoin que de 5.5 dB, donc nous avons un gain égale à 1dB.

3.5.2 Effet du nombre d'itérations sur les performances des codes turbo

Les courbes des figures 3.16 à 3.18 montrent bien l'évolution du TEB en faisant varier le nombre d'itération. Ces résultats ont été obtenus en considérant un code turbo fournis par deux codeurs récursifs et systématiques de longueur de contrainte K=3 et de polynôme générateur [7, 5].

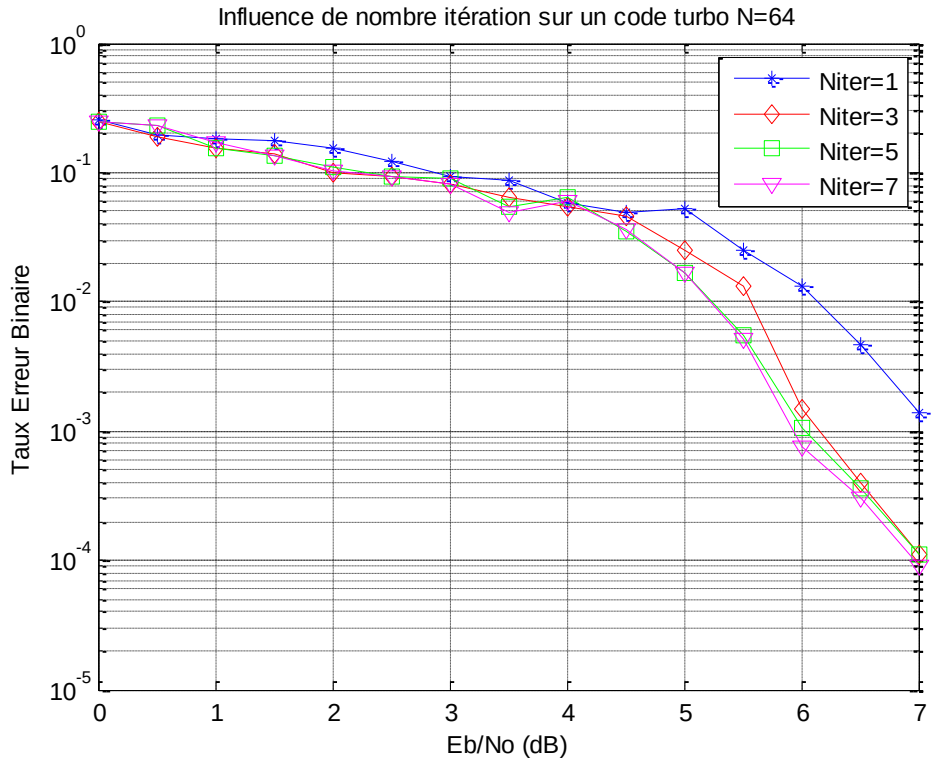

Fig. 3.16- Influence de nombre d'itération sur les performances des codes turbo dans un canal de Rayleigh avec K=3, G= (1,7/5), R=1/3 et la taille d'entrelaceur aléatoire N= 64 bits

