UNIVERSITE DE LIMOGES

ED S2I : Sciences et Ingénierie pour l'Information

FACULTE DES SCIENCES ET TECHNIQUES

Année : 2010 **Thèse N° 72-2010**

THESE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITE DE LIMOGES

Discipline : "Electronique des Hautes Fréquences, Photonique et Systèmes"

Spécialité : "Télécommunications"

présentée et soutenue par

Naim KHODOR

Le 3 décembre 2010

Application des fonctions génératrices de chaos à la réalisation de codeurs de canal

Thèse dirigée par Jean-Pierre CANCES

JURY :

Monsieur Raymond QUERE	*Professeur à l'Université de Limoges*	*Président*
Madame Danièle FOURNIER-PRUNARET	*Professeur à l'INSA Toulouse*	*Rapporteur*
Monsieur Safwan EL ASAAD	*Maître de conférences à l'Université de Nantes*	*Rapporteur*
Monsieur Gilles BUREL	*Professeur à l'université de Bretagne Occidentale*	*Examinateur*
Monsieur Thierry BERGER	*Professeur à l'Université de Limoges*	*Examinateur*
Monsieur Jean-Pierre CANCES	*Professeur à l'ENSIL Limoges*	*Examinateur*

A ma chère famille

Remerciements

Ces travaux de doctorat se sont déroulés au laboratoire XLIM. Je désire ainsi remercier le Professeur Dominique Cross pour m'y avoir accueilli. Je tiens également à remercier le Professeur Raymond QUÉRÉ pour m'avoir permis de mener ces travaux dans le département Circuits Composants Signaux Systèmes qu'il dirige et pour m'avoir fait l'honneur de présider le jury de cette thèse.

Ma reconnaissance va également au Professeur Jean-Pierre CANCES pour avoir encadré ces recherches effectuées. Au cours de ces trois années, votre grande disponibilité, votre rigueur scientifique, votre enthousiasme et vos précieux conseils m'ont permis de travailler dans les meilleures conditions. La confiance que vous m'avez accordée ainsi que nos nombreuses discussions m'ont permis de progresser et de mieux appréhender les différentes facettes de la recherche. Soyez assuré, Monsieur, de toute mon estime et de mon profond respect.

J'exprime mes sincères et respectueux remerciements aux Madame Danièle FOURNIER-PRUNARET, Professeur à l'INSA Toulouse, et Monsieur Safwan EL ASAAD, Maître de conférences à l'Université de Nantes, pour avoir accepté de juger ce travail et assurer la tâche de rapporteurs.

Messieurs Gilles BUREL et Thierry BERGER, je suis très touché de l'honneur que vous me faites en acceptant d'examiné cette thèse et d'avoir pris un part de ce jury.

Mes remerciements vont également à Marie-Claude pour la quantité de détails pris en charge, et son aide précieuse, sympathique et indispensable.

J'exprime toute ma gratitude aux personnels du groupe Etudes des Systèmes de Télécommunications de l'ENSIL, notamment Monsieur Vahid MEGHDADI pour nos multiples discussions et le suivi de ce travail durant ces quelques années passées.

Un grand merci à mes collègues Maissa, Zahoor, Thierno, Hamid, Amir, Stéphanie, Sina, Nicolas et tous les autres. Bon courage à ceux qui me suivent …

Depuis huit ans, je partage avec eux des moments plus qu'agréables. Merci à Jad, Kenaan et Sayed qui savent bien combien la réussite de celles-ci dépend en particulier des grands moments passés ensemble.

J'ai également une pensée pour mes amis Hamzeh, Stéphane, Alain, Abdel, Hajj, Saleh, Chouman, Konso, Chreim, Lab … pour les soirées inoubliables et nos week-ends de «décompression».

Mes chers parents, ma sœur et mes frères, Merci pour tout. Je ne serai jamais arrivé sans votre support pendant toutes ces années. Je vous souhaite le meilleur …

Table des matières

INTRODUCTION GÉNÉRALE

DÉFINITION D'UN SYSTÈME CHAOTIQUE :

La **théorie du chaos** traite des systèmes dynamiques rigoureusement déterministes, mais qui présentent un phénomène fondamental d'instabilité appelé « sensibilité aux conditions initiales » ce qui les rend non prédictibles en pratique sur le « long » terme.

Le chaos est défini généralement comme un comportement particulier d'un système dynamique défini par des équations déterministes. Du point de vue mathématique la notion générale de système dynamique est définie à son tour à partir d'un ensemble de variables qui forment le vecteur d'état $\mathrm{x} = \{x_i \in \mathbb{R}\}, i = 1 \ldots n$ où n représente la dimension du vecteur. Ce jeu de variables a la propriété de caractériser complètement l'état instantané du système dynamique générique. L'ensemble de tous les états pouvant être pris par le système s'appelle l'espace des phases. Le processus évolue de manière déterministe si ses états futurs sont caractérisés par la connaissance de son présent et de ses états passés. En conclusion, la notion de déterminisme provient du fait que le système considéré est complètement caractérisé par son état initial et sa dynamique.

De manière générale, on peut classer les systèmes dynamiques chaotiques en deux catégories :

Système dynamique à temps continu.

Système dynamique a temps discret.

Un système dynamique en temps continu est décrit par un système d'équations différentielles :

$$\dot{x}(t) = \mathbf{F}\left(x(t), t\right) \qquad (1)$$

où $\mathbf{F} : \mathbb{R}^n \times \mathbb{R}^+ \mapsto \mathbb{R}^n$ désigne la dynamique du système. Si on associe à cette dynamique un état initial : $x_0 = x(t_0)$, pour chaque couple choisis (x_0, t_0) on peut identifier une solution unique du système définie à l'aide de l'équation (1). L'évolution des ensembles d'états successifs du système à chaque instant t, s'appelle trajectoire. Les systèmes de Rössler [1] et de Lorenz [2] sont parmi les systèmes dynamiques chaotiques les plus connus. Le système de Lorenz est défini par les équations suivantes :

$$\begin{cases} \dfrac{du}{dt} = \sigma(v-u) \\ \dfrac{dv}{dt} = -uw + \rho.u + v \\ \dfrac{dw}{dt} = uv + \beta w \end{cases} \qquad (2)$$

où *u,v* et *w* sont les variables d'état du système, σ, ρ *et* β sont les paramètres réels. Les paramètres et les conditions initiales de l'équation (2) ont été choisis de la manière suivante : $\sigma = 10; \rho = 28; \beta = 2.5$ avec (u_0, v_0, w_0) = (1.5, 4.8, 19.5). On observe que la dynamique du système de Lorentz donnée par (2) est indépendante de l'instant considéré, et généralement ce système est qualifié d'autonome. la figure suivante illustre une trajectoire particulière du système de Lorenz.

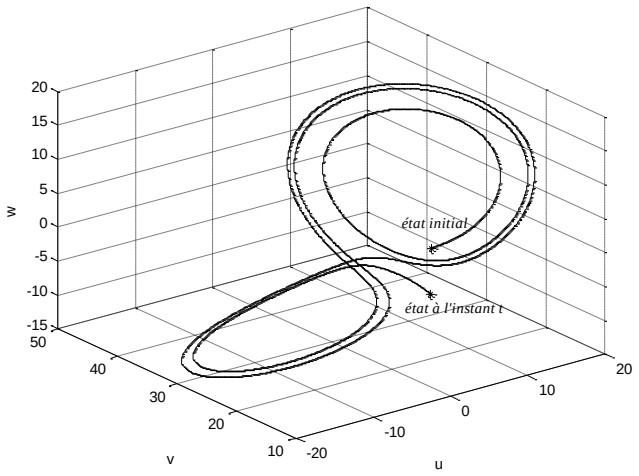


Figure 1 : exemple de trajectoire pour un système de Lorenz

Pour les systèmes dynamiques a temps discret, on a le modèle général suivant :

$$x(k+1) = \mathbf{G}(x(k), k) \qquad (3)$$

Où $\mathbf{G} : \mathbb{R}^n \times \mathbb{Z}^+ \mapsto \mathbb{R}^n$ est une fonction au moins continue ou continue par morceaux qui définit la dynamique du système discret. En fixant un état initial $\boldsymbol{x}_0 = \boldsymbol{x}(k_0)$, nous générons une solution unique de G.

Un exemple est donné par la fonction polynôme de Chebychev d'ordre 2 :

$$x_{k+1} = 2x_k^2 - 1 \qquad (4)$$

Où $\{x_k\}$ est réel.

La figure 2 donne une représentation de la fonction **G** défini par (4) dans le plan (x_k, xk_{+1}).

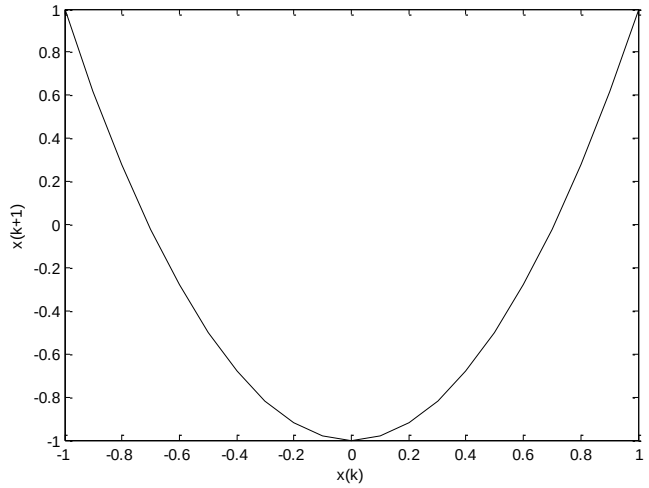


Figure 2 : La trajectoire pour la récurrence (4) avec $x_0 = 0.15$

I. COMPORTEMENT DES SYSTÈMES DYNAMIQUES

A partir d'un état initial x_0 et après un régime transitoire, la trajectoire d'un système dynamique atteint un régime limité de l'espace des phases. Ce comportement asymptotique obtenu pour $t,k \to \infty$ est une des caractéristiques les plus importantes à étudier pour tout système dynamique. Si dans le cas d'un système linéaire la solution asymptotique est indépendante de la condition initiale et unique, en présence de non – linéarité il existe une plus grande variété de régimes permanents différents, parmi lesquelles on trouve, par ordre de complexité : point d'équilibre, solution périodiques, solution quasi-périodiques et chaos, respectivement. Il faut préciser que cette fois le comportement développé par un système dynamique particulier est fortement dépendant de la condition initiale choisie [3][4].

Pour illustrer les comportements, nous prenons comme exemple l'équation logistique définie de la manière suivante [5] :

$$x_{k+1} = r\left(1 - x_k\right)x_k \qquad (5)$$

Le mécanisme de construction d'une séquence est tout d'abord montré sous la forme d'un diagramme en toile (*web diagram* [6]figure 3). Cette méthode permet la génération de la séquence choisie, graphiquement en utilisant la projection des états successifs par rapport à la diagonale principale figure 3.

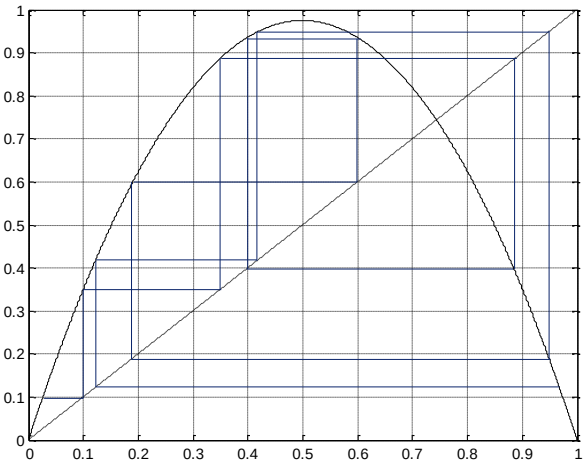


Figure 3 : Diagramme d'évolution pour la fonction logistique (WEB DIAGRAM), r = 3.9

Suivant les valeurs de r et la valeur initial x_0 de la suite x_k, celle-ci présente des comportements très différents. L'étude de l'évolution de la dynamique du système vers un comportement chaotique consiste à analyser la variation de la valeur du paramètre r, appelé aussi paramètre de bifurcation (figure 4). On appelle cette représentation diagramme de bifurcation parce que le comportement asymptotique subit, pour des valeurs du paramètre *r* bien déterminés, une bifurcation de l'ensemble des états limites. Dans le cas continu la bifurcation se manifeste comme une multiplication des trajectoires possibles.

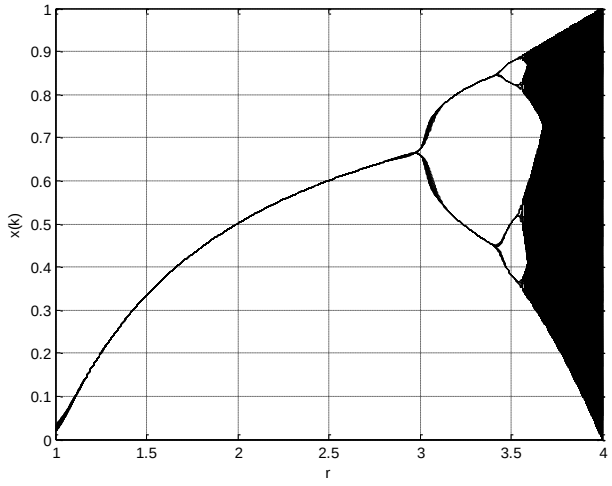


Figure 4: Le diagramme de bifurcation pour la fonction logistique

Pour chaque type de régime permanent on a :

- *points d'équilibre* : Dans ce cas, la solution asymptotique est représentée par un point, sa valeur étant déterminée en fonction de la condition initiale choisie. Ainsi, pour des conditions initiales différentes on peut retrouver plusieurs points d'équilibres. De même ces points peuvent être stables ou instables suivant que les trajectoires voisines convergent ou divergent entre-elles.

Dans le cas de la dynamique logistique, on observe que pour toute valeur $r \in [1,3]$, le régime permanent est formé par un point limite stable, sa valeur étant dépendante du choix de paramètre *r*. La figure 5 nous donne un aperçu d'une telle trajectoire pour *r* = 2. Ainsi on observe qu'après une période de transition relativement courte la séquence se stabilise autour du point fixe qui cette fois est : $x_\infty = 0.5$.

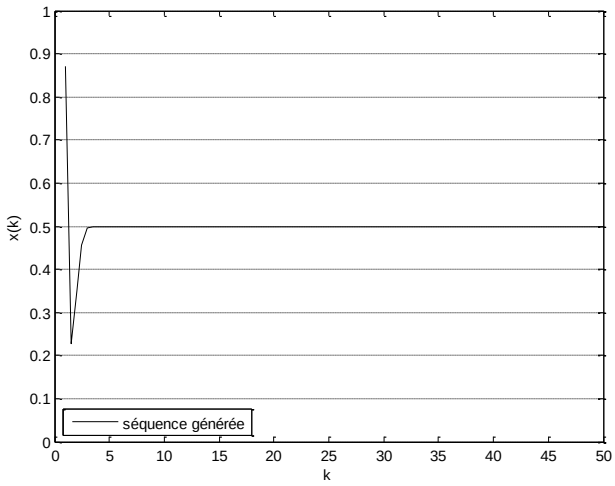


Figure 5 : séquence générée et états limites pour r = 2.

- *régime périodique* : Le régime asymptotique permanent périodique correspond à une trajectoire dont les répliques d'une portion élémentaire sont espacées à des intervalles $nT, n \in \mathbb{N}^+$, *T* désignant la période. Pour la fonction logistique par exemple le choix de *r* = 3.2 nous garantit que l'ensemble des états limites est formé par deux points, et la période correspond à deux échantillons (figure 6). .

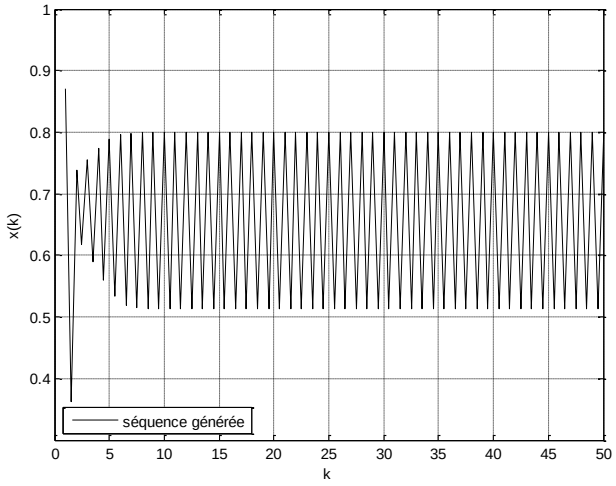


Figure 6 : séquence générée et états limites pour r = 3.2.

- *régime quasi-périodique* : correspond à une somme de solutions périodiques dont le rapport des périodes est un nombre irrationnel. Un régime quasi-périodique peut être représenté dans l'espace d'état par un tore.

- *régime chaotique* : le régime chaotique est par définition tout régime permanent qui n'appartient à aucune des classes présentées antérieurement. Une telle solution a une trajectoire asymptotique bornée avec une extrême sensibilité aux conditions initiales. Ainsi deux trajectoires générées à partir de CI (conditions initiales) très proches, vont diverger très vite l'une par rapport à l'autre. Cette sensibilité par rapport aux CI traduit aussi le comportement en apparence stochastique des générateurs chaotiques, de telle sorte qu'une prévision à long terme du comportement du système est impossible. Dans la figure 7 un exemple est donné pour deux CI espacées par une valeur de 10^{-4} et on peut observer que juste après quelques itérations les deux trajectoires divergent et deviennent non corrélées.

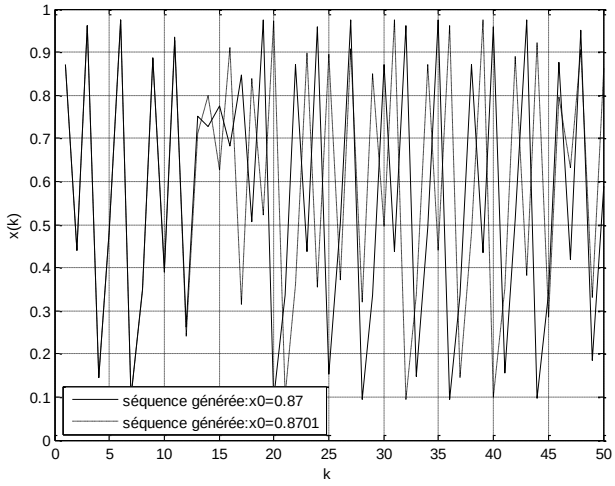


Figure 7 : séquences générées et sensibilité aux CI pour r = 3.9

Généralement, l'ensemble des solutions asymptotiques stables décrites ci-dessus est qualifié d'attracteur ; il représente la région de l'espace d'état au voisinage de laquelle les trajectoires restent confinées lorsque, $t, k \to \infty$. En parallèle avec la définition de l'attracteur, apparaît la notion de bassin d'attraction qui est défini comme la région de l'espace d'état formée par l'ensemble des C.I à partir desquelles l'attracteur sera atteint.

Le diagramme de bifurcation donné dans la figure 4 résume l'évolution d'un système dynamique non linéaire vers le chaos. Il porte en abscisse les différentes valeurs de r et en ordonnée les valeurs x_k après un grande nombre d'itérations. Par exemple, pour la fonction logistique le diagramme montre que pour une valeur critique $r < r_c \; avec \; r_c = 3.57$, la suite x_k présente un comportement périodique suivant une structure ordonnée. Lorsque $r \geq r_c \; avec \; r_c = 3.57$, pour la plupart des valeurs de r la suite s'apparente à un cycle d'ordre infini. De plus a chaque valeur x_0 correspond une orbite différente, alors que pour les valeurs de $r < r_c \forall \; x_0 \in]0,1]$, la suite converge vers un ensemble fini de points. Pour certain valeur de $r > r_c$ le système peut devenir chaotique.

II. CHAOS ET COMMUNICATION

La dynamique non linéaire et la théorie du chaos ont attiré l'attention de nombreux chercheurs dans de nombreux domaines différents [7][8]. Cependant, à part les capacités de modélisation

de la dynamique non - linéaire et la théorie du Chaos, les signaux produits par les systèmes chaotiques conviennent particulièrement bien à d'autres types d'application grâce a leurs caractéristiques particulières. Par exemple, les séquences chaotiques sont intéressantes pour l'analyse du signal, la synthèse de signaux et pour les communications numériques et analogiques [9][10]. Parmi les propriétés des signaux chaotiques qui les rendent si intéressants, on peut citer leur génération facile et une faible probabilité de détection. Par conséquent, il n'est pas surprenant qu'ils aient été considérés depuis longtemps pour les applications de communication et de cryptographie [11][12][13][14][15][16][28].
Le comportement chaotique de ces systèmes rend leurs utilisations très importantes pour les systèmes de communication sécurisés. L'intérêt d'utiliser des signaux chaotiques dans ces systèmes réside dans trois propriétés fondamentales des signaux et systèmes chaotiques :

- un signal chaotique est obtenu à partir d'un processus purement déterministe ; il est donc possible de le reconstituer en se plaçant dans les mêmes conditions que celles qui ont contribué à le créer ;
- un système chaotique engendre un signal à large spectre et peut donc permettre de transmettre des signaux très variés.
- Deux trajectoires de signaux chaotiques issues d'un même système chaotique, mais obtenues à partir de conditions initiales différentes, ont une inter-corrélation très faible.

Ces systèmes ont été étudiés dans les systèmes de communication Multi-Utilisateurs [17][18][21][29][30][31][32], par exemple, pour les systèmes d'accès multiple par code (CDMA), en plus, ils ont également été considérés comme des candidats potentiels à la réalisation de codeurs de canal [33][34][35][36][37].

III. PLAN DU MÉMOIRE DE THÈSE

C'est dans ce contexte de candidats potentiels à la réalisation de codeurs de canal que se situe le travail de thèse présenté dans ce mémoire. Nous allons étudier différents schémas de codeurs de canal générés à partir de signaux chaotiques et quantifier leur efficacité selon les différents contextes de transmission étudiés. Nous distinguons principalement deux contextes.

Le premier concerne les systèmes que nous pourrions appeler pseudo-chaotique où l'alphabet de sortie du modulateur ou encore la constellation transmise reste celle d'une modulation MAQ-*N* classique. Cependant, le codeur de canal classique à base de OU exclusif est remplacé par une fonction non-linéaire de type modulo. On peut obtenir, ce faisant, des codeurs de canal avec des performances intéressantes à condition de bien optimiser la distance libre obtenue.

Le second concerne le cas beaucoup plus complexe où la constellation transmise présente une distribution uniforme (i.e. la plus aléatoire possible) dans un intervalle donné. Dans ce cas le signal modulé présente une distribution uniforme et un comportement que l'on pourrait qualifier de réellement chaotique. Concevoir des codeurs de canal performants dans ce contexte est un défi difficile. Nous verrons en effet, dans le chapitre 3, que l'état de l'art avec des fonctions de mapping chaotiques mono-dimensionnelles ne permet même pas d'obtenir des performances aussi bonnes qu'un système NRZ non codé. Fort heureusement, les travaux de Kozic [41][46] ont montré que l'utilisation de mapping multi-dimensionnels permettait d'améliorer sensiblement les performances en utilisant une détection à Maximum de Vraisemblance à base d'algorithme de Viterbi en réception. En particulier, nous montrerons dans le chapitre 4 que l'utilisation de mapping de dimensions 2 ou 3 permettait d'obtenir de meilleures performances qu'un système non codé. Cependant, l'amélioration des performances reste encore faible et ne justifie pas l'emploi de systèmes aussi complexes. C'est pourquoi nous proposons dans le chapitre 5 une généralisation à des mapping de grande dimension obtenus à partir de matrices creuses de grande taille. Cette fois, nous devons opérer avec un algorithme de décodage différent pour limiter la complexité du récepteur. Nous proposons à partir des idées de Kozic l'utilisation d'un graphe factoriel pour modéliser l'ensemble codage plus modulation et ainsi dérouler l'algorithme de propagation de croyance (B.P : Belief Propagation) sur l'ensemble de ce graphe. Les performances obtenues sont excellentes et augurent d'une réelle possibilité d'envisager les codes proposés comme des alternatives crédibles aux codes performants que sont les turbo codes et les codes LDPC.

CHAPITRE I CODAGE CORRECTEUR D'ERREUR UTILISANT UNE FONCTION GÉNÉRATRICE DE CHAOS AVEC MAPPING CLASSIQUE (GRAY)

I. INTRODUCTION

Le travail de thèse présenté dans ce mémoire a d'abord eu pour cadre le projet ANR ACSOM (Apport du Chaos dans les Systèmes de transmission Optique et Micro-onde) où nous avons d'abord travaillé sur des schémas de codeur non-linéaires sans considérer le problème du mapping de sortie comme faisant partie de la génération de chaos. C'est ainsi que les résultats présentés dans ce chapitre concernent ceux de codeurs non-linéaires opérant à base de fonctions troncature ou modulo bien connues en génération de chaos mais dont le mapping de sortie reste celui de modulations MDP-*N* ou MAQ-*N* classiques (mapping de Gray par exemple). Ainsi, la constellation des symboles transmis est celle d'une modulation MAQ-*N* classique. Les schémas présentés dans ce chapitre ne sont donc pas des systèmes complètement chaotiques nous les qualifierions plutôt de semi-chaotique. Certains travaux existent dans la littérature dans ce domaine même s'ils restent marginaux [19][20].

Un vrai système chaotique doit contenir un mapping de sortie qui inclut une fonction génératrice de chaos pour que sa constellation transmise soit la plus imprévisible possible et possède une distribution de probabilité qui s'approche d'une loi uniforme sur l'ensemble des valeurs possibles à transmettre. La raison du choix de ces systèmes semi-chaotiques est qu'historiquement nous avons d'abord utilisé, en le modifiant largement, un schéma de génération de chaos issu des travaux de Mr A. Layec [21] où nous avons voulu montré qu'une détection à Maximum de Vraisemblance avec l'algorithme de Viterbi permettait d'obtenir de meilleures performances que la simple reconstruction d'états d'un filtre à réponse impulsionnelle infinie qui était utilisée jusqu'alors. C'est cette première approche qui a été retenue et que nous présentons dans ce chapitre. Il faut signaler que récemment, certain auteurs[22][23][24] se sont intéressés à l'utilisation de codeurs à base de chaos avec un mapping de sortie inspiré des règles de partitionnement de Ungerboeck mais dont en sortie l'alphabet appartient à une modulation MAQ-*N* classique.

Nous montrons d'abord la structure de base étudiée avec l'algorithme de décodage par simple reconstruction des états puis nous développons nos propres algorithmes pour proposer une détection à Maximum de Vraisemblance et une utilisation de ce schéma de codage en tant que composant d'un turbo-code à concaténation parallèle. Enfin, une étude des performances de ce codeur en présence de canaux sélectifs en fréquence est proposée pour conclure le chapitre.

II. STRUCTURE DE BASE

Le schéma que nous avons utilisé pour notre encodeur non-linéaire est donné sur la figure ci-dessous. Il s'agit principalement d'un structure de filtre à réponse impulsionnelle infinie de la forme : $H(z) = \frac{Y(z)}{X(z)} = \frac{1}{D(z)} = \frac{1}{d_0 + d_1.z^{-1} + d_2.z^{-2} + ... + d_n.z^{-n}}$ (1).

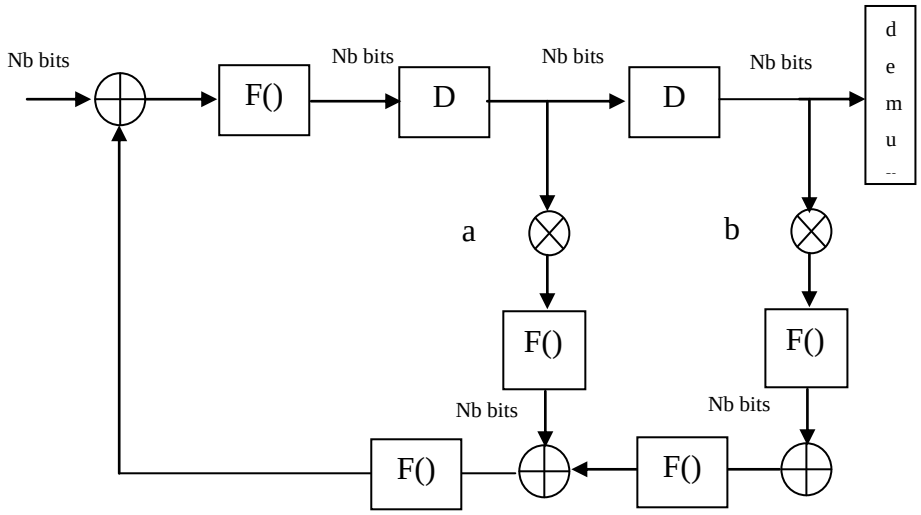


Figure 1 : structure de base de l'encodeur

Les opérateurs F(.) sont simplement des opérateurs de saturation qui permettent de conserver les mêmes formats en sortie de telle ou telle opération que les formats d'entrée. Ceci est particulièrement important pour les opérations de multiplication où le risque de dépassement du format en sortie est élevé. Pour les calculs de performances nous avons pris un nombre limité de bits avec une valeur limitée de Nb au maximum égale à 6 et nous avons opéré une troncature des bits de poids fort pour chaque résultat d'opération arithmétique de manière à toujours conserver un nombre de bits en sortie égal à Nb. On obtient alors une caractéristique non linéaire pour F(.) que l'on peut représenter sous la forme du schéma de la figure 2.

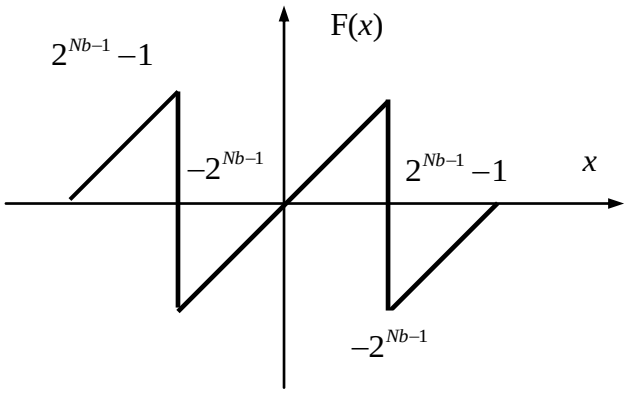


Figure 2 : Représentation de la fonction troncature ou modulo

On peut alors considérer le schéma d'encodage de la figure 1 de deux façons :

- soit comme un simple filtre RII et dans ce cas on peut reconstituer la séquence des symboles envoyés avec l'équation de récurrence issue de (1) :

$$X(z) = Y(z).(d_0 + d_1.z^{-1} + d_2.z^{-2} + ... + d_n.z^{-n}) \qquad (2)$$

  Ce qui conduit au décodage des $x(n)$ à partir des $y(n)$ en utilisant la relation :

$$x(m) = y(m).d_0 + y(m-1).d_1 + y(m-2).d_2 + ... + y(m-n).d_n \qquad (3)$$

  L'inconvénient de l'équation est évidemment son extrême sensibilité au bruit comme Frey l'avait déjà observé dans son article fondateur sur le chaos. C'est pour ça que dans la thèse de Mr A. Layec c'est une modulation de fréquence numérique moins sensible au bruit que les modulations de phase MDP-*N* qui a été retenue.

- soit comme un codeur convolutif dont les états sont constitués par le contenu des deux registres à Nb bits sur la vois supérieure de l'encodeur de la figure 1. En prenant Nb égal à 6 on obtient un codeur convolutif à $2^{12} = 4096$ états, ce qui est considérable (à titre d'exemple le codeur convolutif le plus complexe de la norme 3G est un codeur à 512 états). Le rendement de ce codeur est égal à un puisque le nombre de bits en sortie (Nb) est égal au nombre de bits en entrée (Nb).

Il y a de plus une difficulté supplémentaire liée à l'utilisation de ce codeur ; il s'agit du choix des paramètres *a* et *b* du filtre de la figure 1. Le choix de ces paramètres peut se faire en essayant de maximiser la distance libre de ce codeur. Le problème est que, contrairement à un codeur linéaire, il n'est pas possible d'utiliser une séquence de référence toute à zéro et il faut donc tester toutes les séquences possibles au départ de tous les états. Pour calculer sa valeur il faut déterminer les trajectoires dans le treillis qui partent depuis le même état S_i=S_i^* et qui évoluent ensuite à travers des chemins disjoints (après L coups d'horloge) vers le même état S_k=S_k^*, non nécessairement égale a S_i. On parlera alors de boucles de longueur L. Une recherche exhaustive dans le cas proposé ici s'avère donc impossible (4096 états !!) et nous nous sommes contentés de tests limités à des séquences de longueur L faible ($L < 5$). Nous présentons dans le tableau ci-dessous les résultats que nous avons obtenus avec différentes valeurs de Nb pour les meilleurs couples de valeurs de (*a*, *b*) sélectionnés.

Nombre de bits Nb	3	4	5	6
Distance libre df	**4**	**6**	**7**	**11**

Tableau 1 : distances libres du codeur de la figure 1 pour différentes valeurs de a et b

On constate que les distances libres obtenues restent très faibles, à titre de comparaison il faut se rappeler qu'un simple codeur convolutif binaire (5, 7) a une distance libre de 5 et qu'un codeur convolutif binaire (133, 171) a une distance libre de 10. Ceci est particulièrement pénalisant pour les valeurs importantes de Nb, en effet le codeur dans ce cas sort les bits par groupe de 6, et une distance libre de 11 signifie qu'entre les deux chemins les plus proches qui s'écartent puis se rejoignent dans le treillis il n'y a que 11 bits qui diffèrent en sortie. Par contre, il faut signaler que pour chaque valeur de df obtenue, il existe à priori une infinité de choix de valeurs de a et de b. En fait, nous avons toujours obtenu au moins 10 valeurs de couples (a,b) qui permettaient d'obtenir les valeurs de df annoncées.

Nous avons ensuite tracé les performances des codes obtenus sur canal de Gauss pour les cas Nb = 3, 4 et 6. Le problème qui se pose dans ce cas est d'abord le choix de la modulation. Nous avons opté pour des modulations de type MDP-8 pour Nb = 3 bits, puis MAQ-16 pour Nb = .4 bits et enfin MDP-8 pour Nb = 6 bits, ce qui oblige dans ce cas à couper en deux les blocs de 6 bits qui sortent du codeur. Ainsi, dans le cas où Nb = 6 bits, les 6 bits de sortie du

codeur sont convertis en 2 symboles MDP-8 (mapping de Gray). Dans les cas Nb = 3 et 4, on a utilisé un algorithme simple de Viterbi pour le décodage en ce sens où on cumule les métriques sur l'ensemble de la durée des paquets. Les simulations sont réalisées sous Matlab. Par contre, dans le cas Nb = 6, au vu du nombre des états, nous avons utilisé un algorithme de Viterbi à fenêtre glissante en essayant d'optimiser la taille de cette fenêtre pour avoir le meilleur compromis performance-complexité. Les courbes tracées en fonction du rapport signal à bruit (Es/N0) sont données ci-dessous.

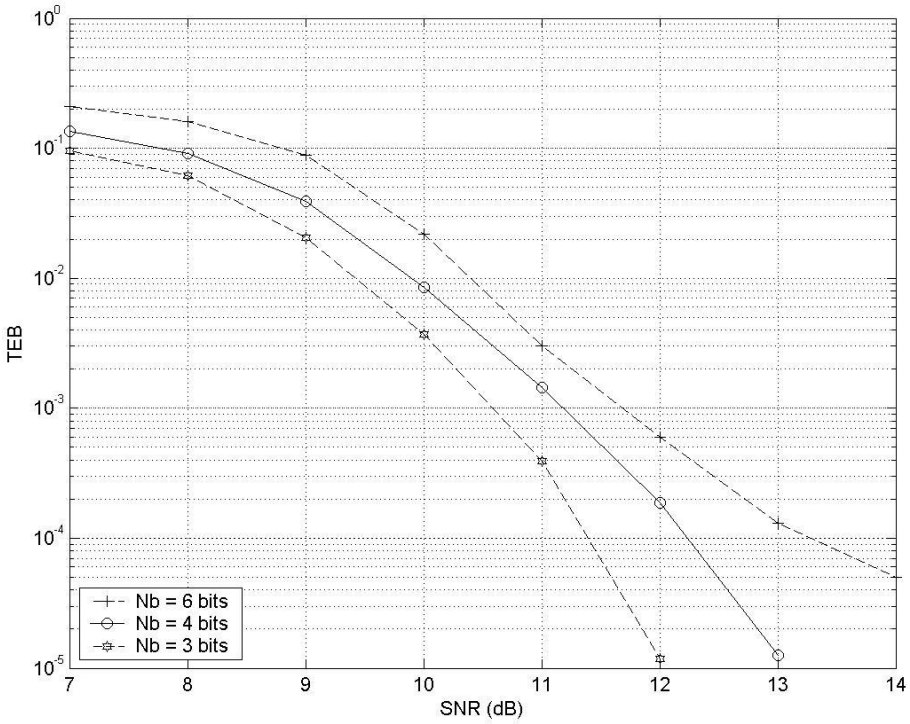


Figure 3 : Performances du codage chaotique (Figure 1) sur canal à bruit additif Gaussien

On peut remarquer un effet plancher dans le cas Nb = 6 bits. Ceci peut provenir d'un mauvais choix de la taille de la fenêtre glissante pour l'algorithme de Viterbi (taille qui a été d'abord prise égale à 128) mais, même en augmentant la largeur de cette fenêtre, nous avons constaté le même phénomène. Une autre explication plausible est que la distance minimale très faible (11) génère des mots d'erreur de poids faible qu'il est très difficile d'éliminer même en augmentant le SNR.

A titre de comparaison, nous avons tracé sur la Figure 4 les résultats obtenus par reconstruction d'états du filtre à l'aide de l'équation (3). Il est clair que l'utilisation du Maximum de Vraisemblance à l'aide de l'algorithme de Viterbi permet d'améliorer

considérablement les performances avec des gains supérieurs à 3 dB pour les cas Nb = 4 bits et Nb = 6 bits.

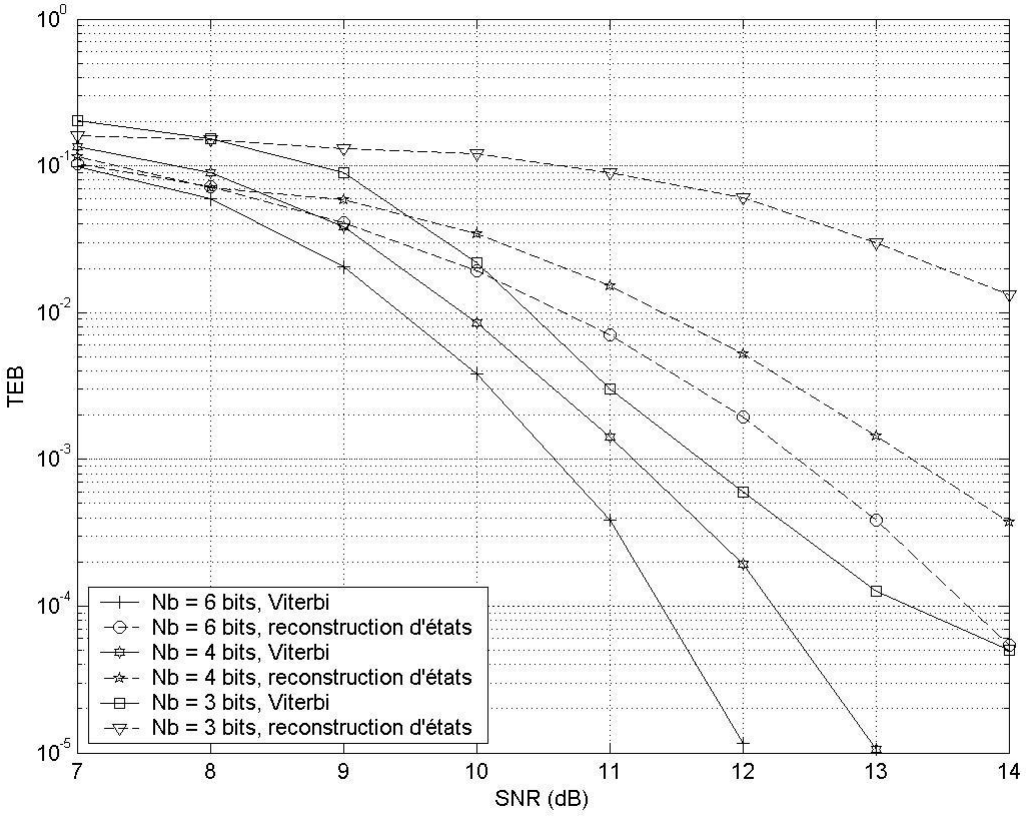


Figure 4 : Performances du codage chaotique (Figure 1) sur canal à bruit additif Gaussien, comparaison entre l'algorithme de Viterbi et la reconstruction d'états du filtre (3)

III. AMÉLIORATION DES PERFORMANCES DU CODEUR

Il est clair qu'en l'état ce codeur ne possède pas de redondance et donc aucun pouvoir de correction réel. Cependant son effet de mémoire (2 × Nb bits) lui permet éventuellement d'être utilisé en concaténation série avec un décodeur de canal conventionnel pour échanger des informations extrinsèques. Il faut lui adjoindre une information supplémentaire pour en faire un codeur à part entière, c'est pourquoi nous ajoutons l'information systématique d'entrée, ce faisant nous obtenons un codeur M-aire (sortie sur 2 × Nb bits) systématique récursif dont la structure est semblable à celle des codeurs convolutifs utilisés en

concaténation parallèle pour les turbo-codes. Nous travaillons donc sur les structures suivantes

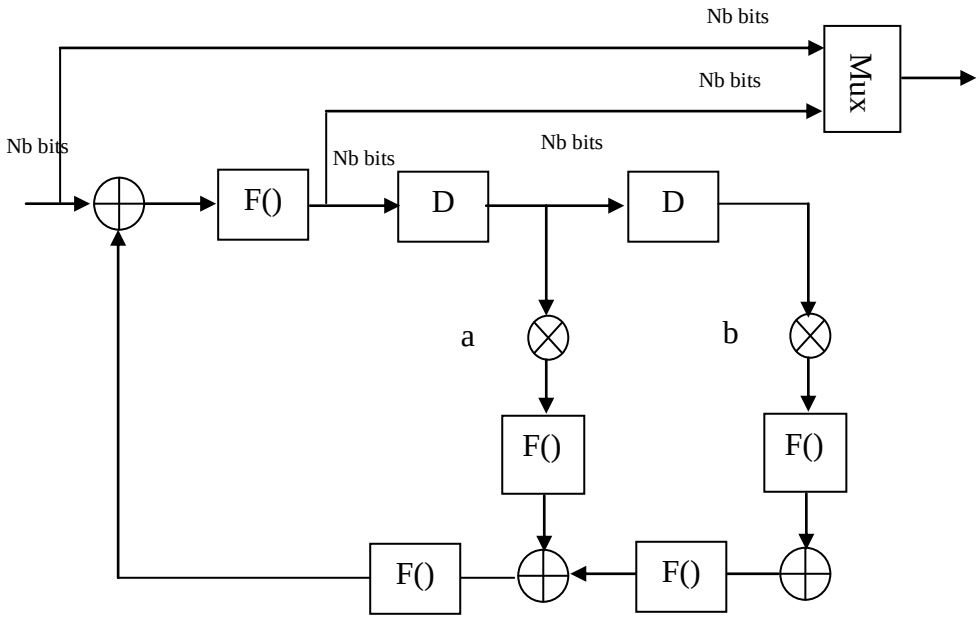


Figure 5 : Structure du nouveau codeur utilisé

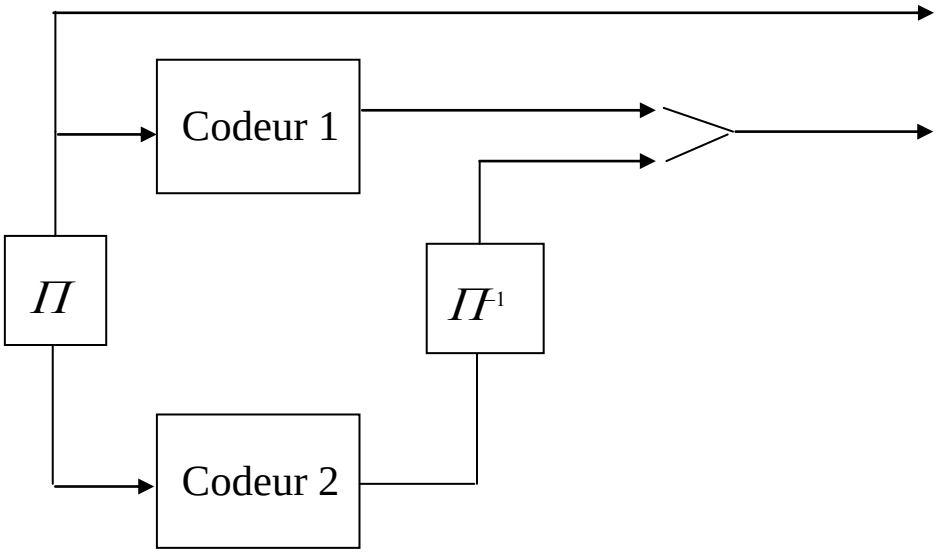


Figure 6 : Structure du turbo-codeur chaotique en concaténation parallèle

Pour des raisons évidentes de complexité de l'algorithme de décodage, nous nous limiterons ici à Nb = 3 bits pour le codeur de la figure 6. Dans le cas du codeur de la figure 5 le choix des paramètres a et b a été fait une fois de plus pour optimiser la distance libre du codeur ; les essais limités que nous avons effectués nous ont permis d'obtenir les performances suivantes en termes de distance libre.

Nombre de bits Nb	3	4	5	6
Distance libre df	**8**	**10**	**11**	**15**

Tableau 2 : distances libres du codeur de la figure 5 pour différentes valeurs de a et b

L'algorithme de décodage utilisé est du type Viterbi pour le schéma de la figure 5 et du type Max-logMAP pour le schéma de la figure 6. Le principe de l'algorithme Max-logMAP est décrit à la fin de ce chapitre. Les performances obtenues par le code de la figure 5 sont illustrées sur la figure 7.

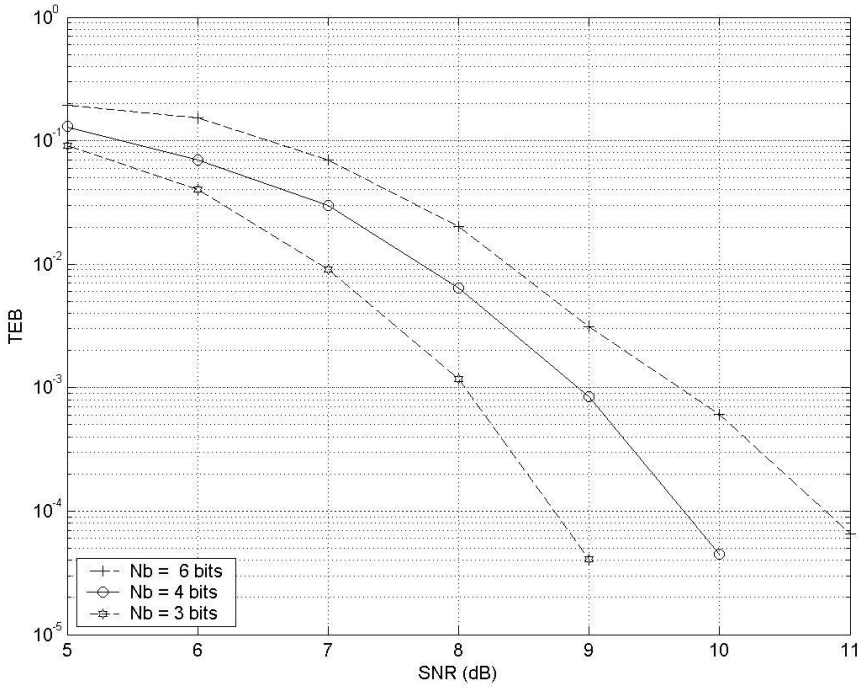


Figure 7 : Performances du codage chaotique (Figure 5) sur canal à bruit additif Gaussien

Par rapport aux résultats de la figure 3 on voit que l'effet plancher pour le cas Nb = 6 bits a quasiment disparu. De plus, les performances dans les cas Nb = 4 bits et Nb = 3 bits se

trouvent améliorées. Par exemple avec un TEB de 10^{-4} et Nb = 4 bits le SNR nécessaire sur la figure 7 est égal à 9.7 dB tandis que le SNR requis pour un TEB à 10^{-4} sur la figure 3 est de 12.3 dB. On obtient un gain de 2.6 dB. Avec un TEB de 10^{-4} et Nb = 3 bits le SNR nécessaire sur la figure 7 est égal à 8.7 dB tandis que le SNR requis pour un TEB à 10^{-4} sur la figure 3 est de 11.4 dB. On obtient un gain de 2.7 dB.