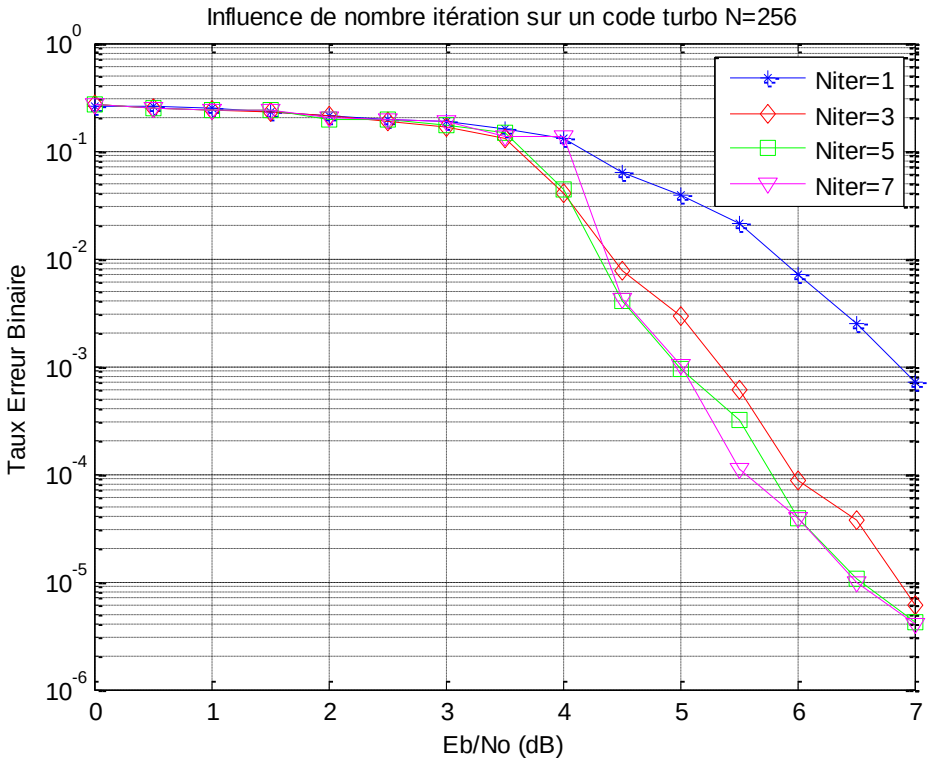

Fig. 3.17 - Influence de nombre d'itération sur les performances des codes turbo dans un canal de Rayleigh avec K=3, G= (1,7/5), R=1/3 et la taille d'entrelaceur aléatoire N= 256 bits

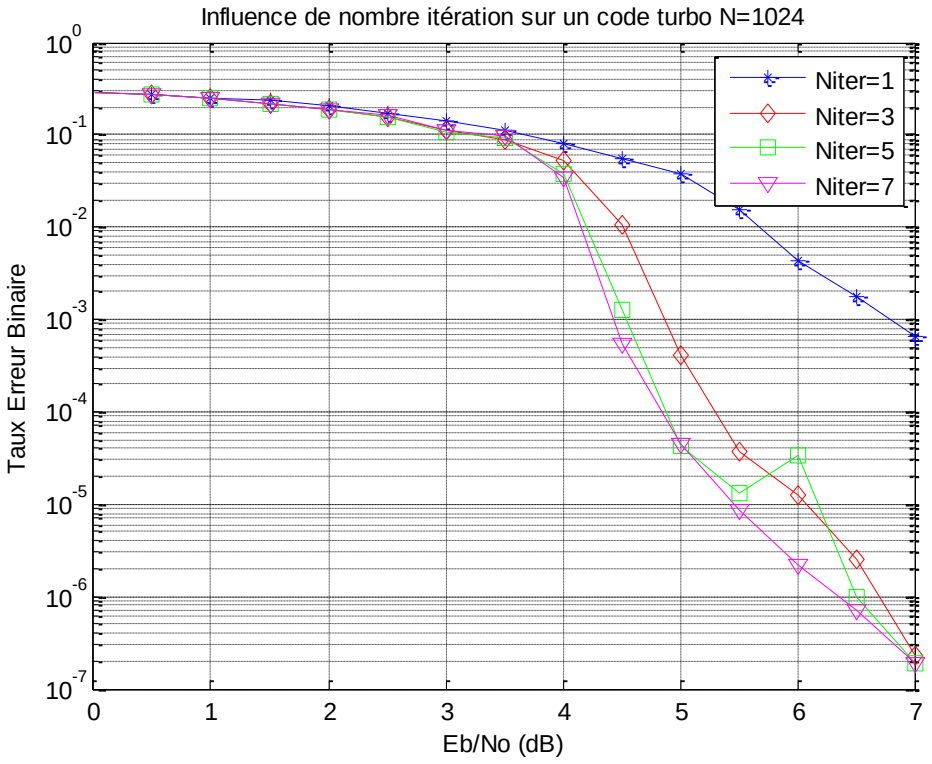

Fig. 3.18 - Influence de nombre d'itération sur les performances des codes turbo dans un canal de Rayleigh avec K=3, G= (1,7/5), R=1/3 et la taille d'entrelaceur aléatoire N= 1024 bits