Les performances du turbo codeur à concaténation parallèle sont illustrées sur la figure 8 dans le cas Nb = 3 bits et sans poinçonnage alternatif des symboles de parité du codeur supérieur et du codeur inférieur. Un problème classique de l'utilisation des turbo-codes réside dans la taille et le type de l'entrelaceur choisi. Nous avons pris dans nos simulations des entrelaceurs de type S-random ou pour une taille N de paquet donné l'on est assuré que deux positions successives en sortie sont au moins écartées de $\sqrt{N/2}$. Pour la taille des blocs on prend une taille raisonnable de 512.

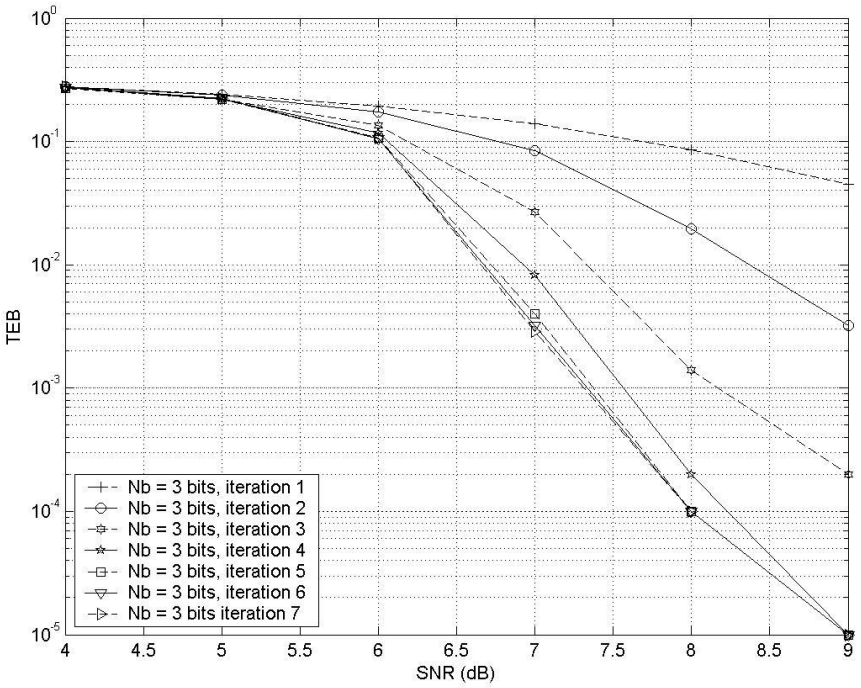


Figure 8 : Performances du turbo codeur chaotique (Figure 6) sur canal à bruit additif Gaussien et sans poinçonnage

On constate, malgré l'effet de plancher dû à l'utilisation d'un codeur à faible distance libre et à la concaténation parallèle utilisée, une amélioration des performances par rapport à la courbe de la figure 7. Le TEB de 10-4 est atteint au bout de la cinquième itération à un SNR

de 8 dB alors que sur la courbe de la figure 7, cette valeur de TEB est obtenue avec un SNR de 8.7 dB. On obtient donc un gain de 0.7 dB avec le schéma turbo non poinçonné

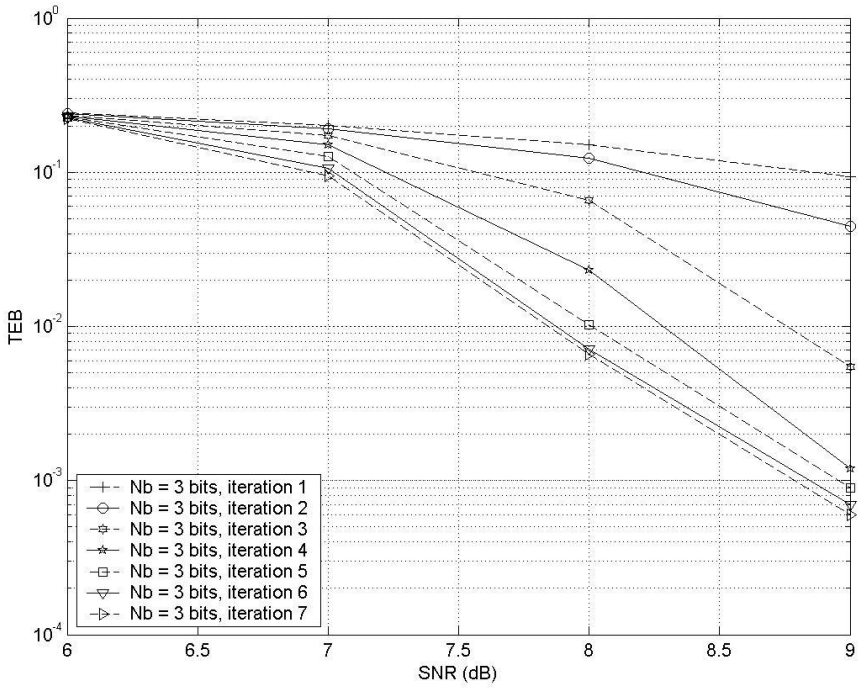


Figure 9 : Performances du turbo codeur chaotique (Figure 6) sur canal à bruit additif Gaussien et avec poinçonnage

Les performances du schéma poinçonné sont illustrées sur la figure 9 avec également 7 itérations. On constate une dégradation des performances par rapport aux résultats de la figure 8. Pour un TEB de 10^{-3} le SNR nécessaire est égal à 8.8 dB à la septième itération pour le codeur poinçonné de la figure 6 tandis que la même performance (TEB = 10^{-3}) est obtenu à un SNR de 7.4 dB pour le codeur turbo non poinçonné, soit une perte de 1.4 dB. Par rapport au schéma conventionnel de la figure 5, le turbo codeur poinçonné est moins performant puisque le TEB de 10^{-3} est obtenu à SNR = 8 dB pour le codeur conventionnel contre 8.8 dB pour le turbo-code poinçonné.

IV. UTILISATION DU CODE CHAOTIQUE DANS LE CAS D'UN CANAL SÉLECTIF EN FRÉQUENCE

Dans cette partie on regarde les performances obtenues avec ce codeur dans le cas d'un canal sélectif en fréquence. Nous proposons d'insérer le schéma de codage de la figure 5 dans la chaîne de transmission suivante.

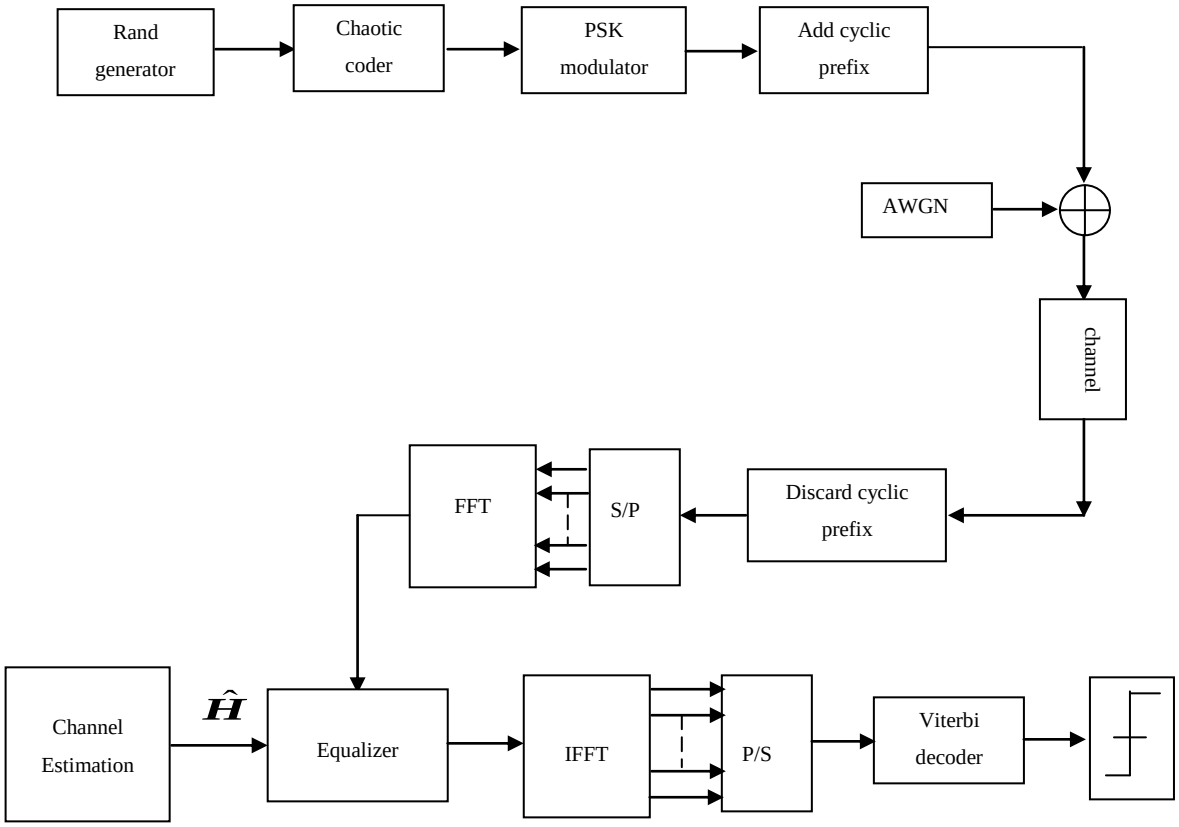


Figure 10 : Chaîne de transmission utilisant le codeur chaotique dans Le cadre d'une transmission sur canal sélectif en fréquence

Nous choisissons, pour simplifier l'algorithme d'égalisation, d'opérer un passage dans le domaine fréquentiel en réception à l'aide d'un FFT. Ce faisant, et même si on n'utilise pas une modulation multi-porteuses de type OFDM, on obtient en réception de pouvoir transformer un produit de convolution en un produit simple. Cependant, et de la même façon qu'en OFDM, l'insertion d'un préfixe cyclique en tête de chaque paquet permet d'éviter en réception les interférences entre blocs adjacents. Le principe de l'égalisation dans le domaine

fréquentiel est le suivant. Un bloc de signaux $d(k),(0 \le k \le N-1)$ de longueur N égale à la taille de la FFT choisie est transmis sur le canal. Un préfixe cyclique de taille au moins égale à l'ordre de sélectivité du canal est inséré entre chaque bloc pour éliminer les interférences entre blocs. En supposant que le canal s'écrive : $\boldsymbol{h} = [h(0), h(1), ..., h(M)]^T$, le vecteur du signal reçu s'écrit dans ce cas.

$$\boldsymbol{y} = \tilde{\boldsymbol{H}}.\boldsymbol{d} + \boldsymbol{n} \qquad (4)$$

avec $\boldsymbol{d} = [d(0), d(1), ..., d(N-1)]^T$, $\tilde{\boldsymbol{H}}$ est une matrice de Toeplitz circulante en bloc (matrice de convolution). Le vecteur $\boldsymbol{Y}$ dans le domaine fréquentiel s'exprime par :

$$\boldsymbol{Y} = \boldsymbol{F}.\boldsymbol{y} = \boldsymbol{\Lambda}.\boldsymbol{d} + \boldsymbol{F}.\boldsymbol{n} \qquad (5)$$

où $\boldsymbol{F}$ est la matrice de transformation de Fourier discrète : $F_{l,k} = (1/\sqrt{N}).\exp(-j.(2.\pi/N).l.k)$. $0 \le l, k \le N-1$. Une matrice de FFT diagonalise une matrice circulante de convolution donc $\boldsymbol{\Lambda}$ est une matrice diagonale avec comme élément numéro k sur la diagonale principale :

$$H_k = \sum_{l=0}^{M} h(l).\exp(-j.\frac{2.\pi}{N}.k.l) \qquad (6)$$

et on a la relation : $\boldsymbol{H} = \sqrt{N}.\boldsymbol{F}.\tilde{\boldsymbol{h}}$ (7) avec : $\boldsymbol{H} = [H_0, H_1, ..., H_{N-1}]^T$.

Pour réaliser l'égalisation du canal, il faut d'abord estimer les composantes fréquentielles H_k de ce dernier. Pour cela on utilise une séquence de symboles pilotes : $\boldsymbol{p} = [p_0, p_1, ..., p_{N-1}]^T$, avec cette séquence on peut réécrire (5) sous la forme : $\boldsymbol{Y} = \boldsymbol{P}.\boldsymbol{H} + \boldsymbol{N}$ (8) avec $\boldsymbol{P}$ qui est une matrice diagonale correspondant à la TF des symboles pilotes. Le filtre de Wiener estimateur de $\boldsymbol{H}$ s'écrit alors :

$$\hat{\boldsymbol{H}}_{MMSE} = \boldsymbol{R}_{\boldsymbol{HY}}.\boldsymbol{R}_{\boldsymbol{YY}}^{-1}.\boldsymbol{Y} \qquad (9)$$

Où : $\boldsymbol{R}_{\boldsymbol{HY}} = E[\boldsymbol{H}.\boldsymbol{Y}^H] = \boldsymbol{R}_{\boldsymbol{HH}}.\boldsymbol{P}^H$ (10)

Et : $\boldsymbol{R}_{\boldsymbol{YY}} = E[\boldsymbol{Y}.\boldsymbol{Y}^H] = \boldsymbol{P}.\boldsymbol{R}_{\boldsymbol{HH}}.\boldsymbol{P}^H + \sigma_n^2.\boldsymbol{I}_N$ (11)

$\boldsymbol{R_{HY}}$ et $\boldsymbol{R_{YY}}$ sont les matrices d'intercorrélation entre $\boldsymbol{H}$ et $\boldsymbol{Y}$ et la matrice d'autocorrélation de $\boldsymbol{Y}$. $\boldsymbol{R_{HH}}$ est la matrice d'autocorrelation de la fonction de transfert du canal dans le domaine fréquentiel $\boldsymbol{H}$, et elle est supposée connue au récepteur. On obtient alors l'estimateur de Wiener ou des moindres carrés sous la forme :

$$\hat{\boldsymbol{H}}_{MMSE} = \boldsymbol{R_{HH}}(\boldsymbol{R_{HH}} + \sigma_n^2(\boldsymbol{P}^H\boldsymbol{P})^{-1})^{-1}.\hat{\boldsymbol{H}}_{LS} = \boldsymbol{Q}.\hat{\boldsymbol{H}}_{LS} \qquad (12)$$

Avec : $\hat{\boldsymbol{H}}_{LS} = \boldsymbol{P}^{-1}.\boldsymbol{Y}$ et : $\boldsymbol{Q} = \boldsymbol{R_{HH}}.(\boldsymbol{R_{HH}} + \sigma_n^2.(\boldsymbol{P}^H\boldsymbol{P})^{-1})^{-1}$.

L'égalisation du canal dans le domaine fréquentiel consiste à appliquer alors les coefficients $C_k = \dfrac{H_k^*}{|H_k|^2 + \sigma_n^2/\sigma_d^2}$ (13) sur chaque composante fréquentielle du signal reçu. Après égalisation fréquentielle et Transformée de Fourier inverse le signal reçu devient le vecteur $\boldsymbol{z}$ égal à :

$$\begin{aligned} \boldsymbol{z} &= \boldsymbol{F}^H.\boldsymbol{C}.\boldsymbol{Y} \\ &= \boldsymbol{F}^H.\boldsymbol{C}.\boldsymbol{\Lambda}.\boldsymbol{d} + \boldsymbol{F}^H.\boldsymbol{C}.\boldsymbol{F}.\boldsymbol{n} \end{aligned} \qquad (13)$$

Ce sont les échantillons de $\boldsymbol{z}$ dans le domaine temporel qui rentrent dans le décodeur de Viterbi.

La figure ci-dessous donne des exemples de résultats de simulation avec une FFT de taille 256, le premier mot OFDM contient des symboles pilotes. On insert un mot OFDM contenant des pilotes tous les dix mots OFDM, ce qui donne une efficacité de remplissage de trame de 9/10.

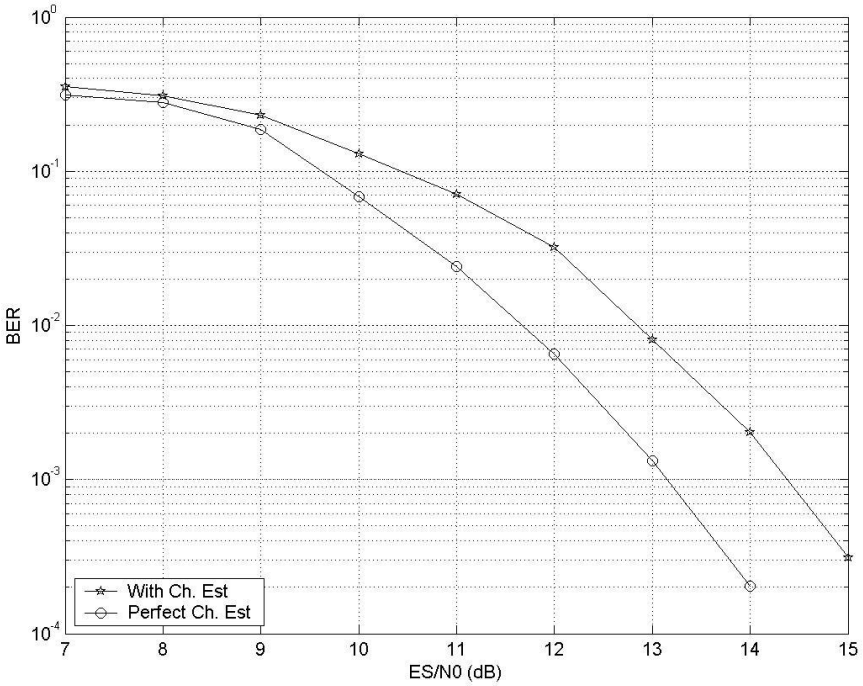


Figure 11 : Exemple de performances sur canal sélectif en fréquence (Nb = 3 bits)

V. ANNEXE : ALGORITHME MAX-LOGMAP

Le procédé de décodage est très semblable à celui des turbo-codes binaires mais il faut utiliser ici des probabilités au niveau des symboles. L'algorithme de décodage pour les treillis non binaires est appelé symbol-by-symbol MAP algorithm.

Le décodeur calcule le rapport de vraisemblance logarithmique $\Lambda(c_t = i)$ de chaque groupe de bits d'information *ct* = *i*. La sortie "soft" $\Lambda(c_t = i)$ est donnée par [25]

$$\Lambda(c_t = i) = \log \frac{Pr\left(c_t = i \middle| \mathbf{r}_1^{\tau}\right)}{Pr\left(c_t = 0 \middle| \mathbf{r}_1^{\tau}\right)} = \log \frac{\sum_{(l,l') \in B_t^i} \alpha_{t-1}(l') \gamma_t^i(l',l) \beta_t(l)}{\sum_{(l,l') \in B_t^0} \alpha_{t-1}(l') \gamma_t^0(l',l) \beta_t(l)} \qquad (14)$$

où *i* représente un groupe de bits d'information dans l'ensemble $\{0,1,2\ldots,2^k - 1\}$, et les probabilités $\gamma_t^i(l',l), \alpha_t(l)$ et $\beta_t(l)$ peuvent être calculées récursivement [26]. Le symbole *i*

avec le rapport de vraisemblance logarithmique le plus élevé dans l'équation (35), $i \in \{0,1,2\ldots,2^k-1\}$ est alors choisi comme symbole décidé (hard decision).

Le gros problème de l'algorithme MAP réside dans l'utilisation importante du calcul d'exponentielles et de multiplications pour calculer les rapports de maximum de vraisemblance. Cela peut conduire à des problèmes d'instabilité. Pour éviter ces problèmes, il est plus avantageux de travailler au niveau des logarithmes des rapports de vraisemblance. On utilise alors les logarithmes de probabilité pour le calcul de $\gamma_t^i(l',l)$, $\alpha_t(l)$ et $\beta_t(l)$. Ces logarithmes de probabilité sont alors notés $\bar{\gamma}_t^i(l',l)$, $\bar{\alpha}_t(l)$ et $\bar{\beta}_t(l)$ respectivement (avec des notations évidentes).L'algorithme de calcul des coefficients $\bar{\alpha}_t(l)$ procède de la façon suivante (algorithme ascendant)

$$\bar{\alpha}_t(l) = \log \sum_{l'=0}^{M_s-1} e^{\bar{\alpha}_{t-1}(l')+\bar{\gamma}_t(l',l)} \qquad (15)$$

avec la condition initiale $\bar{\alpha}_0(0) = 0$ et $\bar{\alpha}_0(l) = -\infty$ pour $l \neq 0$.

et les probabilités $\bar{\beta}_t(l)$ se calculent par l'algorithme descendant

$$\bar{\beta}_t(l) = \log \sum_{l'=0}^{M_s-1} e^{\bar{\beta}_{t+1}(l')+\bar{\gamma}_{t+1}(l',l)} \qquad (16)$$

avec $\bar{\beta}_\tau(0) = 0$ et $\bar{\beta}_\tau(l) = -\infty$ pour $l \neq 0$

avec les probabilité de transition sur le canal données par

$$\bar{\gamma}_t(l',l) = \log \sum_{i=0}^{2^k-1} \bar{\gamma}_t^i(l',l) \qquad (17)$$

et :

$$\bar{\gamma}_t^i(l',l) = \begin{cases} \dfrac{p_t(i)}{p_t(0)} \exp\left(-\dfrac{\left(r_{t,I} - x_{t,I}^i(l)\right)^2 + \left(r_{t,Q} - x_{t,Q}^i(l)\right)^2}{2\sigma^2} \right) & pour\,(l,l') \in B_t^i \\ 0 & otherwise \end{cases} \qquad (18)$$

B_t^i correspond à l'ensemble des transitions $S_{t-1} = l' \rightarrow S_t = l$ lorsque le symbole d'entrée est c_t=i. $r_{t,I}$ et $r_{t,Q}$ sont les composantes en phase et en quadrature du signal reçu. $p_t(i)$ représente la probabilité à priori que le symbole c_t soit égal à i. Les coefficients $\bar{\alpha}_t(l)$ et $\bar{\beta}_t(l)$ dans les équations (15) et (16) peuvent se calculer en utilisant l'algorithme du Jacobien.

$$\log\left(e^{\delta_1} + e^{\delta_2}\right) = \max(\delta_1,\delta_2) + \log\left(1 + e^{-|\delta_2-\delta_1|}\right) \qquad (19)$$
$$= \max(\delta_1,\delta_2) + f_c\left(|\delta_2 - \delta_1|\right)$$

$f_c(.)$ est une fonction de correction que l'on peut implanter en mémoire ROM (look-up table). L'expression $\log\left(e^{\delta_1}+\ldots+e^{\delta_n}\right)$ se calcule de façon récursive en utilisant l'équation (19) par la procédure suivante.

$$\begin{aligned}\log\left(e^{\delta_1}+\ldots+e^{\delta_n}\right) &= \log\left(\Delta+e^{\delta_n}\right), \qquad \Delta = e^{\delta_1}+\ldots+e^{\delta_{n-1}} = e^{\delta} \\ &= \max\left(\log(\Delta),\delta_n\right)+ f_c\left(\left|\log(\Delta)-\delta_n\right|\right) \qquad (20) \\ &= \max\left(\delta,\delta_n\right)+ f_c\left(\left|\delta-\delta_n\right|\right)\end{aligned}$$

Le processus itératif de décodage de l'algorithme symbol-by-symbol MAP est similaire à celui utilisé dans le cadre des turbo-codes binaires mais la nature de l'information échangée entre les deux décodeurs est différente [27]. Dans le cas des turbo-codes binaires on peut décomposer la sortie du décodeur MAP en trois termes: l'information à priori générée par l'autre décodeur, l'information systématique présente à l'entrée du décodeur et l'information extrinsèque générée par le décodeur lors de cette nouvelle itération. L'information extrinsèque dans le cas des turbo-codes binaires est indépendante de l'information à priori et de l'information systématique. Par contre, dans le cas des turbo-modulations codées en treillis, on ne peut plus séparer l'influence de l'information systématique et de l'information systématique, elles sont présentes toutes les deux à l'intérieur d'un symbole codé. Dans ce cas ce ne sera plus l'information extrinsèque seule qui sera échangée entre les deux décodeurs mais l'ensemble de l'information extrinsèque et de l'information systématique. L'ensemble information extrinsèque plus information systématique en sortie du premier décodeur est notée par $\Lambda_{1,es}\left(c_t=i\right)$ et peut s'obtenir par

$$\Lambda_{1,es}\left(c_t=i\right)=\Lambda_1\left(c_t=i\right)-\log\frac{p_t(i)}{p_t(0)} \qquad (21)$$

L'ensemble information extrinsèque plus information systématique $\Lambda_{1,es}\left(c_t=i\right)$ est alors utilisé (après entrelacement) comme probabilité à priori dans le prochain étage de décodage. Après entrelacement cet ensemble est noté : $\tilde{\Lambda}_{1,es}\left(c_t=i\right)$. L'ensemble information extrinsèque plus information systématique du second décodeur est donné par

$$\Lambda_{2,es}\left(c_t=i\right)=\Lambda_2\left(c_t=i\right)-\tilde{\Lambda}_{1,es}\left(c_t=i\right) \qquad (22)$$

Lors de la prochaine itération de décodage, le second terme de droite de l'équation (21) est remplacé par l'ensemble information extrinsèque plus information systématique issu du second étage de décodage noté $\tilde{\Lambda}_{2,es}\left(c_t=i\right)$.

Il faut remarquer que le décodage symbol-by-symbol MAP implique que chaque décodeur ne peut utiliser la même information systématique qu'une seule fois à chaque itération [27]. Pour les schémas de turbo-modulations codées en treillis chaque décodeur reçoit alternativement la sortie bruitée de son propre codeur ce qui implique que le symbole codé reçu en seconde position appartient à l'autre décodeur et doit être traité comme poinçonné.

Par exemple, si on considère ici le premier décodeur, pour chaque signal reçu impair l'algorithme de décodage procède exactement de la même façon que pour les turbo-codes binaires car le décodeur reçoit bien le symbole en provenance de son propre codeur. La seule différence réside dans la nature de l'information transférée au second décodeur qui comprend l'ensemble information extrinsèque plus information systématique. Cependant dans le cas des signaux reçus pairs, le décodeur reçoit le symbole poinçonné dans lequel le symbole de parité est généré par l'autre codeur. Dans ce cas le décodeur ignore ce symbole en mettant à zéro la métrique de transition correspondante dans le treillis. A ce stage la seule entrée dans le treillis est la composante à priori obtenue à partir de l'autre décodeur. Cette composante contient l'information systématique.

CHAPITRE II **CODAGE CORRECTEUR D'ERREUR BASÉ SUR DES FONCTIONS CHAOTIQUES UNIDIMENSIONNELLES**

I. INTRODUCTION

L'idée d'utiliser les signaux chaotiques pour transporter l'information a été proposée par Frey en 1993 [28], Depuis ce temps, l'idée de transmettre de l'information à l'aide de systèmes générateurs de chaos pour mieux la sécuriser a été reprise par de nombreux chercheurs et dans de nombreux domaines des communications numériques comme le cryptage de l'information, le codage de canal ou l'étalement de spectre…Le comportement chaotique de ces systèmes rend leur utilisation très intéressante pour les communication sécurisées.

La principale utilisation des systèmes générateurs de chaos en communications comme en atteste la littérature très abondante [21][29][30][31][32] dans le domaine concerne les systèmes de transmission à étalement de spectre où les séquences d'étalement sont générées à l'aide de systèmes chaotiques et peuvent présenter des propriétés d'intercorrélation plus intéressantes que les séquences d'étalement classiques comme Gold, Walsh-Hadamard, Kasami etc…Un contexte d'utilisation pratique naturel est bien sûr constitué par les réseaux de radiocommunication cellulaire de troisième et quatrième génération (UMTS, HSPA, LTE…).

D'autres applications intéressantes sont apparues récemment pour les systèmes à base de chaos. L'idée de les utiliser pour concevoir des cœurs de canal n'est apparue qu'assez récemment avec les travaux principalement de Kozic et d'Escribano [33][34][35][36][37].

La première approche [38]qui a été retenue consiste à utiliser une fonction de mapping monodimensionnelle non-linéaire inspirée des fonctions de base *mod map, tent map* ou *Bernoulli Shift map* en sortie d'un codeur convolutif classique binaire ou *M*-aire pour obtenir une densité de probabilité des messages transmis telle que les signaux générés ressemblent le plus possible à du bruit blanc (chaos parfait). Les fonctions de mapping retenues dans ce chapitre seront de type discret, i.e. elles seront décrites par une équation du type : $x_{n+1} = f(x_n)$ qui décrit l'évolution de l'état ou de la sortie du système entre deux instants d'échantillonnage consécutifs $n.T_e$ et $(n+1).T_e$. Cependant, le choix d'une fonction de mapping non linéaire peut rendre délicate la tâche du décodeur en réception; en d'autres termes le démapping peut présenter une complexité prohibitive.

C'est cette première approche que nous allons expliciter dans ce chapitre avec différentes fonctions de mapping. Cependant, nous montrerons que les performances obtenues restent assez limitées puisque souvent inférieures à celles d'un système de transmission non-codé BPSK, ceci est dû à l'utilisation d'un mapping monodimensionnel. Les chapitres suivants montreront que l'utilisation de mappings multidimensionnels permet d'augmenter considérablement les performances et de dépasser cette fois ci nettement celles d'un système non codé.

Le plan de ce chapitre est le suivant. Dans la seconde partie nous rappelons le principe d'encodage retenu pour différents types de mapping non-linéaires. La troisième partie est consacré au calcul des performances théoriques des codeurs obtenues. La quatrième partie, quant à lui, est consacré aux algorithmes de décodage évolués qui permettent de tenir compte de la redondance présente entre deux états successifs du codeur pour obtenir une corrélation semblable à celle d'un codeur convolutif. On peut alors utiliser des algorithmes de décodage tels que l'algorithme de Viterbi ou l'algorithme BCJR issu du décodage souple des turbo-codes convolutifs.

II. ENCODAGE DES DONNÉES BINAIRES AVEC DES FONCTIONS CHAOTIQUES DISCRÈTES

Pour encoder une séquence binaire de longueur N contenant l'information, $b_n \in \{0,1\}$, n = 1,…,N nous utilisons une méthode bien connue [39][40][41] qui emploie un effet de troncature. La séquence b_n à transmettre est une séquence de bits de distribution uniforme : $P(b_n = 0) = P(b_n = 1) = 1/2$; les bits b_n sont supposés indépendants i.e. non-corrélés ce qui revient à dire que : $E\{b_n . b_{n+m}\} = \delta_{0,m}$ avec : $\begin{matrix} \delta_{0,m} = 0 \text{ si } m \neq 0 \\ \delta_{0,m} = 1 \text{ si } m = 0 \end{matrix}$.

Les mappings utilisés dans ce chapitre sont limités à l'intervalle [0 ,1], pour plus de facilité dans les calculs de SNR's. La fonction de mapping vérifie donc les propriétés suivantes :

$$f(x) : [0,1] \rightarrow [0,1] \quad (1)$$

Nous allons étudier d'abord les propriétés de deux mappings bien connues à savoir le mapping appelé fonction Bernoulli Shift map et le mapping dit logistic map modifiée

(modified logistic map). L'expression de la fonction de correspondance ou de mapping Bernoulli Shift map est donnée par :

$$x_{n+1} = f(x_n) = \begin{cases} 2x_n & \text{si} \quad x_n < \frac{1}{2} \\ 2x_n - 1 & \text{si} \quad x_n \geq \frac{1}{2} \end{cases} \qquad (2)$$

La figure 1 montre le diagramme de la fonction Bernoulli Shift map que l'on peut résumer par l'équation mathématique : $f(x) = 2.x \mod 1$.

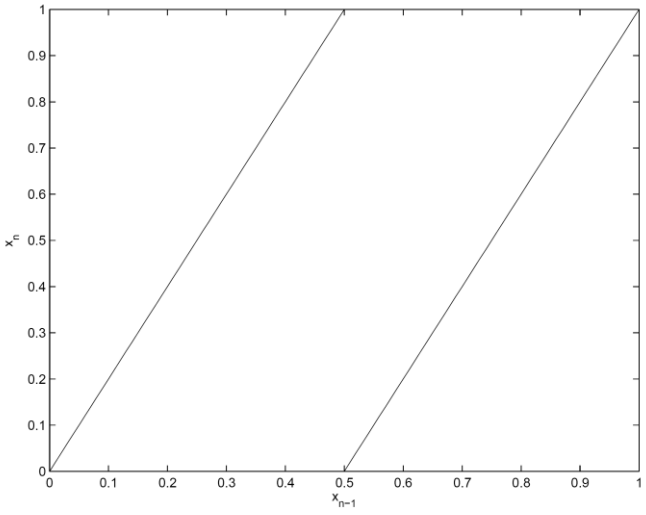


Figure 1 : diagramme de la fonction Bernoulli Shift map

L'expression de la fonction de correspondance ou de mapping logistic map modifiée (figure 2), est donnée quant à elle par :

$$x_{n+1} = f(x_n) = \begin{cases} 4x_n(1-x_n) & \text{si} \quad x_n < \frac{1}{2} \\ 1-4x_n(1-x_n) & \text{si} \quad x_n \geq \frac{1}{2} \end{cases} \qquad (3)$$

Sa représentation graphique est donnée sur le graphe de la figure 2.

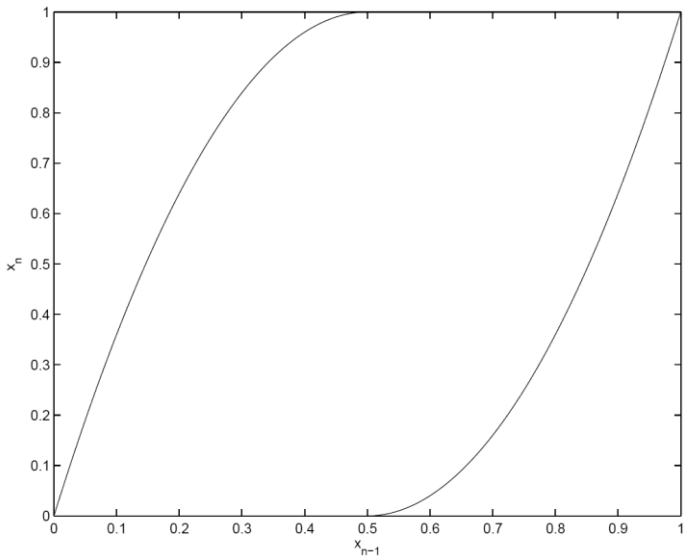


Figure 2 : diagramme de la fonction logistic map modifiée

L'encodage d'une séquence de bits à transmettre $[b_0, b_1, b_2, ..., b_{N-1}, b_N]$ se réalise alors de la façon suivante. Dans le cas de la fonction Bernoulli Shift map, on définit l'état symbolique du système par :

$$r = \sum_{n=0}^{N} b_n . 2^{-n} \quad (4)$$

On définit de même l'état initial de la séquence chaotique générée par : $x_0 = r$. Il suffit alors d'appliquer la règle d'encodage donnée par l'équation (2) pour obtenir la séquence chaotique de sortie de l'encodeur. On peut récupérer les bits d'information transmis par la formule très simple :

$$b_n = \lfloor x_n + 1/2 \rfloor \quad (5)$$

Où $\lfloor x \rfloor$ représente la partie entière de *x* c'est-à-dire le plus grand entier plus petit que *x*.

La même règle d'encodage et de décodage donnée par (5) peut encore s'appliquer dans le cas de la fonction logistic map modifiée à condition de prendre ici :

$$x_0 = \cos^2[\frac{\pi}{2}.(1-r)] \quad (6)$$

Lorsque l'on peut appliquer les mêmes règles d'encodage et de décodage pour des fonctions de mapping différentes, on dit que les fonctions de mapping en question sont conjuguées [40]. Ainsi, les fonctions Bernoulli Shift map et logistic map modifiée sont bien conjuguées à

condition de prendre : $x_0 = r$ pour la correspondance Bernoulli Shift map et $x_0 = \cos^2[\frac{\pi}{2}.(1-r)]$ pour la correspondance logistic map modifiée.

On a de plus la propriété fondamentale suivante extraite de [42]. Lorsque deux fonctions de mapping sont conjuguées, en particulier lorsque l'on cherche une fonction de mapping conjuguée avec la fonction de correspondance Bernoulli Shift map, la fonction d'initialisation $g(r)$ peut être calculée à partir de la densité de probabilité $p(x)$ supposée connue de la fonction de mapping utilisée grâce à la formule : $g(r) = F^{-1}(r)$ avec : $F(r) = \int_0^x p(u).du$ qui représente la fonction de répartition (l'intégrale de la densité de probabilité) de la fonction de mapping ou de façon équivalente la fonction de répartition des symboles transmis par le codeur.

Par exemple, on sait que la p.d.f des symboles transmis est uniforme dans le cas de la fonction Bernoulli Shift map, i.e. $p(x) = 1,\ \forall x \in [0,1]$ tandis qu'elle est égale à : $p(x) = \frac{1}{\pi.\sqrt{x.(1-x)}},\ \forall x \in [0,1]$ pour la fonction logistic map modifiée. La fonction de répartition de la correspondance logistic map modifiée est alors égale à : $F(x) = 1-(2/\pi).\cos(\sqrt{x})$ et on peut vérifier facilement que : $g(r) = F^{-1}(r)$. La propriété est immédiate pour la fonction Bernoulli Shift map.

Il peut être intéressant, à partir de ces propriété, de vouloir construire des fonctions de correspondance performantes qui permettent d'obtenir de meilleures résultats qu'avec les fonctions Bernoulli Shift map et logistic map modifiée. Nous allons en donner un exemple concret dans les lignes qui suivent.

Pour cela, Escribano & al [33] ont montré qu'il fallait rechercher des fonctions de mapping antisymétriques par rapport au point $x = 0.5$. Ceci s'explique par le fait que les fonctions de mapping symétriques comme la tent map ou la logistic map conduisent à la synthèse de codeurs avec des distances libres médiocres (voire nulles) [43]. Partant de cette remarque, on peut chercher une fonction telle que sa p.d.f soit maximale en 0.5 ; la fonction recherchée possèdera la même forme de p.d.f que la fonction logistic map modifiée mais sera translatée par rapport à cette dernière. On obtient alors pour la p.d.f souhaitée :

$$\begin{aligned} p(x) &= \frac{1}{\pi.\sqrt{(x+1/2).(1/2-x)}} \quad \forall x \in [0,1/2] \\ p(x) &= \frac{1}{\pi.\sqrt{(x-1/2).(3/2-x)}} \quad \forall x \in]1/2,1] \end{aligned} \tag{7}$$

Nous allons alors chercher le mapping correspondant à cette p.d.f pour que la nouvelle fonction de correspondance ainsi construite soit conjuguée à la fonction Bernoulli Shift map ce qui permet de conserver la règle de décodage donnée en (5). Pour trouver la nouvelle fonction de mapping, on peut se servir de la règle de construction suivante extraite de [42]: lorsque la fonction de correspondance cherchée est conjuguée de la fonction Bernoulli shift map, si la condition initiale d'encodage pour la fonction Bernoulli shift map s'écrit : $x_0^B = r$ et si la condition initiale d'encodage pour la fonction de mapping conjuguée s'écrit : $x_0^C = r$, alors on doit avoir la relation suivante pour chacun des symboles générés :

$$f_C^n(x_0^C) = f_C^n[g(r)] = g[f_B^n(r)] = g[f_B^n(x_0^C)] \quad (8)$$

Où $f_C^n[x]$ désigne la nième itération de la fonction de mapping conjuguée de la fonction de Bernoulli et $f_B^n[x]$ est la nième itération de la fonction de mapping Bernoulli shift map. Cette dernière égalité est la conséquence directe de la définition des fonctions de mapping conjuguées puisque l'on a (si f_1 et f_2 désignent les deux fonctions de mapping conjuguées) la relation : $f_1(x) = \phi\{f_2[\phi^{-1}(x)]\}$ où la fonction $\phi(x)$ désigne une bijection de l'intervalle [0, 1] sur lui-même. A partir des relations (7) et (8) et en tenant compte de la symétrie de la p.d.f par rapport au point 0.5, $g(r)$ possède les propriétés suivantes :

$$\begin{aligned} &g(0) = 0, \quad g(\tfrac{1}{2}) = \tfrac{1}{2}, \quad g(1) = 1, \\ &g([0, \tfrac{1}{2}]) = [0, \tfrac{1}{2}], \quad g([\tfrac{1}{2}, 1]) = [\tfrac{1}{2}, 1] \end{aligned} \quad (9)$$

Où $g(r)$ représente une fonction monotone croissante. On a également de façon simple les résultats suivants :

$$f_C[g(r)] = \begin{cases} g(2.r) & \text{si } r \le 0.5 \\ g(2.r) & \text{si } r > 0.5 \end{cases} \quad (10)$$

De plus, en tenant compte du fait que $g(r)$ est inversible et en posant $g(r) = x$ et $r = g^{-1}(x)$, on obtient :

$$f_C[x] = \begin{cases} g(2.g^{-1}(x)) & \text{si } x \le 0.5 \\ g(2.g^{-1}(x)) & \text{si } x > 0.5 \end{cases} \quad (11)$$

Ceci donne l'expression de la nouvelle fonction de mapping appelée flip-flop map map. Quelques calculs mathématiques simples conduisent alors au résultat final :

$$F(x)=\begin{cases}\frac{1}{2}-\frac{2}{\pi}.\cos(\sqrt{x+\frac{1}{2}}) & \text{si } x\le\frac{1}{2}\\ \frac{3}{2}-\frac{2}{\pi}.\cos(\sqrt{x-\frac{1}{2}}) & \text{si } x>\frac{1}{2}\end{cases} \quad (12)$$

$$g(r)=\begin{cases}\cos^2[\frac{\pi}{2}.(\frac{1}{2}-r)]-\frac{1}{2} & \text{si } x\le\frac{1}{2}\\ \cos^2[\frac{\pi}{2}.(\frac{3}{2}-r)]+\frac{1}{2} & \text{si } x>\frac{1}{2}\end{cases} \quad (13)$$

Et :

$$f(x_n)=\begin{cases}4.x_n.\sqrt{\frac{1}{4}-x_n^2} & \text{si } 0\le x_n<\frac{1}{2.\sqrt{2}}\\ 1-4.x_n.\sqrt{\frac{1}{4}-x_n^2} & \text{si } \frac{1}{2.\sqrt{2}}\le x_n<\frac{1}{2}\\ 4.(1-x_n).\sqrt{\frac{1}{4}-(1-x_n)^2} & \text{si } \frac{1}{2}\le x_n<1-\frac{1}{2.\sqrt{2}}\\ 1-4.(1-x_n).\sqrt{\frac{1}{4}-(1-x_n)^2} & \text{si } 1-\frac{1}{2.\sqrt{2}}\le x_n\le 1\end{cases} \quad (14)$$

La nouvelle fonction de mapping flip-flop map est représentée sur la figure 3 ci-dessous :

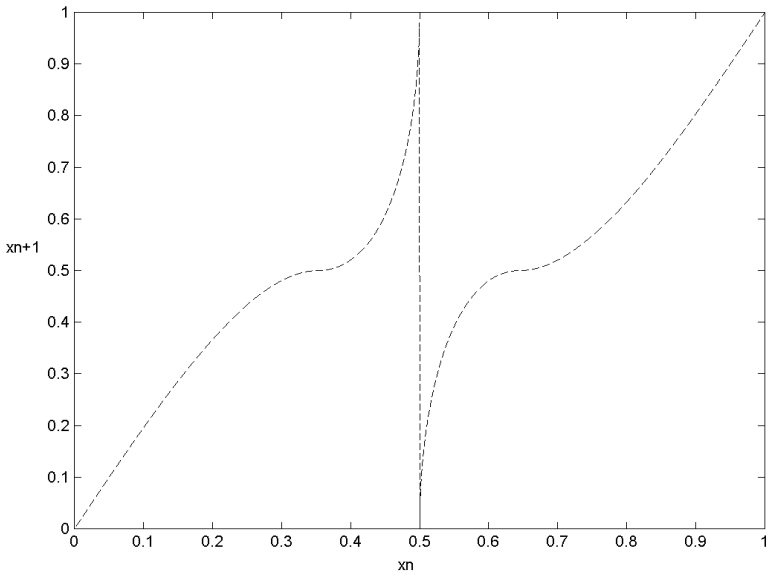


Figure 3 : diagramme de la fonction flip-flop map

Les différentes fonctions $g(x)$ sont représentées sur la figure 4 ci-dessous :

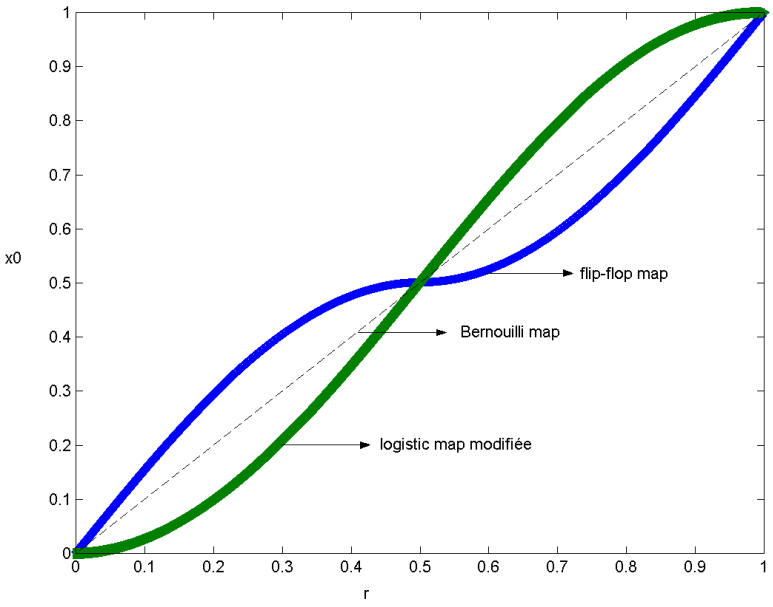


Figure 4 : Les différentes fonctions $x_0 = g(r)$ *pour les trois fonctions de mapping proposées*

Remarques fondamentales :

- Dans le cas d'un système réel il est clair que la taille N des messages à transmettre atteint très rapidement plusieurs dizaine de milliers de bits ; dans ce cas la règle d'encodage proposée n'est plus applicable car elle suppose une précision infinie. Pour pouvoir continuer à encoder les messages transmis une méthode simple consiste cependant à partitionner les messages en blocs de D bits et à ensuite encoder chacun de ces blocs en suivant la règle de codage donnée précédemment. On aura dans ce cas :

$$\begin{aligned} r_n^{'} &= \sum_{m=n}^{n+D-1} b_m.2^{-m+n-1} \\ x_n^{'} &= g(r_n^{'}) \end{aligned} \qquad (15)$$

Par rapport à une séquence codée x_n obtenue avec une précision infinie $D \to +\infty$, on peut toujours écrire :

$$x_n^{'} = x_n + \eta_n \qquad (16)$$

Où $\eta_n = \vartheta(2^{-D})$ est le bruit de quantification que l'on considère approximativement comme un bruit blanc dont la puissance décroit rapidement avec *D*. Dans la pratique une valeur de *D* comprise entre 15 et 20 sera suffisante pour considérer l'effet de la troncature comme négligeable. Ce fait sera justifié par l'évaluation théorique du TEB dans le chapitre suivant.

- On peut toujours donner une interprétation de l'encodage avec des fonctions chaotiques à l'aide d'un schéma basé sur le principe des modulations codées en treillis. La séquence des bits à transmettre est codée à l'aide de deux étapes :

  Première étape : calcul du nouvel état z_n à partir de l'ancien état z_{n-1}

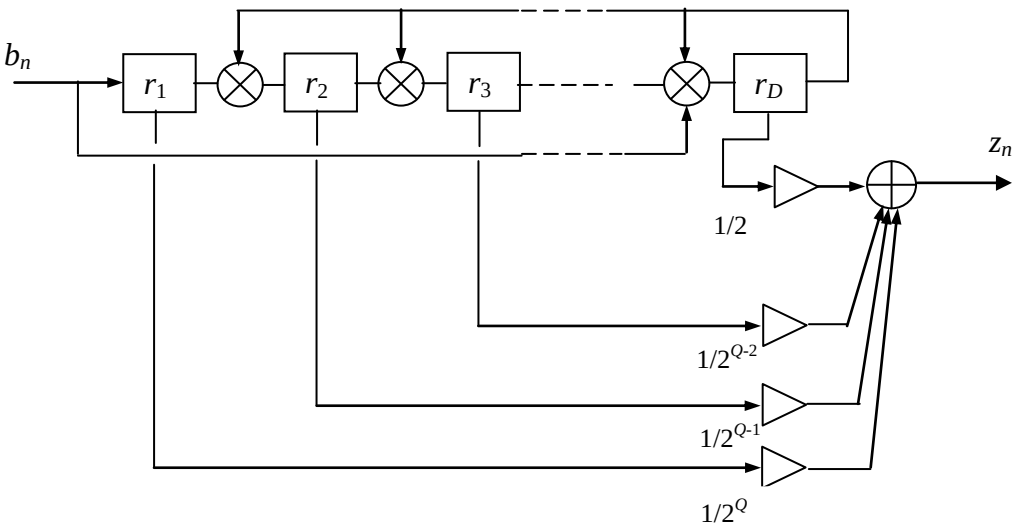
  $$z_n = f(z_{n-1}, b_n) + b_n / 2^{-D}$$

  Deuxième étape : calcul de la sortie

  $$x_n = g(z_n) = 2.z_n - 1$$

  Avec : $f(.,0) = f_0(.)$ et $f(.,1) = f_1(.)$ qui correspondent aux deux tronçons de mapping situés de part et d'autre du point 0.5 et qui laisse l'intervalle [0, 1] invariant.

Une représentation sous la forme d'un codage convolutif peut alors être obtenue comme le montre le schéma ci-dessous :



III. DÉCODAGE ET PERFORMANCES THÉORIQUES.

Le canal de propagation considéré pour évaluer la performance de système proposé est un canal à bruit additif blanc Gaussien. Dans ce cas si la séquence envoyée sur le canal est x_n, la séquence reçue y_n s'écrira sou la forme :

$$y_n = x_n + n_n \quad (17)$$

Avec n_n qui est un bruit additif blanc Gaussien de moyenne nulle et de variance σ^2. La densité de probabilité ou p.d.f d'un tel bruit prend la forme bien connue :

$$p(x) = \frac{1}{\sqrt{2.\pi}.\sigma}.e^{-x^2/\sigma^2} \quad (18)$$

Quant au modèle de transmission incluant le canal de propagation il est illustré sur la figure 5 ci-dessous.

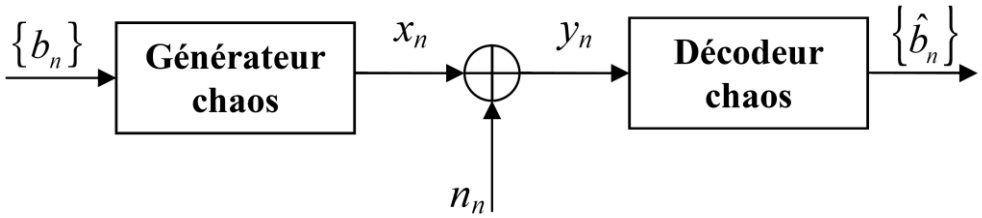


Figure 5 : Schéma de transmission à base d'encodeur chaotique

Connaissant la règle de décodage très simple proposée en équation (5) il est possible, dans un premier temps, de décoder le message reçu symbole par symbole en utilisant la comparaison suivante (hard decoding) :

$$\begin{aligned} y_n < \frac{1}{2} &\rightarrow \hat{b}_n = 0 \\ y_n \geq \frac{1}{2} &\rightarrow \hat{b}_n = 1 \end{aligned} \quad (19)$$

Il est clair que les performances obtenues seront médiocres car on ne tient absolument pas compte de la redondance qui existe entre deux symboles transmis consécutifs. Cependant, sachant que l'on connaît les p.d.f's de chaque fonction de mapping, il est facile dans ce contexte de calculer une expression exacte de la probabilité d'erreur sur canal AWGN.

$$P_e = \text{Proba}(y_n < \frac{1}{2}, x_n \geq \frac{1}{2}) + \text{Proba}(y_n \geq \frac{1}{2}, x_n < \frac{1}{2}) \quad (20)$$

Par symétrie on obtient immédiatement :

$$\begin{aligned} P_e &= 2.\text{Proba}(y_n \geq \frac{1}{2}, x_n < \frac{1}{2}) = 2.\text{Proba}(n_n \geq \frac{1}{2} - x_n, x_n < \frac{1}{2}) \\ &= 2.\int_0^{1/2} p(x).\int_{1/2-x}^{+\infty} p(r).dr.dx = \int_0^{1/2} p(x).erfc[(\frac{1}{2}-x).\frac{1}{\sqrt{2}.\sigma}].dx \end{aligned} \quad (21)$$

En tenant compte de la relation bien connue : $\int_x^{+\infty} p(r).dr = \frac{1}{2}.erfc(\frac{x}{\sqrt{2}.\sigma})$ (22). Le résultat de (21) pour les différents mapping testés s'écrit :

- Bernoulli map : $P_e^B = \frac{1}{2}.erfc[\sqrt{\frac{3.E_b}{N_0}}] + \frac{1}{\sqrt{12.\pi.\frac{E_b}{N_0}}}.(1-e^{\frac{-3.E_b}{N_0}})$ (23)
- Logistic map modifiée : $P_e^M = \int_0^{1/2} \frac{1}{\pi.\sqrt{x.(1-x)}}.erfc[(1-2.x).\sqrt{\frac{2.E_b}{N_0}}].dx$ (24)
- Flip-flop map : $P_e^F = \int_0^{1/2} \frac{1}{\pi.\sqrt{(x+\frac{1}{2}).(\frac{1}{2}-x)}}.erfc[(1-2.x).\sqrt{\frac{2}{3-\frac{8}{\pi}}.\frac{E_b}{N_0}}].dx$ (25)

La quantité $\frac{E_b}{N_0}$ désigne classiquement le rapport de l'énergie par bit transmis sur la densité spectrale du bruit additif blanc. Dans notre cas, nous avons : $\frac{E_b}{N_0} = P/(2.\sigma^2)$ où σ^2 désigne la puissance du bruit additif. La quantité *P* quant à elle représente l'énergie utile par symbole transmis, elle varie bien entendu selon le mapping utilisé. Dans le cas de la fonction Bernoulli map, en supposant *D* (voir équation (15)) suffisamment grand pour qu'on puisse négliger le bruit de quantification (i.e. $\eta_n \to 0$ dans l'équation (16)), on obtient : $\sigma_x^2 = \frac{1}{12}$. De même, avec les mêmes hypothèses, on trouve $\sigma_x^2 = \frac{1}{8}$ pour la fonction logistic map modifiée et

$\sigma_x^2 = \frac{3}{8} - 1/\pi$ pour la fonction flip-flop map. Les résultats des calculs (23-25) sont illustrés sur la figure 6. Il apparaît clairement sur cette courbe que les systèmes à base de mapping conjugués de la fonction Bernoulli Shift map ne sont absolument pas compétitifs par rapport à un système BPSK non codé. Le système BPSK non codé est particulièrement intéressant à étudier en ce sens qu'il représente un cas limite des systèmes étudiés avec une densité de probabilité composée de deux Diracs centrés respectivement en 0 et en 1. Au niveau de la comparaison entre les mapping conjugués de la fonction Bernoulli Shift map, c'est la fonction flip-flop map qui présente les plus mauvais résultats. Ceci est dû au fait que cette fonction possède le plus grand nombre de points autour du seuil 1/2 de tous les mapping utilisés. On peut constater aussi que la fonction logistic map modifiée permet d'améliorer légèrement les performances du mapping Bernoulli Shift map; ceci se justifie par une densité de points moindre autour du seuil 1/2.

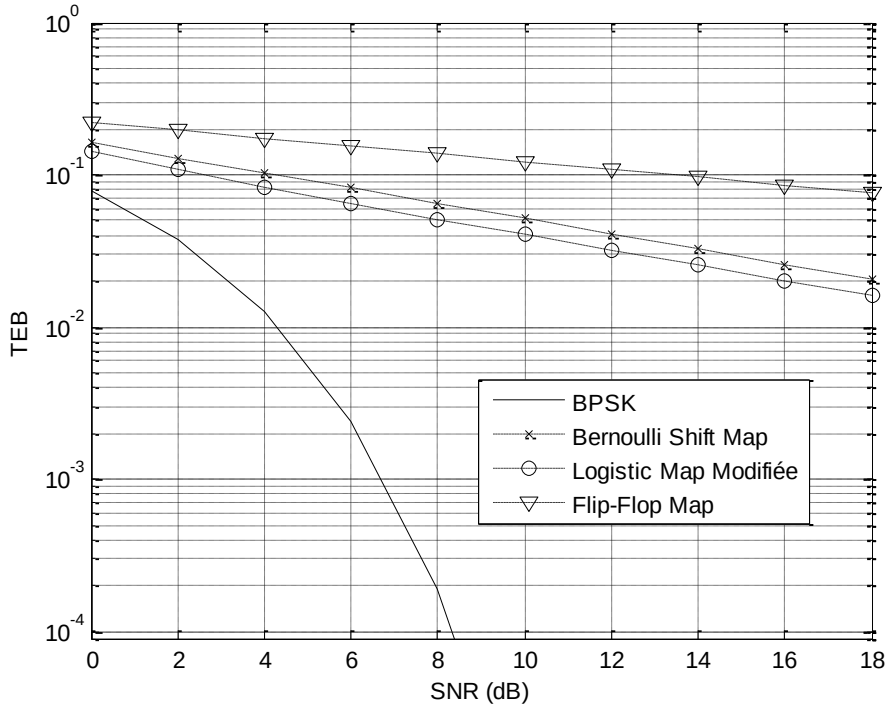


Figure 6 : Performances des différentes fonctions de mapping sur canal AWGN

Il faut bien comprendre que ces performances sont dans l'ensemble très médiocres car l'algorithme de décodage très simple présenté en (5) n'exploite aucune redondance entre les données transmises. Des versions plus sophistiquées d'algorithmes de décodage conduiront à de bien meilleures performances.

Il est également possible de caractériser l'effet de troncature qui consiste à opérer sur des blocs de taille réduite *D*. En effet, les calculs (23-25) supposent $D \to +\infty$. En fait, tenant compte d'une valeur finie de *D*, on peut écrire la densité de probabilité (pdf) de la fonction Bernoulli Shift map sous la forme :

$$p(x) = \sum_{i=0}^{2^D-1} \frac{1}{2^D}.\delta(x - \frac{i}{2^D}) \quad (26)$$

L'écriture (26) ci-dessus s'interprète très facilement, cela signifie que chacun des 2^D échantillons possède une probabilité $\frac{1}{2^D}$ (pour *D* assez grand, cette densité de probabilité tend vers la densité uniforme dans l'intervalle [0, 1]). En utilisant la règle (21) on arrive alors à :

$$P_e = \frac{1}{2^D}.\sum_{i=0}^{2^{D-1}-1} \frac{1}{2}.erfc\left[\left(\frac{1}{2} - \frac{i}{2^D}\right).\frac{1}{\sigma.\sqrt{2}}\right] + \frac{1}{2^D}.\sum_{i=2^{D-1}}^{2^D-1}\left\{1 - \frac{1}{2}.erfc\left[\left(\frac{1}{2} - \frac{i}{2^D}\right).\frac{1}{\sigma.\sqrt{2}}\right]\right\} \quad (27)$$

Pour traduire cette expression (27) en termes de rapport signal à bruit : E_b / N_0, il suffit de remarquer que l'on a la relation :

$$\sigma_x^2 = \frac{1}{12}.\frac{2^{2D} + 2^{D+3} - 3}{2^{2D}} \quad (28)$$

Il est clair également d'après (27) que dans le cas où $\sigma \to 0$ il reste toujours un plancher de TEB égal à : $\frac{1}{2^{D+1}}$. Cependant, en pratique, des valeurs de *D* comprises entre 15 et 20 permettent de réduire ce plancher à des valeurs tellement faibles qu'elles pourront être négligées en pratique.

IV. ALGORITHMES DE DÉCODAGE AVANCÉS

Pour améliorer les performances du premier algorithme de décodage proposé il est clair qu'il faut exploiter la redondance présente dans les messages successifs transmis. Il est ainsi facile de constater que chaque symbole x_n a *D* - 1 bits en commun avec les symbole x_{n-1} et x_{n+1}, *D* - 2 bits en commun avec les symboles x_{n-2} et x_{n+2} et ainsi de suite …La première méthode de décodage est une méthode que l'on pourrait qualifier de méthode ad-hoc qui tient compte

de certaines propriétés particulières des systèmes chaotiques à savoir que deux trajectoires issues de conditions initiales très proches vont diverger rapidement et de même deux trajectoires issues de conditions initiales différentes peuvent converger vers le même état. Ainsi, pour le décodage du symbole : y_n, on peut considérer le symbole y_{n+M-1} et reconstituer toutes les trajectoires possibles depuis l'instant *n* qui conduisent à la valeur y_{n+M-1}. A chaque instant d'échantillonnage on a deux expressions possibles pour les fonctions de mapping selon que x_n est plus grand ou plus petit que 0.5. On peut donc voir facilement que l'on aura en tout 2^{M-1} trajectoires possibles pour remonter depuis y_{n+M-1} jusqu'à y_n. Par exemple, dans le cas de la fonction Bernoulli Shift map, on a les deux antécédents possibles :

$$\begin{aligned} z_n^0 &= f_1^{-1}(y_{n+1}) = \frac{y_{n+1}}{2} \\ z_n^1 &= f_2^{-1}(y_{n+1}) = \frac{y_{n+1}}{2} + \frac{1}{2} \end{aligned} \qquad (29)$$

Attention, on suppose bien entendu que toutes les valeurs y_{n+M-1} ont été normalisées pour rester dans l'intervalle [0, 1] pour des raisons de stabilité puisque les fonctions de mapping sont toutes définies dans cet intervalle. On a donc la règle de saturation :

$$\begin{aligned} y_{n+M-1} > 1 &\rightarrow y_{n+M-1} = 1 \\ y_{n+M-1} < 0 &\rightarrow y_{n+M-1} = 0 \end{aligned} \qquad (30)$$

Dans le cas de la fonction logistic map modifiée et toujours avec *M* = 2, les deux antécédents s'écrivent :

$$\begin{aligned} z_n^0 &= f_1^{-1}(y_{n+1}) = \frac{1}{2} - \frac{1}{2}\sqrt{1 - y_{n+1}} \\ z_n^1 &= f_2^{-1}(y_{n+1}) = \frac{1}{2} + \frac{1}{2}\sqrt{y_{n+1}} \end{aligned} \qquad (31)$$

Pour le cas de la fonction flip-flop map, on obtient (cf (14)) non pas deux mais quatre antécédents :

Si $y_{n+1} < 1/2$:

$$\begin{aligned} z_n^0 &= \frac{1}{2}\sqrt{\frac{1}{2} - \frac{1}{2}\sqrt{1-4.y_{n+1}^2}} \\ z_n^1 &= 1 - \frac{1}{2}\sqrt{\frac{1}{2} - \frac{1}{2}\sqrt{1-4.y_{n+1}^2}} \end{aligned} \quad (32)$$

Si $y_{n+1} \geq 1/2$:

$$\begin{aligned} z_n^0 &= \frac{1}{2}\sqrt{\frac{1}{2} + \frac{1}{2}\sqrt{1-4.(1-y_{n+1})^2}} \\ z_n^1 &= 1 - \frac{1}{2}\sqrt{\frac{1}{2} - \frac{1}{2}\sqrt{1-4.(1-y_{n+1})^2}} \end{aligned} \quad (33)$$

On note les séquences $\left\{z_k^{s_l}\right\}$ résultant reconstruites avec : $k = n \ldots n+M-1$, $l = 1 \ldots 2^{M-1}$. Le vecteur $\boldsymbol{s}_l$ définit la $l^{ième}$ trajectoire sous la forme : $(01100...01)$ avec à chaque fois 0 ou 1 qui identifie le $k^{\text{ième}}$ bit obtenu en remontant les trajectoires avec les règles (29-33). La trajectoire décidée ou trajectoire à Maximum de Vraisemblance est celle qui conduit à la métrique cumulée : $\sum_{k=n}^{n+M-1}\left(z_k^{s_{d_n}} - y_k\right)^2$ la plus petite parmi toutes les trajectoires possibles, i.e. elle vérifie:

$$\sum_{k=n}^{n+M-1}\left(z_k^{s_{d_n}} - y_k\right)^2 \leq \sum_{k=n}^{n+M-1}\left(z_k^{s_{n_l}} - y_k\right)^2 \quad (34)$$

Avec : $(m = 1 \ldots 2^{M-1})$ et $d_n \in \left\{1 \ldots 2^{M-1}\right\}$. Le bit décidé $\hat{b}_n$ est obtenu selon la règle de décision :

$$\begin{aligned} z_n^{s_{d_n}} < 1/2 &\rightarrow \hat{b}_n = 0 \\ z_n^{s_{d_n}} \geq 1/2 &\rightarrow \hat{b}_n = 1 \end{aligned} \quad (35)$$

Comme on le voit, ce premier algorithme tire parti de la redondance entre y_n et les M symboles qui suivent. Par contre, il ne tient pas compte de la redondance entre y_n et l'ensemble de la séquence transmise et ceci pénalise fortement les performances. Pour remédier à cet inconvénient on peut penser à appliquer l'algorithme récursivement. Une fois que la séquence la plus vraisemblable $z_n^{s_{d_n}}$ a été calculée pour chaque valeur de n = 1,...,N, le

même algorithme déjà décrit précédemment peut être appliqué en remplaçant y_n par la séquence $z_n^{s_{d_n}}$ correspondante. De cette manière, il est clair que la redondance présente dans la séquence complète depuis y_n est propagée pour être utilisée lors de la prochaine itération. Ce nouvel algorithme peut être qualifié de méthode ad-hoc récursive. En fait, ces deux algorithmes ad hoc fonctionnent assez bien à forts SNR's mais comportent un plancher d'erreur qui dépend entre autres de la longueur du bloc M et du nombre d'itérations

Pour améliorer encore les performances on peut également recourir à des algorithmes puissants classiques comme l'algorithme de Viterbi basé sur le principe du maximum de vraisemblance (ML) ou l'algorithme BCJR basé sur le critère du Maximum de Vraisemblance à Posteriori (MAP). Ces deux algorithmes opèrent ici par fenêtre glissante. Pour leur mise en œuvre il est nécessaire de procéder à une étape de quantification. Ainsi les symboles x_k transmis subissent une étape de quantification avant d'être envoyés sur le canal. Cette étape de quantification consiste à partager l'intervalle [0, 1] en une série d'intervalles disjoints I_i de la forme $[i/P,(i+1)/P[$ pour i = 0, 1,…,P-1, le centre de ces intervalles étant de la forme : $c_i = i/P + 1/2.P$. P représente le nombre d'intervalles que l'on prend en pratique égal à une puissance de 2 pour simplifier l'implémentation des algorithmes de Viterbi et du BCJR. De cette façon le seuil 0.5 représentera à la fois le point le plus élevé dans un intervalle donné et le point le moins élevé dans un autre intervalle. Ainsi le fait de savoir si un point x_k se trouve dans un intervalle I_i permet d'affecter la valeur 0 ou 1 au bit à décoder correspondant. Si l'on substitue la séquence originelle à transmettre par la séquence des intervalles où se trouvent les symboles correspondants on obtient une représentation symbolique de la séquence transmise qui peut être décrite comme un processus de Markov du premier ordre avec une matrice de transition $\boldsymbol{T}$. Le terme $t_{i,j}$ représente la probabilité de transition entre l'intervalle i et l'intervalle j. Dans le cas de la fonction de mapping Bernoulli Shift map, l'opération de multiplication modulo 1 se traduit par une probabilité de transition égale à 0.5 depuis l'intervalle d'origine vers deux intervalles adjacents. Par exemple, dans le cas P = 4, on obtient la matrice de transition :

$$\boldsymbol{T} = \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} \quad (36)$$

Dans le cas de fonctions de mapping plus sophistiquées il est plus difficile de définir des matrices de transition. On peut cependant y parvenir de la façon suivante : la probabilité de transition entre les états correspondants aux intervalles I_i et I_j est égal au quotient de la longueur de l'intersection entre l'intervalle image de la fonction de mapping $f(I_i)$ et l'intervalle I_j et de la longueur de l'intervalle : $f(I_i)$.

$$t_{i,j} = \frac{\text{longueur}[f(I_i) \cap I_j]}{\text{longueur}[f(I_i)]} \quad (37)$$

On considère alors la séquence symbolique associée aux symboles x_k d_k^i, $k = 0,...,L-1$, où $d_k^i = c_i$ (0 ou 1) selon que x_k appartient ou non à l'intervalle I_i. L représente la longueur du block nécessaire (largeur de la fenêtre glissante) au décodage (à l'émergence d'un chemin) d'un symbole x_k. De la même façon, on affirmera que le processus de Markov est dans l'état $s_k = i$ aux instants $k = 1,...,L$ si x_k. appartient à l'intervalle : I_i. L'état de départ est pris égal à 0 tandis que, au fur et à mesure que l'algorithme de Viterbi (ou l'algorithme BCJR) se déroule, l'état initial est remis à jour d'après les valeurs obtenues lors de l'étape précédente.

V. RÉSULTATS DE SIMULATION

Les résultats sont donnés sur les figures 7 à 14 pour les différents algorithmes présentés précédemment et pour différents paramètres. Dans tous les cas, on prend D = 20 bits par symbole. De cette façon on s'assure que la différence entre x_n et $x_n^{'}$ est négligeable. Dans tous les cas de figure c'est la fonction de mapping Bernoulli shift map qui possède les meilleures performances devant la fonction logistique map modifiée (performances assez prôches) et devant la fonction flip-flop map (de loin la moins performlante). La figure 7 représente les résultats de simulation obtenus pour la méthode ad-hoc en faisant varier la taille des blocs M dans le cas où on utilise la fonction de mapping Bernoulli shift map. Les figures 8 et 9 illustrent les performances obtenues avec ce même algorithme respectivement dans le cas des mapping avec la fonction logistic map modifiée et la fonction flip-flop map.

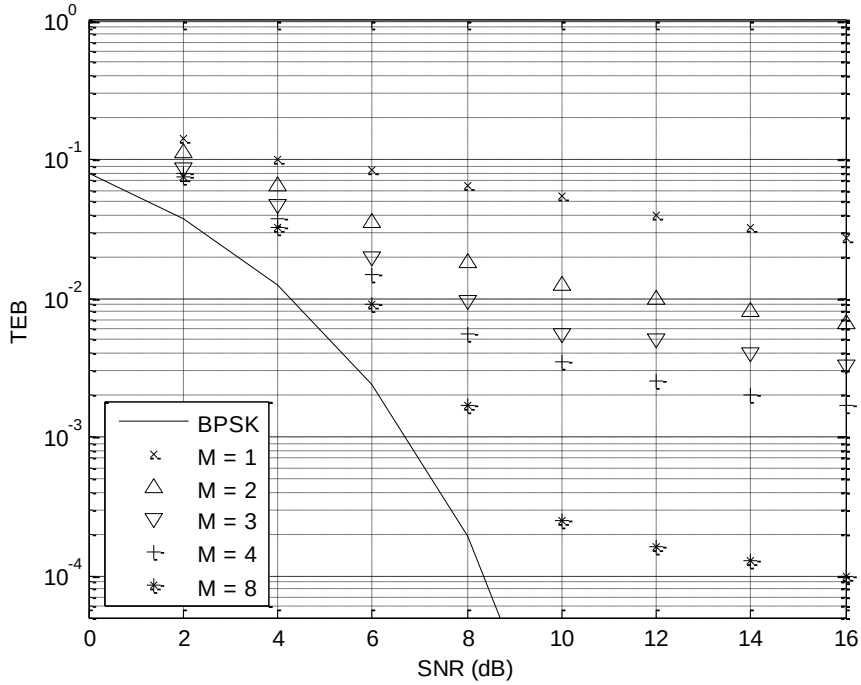


Figure 7 : Résultats de TEB pour la fonction de mapping Bernoulli shift map avec l'algorithme de décodage ad-hoc pour D = 20 bits et différentes valeurs de M : M = 1, 2, 3, 4 symboles

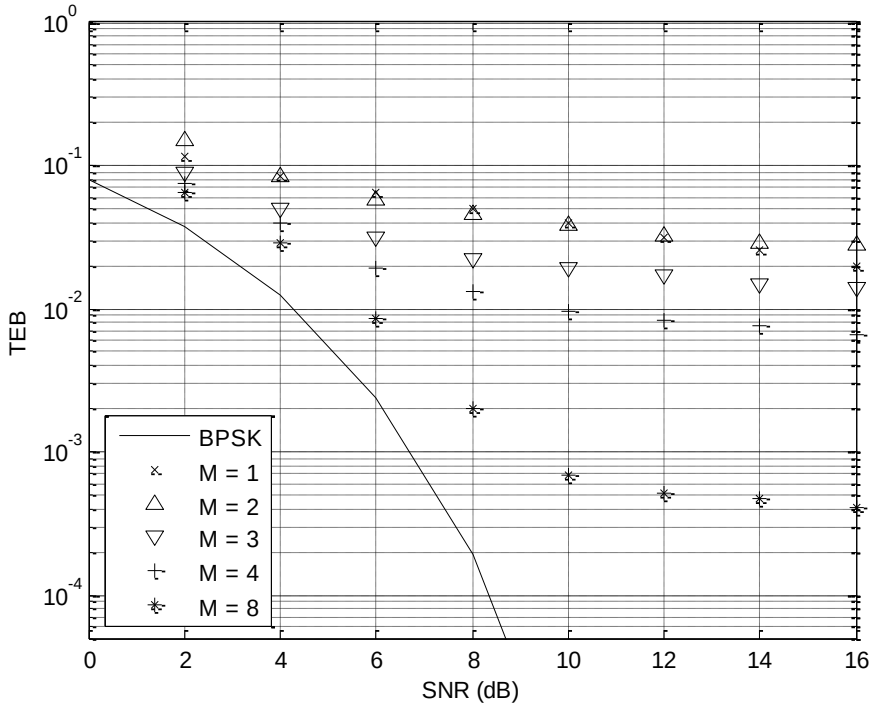


Figure 8 : Résultats de TEB pour la fonction logistic map modifiée avec l'algorithme de décodage ad-hoc pour D = 20 bits et différentes valeurs de M : M = 1, 2, 3, 4 symboles

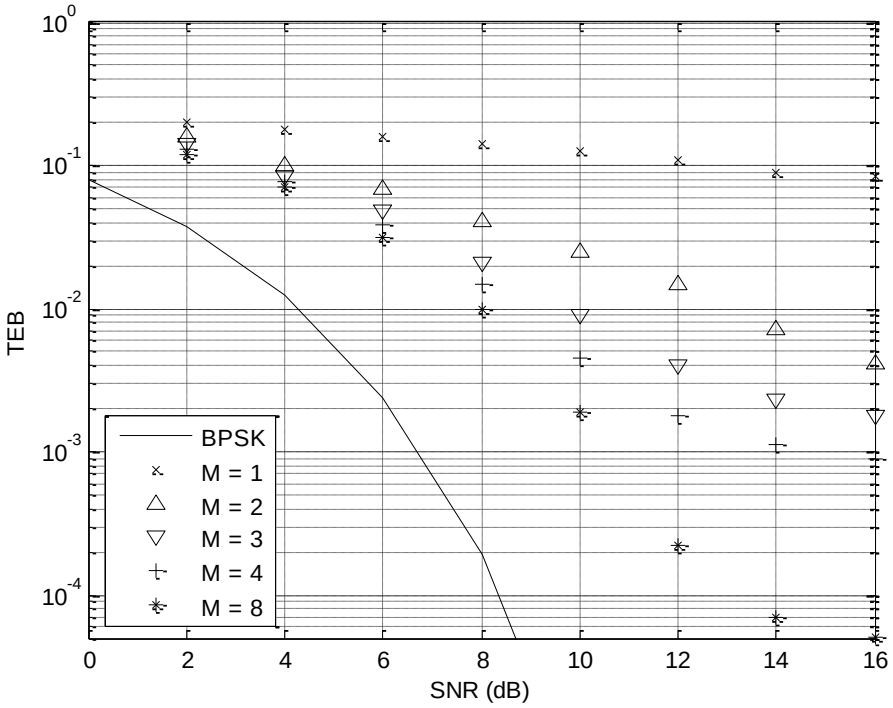


Figure 9 : Résultats de TEB pour la fonction flip-flop map avec l'algorithme de décodage ad-hoc pour D = 20 bits et différentes valeurs de M : M = 1, 2, 3, 4 symboles

On voit clairement sur ces courbe l'amélioration des performances avec l'augmentation de la valeur de *M*. Cependant, on distingue sur chaque courbe une zone où le TEB diminue de façon régulière avec le SNR et une autre zone où le TEB entre en saturation avec un effet de plancher prononcé. La raison de la présence de cet effet plancher vient du fait que certains échantillons y_{n+M-1} peuvent se retrouver en dehors de l'intervalle [0, 1] comme nous allons le montrer dans le calcul théorique qui suit. Les symboles transmis sont d'autant mieux décodés que le nombre d'échantillons à prendre en compte augmente ce qui permet de mieux exploiter la redondance entre symboles consécutifs. Cependant, la redondance présente dans les symboles successifs transmis n'est que partiellement exploité par cet algorithme.

Pour illustrer ces interprétations on peut essayer de calculer théoriquement la probabilité d'erreur pour le cas $M = 2$ (Lorsque $M > 2$, le calcul devient impossible). Pour mener à bien ce calcul, on peut distinguer trois cas :

- $y_{n+1} < 0$

  Dans ce cas on a $z_n^0 = 0$ et $z_n^1 = 1/2$ et on obtient :

$$p_{e1} = P\left[y_{n+1} < 0, \left(z_n^1 - y_n\right)^2 < \left(z_n^0 - y_n\right)^2, x_n < \frac{1}{2} \right] + P\left[y_{n+1} < 0, \left(z_n^0 - y_n\right)^2 < \left(z_n^1 - y_n\right)^2, x_n \geq \frac{1}{2} \right] \quad (38)$$

- $y_{n+1} > 1$

  Dans ce cas, $z_n^0 = 1/2$ et z_n^1=1 , le symbole décodé sera toujours égal à 1, on obtient donc :

$$p_{e2} = P\left[y_{n+1} > 1, x_n < \frac{1}{2} \right] \quad (39)$$

- $0 \leq y_{n+1} \leq 1$

  Ici aucune restriction ni normalisation ne s'applique et l'on aboutit à :

$$p_{e3} = P\left[0 \leq y_{n+1} \leq 1, \left(z_n^1 - y_n\right)^2 < \left(z_n^0 - y_n\right)^2, x_n < \frac{1}{2} \right] + P\left[0 \leq y_{n+1} \leq 1, \left(z_n^0 - y_n\right)^2 < \left(z_n^1 - y_n\right)^2, x_n \geq \frac{1}{2} \right] \quad (40)$$

En se rappelant que, dans le cas de la fonction de mapping Bernoulli shift map on a : $y_{n+1} = x_{n+1} + n_{n+1} = f(x_n) + n_{n+1},\ z_n^0 = y_{n+1}/2,\ z_n^1 = y_{n+1}/2 + 1/2,\ y_n = x_n + n_n$ et en supposant de plus que n_n et n_{n+1} sont des échantillons de processus Gaussiens indépendants et identiquement distribués, on arrive au résultat :

- $y_{n+1} < 0$

$$\begin{aligned} p_{e1}^B &= \frac{1}{4}\int_0^{1/2} \operatorname{erfc}\left(2x\sqrt{12\frac{E_b}{N_0}} \right) \operatorname{erfc}\left[\left(\frac{1}{4} - x\right)\sqrt{12\frac{E_b}{N_0}} \right] dx \\ &+ \frac{1}{2}\int_{1/2}^{1} \operatorname{erfc}\left(2x\sqrt{12\frac{E_b}{N_0}} \right) \left\{ 1 - \frac{1}{2}\operatorname{erfc}\left[\left(\frac{1}{4} - x\right)\sqrt{12\frac{E_b}{N_0}} \right] \right\} dx \end{aligned} \quad (41)$$

- $y_{n+1} > 1$

$$p_{e2}^B = \frac{1}{4}\operatorname{erfc}\left(\sqrt{12\frac{E_b}{N_0}} \right) + \frac{1}{2\sqrt{48\pi\frac{E_b}{N_0}}}\left(1 - e^{-12E_b/N_0}\right) \quad (42)$$

- $0 \le y_{n+1} \le 1$

$$p_{e3}^{B} = \frac{1}{\sqrt{48\pi \frac{E_b}{N_0}}} \int_0^1 \left(\frac{1}{2} - \frac{x}{2}\right) \left\{ \operatorname{erfc}\left[\left(\frac{1}{4} - \frac{x}{2}\right)\sqrt{12\frac{E_b}{N_0}}\right] + \operatorname{erfc}\left[\left(\frac{1}{4} + \frac{x}{2}\right)\sqrt{12\frac{E_b}{N_0}}\right] \right\} \mathrm{dx} \quad (43)$$

Il est également possible d'obtenir ces probabilités d'erreur avec la fonctions de mapping logistic map modifiée :

- $y_{n+1} < 0$

$$\begin{aligned} p_{e1}^{M} &= \frac{1}{2}\int_0^{1/2} \frac{1}{\pi\sqrt{x(1-x)}} \operatorname{erfc}\left[2\left(\frac{1}{4} - x\right)\sqrt{2\frac{E_b}{N_0}}\right] \times \left(1 - \frac{1}{2}\operatorname{erfc}\left\{2\left[-4x(1-x)\right]\sqrt{2\frac{E_b}{N_0}}\right\}\right) dx \\ &+ \int_{1/2}^1 \frac{1}{\pi\sqrt{x(1-x)}} \left\{1 - \frac{1}{2}\operatorname{erfc}\left[2\left(\frac{1}{4} - x\right)\sqrt{2\frac{E_b}{N_0}}\right]\right\} \times \left(1 - \frac{1}{2}\operatorname{erfc}\left\{2\left[-4x(1-x)\right]\sqrt{2\frac{E_b}{N_0}}\right\}\right) dx \end{aligned} \quad (44)$$

- $y_{n+1} > 1$

$$p_{e2}^{M} = \frac{1}{2}\int_0^{1/2} \frac{1}{\pi\sqrt{x(1-x)}} \operatorname{erfc}\left\{2\left[1 - 4x(1-x)\right]\sqrt{2\frac{E_b}{N_0}}\right\} dx \quad (45)$$

- $0 \le y_{n+1} \le 1$

$$\begin{aligned} p_{e3}^{M} &= \int_0^{1/2} p(x) \int_{-h(x)}^{1-h(x)} \sqrt{\frac{2}{\pi}\frac{E_b}{N_0}} e^{8\frac{E_b}{N_0}z^2} \operatorname{erfc}\left\{\left[1 + \frac{1}{2}\sqrt{h(x)+z} - \frac{1}{2}\sqrt{1-h(x)-z} - x\right]\sqrt{2\frac{E_b}{N_0}}\right\} dz dx \\ &+ \int_{1/2}^1 p(x) \int_{-1+h(x)}^{h(x)} \sqrt{\frac{2}{\pi}\frac{E_b}{N_0}} e^{8\frac{E_b}{N_0}z^2} \left(2 - \operatorname{erfc}\left\{\left[1 + \frac{1}{2}\sqrt{h(x)+z} - \frac{1}{2}\sqrt{1-h(x)-z} - x\right]\sqrt{2\frac{E_b}{N_0}}\right\}\right) dz dx \end{aligned} \quad (46)$$

Avec : $p(x) = \frac{1}{\pi\sqrt{x(1-x)}}$ et $h(x) = 4x(1-x)$.

Toutes les intégrations, à l'exception de l'équation (42), doivent être faîtes numériquement. Les résultats sur le calcul de la probabilité p_{e2}^{M} montrent bien à chaque fois la présence d'un taux d'erreur plancher à forts SNR's. La raison de la présence de cet effet plancher apparaît plus clairement maintenant. Par exemple dans le cas de l'équation (45), lorsque : $y_{n+1} > 1$, on

obtient comme valeurs inverses possibles $\frac{1}{2}$ et 1, les deux valeurs conduisant à la même prise de décision $\hat{b}_n = 1$ et ceci correspondra à chaque fois à une erreur lorsque : $x_n < \frac{1}{2}$ a été réellement transmis. Les autres termes dans le calcul des probabilités d'erreur tel que (41) et (44) ne sont importants qu'à faibles SNR's et ne présentent aucun effet plancher à forts SNR's. La conclusion de cette analyse est simple : le fait d'augmenter *M* permet de mieux compenser les phénomènes d'erreur de décision produits par des symboles reçus qui se trouvent en dehors de l'intervalle [0, 1]. Ceci montre en fait toute l'importance d'inclure le maximum d'échantillons possible en sortie du canal pour dérouler les algorithmes de décodage.

Les performances de l'algorithme de décodage ad-hoc récursif sont illustrées sur les figures 10 à 12.

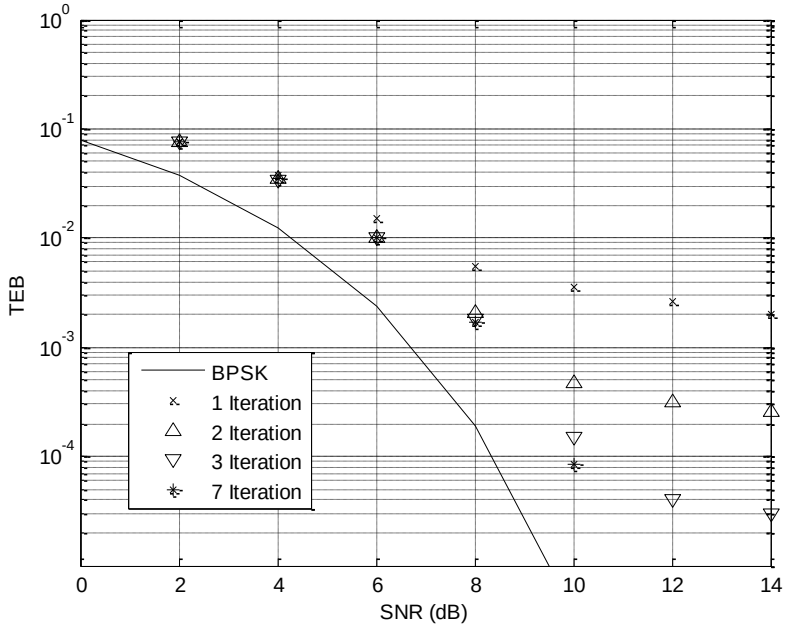


Figure 10 : Résultats de TEB pour la fonction de mapping Bernoulli shift map avec l'algorithme de décodage ad-hoc récursif pour D = 20 bits, M = 4 symboles et pour différents nombres d'itérations de l'algorithme (1, 2, 3 et 7).

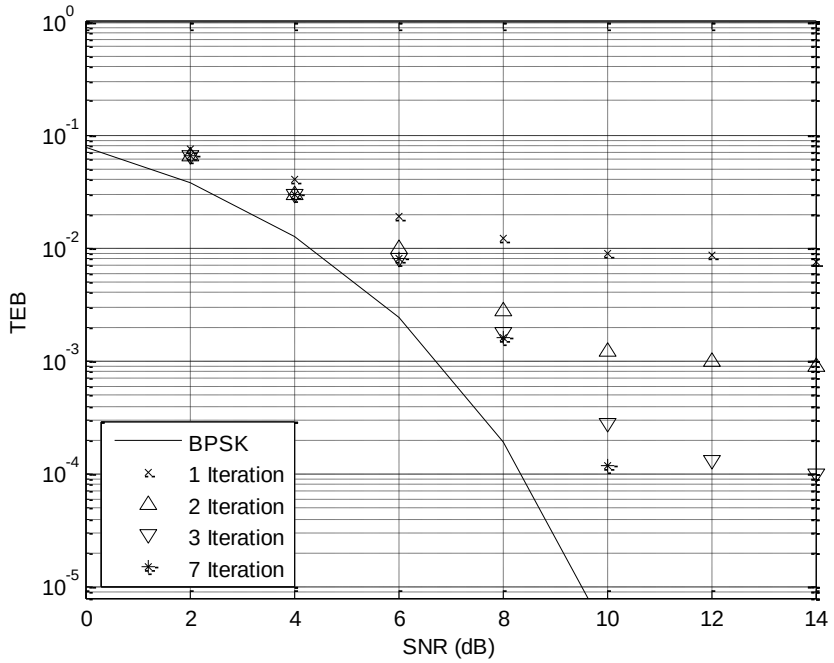


Figure 11 : Résultats de TEB pour la fonction de mapping logistic map modifiée avec l'algorithme de décodage ad-hoc récursif pour D = 20 bits, M = 4 symboles et pour différents nombres d'itérations de l'algorithme (1, 2, 3 et 7).

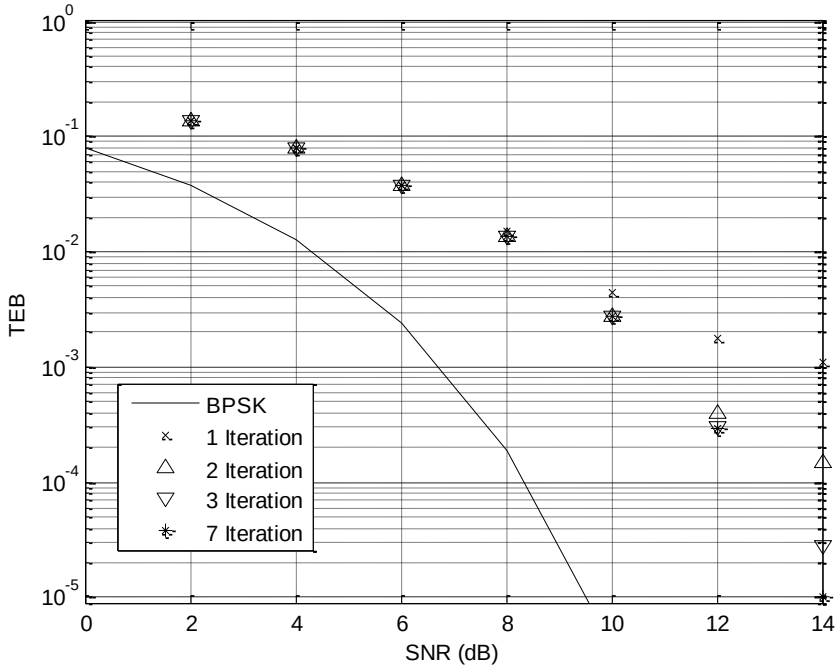


Figure 12 : Résultats de TEB pour la fonction de mapping flip-flop map avec l'algorithme de décodage ad-hoc récursif pour D = 20 bits, M = 4 symboles et pour différents nombres d'itérations de l'algorithme (1, 2, 3 et 7).

On voit clairement sur ces figures l'intérêt d'utiliser la redondance présente dans toute la séquence (non seulement la redondance est exploitée entre y_n et y_{n+M} mais aussi entre y_n et y_{n-M}). On peut de plus constater que pour des TEB's de l'ordre de 10^{-4},10^{-5}, l'écart avec le système BPSK non codé se réduit fortement par rapport à l'utilisation du simple algorithme ad-hoc puisqu'il devient de l'ordre de 1.5-2 dB, ce qui représente un avantage de plusieurs dB (4-6) par rapport à la version non-récursive de cet algorithme. Cependant la complexité de décodage passe de $\vartheta(2^M)$ à $\vartheta(q.2^M)$ où q est le nombre d'itérations.

Les performances de l'algorithme de Viterbi sont illustrées en figure 13 dans le cas P = 32 et L = 10.

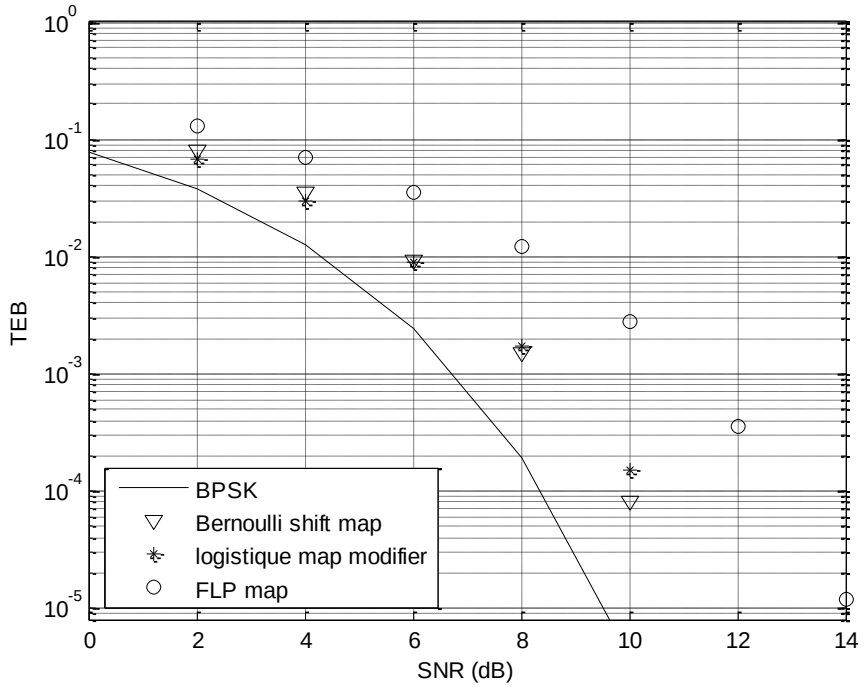


Figure 13 : Résultats de TEB pour les fonctions de mapping Bernoulli shift map, logistic map modifiée et flip-flop map avec l'algorithme de décodage de Viterbi pour P = 32 états et L = 10 symboles.

On constate sur cette courbe que les résultats obtenus avec Viterbi sont très proches de ceux réalisés avec l'algorithme de décodage ad-hoc récursif au bout de la septième itération. L'écart entre ces deux algorithmes est inférieur à 0.5 dB pour des TEB's de l'ordre de 10^{-4},10^{-5}. La constatation est la même avec l'algorithme BCJR en figure 14 même si dans ce cas on constate ne amélioration non négligeable des performances de Viterbi et de l'algorithme ad-hoc récursif (amélioration de l'ordre de 0.8 dB). Ainsi, l'algorithme BCJR apparaît comme le plus performant. Sa complexité étant de l'ordre de $\vartheta(L.P)$ avec P qui est

une puissance de 2 reste tout à fait comparable à celle de l'algorithme de Viterbi et de l'algorithme ad-hoc récursif.

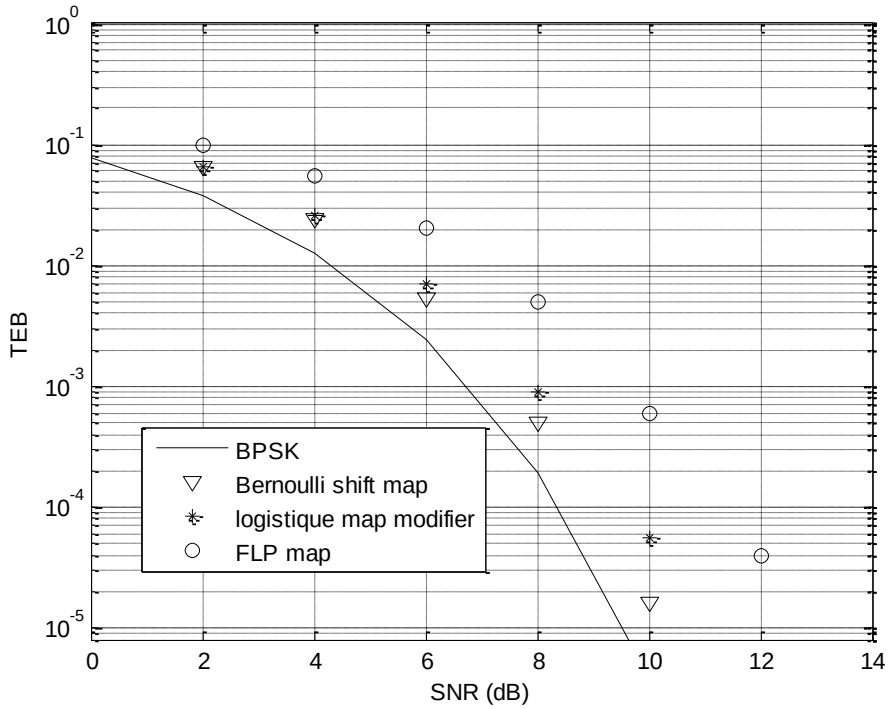


Figure 14 : Résultats de TEB pour les fonctions de mapping Bernoulli shift map, logistic map modifiée et flip-flop map avec l'algorithme de décodage BCJR pour P = 32 états et L = 10 symboles.