Interprétation

On peut remarquer que le TEB décroit avec l'augmentation de nombre d'itération de processus du décodage itératif. En effet, dans ce cas, le pouvoir de correction augmente et le code est capable de corriger plus d'erreurs.

On remarque aussi qu'à partir d'un certain nombre d'itération, l'augmentation de ce nombre ne garantit pas un gain en puissance. Mais par contre, la complexité et la latence du décodeur turbo augmente.

Prenant l'exemple de la taille de l'entrelaceur aléatoire N=256 (figure 3.17), le tableau 3.2 présente le rapport E_b/N_0 requis pour assurer un $TEB = 10^{-3}$, en fonction du nombre d'itération.

On remarque qu'après la cinquième itération, les performances restant les mêmes.

Nombre d'itération	1	3	5	7
E_b/N_0 requis pour assurer un $TEB = 10^{-3}$	6.8 dB	5.4 dB	5 dB	5 dB

Tableau 3.2 - E_b/N_0 requis par un code turbo pour assurer un $TEB = 10^{-3}$ sur un canal de Rayleigh pour différent nombre d'itération.

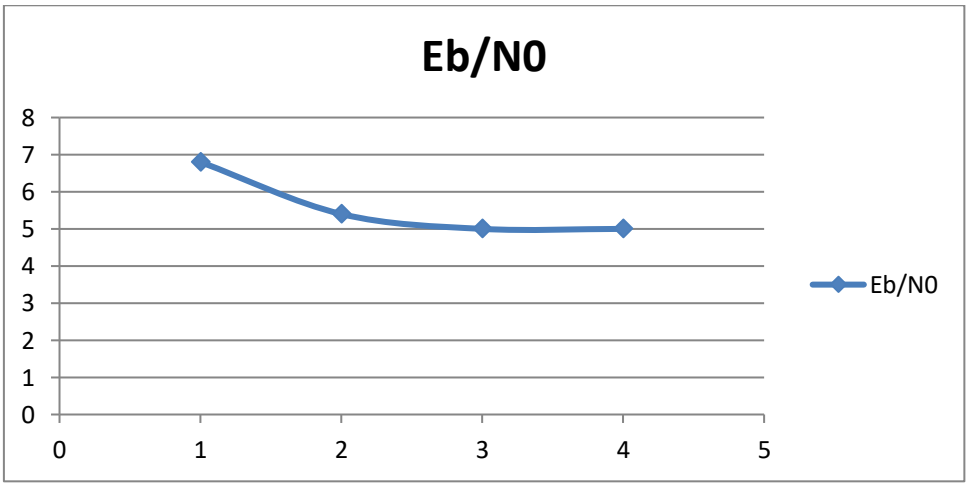

Fig. 3.19 –Effet du nonmbre d'itérartion $\boldsymbol{E_b/N_0}$ **pour assure** un TEB = 2.10^{-5}

3.5.3 Effet de la longueur de contrainte sur les performances des codes turbo

Pour montrer l'effet de la longueur de contrainte sur les performances de codage turbo, nous avons utilisé deux codes turbo : le premier avec une longueur de contrainte k = 3 et un polynôme générateur (7,5) et le deuxième avec une longueur de contrainte k = 4 et un polynôme générateur (13,15).