VI. CONCLUSION

Dans ce chapitre nous avons examiné les performances de différentes fonctions de mapping chaotiques monodimensionnelles basées sur des transformations de la célèbre fonction Bernoulli shift map. Après avoir montré comment il était possible d'encoder une séquence binaire à l'aide ces fonctions non linéaires, nous avons développé et testé différents algorithmes de décodage. Parmi eux nous avons proposé un algorithme baptisé algorithme de décodage ad-hoc récursif qui est quasiment aussi performant que les algorithmes classiques à base de Maximum de Vraisemblance (ML) ou de Maximum de Vraisemblance à Postériori (MAP) et qui présente une complexité d'implémentation réduite.

CHAPITRE III MODULATION CODÉE BASÉE SUR DES FONCTIONS CHAOTIQUES MULTIDIMENSIONNELLES

I. INTRODUCTION :

Dans ce chapitre nous étudions une nouvelle classe de modulations codées non linéaires basée sur des systèmes dynamiques chaotiques à temps discrets. Le gros intérêt de cette classe de modulation est d'associer le codage de canal avec le « mapping » pour obtenir une constellation en sortie qui ressemble, lorsque le pas de quantification est suffisamment petit, à un signal réellement imprévisible donc chaotique au sens de la définition de Frey.

L'idée de développer une telle modulation codée est basée sur le contrôle approprié des systèmes dynamiques. Ceci peut être réalisé en utilisant la dynamique symbolique. L'espace d'état du système chaotique est divisée en un nombre fini de régions qui sont marquées par des symboles. De cette manière, une trajectoire dans l'espace état correspond à une séquence symbolique infinie. Puisque la trajectoire est déterminée par une condition initiale, à chaque état initial correspond une séquence symbolique infinie. Au lieu d'essayer de fixer une condition initiale avec une précision infinie, on contrôle la trajectoire avec des entrées de commande de faible amplitude afin d'obtenir une séquence symbolique souhaitée.

Le nouveau modèle des modulations codées en treillis est donc obtenu par la combinaison de ces deux approches :

L'utilisation des systèmes de contrôle pour les systèmes en temps continu [44].

L'utilisation de la dynamique symbolique pour les communications numériques [45].

Dans ce chapitre, nous nous basons sur les travaux antérieurs de S. Kozic [46] concernant les modulations codées multidimensionnel basées sur des systèmes chaotiques.

Le but est d'optimiser les performances du schéma de modulation en calculant la distribution du spectre des distances du codeur. Nous allons montrer dans ce chapitre que la distribution du spectre des distances du codeur peut être approximée soit par une loi de Rayleigh, soit par un mélange de lois Gaussiennes ou de lois de Rayleigh. La caractérisation précise du spectre des distances du codeur nous permet bien sûr d'optimiser les paramètres de ce dernier.

Puis nous étudions les performances du schéma de modulation chaotique proposé lorsqu'il est concaténé avec un code temps-espace en bloc de type Alamouti [47]. Ces études sont faites dans le cas de canaux non-sélectifs en fréquence mais en tenant compte éventuellement d'une forte sélectivité temporelle (effet Doppler). Utilisant la modélisation de la densité de probabilité des distances du codeur, nous sommes capables d'obtenir des expressions

théoriques pour la probabilité d'erreur par symbole ou par bit. Ces résultats complètent ceux de Jun He and Pooi Yuen Kam [48]qui ont étudié le comportement du schéma d'Alamouti dans les canaux très sélectifs en temps. Dans tous les cas, nous avons quantifié le gain de diversité du schéma de transmission proposé. C'est ainsi que nous montrons que la technique MRC classique pour le décodage d'Alamouti conduit à une perte de diversité complète sur des canaux très sélectifs en temps. Pour restaurer une certaine diversité nous avons proposé une égalisation ZF (Zero Forcing) et nous avons quantifié son efficacité.

II. MODULATION CODÉE CHAOTIQUE À BASE DE FONCTIONS MULTIDIMENSIONNELLES

Le système étudié et proposé auparavant par S. Kozic [46]est constitué d'un codeur convolutif linéaire et d'un mapping non-linéaire multidimensionnel en sortie du codeur convolutif. Un tel schéma qui associe mapping et codage de canal est semblable dans son principe aux modulations codées en treillis.

La fonction multidimensionnelle utilisée dans le mapping en sortie du codeur est définie par :

$$\boldsymbol{x}_{k+1} = f_{\boldsymbol{A}}\left(\boldsymbol{x}_k\right) = 2\boldsymbol{A}.\boldsymbol{x}_k \bmod 1 \quad (1)$$

Où $\boldsymbol{A} = \{a_{ij}\}$ est un matrice de taille $n \times n$. $\boldsymbol{x}_k$ est un vecteur de symboles analogique qui peut s'interpréter comme la sortie d'une modulation codée multidimensionnelle.

Les séquences symboliques de $f_{\boldsymbol{A}}$ sont définies par un partitionnement n-dimensionnel de l'espace, i.e. : l'espace est subdivisé en 2^n segments $\prod = \{P_1, P_2, \ldots, P_2{}^n\}$ avec la règle :

$$s_k = i \quad si \quad \boldsymbol{x}_k \in P_i \quad (2)$$

L'état x_k de la fonction $f_{\boldsymbol{A}}$ peut être exprimé en fonction des vecteurs d'entrée binaires :

$$\boldsymbol{b}_k = \left[b_{(k-1)n+1}, b_{(k-1)n+2}, \ldots, b_{kn} \right]^T$$

Ces vecteurs sont obtenus en partitionnant le vecteur des bits à transmettre $\boldsymbol{b}$ en blocs $\boldsymbol{b}_k$ de n bits. La sortie $\boldsymbol{x}_k$ du codeur se calcule alors de la manière suivante :

$$\boldsymbol{x}_k = \sum_{i=0}^{Q} 2^{-(i+1)} \boldsymbol{A}^{Q-i} \boldsymbol{b}_{k+i-Q} \ mod\, 1 + \frac{q}{2}\boldsymbol{p} \quad (3)$$

Où $\boldsymbol{p} = (2\boldsymbol{A}-\boldsymbol{I})^{-1}.\boldsymbol{e}$ avec $\boldsymbol{e} = [1,1,...,1]^T$ et $q = 2^{-Q-1}$ La somme des termes dans ce développement binaire multidimensionnel est finie et donc les vecteurs binaires $\boldsymbol{b}_{k+i-Q}$ qui entrent dans le modulateur permettent de contrôler la trajectoire $\boldsymbol{x}_k$ en sortie. A chaque itération les vecteurs d'entrée binaires sont décalés d'une ou de plusieurs position vers la gauche (shift register) de façon totalement similaire à un codeur convolutif classique.

L'expression du signal multidimensionnel de sortie $\boldsymbol{x}_k$ appelé aussi signal modulé peut être réécrite (3) sous une autre forme qui montre l'évolution des états du codeur entre les états k et k+1 en fonction du vecteur d'entrée $\boldsymbol{b}_{k+1}$:

$$\boldsymbol{x}_{k+1} = f_{\boldsymbol{A}}(\boldsymbol{x}_k) + q(\boldsymbol{b}_{k+1} - 1/2.\boldsymbol{e}) \ mod\, 1, \quad \boldsymbol{e} = [1,1,...,1]^T \quad q = 2^{-Q-1} \quad (4)$$

Dans le cas asymptotique, lorsque le pas de quantification tend vers zéro i.e. Q tend vers l'infini, il est clair que q tend vers zéro et on retrouve l'équation (1) du mapping multidimensionnel. La partie $q(\boldsymbol{b}_{k+1} - 1/2.\boldsymbol{e})$ se comporte alors comme une perturbation stochastique qui permet de contrôler la trajectoire de sortie des vecteurs $\boldsymbol{x}_k$. Ainsi, ce système chaotique peut-il être représenté sous la forme suivante (Fig. 1) :

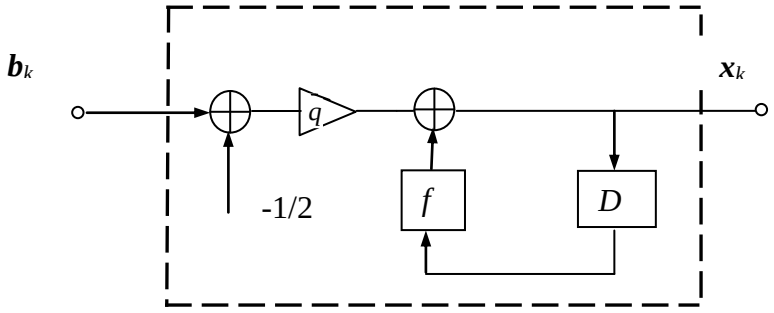


Figure 1 : Système chaotique multidimensionnel contrôlé par perturbation stochastique

Le vecteur de sortie $\boldsymbol{x}_k$ peut prendre $M = 2^{n(Q+1)}$ différentes valeurs en fonction des valeurs de $b_k, \ldots, b_{k-Qn}$. Par conséquent, la modulation codée proposée est une modulation numérique comportant M niveaux discrets.

II.1. Structure du codeur

Le système décrit par l'expression (3) peut être représenté comme un codeur convolutif linéaire de rendement $\eta = 1/(n.(Q+1))$, où, à chaque coup d'horloge, un bit b_k entrant donne naissance à $(Q+1)$ vecteur **v** chacun de n bits.

Ce codeur est décrit par:

$$\boldsymbol{h}_i(D) = \frac{\mathbf{v}_i(D)}{b(D)} = D^{n(Q-i)}\left[1D...D^{n-1}\right]^T \quad (5)$$

où D est un retard de telle sorte que chaque séquence d'entrée $b_1,b_2,\ldots,b_k,\ldots$ est représentée par la série infinie :

$$b(D) = b_1 + b_2 D + b_3 D^2 + ... \quad (6)$$

Le signal de constellation (*mapping*) est décrit par la relation:

$$\boldsymbol{x}(D) = \sum_{i=0}^{Q} 2^{-(i+1)} \boldsymbol{A}^{Q-i} \mathbf{v}_i(D) \bmod 1 \quad (7)$$

Donc, l'expression (3) peut être interprétée comme la sortie d'un codeur d'une modulation codée en treillis (TCM), comme illustré sur la figure 2 :

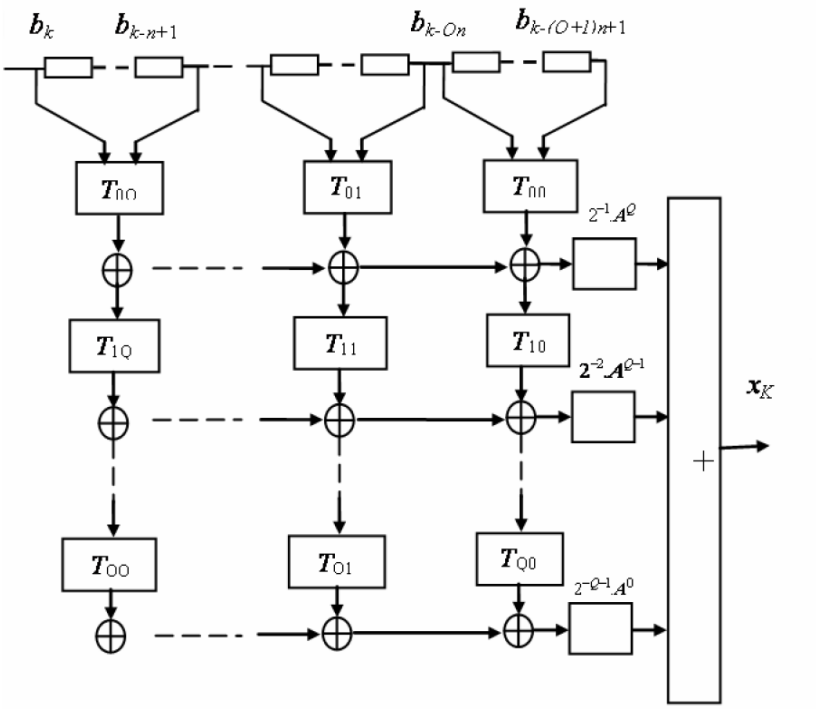


Figure 2: schéma de la modulation codée chaotique multidimensionnelle

Le code convolutif peut être décrit sous une forme matricielle $\boldsymbol{T}_{ij} = \left\{ t_{lm}^{ij} \right\}, \left(l, m = 0, \overline{n-1} \right)$

$$\boldsymbol{h}_{ij}\left(D\right) = D^{n(Q-j)}.\begin{bmatrix} t_{0,0}^{(i,j)} + & t_{0,1}^{(i,j)} D + & \cdots & t_{0,n-1}^{(i,j)} D^{n-1} \\ t_{1,0}^{(i,j)} + & t_{1,1}^{(i,j)} D + & \cdots & t_{1,n-1}^{(i,j)} D^{n-1} \\ \vdots & \vdots & \cdots & \\ t_{n-1,0}^{(i,j)} + & t_{n-1,1}^{(i,j)} D + & \cdots & t_{n-1,n-1}^{(i,j)} D^{n-1} \end{bmatrix}$$

$$\boldsymbol{h}_{i}\left(D\right) = \frac{\mathbf{v}_{i}\left(D\right)}{b\left(D\right)} = \sum_{j=0}^{Q} \boldsymbol{h}_{ij}\left(D\right) \quad (8)$$

Les blocks des matrices $\boldsymbol{T}$ utilisées pour l'optimisation du codeur sont : $\boldsymbol{T}_{shift}$ et $\boldsymbol{T}_{tent}$.

$$\boldsymbol{T} = \begin{cases} \boldsymbol{T}_{ij} = \boldsymbol{T}_{shift} & i = j \\ \boldsymbol{T}_{ij} = \boldsymbol{T}_{tent} & i \neq j \end{cases} \quad (9)$$

Où $\boldsymbol{T}_{shift} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \cdots \end{bmatrix}$ et $\boldsymbol{T}_{tent} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & \cdots \end{bmatrix}$.

Il existe une autre forme plus générale pour le codeur chaotique donnée par l'équation :

$$\boldsymbol{x}_{k} = \sum_{i=0}^{Q_a} 2^{-(i+1)} . \boldsymbol{A}^{Q_a - i} . \mathbf{v}_{i}(D) + \sum_{i=Q_a+1}^{Q} 2^{-(i+1)} . \mathbf{v}_{i}(D) \bmod(1) \quad (10)$$

Cette équation est plus générale que (7) car elle fait intervenir un paramètre supplémentaire Q_a . A partir de l'indice Q_a +1, la matrice $\boldsymbol{A}$ est remplacée par la matrice identité $\boldsymbol{I}$. Nous avons constaté que cette écriture permettait (pour des faibles valeurs de Q et Q_a) d'obtenir de meilleures distances libres que la forme (7).

II.2. Distance minimale et spectre de distances

Les performances des codes en treillis dépendent bien sûr de l'algorithme de décodage mais aussi et surtout de la distribution des distances du codeur. Le paramètre le plus important en codage de canal est la distance minimale d_{free} définie comme la distance minimale entre deux séquences qui partent d'un même état du codeur et qui reconvergent vers un même état (non nécessairement le même que l'état initial) au bout d'un certain nombre de périodes

d'horloges. Pour les codeurs linéaires ce calcul est relativement aisé car on peut prendre comme séquence de référence la séquence de bits tous égaux à zéro. Malheureusement, cette simplification n'est plus valable pour un codeur non-linéaire tel que le schéma considéré dans ce chapitre. Pour calculer sa valeur il faut déterminer les trajectoires dans le treillis qui partent depuis le même état S_i=S_i^* et qui évoluent ensuite à travers des chemins disjoints (après L coups d'horloge) vers le même état S_k=S_k^*, non nécessairement égale a S_i. On parlera alors de boucles de longueur L.

La distance correspondant de S_i et S_k est donnée par :

$$d^2_{L,S_i,S_k} = \sum_{m=1}^{L-1} \left\| \boldsymbol{x}_m - \boldsymbol{x}^*_m \right\|^2 \qquad (11)$$

Ce calcul doit être fait pour tous les états initiaux du codeur et pour toutes les valeurs possibles de L. Ce calcul est d'une complexité rapidement prohibitive puisqu'il faut tester $2^{n(Q+1)}.2^{nL}$ trajectoires de longueur L. Pour obtenir une complexité raisonnable nous avons imposé à L de varier dan l'intervalle $[Qn+1,\ n.(Q+m)]$, ce qui implique que la longueur des boucles varie de $Qn+1$ (la longueur de contrainte plus un) à $n.(Q+m)$ avec m = 3 au maximum (les simulations effectuées ont montré que cette valeur était suffisante).

Plusieurs paramètres entrent dans l'optimisation du codeur : les matrices $\boldsymbol{T}_{ij}$, les matrices $\boldsymbol{A}$, le paramètre Q, etc…Pour simplifier notre recherche nous avons imposé des formes particulières aux matrices $\boldsymbol{A}$ et $\boldsymbol{T}$.

Pour la matrice $\boldsymbol{A}$ nous avons pris la forme suivante [46]:

$$\boldsymbol{A} = \begin{pmatrix} 1 & -1 \\ a_{21} & 1 \end{pmatrix} pour \quad n = 2$$

$$\boldsymbol{A} = \begin{pmatrix} 1 & 1 & 1 \\ a_{21} & 1 & 1 \\ a_{31} & a_{32} & 1 \end{pmatrix} pour \quad n = 3 \quad (12)$$

Le choix des paramètres a_{ij} se fait en optimisant la distance libre du codeur.

Pour les matrices $\boldsymbol{T}$ nous choisissons systématiquement : $\boldsymbol{T}_{i,j}$=$\boldsymbol{T}_{shift}$ pour i=j et $\boldsymbol{T}_{i,j}$=$\boldsymbol{T}_{tent}$ pour $i \neq j$

Par exemple la figure 3 illustre le spectre obtenu pour le cas n = 2 avec a_{21} = 8 et $Q = Q_a$ = 3. Attention, il s'agit ici d'un histogramme et non d'une densité de probabilité (p.d.f).

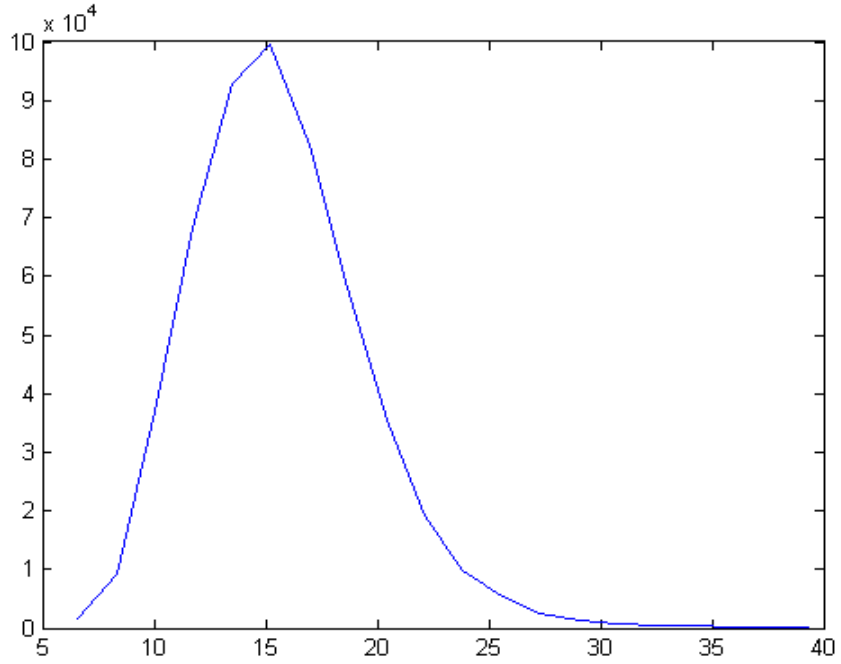


Figure 3: spectre de Distance pour n = 2 avec a_{21}= 8

En fait, la distribution normalisée ou p.d.f de ce spectre est très bien approximée par la fonction (distribution de Rayleigh):

$$f_C(x) = \frac{(x-m_j)}{\sigma_j^2}.e^{-(x-m_j)^2/2.\sigma_j^2}, x \geq m_j \qquad (13)$$
$$f_C(x) = 0, x < m_j$$

Par exemple avec le spectre de distance obtenue à la figure 3, en utilisant le technique MMSE classique pour obtenir les paramètres, on aboutit à : $m_j \cong \sigma_j^2 \cong 6.7$ (ce qui correspond à une distance minimale de codeur égale $d_{free} \cong 6.7$). La comparaison entre la distribution de Rayleigh et le spectre de distance normalisé obtenu est illustrée dans la figure.4 (l'erreur quadratique moyenne obtenue est inférieure à 10^{-3}):

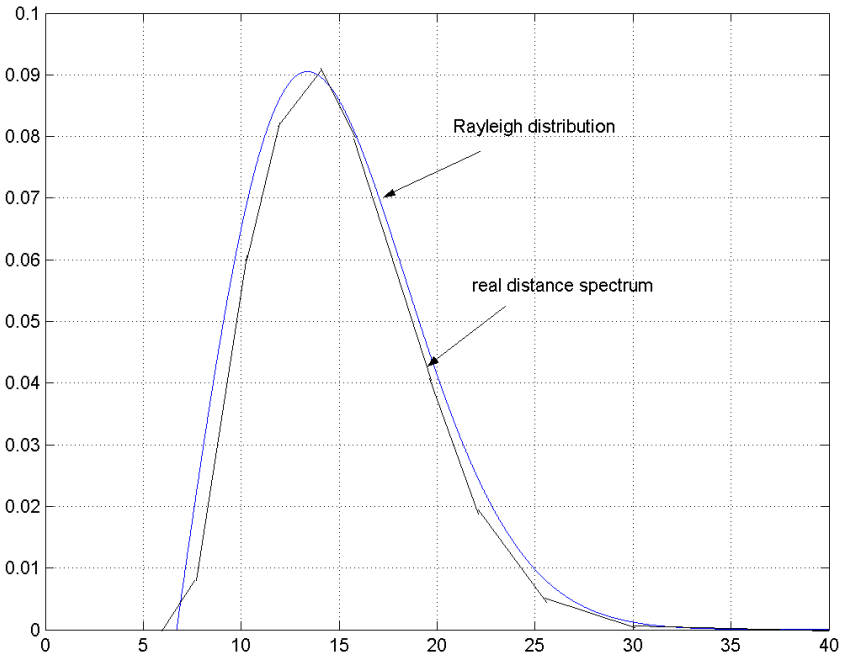


Figure 4: spectre de Distance et distribution de Rayleigh

Pour certains cas, il existe des formes plus compliquées pour la distribution des distances du codeur. Par exemple, le spectre de distance illustré dans la figure. 5 est obtenue pour : $\boldsymbol{T}_{i,j}=\boldsymbol{T}_{shift}$ pour $i=j$ et $\boldsymbol{T}_{i,j}=\boldsymbol{T}_{tent}$ pour $i \neq j$, avec $a_{21} = 5$, $n = 2$ et $Q = Q_a = 3$.

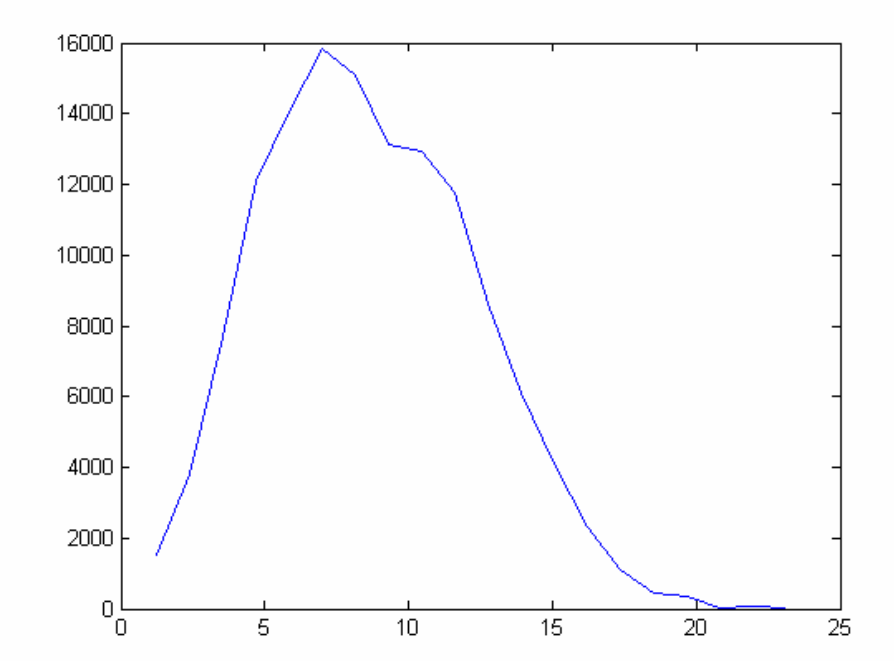


Figure 5: spectre de Distance pour n =2 avec a_{21}=5.

Dans ces cas, la fonction de densité de probabilité peut être approximée comme nous allons le voir dans le prochain paragraphe à l'aide d'une mixture de lois Gaussiennes ou de Rayleigh [51]

II.2.1. Approximation du spectre de distances à l'aide d'un mélange de lois Gaussiennes

La distribution normalisée du spectre prend la forme:

$$f_C(x) = \sum_{n=1}^{J} \pi_n . \frac{1}{\sqrt{2.\pi}.\sigma_n} . e^{-(x-m_n)^2/2.\sigma_n^2} = \sum_{n=1}^{J} \pi_n . \mathcal{N}(m_n, \sigma_n^2) \quad (14)$$

Il est clair que trouver les paramètres J, π_n, m_n et σ_n, est un problème d'optimisation compliqué. L'algorithme EM (Expectation Maximization) permet de réduire considérablement cette complexité [49][50][51] .

Pour un nombre fixé J de lois gaussiennes et pour un ensemble d'observations données $\Xi \equiv \{\xi_i, i = 1,...,n\}$, l'algorithme doit déterminer l'ensemble $\boldsymbol{\theta} \equiv \{\pi_j, m_j, \sigma_j, j = 1,...,J\}$ en cherchant la solution du problème à Maximum de Vraisemblance suivant :

$$\hat{\boldsymbol{\theta}} = \arg \max_{\boldsymbol{\theta}: \sum_{j=1}^{J} \pi_j = 1} \log p_{\boldsymbol{\theta}}(\Xi)$$

$$= \arg \max_{\boldsymbol{\theta}: \sum_{j=1}^{J} \pi_j = 1} \sum_{i=1}^{n} \log \sum_{j=1}^{J} \pi_j . \phi(\xi_i; m_j, \sigma_j^2) \quad (15)$$

Notant $\phi(x; \mu, \sigma^2)$ la densité de probabilité d'une loi Gaussienne de paramètres μ et σ^2 : $\mathcal{N}(\mu, \sigma^2)$. Pour résoudre (14) l'algorithme E.M procède en deux étapes à partir d'un jeu initial de coefficients $\boldsymbol{\theta}^{(0)}$: une étape de moyennage et une étape de maximisation. Les deux étapes se résument de la façon suivante [51] :

- Etape de moyennage, on calcule:

$$Q(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}^{(i)}) = E_{\boldsymbol{\theta}^{(i)}} \{\log p_{\boldsymbol{\theta}}(\boldsymbol{X}) \,|\, \Xi\} \quad (16)$$

- Etape de maximisation, on calcule :

$$\boldsymbol{\theta}^{(i+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}^{(i)}) \qquad (17)$$

Pour procéder à ces étapes de calcul, il faut définir l'ensemble des données cachées $\boldsymbol{Z} = \{\boldsymbol{z}_i, i = 1,...,n\}$ où $\boldsymbol{z}_i$ est un vecteur de dimension *J* qui sert de fonction indicatrice.

$$z_{i,j} = \begin{cases} 1, si\ \xi_i \cong \mathcal{N}(m_j, \sigma_j^2) \\ 0 \ \text{sinon} \end{cases} \qquad (18)$$

L'ensemble des données complètes est donné par : $\boldsymbol{X} \cong (\Xi, \boldsymbol{Z})$, et on obtient :

$$p_{\boldsymbol{\theta}}(\Xi, \boldsymbol{Z}) = \prod_{i=1}^{n}\prod_{j=1}^{J}\left[\pi_j.\phi(\xi_i; m_j, \sigma_j^2)\right]^{z_{i,j}} \qquad (19)$$

La fonction de vraisemblance logarithmique des données complètes devient alors:

$$\log p_{\boldsymbol{\theta}}(\Xi, \boldsymbol{Z}) = \sum_{i=1}^{n}\sum_{j=1}^{J} z_{i,j}.\log \pi_j + \sum_{i=1}^{n}\sum_{j=1}^{J} z_{i,j}.[-\frac{1}{2}.\log(\sigma_j^2) - \frac{(\xi_i - m_j)^2}{2.\sigma_j^2}] + C \qquad (20)$$

où *C* est une constante quelconque.

On peut alors préciser complètement le déroulement de l'algorithme E.M, on a :

$$Q(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}') = E'_{\boldsymbol{\theta}}\{\log p_{\boldsymbol{\theta}}(\Xi, \boldsymbol{Z}) \,|\, \Xi\}$$

$$Q(\boldsymbol{\theta} \,|\, \boldsymbol{\theta}') = \sum_{i=1}^{n}\sum_{j=1}^{J} \hat{z}_{i,j}.[\log \pi_j - \log(\sigma_j) \ - \frac{(\xi_i - m_j)^2}{2.\sigma_j^2}] + C \qquad (21)$$

Avec :

$$\begin{aligned} \hat{z}_{i,j} &= E'_{\boldsymbol{\theta}}\left\{ z_{i,j} \,\middle|\, \Xi, \boldsymbol{\theta}' \right\} = P'_{\boldsymbol{\theta}}\left\{ z_{i,j} = 1 \,|\, \xi_i \right\} \\ &= \frac{p'_{\boldsymbol{\theta}}(\xi_i \,|\, z_{i,j} = 1).P'_{\boldsymbol{\theta}}(z_{i,j} = 1)}{\sum_{l=1}^{J} p'_{\boldsymbol{\theta}}(\xi_i \,|\, z_{i,l} = 1).P'_{\boldsymbol{\theta}}(z_{i,l} = 1)} \\ &= \frac{\phi(\xi_i; m'_j, \sigma'^2_j).\pi'_j}{\sum_{l=1}^{J} \phi(\xi_i; m'_l, \sigma'^2_l).\pi'_l} \end{aligned} \qquad (22)$$

Pour l'étape de maximisation, on calcule les dérivées partielles.

- $\{\pi_j\}$ est obtenu par:

$$\frac{\partial Q(\boldsymbol{\theta},\boldsymbol{\theta}')}{\partial \pi_j} = 0 \Rightarrow \pi_j = \frac{1}{n}.\sum_{i=1}^{n} \hat{z}_{i,j}, \quad j = 1,...,J \qquad (23)$$

- m_j est obtenue par:

$$\frac{\partial Q(\boldsymbol{\theta},\boldsymbol{\theta}')}{\partial m_j} = 0 \Rightarrow \sum_{i=1}^{n} \hat{z}_{i,j}.[\frac{2.(\xi_i - m_j)}{2.\sigma_j^2}] = 0, \quad j = 1,...,J$$

$$\Rightarrow \frac{m_j}{\sigma_j^2}\sum_{i=1}^{n} \hat{z}_{i,j} = \sum_{i=1}^{n} \frac{\xi_i.\hat{z}_{i,j}}{\sigma_j^2}, \quad j = 1,...,J$$

$$\Rightarrow m_j = \frac{\sum_{i=1}^{n} \xi_i.\hat{z}_{i,j}}{\sum_{i=1}^{n} \hat{z}_{i,j}}, \quad j = 1,...,J \qquad (24)$$

Enfin, la variance σ_j est obtenue à l'aide de la résolution de l'équation suivante :

$$\frac{\partial Q(\boldsymbol{\theta},\boldsymbol{\theta}')}{\partial \sigma_j} = 0 \Rightarrow \sum_{i=1}^{n} \hat{z}_{i,j}.[-\frac{1}{\sigma_j} + \frac{(\xi_i - m_j)^2}{\sigma_j^3}] = 0, \quad j = 1,...,J$$

$$\Rightarrow \sigma_j^2.\sum_{i=1}^{n} \hat{z}_{i,j} = \sum_{i=1}^{n} \hat{z}_{i,j}.(\xi_i - m_j)^2, \quad j = 1,...,J$$

$$\sigma_j^2 = \frac{\sum_{i=1}^{n} \hat{z}_{i,j}.(\xi_i - m_j)^2}{\sum_{i=1}^{n} \hat{z}_{i,j}}, \quad j = 1,...,J \qquad (25)$$

Finalement, les étapes de calcul de l'algorithme E.M peuvent être résumées de la façon suivante : à partir d'un jeu initial de paramètres $\boldsymbol{\theta}^{(0)}$ et des données observées $\{\xi_i\}$:

- Faire $\boldsymbol{\theta}' = \boldsymbol{\theta}^{(i-1)}$ et calculer $\{\hat{z}_{i,j}, i = 1,...,n; j = 1,...J\}$ d'après la formule (22).
- Calculer $\{\pi_j, j = 1,...,J\}$ d'après (23) et calculer $\{m_j, j = 1,...,J\}$, $\{\sigma_j, j = 1,...,J\}$ d'après les équations (24-25). A partir de là, faire $\boldsymbol{\theta}^{(i)} = \boldsymbol{\theta}$.

Pour la distribution illustrée en figure 5, on peut trouver une mixture de deux lois Gaussiennes avec les paramètres π_1=0.6, m_1=4.2 et σ_1=2.24. π_2=0.4, m_2=9.5 et σ_2=2.44. Les résultats sont illustrés sur la figure 6 :

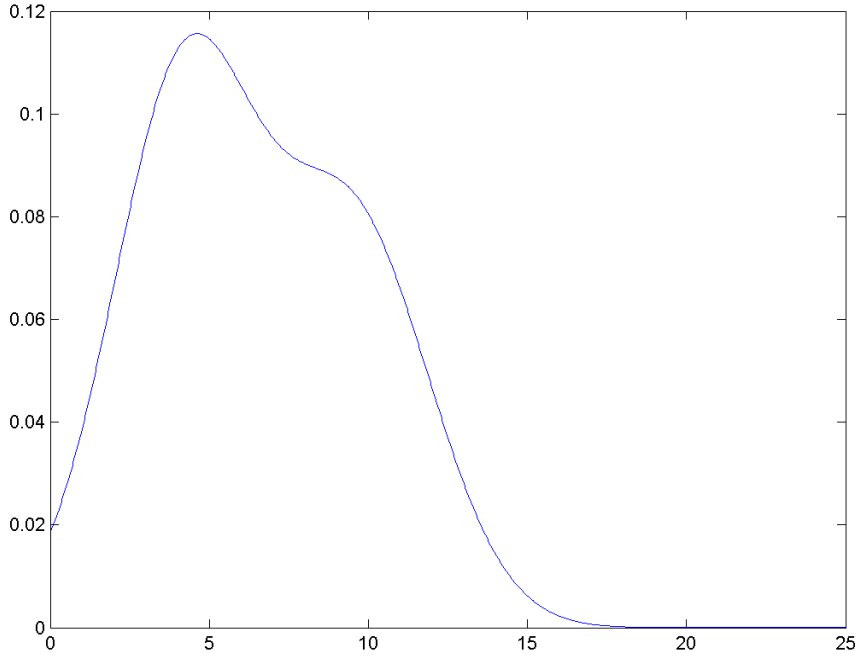


Figure. 6: distribution de la densité de la probabilité normalisée en figure 5 obtenue par un mélange de lois gaussiennes.

II.2.2. Mélange de lois de Rayleigh

La distribution normalisée du spectre prend la forme:

$$f_C(x) = \sum_{n=1}^{J} \pi_n . \frac{(x - m_n)}{\sigma_n^2} . e^{-(x-m_n)^2/2.\sigma_n^2} = \sum_{n=1}^{J} \pi_n . \mathcal{R}(m_n, \sigma_n^2) \qquad (26)$$

Ou $\mathcal{R}(m_n, \sigma_n^2)$ représente une loi de Rayleigh de paramètres : m_n et σ_n^2. Utilisant la même approche et les mêmes notations que pour un mélange de lois gaussiennes, on peut obtenir :

$$\log p_{\boldsymbol{\theta}}(\Xi, \boldsymbol{Z}) = \sum_{i=1}^{n}\sum_{j=1}^{J} z_{i,j} . \log \pi_j + \sum_{i=1}^{n}\sum_{j=1}^{J} z_{i,j} . [\log(\xi_i - m_j) - \log(\sigma_j^2) - \frac{(\xi_i - m_j)^2}{2.\sigma_j^2}] + C \qquad (27)$$

Notant $\psi(x; \mu, \sigma^2)$ la densité de probabilité d'une variable aléatoire $\mathcal{R}(\mu, \sigma_n^2)$, on peut obtenir :

$$Q(\boldsymbol{\theta}\left|\boldsymbol{\theta}'\right.) = E_{\boldsymbol{\theta}}^{'}\left\{\log p_{\boldsymbol{\theta}}(\Xi, \boldsymbol{Z})\left|\Xi\right.\right\}$$

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}') = \sum_{i=1}^{n}\sum_{j=1}^{J}\hat{z}_{i,j}.[\log\pi_j + \log(\xi_i - m_j) - 2.\log(\sigma_j) - \frac{(\xi_i - m_j)^2}{2.\sigma_j^2}] + C \qquad (28)$$

Avec

$$\hat{z}_{i,j} = E_{\boldsymbol{\theta}}^{'}\left\{z_{i,j}\left|\ \Xi, \boldsymbol{\theta}'\right.\right\} = P_{\boldsymbol{\theta}}^{'}\left\{z_{i,j} = 1\middle|\xi_i\right\}$$
$$= \frac{\psi(\xi_i; m_j^{'}, \sigma_j^{'2}).\pi_j^{'}}{\sum_{l=1}^{J}\psi(\xi_i; m_l^{'}, \sigma_l^{'2}).\pi_l^{'}} \qquad (29)$$

L'étape de maximisation se résume comme suit :

Pour obtenir $\{\pi_j\}$:

$$\frac{\partial Q(\boldsymbol{\theta}, \boldsymbol{\theta}')}{\partial \pi_j} = 0 \ \Rightarrow \pi_j = \frac{1}{n}.\sum_{i=1}^{n}\hat{z}_{i,j}, \quad j = 1,..., J \qquad (30)$$

Pour obtenir $\{m_j\}$:

$$\frac{\partial Q(\boldsymbol{\theta}, \boldsymbol{\theta}')}{\partial m_j} = 0 \Rightarrow \sum_{i=1}^{n}\hat{z}_{i,j}.[-\frac{1}{\xi_i - m_j} + \frac{(\xi_i - m_j)}{\sigma_j^2}] = 0, \quad j = 1,..., J$$

$$\sum_{i=1}^{n}\hat{z}_{i,j}.[\frac{-\sigma_j^2 + (\xi_i - m_j)^2}{\sigma_j^2.(\xi_i - m_j)^2}] = 0, \quad j = 1,..., J$$

$$\sum_{i=1}^{n}\hat{z}_{i,j}.\sigma_j^2 = \sum_{i=1}^{n}\hat{z}_{i,j}.(\xi_i^2 - 2.m_j.\xi_i + m_j^2), \quad j = 1,..., J$$

$$\sum_{i=1}^{n}\hat{z}_{i,j}.\xi_i^2 - 2.m_j.\sum_{i=1}^{n}\hat{z}_{i,j}.\xi_i + m_j^2.\sum_{i=1}^{n}\hat{z}_{i,j} = \sum_{i=1}^{n}\hat{z}_{i,j}.\sigma_j^2, \quad j = 1,..., J \qquad (31)$$

Pour obtenir σ_j, on a :

$$\frac{\partial Q(\boldsymbol{\theta}, \boldsymbol{\theta}')}{\partial \sigma_j} = 0 \Rightarrow \sum_{i=1}^{n}\hat{z}_{i,j}.[-\frac{2}{\sigma_j} + \frac{(\xi_i - m_j)^2}{\sigma_j^3}] = 0, \quad j = 1,..., J$$

$$\Rightarrow 2.\sigma_j^2.\sum_{i=1}^{n}\hat{z}_{i,j} = \sum_{i=1}^{n}\hat{z}_{i,j}.(\xi_i - m_j)^2, \quad j = 1,..., J \qquad (32)$$

$$\sigma_j^2 = \frac{\sum_{i=1}^{n}\hat{z}_{i,j}.(\xi_i - m_j)^2}{2.\sum_{i=1}^{n}\hat{z}_{i,j}}, \quad j = 1,..., J \qquad (33)$$

Les équations (31-33) constituent un ensemble d'équations non linéaires couplées, on peut utiliser la primitive fsolve de Matlab pour trouver des solutions admissibles.

II.2.3. Codeur optimisés, exemples de performances sur canal AWGN

Les figures 7, 8 montrent les distributions du spectre dans le cas $n = 3$ et $Q = Q_a = 2$, avec différentes matrices $\boldsymbol{A}$, toujours avec $\boldsymbol{T}_{i,j} = \boldsymbol{T}_{shift}$ pour $i = j$ et $\boldsymbol{T}_{i,j} = \boldsymbol{T}_{tent}$ pour $i \neq j$.

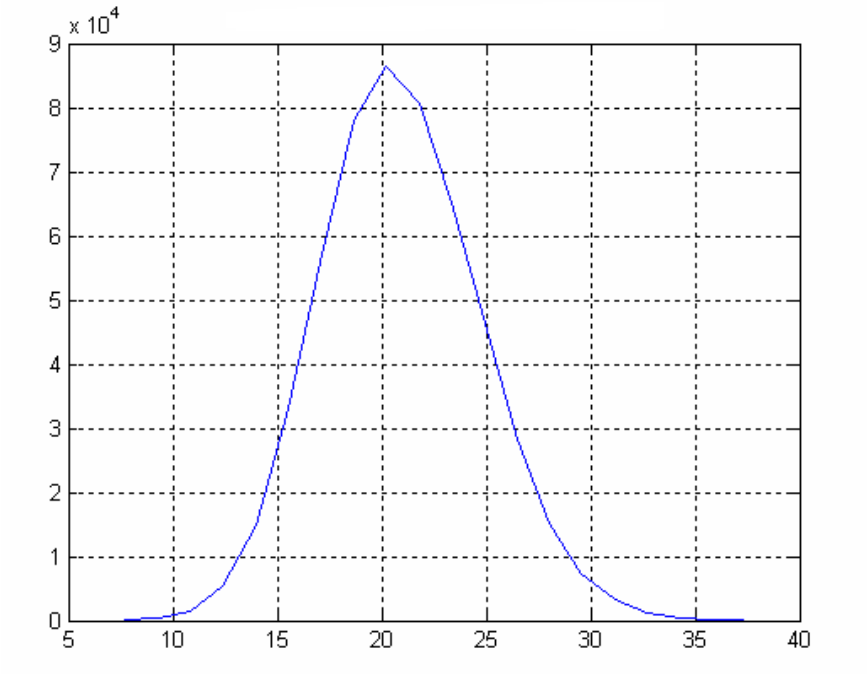


Figure. 6: spectre de distance pour $\boldsymbol{A} = \begin{pmatrix} 1 & 1 & 1 \\ 2 & 1 & 1 \\ -2 & 8 & 1 \end{pmatrix}$, *n = 3,* Q = Q_a = *2*

Les meilleurs résultats obtenus sont donnés sur la figure 8 où la distance minimale est égale $d_{\text{free}} \cong 13$.

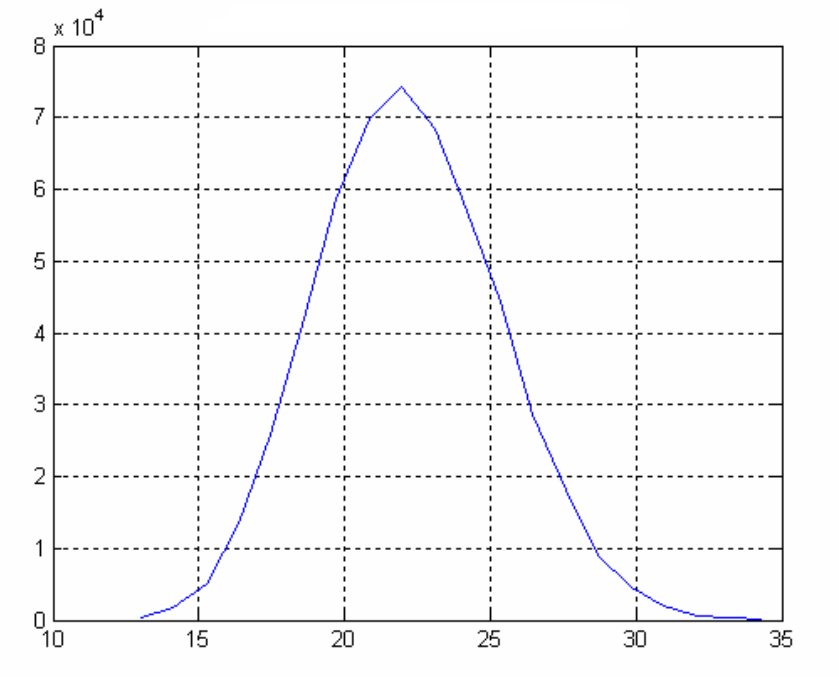


Figure. 7: spectre de distance pour $\boldsymbol{A} = \begin{pmatrix} 1 & 1 & 1 \\ 4 & 1 & 1 \\ -6 & 2 & 1 \end{pmatrix}$, $n = 3$, $Q = Q_a = 2$

On présente alors quelques résultats de simulation concernant le taux d'erreur binaire en fonction du rapport signal sur bruit sur un canal gaussien AWGN, en utilisant les optimisations obtenue par le calcul du spectre des distances.

L'algorithme de décodage appliqué, pour décoder les séquences chaotiques, est l'algorithme de Viterbi. Cet algorithme est rapidement limité par la complexité en le nombre d'états, qui est égale à $2^{n.(Q+1)}$.

La figure 8 montre les résultats pour $n = 2$ $a =$ 8 ou 5, $Q = 3$ et différentes valeurs de Q_a.

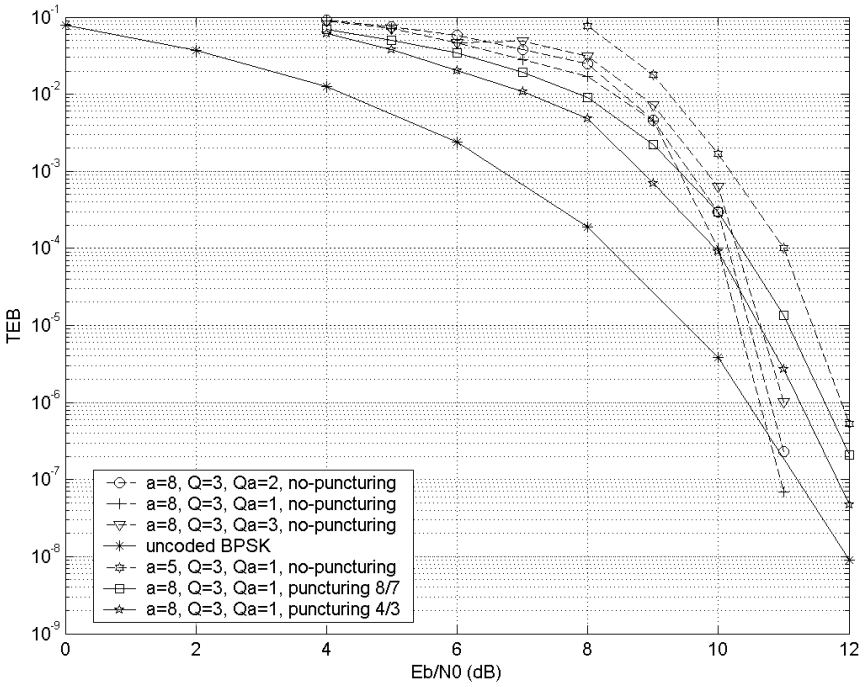


Figure. 8: *Performances de la modulation codée chaotique sur canal AWGN, n = 2 a = 8 ou 5, Q = 3*

Le codeur chaotique est plus performant que le système BPSK non codée pour un rapport signal sur bruit (SNR) élevé grâce à des bonnes propriétés asymptotiques avec une distance libre suffisante. Cependant la faiblesse de ce type de codage réside dans un rendement de codage très faible : en effet lorsque l'on décale les bits un par un pour obtenir le gain de codage le plus élevé possible le rendement obtenu est égal à : $\frac{1}{n.(Q+1)}$, ce qui est effectivement très faible. Il existe plusieurs solutions pour l'améliorer.

La première est de faire entrer les bits dans le codeur par des groupes de *k* bits. Dans ce cas le taux de codage est égal $\frac{k}{n.(Q+1)}$. Cependant, cela réduit le degré de corrélation entre deux états et rend le treillis non binaire. Nous n'avons pas constaté sur les exemples testés d'améliorations des performances avec cette méthode. Au contraire, une nette dégradation du TEB a toujours été observée.

La seconde méthode consiste à utiliser des codes poinçonnés (punctured codes) pour augmenter le rendement de codage.

La figure 8 montre des résultats avec les meilleurs motifs de poinçonnage pour des taux de poinçonnage respectivement égaux à : 9/7 et 3/2 que nous avons trouvés expérimentalement dans le cas : $a = 8$, $Q = 3$, $Q_a = 1$. Les performances illustrées sur la figure 8 montrent une meilleure pente asymptotique à forts SNR's (gain de diversité plus élevé) que le système non-codé ; cependant si on considère une plage pratique d'utilisation d'un tel codeur en terme de rapport signal à bruit, il est clair que le système non-codé est préférable à cette modulation codée chaotique poinçonnée.
En d'autres termes, il apparaît nettement que pour $n = 2$, la modulation codée chaotique n'apporte pas de gain suffisant par rapport à un système non codé.

Remarque : Comme il était prévu par les résultats du spectre de distances, les performances du code avec la matrice $\boldsymbol{A} = \begin{bmatrix} 1 & -1 \\ 5 & 1 \end{bmatrix}$ sont dégradées par rapport au cas $\boldsymbol{A} = \begin{bmatrix} 1 & -1 \\ 8 & 1 \end{bmatrix}$.

La figure 9 montres le taux d'erreur binaire en fonction du rapport signal dans le cas $n = 3$ pour $Q = 2$ et $\boldsymbol{A} = \begin{pmatrix} 1 & 1 & 1 \\ 2 & 1 & 1 \\ -2 & 8 & 1 \end{pmatrix}$ (cf spectre des distances de la figure 6, la distance minimale est égale à 6). Le nombre d'états pour $n = 3$ et $Q = 2$ est égal à $2^9 = 512$. Grâce à une meilleure distance libre par rapport au cas $n = 2$, les performances du BER sont améliorées et le codeur chaotique devient plus performant que le système BPSK non codé à partir d'un rapport signal sur bruit égale à 8,5 dB. Les meilleurs codes poinçonnés, avec des taux de poinçonnage égaux à 9/7 et 3/2, sont incapables de dépasser le système BPSK non codé même à forts rapport signal sur bruit car la distance libre diminue rapidement avec le poinçonnage.

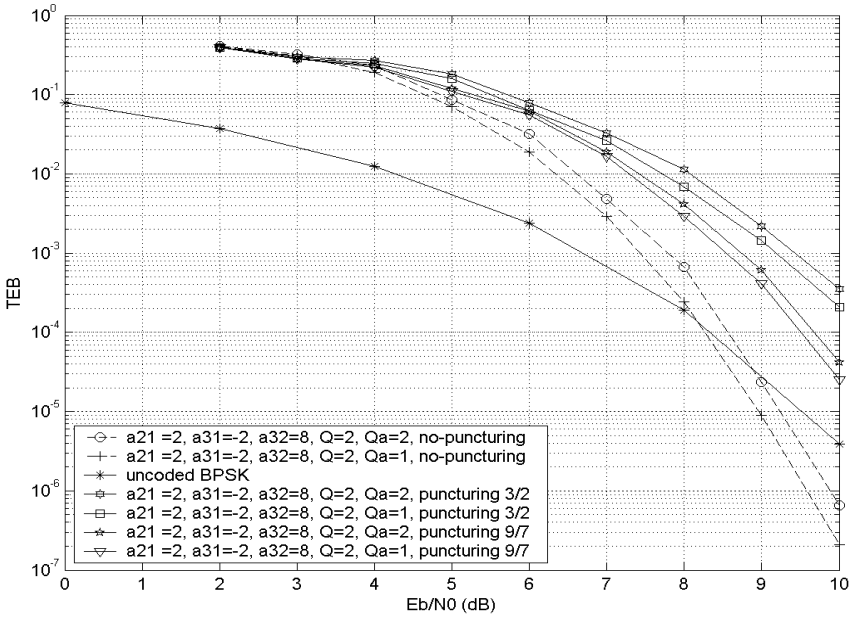


Figure 9: *Performances de la modulation codée chaotique sur canal AWGN, n = 3, Q = 2,*

$$\boldsymbol{A} = \begin{pmatrix} 1 & 1 & 1 \\ 2 & 1 & 1 \\ -2 & 8 & 1 \end{pmatrix}.$$

Les meilleures performances du codeur pour $n = 3$, $Q = 2$ sont obtenues avec la matrice $\boldsymbol{A} = \begin{pmatrix} 1 & 1 & 1 \\ 4 & 1 & 1 \\ -6 & 2 & 1 \end{pmatrix}$ et elles sont données sur la figure.10. Cette fois avec une distance minimale nettement supérieure (égale à 13), les performances des codes poinçonnés (9/7 et 3/2) peuvent dépasser les performances du système BPSK non codé à forts rapport signal à bruit.

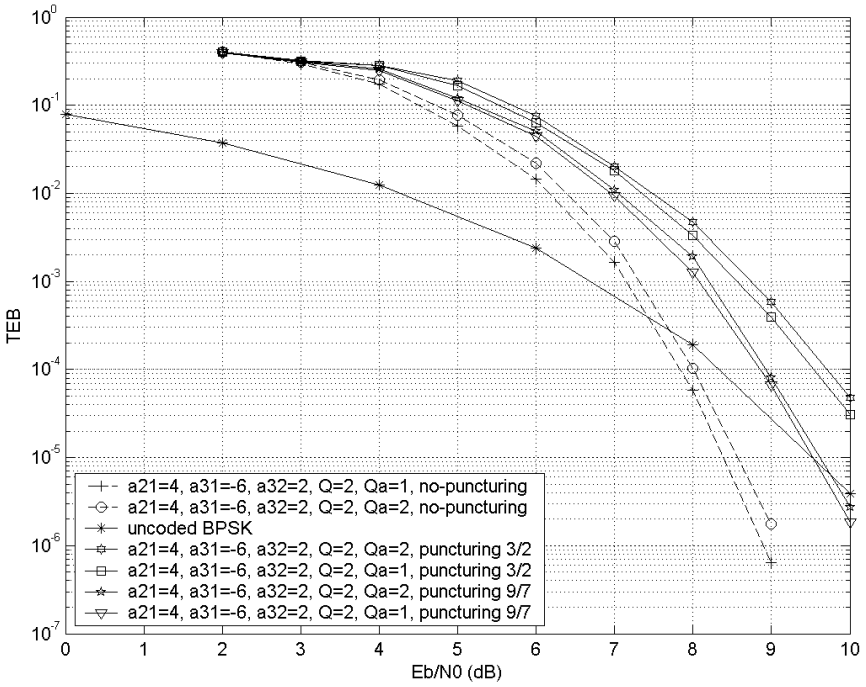


Figure 10: *Performances de la modulation codée chaotique sur canal AWGN, n = 3, Q = 2,*

$$\boldsymbol{A} = \begin{pmatrix} 1 & 1 & 1 \\ 4 & 1 & 1 \\ -6 & 2 & 1 \end{pmatrix}.$$

Pour compléter ces résultats, la figure.11 montre l'évolution des performances avec l'augmentation de *Q* (diminution du pas de quantification) ce qui conduit à une augmentation importante du nombre d'états du codeur. Pour ces résultats, l'optimisation du spectre des distances du codeur conduit pour la matrice $\boldsymbol{A}$ à $\boldsymbol{A} = \begin{pmatrix} 1 & 1 & 1 \\ 2 & 1 & 1 \\ 3 & -6 & 1 \end{pmatrix}$ avec une distance libre de 8.

Il est clair que l'utilisation d'un pas de quantification plus faible va rendre le système plus sensible au bruit additif ; on peut donc s'attendre à une diminution des performances pour le TEB. Par exemple avec $n = 3$ et $Q = 3$ on obtient un codeur à 4096 états.

Les résultats de la figure 11 confirment bien cette analyse, on constate bien que l'augmentation du niveau de quantification entraîne des pertes. Quand on compare les résultats des figures 10 et 11, la perte en termes de SNR pour un BER de 10^{-4}, 10^{-5} est d'environ 1 dB. De plus, les codes poinçonnés sont incapables de dépasser le système BPSK non codé même à forts SNR's.

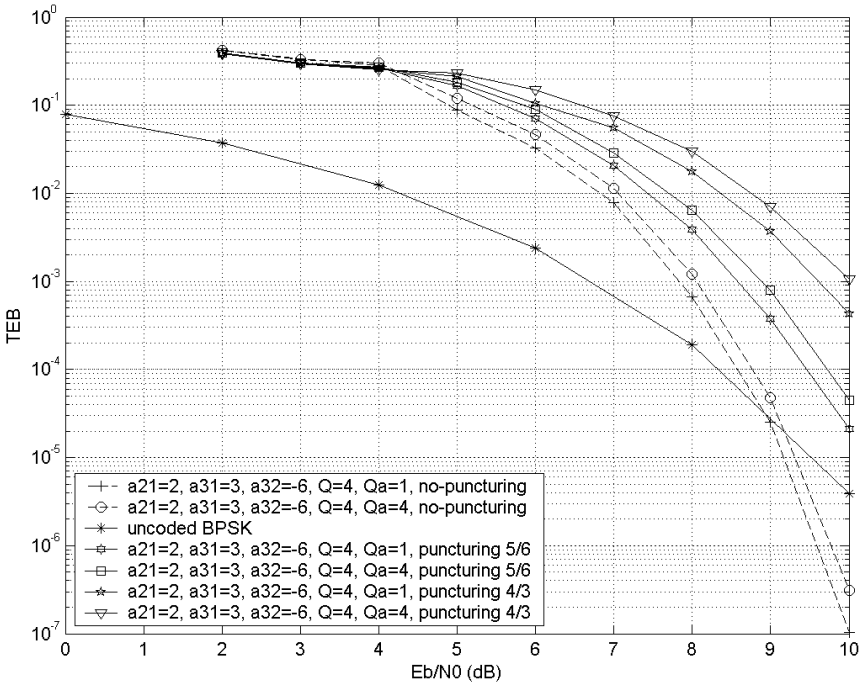


Figure 11: *Performances de la modulation codée chaotique sur canal AWGN, n = 3, Q = 3,*

$$\boldsymbol{A} = \begin{pmatrix} 1 & 1 & 1 \\ 2 & 1 & 1 \\ 3 & -6 & 1 \end{pmatrix}..$$

III. CONCATÉNATION DU CODEUR CHAOTIQUE AVEC UN CODE TEMPS ESPACE EN BLOC (STBC) DE TYPE ALAMOUTI.

III.1. Calcul de la PEP (Pairwise Error Probability)

Le but de cette partie est d'étudier le comportement du codeur chaotique dans un contexte plus général et plus complexe, lorsqu'il est concaténé avec un code temps espace en bloc de type STBC, code prévu pour une utilisation en antennes multiples (figure 12). Le schéma de code STBC retenu est celui bien connu d'Alamouti [47].

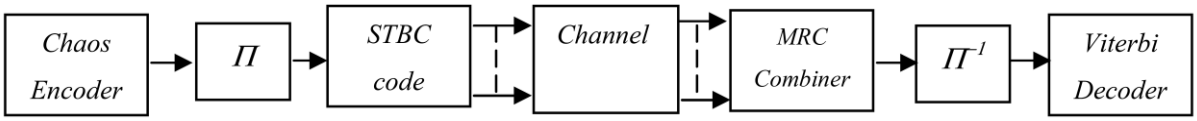


Figure 12: Schéma de transmission pour le codeur chaotique concaténé avec un code STBC (Alamouti)

Pour simplifier, en considérant le cas où $n = 2$, $\boldsymbol{x}_k$ sera un vecteur de deux symboles analogique transmis : $\boldsymbol{x}_k = \begin{pmatrix} x_1(k) \\ x_2(k) \end{pmatrix}$. Ces deux symboles seront transmis sur deux antennes d'émission en utilisant le schéma d'Alamouti avec la matrice temps espace : $\boldsymbol{S} = \begin{pmatrix} x_1(k) & -x_2^*(k) \\ x_2(k) & x_1^*(k) \end{pmatrix}$, ou dans le cas des symbole réel transmis : $\boldsymbol{S} = \begin{pmatrix} x_1(k) & -x_2(k) \\ x_2(k) & x_1(k) \end{pmatrix}$.

Dans la suite, les calculs seront faits en considérant des canaux non sélectifs en fréquence (ce qui peut être obtenu en utilisant des modulations multiporteuses (OFDM)) mais fortement sélectifs en temps avec la possibilité de considérer des variations temporelles entre deux symboles transmis successifs dans les cas extrêmes. De plus, dans les calculs qui suivent, nous considèrerons uniquement le cas d'un système multiantennes (MIMO) avec deux antennes d'émission et une antenne de réception.

Dans le cas de canaux quasi-statiques non sélectifs en fréquence, les paramètres des canaux restent constants sur la durée d'un paquet transmis mais changent complètement d'un paquet à un autre et il n'est pas nécessaire d'utiliser un entrelaceur entre le codeur de canal et le code temps espace STBC.

Le signal reçu dans deux créneaux temporels successifs s'écrit :

$$\begin{aligned} y_1(n) &= h_{11}x_1(n) + h_{21}x_2(n) + n_1(n) \\ y_1(n+1) = y_2(n) &= -h_{11}x_2^*(n) + h_{21}x_1^*(n) + n_2(n) \end{aligned} \qquad (34)$$

h_{11} et h_{21} sont deux variables aléatoire complexe de moyenne nulle et de variance égale a $2\sigma_h^2$

n_1 et n_2 sont deux échantillons de bruit blanc additif gaussien complexe de moyenne zéro et de variance égale a $2\sigma^2$.

Dans le cas particulier des codes STBC orthogonaux comme celui d'Alamouti, la détection à Maximum de Vraisemblance des signaux reçus $y_1(n)$ se réduit de façon remarquable à la technique MRC (Maximum Ratio Combining) souvent utilisée en réception pour des canaux sélectifs en fréquence.

Par combinaison MRC (Maximum Ratio Combining) on obtient alors les variables décisionnelles $Z_1(n)$ et $Z_2(n)$ définies par :

$$Z_1(n) = h_{11}^*.y_1(n) + h_{21}.y_1^*(n+1)$$
$$Z_2(n) = h_{21}^*.y_1(n) - h_{11}.y_1^*(n+1)$$

Ce qui peut encore se réécrire sous la forme :

$$\begin{aligned} Z_1(n) &= (\ |h_{11}|^2 + |h_{21}|^2).x_1(n) + h_{11}^*.n_1(n) + h_{21}.n_2^*(n) \\ Z_2(n) &= (\ |h_{11}|^2 + |h_{21}|^2).x_2(n) + h_{21}^*.n_1(n) - h_{11}.n_2^*(n) \end{aligned} \quad (35)$$

On peut alors étudier la Pairwise Error Probability (PEP) à partir de l'équation (35). En notant B= ($|h_{11}|^2$+$|h_{21}|^2$), Il y aura alors une faute dans la détection au niveau du décodeur de Viterbi, lorsqu'on envoie la séquence : $\boldsymbol{x} = (x_1(m), x_2(m), ..., x_1(m+L-1), x_2(m+L-1))$, à chaque fois que la métrique cumulée sur une boucle de longueur L sera plus petite sur le chemin erroné que sur le chemin réel. Nous appellerons : $\boldsymbol{x}' = (x_1'(m), x_2'(m), ..., x_1'(m+L-1), x_2'(m+L-1))$ la séquence erronée.

Ceci se traduit par l'équation :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | h_{11}, h_{21}, \boldsymbol{x}) = \text{Proba}(\sum_{n=m}^{L+m-1} \left| Z_1(n) - (\ |h_{11}|^2 + |h_{21}|^2).x_1(n) \right|^2 + \left| Z_2(n) - (\ |h_{11}|^2 + |h_{21}|^2).x_2(n) \right|^2$$
$$< \sum_{n=m}^{L+m-1} \left| Z_1(n) - (\ |h_{11}|^2 + |h_{21}|^2).x_1'(n) \right|^2 + \left| Z_2(n) - (\ |h_{11}|^2 + |h_{21}|^2).x_2'(n) \right|^2) \quad (36)$$

Transformant (36), on obtient :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | h_{11}, h_{21}, \boldsymbol{x})$$
$$= \text{Proba}(B^2.[\sum_{n=m}^{L+m-1} \left[x_1^2(n) - x_1'^2(n) \right] + \left[x_2^2(n) - x_2'^2(n) \right]] <$$
$$2.B.\sum_{n=m}^{L+m-1} \text{Re}(Z_1(n)).\left[x_1(n) - x_1'(n) \right] + \text{Re}(Z_2(n)).\left[x_2(n) - x_2'(n) \right])$$

Avec

$$\mathrm{Re}(Z_1(n)) = B.x_1(n) + N_1^R(n)$$
$$\mathrm{Re}(Z_2(n)) = B.x_2(n) + N_2^R(n)$$

et

$$N_1^R(n) = \mathrm{Re}(N_1(n)) = \mathrm{Re}(h_{11}^*.n_1(n) + h_{21}.n_2^*(n))$$
$$N_2^R(n) = \mathrm{Re}(N_2(n)) = \mathrm{Re}(h_{21}^*.n_1(n) - h_{11}.n_2^*(n)) \quad (37)$$

On obtient alors la règle suivante :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | h_{11}, h_{21}, \boldsymbol{x})$$
$$= \mathrm{Proba}(2.B.\sum_{n=m}^{L+m-1} N_1^R(n).(x_1(n) - x_1'(n)) + N_2^R(n).(x_2(n) - x_2'(n)) >$$
$$B^2.(\sum_{n=m}^{L+m-1} (x_1(n) - x_1'(n))^2 + (x_2(n) - x_2'(n))^2)) \quad (38)$$

Notant :

$$h_{11} = h_{11}^R + j.h_{11}^I$$
$$h_{21} = h_{21}^R + j.h_{21}^I$$
$$n_1(n) = n_1^R(n) + j.n_1^I(n)$$
$$n_2(n) = n_2^R(n) + j.n_2^I(n)$$

On suppose que :

$$E(\left| h_{11}^R \right|^2) = E(\left| h_{21}^R \right|^2) = E(\left| h_{11}^I \right|^2) = E(\left| h_{21}^I \right|^2) = \sigma_h^2$$ (h_{11} et h_{21} sont considérées ici comme des constantes).

On suppose également que :

$$E(n_1^R(n).n_1^I(n)) = E(n_1^R(n).n_2^R(n)) = E(n_1^R(n).n_2^I(n))$$
$$= E(n_1^I(n).n_2^R(n)) = E(n_1^I(n).n_2^I(n)) = E(n_2^R(n).n_2^I(n)) = 0 \quad (39)$$

et:

$$E(n_1^I(n).n_1^I(n)) = E(n_1^R(n).n_1^R(n)) = \ldots$$
$$\ldots = E(n_2^I(n).n_2^I(n)) = E(n_2^R(n).n_2^R(n)) = \sigma^2$$

On en déduit que :

$$E(\left| N_1^R(n) \right|^2) = |h_{11}|^2.E(n_1^2(n)) + |h_{21}|^2.E(n_2^2(n)) = \sigma^2(|h_{11}|^2 + |h_{21}|^2) = B.\sigma^2 \quad (40)$$

$$E(\left| N_2^R(n) \right|^2) = E(\left| N_1^R(n) \right|^2) = B.\sigma^2$$

La variable aléatoire $D = 2.B.\sum_{n=m}^{L+m-1} N_1^R(n).\left[x_1(n) - x_1^{'}(n)\right] + N_2^R(n).\left[x_2(n) - x_2^{'}(n)\right]$ pour les séquences ***x*** et ***x'*** données, est clairement une variable gaussienne aléatoire puisqu'elle est obtenue comme une combinaison de variables aléatoires gaussiennes. Sa moyenne est nulle comme celles de $N_1^R(n)$ et $N_2^R(n)$, et sa variance est égale à:

$$\Sigma = 4.B.\sigma^2 B^2.\sum_{n=m}^{L+m-1}\left[x_1(n) - x_1^{'}(n)\right]^2 + \left[x_2(n) - x_2^{'}(n)\right]^2 \qquad (41)$$

$$\Sigma = 4.B^3.C^2\sigma^2 \quad \text{avec} \quad C^2 = \sum_{n=m}^{L+m-1}\left[x_1(n) - x_1^{'}(n)\right]^2 + \left[x_2(n) - x_2^{'}(n)\right]^2$$

On peut alors calculer la probabilité d'erreur (PEP) conditionnellement à $h_{11}, h_{21}, \boldsymbol{x}$, sous la forme :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | h_{11}, h_{21}, \boldsymbol{x}) = P(D > A) \qquad (42)$$

Avec $A = B^2.\sum_{n=m}^{L+m-1}\left[x_1(n) - x_1^{'}(n)\right]^2 + \left[x_2(n) - x_2^{'}(n)\right]^2 = B^2.C^2$

Donc finalement:

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | h_{11}, h_{21}, \boldsymbol{x}) = \frac{1}{2\sqrt{2.\pi}\sigma.B^{3/2}.C}.\int_{B^2C^2}^{+\infty} e^{-\frac{x^2}{8.B^3.C^2.\sigma^2}}.dx \qquad (43)$$

On utilise alors la nouvelle variable : $u = x/(2.\sqrt{2}.B^{3/2}.C.\sigma)$, on obtient :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | h_{11}, h_{21}, \boldsymbol{x}) = \frac{1}{\sqrt{\pi}}.\int_{\frac{B^{1/2}C}{2.\sqrt{2}.\sigma}}^{+\infty} e^{-u^2}.du \qquad (44)$$

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | h_{11}, h_{21}, \boldsymbol{x}) = \frac{1}{2}.erfc\left(\frac{B^{1/2}C}{2.\sqrt{2}.\sigma}\right) \qquad (45)$$

En utilisant la définition de *B* et *C*, on obtient finalement la probabilité d'erreur conditionnelle :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | h_{11}, h_{21}, \boldsymbol{x}) = \frac{1}{2}.erfc\left(\frac{(|h_{11}|^2 + |h_{21}|^2)^{1/2}.\left(\sum_{n=m}^{L+m-1}\left[x_1(n) - x_1'(n)\right]^2 + \left[x_2(n) - x_2'(n)\right]^2\right)^{1/2}}{2.\sqrt{2}.\sigma}\right) \quad (46)$$

Pour continuer le calcul de la probabilité d'erreur et moyenner sur les paramètres du canal, on se sert du fait que la variable $B = (|h_{11}|^2 + |h_{21}|^2)$ est une variable χ 2 (Chi 2) de degré 4 ce qui s'écrit:

$$p_B(x) = \frac{1}{\sigma_h^d.2^{d/2}.\Gamma(d/2)}.x^{d/2-1}.\exp(-\frac{x}{2.\sigma_h^2}) \quad (47)$$

avec d = 4, moyenner l'équation (47) sur B, conduit à :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | \boldsymbol{x}) = \int_0^{+\infty} \frac{1}{2}.erfc(\frac{x^{1/2}C}{2.\sqrt{2}.\sigma}).\frac{1}{4.\sigma_h^4.\Gamma(2)}.x.e^{-x/2.\sigma_h^2}.dx$$

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | \boldsymbol{x}) = \frac{1}{8.\sigma_h^4}\int_0^{+\infty} erfc(\frac{x^{1/2}C}{2.\sqrt{2}.\sigma}).x.e^{-x/2.\sigma_h^2}.dx \quad (48)$$

(En utilisant l'égalité: $\Gamma(n) = (n-1)!$, on a bien sûr : $\Gamma(2) = 1$)

Pour obtenir (48), il faut calculer $H = \int_0^{+\infty} erfc(\frac{x^{1/2}C}{2.\sqrt{2}.\sigma}).x.e^{-x/2.\sigma_h^2}.dx$

En utilisant les variables suivantes:

$$\begin{aligned} du &= x.e^{-x/2.\sigma_h^2}.dx \\ v &= erfc(\frac{x^{1/2}C}{2.\sqrt{2}.\sigma}) \end{aligned} \quad => \quad \begin{aligned} u &= -2.x.\sigma_h^2.e^{-x/2.\sigma_h^2} - 4.\sigma_h^4.e^{-x/2.\sigma_h^2} \\ dv &= -\frac{1}{2.\sqrt{2.\pi}\sigma}.e^{-\frac{x.C^2}{8.\sigma^2}}.x^{-1/2}.C.dx \end{aligned}$$

Il est possible de calculer H en intégrant par parties.

On obtient:

$$H = \left[(-2.x.\sigma_h^2.e^{-x/2.\sigma_h^2} - 4.\sigma_h^4.e^{-x/2.\sigma_h^2}).erfc(\frac{x^{1/2}C}{2.\sqrt{2}.\sigma}) \right]_0^{+\infty}$$

$$- \int_0^{+\infty} (2.x.\sigma_h^2.e^{-x/2.\sigma_h^2} + 4.\sigma_h^4.e^{-x/2.\sigma_h^2}).\frac{C}{2.\sqrt{2.\pi}\sigma}.e^{-\frac{x.C^2}{8.\sigma^2}}.x^{-1/2}.dx$$

$$= 4.\sigma_h^4 - \frac{C}{\sqrt{2\pi}.\sigma}\int_0^{+\infty} \sigma_h^2.x^{1/2}.e^{-\frac{x.C^2}{8.\sigma^2}}.e^{-x/2.\sigma_h^2}.dx - \frac{C}{\sqrt{2\pi}.\sigma}.\int_0^{+\infty} 2.\sigma_h^4.x^{-1/2}.e^{-\frac{x.C^2}{8.\sigma^2}}.e^{-x/2.\sigma_h^2}.dx \quad (49)$$

Donc il reste à évaluer les intégrales suivantes :

$$I = \int_0^{+\infty} x^{1/2}.e^{-\frac{x.C^2}{8.\sigma^2}}.e^{-x/2.\sigma_h^2}.dx \text{ et } J = \int_0^{+\infty} x^{-1/2}.e^{-\frac{x.C^2}{8.\sigma^2}}.e^{-x/2.\sigma_h^2}.dx .$$

Pour calculer *I* et *J*, on utilise la variable $u = \frac{C^2.\sigma_h^2 + 4.\sigma^2}{8.\sigma^2.\sigma_h^2}.x$ et on obtient :

$$I = \left(\frac{8.\sigma^2.\sigma_h^2}{C^2.\sigma_h^2 + 4.\sigma^2} \right)^{3/2} . \int_0^{+\infty} u^{1/2}.e^{-u}.du = \left(\frac{8.\sigma^2.\sigma_h^2}{C^2.\sigma_h^2 + 4.\sigma^2} \right)^{3/2} .\Gamma(3/2) \quad (50)$$

et

$$J = \left(\frac{8.\sigma^2.\sigma_h^2}{C^2.\sigma_h^2 + 4.\sigma^2} \right)^{1/2} . \int_0^{+\infty} u^{-1/2}.e^{-u}.du = \left(\frac{8.\sigma^2.\sigma_h^2}{C^2.\sigma_h^2 + 4.\sigma^2} \right)^{1/2} .\Gamma(1/2) \quad (51)$$

En utilisant les égalités pour fonction $\Gamma : \left(\Gamma(z) = \int_0^{+\infty} t^{z-1}.e^{-t}.dt \right)$

$$\Gamma(z+1) = z.\Gamma(z)$$

$$\Gamma(z).\Gamma(1-z) = \frac{\pi}{\sin(\pi.z)} \quad (52)$$

On obtient: $\Gamma(1/2) = \sqrt{\pi}$ et $\Gamma(3/2) = \sqrt{\pi}/2$ (53)

Ensuite, en reportant ces résultats dans (49), nous trouvons :

$$H = 4.\sigma_h^4 - \frac{C}{\sqrt{2\pi}.\sigma}\sigma_h^2.\left(\frac{8.\sigma^2.\sigma_h^2}{C^2.\sigma_h^2 + 4.\sigma^2} \right)^{3/2} \sqrt{\pi}/2 - \frac{C}{\sqrt{2\pi}.\sigma}.2.\sigma_h^4.\left(\frac{8.\sigma^2.\sigma_h^2}{C^2.\sigma_h^2 + 4.\sigma^2} \right)^{1/2} \sqrt{\pi}$$

$$H = 4.\sigma_h^4 - \frac{C}{\sqrt{2\pi}}\sigma^2.\sigma_h^2.\left(\frac{8.\sigma_h^2}{C^2.\sigma_h^2 + 4.\sigma^2} \right)^{3/2} \sqrt{\pi}/2 - \frac{C}{\sqrt{2\pi}}.2.\sigma_h^4.\left(\frac{8.\sigma_h^2}{C^2.\sigma_h^2 + 4.\sigma^2} \right)^{1/2} \sqrt{\pi} \quad (54)$$

Nous obtenons finalement la probabilité d'erreur pour une séquence ***x*** de longueur donnée *L* :

$$P_e(\boldsymbol{x}\to\boldsymbol{x}'|\boldsymbol{x})=\frac{1}{8.\sigma_h^4}\left[4.\sigma_h^4-\frac{C.\sqrt{\pi}}{2.\sqrt{2\pi}}\sigma^2.\sigma_h^2.\left(\frac{8.\sigma_h^2}{C^2.\sigma_h^2+4.\sigma^2}\right)^{3/2}-\frac{C.\sqrt{\pi}}{\sqrt{2\pi}}.2.\sigma_h^4.\left(\frac{8.\sigma_h^2}{C^2.\sigma_h^2+4.\sigma^2}\right)^{1/2}\right]$$

$$P_e(\boldsymbol{x}\to\boldsymbol{x}'|\boldsymbol{x})=\frac{1}{2}-\frac{C.\sigma^2.\sigma_h}{(C^2.\sigma_h^2+4.\sigma^2)^{3/2}}-\frac{C.\sigma_h}{2.(C^2.\sigma_h^2+4.\sigma^2)^{1/2}} \qquad (55)$$

Gain de diversité : Pour des SNR's élevés, il est intéressant de calculer le gain de diversité obtenu. En utilisant des développements limités classiques on arrive à :

$$\begin{aligned}P_e(\boldsymbol{x}\to\boldsymbol{x}'|\boldsymbol{x})&=\frac{1}{2}-\frac{C.\sigma^2.\sigma_h}{C^3.\sigma_h^3}\frac{1}{(1+4.\sigma^2/C^2.\sigma_h^2)^{3/2}}-\frac{C.\sigma_h}{2.C.\sigma_h}\frac{1}{(1+4.\sigma^2/C^2.\sigma_h^2)^{1/2}}\\&=\frac{1}{2}-\frac{\sigma^2}{C^2.\sigma_h^2}.(1-\frac{3}{2}.4.\sigma^2/C^2.\sigma_h^2+\frac{30.\sigma^4}{C^4.\sigma_h^4}+\vartheta(\sigma^4))-\frac{1}{2}.(1-\frac{1}{2}.4.\sigma^2/C^2.\sigma_h^2+\frac{6.\sigma^4}{C^4.\sigma_h^4}+\vartheta(\sigma^4))\\&=\frac{6.\sigma^4}{C^4.\sigma_h^4}+\vartheta(\sigma^4) \qquad (56)\end{aligned}$$

Avec : $\sigma^2=\frac{1}{6.(Q+1).n.(E_b/N_0)}$, (voir 61) on obtient:

$$P_e(\boldsymbol{x}\to\boldsymbol{x}'|\boldsymbol{x})_{|\text{high SNR's}}\approx\frac{1}{6.C^4.\sigma_h^4.n^2.(Q+1)^2.(E_b/N_0)^2} \qquad (57)$$

On voit donc (l'exposant de E_b/N_0 à haut SNR étant égal à 2), que l'on obtient un gain de diversité égal à deux.

Pour calculer la probabilité complète : $P_e(\boldsymbol{x}\to\boldsymbol{x}')$, il est nécessaire d'étudier le spectre des distances du codeur, en d'autres termes nous avons besoin de moyenner (55) sur la distribution *C*.

Comme nous l'avons vu aux paragraphes 1.2.2.1 et 1.2.2.2, la distribution du spectre des distances du codeur chaotique peut être approximée avec une bonne précision par un mélange de lois gaussiennes ou de lois de Rayleigh. L'obtention de la PEP définitive nécessite donc de moyenner l'équation (55) par rapport à ces types de distributions. Pour ce faire, il est plus commode d'utiliser des développements en série entière (D.S.E) pour les fractions rationnelles où intervient *C*.

Pour exprimer P_e en fonction de σ^2, nous développons donc $P_e(\boldsymbol{x} \to \boldsymbol{x}'|\boldsymbol{x})$ en série entière en utilisant les règles suivantes.

$$\frac{1}{(1+4.\sigma^2/x^2\sigma_h^2)^{1/2}} = 1+\sum_{n=1}^{+\infty}\frac{(-1)^n.2^{n-1}.(2n)!}{(n!)^2}.\frac{\sigma^{2n}}{\sigma_h^{2n}}.\frac{1}{x^{2n}} \text{ Pour: } 4.\sigma^2/x^2\sigma_h^2 < 1$$

Et

$$\frac{1}{(1+4.\sigma^2/x^2\sigma_h^2)^{3/2}} = 1+\sum_{n=1}^{+\infty}\frac{(-1)^n.2^n.(2n+1)!}{(n!)^2}.\frac{\sigma^{2n}}{\sigma_h^{2n}}.\frac{1}{x^{2n}} \text{ pour: } 4.\sigma^2/x^2\sigma_h^2 < 1$$

Ces calculs proviennent bien sûr de :

$$(1+x)^{\alpha} = 1+\alpha.x+\frac{\alpha.(\alpha-1)}{2!}.x^2+...+\frac{\alpha.(\alpha-1)...(\alpha-n+1)}{n!}.x^n+... \quad \text{pour } |x|<1$$

On a alors:

$$P_e(\boldsymbol{x} \to \boldsymbol{x}'|\boldsymbol{x}) = \frac{1}{2}-\frac{1}{2}.\frac{1}{(1+4.\sigma^2/C^2.\sigma_h^2)^{1/2}}-\frac{\sigma^2}{\sigma_h^2.C^2}.\frac{1}{(1+4.\sigma^2/C^2.\sigma_h^2)^{3/2}}$$

Pour des SNR's suffisamment élevés et donc pour des valeurs de σ^2 suffisamment faibles on a : $4.\sigma^2/x^2\sigma_h^2 < 1$ et donc nous pouvons écrire:

$$P_e(\boldsymbol{x} \to \boldsymbol{x}'|\boldsymbol{x}) = \frac{1}{2}-\frac{1}{2}.(1+\sum_{n=1}^{+\infty}\frac{(-1)^n.2^{n-1}.(2n)!}{(n!)^2}.\frac{\sigma^{2n}}{\sigma_h^{2n}}.\frac{1}{C^{2n}})+...$$
$$-\frac{\sigma^2}{\sigma_h^2.C^2}.(1+\sum_{n=1}^{+\infty}\frac{(-1)^n.2^n.(2n+1)!}{(n!)^2}.\frac{\sigma^{2n}}{\sigma_h^{2n}}.\frac{1}{C^{2n}})$$

On obtient:

$$P_e(\boldsymbol{x} \to \boldsymbol{x}'|\boldsymbol{x}) = \sum_{m=1}^{+\infty}\frac{(-1)^m.2^{m-1}.(2m+2)!}{((m+1)!)^2}.\frac{\sigma^{2(m+1)}}{\sigma_h^{2(m+1)}}.\frac{1}{C^{2(m+1)}}$$
$$+\sum_{n=1}^{+\infty}\frac{(-1)^{n+1}.2^n.(2n+1)!}{(n!)^2}.\frac{\sigma^{2(n+1)}}{\sigma_h^{2(n+1)}}.\frac{1}{C^{2(n+1)}}$$
$$P_e(\boldsymbol{x} \to \boldsymbol{x}'|\boldsymbol{x}) = \sum_{n=1}^{+\infty}\frac{(-1)^{n+1}.2^n.n.(n+1).(2n+1)!}{((n+1)!)^2}.\frac{\sigma^{2(n+1)}}{\sigma_h^{2(n+1)}}.\frac{1}{C^{2(n+1)}} \qquad (58)$$

C'est cette expression que nous allons utiliser pour le moyennage de $P_e(\boldsymbol{x} \to \boldsymbol{x}'|\boldsymbol{x})$ avec les lois de Gauss et de Rayleigh.

- ***Distribution gaussienne :***

Avec la fonction de densité de probabilité $p(x) = \frac{1}{\sqrt{2.\pi}.\sigma_j}.e^{-(x-m_j)^2/2.\sigma_j^2}$ pour la variable aléatoire C^2 et en utilisant le fait que la fonction $f(x) = e^{-(x-m_j)^2/2.\sigma_j^2}$ prend des valeurs significatifs sur l'intervalle $[d_{\min}, +\infty[$ nous avons :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}') = \sum_{n=1}^{+\infty} \frac{(-1)^{n+1}.2^n.n.(n+1).(2n+1)!}{((n+1)!)^2} . \frac{\sigma^{2(n+1)}}{\sigma_h^{2(n+1)}} . \int_{d_{\min}}^{+\infty} \frac{1}{\sqrt{2.\pi}.\sigma_j} \frac{e^{-(x-m_j)^2/2.\sigma_j^2}.dx}{x^{n+1}} \qquad (59)$$

On a:

$$\int_{d_{\min}}^{+\infty} \frac{e^{-(x-m_j)^2/2.\sigma_j^2}.dx}{x^{n+1}} = \int_{d_{\min}-m_j}^{+\infty} \frac{e^{-u^2/2.\sigma_j^2}.du}{(u+m_j)^{n+1}} = \int_{d_{\min}-m_j}^{m_j} \frac{e^{-u^2/2.\sigma_j^2}.du}{(u+m_j)^{n+1}} + \int_{m_j}^{+\infty} \frac{e^{-u^2/2.\sigma_j^2}.du}{(u+m_j)^{n+1}}$$

Utilisant l'expression classique pour x < 1:

$$\frac{1}{(1+x)^k} = \sum_{n=k}^{+\infty} (-1)^n n.(n-1)...(n-k+2).x^{n-k+1}$$
$$= \sum_{n=k}^{+\infty} (-1)^n . \frac{n!}{(n+1-k)!}.x^{n-k+1}$$

On obtient:

$$\int_{d_{\min}-m_j}^{m_j} \frac{e^{-u^2/2.\sigma_j^2}.du}{(u+m_j)^{n+1}}$$
$$= \frac{1}{m_j^{n+1}} . \sum_{p=n+1}^{+\infty} (-1)^p.p.(p-1)...(p-(n+1)+2) \int_{d_{\min}-m_j}^{m_j} (u/m_j)^{p-(n+1)+1} e^{-u^2/2.\sigma_j^2}.du$$
$$= \sum_{p=n+1}^{+\infty} \frac{1}{m_j^{p+1}}.(-1)^p.p.(p-1)...(p-n+1) \int_{d_{\min}-m_j}^{m_j} u^{p-n} e^{-u^2/2.\sigma_j^2}.du$$
$$= \sum_{p=n+1}^{+\infty} \frac{1}{m_j^{p+1}} (-1)^p . \frac{p!}{(p-n)!} . \int_{d_{\min}-m_j}^{m_j} u^{p-n} e^{-u^2/2.\sigma_j^2}.du$$

Et, pour la seconde intégrale, on a:

$$\int_{m_j}^{+\infty} \frac{e^{-u^2/2.\sigma_j^2}.du}{(u+m_j)^{n+1}} = \sum_{p=n+1}^{+\infty} (-1)^p . \frac{p\ !}{(p-n)!} . (m_j)^{p-n} \int_{m_j}^{+\infty} \frac{e^{-u^2/2.\sigma_j^2}}{u^{p-1}} .du$$

Utilisant :

$$I_p = \int_{d_{\min}-m_j}^{m_j} u^p e^{-u^2/2.\sigma_j^2} .du \quad \text{et} \quad J_p = \int_{m_j}^{+\infty} \frac{e^{-u^2/2.\sigma_j^2}}{u^{p-1}} .du$$

Nous obtenons finalement :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}') = \sum_{n=1}^{+\infty} \frac{(-1)^{n+1}.2^n.n.(n+1).(2n+1)!}{((n+1)!)^2} . \frac{\sigma^{2(n+1)}}{\sqrt{2.\pi}.\sigma_j.\sigma_h^{2(n+1)}} . [\sum_{p=n+1}^{+\infty} \frac{(-1)^p.p\ !}{(p-n)!} . (\frac{I_{p-n}}{m_j^{p+1}} + (m_j)^{p-n}.J_p)] \quad (60)$$

- ***Distribution de Rayleigh*** :

Avec la fonction de la densité de probabilité : $p(x) = \frac{(x-m_j)}{\sigma_j^2} .e^{-(x-m_j)^2/2.\sigma_j^2}$, on a :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}') = \sum_{n=1}^{+\infty} \frac{(-1)^{n+1}.2^n.n.(n+1).(2n+1)!}{((n+1)!)^2} . \frac{\sigma^{2(n+1)}}{\sigma_h^{2(n+1)}} . \int_{m_j}^{+\infty} \frac{(x-m_j)}{2.\sigma_j^2} \frac{e^{-(x-m_j)^2/2.\sigma_j^2}.dx}{x^{n+1}} \quad (61)$$

On peut écrire:

$$\int_{m_j}^{+\infty} \frac{(x-m_j).e^{-(x-m_j)^2/2.\sigma_j^2}.dx}{x^{n+1}} = \int_0^{+\infty} \frac{u.e^{-u^2/2.\sigma_j^2}.du}{(u+m_j)^{n+1}} = \int_0^{m_j} \frac{u.e^{-u^2/2.\sigma_j^2}.du}{(u+m_j)^{n+1}} + \int_{m_j}^{+\infty} \frac{u.e^{-u^2/2.\sigma_j^2}.du}{(u+m_j)^{n+1}}$$

On obtient :

$$\int_0^{m_j} \frac{u.e^{-u^2/2.\sigma_j^2}.du}{(u+m_j)^{n+1}}$$

$$= \frac{1}{m_j^{n+1}} . \sum_{p=n+1}^{+\infty} (-1)^p . p.(p-1)...(p-(n+1)+2) \int_0^{m_j} u.(u/m_j)^{p-(n+1)+1} e^{-u^2/2.\sigma_j^2} .du$$

$$= \sum_{p=n+1}^{+\infty} \frac{1}{m_j^{p+1}} .(-1)^p . p.(p-1)...(p-n+1) \int_0^{m_j} u^{p-n} e^{-u^2/2.\sigma_j^2} .du$$

$$= \sum_{p=n+1}^{+\infty} \frac{1}{m_j^{p+1}} (-1)^p . \frac{p!}{(p-n)!} . \int_0^{m_j} u^{p-n} e^{-u^2/2.\sigma_j^2} . du$$

$$= \sum_{p=n+1}^{+\infty} \frac{1}{m_j^{p+1}} (-1)^p . \frac{p!}{(p-n)!} . I'_{p-n}$$

Avec $I'_{p-n} = \int_0^{m_j} u^{p-n} e^{-u^2/2.\sigma_j^2} . du$

et:

$$\int_{m_j}^{+\infty} \frac{u.e^{-u^2/2.\sigma_j^2} . du}{(u+m_j)^{n+1}} = \sum_{p=n+1}^{+\infty} (-1)^p . \frac{p!}{(p-n)!} . (m_j)^{p-(n+1)+1} \int_{m_j}^{+\infty} \frac{e^{-u^2/2.\sigma_j^2}}{u^p} . du$$

$$= \sum_{p=n+1}^{+\infty} (-1)^p . \frac{p!}{(p-n)!} . (m_j)^{p-n} . J_p$$

Avec $J_p = \int_{m_j}^{+\infty} \frac{e^{-u^2/2.\sigma_j^2}}{u^p} . du$

Finalement on obtient :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}') = \sum_{n=1}^{+\infty} \frac{(-1)^{n+1} . 2^n . n . (n+1) . (2n+1)!}{((n+1)!)^2} . \frac{\sigma^{2(n+1)}}{2.\sigma_j^2 . \sigma_h^{2(n+1)}} . [\sum_{p=n+1}^{+\infty} \frac{(-1)^p \, p!}{(p-n)!} . (\frac{I'_{p-n}}{m_j^{p+1}} + (m_j)^{p-n} . J_p)] \quad (62)$$

Remarque:

Les intégrales dans les calculs (60-62) peuvent être calculées de manière récursive. Nous avons:

$$J_p = \int_{m_j}^{+\infty} \frac{e^{-u^2/2.\sigma_j^2}}{u^p} . du = -\sigma_j^2 \int_{m_j}^{+\infty} -\frac{u}{\sigma_j^2} \frac{e^{-u^2/2.\sigma_j^2}}{u^{p+1}} . du$$

$$= -\sigma_j^2 [\frac{e^{-u^2/2.\sigma_j^2}}{u^{p+1}}]_{m_j}^{+\infty} - (p+1).\sigma_j^2 . \int_{m_j}^{+\infty} \frac{e^{-u^2/2.\sigma_j^2}}{u^{p+2}} . du$$

$$= \sigma_j^2 . \frac{e^{-m_j^2/2.\sigma_j^2}}{(m_j)^{p+1}} - (p+1).\sigma_j^2 . J_{p+2}$$

Ce qui donne :

$$J_{p+2} = \frac{e^{-m_j^2/2.\sigma_j^2}}{(p+1).(m_j)^{p+1}} - \frac{1}{\sigma_j^2 . (p+1)} . J_p$$

En fonction de la parité de *p*, la formule précédente permet de calculer J_p à partir des premières valeurs J_1 ou J_0.

Pour J_1, on a:

$$J_1 = \int_{m_j}^{+\infty} \frac{e^{-u^2/2.\sigma_j^2}}{u}.du = -\sigma_j^2.\int_{m_j}^{+\infty} \frac{(-u/\sigma_j^2).e^{-u^2/2.\sigma_j^2}}{u^2}.du$$

$$= \int_{m_j^2/2.\sigma_j^2}^{+\infty} \frac{e^{-x}}{x}.dx = EI_1(m_j^2/2.\sigma_j^2)$$

Avec: $EI_1(x) = \int_x^{+\infty} \frac{e^{-t}}{t}.dt$

$$EI_1(x) = \int_x^{+\infty} \frac{e^{-t}.dt}{t} = \int_1^{+\infty} \frac{e^{-xt}.dt}{t} = e^{-x}.\int_0^{+\infty} \frac{e^{-xt}}{1+t}.dt = -\ln x - \gamma + \sum_{n=1}^{+\infty} \frac{(-1)^{n-1}.x^n}{n.n!}$$

Avec γ la constante d'Euler ($\gamma \approx 0.5773$).

Pour J_0, on a:

$$J_0 = \int_{m_j}^{+\infty} e^{-u^2/2.\sigma_j^2}.du = \sqrt{2}.\sigma_j \int_{m_j/\sqrt{2}.\sigma_j}^{+\infty} e^{-x^2}.dx = \sqrt{\frac{\pi}{2}}.\sigma_j.\mathrm{erfc}(m_j/\sqrt{2}.\sigma_j)$$

Concernant $I_p^{'}$ le calcule est simple en utilisant le développement en série entière de la fonction $e^{-u^2/2.\sigma_j^2}$

$$e^{-u^2/2.\sigma_j^2} = \sum_{n=0}^{+\infty}(-1)^n \frac{u^{2n}}{n!}.\frac{1}{(2.\sigma_j^2)^n} = \sum_{n=0}^{+\infty}(-1)^n \frac{u^{2n}}{2^n.n!}.\frac{1}{\sigma_j^{2n}},$$

On obtient:

$$I_p^{'} = \sum_{n=0}^{+\infty}(-1)^n.\frac{1}{2^n.n!.\sigma_j^{2n}}.\int_0^{m_j} u^{2n}.u^p.du = \sum_{n=0}^{+\infty}(-1)^n.\frac{1}{2^n.n!.\sigma_j^{2n}}.\frac{(m_j)^{2n+p+1}}{(2n+p+1)}$$

Le calcule de I_p est complètement similaire à celui de $I_p^{'}$.

Remarque :

L'expression des probabilités d'erreur (60-62) sont des développements en série en fonction de la variance σ^2. Ils peuvent aussi être exprimés en fonction du rapport E_b/N_0 en tenant compte du fait que:

$$E_b / N_0 = E_s /(n.(Q+1).N_0) = \frac{1}{3.(Q+1).n.N_0} = \frac{1}{6.(Q+1).n.\sigma^2}$$
$$\sigma^2 = \frac{1}{6.(Q+1).n.(E_b / N_0)} \quad (63)$$

La quantité : $E_s = 1/3$ correspond à l'énergie moyenne par symbole transmis. Le calcule complet de la probabilité d'erreur dans (60-62) est utilisé pour obtenir une limite supérieur de la probabilité d'erreur binaire (Bit Error Probability (BEP)) pour le schéma de codage comprenant le codeur chaotique concaténé au code STBC d'Alamouti.