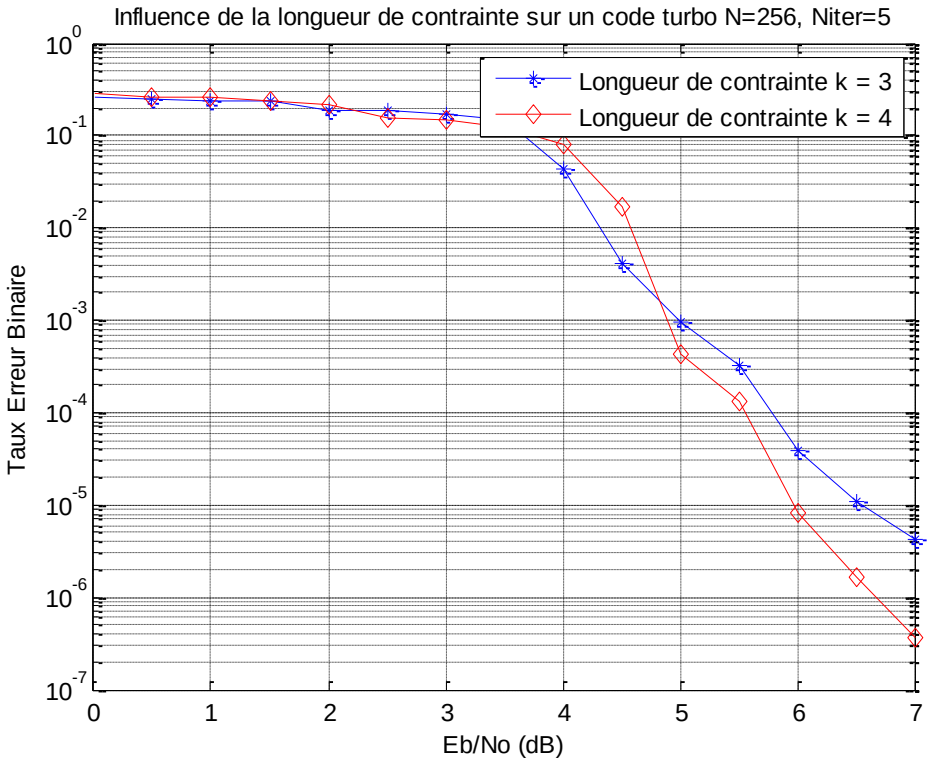

Fig. 3.20 - Influence de la longueur de contrainte sur les performances des codes turbo dans un canal de Rayleigh avec R=1/3, la taille d'entrelaceur aléatoire N= 256 bits et un nombre d'itération= 5