III.1.1. Probabilité d'erreur binaire

On considère tout d'abord une partie du treillis entre deux instants d'échantillonnage successifs *m* et *m*+1 et on suppose qu'une séquence $\boldsymbol{x}$ a été transmise a partir d'un état donné $\boldsymbol{S}$. nous avons un nombre d'état total égal a N_S. On appelle $E(\mathrm{w},d,l)$ l'événement défini par la condition suivante : une autre séquence $\boldsymbol{x}_d$, qui part du même état $\boldsymbol{S}$, avec une distance de Hamming w entre $\boldsymbol{x}$ et $\boldsymbol{x}_d$ et avec une distance euclidienne entre $\boldsymbol{G}(\boldsymbol{x})$ et $\boldsymbol{G}(\boldsymbol{x}_d)$ égale a *d* ($\boldsymbol{G}$ est la fonction d'encodage) et de longueur *l* est active c'est-à-dire qu'elle possède la plus petite métrique dans l'intervalle (*m*, *m*+1). En outre, on note $e(\mathrm{w},d,l)$ l'événement d'erreur à l'instant *m*, ayant une distance de Hamming w par rapport à $\boldsymbol{x}$, une distance euclidienne entre $\boldsymbol{G}(\boldsymbol{x})$ et $\boldsymbol{G}(\boldsymbol{x}_d)$ égale a *d* et une longueur *l*. La probabilité de l'évènement $E(\mathrm{w},d,l)$ peut être majorée simplement par l'expression :

$$P[E(\mathrm{w},d,l)] \le l.P[e(\mathrm{w},d,l)] = l.C_{\mathrm{w},d,l}.P_e(\boldsymbol{x} \to \boldsymbol{x}_d) \quad (64)$$

Avec $P[e(\mathrm{w},d,l)]$ qui représente la probabilité d'un évènement d'erreur produit par un chemin incorrect $\boldsymbol{x}_d$ avec une distance de Hamming w par rapport à $\boldsymbol{x}$, une distance euclidienne *d* entre $\boldsymbol{G}(\boldsymbol{x})$ et $\boldsymbol{G}(\boldsymbol{x}_d)$, et une longueur *l*. $C_{\mathrm{w},d,l}$ est le nombre moyen de tels chemins. L'inégalité obtenue en (64) repose sur le fait que pour être actif ans l'intervalle (*m*, *m*+1), l'évènement d'erreur $e(\mathrm{w},d,l)$ de longueur *l* doit avoir débuté entre *m*-*l*+1 et *m*.

La probabilité d'erreur par bit conditionnellement à la séquence $\boldsymbol{x}$ et à l'état $\boldsymbol{S}$: $P_b(e \mid \boldsymbol{x}, \boldsymbol{S}) = P_b^{\boldsymbol{x},\boldsymbol{S}}$ peut être définie par la probabilité d'obtenir un bit erroné dans l'intervalle (m, m+1) et est donnée par :

$$P_b^{\boldsymbol{x},\boldsymbol{S}} \leq \sum_{\mathrm{w}=1}^{+\infty} \sum_{d=d\min}^{+\infty} \sum_{l=1}^{+\infty} P_b^{\boldsymbol{x},\boldsymbol{S}}[e \mid E(\mathrm{w},d,l)].P[E(\mathrm{w},d,l)] \quad (65)$$

Avec, pour un codeur de rendement : $k_0 / n_0 = 1/n.(Q+1)$, le résultat suivant

$$P_b^{\boldsymbol{x},\boldsymbol{S}}[e \mid E(\mathrm{w},d,l)] = \frac{\mathrm{w}}{l} \quad (66)$$

En fait, w est le nombre de bits qui diffère entre $\boldsymbol{x}$ et $\boldsymbol{x}_d$ et l est le nombre total de bits transmis dans $\boldsymbol{x}$ et $\boldsymbol{x}_d$ (les bits entrants sont encodés les uns après les autres). En utilisant (66), l'équation (65) peut se réécrire sous la forme:

$$P_b^{\boldsymbol{x},\boldsymbol{S}} \leq \sum_{\mathrm{w}=1}^{+\infty} \sum_{d=d\min}^{+\infty} \sum_{l=1}^{+\infty} \mathrm{w}.C_{\mathrm{w},d,l}.P_e(\boldsymbol{x} \to \boldsymbol{x}_d) \quad (67)$$

Pour obtenir P_e il faut alors moyenner (67) sur l'ensemble des états de départ et sur l'ensemble des séquences de référence transmises. Si on assume des densités de probabilité uniformes pour chacune de ces distributions (la probabilité d'être dans un état donné et la probabilité de transmettre ne séquence donnée) on aboutit à :

$$P_b \leq \frac{1}{2^l.N_s} \sum_{\boldsymbol{x}} \sum_{\boldsymbol{S}} \sum_{\mathrm{w}=1}^{+\infty} \sum_{d=d\min}^{+\infty} \sum_{l=1}^{+\infty} \mathrm{w}.C_{\mathrm{w},d,l}.P_e(\boldsymbol{x} \to \boldsymbol{x}_d) \quad (68)$$

Dans le cas de notre modulation code chaotique en treillis, on peut remplacer les sommations discrètes sur d et l par des intégrales (sommes continues) car, comme nous l'avons déjà vu, le spectre du code est très riche et, pour un évènement d'erreur de longueur w, la densité de probabilité de d peut être approximée par un mélange e lois gaussiennes ou de lois de Rayleigh. On obtient alors le résultat final suivant :

$$P_b \leq \sum_{w=1}^{+\infty} w.P_{e,w}(\boldsymbol{x} \to \boldsymbol{x}') \qquad (69)$$

Avec $P_{e,w}(\boldsymbol{x} \to \boldsymbol{x}')$ qui représente la PEP pour les paires de séquences ayant une distance de Hamming égale à w entre elles. $P_{e,w}(\boldsymbol{x} \to \boldsymbol{x}')$ a déjà été calculée en (60-62).

III.1.2. Résultats de simulation

Nous avons vérifié la formule (69) en utilisant le codeur chaotique de la Figure. 1 (a = 8, n = 2, Q = 3) et le codeur optimisé pour n = 3, Q = 2. Nous comparons la limite supérieure de (69) avec les résultats de simulation sur la Figure. 13. On peut voir que les deux courbes sont très proches les unes des autres dans les deux cas (n = 2 et n = 3). Ceci montre l'exactitude des formules (60, 62) et (69). Le système avec n = 3, Q = 2 à un gain de diversité élevé à forts SNR's; en effet on voit que sa pente est plus importante que dans le cas n = 2, Q = 3. Cela est dû à une meilleure distance libre obtenue grâce à l'étude du spectre des distances.

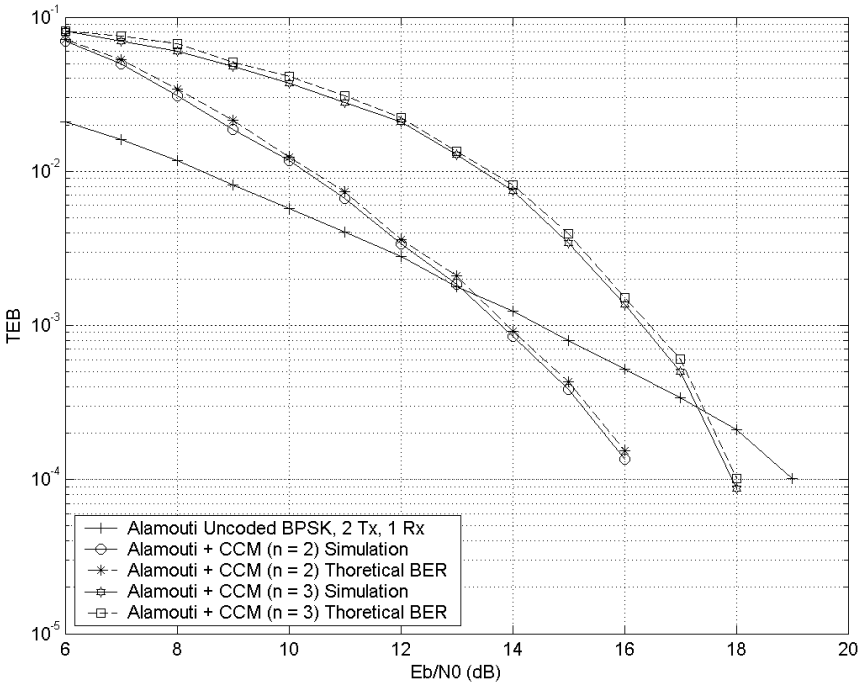


Fig. 13: Performances (TEB) de la concaténation Codeur Chaotique et Code Temps-Espace en bloc, comparaison résultats théoriques et simulations

III.2. Cas d'un canal sélectif en temps

Les équations correspondant au modèle équivalent en bande de base à la sorti de l'antenne de réception sont semblables à (34), cette fois nous considérons que les paramètres du canal peuvent varier d'un échantillon à l'autre, ce qui correspond à des canaux hautement sélectifs en temps. On obtient le système d'équations :

$$\begin{aligned} y(2n) &= h_{11}(2n).x(2n) + h_{21}(2n).x(2n+1) + n(2n) \\ y(2n+1) &= h_{21}(2n+1).x^*(2n) - h_{11}(2n+1).x^*(2n+1) + n(2n+1) \end{aligned} \quad (70)$$

On peut écrire ces équations sous la forme matricielle équivalente:

$$\begin{pmatrix} y(2n) \\ y^*(2n+1) \end{pmatrix} = \begin{pmatrix} h_{11}(2n) & h_{21}(2n) \\ h_{21}^*(2n+1) & -h_{11}^*(2n+1) \end{pmatrix} \begin{pmatrix} x(2n) \\ x(2n+1) \end{pmatrix} + \begin{pmatrix} n(2n) \\ n^*(2n+1) \end{pmatrix}$$

$$\boldsymbol{Y}(n) = \boldsymbol{H}(n).\boldsymbol{X}(n) + \boldsymbol{N}(n) \quad (71)$$

III.2.1. Utilisation de l'égalisation MRC classique

L'utilisation de l'égalisation MRC classique consiste à multiplier le vecteur $\boldsymbol{Y}$ reçu par la matrice $\boldsymbol{H}^H(n)$, ce qui conduit à:

$$\boldsymbol{H}^H(n).\boldsymbol{Y}(n) = \boldsymbol{H}^H(n).\boldsymbol{H}(n).\boldsymbol{X}(n) + \boldsymbol{H}^H(n).\boldsymbol{N}(n) \quad (72)$$

Nous avons:

$$\boldsymbol{H}^H(n).\boldsymbol{H}(n)=\begin{pmatrix} h_{11}^*(2n) & h_{21}(2n+1) \\ h_{21}^*(2n) & -h_{11}(2n+1) \end{pmatrix}.\begin{pmatrix} h_{11}(2n) & h_{21}(2n) \\ h_{21}^*(2n+1) & -h_{11}^*(2n+1) \end{pmatrix}$$
$$=\begin{pmatrix} |h_{11}(2n)|^2+|h_{21}(2n+1)|^2 & h_{11}^*(2n).h_{21}(2n)-h_{11}^*(2n+1).h_{21}(2n+1) \\ h_{21}^*(2n).h_{11}(2n)-h_{11}(2n+1).h_{21}^*(2n+1) & |h_{21}(2n)|^2+|h_{11}(2n+1)|^2 \end{pmatrix}$$

De même que dans [48], nous utilisons les notations suivantes :

$$\zeta_1(n)=|h_{11}(2n)|^2+|h_{21}(2n+1)|^2$$
$$\zeta_2(n)=|h_{21}(2n)|^2+|h_{11}(2n+1)|^2$$
$$\psi(n)=h_{11}^*(2n).h_{21}(2n)-h_{11}^*(2n+1).h_{21}(2n+1)$$

La présence d'éléments non nuls hors diagonale dans la matrice $\boldsymbol{H}^H(n).\boldsymbol{H}(n)$ provoque de l'interférence entre symboles. Par conséquent, les variables décisionnelles : $Z(2.n)$ et $Z(2.n+1)$ doivent inclure la présence de ces termes d'interférence et on aboutit à :

$$\begin{cases} Z(2n)=\left(|h_{11}(2n)|^2+|h_{21}(2n+1)|^2\right).x(2n)+\psi(n).x(2n+1)+h_{11}^*(2n).n(2n)+h_{21}(2n+1).n(2n+1) \\ Z(2n+1)=\left(|h_{21}(2n)|^2+|h_{11}(2n+1)|^2\right).x(2n+1)+\psi^*(n).x(2n)+h_{21}^*(2n).n(2n)-h_{11}(2n+1).n(2n+1) \end{cases}$$

Si on suppose connu le coefficient $h_{11}(2n)$ i.e. pour $h_{11}(2n)$ fixé, le coefficient du canal $h_{11}(2n+1)$ est une variable aléatoire gaussienne de moyenne : $R_h(1).h_{11}(2n)$ et de variance : $1-|R_h(1)|^2$. En effet on a, en utilisant le modèle de Jake classique :

$$E[h_{11}^*(2n).h_{11}(2n+1)]=\sigma_h^2.R_h(1) \quad (73)$$

Avec : $\sigma_h^2=E[h_{11}^*(2n).h_{11}(2n)]$ (74) et $R_h(l)=J_0(2.\pi.f_d.T_s.l)$ où $J_0(.)$ est la fonction de Bessel de première espèce d'ordre zéro et f_d est le décalage Doppler maximal. En remplaçant σ_h^2 extrait de (74) dans l'expression (73), on obtient:

$$E[h_{11}^*(2n).h_{11}(2n+1)-R_h(1).h_{11}^*(2n).h_{11}(2n)]=0 \quad (75)$$

Si on suppose connu le gain de canal : $h_{11}(2n)$, la variable aléatoire $h_{11}(2n+1)$ est une variable aléatoire gaussienne complexe puisqu'elle est obtenue comme combinaison linéaire de variables gaussiennes. Pour calculer sa moyenne, on peut réécrire (75) en simplifiant les termes fixes $h_{11}^{*}(2n)$:

$$E[h_{11}(2n+1) - R_h(1).h_{11}(2n)]_{|h_{11}(2n)} = 0$$

Ce qui conduit à :

$$E[h_{11}(2n+1)]_{|h_{11}(2n)} = R_h(1).h_{11}(2n) \quad (76)$$

Pour la variance, on a le calcul suivant (on suppose: $\sigma_h^2 = 1/2$):

$$\begin{aligned}
\sigma_h^2 &= E[(h_{11}(2n+1) - R_h(1).h_{11}(2n)).(h_{11}(2n+1) - R_h(1).h_{11}(2n))^{*}] \\
&= E[(h_{11}(2n+1).h_{11}^{*}(2n+1)] - R_h(1).E(h_{11}(2n+1).h_{11}^{*}(2n)) \\
&\quad - R_h^{*}(1).E(h_{11}(2n).h_{11}^{*}(2n+1)) + |R_h(1)|^2 \qquad (77) \\
&= 1 - 2.\sigma_h^2.R_h(1).R_h^{*}(1) - 2.\sigma_h^2.R_h(1).R_h^{*}(1) + |R_h(1)|^2 \\
&= 1 - |R_h(1)|^2
\end{aligned}$$

Donc, finalement on a bien obtenu :

$$h_{11}(2n+1)_{|h_{11}(2n)} = \mathcal{N}(R_h(1).h_{11}(2n), 1 - |R_h(1)|^2)$$

De même, on pourrait démontrer que:

$$h_{21}(2n)_{|h_{21}(2n+1)} = \mathcal{N}(R_h(1).h_{21}(2n+1), 1 - |R_h(1)|^2)$$

Utilisant ces deux propriétés fondamentales, il est facile de démontrer que le terme d'interférences $\psi(n)$, conditionnellement aux paramètres de canal $h_{11}(2n)$ et $h_{21}(2n+1)$, est une variable aléatoire gaussienne complexe, de moyenne nulle et de variance: $(|h_{11}(2n)|^2 + |h_{21}(2n+1)|^2).(1 - |R_h(1)|^2)$, soit :

$$\psi(n)_{|h_{11}(2n), h_{21}(2n+1)} = \mathcal{N}(0, (|h_{11}(2n)|^2 + |h_{21}(2n+1)|^2).(1 - |R_h(1)|^2)) \quad (78)$$

En supposant connus les paramètres du canal : $h_{11}(2m), h_{11}(2m+1),..., h_{21}(2(L+m-1)+1)$, il est alors possible d'exprimer la PEP sous la forme :

$$\begin{aligned} P_e(\boldsymbol{x} \to \boldsymbol{x}')\big|_{\boldsymbol{H}(n),n=m:L+m-1}, \boldsymbol{x}) = \\ \text{Proba}[\sum_{n=m}^{L+m-1} \left|Z(2n) - \zeta_1(n).x(2n)\right|^2 + \left|Z(2n+1) - \zeta_2(n).x(2n+1)\right|^2 \\ < \sum_{n=m}^{L+m-1} \left|Z(2n) - \zeta_1(n).x'(2n)\right|^2 + \left|Z(2n+1) - \zeta_2(n).x'(2n+1)\right|^2] \end{aligned} \quad (79)$$

On remplace ensuite les variables décisionnelles par leurs expressions :

$$\begin{aligned} Z(2n) &= \zeta_1(n).x(2n) + N_1(n) \\ Z(2n+1) &= \zeta_2(n).x(2n+1) + N_2(n) \end{aligned}$$

Avec :

$$\begin{aligned} N_1(n) &= \psi(n).x(2n+1) + h_{11}^*(2n).n(2n) + h_{21}(2n+1).n(2n+1) \\ N_2(n) &= \psi^*(n).x(2n) + h_{21}^*(2n).n(2n) - h_{11}(2n+1).n(2n+1) \end{aligned}$$

On obtient la PEP sous une nouvelle forme:

$$\begin{aligned} P_e(\boldsymbol{x} \to \boldsymbol{x}')\big|_{\boldsymbol{H}(n),n=m:L+m-1}, \boldsymbol{x}) = \text{Proba}\Bigg(\sum_{n=m}^{L+m-1} 2.\text{Re}(\zeta_1(n).[N_1^*(n).(x(2n) - x'(2n))] + ... \\ + 2.\text{Re}(\zeta_2(n).[N_2^*(n).(x(2n+1) - x'(2n+1))] \\ > \sum_{n=m}^{L+m-1} \left|\zeta_1(n)\right|^2 \left[\left|x'(2n) - x(2n)\right|^2\right] + \left|\zeta_2(n)\right|^2 \left[\left|x'(2n+1) - x(2n+1)\right|^2\right] \Bigg) \end{aligned} \quad (80)$$

La variable aléatoire:

$$X = \sum_{n=m}^{L+m-1} 2.\text{Re}[\zeta_1(n).[N_1^*(n).(x(2n) - x'(2n))] + \zeta_2(n).[N_2^*(n).(x(2n+1) - x'(2n+1))]]$$

peut être caractérisée de la façon suivante. *X*, en supposant les paramètres du canal fixés, est une variable aléatoire gaussienne de moyenne nulle et de variance :

$$\sigma^2_{X|h_{11}(2n),h_{21}(2n+1)} = \sum_{n=m}^{m+L-1} 4.\xi_1^2(n).(x(2n)-x'(2n))^2.[\xi_1(n).(1-|R_h(1)|^2).x^2(2n+1)+\sigma^2.\xi_1(n)]+...$$
$$+4.\xi_2^2(n).(x(2n+1)-x'(2n+1))^2.[\xi_1(n).(1-|R_h(1)|^2).x^2(2n)+\sigma^2.\xi_2(n)] \quad (81)$$

On peut alors préciser la valeur de la PEP dans l'équation (80) :

$$P_e(\boldsymbol{x}\to\boldsymbol{x}'|_{\boldsymbol{H}(n),n=m:L+m-1},\boldsymbol{x}) = \frac{1}{2}.erfc\left(\frac{\sum_{n=m}^{m+L-1} |\ \xi_1(n)\ |^2.C_{2n}^2 + |\ \xi_2(n)\ |^2.C_{2n+1}^2}{2.\sqrt{2}.\left[\sum_{n=m}^{m+L-1} \xi_1^2(n).C_{2n}^2.[\xi_1(n).(1-|R_h(1)|^2).x^2(2n+1)+\sigma^2.\xi_1(n)]+\xi_2^2(n).C_{2n+1}^2.[\xi_1(n).(1-|R_h(1)|^2).x^2(2n)+\sigma^2.\xi_2(n)]\right]^{1/2}}\right) \quad (82)$$

avec: $C_{2n}^2 = (x(2n)-x'(2n))^2$ et $C_{2n+1}^2 = (x(2n+1)-x'(2n+1))^2$

Pour aller plus loin dans le calcul on utilise les approximations suivantes (malgré l'hypothèse de départ d'une forte sélectivité temporelle du canal):

$$|h_{21}(2n+1)|^2 + |h_{11}(2n)|^2 \approx |h_{11}(2n+1)|^2 + |h_{21}(2n)|^2$$
$$\approx |h_{11}(2m+1)|^2 + |h_{21}(2m)|^2 \qquad \forall n \in [m, m+L-1]$$

C'est à dire: $\zeta_1(n) = \zeta_2(n) \approx \zeta_1(m), \forall n \in [m, m+L-1]$, on obtient :

$$P_e(\boldsymbol{x}\to\boldsymbol{x}'|_{h_{11}(2m+1),h_{21}(2m)},\boldsymbol{x})$$
$$\approx \frac{1}{2}.erfc\left(\frac{|\ \xi_1(m)\ |^{1/2}\sum_{n=m}^{m+L-1}(C_{2n}^2+C_{2n+1}^2)}{2.\sqrt{2}.\left[\sum_{n=m}^{m+L-1} C_{2n}^2.[(1-|R_h(1)|^2).x^2(2n+1)+\sigma^2]+C_{2n+1}^2.[(1-|R_h(1)|^2).x^2(2n)+\sigma^2]\right]^{1/2}}\right) \quad (83)$$

Pour compléter le calcule de PEP, nous avons à moyenner (73) sur la distribution de $\xi_1(m)$. Il n'est pas difficile de montrer que la variable aléatoire $\xi_1(m)$ suit une distribution de type $\chi 2$ (Chi 2) de degré 4, ce qui s'écrit:

$$p_{\xi_1(m)} = \frac{1}{4.\sigma_h^4.\Gamma(2)}.x.\exp(-\frac{x}{2.\sigma_h^2}).$$

$$P_e(\boldsymbol{x}\to\boldsymbol{x}'\big|_{h_{11}(2m+1),h_{21}(2m)},\boldsymbol{x})$$
$$=\frac{1}{8.\sigma_h^4}\int_0^{+\infty} erfc\left(\frac{x^{1/2}\sum_{n=m}^{m+L-1}(C_{2n}^2+C_{2n+1}^2)}{2.\sqrt{2}.\left[\sum_{n=m}^{m+L-1}C_{2n}^2.[(1-\left|R_h(1)\right|^2).x^2(2n+1)+\sigma^2]+C_{2n+1}^2.[(1-\left|R_h(1)\right|^2).x^2(2n)+\sigma^2]\right]^{1/2}}\right).x.e^{-x/2.\sigma_h^2}.dx$$

On pose alors:

$$\sigma'=\left[\sum_{n=m}^{m+L-1}C_{2n}^2.[(1-\left|R_h(1)\right|^2).x^2(2n+1)+\sigma^2]+C_{2n+1}^2.[(1-\left|R_h(1)\right|^2).x^2(2n)+\sigma^2]\right]^{1/2}$$

et $C=\sum_{n=m}^{m+L-1}(C_{2n}^2+C_{2n+1}^2)$, on a :

$$P_e(\boldsymbol{x}\to\boldsymbol{x}'\big|_{h_{11}(2m+1),h_{21}(2m)},\boldsymbol{x})=\frac{1}{8.\sigma_h^4}\int_0^{+\infty} erfc\left(\frac{x^{1/2}C}{2.\sqrt{2}.\sigma'}\right).x.e^{-x/2.\sigma_h^2}.dx \qquad (84)$$

Utilisant (54), on obtient:

$$P_e(\boldsymbol{x}\to\boldsymbol{x}'\big|\boldsymbol{x})=\frac{1}{2}-\frac{C.\sigma'^2.\sigma_h}{(C^2.\sigma_h^2+4.\sigma'^2)^{3/2}}-\frac{C.\sigma_h}{2.(C^2.\sigma_h^2+4.\sigma'^2)^{1/2}} \qquad (85)$$

Pour les valeurs de SNR's élevés, on a:

$$\sigma'\approx\left[\sum_{n=m}^{m+L-1}C_{2n}^2.[(1-\left|R_h(1)\right|^2).x^2(2n+1)]+C_{2n+1}^2.[(1-\left|R_h(1)\right|^2).x^2(2n)]\right]^{1/2}$$

qui est indépendant de σ^2 et l'expression (85) montre alors que nous obtenons un plancher d'erreur irréductible à forts SNR's. Dans ce cas l'égaliseur MRC perd complètement le gain de diversité celui-ci devient nul.

Par conséquent, la sélectivité de temps du canal fait que l'égaliseur (MRC) classique linéaire fait perdre toute la diversité du schéma d'Alamouti (diversité d'ordre 2 classiquement). Pour rétablir certaine diversité il faudra donc supprimer les termes d'interférences dans les expressions $Z(2n), Z(2n+1)$.

Pour obtenir le taux d'erreur binaire final, nous avons à moyenner (85) sur la distribution du spectre des distances du codeur chaotique. En utilisant les approximations à base de mélange de lois gaussiennes ou de Rayleigh, nous sommes capables d'obtenir une limite inférieure en terme de E_b / N_0.

Les résultats de simulation mettent en évidence le phénomène de plancher de taux d'erreur binaire pour des rapports signal à bruit élevés comme le montre la figure 14. Selon la sélectivité temporelle du canal ou la valeur de la fréquence doppler, les résultats de simulation coïncident bien avec les limites inférieures théoriques sauf pour le cas $f_d.T_s = 0.01$où notre hypothèse qui consiste à considérer les paramètres de canal constant sur toute la longueur des séquences divergentes est trop optimiste.

La différence entre les résultats de simulation et la limite inférieure dans ce cas est de l'ordre de 2 dB, tandis qu'elle reste inférieure à 1 dB pour les fréquences doppler peu élevées.

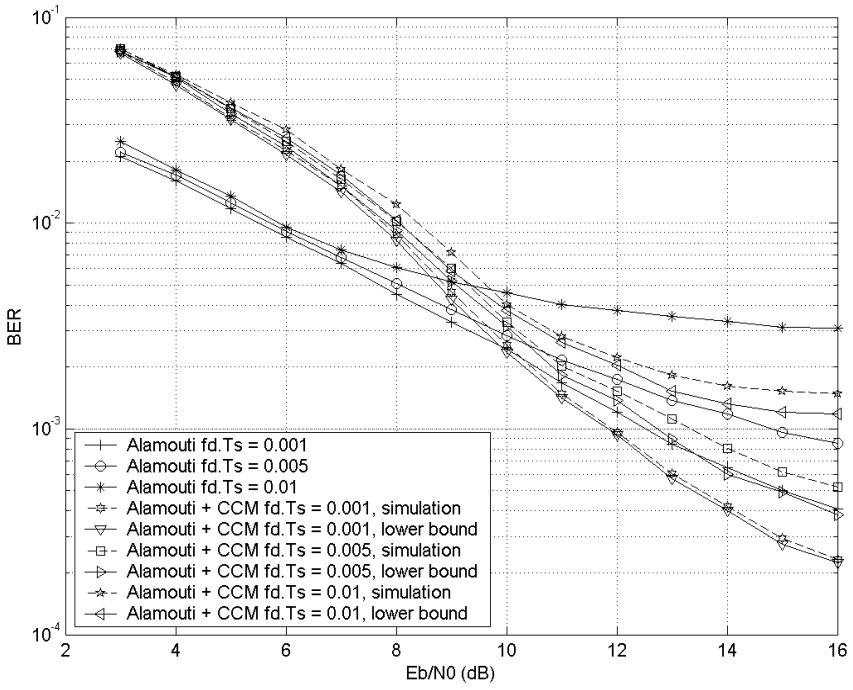


Fig. 14: résultats de simulation de MRC et limites inférieures théoriques pour des canaux sélectifs en temps

III.2.2. Egalisation Zero-Forcing (ZF)

Pour réduire le plancher d'erreur provenant de l'interférence entre symboles causée par les éléments hors diagonale : $\psi(n)$, il faut construire des variables décisionnelles qui atténuent

l'influence de $\psi(n)$. Pour supprimer l'interférence entre symbole, due a la variation temporelle dues coefficients du canal, une idée naturelle consiste à diagonaliser $\boldsymbol{H}(n)$, c'est-à-dire à forcer l'interférence entre symboles à devenir nulle.

La matrice qui diagonalise $\boldsymbol{H}(n)$ est donnée par :

$$\boldsymbol{Q}(n)=\begin{pmatrix} h_{11}^{*}(2n+1) & h_{21}(2n) \\ h_{21}^{*}(2n+1) & -h_{11}(2n) \end{pmatrix} \quad (86)$$

En multipliant les deux cotés de (71) par $\boldsymbol{Q}(n)$, on obtient :

$$\boldsymbol{Q}(n).\boldsymbol{Y}=\boldsymbol{Q}(n).\boldsymbol{H}(n).X+\boldsymbol{Q}(n).\boldsymbol{N} \quad (87)$$

La matrice $\boldsymbol{Q}(n).\boldsymbol{H}(n)$ est égale a :

$$\boldsymbol{Q}(n).\boldsymbol{H}(n)=\begin{pmatrix} h_{11}^{*}(2n+1).h_{11}(2n)+h_{21}(2n).h_{21}^{*}(2n+1) & 0 \\ 0 & h_{21}(2n).h_{21}^{*}(2n+1)+h_{11}(2n).h_{11}^{*}(2n+1) \end{pmatrix}$$

On pose : $\lambda(n)=h_{11}^{*}(2n+1).h_{11}(2n)+h_{21}(2n).h_{21}^{*}(2n+1)$.

Utilisant le modèle équivalent (87), on obtient, après diagonalisation, le modèle mathématique suivant :

$$\begin{aligned} Z(2n) &= \lambda(n).x(2n)+h_{11}^{*}(2n+1).n(2n)+h_{21}(2n).n^{*}(2n+1) \\ Z(2n+1) &= \lambda(n).x(2n+1)+h_{21}^{*}(2n+1).n(2n)-h_{11}(2n).n^{*}(2n+1) \end{aligned} \quad (88)$$

En supposant les paramètres du canal : $h_{11}(2m), h_{11}(2m+1),..., h_{21}(2(L+m-1)+1)$ connus, on peut calculer la PEP :

$$P_e(\boldsymbol{x}\rightarrow\boldsymbol{x}'\big|_{\boldsymbol{H}(n),n=m:L+m-1},\boldsymbol{x})=\text{Proba}\left(\sum_{n=m}^{L+m-1}\left|Z(2n)-\lambda(n).x(2n)\right|^2+\left|Z(2n+1)-\lambda(n).x(2n+1)\right|^2\right)$$
$$<\sum_{n=m}^{L+m-1}\left|Z(2n)-\lambda(n).x'(2n)\right|^2+\left|Z(2n+1)-\lambda(n).x'(2n+1)\right|^2$$

On obtient finalement :

$$P_e(\boldsymbol{x}\to\boldsymbol{x}'\big|_{\boldsymbol{H}(n),n=m:L+m-1},\boldsymbol{x}) = \text{Proba}\Bigg(\sum_{n=m}^{L+m-1} 2.\text{Re}(|\lambda(n)|^2.x^*(2n).(x'(2n)-x(2n))+\ldots$$
$$+\lambda(n).N_1^*(n).(x'(2n)-x(2n))+|\lambda(n)|^2.x^*(2n+1).(x'(2n+1)-x(2n+1))+\ldots$$
$$+\lambda(n).N_2^*(n).(x'(2n+1)-x(2n+1))$$
$$< \sum_{n=m}^{L+m-1} |\lambda(n)|^2\Big[\ |x'(2n)|^2-|x(2n)|^2+|x'(2n+1)|^2-|x(2n+1)|^2\Big]\Bigg) \quad (89)$$

avec :

$$Z(2n)=\lambda(n).x(2n)+N_1(n)$$
$$Z(2n+1)=\lambda(n).x(2n+1)+N_2(n)$$

et :

$$N_1(n)=h_{11}^*(2n+1).n(2n)+h_{21}(2n).n^*(2n+1)$$
$$N_2(n)=h_{21}^*(2n+1).n(2n)-h_{11}(2n).n^*(2n+1)$$

En reportant dans (89), on obtient :

$$P_e(\boldsymbol{x}\to\boldsymbol{x}'\big|_{\boldsymbol{H}(n),n=m:L+m-1},\boldsymbol{x}) = \text{Proba}\Bigg(\sum_{n=m}^{L+m-1} 2.\text{Re}(\lambda(n).[N_1^*(n).(x(2n)-x'(2n))+\ldots$$
$$+N_2^*(n).(x(2n+1)-x'(2n+1))$$
$$> \sum_{n=m}^{L+m-1} |\lambda(n)|^2\Big[\ |x'(2n)-x(2n)|^2+|x'(2n+1)-x(2n+1)|^2\Big]\ \Bigg) \quad (90)$$

Pour développer la calcule de la probabilité d'erreur, il faut alors caractériser la variable aléatoire:

$$X=\sum_{n=m}^{L+m-1} 2.\text{Re}(\lambda(n).[N_1^*(n).(x(2n)-x'(2n))+N_2^*(n).(x(2n+1)-x'(2n+1))]$$

En supposant les paramètres du canal connus, X est une variable aléatoire gaussienne de moyenne nulle et de variance σ_X^2 .

$$X = \sum_{n=m}^{L+m-1} 2.\mathrm{Re}\Big[(\lambda(n).[h_{11}(2n+1).n^*(2n) + h_{21}^*(2n).n(2n+1)].(x(2n) - x'(2n)) + \ldots \\ + \lambda(n).[h_{21}(2n+1).n^*(2n) - h_{11}^*(2n).n(2n+1)].(x(2n+1) - x'(2n+1)) \Big] \quad (91)$$

On décompose chaque $\lambda(n), h_{11}(2n), n(2n), h_{21}(2n), n(2n+1)...$ avec ces parties réels et imaginaires.

$$\lambda(n) = \lambda^R(n) + j.\lambda^I(n)$$
$$h_{11}(2n) = h_{11}^R(2n) + j.h_{11}^I(2n)$$
$$n(2n) = n^R(2n) + j.n^I(2n)$$
$$h_{21}(2n) = h_{21}^R(2n) + j.h_{21}^I(2n)$$
$$n(2n+1) = n^R(2n+1) + j.n^I(2n+1)$$

En remplaçant ces expressions dans (91), on obtient:

$$X = \sum_{n=m}^{L+m-1} 2.(\lambda^R(n) + j.\lambda^I(n)).\Big[(h_{11}^R(2n+1) + j.h_{11}^I(2n+1)).(n^R(2n) - j.n^I(2n)) + \ldots \\ + (h_{21}^R(2n) - j.h_{21}^I(2n)).(n^R(2n+1) + j.n^I(2n+1)) \Big].(x(2n) - x'(2n)) \\ + 2.(\lambda^R(n) + j.\lambda^I(n)).\Big[(h_{21}^R(2n+1) + j.h_{11}^I(2n+1)).(n^R(2n+1) + j.n^I(2n+1)) + \ldots \\ - (h_{11}^R(2n) - j.h_{11}^I(2n)).(n^R(2n+1) + j.n^I(2n+1)) \Big].(x(2n+1) - x'(2n+1)) \quad (92)$$

$$X = \sum_{n=m}^{L+m-1} 2.(x(2n) - x'(2n)).[\lambda^R(n).[h_{11}^R(2n+1).n^R(2n) + h_{11}^I(2n+1).n^I(2n) + h_{21}^R(2n).n^R(2n) + h_{21}^I(2n).n^I(2n+1)] + \ldots \\ -\lambda^I(n).[h_{11}^I(2n+1).n^R(2n) - h_{11}^R(2n+1).n^I(2n) + h_{21}^R(2n).n^I(2n+1) - h_{21}^I(2n).n^R(2n+1)]] + \ldots \\ +2.(x(2n+1) - x'(2n+1)).[\lambda^R(n).[h_{21}^R(2n+1).n^R(2n) + h_{21}^I(2n+1).n^I(2n) - h_{21}^R(2n).n^R(2n+1) - h_{11}^I(2n).n^I(2n+1)] + \ldots \\ -\lambda^I(n).[h_{21}^I(2n+1).n^R(2n) - h2_{11}^R(2n+1).n^I(2n) + h_{11}^I(2n).n^R(2n+1) - h_{11}^R(2n).n^I(2n+1)]]$$

A partir de la dernière expression, on obtient finalement:

$$\sigma_X^2 = \sum_{n=m}^{L+m-1} 4.(x(2n) - x'(2n))^2.\sigma^2.[\ |h_{11}(2n+1)|^2 + |h_{21}(2n)|^2].|\lambda(n)\ |^2 + \ldots \\ + 4.(x(2n+1) - x'(2n+1))^2.\sigma^2.[\ |h_{21}(2n+1)|^2 + |h_{11}(2n)|^2].|\lambda(n)\ |^2 \quad (93)$$

Pour obtenir (93), on utilisant le fait que $n^R(2n), n^I(2n), n^R(2n+1), n^I(2n+1)...$ sont des bruits blancs, c'est-à-dire :

$$E[n^R(k).n^I(l)] = 0, \forall k \neq l \text{ et } E[n^R(k).n^R(l)] = E[n^I(k).n^I(l)] = 0, \forall k \neq l$$

$$E[(n^R(k))^2] = E[(n^I(k))^2] = \sigma^2$$

Ainsi, on obtient :

$$\sigma_X^2 = 4.\sigma^2 \sum_{n=m}^{L+m-1} |\lambda(n)|^2 \left\{ [|h_{11}(2n+1)|^2 + |h_{21}(2n)|^2].(x'(2n) - x(2n))^2 + ... \right.$$
$$\left. ... + [|h_{21}(2n+1)|^2 + |h_{11}(2n)|^2].(x'(2n+1) - x(2n+1))^2 \right\}$$

En utilisant la distribution gaussienne de *X*, on calcule alors la PEP :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | \lambda(m), h_{11}(2m+1), h_{21}(2m), ..., h_{11}(2(m+L-1)+1), h_{21}(2(m+L-1)), \boldsymbol{x}) :$$

On obtient les résultats suivants.

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | \lambda(m), h_{11}(2m+1), h_{21}(2m), ..., h_{11}(2(m+L-1)+1), h_{21}(2(m+L-1), \boldsymbol{x})$$
$$= \frac{1}{2}.erfc\left(\frac{\sum_{n=m}^{m+L-1} |\lambda(n)|^2 [C_{2n}^2 + C_{2n+1}^2]}{\sqrt{2}.\sigma.\left(\sum_{n=m}^{m+L-1} |\lambda(n)|^2 .[\left(|h_{11}(2n+1)|^2 + |h_{21}(2n)|^2\right).C_{2n}^2 + \left(|h_{21}(2n+1)|^2 + |h_{11}(2n)|^2\right).C_{2n+1}^2]\right)^{1/2}}\right) \quad (94)$$

Avec : $C_{2n}^2 = (x(2n) - x'(2n))^2$ et $C_{2n+1}^2 = (x(2n+1) - x'(2n+1))^2$

En utilisant la forme de Graig de la fonction erfc(.), on peut également écrire:

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | \lambda(m), h_{11}(2m+1), h_{21}(2m), h_{11}(2(m+L-1)+1), h_{21}(2(m+L-1), \boldsymbol{x})$$
$$= \frac{1}{\pi}.\int_0^{\pi/2} \exp\left(\frac{-\left(\sum_{n=m}^{m+L-1} |\lambda(n)|^2 [C_{2n}^2 + C_{2n+1}^2]\right)^2 .d\theta}{2.\sigma^2.\left(\sum_{n=m}^{m+L-1} |\lambda(n)|^2 .[\left(|h_{11}(2n+1)|^2 + |h_{21}(2n)|^2\right).C_{2n}^2 + \left(|h_{21}(2n+1)|^2 + |h_{11}(2n)|^2\right).C_{2n+1}^2]\right).\sin^2\theta}\right)$$

Pour aller plus loin dans le calcul précédent, on utilise l'hypothèse : $\lambda(n) = cste, \forall n \in [m, L+m-1]$, c'est-à-dire, on considère d'abord le cas d'effets doppler modérés, on obtient :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | \lambda(m), h_{11}(2m+1), h_{21}(2m), \boldsymbol{x})$$

$$= \frac{1}{\pi}.\int_0^{\pi/2} \exp\left(\frac{-|\lambda(m)|^4.\left(\sum_{n=m}^{m+L-1}[C_{2n}^2 + C_{2n+1}^2]\right)^2.d\theta}{2.\sigma^2.|\lambda(m)|^2\left(\sum_{n=m}^{m+L-1}[\left(|h_{11}(2n+1)|^2 + |h_{21}(2n)|^2\right).C_{2n}^2 + \left(|h_{21}(2n+1)|^2 + |h_{11}(2n)|^2\right).C_{2n+1}^2]\right).\sin^2\theta}\right)$$

$$= \frac{1}{\pi}.\int_0^{\pi/2} \exp\left(\frac{-|\lambda(m)|^2.\left(\sum_{n=m}^{m+L-1}[C_{2n}^2 + C_{2n+1}^2]\right)^2.d\theta}{2.\sigma^2.\left(\sum_{n=m}^{m+L-1}[\left(|h_{11}(2n+1)|^2 + |h_{21}(2n)|^2\right).C_{2n}^2 + \left(|h_{21}(2n+1)|^2 + |h_{11}(2n)|^2\right).C_{2n+1}^2]\right).\sin^2\theta}\right)$$

En outre (si on considère le cas d'effet Doppler modérés), on suppose:

$$|h_{21}(2n+1)|^2 + |h_{11}(2n)|^2 \approx |h_{11}(2n+1)|^2 + |h_{21}(2n)|^2$$
$$\approx |h_{11}(2m+1)|^2 + |h_{21}(2m)|^2 \qquad \forall n \in [m, m+L-1]$$

On utilise de plus le lemme suivant :

Lemme: si x est une variable aléatoire gaussien de moyen m_x et de variance σ_x^2, on a :

$$E[\exp(w.x^2)] = \frac{\exp\left(\frac{w.m_x^2}{1-2.w.\sigma_x^2}\right)}{\sqrt{1-2.w.\sigma_x^2}} \qquad (95)$$

On obtient alors l'expression suivante de la PEP :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | \lambda(m), h_{11}(2m+1), h_{21}(2m), \boldsymbol{x})$$
$$\approx \frac{1}{\pi}.\int_0^{\pi/2} \exp\left(\frac{-|\lambda(m)|^2.\left(\sum_{n=m}^{m+L-1}[C_{2n}^2 + C_{2n+1}^2]\right).d\theta}{2.\sigma^2.\left(|h_{11}(2m+1)|^2 + |h_{21}(2m)|^2\right).\sin^2\theta}\right) \qquad (96)$$

Pour moyenner (96) sur la distribution de $\lambda(m)$, on peut remarquer que conditionnellement aux paramètres de canal : $h_{11}(2m+1), h_{21}(2m)$, $\lambda(m)$ est une variable aléatoire gaussienne caractérisée par :

$$\lambda(m) = \mathcal{N}(R_h(1).[|h_{11}(2m+1)|^2 + |h_{21}(2m)|^2], (1-|R_h(1)|^2)[|h_{11}(2m+1)|^2 + |h_{21}(2m)|^2]^2)$$

Ainsi $\mathrm{Re}(\lambda(m))$ est indépendant de $\mathrm{Im}(\lambda(m))$ et on peut moyenner la probabilité d'erreur conditionnel séparément comme suit:

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | h_{11}(2m+1), h_{21}(2m), \boldsymbol{x})$$
$$\approx \frac{1}{\pi}.\int_0^{\pi/2} E[\exp(\frac{-|\mathrm{Re}(\lambda(m))|^2.(\sum_{n=m}^{m+L-1}[C_{2n}^2 + C_{2n+1}^2])}{2.\sigma^2.(|h_{11}(2m+1)|^2 + |h_{21}(2m)|^2).\sin^2\theta})].E[\exp(\frac{-|\mathrm{Im}(\lambda(m))|^2.(\sum_{n=m}^{m+L-1}[C_{2n}^2 + C_{2n+1}^2]).d\theta}{2.\sigma^2.(|h_{11}(2m+1)|^2 + |h_{21}(2m)|^2).\sin^2\theta})]$$

Appliquant le lemme (95) ci-dessus, on aboutit à:

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | h_{11}(2m+1), h_{21}(2m), \boldsymbol{x})$$
$$\approx \frac{1}{\pi}.\int_0^{\pi/2} \frac{\exp(-\frac{|R_h(1)|^2.\gamma.(|h_{11}(2m+1)|^2 + |h_{21}(2m)|^2)}{\sin^2\theta + (1-|R_h(1)|^2).\gamma})}{1+(1-|R_h(1)|^2).\gamma/\sin^2\theta} d\theta \quad (97)$$

avec: $\gamma = \sum_{n=m}^{m+L-1} \frac{[C_{2n}^2 + C_{2n+1}^2]}{2.\sigma^2}$ (98)

On sait que : $|h_{11}(2m+1)|^2 + |h_{21}(2m)|^2$ suit une distribution de type $\chi 2$ (Chi 2) de degré 4, ce qui s'écrit:

$$p_B(x) = \frac{1}{\sigma_h^d.2^{d/2}.\Gamma(d/2)}.x^{d/2-1}.\exp(-\frac{x}{2.\sigma_h^2}) \ .$$

Moyennant la probabilité d'erreur sur sa distribution, on obtient :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}' | \boldsymbol{x}) \approx \frac{1}{\pi}.\int_0^{\pi/2} \frac{1}{1+(1-|R_h(1)|^2).\gamma/\sin^2\theta}.\int_0^{+\infty} \exp(-\frac{|R_h(1)|^2.\gamma.x)}{\sin^2\theta + (1-|R_h(1)|^2).\gamma}).p_B(x).dx.d\theta$$

C'est-à-dire :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}'|\boldsymbol{x}) \approx \frac{1}{\pi}.\int_0^{\pi/2} \frac{1}{1+(1-|R_h(1)|^2).\gamma/\sin^2\theta}.\int_0^{+\infty} x.\exp(-\frac{|R_h(1)|^2.\gamma.x)}{\sin^2\theta+(1-|R_h(1)|^2).\gamma}-x).dx.d\theta \quad (98)$$

En Appliquant l'identité classique: $\int_0^{+\infty} x^n.\exp(-ax).dx = \frac{\Gamma(n+1)}{a^{n+1}} = \frac{n!}{a^{n+1}}$, et après quelques manipulations ,on obtient :

$$P_e(\boldsymbol{x} \to \boldsymbol{x}'|\boldsymbol{x}) \approx \frac{1}{\pi}.\int_0^{\pi/2} \frac{\sin^2\theta.(\sin^2\theta+(1-|R_h(1)|^2).\gamma)}{(\sin^2\theta+\gamma)^2}.d\theta \quad (99)$$

Remarque : l'expression (99) suppose que le paramètre $\lambda(n)$ reste constant sur toute la longueur L de la séquence erronée. Cela correspond à un cas optimiste (comme montre la figure 14) de faible sélectivité temporelle et, pour être plus précis, on peut choisir une meilleure approximation de $\lambda(n)$ sur la longueur du bloc L. on propose de choisir la valeur suivante :

$$\forall n \in m,...,m+L-1,$$
$$\lambda(n) \approx h_{11}^*(2(m+L-1)+1).h_{11}(2m)+h_{21}(2m).h_{21}^*(2(m+L-1)+1) = \hat{\lambda}(m,L)$$

On peut démontrer, que conditionnellement à $h_{11}(2(m+L-1)$ et $h_{21}(2m)$, $\hat{\lambda}(m,L)$ a une distribution gaussienne:

$$\hat{\lambda}(m,L) =$$
$$\mathcal{N}(R_h(L).[|h_{11}(2(m+L-1)+1)|^2+|h_{21}(2m)|^2],(1-|R_h(L)|^2)[|h_{11}(2(m+L-1)+1)|^2+|h_{21}(2m)|^2]^2)$$

Et, en utilisant cette approximation, on obtient:

$$P_e(\boldsymbol{x} \to \boldsymbol{x}'|\boldsymbol{x}) \approx \frac{1}{\pi}.\int_0^{\pi/2} \frac{\sin^2\theta.(\sin^2\theta+(1-|R_h(L)|^2).\gamma)}{(\sin^2\theta+\gamma)^2}.d\theta \quad (100)$$

Il est intéressant d'utiliser (100) pour estimer le nouveau gain de diversité du système. Pour un rapport signal sur bruit élevé on a $\gamma >> \sin^2\theta$ et donc:

$$P_e(\boldsymbol{x} \to \boldsymbol{x}'|\boldsymbol{x}) \approx \frac{1}{\pi}.\int_0^{\pi/2} \frac{\sin^2\theta.(1-|R_h(L)|^2)}{\gamma}.d\theta = \frac{1}{4}.\frac{1-|R_h(L)|^2}{\sum_{n=m}^{m+L-1}\frac{[C_{2n}^2+C_{2n+1}^2]}{2.\sigma^2}}$$

$$P_e(\boldsymbol{x} \to \boldsymbol{x}'|\boldsymbol{x}) \approx \frac{1}{12.(Q+1).n.(E_b/N_0)}.\frac{1-|R_h(L)|^2}{\sum_{n=m}^{m+L-1}[C_{2n}^2+C_{2n+1}^2]} \qquad (101)$$

On voit donc d'après (101) que le gain de diversité est égal à 1. On a rétabli un gain de diversité dans le système mais ce gain reste inférieur au cas d'Alamouti conventionnel.

Pour obtenir les limites de taux d'erreur binaire (BER), nous avons à moyenner (101) sur la distribution du spectre des distances. On obtient les courbes de la figure 15.

Il est clair que le plancher de taux d'erreur binaire à forts SNR's a été évité. Les performances du taux d'erreur binaire ont été améliorées pour les taux Doppler élevés $f_d.T_s = 0.01$, puisque on obtient un taux d'erreur binaire de 7.10^{-4} a $E_b/N_0 = 20$ dB, tandis qu'il sature à $1.7.10^{-3}$, sur la figure 14.

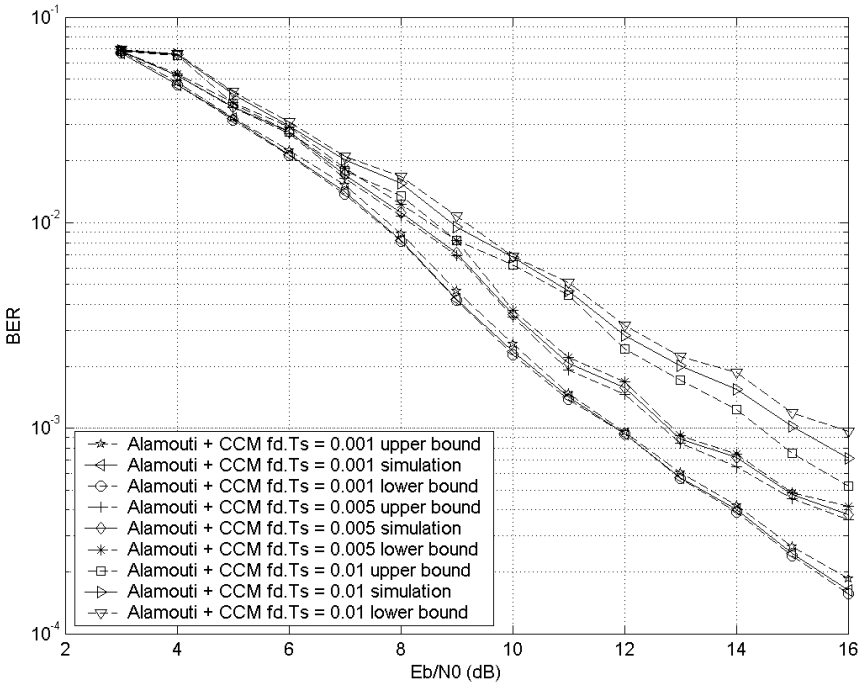


Fig. 15: résultant de simulation ZF et limites inférieures théoriques pour des canaux sélectifs en temps

IV. CONCLUSION

- Dans ce chapitre, nous avons optimisé les performances du schéma de modulation multidimensionnelle codée chaotique proposé antérieurement par S. Kozic.
- Nous avons proposé une nouvelle méthode d'approximation du calcul de la distribution du spectre des distances du codeur à l'aide de lois de Rayleigh ou de mélanges de lois gaussiennes ou de lois de Rayleigh. Cette approximation nous fournit ne méthode d'optimisation de la structure de ces codes. Nous avons montré qu'une telle structure faisant appel à un mapping multidimensionnel permettait d'obtenir des performances en terme de TEB supérieures à celle d'un système no-codé BPSK particulièrement à forts SNR's.
- Puis nous avons étudié les performances du schéma de modulation chaotique proposé lorsqu'il est concaténé avec un code temps-espace en bloc de type Alamouti dans le cas d'un système à deux antennes d'émission et une antenne de réception. Ces études sont été faites dans le cas des canaux quasi-statiques non-sélectifs en fréquence mais sélectifs en temps. Dans tous les cas, nous avons approximé les limites de BER puis nous avons montré que le récepteur perd tout gain de diversité à forts SNR's (gain de diversité nul) avec l'utilisation d'une combinaison MRC (Maximum Ratio Combining). Pour conserver une certaine diversité sur des canaux très sélectifs en temps nous avons proposé une égalisation matricielle de type ZF (Zero Forcing) et nous avons montré que le gain de diversité était égal à un dans ce cas.

CHAPITRE IV MODULATION CODÉE BASÉE SUR DES FONCTIONS CHAOTIQUES DE GRANDES DIMENSIONS

I. INTRODUCTION

Les performances des schémas de mapping générés à base de chaos présentées dans les chapitres précédents restent faibles. En effet, avec des mapping multidimensionnels tels que ceux présentés dans le chapitre précédent on arrive à dépasser les performances d'un système non codé mais dans des proportions relativement modestes (de l'ordre de 3 dB de gain). Il est difficile avec de tels résultats de parler de codage de canal performant! Le but de ce chapitre est de montrer qu'il est possible de construire des codes détecteurs-correcteurs d'erreurs qui offrent d'excellentes performances et qui peuvent se présenter comme une alternative sérieuse aux schémas classiques de codes linéaires performants tels que les turbo-codes ou les codes LDPC.

Pour obtenir de telles performances il faut pousser le concept de mapping multidimensionnel au bout de sa logique avec l'utilisation de matrices de mapping $\boldsymbol{A}$ de grande taille. En effet, l'idée qui consisterait à augmenter la taille mémoire du codeur dans le chapitre précédent n'est pas réellement exploitable car la complexité du décodeur de Viterbi associé explose rapidement (complexité exponentielle avec la taille de la mémoire). Cependant l'utilisation de matrices $\boldsymbol{A}$ de grandes tailles pose aussi le problème de la complexité du décodage en réception. Kozic & al [52] qui ont exploré cette voie en choisissant pour matrices $\boldsymbol{A}$ des matrices de codes à faible densité (LDGM : Low Density Generator Matrix), ont proposé une solution très prometteuse pour ce problème en montrant que l'ensemble des opérations d'encodage-mapping pouvait se mettre sous la forme d'un graphe factoriel (graphe de Tanner [53]) qui permettait d'utiliser un algorithme de décodage itératif basé sur la propagation de croyance (B.P. : Belief Propagation). Le système proposé par Kozic & al dans [52] peut s'interpréter comme un schéma de modulation de type BICM (Bit Interleaved Coded Modulation) où les matrices $\boldsymbol{A}$ servent à la fois d'entrelaceurs et de fonctions de mapping. Il est clair que la mise en facteur sous forme d'un graphe factoriel de l'algorithme de décodage n'est pas aisée pour un code non linéaire utilisant un mapping basé sur des fonctions génératrices de chaos. Cependant, si cette opération peut être réalisée, le code obtenu peut présenter des performances étonnantes et s'avérer une alternative crédible à des turbo-codes ou des codes LDPC.

Dans ce chapitre, nous reprenons l'idée directrice de Kozic issue de [52] en montrant comment on peut factoriser le processus d'encodage-modulation du système mais nous

proposons, pour aller plus loin, d'utiliser des matrices de mapping sur des corps plus complexes que GF(2). En effet, les travaux récents de Declercq et Fossorier [54] ont montré que l'augmentation de l'efficacité spectrale des schémas d'encodage à base de matrices de parité creuse pouvait conduire à une amélioration des performances du décodeur en réception. Cependant, la complexité du décodeur devant rester raisonnable, il faudra là aussi avoir recours à des algorithmes de décodage sous-optimaux pour le décodage des codes LDPC sur GF(q) avec : $q = 2^n$. C'est ainsi que nous proposons l'utilisation de l'algorithme EMS (Extended Min-Sum) proposé initialement par Declercq [55][56][57][58]. Deux versions de cet algorithme seront étudiées : la première travaille directement sur les probabilités et la seconde opère dans le domaine logarithmique afin de simplifier l'implantation matérielle.

II. PROCESSUS D'ENCODAGE

Le système chaotique est défini par une fonction chaotique à grande dimension qui est elle-même basée sur des matrices à faible densité. La fonction chaotique utilisée dans ce chapitre est définie:

$$x_{k+1} = 2.A.x_k \ mod\ q \qquad (1)$$

Où $\boldsymbol{x}_k$ est le vecteur d'entrée de taille n et $\boldsymbol{A}$ est un matrice de dimension $n \times n$ dont les éléments non nuls appartiennent à GF (q), autrement dit dont les éléments appartiennent à l'alphabet : $\mathcal{A} = (0,1,...,q-1)$ avec $q = 2^m$ Le corps de Galois utilisé est un corps d'extension d'ordre 2.

Le schéma d'encodage est donné dans la figure suivante :

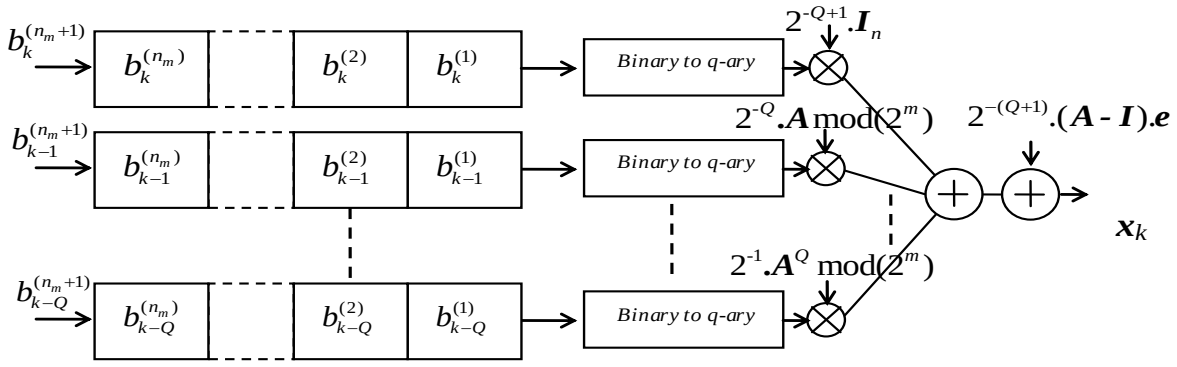


Figure. 1 : schéma d'encodage

L'ensemble des bits $\boldsymbol{b}=(\boldsymbol{b}_1,\boldsymbol{b}_2,...,\boldsymbol{b}_k,...)$ est groupé par des vecteurs $\boldsymbol{b}_{k+i-Q}$ de taille $n_m = n.m$ avec m l'efficacité spectrale utilisée dans le processus de codage et de décodage. Donc on peut écrire : $\boldsymbol{b}_{k+i-Q}=(b_{k+i-Q}^{(1)},b_{k+i-Q}^{(2)},...,b_{k+i-Q}^{(n.m-1)},b_{k+i-Q}^{(n.m)})^T$. Un convertisseur binaire/$GF(q)$ est utilisé pour obtenir les vecteurs $\boldsymbol{d}_{k+i-Q}=(d_{k+i-Q}^{(1)},d_{k+i-Q}^{(2)},...,d_{k+i-Q}^{(n-1)},d_{k+i-Q}^{(n)})^T$ contenant des symboles qui appartiennent à $GF(q)$: $d_i^{(p)}$, $p=1,2,...,n$ appartenant à l'alphabet : $\mathcal{A}=(0,1,...,q-1)$.

Chaque vecteur $\boldsymbol{d}_{k+i-Q}$ obtenu sera multiplié par la matrice à faible densité $\boldsymbol{A}^{Q-i}$ et pondéré par un facteur de $2^{-(i+1)}$ puis les vecteurs résultants sont sommés pour obtenir le vecteur $\boldsymbol{z}_k=\sum_{i=0}^{Q}2^{-(i+1)}.\boldsymbol{A}^{Q-i}.\boldsymbol{d}_{k+i-Q} \bmod(q)$. Finalement, pour obtenir la trajectoire chaotique, le vecteur $2^{-(Q+1)}.(\boldsymbol{A}-\boldsymbol{I}).\boldsymbol{e}$ est ajouté. Avec : $\boldsymbol{e}=[1,1,...,1]^T$.

Ce qui nous donne l'équation suivante :

$$\boldsymbol{x}_k=\sum_{i=0}^{Q}2^{-(i+1)}.\boldsymbol{A}^{Q-i}.\boldsymbol{d}_{k+i-Q}+\frac{2^{-Q}}{2}.\boldsymbol{p} \bmod q \qquad (2)$$

Avec : $\boldsymbol{p}=2^{-Q}.(\boldsymbol{A}-\boldsymbol{I}).\boldsymbol{e}=K.(\boldsymbol{A}-\boldsymbol{I}).\boldsymbol{e}$

L'équation est appelée l'expression à très grande dimension associée avec le système chaotique On peut représenter la règle d'encodage d'une autre façon en utilisant l'équation suivante :

$$\boldsymbol{x}_{k+1}=2.\boldsymbol{A}.\boldsymbol{x}_k \bmod q+2^{-Q}.(\boldsymbol{d}_{k+1}-1/2.\boldsymbol{e}) \bmod q \qquad (3)$$

L'équation (3) est plus simple que l'équation (2) pour comprendre l'algorithme de codage, et, de plus, elle est mieux adaptée à une factorisation du graphe.

L'équation représente un système dynamique contrôlé par des perturbations stochastiques de petites amplitudes 2^{-Q} qui constituent le signal d'entrée. Il est évident que lorsque $Q\to+\infty$, les petites perturbations s'annulent et le signal de sortie tend vers l'état chaotique.

Il est important de noter que le décodage d'un code à faible densité (LDPC) implique que nous utilisons une forme systématique pour le processus d'encodage. C'est le cas ici :

$\boldsymbol{b} = (\boldsymbol{b}_1, \boldsymbol{b}_2, ..., \boldsymbol{b}_k, ...)$ est la partie systématique et $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_k, ...)$ est la partie du contrôle de la redondance.

Le taux de codage global est égal à 0.5 parce que la partie systématique et la partie contrôle de redondance ont la même dimension.

III. GRAPHE FACTORIEL ET ALGORITHME DE DÉCODAGE

Pour obtenir un algorithme de décodage efficace, il faut d'abord représenter le processus d'encodage-mapping sous la forme d'un graphe factoriel. On peut alors utiliser l'algorithme de propagation de croyance pour itérer entre les nœuds de variable et les nœuds de parité afin de décoder le signal reçu. Cependant, dans le cas d'un système non-linéaire comme celui étudié ici, l'obtention de la factorisation sous forme de graphe n'est pas évidente. Kozic dans ses études a proposé plusieurs solutions :

- La première est basée sur l'écriture de l'équation (3) qui peut être interprétée comme une équation de parité généralisée.
- La seconde est basée sur l'écriture de l'équation (2) qui peut être interprétée comme un développement à grande dimension du système chaotique.

La première écriture n'est pas adaptée à un décodage fiable car elle supposerait d'obtenir des informations sur $\boldsymbol{b}_k$ à partir d'informations sur les états $\boldsymbol{x}_k$ qui constituent le graphe des nœuds de variables et des nœuds de parité et, de plus, $\boldsymbol{d}_k$ se retrouve multiplié par une très faible valeur 2^{-Q} qui rend le décodage délicat.

La seconde écriture est donc la mieux exploitable pour la factorisation par graphe ; cependant elle présente l'inconvénient de faire appel à des puissances de : $\boldsymbol{A}$ ce qui peut générer des cycles courts dans le graphe factoriel. Par exemple, même si $\boldsymbol{A}$ ne présente pas de cycle court, le calcul de $\boldsymbol{A}^2$ fait apparaître des cycles d'ordre 4.

La factorisation du graphe de ce codage non linéaire fait intervenir trois étapes. La première étape concerne l'obtention des symboles $\boldsymbol{x}_k$ à partir des vecteurs $\boldsymbol{d}_{k+i-Q}$ et sera nommée graphe du développement du système chaotique de grande dimension. La deuxième étape

concerne le décodage des codes LDPC contenus dans les matrices $\boldsymbol{A}$, ce décodage doit tenir compte que ces matrices sont construites sur un corps $GF(q)$ non binaire et le graphe correspondant est celui d'un code LDPC q-aire conventionnel. Enfin, la troisième étape concerne la façon dont les bits d'informations se décalent au fur et à mesure de l'encodage des vecteurs : $\boldsymbol{b}_{k+i-Q}$. Ce décalage est totalement semblable à l'encodage d'un train binaire par un codeur convolutif. Ce mécanisme étant totalement similaire au comportement d'un codeur convolutif, son graphe sera nommé graphe du codeur convolutif.

C'est la première étape qui est la plus originale et la plus délicate à mettre en œuvre. En fait, tout est basé sur l'écriture de l'équation (2). On définit alors, à partir de cette équation, une fonction génératrice de la forme :

$$g(\boldsymbol{x}_k, \boldsymbol{d}_k, ..., \boldsymbol{d}_{k-Q}) = \begin{cases} 1, \text{ if } \boldsymbol{x}_k = \sum_{i=0}^{Q} 2^{-(i+1)}.\boldsymbol{A}^{Q-i}.\boldsymbol{d}_{k+i-Q} + \frac{q}{2}.\boldsymbol{p} \bmod q \\ 0, \text{otherwise} \end{cases} \qquad (4)$$

Pour aller plus loin dans le développement et la factorisation on définit alors les variables supplémentaires $\boldsymbol{\mu}_{i,j}$ par :

$$\boldsymbol{x}_k = \sum_{i=0}^{Q} \boldsymbol{\mu}_{i,Q-i} + 2^{-(Q+1)}.\boldsymbol{p} \bmod q \qquad (5)$$

On a de façon évidente la relation :

$$\boldsymbol{\mu}_{i,j+1} = \boldsymbol{A}.\boldsymbol{\mu}_{i,j}. \bmod q \qquad (6)$$

avec : $\boldsymbol{\mu}_{i,0} = 2^{-(i+1)}.\boldsymbol{d}_{k+i-Q}$ Avec ces variables la fonction g devient uniquement fonction des variables : $\boldsymbol{\mu}_{i,j}$. Pour continuer à factoriser g on introduit les fonctions $g_{i,j}$ définies par:

$$g_{i,j+1}(\boldsymbol{\mu}_{i,j+1}, \boldsymbol{\mu}_{i,j}) = \begin{cases} 1, \text{ if } \boldsymbol{\mu}_{i,j+1} = \boldsymbol{A}.\boldsymbol{\mu}_{i,j} \bmod q \\ 0 \text{ otherwise} \end{cases} \qquad (7)$$

et g_0 :

$$g_0(\boldsymbol{x}_k, \boldsymbol{\mu}_{0,Q}, \boldsymbol{\mu}_{1,Q-1}, ...) = \begin{cases} 1, \text{ if } \boldsymbol{x}_k = \sum_{i=0}^{Q} \boldsymbol{\mu}_{i,Q-i} + 2^{-(Q+1)}.\boldsymbol{p} \bmod q \\ 0 \text{ otherwise} \end{cases} \qquad (8)$$

On peut alors écrire g sous la forme :

$$g(.) = g_0 . \prod_{i=0}^{Q} \prod_{j=0}^{Q-i} g_{ij}(.) \quad (9)$$

La factorisation de $g(.)$ peut alors se mettre sous la forme d'un graphe représenté sur la Figure.2.

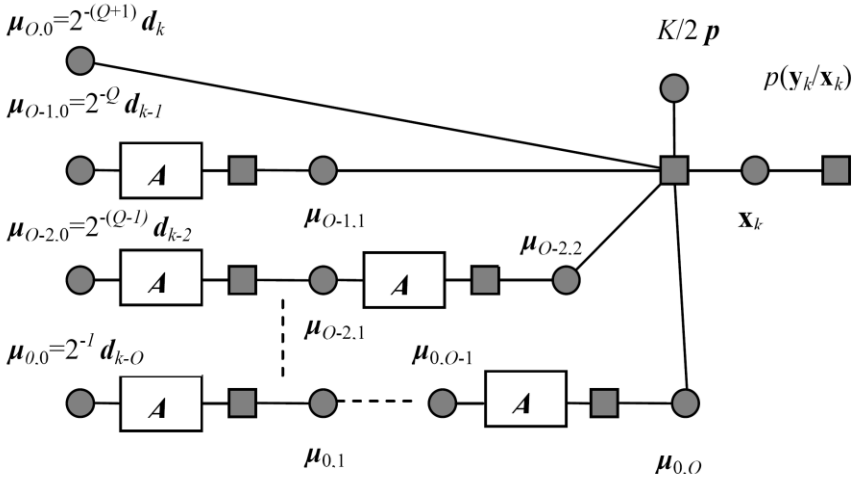


Figure 2 : graphe factoriel correspondant au développement de l'équation (2)

Il est possible de continuer à factoriser la fonction $g(.)$ en suivant le formalisme du décodage des codes LDPC. Pour cela, les variables à gauche de l'équation (7) sont appelées les variables de parité et les variables à droite sont appelées les variables d'information. On définit alors de la même façon que pour les codes LDPC les sous-ensembles suivants : $[\boldsymbol{A} : (a_{l,m})]$ $\mathcal{N}(l) = \{m : a_{lm} \neq 0\}$ qui représente l'ensemble des variables d'information qui participent à l'équation de parité l. De même, on définit $\mathcal{M}(m) = \{l : a_{l,m} \neq 0\}$ comme l'ensemble des symboles de parité qui dépendent de la variable d'information m. Dans ce cas, (7) peut se réécrire :

$$\boldsymbol{\mu}_{i,j+1}^{(l)} = \sum_{m \in \mathcal{N}(l)} a_{lm} . \boldsymbol{\mu}_{i,j}^{(m)} \bmod q \quad m,l \in [1,n] \times [1,n] \quad (10)$$

On pose alors:

$$g_{i,j+1}^{(l)} = \begin{cases} 1, \text{ if } \boldsymbol{\mu}_{i,j+1}^{(l)} = \sum_{m \in \mathcal{N}(l)} a_{lm}.\boldsymbol{\mu}_{i,j}^{(m)} \bmod q \\ 0, \text{ sinon} \end{cases} \qquad (11)$$

Sur la figure (2) les cercles représentent des variables d'information tandis que les carrés représentent les nœuds ou équations de parité. Le décodage symbole par symbole de la trajectoire chaotique complète peut s'écrire :

$$\hat{d}_{k+i-Q}^{(.)} = \arg\max_{d_{k+i-Q}^{(.)} \in \{0,1,\ldots,q-1\}} \sum_{\sim d_{k+i-Q}^{(.)}} p(\boldsymbol{x}_0).\prod_{j=1}^{M+Q} p(\boldsymbol{y}_j \,|\, \boldsymbol{x}_j) \times g(\boldsymbol{x}_j, \boldsymbol{d}_j, \ldots, \boldsymbol{d}_{j-Q}) \times p(\boldsymbol{d}_j, \ldots, \boldsymbol{d}_{j-Q}) \times p(\boldsymbol{x}_{j+1} \,|\, \boldsymbol{x}_j, \boldsymbol{d}_{j+1}) \qquad (12)$$

La quantité $\sim d_{k+1-Q}$ signifie la sommation sur toutes les composantes sauf : d_{k+1-Q} .

La deuxième étape est tout à fait conventionnelle et correspond à la mise sous forme d'un graphe factoriel de la matrice de contrôle de parité d'un code LDPC. Ce graphe peut se mettre sous la forme suivante :

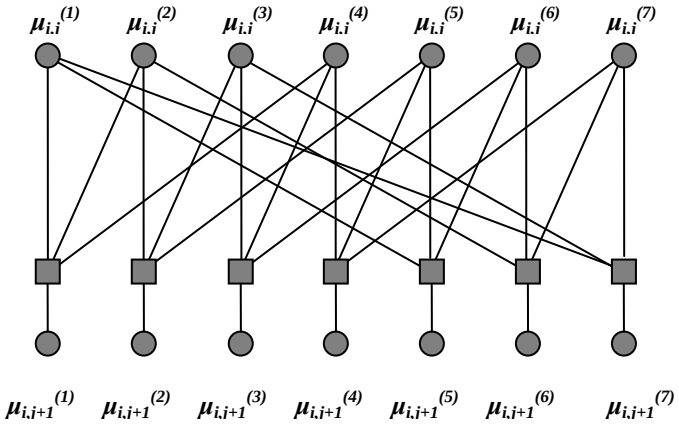


Figure 3 : graphe factoriel de la matrice **A**.

Une fois de plus les cercles représentent les nœuds de variable et les carrés les nœuds de parité.

La troisième étape tient compte des opérations de décalage dans les registres et de conversion binaire-q aire qui représentent les transitions des états entre les instants j et j+1. Cette opération de décalage-conversion binaire-q aire peut être résumée par la probabilité de

transition suivante : $g_c = p(\boldsymbol{x}_{j+1} \left| \boldsymbol{x}_j, \boldsymbol{d}_{j+1} \right.)$. L'état du codeur à l'instant j+1 dépend des séquences de symboles $\boldsymbol{d}_{j+1},...,\boldsymbol{d}_{j+1-Q}$ et il peut être calculé à partir des rapports de vraisemblance des symboles $\boldsymbol{d}_j,...,\boldsymbol{d}_{j+1-Q}$ et du rapport de vraisemblance du symbole supplémentaire : $\boldsymbol{d}_{j+1}$. En combinant ensemble la factorisation de g_c et la mise sous forme de graphe factoriel du développement du système chaotique à grande dimension [équations (2) et (4)] on obtient le graphe factoriel de la Figure 4. Ce graphe tient compte du processus de décalage dans les registres et comprend deux types de calcul de probabilité avant et retour avec le calcul des coefficients α et β comme dans le cas de l'algorithme BCJR.

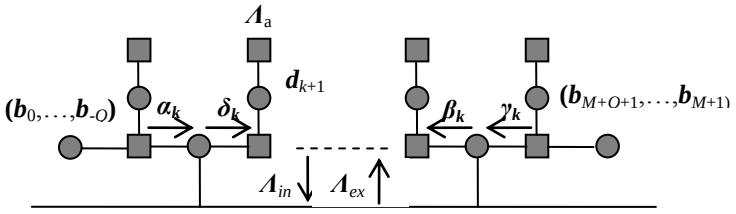


Figure 4 : Graphe factoriel du codeur chaotique complet

Les quantités $\alpha_k, \beta_k, \gamma_k$ et δ_k sont définis de la même façon que dans[52] à la nuance près que nous travaillons au niveau des symboles et non plus au niveau des bits. Ainsi, on doit par exemple transformer des probabilités à priori au niveau des bits en probabilités à priori au niveau des symboles. Ceci est obtenu avec la formule :

$$P[\boldsymbol{d}_{k+i-Q} = (b_{k+i-Q}^{(1)}, b_{k+i-Q}^{(2)}, ..., b_{k+i-Q}^{m})^T] = \prod_{j=1}^{m} \frac{e^{b_{k+i-Q}^{(j)} \cdot \tilde{\Lambda}_e(b_{k+i-Q}^{(j)})}}{1 + e^{b_{k+i-Q}^{(j)} \cdot \tilde{\Lambda}_e(b_{k+i-Q}^{(j)})}} \qquad (13)$$

La quantité $\tilde{\Lambda}_e(b_{k+i-Q}^{(j)})$ désigne le rapport de vraisemblance logarithmique correspondant au bit : $b_{k+i-Q}^{(j)}$. En ce qui concerne le problème inverse on doit exprimer le rapport de vraisemblance logarithmique du bit $b_{k+i-Q}^{(j)}$ à partir des rapports de vraisemblance sur les symboles, on a la formule:

$$\Lambda_e(b_{k+i-Q}^{(j-1).m+k}) = \log \frac{\sum_{d_{k+i-Q}^{(j)}:b_{k+i-Q}^{(j-1).m+k}=1} e^{\Lambda_e(d_{k+i-Q}^{(j)})}}{\sum_{d_{k+i-Q}^{(j)}:b_{k+i-Q}^{(j-1).m+k}=0} e^{\Lambda_e(d_{k+i-Q}^{(j)})}} \qquad (14)$$

Où, de façon évidente, $\Lambda_e(d_{k+i-Q}^{(j)})$ correspond au rapport de vraisemblance logarithmique des symboles : $\boldsymbol{d}_{k+i-Q}^{(j)} = [b_{k+i-Q}^{(j-1).m+1}, b_{k+i-Q}^{(j-1).m+2}, ..., b_{k+i-Q}^{(j-1).m+m}]$.

IV. ALGORITHME DE DÉCODAGE DES CODES LDPC Q-AIRES

L'algorithme de décodage global est le même que celui présenté dans [52] à l'exception notable de l'emploi d'un code LDPC sur un corps non-binaire de type $GF(2^m)$. Le décodage d'un code LDPC q-aire présente de nombreuses spécificités par rapport au cas binaire et a une complexité de décodage bien supérieure ne serait-ce qu'au niveau de la vérification des équations de parité puisqu'il faut trouver tous les tuples ou groupes de symboles qui satisfont certaines équations de parité. Un algorithme performant au sens du compromis performances/complexité et récemment proposé par Declercq & al sous le nom d'algorithme EMS (Extended Min Sum) a été développé pour notre application de codeur chaotique. Nous allons rappeler ici les principales étapes de réalisation du décodage de canal à l'aide de l'algorithme EMS ou log EMS.

Pour l'algorithme de propagation de croyance des codes LDPC construits sur un corps non-binaire $GF(q)$, il est utile d'utiliser une représentation tensorielle des messages transmis. Pour un code LDPC, on définit par $C_{\boldsymbol{H}}$ l'ensemble des mots de code généré sur $GF(q)$ avec $q = 2^m$ et la matrice de contrôle de parité du code $\boldsymbol{H}$ de taille $M \times N$ qui définit le noyau du code est une matrice creuse telle que :

$$C_{\boldsymbol{H}} = \left\{ \boldsymbol{c} \in GF(q)^N \,\middle|\, \boldsymbol{H}.\boldsymbol{c} = \boldsymbol{0} \right\} \qquad (15)$$

Le graphe factoriel du code consiste en un ensemble de nœuds de variables appartenant à *GF*(*q*), connectés à un ensemble de nœuds de parité. Les arêtes qui connectent les deux types de nœud transportent des messages qui représentent les densités de probabilité (pdf's) des mots de code. Puisque ces mots de code sont des variables aléatoires sur le corps *GF*(*q*), les messages sont des densités de probabilité discrètes de taille *q*. La construction des corps de Galois est présentée en Annexe 1. Lorsque le corps de Galois est une extension de *GF*(2), avec typiquement $q = 2^m$, les messages peuvent être convenablement représentés par des tenseurs de taille 2 et de dimension *m*. En effet, sur $GF(2^m)$, les éléments du corps peuvent être représentés à l'aide d'un polynôme $i(x) = \sum_{l=1}^{m} i_l . x^{l-1}$ de degré *p*-1 avec des coefficients binaires (cf Annexe 1). Ainsi n'importe quel ensemble de *m* valeurs binaires $\{i_l\}_{l=1,...,m}$ détermine un unique élément *i*(*x*). En utilisant cette représentation, un message *i*(*x*) sur une arête connectée à un nœud de variable est un tenseur $\{U[i_1,...,i_m]\}_{i_1,...,i_m}$ indexé par les coefficients binaires de *i*(*x*). Par exemple, *U*[0,1,1] correspond à la probabilité $p(i(x) = x + x^2)$ dans *GF* (8).

IV.1. Algorithme de décodage par propagation de croyance sur GF(2m)

Pour un code sur *GF*(*q*), un nœud de parité est associé à l'équation de parité suivante :

$$\sum_{k=1}^{d_c} h_k(x).i_k(x) = 0 \mod m(x) \qquad (16)$$

Où *m*(*x*) est un polynôme primitif de degré *m*-1 sur *GF*(*q*). L'équation (16) exprime que les nœuds de variables nécessaires au déroulement de l'algorithme de propagation de croyance pour un nœud de parité ne sont pas constitués uniquement des mots du code d'origine mais des mots du code multipliés par les valeurs non nulles de la matrice de parité ***H***. La transformation correspondante sur le graphe est réalisée en additionnant des nœuds de variables correspondant à la multiplication des mots de code $i_k(x)$ par les valeurs correspondantes non nulles de ***H***. La transformation correspondante est illustrée sur la figure 1 ci-dessous.

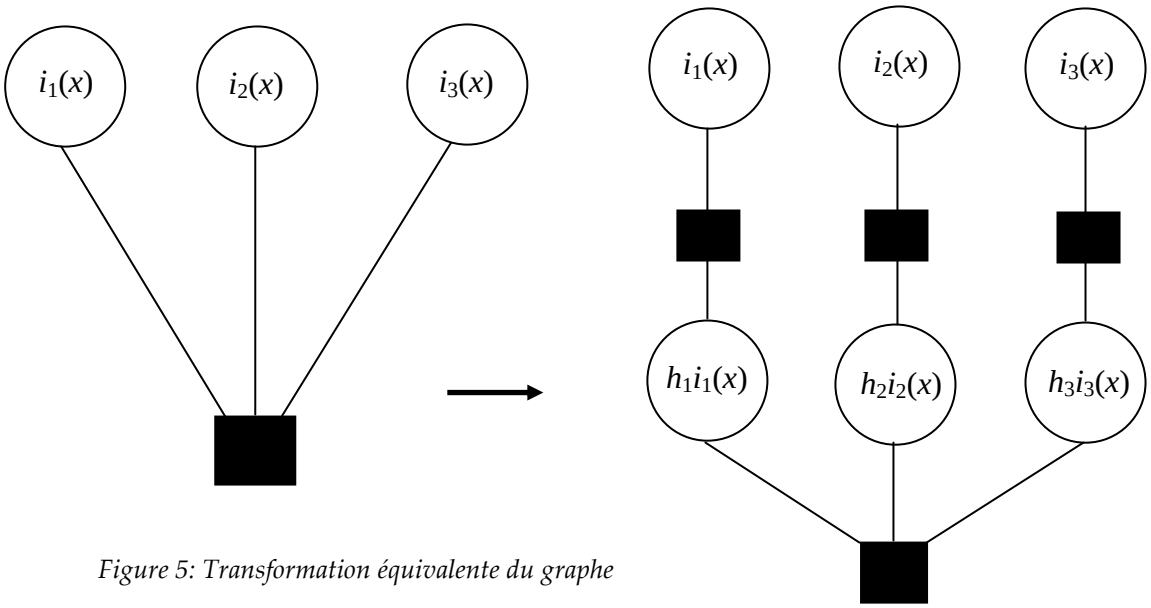


Figure 5: Transformation équivalente du graphe factoriel

La fonction de nœud qui connecte les deux nœuds de variables $i_k(x)$ et $h_k(x).i_k(x)$ réalise une permutation des valeurs des messages. La permutation qui est utilisée pour mettre à jour les messages correspond à la multiplication des indices du tenseur par $h_k(x)$ pour passer du nœud $i_k(x)$ au nœud $h_k(x).i_k(x)$ et à la division des indices par $h_k(x)$ dans l'autre sens. En utilisant cette transformation du graphe factoriel, la mise à jour aux nœuds de parité est une convolution de tous les messages entrants, exactement comme dans le cas binaire.

On utilise les notations suivantes : on note $\{V_{pv}\}_{v=1,...,d_v}$ l'ensemble des messages entrant sur un nœud de variable de degré d_v et $\{U_{vp}\}_{v=1,...,d_v}$ l'ensemble des messages de sortie de ce nœud. L'indice *p*v indique que le message arrive d'un nœud de permutation et se dirige vers un nœud de variables. De la même façon, l'indice v*p* indique l'autre sens de propagation. De la même façon on définit les messages $\{U_{pc}\}_{c=1,...,d_c}$ (respectivement $\{V_{pc}\}_{c=1,...,d_c}$) à l'entrée (respectivement à la sortie) d'un nœud de degré d_c.

L'initialisation du décodeur est réalisée avec le calcul des rapports de vraisemblance du canal noté $L[i_1,...,i_p]$. Dans le cas où le canal a des données binaires et un bruit additif Gaussien le calcul des rapports de vraisemblance prend la forme :

$$L[i_1,...,i_p] = \prod_{n=1}^{p} l(i_n) \quad (17)$$

avec $l(i_n) = \text{Proba}(y_n | b_n = i_n) = \exp((y_n - b_n)^2 / 2.\sigma^2)$ et où b_n est le $n^{\text{ième}}$ bit du symbol correspondant sur $GF(q)$ et y_n est la sortie bruitée du canal de propagation.

Les trois étapes d'une itération dans le décodage s'écrivent alors de la manière suivante :

E_1 : Mise à jour des nœuds de variables

Pour la mise à jour d'un nœud de variables de degré d_v :

$$U_{tp} = L \times . \prod_{v=1, v \neq t}^{d_v} .V_{pv} \quad t = 1,...,d_v \qquad (18)$$

soit encore :

$$\begin{aligned} &U_{tp}[i_1,...,i_p] = L[i_1,...,i_p] \prod_{v=1, v \neq t}^{d_v} V_{pv}[i_1,...,i_p] \\ &(i_1,...,i_p) \in \{0,1\}^p, t = 1,...,d_v \end{aligned} \qquad (19)$$

où $\times .$ est défini comme le produit terme à terme de tenseurs et L désigne le tenseur des rapports de vraisemblance issus du canal (voir équation (17)). De plus, comme les messages représentent des densités de probabilité, on doit normaliser les messages obtenus après l'étape (19) de telle sorte que : $\sum_{i_1,...,i_p} U_{tp}[i_1,...,i_p] = 1$.

E_2 : Etape de permutation, depuis les noeuds de variables vers les nœuds de parité

On a :

$$\text{vec}(U_{pc}) = P_{h(x)}.\text{vec}(U_{vp}) \qquad (20)$$

ou encore :

$$\begin{aligned} &U_{pc}[i_1,...,i_p] = U_{vp}[j_1,...,j_p] \quad (i_1,...,i_p) \in \{0,1\}^p \\ &\text{avec } i(x) = h(x).j(x) \qquad (21) \end{aligned}$$

où $P_{h(x)}$ est une matrice de permutation ($q \times q$) correspondant à $h(x)$ et vec(U) rassemble dans un vecteur colonne toutes les valeurs du tenseur U. On peut remarquer que, puisque les corps de Galois sont des corps cycliques, la matrice de permutation $P_{h(x)}$ représente un décalage cyclique des valeurs du message transmis. Dans l'autre direction, à savoir depuis les nœuds de parité vers les nœuds de variables, on utilisera bien sûr la matrice inverse $P_{h(x)}^{-1}$.

E_3 : mise à jour des noeuds de parité

En utilisant les noeuds secondaires $h(x).i(x)$, tous les nœuds de parité suivent les mêmes règles et ne dépendent plus des entrées non nulles de $\boldsymbol{H}$. L'algorithme de propagation de croyance pour un nœud de parité de degré d_c correspond à un produit de convolution des densités de probabilité sur $GF(2^m)$.

$$\mathrm{V}_{tp} = \otimes_{c=1, c\neq t}^{d_c} U_{pc} \quad t = 1,\ldots,d_c \qquad (22)$$

ou encore :

$$\mathrm{V}_{tp}[i_{t_1},\ldots,i_{t_p}] = \sum_{\{i_c(x)\}_{c\neq t}} \prod_{c=1, c\neq t}^{d_c} U_{pc}[i_{c_1},\ldots,i_{c_p}] \times 1\mathrm{I}_{\sum_{c=1}^{d_c} i_c(x)=0} \quad t = 1,\ldots,d_c \qquad (23)$$

Où la fonction $1\mathrm{I}_S$ désigne une fonction indicatrice égale à 1 si et seulement si la condition S est remplie. On peut aussi exprimer la somme (23) sans l'aide d'une fonction indicatrice en utilisant un ensemble de configurations. On définit l'ensemble suivant :

$$\mathrm{Conf}_{i_t(x)} = \left\{ \{i_c(x)\}_{c\neq t} : \sum_{c=1}^{d_c} i_c(x) = 0 \right\} \qquad (24)$$

En utilisant (24), l'étape E_3 peut se réécrire sous la forme :
Etape somme-produit : mise à jour d'un nœud de parité pour un nœud de degré d_c.

$$\mathrm{V}_{tp}[i_{t_1},\ldots,i_{t_p}] = \sum_{\{i_c(x)\}\in \mathrm{Conf}_{i_t(x)}} \prod_{c=1, c\neq t}^{d_c} U_{pc}[i_{c_1},\ldots,i_{c_p}] \qquad (25)$$

IV.2. Simplification de l'algorithme par utilisation de FFT :

La complexité de l'étape E_3 dans l'algorithme présenté précédemment peut devenir rapidement prohibitive. En effet, le nombre d'opérations élémentaires nécessaires au calcul de V_{tp} croît exponentiellement avec la taille du corps q et le degré des équations de parité d_c. Pour simplifier de façon importante cette étape, plusieurs auteurs ont proposé de réaliser l'étape E_3 dans le domaine fréquentiel afin de transformer la convolution en produit simple. La transformation des densités des variables représentées par des tenseurs dans le corps $GF(2^m)$ depuis le domaine temporel vers le domaine fréquentiel est très simple puisqu'il s'agit de transformée de Fourier d'ordre 2.

Proposition : appelons $U_{i_1,\ldots,i_p}$ un tenseur d'ordre 2 et de dimension m et qui représente une densité de probabilité de la variable aléatoire $i(x) \in GF(2^m)$, alors la transformée de Fourier de $U_{i_1,\ldots,i_p}$ est donnée par :

$$W = \mathcal{F}(U) = U \times_1 \boldsymbol{F} \times_2 \boldsymbol{F} \ldots \times_p \boldsymbol{F} \qquad (26)$$

Où $\times_k$ représente le produit tensoriel sur la $k^{\text{ième}}$ dimension du tenseur et $\boldsymbol{F}$ est la matrice de la transformée de Fourier rapide donnée par :

$$\boldsymbol{F} = \frac{1}{\sqrt{2}} . \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \qquad (27)$$

Avec ces définitions, le produit tensoriel $Z = U \times_k \boldsymbol{F}$ est défini par : (pour $(i_1,\ldots,i_{k-1},i_{k+1},\ldots,i_p) \in \{0,1\}^{p-1}$)

$$Z[i_1,\ldots,i_{k-1},0,i_{k+1},\ldots,i_p] = \frac{1}{\sqrt{2}} . (U[i_1,\ldots,i_{k-1},0,i_{k+1},\ldots,i_p] + U[i_1,\ldots,i_{k-1},1,i_{k+1},\ldots,i_p]) \qquad (28)$$

$$Z[i_1,\ldots,i_{k-1},1,i_{k+1},\ldots,i_p] = \frac{1}{\sqrt{2}} . (U[i_1,\ldots,i_{k-1},0,i_{k+1},\ldots,i_p] - U[i_1,\ldots,i_{k-1},1,i_{k+1},\ldots,i_p]) \qquad (29)$$

En utilisant la transformée de Fourier (31), on change les nœuds de parité en nœuds produits dans le graphe factoriel du code sur $GF(2^m)$. La figure 2 illustre ces transformations dans le cas d'un code LDPC régulier avec $(d_v, d_c) = (2,4)$.

Avec le passage dans le domaine fréquentiel, l'étape E_3 est modifiée de la façon suivante :

E'$_3$: mise à jour des noeuds de parité en utilisant une FFT

La mise à jour dans le domaine fréquentiel s'écrit :

$$V_{tp} = \mathcal{F}\left(\prod_{c=1, c\neq t}^{d_c} .\mathcal{F}(U_{pc}) \right) \quad t = 1,..., d_c \qquad (30)$$

Ainsi, la complexité de l'étape E_3 est réduite en $\vartheta(d_c.m.q)$ with $q = 2^m$. De plus, on peut remarquer que la forme particulière de la FFT dans $GF(2^m)$ fait que l'on ne traite plus que des additions, il n'y a plus de multiplication.

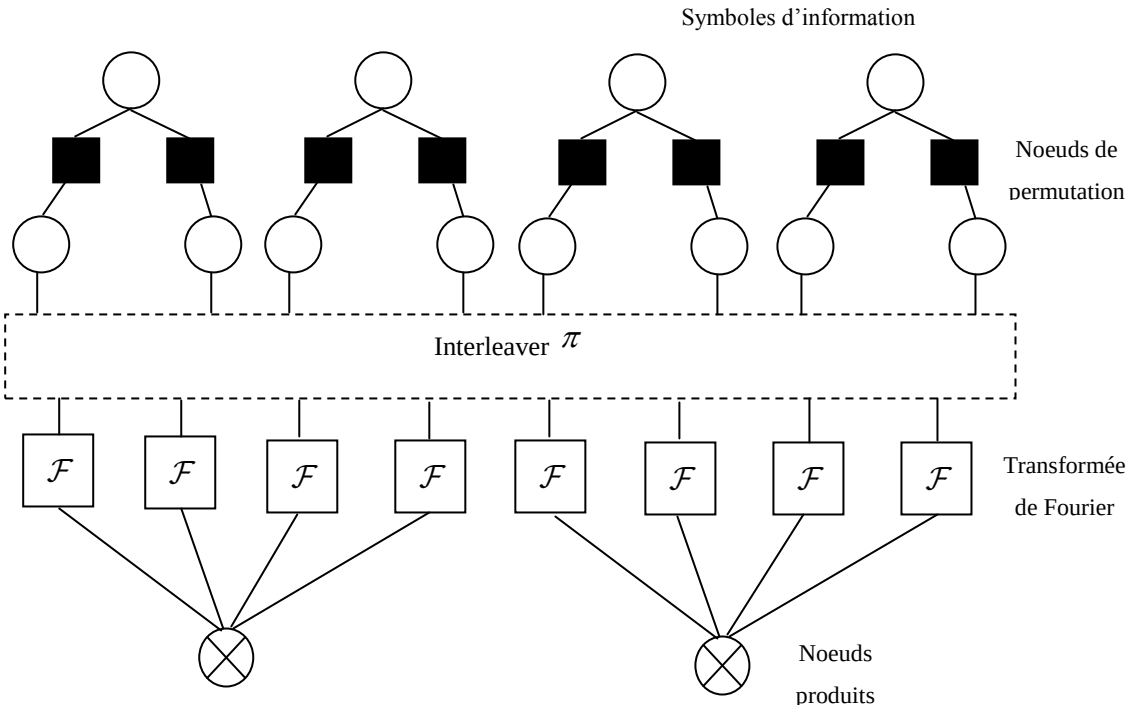


Figure 6 : Graphe factoriel équivalent dans le domaine fréquentiel

IV.3. Algorithme EMS dans le domaine logarithmique :

Nous présentons dans ce paragraphe l'algorithme EMS proposé par Declercq et Fossorier dans [56] qui opère, comme pour le Max-log-MAP des turbo-codes dans le domaine logarithmique à l'aide de rapports de vraisemblance logarithmiques. Le rapport de vraisemblance logarithmique s'exprime sous la forme : $LLR(u) = \log\left(\frac{\Pr oba(u=0)}{\Pr oba(u=1)}\right)$ dans le cas $q = 2$. Dans le cas $q = 2^m$, le message est composé de q composantes dont $q - 1$ qui sont non nulles. On utilise alors la définition suivante pour les rapports de vraisemblance logarithmiques de n'importe quel message tensoriel U :

$$\overline{U}[i_1,...,i_p] = \log\left(\frac{U[i_1,...,i_p]}{U[0,...,0]}\right) \quad \forall (i_1,...,i_p) \in \{0,1\}^p \qquad (31)$$

Ainsi, on a forcément : $\overline{U}[0,...,0] = 0$.