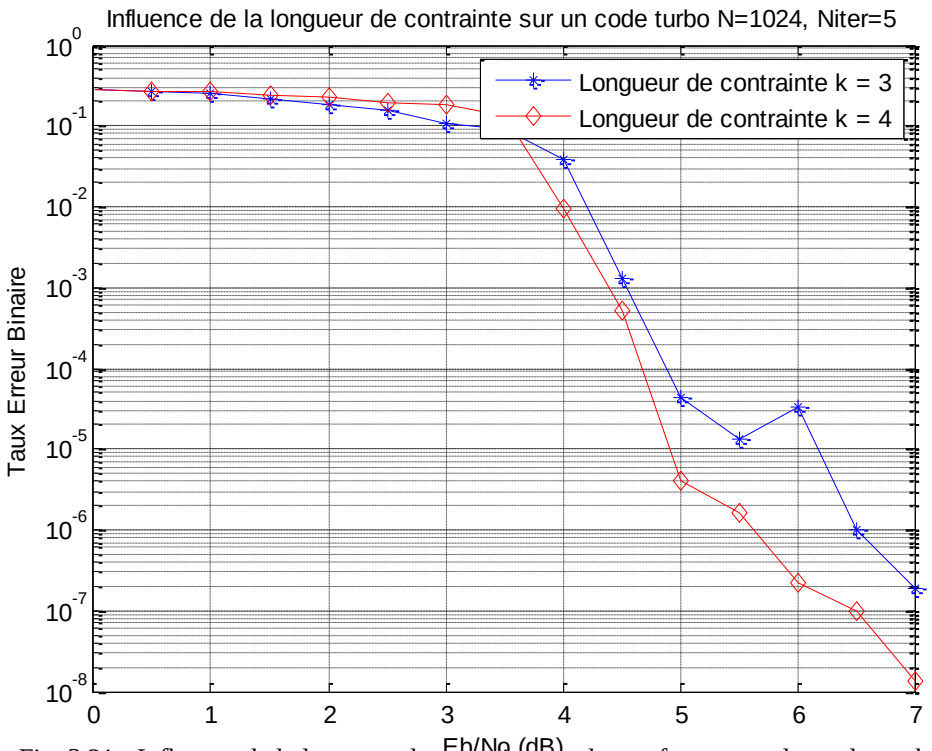

Fig. 3.21 - Influence de la longueur de contrainte sur les performances des codes turbo dans un canal de Rayleigh avec R=1/3, la taille d'entrelaceur aléatoire N= 1024 bits et un nombre d'itération= 5

Interprétation

D'après les résultats obtenus, présentés par les figures 3.20 et 3.21, on constate que pour des faibles rapports signal sur bruit, il semble que l'utilisation de code turbo avec une longueur k= 4 n'apporte pas une grande amélioration par rapport au code turbo avec une longueur de contrainte k = 3. Donc le gain obtenu en augmentant la longueur de contrainte devient faible. Cependant la complexité et le délai de décodage augmentent.

Prenant l'exemple où la taille de l'entrelaceur aléatoire N= 1024 (figure 3.21), pour k = 3 le rapport $E_b/N_0 = 2\ dB$ assure une probabilité d'erreur $TEB = 1.9 \times 10^{-1}$ plus proche de celle dans le cas k = 4. Par contre le rapport $E_b/N_0 = 5\ dB$ pour k = 4 assure une probabilité d'erreur $TEB = 4 \times 10^{-6}$ est grand que dans le cas k = 3 ($TEB = 4 \times 10^{-5}$).

3.6 Comparaison entre l'effet de canal BBGA et canal de Rayleigh

Pour montrer la différence entre le comportement du code turbo sur les canaux BBGA et Rayleigh, nous avons comparés les résultats obtenus pour un code turbo avec un taux de codage $R = 1/3$, une longueur de contrainte k = 3 et un polynôme générateur (7,5), avec un entrelaceur aléatoire de taille N=1024 bits.

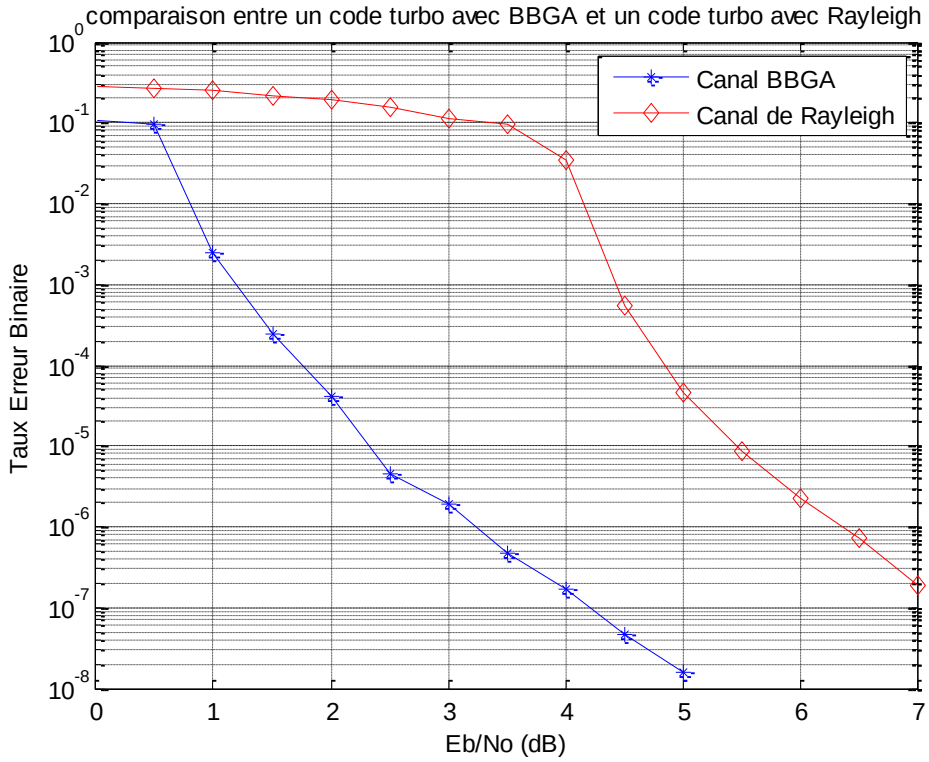

Fig. 3.22 – Comparaison entre un code turbo avec un canal BBGA et un code turbo avec un canal de Rayleigh avec K=3, G= (1,7/5), R=1/3 et la taille d'entrelaceur aléatoire N= 1024 bits

Interprétation

Type de canal	Canal BBGA	Canal de Rayleigh
E_b/N_0 requis pour assurer un $TEB = 10^{-6}$	3.3 dB	6.45 dB

Tableau 3.3 - E_b/N_0 requis par un code turbo pour assurer un $TEB = 10^{-6}$ pour deux types de canaux

D'après la figure 3.22 et tableau 3.3, on remarque que le TEB est relativement faible pour un canal BBGA. Ceci s'explique le faible bruit présent sur le canal BBGA.

3.7 Conclusion

Dans ce chapitre, nous avons étudié l'influence de nombre d'itération de décodage, l'influence de la longueur de contrainte, l'influence de la taille d'entrelacement et l'influence de canal de transmission sur les performances de codage turbo.

Les résultats de simulation montrent que la performance d'un code turbo augmente au fur et à mesure que la taille d'entrelacement et le nombre d'itération augmente quel que soit le type de canal.

Les simulations monteront qu'un code turbo avec une longueur de contrainte k = 4 donne des meilleurs résultats que celles obtenus avec un code turbo k = 3. Une comparaison entre le canal BBGA et le canal de Rayleigh, montre l'importance d'utilisé le codage turbo sur les canaux bruité en raison du niveau de gain obtenu.

Conclusion générale

Selon le cahier de charge exposé, notre travail concernant l'étude du turbo codage en utilisant le MAP, a englobé deux parties : une partie théorique et l'autre une simulation.
La partie théorique a pour but de dégager les fondements théoriques de ces codeurs de canaux, à s'avoir : les théorèmes fondamentaux du codage de canal, le principe des turbo codes convolutifs, la notion de l'information extrinsèque et le principe de décodage itératif, la notion de l'algorithme MAP, et de dégager aussi les autres fonctions d'un système de communications numérique, comme la modulation BPSK.
Le deuxième partie était la mise en œuvre d'une simulation permettant d'évaluer les performances des turbo décodeurs, et cela en estimant le taux d'erreurs binaire BER. Le décodage étant réalisé à l'aide d'un décodeur itératif basé sur le MAP. Nous avons étudié les entrelaceurs.

L'étude des performances des codes turbo a commencé au chapitre 3. Nous avons commencé l'étude avec un code turbo avec un entrelaceur aléatoire. Nous avons également pu constater que les performances étaient améliorées ou fur et à mesure que nous augmentions la taille d'entrelacement, ainsi que d'autres paramètres de code turbo à ajusté, comme, le choix de la longueur de contrainte de code. Par la suite nous avons étudié l'influence de le type de canal de communication tel que BBGA ou Rayleigh, afin d'obtenir des meilleur performances.

Bibliographies

[1] Adriana Dinu, Joëlle Barral, '' Codes correcteurs d'erreurs: implémentation des turbo codes ''.

[2] Afif Hani Osseiran, '' On the decoding of turbo codes '', mémoire de maitrise, université de Montréal, Octobre 1999.

[3] Khaled Lajnef, '' Etude des performances des codes turbo '', mémoire de maitrise, école polytechnique de Montréal, Juin 2001.