Le but de l'algorithme EMS est de simplifier encore l'étape E'$_3$ obtenue en travaillant dans le domaine fréquentiel avec une FFT. Pour cela, la première idée est de sélectionner (on suppose un corps avec un nombre élevé d'éléments typiquement m est au moins égal à 6) dans chaque message arrivant $\overline{U}_{pc}$ un nombre limité de valeurs à savoir les n_m valeurs les plus élevées. Toute la difficulté est bien sûr de bien choisir n_m. Les auteurs de l'article original évoquent une possibilité de changer sa valeur d'un message à l'autre en tenant compte d'une dynamique maximale. Cependant, pour des raisons de simplicité, on se contentera ici de prendre une valeur n_m fixe. On note : $\overline{u}_{pc}^{(k_c)}$, $k_c = 1,...,n_m$ les n_m valeurs les plus élevées de $\overline{U}_{pc}$. L'élément associé à une contrainte de parité dans le corps de Galois $GF(2^m)$ est noté:

$$\begin{aligned} \overline{u}_{pc}^{(k_c)} &= \log\left(\frac{\Pr oba(h_c(x).i_c(x) = \alpha_c^{(k_c)}(x))}{\Pr oba(h_c(x).i_c(x) = 0)}\right) \\ &= \overline{U}_{pc}[\alpha_{c_1}^{(k_c)}...\alpha_{c_p}^{(k_c)}] \end{aligned} \qquad (32)$$

A partir ce cet ensemble de n_m valeurs, on construit l'ensemble suivant des configurations :

$$\mathrm{Conf}(n_m) = \left\{ \boldsymbol{\alpha_k} = [\alpha_1^{(k_1)}(x),...,\alpha_{d_c-1}^{(k_{d_c-1})}(x)]^T : \forall \boldsymbol{k} = [k_1,...,k_{d_c-1}]^T \in \{1,...,n_m\}^{d_c-1} \right\} \quad (33)$$

Chaque vecteur de taille $d_c - 1$ dans le corps $GF(2^m)$ est appelé une configuration. L'ensemble $\mathrm{Conf}(n_m)$ correspond à l'ensemble des configurations construites à partir des n_m valeurs de probabilité les plus grandes dans chaque message entrant. Ainsi, son cardinal est :

$$\left|\mathrm{Conf}(n_m)\right| = n_m^{d_c-1} \quad (34)$$

$\mathrm{Conf}(1)$ contient seulement une configuration qui sera appelée la configuration d'ordre zéro. Pour certains cas, lorsque les valeurs de d_c et n_m sont importantes, le nombre de configurations dans $\mathrm{Conf}(n_m)$ est trop important. Pour le réduire, on considère le sous-ensemble suivant de $\mathrm{Conf}(n_m)$ pour $n_c \le d_c - 1$:

$$\mathrm{Conf}(n_m, n_c) = \mathrm{Conf}(n_m)^{(0)} \cup \mathrm{Conf}(n_m)^{(1)} \cup ... \cup \mathrm{Conf}(n_m)^{(n_c)} \quad (35)$$

Où $\mathrm{Conf}(n_m)^{(l)}$ désigne le sous-ensemble de configurations qui diffèrent de la configuration d'ordre zéro pour exactement *l* entrées. L'ensemble $\mathrm{Conf}(n_m, n_c)$ est donc le sous-ensemble des configurations appartenant à $\mathrm{Conf}(n_m)$ qui diffèrent par au moins n_c entrées de la configuration d'ordre zéro. Le nombre d'éléments dans $\mathrm{Conf}(n_m, n_c)$ est égal à:

$$\left|\mathrm{Conf}(n_m, n_c)\right| = \sum_{k=0}^{n_c} C_{d_c-1}^{k} (n_m - 1)^k \equiv C_{d_c-1}^{n_c} . n_m^{n_c} \quad (36)$$

L'idée d'utiliser $\mathrm{Conf}(n_m, n_c)$ à la place de $\mathrm{Conf}(n_m)$ est, une fois de plus, de simplifier la complexité car la cardinalité de $\mathrm{Conf}(n_m, n_c)$ est généralement bien inférieure à celle de $\mathrm{Conf}(n_m)$. On a de plus : $\mathrm{Conf}(n_m) = \mathrm{Conf}(n_m, d_c - 1)$.

Un degré de confiance est attribué à chaque configuration, cette confiance est calculée par la formule :

$$L(\boldsymbol{\alpha_k}) = \sum_{c=1,...,d_c-1} \bar{u}_c^{(k_c)} \quad (37)$$

A partir des degrés de confiance $L(\boldsymbol{\alpha_k})$ des configurations $\mathrm{Conf}(n_m, n_c)$, on est alors capable de définir un algorithme simplifié de propagation de croyance très efficace que les auteurs de l'article ont appelé algorithme EMS (Extended Min-Sum). Pour détailler cet algorithme on note $\mathrm{Conf}_{i_{d_c}(x)}(n_m, n_c)$ le sous-ensemble de $\mathrm{Conf}(n_m, n_c)$ défini par les contraintes aux nœuds de parité :

$$\mathrm{Conf}_{i_{d_c}(x)}(n_m, n_c) = \left\{ \boldsymbol{\alpha_k} \in \mathrm{Conf}(n_{\boldsymbol{m}}, n_c) : h_{d_c}(x).i_{d_c}(x) + \sum_{c=1}^{d_c\text{-}1} \alpha_c^{(k_c)}(x) = 0 \right\} \quad (38)$$

Un problème peut se poser lorsque l'ensemble $\mathrm{Conf}(n_m, n_c)$ est vide. Si l'ensemble $\mathrm{Conf}(n_m, n_c)$ est vide pour certaines valeurs de $i_{d_c}(x)$, cela peut poser un problème de convergence pour l'algorithme EMS. On peut cependant contourner ce problème en utilisant les sous-ensembles $\mathrm{Conf}_{i_{d_c}(\mathrm{x})}(q,1)$ qui ne sont jamais vides pour n'importe quelle valeur $i_{d_c}(x) \in GF(q)$.

Algorithme EMS :

En utilisant les notations précédentes, on arrive alors à l'algorithme suivant :

E_1 : Mise à jour des nœuds de variables

Pour la mise à jour d'un nœud de variables de degré d_{v} :

$$\overline{U}_{tp} = \overline{L} + \sum_{\mathrm{v}=1, \mathrm{v}\neq t}^{d_{\mathrm{v}}} \overline{\mathrm{V}}_{p\mathrm{v}} \quad t = 1,\ldots,d_{\mathrm{v}} \qquad (39)$$

Ou encore :

$$\overline{U}_{tp}[i_1,\ldots,i_p] = \overline{L}[i_1,\ldots,i_p] + \sum_{\mathrm{v}=1, \mathrm{v}\neq t}^{d_{\mathrm{v}}} \overline{\mathrm{V}}_{p\mathrm{v}}[i_1,\ldots,i_p] \quad (i_1,\ldots,i_p) \in \{0,1\}^p \qquad (40)$$

E_2 : Etape de permutation, depuis les noeuds de variables vers les nœuds de parité

On a :

$$\text{vec}(\overline{U}_{pc}) = P_{h(x)}.\text{vec}(\overline{U}_{\text{vp}}) \quad (41)$$

ou encore :

$$\overline{U}_{pc}[i_1,...,i_p] = \overline{U}_{\text{vp}}[j_1,...,j_p] \quad (i_1,...,i_p) \in \{0,1\}^p$$
$$\text{avec } i(x) = h(x).j(x) \qquad (42)$$

L'étape de permutation depuis les nœuds de parité vers les nouds de variables est réalisée avec la matrice $P_{h(x)}^{-1}$.

E_3 : mise à jour des noeuds de parité

A partir des $d_c - 1$ messages arrivant : $\overline{U}_{pc}$, construire les ensembles :

$S_{i_{d_c}}(x) = \text{Conf}_{i_{d_c}(x)}(q,1) \cup \text{Conf}_{i_{d_c}(x)}(n_m, n_c)$; On obtient alors :

$$\overline{\text{V}}_{d_c p}[i_{d_{c_1}},...,i_{d_{c_p}}] = \max_{\boldsymbol{\alpha_k} \in S_{i_{d_c}}(x)} \{L(\boldsymbol{\alpha_k})\} \quad (i_{d_{c_1}},...,i_{d_{c_p}}) \in \{0,1\}^p \qquad (43)$$

Traitement final :

$$\overline{\text{V}}_{cp}[i_1,...,i_p] = \overline{\text{V}}_{cp}[i_1,...,i_p] - \overline{\text{V}}_{cp}[0,...,0] \qquad (44)$$
$$(i_1,...,i_p) \in \{0,1\}^p \quad c = 1,...,d_c$$

V. RÉSULTATS DE SIMULATION

Dans cette partie nous présentons quelques résultats de simulation pour montrer les performances du système proposé en utilisant l'algorithme EMS logarithmique pour les codes LDPC sur *GF*(*q*). Puisque le but est la comparaison de notre système avec le travail de Kozic & al [52], on utilise ses résultats comme référence pour notre système proposé. Nos matrices

A sont générées avec les mêmes paramètres par exemple, pour n = 512 et n = 1024 comme taille du vecteur binaire d'entrée avec Q = 2 ou 3 (cf Figure 1) et une matrice ***A*** avec un poids de 3 sur chaque colonne, nous avons simulé le système de Kozic en binaire. Pour les systèmes proposés ici sur $GF(q)$ nous prenons, à cause de la complexité de l'algorithme de décodage, une efficacité spectrale limitée avec q = 4 et q = 8 ; c'est-à-dire on travaille avec GF (4) et GF (8) avec des efficacités spectrales respectivement égales à 2 et 3 bit/s/Hz.

Les résultats sont illustrées sur les figures 7 et 8 avec respectivement des tailles de bloc égales à 512 et 1024. On peut remarquer (Figure 7) que les performances de nos codes chaotiques sur $GF(4)$ et $GF(8)$ de taille 512 sont comparables à celles des codes binaires de Kozic & al. Sur la figure 8 il apparaît clairement que nos codes de taille 1024 sont plus performants que les codes binaires de Kozic exhibant la même taille. Pour Q = 2, le gain pour un TEB référence de 10^{-5} est approximativement égal à 0.5 dB pour $GF(8)$ et devient égal à 0.75 dB pour $GF(4)$. L'amélioration des performances est encore plus importante dans le cas Q = 3 ; on obtient sur la figure 8, 1 dB de gain pour $GF(8)$ et 1.5 dB de gain pour $GF(4)$. Les gains de diversité c'est-à-dire les pentes des courbes de TEB à forts SNR's sont identiques entre les différents codes. Ces résultats confirment bien les résultats connus de la littérature sur les codes LDPC à savoir que l'utilisation de corps étendus $GF(q)$ permet de construire des codes plus performants que leurs homologues binaires.

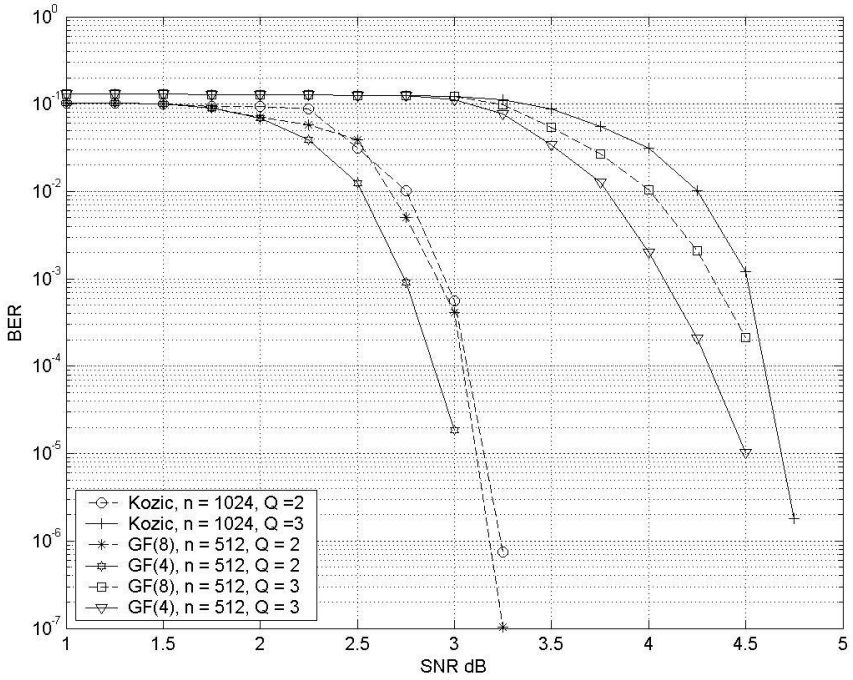


Figure 7 : Résultats de simulation des codes chaotiques sur GF(q) pour des tailles de bloc égales à 512

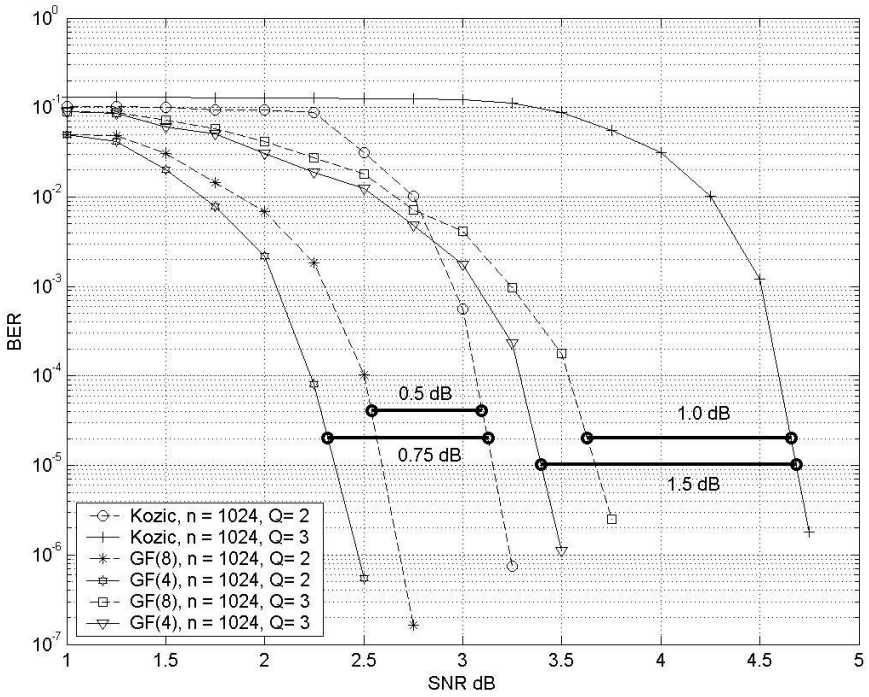


Figure 8 : Résultats de simulation des codes chaotiques sur GF(q) pour des tailles de bloc égales à 1024

VI. CONCLUSION

Nous avons présenté dans ce chapitre un schéma de codage chaotique performant à l'aide d'une fonction de mapping opérant à partir d'une matrice ***A*** de grande taille. Nous avons du définir un algorithme de décodage particulier pour ce système en nous basant sur la théorie des graphes factoriels. Par rapport aux travaux initiaux de Kozic nous avons proposé d'utiliser des matrices ***A*** définies sur un corps de Galois *GF*(*q*). Un algorithme de décodage sous-optimal performant basé sur l'algorithme EMS de Declercq et Fosserier a été mis au point qui permet aux codes synthétisés à partir de mapping chaotique d'obtenir d'excellentes performances et de devenir des alternatives réellement crédibles aux turbo-codes et autres codes LDPC.

VII. ANNEXE : CORPS DE GALOIS

VII.1. Corps De galois : Définition

Un corps de Galois à $q = 2^m$ éléments noté $GF(q) = GF(2^m)$, où m est un entier positif est défini comme une extension polynomiale du corps $GF(2)$ constitué simplement des éléments 0 et 1. Le polynôme $g(x)$ à utiliser pour construire le corps $GF(q)$ doit être irréductible c'est-à-dire non factorisable dans $GF(2)$, de degré m et à coefficients dans $GF(2)$. Les éléments du corps de Galois $GF(q)$ sont définis modulo $g(x)$ et ainsi, chaque élément du corps peut être représenté par un polynôme de degré au plus égal à $(m - 1)$ et à coefficients dans $GF(2)$.

Exemple :
Considérons un polynôme irréductible dans le corps GF(2) de degré m = 2 :

$$g(x) = x^2 + x + 1 \quad (45)$$

Ce polynôme permet de construire un corps de Galois à 4 éléments. Les éléments de ce corps sont de la forme :

$$a\alpha + b \quad \text{où} \quad a, b \in GF(2) \quad (46)$$

soit :

$$\text{GF(4)} : \{0, 1, \alpha, \alpha + 1\} \quad (47)$$

On peut remarquer que si on élève l'élément α aux puissances successives 0,1 et 2, on obtient tous les éléments du corps GF(4) à l'exception de l'élément 0. En effet, le carré de α est encore égal à $(\alpha + 1)$ modulo $\alpha^2 + \alpha + 1$. L'élément α est appelé élément primitif du corps GF(4).

VII.1.1. Elément primitif d'un corps de Galois

On appelle élément primitif d'un corps de Galois $GF(q)$, un élément de ce corps qui, lorsqu'il est élevé aux puissances successives 0, 1, 2,…, $(q - 2)$; $q = 2^m$ permet de retrouver tous les éléments du corps sauf l'élément 0. Tout corps de Galois possède au moins un élément primitif. Si α est un élément primitif du corps $GF(q)$ alors les éléments de ce corps sont :

$$GF(q) : \{0, \alpha^0, \alpha^1, ..., \alpha^{q-2}\} \text{ avec } \alpha^{q-1} = 1 \quad (48)$$

Propriétés :

- Un élément primitif α d'un corps de Galois GF(q) est racine de l'équation $x^n + 1 = 0$ si $n = q - 1$. En effet, si on tient compte du fait que $\alpha^{q-1} = 0$ et que $n = q - 1$ alors $\alpha^n + 1 = 0$ (rappelons que –1 = 1 dans le corps GF(2) ainsi que dans un corps GF(2^m)).
- Si l'élément α est racine de $x^n + 1$ alors α^j est aussi racine de $x^n + 1$. En effet, nous avons $\alpha^n = 1$ et ainsi nous pouvons écrire :

$$(\alpha^n)^j = (1)^j = 1$$

En permutant les exposants n et j, nous obtenons :

$$(\alpha^j)^n = 1 \Leftrightarrow (\alpha^j)^n + 1 = 0$$

Ces deux propriétés peuvent être utilisées pour factoriser le polynôme $x^n + 1$ (n = q – 1) dans un corps de Galois à $q = 2^m$ éléments. En effet, si α est un élément primitif du corps GF(2^m) alors $\alpha, \alpha^2, ..., \alpha^n$ sont aussi des racines de $x^n + 1$ et ainsi nous pouvons écrire :

$$x^n + 1 = \prod_{j=1}^{n} (x + \alpha^j) \quad (49)$$

Exemple :

Factorisons le polynôme $x^3 + 1$ sur le corps $GF(4)$ construit à partir du polynôme $g(x) = x^2 + x + 1$. Nous avons vu que α est un élément primitif du corps $GF(4)$ et ainsi, les 4 éléments de ce corps sont :

$$GF(4) : \{0, 1, \alpha, \alpha^2\} \quad (50)$$

Nous pouvons écrire :

$$x^3 + 1 = \prod_{j=1}^{3} (x + \alpha^j) = (x + \alpha).(x + \alpha^2).(x + \alpha^3) \quad (51)$$

En développant cette expression, on obtient :

$$x^3 + 1 = x^3 + x^2.(\alpha^2 + \alpha + 1) + x.(\alpha^2 + \alpha + 1) + 1 = x^3 + 1 \quad (52)$$

En tenant compte du fait que les opérations dans $GF(4)$ sont faîtes modulo $\alpha^2 + \alpha + 1$, le deuxième terme de l'égalité (52) est bien égal à $x^3 + 1$.

VII.1.2. Polynôme minimal associé à un élément β d'un corps de Galois

VII.1.2.1. Cas des polynômes à coefficients dans *GF*(2)

Le polynôme minimal $m_\beta(x)$ à coefficients dans *GF*(2) associé à un élément quelconque β d'un corps de Galois *GF*(2^m), est un polynôme de degré au plus égal à m, ayant β comme racine. Ce polynôme est unique et irréductible dans *GF*(2) mais deux éléments du corps de Galois peuvent avoir le même polynôme minimal. Si β est un élément primitif du corps de Galois *GF*(2^m) alors le polynôme $m_\beta(x)$ est exactement de degré *m*.

Avant de calculer le polynôme minimal associé à un élément β rappelons qu'un polynôme à coefficients dans *GF*(2) vérifie la propriété suivante :

$$\left[f(x) \right]^2 = f(x^2) \Rightarrow \left[f(x) \right]^{2^p} = f(x^{2^p}) \qquad (53)$$

Ainsi si β est racine du polynôme *f*(*x*) alors $\beta^2, \beta^4, \ldots$ sont aussi des racines de ce polynôme. Si le polynôme *f*(*x*) est de plus irréductible dans *GF*(2) et de degré *m*, alors si β est racine de ce polynôme, les autres racines de ce polynôme sont simplement : $\beta^2, \beta^4, \ldots, \beta^{2^{m-1}}$.

Cette propriété des polynômes à coefficients dans *GF*(2), nous permet d'écrire le polynôme minimal associé à β sous la forme :

$$m_\beta(x) = (x+\beta).(x+\beta^2).(x+\beta^4)\ldots. \qquad (54)$$

ou encore :

$$m_\beta(x) = (x+\beta).(x+\beta^2).(x+\beta^4)\ldots.(x+\beta^{2^{m-1}}) \qquad (55)$$

si β est un élément primitif du corps *GF*(2^m).

Exemple :

Calculons le polynôme minimal associé à l'élément primitif α du corps de Galois *GF*(4).

$$GF(4) : \left\{0, 1, \alpha, \alpha^2\right\}$$

Le polynôme minimal associé à l'élément α a donc pour racine α et α^2 (m = 2) et il a pour expression :

$$m_\alpha(x) = (x+\alpha).(x+\alpha^2) = x^2 + x.(\alpha+\alpha^2) + \alpha^3 \qquad (56)$$

En tenant compte du fait que $\alpha^3 = 1$ et que $\alpha + \alpha^2 = 1$ dans le corps *GF*(4), le polynôme $m_\alpha(x)$ est encore égal à :

$$m_\alpha(x) = x^2 + x + 1 \qquad (57)$$

Pour clore ce paragraphe sur la notion de polynôme minimal, on peut rajouter la propriété suivante :

Soient $\beta_1,...,\beta_i,...,\beta_N$ les éléments d'un corps de Galois *GF*(2^m) et soient $m_{\beta_1}(x),...,m_{\beta_i}(x),...,m_{\beta_N}(x)$ les polynômes minimaux associés respectivement à ces éléments. Alors le polynôme *g*(*x*), de plus bas degré, à coefficients dans *GF*(2) ayant $\beta_1,...,\beta_i,...,\beta_N$ comme racines est égal à :

$$g(x) = \text{P.P.C.M}(m_{\beta_1}(x),...,m_{\beta_i}(x),...,m_{\beta_N}(x)) \qquad (58)$$

où P.P.C.M est l'abréviation de Plus Petit Commun Multiple. Si les polynômes minimaux sont distincts, (ce qui n'est pas toujours le cas puisque deux éléments d'un corps *GF*(2^m) peuvent avoir le même polynôme minimal) alors le polynôme $g(x)$ est simplement égal à :

$$g(x) = \prod_{i=1}^{N} m_{\beta_i}(x) \qquad (59)$$

VII.1.2.2. Cas des polynômes à coefficients dans *GF*(*q*)

Le polynôme minimal $m_\beta(x)$, à coefficients dans le corps de Galois *GF*(*q*) associé à un élément $\beta = \alpha^j$ (α élément primitif du corps *GF*(*q*)) de ce corps, est le polynôme de plus bas degré ayant β comme racine. En utilisant une généralisation de la relation (53) au cas des polynômes à coefficients dans *GF*(*q*), nous pouvons écrire :

$$[\, f(x)\,]^q = f(x^q) \Rightarrow [\, f(x)\,]^{q^p} = f(x^{q^p}) \qquad (60)$$

Ainsi, si β est racine du polynôme *f*(*x*) alors $\beta^q, \beta^{q^2}, ...$ sont aussi des racines de ce polynôme.

En tenant compte du fait que dans le corps *GF*(*q*) $\alpha^{q-1} = 1$, alors $\beta^{q^p} = (\alpha^j)^{q^p} = \alpha^j = \beta$ et ainsi, le polynôme minimal $m_\beta(x)$ est simplement égal à :

$$m_\beta(x) = x + \beta \qquad (61)$$

VII.1.3. 1.3 Polynôme primitif

Un polynôme à coefficients dans *GF*(2) est primitif si il est le polynôme minimal associé à un élément primitif d'un corps de Galois. Dans l'exemple présenté dans le paragraphe 1.2, le polynôme $m_\alpha(x)$ était un polynôme primitif.

Un polynôme primitif est donc irréductible dans *GF*(2) et peut par conséquent être utilisé pour engendrer un corps de Galois. Lorqu'un polynôme primitif est utilisé pour engendrer un corps de Galois, tous les éléments du corps sont obtenus en élevant l'élément primitif, racine du polynôme primitif à des puissances successivement croissantes. Les principaux polynômes primitifs étant répertoriés dans la littérature, la construction d'un corps de Galois à $q = 2^m$ éléments peut alors se faire simplement en utilisant un polynôme primitif de degré *n*.

Exemples de polynômes primitifs

Degré du polynôme	Polynômes primitifs
2	$x^2 + x + 1$
3	$x^3 + x + 1$
4	$x^4 + x + 1$
5	$x^5 + x^2 + 1$
6	$x^6 + x + 1$
7	$x^7 + x^3 + 1$
8	$x^8 + x^4 + x^3 + x^2 + 1$
9	$x^9 + x^4 + 1$
10	$x^{10} + x^3 + 1$

Pour terminer cette partie introductive aux corps de Galois, nous donnons un exemple de corps de Galois à $q = 16$ (m = 4) éléments construit à partir du polynôme primitif $x^4 + x + 1$. Les éléments de ce corps sont :

$$GF(16) : \left\{0, 1, \alpha, \alpha^2, \alpha^3, ..., \alpha^{14}\right\} \qquad (62)$$

A ces 16 éléments on peut associer une représentation polynomiale ainsi qu'une représentation binaire. La représentation polynomiale d'un élément de ce corps est de la forme :

$$a.\alpha^3 + b.\alpha^2 + c.\alpha + d \qquad (63)$$

où *a, b, c d* sont des coefficients binaires appartenant à *GF*(2). Le corps de Galois *GF*(16) étant constitué de 16 éléments, la représentation binaire d'un élément de ce corps est faîte à l'aide de 4 symboles binaires appartenant à *GF*(2). Ces 4 symboles sont respectivement égaux aux valeurs prises par les coefficients *a*, *b*, *c* et *d*.

Eléments du corps *GF*(16)	Représentation polynomiale	Représentation binaire
0	0	0000
1	1	0001
α	α	0010
α^2	α^2	0100
α^3	α^3	1000
α^4	$\alpha+1$	0011
α^5	$\alpha^2+\alpha$	0110
α^6	$\alpha^3+\alpha^2$	1100
α^7	$\alpha^3+\alpha+1$	1011
α^8	α^2+1	0101
α^9	$\alpha^3+\alpha$	1010
α^{10}	$\alpha^2+\alpha+1$	0111
α^{11}	$\alpha^3+\alpha^2+\alpha$	1110
α^{12}	$\alpha^3+\alpha^2+\alpha+1$	1111
α^{13}	$\alpha^3+\alpha^2+1$	1101
α^{14}	α^3+1	1001

Eléments du corps de Galois GF(16)

Dans un corps la somme de deux éléments du corps est un élément du corps. Nous pouvons vérifier cette propriété pour le corps de Galois *GF*(16). Faisons par exemple la somme des éléments α^5 et α^8. En utilisant les représentations polynomiales de ces deux éléments, nous avons à faire la somme de $(\alpha^2+\alpha)$ et de (α^2+1) ce qui donne $(\alpha+1)=\alpha^4$ puisque $2.\alpha^2=0$. Pour faire la somme de α^5 et α^8 on peut aussi utiliser leur représentation binaire. Dans ce cas on doit faire la somme bit à bit de 0110 et de 0101 ce qui donne 0011 soit encore α^4.

Conclusion générale et perspectives

Dans cette thèse nous avons abordé une thématique encore peu explorée dans le domaine des communications numériques à savoir l'utilisation de fonctions génératrices de chaos pour synthétiser des codeurs de canal.

Nous avons, dans une première approche, étudié des systèmes pseudo-chaotiques où nous avons remplacé l'encodage classique linéaire à base de OU exclusif par une structure de filtre numérique à réponse impulsionnelle infinie qui utilise des opérateurs de saturation en sortie des multiplieurs ou des additionneurs pour générer le chaos. Cette structure a été étudiée plus en détails dans la thèse de Mr Alan Layec. Cependant, nous avons conservé en sortie un mapping conventionnel (Gray par exemple) ce qui fait que la constellation transmise est exactement celle d'une modulation MAQ- N. Nous avons testé la structure du codeur dans différents contextes et nous l'avons modifié en rajoutant une sortie systématique pour pouvoir obtenir la structure d'un turbo-code à concaténation parallèle. Nous avons montré que par optimisation de la distance libre il était possible de choisir des paramètres du filtre à réponse impulsionnelle infinie tels que le codeur obtenu présente de bonnes performances y compris à forts SNR's en évitant les effets de plancher. Enfin, l'utilisation de ce codeur dans le cadre d'une transmission sur des canaux sélectifs en fréquence a été testée et les performances quantifiées. Pour l'égalisation, nous avons utilisé un passage dans le domaine fréquentiel ce qui permet d'obtenir des structures à faible complexité.

Cependant, il apparaît clairement que pour obtenir un système réellement chaotique il faut que la constellation de sortie soit associée au processus de génération de chaos. En effet, il faut que la constellation de sortie présente toutes les caractéristiques d'un signal imprévisible avec par exemple une distribution pour la densité de probabilité des signaux qui s'apparente à celle d'une loi uniforme. C'est ainsi que nous nous sommes intéressés dans le chapitre 2 à la réalisation de schémas d'encodage-modulation qui utilisent des fonctions génératrices de chaos comme la fonction Bernoulli Shift Map. Les fonctions testées dans ce chapitre sont toutes mono-dimensionnelles. Nous avons proposé différents algorithmes de décodage pour essayer de tirer au mieux parti de la corrélation présente entre les échantillons successifs transmis sur le canal. Les performances ont été testées sur canal à bruit additif Gaussien. Nous avons constaté pour les différentes fonctions d'encodage-mapping testées qu'il est difficile voire impossible avec un mapping mono-dimensionnel d'obtenir de meilleurs résultats qu'un système MDP2-NRZ non codé, ce qui est bien sûr un objectif minimal pour tout codeur de canal.

Dans le troisième chapitre nous avons généralisé les fonctions de mapping mono-dimensionnelles du chapitre 2 à des mapping multi-dimensionnels faisant appel à des

multiplications matricielles associées à des opérateurs modulo. C'est à partir de ce chapitre qu'apparaissent les contributions originales de ce travail de thèse. L'augmentation de la dimension du mapping peut être interprétée avec beaucoup de précautions comme étant similaire à la diminution du rendement d'un code avec l'augmentation de la part de redondance. Cependant, l'augmentation de la dimension nécessite une optimisation plus poussée des codeurs obtenus. Nous avons proposé en nous basant sur une modélisation approximative du spectre du code à base de mélange de lois Gaussiennes ou de Rayleigh une façon d'optimiser la recherche des paramètres des matrices d'encodage-mapping. Les performances ont été alors quantifiées de façon théorique sur des canaux à bruit additif Gaussien mais également sur des canaux quasi-statiques à évanouissements par blocs non-sélectifs en fréquence. Les résultats de simulation ont bien confirmé les calculs théoriques justifiant la modélisation choisie pour la distribution du spectre des distances. Cette approche originale vient compléter les travaux de Kozic dans le domaine. Cependant, il faut signaler que les structures de codes employés ici s'apparentent à celles de filtres à réponse impulsionnelle finie. C'est ainsi qu'une des conditions énoncées par Frey dans son article fondateur en 1993, à savoir qu'une suite finie en entrée provoque une sortie la plus étendue possible dans le domaine temporel, n'est pas satisfaite. Il faudrait donc, et c'est une perspective de recherche à court terme, généraliser les structures de codeurs proposées dans ce chapitre au cas des filtres à réponse impulsionnelle infinie. Cependant, les premiers essais effectués montrent que ces dernières structures exhibent des distances libres inférieures à leurs homologues à réponse finie.

L'amélioration des performances obtenue avec des mapping multi-dimensionnels d'ordre 2 ou 3 s'avère cependant assez marginale. Pour obtenir une nette amélioration des performances il faut utiliser des matrices de grande taille pour la génération de chaos. C'est ainsi que nous proposons d'utiliser dans ce chapitre des matrices creuses de type LDPC pour la génération des produits matriciels. Comparé aux travaux de Kozic, nous avons utilisé des matrices génératrices de codes LDPC sur des corps de base non-binaires ($GF(q)$). L'algorithme de décodage par propagation de croyance proposé se base sur un graphe factoriel qui tient compte de l'ensemble de la chaîne d'encodage-modulation ou d'encodage-mapping et qui permet de conserver une complexité raisonnable. Nous avons utilisé l'algorithme EMS (Extended Min-Sum) proposé récemment par Fossorier et Declercq pour le décodage des codes LDPC sur $GF(q)$. Les performances obtenues deviennent tout à fait satisfaisantes à un point tel qu'il devient raisonnable d'envisager ces codes comme de réelles alternatives à des solutions très performantes comme les turbo-codes ou les codes LDPC. Les perspectives de ce

chapitre concernent d'abord l'étude du seuil de déclenchement de l'effet cascade (Water Fall region) de ces codes à l'aide des courbes d'EXIT CHARTS. Ce travail est en train d'être effectué. Une autre perspective concerne évidemment l'implantation hardware de tels codes avec une évaluation de la complexité matérielle nécessaire à leur réalisation.

Bibliographie

[1]. O. E. Rössler**, "**An equation for continuous chaos**"**. *Physics Letters A* Volume 57, Issue 5, Pages 397-398, 1976.

[2]. E.N. Lorenz, **"**Deterministic non periodic flow**"**. *Proc. J. atmospheric science*, 20, pages 130-141, 1963.

[3]. H-O Peitgen, H. Jürgens, D. Saupe, **"**Chaos and Fractals**"** *New Frontiers of science. Springer*, February 1993.

[4]. A. Serbanescu. **"**Electronique, physique et signal pour les telecommunications**"**, *chapter Systèmes et signaux face au chaos*. Ed. Tehnica, 1997.

[5]. R. L. Kraft. **"**Chaos, cantor sets, and hyperbolicity for the logistic maps**"**. *Am. Math. Mon.*, 106:400–408, 1999.

[6]. M. Tabor. **"**Chaos and Integrability in Nonlinear Dynamics**"** *An Introduction*. Wiley, 1989.

[7]. J Fröhlich; S. P. Novikov and D. Ruelle. **"**Dynamical Systems, Ergodic Theory and Applications**"**. *Springer Verlag*, 2000

[8]. J. C. Sprott. **"**Chaos and time-series analysis**"**. *Oxford University Press*, 2003

[9]. W. Schwarz T. Schimming, M. Götz. **"**Signal modelling using piecewise linear chaotic generators**"**. *Proceedings of the EUSIPCO-98*, 3:1377–1380, 1998

[10]. R. Rovatti, G. Mazzini, G. Setti, and A. Giovanardi. **"**Statistical modeling and design of discrete-time chaotic processes**",** *advanced finite-dimensional tools and applications*. 90(5):820–841, 2002

[11]. C. Grebogi E. Bollt, Y. C. Lai. **"**Channel capacity and noise resistance in communicating with chaos**"**. *Physical review letters*, 79, n°19:pp. 3787–3790, 1997.

[12]. R. Rovatti M. P. Kennedy and G. Setti. **"**Chaotic Electronics in telecommunications**"**. *CRC Press, Inc. Boca Raton*, FL, USA, 2000.

[13]. C. Grebogi S. Hayes and E. Ott. **"**Communicating with chaos**"**. *Phys. Rev. Lett*, 70:3031–3034, 1993.

[14]. F. C. M. Lau and C. K. Tse. **"**Chaos-Based Digital Communication Systems**"**. *Springer*, 2003.

[15]. J-M Liu L. E. Larson and L. S. Tsimring. **"**Digital Communications Using Chaos and Nonlinear Dynamics**"**. *Springer*, 2006.

[16]. Blackledge Jonathan M Blackledge. **"**Cryptanalysis: Chaos and Fractals**"**. *Horwood Publishing Limited*, 2000

[17]. A. Abel and W. Schwarz. **"***Chaos communications-principles, schemes, and system analysis***"**. 90(5):691–710, 2002

[18]. A. I. Panas A. S. Dmitriev, M. Hasler and K. V. Zakharchenko. **"**Basic principles of direct chaotic communications**"**. *Nonlinear Phenomena in Complex Systems*, 6:1–14, 2003

[19]. S. A. Barbulescu, A. Guidi, S. S. Pietrobon, **"**Chaotic turbo codes**"**, *IEEE Conf. International Symposium on Information Theory (ISIT)*, pp. 123, June 2000

[20]. A.M. Guidi, **"**Turbo and LDPC Coding for the AWGN and Space-Time Channel**"**, *PhD Thesis, University of South Australia*, June 2006.

[21]. A.Layec. **"**Développement de modèles de CAO pour la simulation système des systèmes de communication. Application aux communications chaotiques**"**. *PhD thesis, Xlim University of Limoges*, 2006

[22]. Vladeanu, S. El Assad, J.C. Carlach, R. Quere, **"**Improved Frey Chaotic Digital Encoder for Tellis-Coded Modulation**",** *IEEE Trans. on Circuits and Systems II: express Briefs*, Vol. 56, Issue 6, pp. 509-513, June 2009

[23]. Vladeanu, S. El Assad, J.C. Carlach, R. Quere, **"**Chaotic Digital Encoding for 2D Trellis-Coded Modulation**"**, *5th Advanced International Conference on Telecommunications*, Venice, May 2008

[24]. Vladeanu, S. El Assad, J.C. Carlach, R. Quere, **"**Recursive GF(2N) encoders using left-circulate function for optimum PSK-TCM schemes**"**, *Signal Processing*, Vol. 90, Issue 9, pp. 2708-2713, September 2010

[25]. B. Vucetic, **"**Iterative decoding algorithm**"**, *in PIMRC'97*, Helsinki, Finland, Sep. 1997, pp.99-120

[26]. L. Bahl, J. Cocke, F. Jelinek and J. Raviv, **"**Optimal decoding of linear codes for minimizing symbol error rate**"**, *IEEE Trans. Inform. Theory*, vol. IT-20, pp; 284-287, March 1979

[27]. P. Robertson, and T. Woerz, **"**Novel coded modulation scheme employing turbo codes**"**, *Electronics Letters*, vol. 31, n° 18, Aug. 1995, pp. 1546-1547

[28]. R. Frey. **"**Chaotic digital encoding: an approach to secure communication**"**. 40(10):660–666, 1993.

[29]. G.Mazzini, R. Rovatti, and G. Setti. **"**Capacity of chaos-based asynchronous ds-cdma systems with exponentially vanishing autocorrelations**"**. *Electronics Letters*, 38(25):1717–1718, 2002

[30]. G.Setti, R. Rovatti, G. Mazzini, **"**Performance of chaos-based asynchronous DS-CDMA with different pulse shapes**"**, *IEEE Communications Letters*, Vol. 8, n° 7, pp. 416-418, 2004

[31]. G.Mazzini, G. Setti, R. Rovatti, **"**Chaotic complex spreading sequences for asynchronous DS-CDMA, Part I: System modelling and results**",** *IEEE Trans. Circuits Syst*. II, vol. 44, no 10, pp. 937-947, Oct. 1997.

[32]. R. Rovatti ,G. Mazzini, G. Setti, **"**Chaotic complex spreading sequences for asynchronous DS-CDMA, Part II: some theoretical performance bounds**"**, *IEEE Trans. Circuits Syst. II*, vol. 45, no 4, pp. 496-506, Apr. 1998.

[33]. F. J. Escribano, L. Lopez, M. A. F. Sanjuan, **"**Evaluation of channel coding and decoding algorithms using discrete chaotic maps**"**, *Chaos, American Institute of Physics*, Issue 16, 1054-1500/2006/16(1)/013103/12.

[34]. M. A. F. Sanjuan I. P. Marino, L. Lopez. **"**Channel coding in communications using chaos**"**. *Physics letters*. A, 295, n°4:185–191, 2002.

[35]. E. Bollt, Y. C. Lai, C. Grebogi, **"**Channel capacity and noise resistance in communicating with chaos**"**, *Physical review letters American Physical Society*, Vol. 79, n° 19, pp. 3787-3790, 1997.

[36]. T. Schimming and M. Hasler, **"**Coded Modulation based on Controlled 1-D and 2-D piecewise linear chaotic maps**"**, *IEEE Conf. International Symposium on Circuit and Systems (ISCAS),* pp. 762-765, May 2003.

[37]. B. Chen and G. W. Wornel, **"**Analog Error Correcting Codes based on Chaotica Dynamical Systems**"**, *IEEE Trans. Comm*., Vol. 46, pp. 881-890, 1998.

[38]. Baranovsky, A. and Daems, D., **"**Design of One-Dimensional Chaotic Maps with Prescribed Statistical Properties**"**, *Int. J. Bifurcation Chaos 5*, 1585 (1995).

[39]. Schimming, T. and Hasler, M., **"**Coded Modulations Based on Controlled 1-D and 2-D Piecewise Linear Chaotic Maps**",** *in Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS) 2003*, Vol. 3, pp 762–765, Bangkok, Thayland (2003).

[40]. M. S. Baptista and L. Lopez*, Phys. Rev*. E 65, 055201 (2002).

[41]. S. Kozic, K. Oshima and T. Schimming, **"**Nonlinear Dynamics of Electronic Systems 2003, NDES 2003**"**, *IEEE International Workshop*, Scuol, Switzerland, 2003, pp. 141-144.

[42]. M. P. Kennedy, R. Rovatti and G. Setti, **"**Chaotic Electronics in Telecommunications**"** (CRC Press, Boca Raton 2000).

[43]. F. J. Escribano, L. López, and M. A. F. Sanjuán, **"**Chaos-Coded Modulations over Rician and Rayleigh Flat Fading Channels**"**, *IEEE Transactions on Circuits and Systems—II: Express Briefs,* vol. 55, n°. 6, june 2008, pp. 581-585

[44]. Scott Hayes. **"**Communication with Chaos**"**. *Physical Review Letters*, 70:3031–3034, 1993.

[45]. Z. Galias and G.M. Maggio. **"**On the Optimal Labeling for Pseudo-chaotic Phase Hopping**"**. *In International Symposium on Circuit and Systems ISCAS*, pages 883–886, May 2002.

[46]. S. Kozic, **"**Channel coding and modulation based on chaotic systems**"**, *PhD Thesis EPFL*, no 3634 (2006).

[47]. S. M. Alamouti, **"**A simple transmit diversity technique for wireless communications**"**, *IEEE Journal on Selected Areas in Communications*, Vol. 16, Issue 8, pp:1451-1458, Oct. 1998.

[48]. J. He, P. Y. Kam, **"**Bit Error Performance of Orthogonal Space-Time Block Codes over Time-Selective Channel**"**, *Proceedings of IEEE International Conference on Communications, 2007*, *ICC '07*, pp: 4604-4609, June 2007.

[49]. B. Lu, Y. Guosen, X. Wang, **"**Performance analysis and design optimization of LDPC-coded MIMO OFDM systems**"**, *IEEE Trans. on Signal Processing*, Vol. 52, Issue 2, pp: 348-361, Feb. 2004

[50]. J. A. Fessler and A. O. Hero, **"**Space-alternating generalized expectation maximization algorithm**",** *IEEE Trans. on Sig. Process*, vol. 42, n°10, pp. 2664-2677, Oct. 94.

[51]. C. N. Georghiades, and J. C. Han, **"**Sequence estimation in the presence of random parameters via the EM algorithm**"**, *IEEE Trans. on Comm.*, vol. 45, n°3, pp. 300-308, March 97.

[52]. S Kozic, M. Hasler, **"**Low-Density Codes Based on Chaotic Systems for Simple Encoding**"**, *IEEE Transactions on Circuits and Systems* I: Vol. 56, Issue 2, pp. 405-415, Feb. 2009.

[53]. R. Tanner. **"**A recursive approach to low complexity codes**"**. *IEEE Transactions on Information Theory*, 27(5) :533–547, Sept. 1981.

[54]. D Declercq, M Fossorier, **"**Decoding Algorithms for Nonbinary LDPC codes Over GF(*q*) **"**, *IEEE Trans. Inform. Theory*, April 2007, pp. 633-643.

[55]. C. Poulliat, M. Fossorier and D. Declercq, **"**Design of regular (2,dc)-LDPC codes over GF(q) using their binary images**"**, *in IEEE Trans. Commun.*, vol. 56(10), pp. 1626 - 1635, October 2008.

[56]. L. Barnault and D. Declercq. **"**Fast Decoding Algorithm for LDPC over GF(2*q*)**"**. In *ITW'03*,Paris, France, 2003.

[57]. A. Voicila, D. Declercq, F. Verdier, M.P. Fossorier, P. Urard *,* "Low Complexity, low memory EMS algorithm for non-binary LDPC codes" , *ICC* 2007 - June 2007

[58]. A. Voicila, D. Declercq, M.P. Fossorier, F. Verdier, P. Urard*,* "Décodage des codes LDPC non-binaires : un algorithme à très faible complexité", *Gretsi* 2007 - September 2007 .

Listes des publications

▪ *Publications dans des revues internationales*

- Naim Khodor, Jean-pierre Cances, Vahid Meghdadi and Raymond Quere, "Performances of Chaos Coded Modulation Schemes Based on Mod-MAP Mapping and High Dimensional LDPC Based Mod-MAP Mapping with Belief Propagation," *International Journal of Communications, Network and System Sciences, Volume 3, Number 6 (Jun. 2010), pp. 495-577.*

- Naim Khodor, Jean-pierre Cances, Vahid Meghdadi and Raymond Quere, «Performances of Chaos Coded Modulation concatenated with Alamouti's Space-Time Block Code", *Annals of Telecommunications* (9 April 2011), pp. 1-29.

▪ *Publications dans les congrès internationaux*

- Naim Khodor, Jean-pierre Cances, Vahid Meghdadi and Raymond Quere, " Study of the Distance Spectrum of Chaos-Coded Modulation," *in SOFTCOM2009, 17th International Conference on Software, Telecommunications and Computer Networks*, September 24-26 2009, Hvar, Croatia.

- Naim Khodor, Jean-pierre Cances, Vahid Meghdadi and Raymond Quere, "Performances of Chaos Coded Modulation schemes based on High Dimensional LDPC Mod-MAP mapping with Belief Propagation Decoding," *in CSNDSP2010, 7th IEEE & IET International Symposium on Communication Systems, Networks and Digital Signal Processing*, Northumbria University, Newcastle upon Tyne, United Kingdom, July 21-23 2010.

- Naim Khodor, J.P. Cances, V. Meghdadi and R.Quere, "Theoretical expression of error event probability for a Trellis Chaos Coded Modulation concatenated with Space-Time Block Code" *in EUSIPCO2010, European Signal Processing Conference* August 23-27 2010, Aalborg, Denmark.

▪ *Publications dans les congrès nationaux*

- Naim Khodor, Jean-pierre Cances, Vahid Meghdadi and Raymond Quere,"Performances de codeurs de canal chaotiques concaténés avec des codes STBC", *GRETSI 2009, 8-11 Sep 2009.*

APPLICATION DES FONCTIONS GENERATRICES DE CHAOS A LA REALISATION DE CODEURS DE CANAL.

Résumé :

Dans ce mémoire de thèse nous présentons une utilisation originale des fonctions génératrices de chaos à savoir leur emploi en tant que codeur de canal. Après avoir mis au point des structures semi-chaotiques dans lesquelles le mapping ne faisait pas partie de la génération de chaos, nous avons associé le processus de génération de chaos au mapping de sortie de l'encodeur. Les performances obtenues lorsque le mapping de sortie fait appel à des matrices génératrices de codes LDPC de grande taille sont excellentes et permettent d'envisager ces schémas d'encodage CCM (Chaos Coded Modulation) comme des alternatives crédibles à des turbo-codes ou à des codes LDPC. Des algorithmes originaux de caractérisation du spectre des distances de ces schémas CCM sont également proposés dans cette thèse.

APPLICATION OF GENERATING FUNCTIONS OF CHAOS TO THE REALIZATION OF CHANNEL ENCODERS.

Abstract:

In this work we present our contribution to the study of channel coding using chaos based generating functions. We have at first proposed to use chaotic function just as binary generators without taking into account the output mapping in the chaos process. Then, we include the chaos based mono-dimensional generator functions such as the Bernoulli Shift Map into the mapping process. The poor performances we obtained forced us to work with multi-dimensional chaos based generator functions. Using high dimensional LDPC based matrices for the joint channel-coding and mapping function enables us to obtain outstanding performances. In this case the obtained schemes constitute good alternative schemes for powerful channel coders such as turbo-codes or LDPC codes.

Discipline : "Electronique des Hautes Fréquences, Photonique et Systèmes"

Mots clés :

Codage correcteur d'erreur chaotique
Modulation chaotique
spectre de distance
mélanges gaussiennes
mélanges du Rayleigh
Code temps-espace en bloc
Chaos coded modulation
LDPC Codes GF(q)
Iterative decoding
Extended Min-Sum (EMS).

Adresse du laboratoire : XLIM, Département C2S2., Faculté des Sciences et Techniques – Université de Limoges, 123 avenue Albert Thomas – 87060 Limoges Cedex