[4] J. Oswald, '' Théorie de l'Information ou Analyse Diacritique des systèmes '', Ed. Masson, 1986.

[5] A. Poli et Li. Huguet, '' Codes Correcteurs, Théorie et Applications '', Ed. Masson, 1989.

[6] BOUTAGHANE Massine et AOUDIA Hakim, '' ÉTUDE DE MODÈLES DE CANAUX MIMO STOCHASTIQUES '', Université Abderrahmane Mira, Faculté de Technologie , Département de génie électrique, Béjaia, juin 2014.

[7] Pierre Senellart, '' Codes correcteurs d'erreurs la révolution des turbo-codes '', Télécom Paris Tech, Département Informatique et Réseaux, Paris Cedex 13 France, 2007.

[8] Haisheng Liu, ''Contributions à la maîtrise de la consommation dans des turbo-décodeurs'', TELECOM, l'Université de Bretagne-Sud, Bretagne, 1 juillet 2009.

[9] Chemsa Ali, Saigaa Djamel, Taleb-Ahmed Abdelmalik, '' Nouvelle approche pour optimiser un décodeur itératif '', CORESA, Lille, France, 24 Mai 2012.

[10] Haithem Ben Chikha, '' Etude et amélioration de turbo-codage distribué pour les réseaux coopératifs '', thèse de doctorat, université de Valenciennes et du Hainaut-Cambrésis et l'école nationale d'ingénieurs de Tunis, Tunisie, Avril 2012.

[11] Berrou, Glavieux et Thitirnajshima, '' Near shannon limit error correcting coding and decoding: turbo codes '', IEEE transmission and communication, proceedings of patents, Geneva, Switzerland, pp.1064-1070, May 1993.

[12] Alaa Eldin Hassan, Mona Shokair, '' Proposed deterministic interleavers for CCSDS turbo code standard '', journal of theoritical and applied information technology (JATIT), pp.29-33, January 2010.

[13] J. B. Doré, '' Optimisation conjointe de codes LDPC et de leurs architectures de décodage et mise en œuvre sur FPGA '', thèse de doctorat, INSA de Rennes, France, octobre 2007.

Bibliographie

[14] B. Ferhat et A. Guemari, ‘’ Les turbo codes convolutifs avec modulation BPSK pour un canal AWGN ‘’, Projet de fin d’étude, Université de Kasdi Merbah, département de mécanique et d’électronique, Ouaregla, Juin 2008.

[15] Y. Mori, ‘’ Codage de source et de canal ‘’, Vol 6, Ed. LAVOISIER, Paris-France, 2006.

[16] Frédéric LEHMANN, ‘’ Les Systèmes de Décodage Itératif et leurs Applications aux Modems Filaires et Non-filaires ‘’, thèse de doctorat, Institut National Polytechnique De Grenoble, France, décembre 2002.

[17] IFTENE Essedik, ‘’ Etude des structures d’entrelaceurs pour le codage turbo du canal pour l’optimisation des systèmes de communication par satellite ‘’, thèse de magister, Université des Sciences et de la Technologie d’Oran Mohamed Boudiaf, 2015.

[18] A. H. Osseiran, ‘’ Sur le décodage des codes turbo ‘’, Mémoire de maîtrise des sciences appliquées (Génie électrique), Oct 1999, Université de Montréal, Ecole Polytechnioue de Montréal, Canada.

[19] G. Royer, ‘’ Evaluation des entrelaceurs au sein des Codes Turbo par simulations ‘’, Mémoire de Maîtrise des sciences appliquée, Ecole polytechnique de Montréal, Nov 2000.

[20] J. Briffa, ‘’ Interleavers for Turbo Codes ‘’, Master dissertation of Philosophy of the University of Malta, Faculty of Engineering Oct 1999.

[21] C. Berrou, A. Glavieux, ‘’ turbo codes : principe et application ‘’, département électronique, département signal et communications, ENST, France. 1995.