الجـــــــــــــــــــــــمـهـوريـــــة الجـــــــــــــــزائـريـة الديمـــــــقـراطيـة

الـشـــــــــــعبيـة

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التـــــــــعـليـم العـــــــالي و البـــــــــحث العـــــلـمـــي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جـــامعة الدكتـــور الطاهـــــر مـــولاي سعيـــــــدة–

Université Saida Dr Tahar Moulay –

Faculté de TECHNOLOGIE

MEMOIRE

Projet de recherche présenté pour l'obtention du Diplôme de MASTER

En : Télécommunications

Spécialité : Systèmes et Télécommunications

Par : ACHOUR Chahinaz et CHABANE Nawel

Sujet

Les turbo Décodeurs TBCJR-LOG MAP

Soutenue publiquement le …, devant le jury composé de :

Mr. TAMI Abdelkader	MCA	Univ. Saida	Président
Dr. A.Ouardi	MCA	Univ. Saida	Rapporteur
Mr. OUESSAI Asmaa	MCB	Univ. Saida	Examinateur

Année universitaire 2022/2023

Remercîment

Nous tenons à remercier tout d'abord « Allah» le tout Puissant qui nous a donné durant toutes ces années la santé, le courage et la foi.

Le première personne que nous tenons à remercier est notre encadrant Dr.Ouardi.A, pour l'orientation, la confiance, la patience qui ont constitué un apport considérable sans lequel ce travail n'aurait pas pu être mené au bon port. Qu'il trouve dans ce travail un hommage vivant à sa haute personnalité.

Nos vifs remerciements vont également aux membres du jury pour l'intérêt qu'ils ont porté à notre recherche en acceptant d'examiner notre travail et de l'enrichir par leurs propositions.

Enfin, nous tenons également à remercier toutes les personnes qui ont participé de près ou de loin à la réalisation de ce travail.

Dédicace

Je dédie ce modeste travail en signe de respect et de reconnaissance À ma mère, la plus formidable des mamans. À mon père qui m'a éclairé mon chemin.

Á ma très chère mère :

Quoi je fasse ou que je dise, je ne saurai point te remercier comme il se doit. Ton affection me couvre, ta bienveillance me guide et ta présence à mes côtés a toujours été ma source de force pour affronter les différents obstacles.

Á mon très cher père :

Tu as toujours été à mes côtés pour me soutenir et m'encourager. Que ce travail traduit ma gratitude et mon affection.

A grand merci À mes frères .Enfin je n'oublierai pas toutes les personnes, qui par leurs actions, gestes, paroles ou écoutes m'ont soutenu tout au long de ce chemin.

Je tiens à adresser de vifs et sincères remerciements à Professeur "Chetioui. Mohammed "Qui était comme un père Pendant ces années il n'a cessé de me guider, de m'aider et de m'encourager. Je le remercie de tout cela ainsi que du Soutien.

ACHOUR Chahinaz

Dédicace

Je dédie ce modeste travail

A la personne qui m'a appris à quel point la patience est un chemin vers le succès.... soutien et force... et que j'ai souhaité être à mes côtés pendant cette période.

Mon père, qui a un esprit pur et pur âme, que Dieu lui fasse miséricorde.

A celle dont la satisfaction est mon but et mon ambition... A celle qui a éclairé mon chemin par des prières et des supplications.

La plus formidable des mamans, que Dieu prolonge sa vie.

A ceux qui m'ont soutenu moralement et émotionnellement.

Mes frères Ibrahim et Mohamed, que Dieu éclaire leur chemin.

A celui qui m'a soutenu tout au long de ces années et ne m'a pas quitté malgré toutes les difficultés, et dont je souhaitais qu'il soit mon compagnon en ce moment.

Ma meilleure amie BEY Meriem Ikram.

CHABANE Nawel

Résumé :

L'objectif de ce mémoire est d'explorer une technique de codage convolutionnel avec décodage itératif avec entrelacement afin de proposer des solutions simples aux problèmes de la complexité et de la latence du codage et du décodage Turbo conventionnel.

L'algorithme MAP était le premier algorithme utilisé dans un processus décodage itératif. Cet algorithme est optimal en termes de minimiser le BER, contrairement à l'algorithme de Viterbi, qui minimise la probabilité d'un chemin incorrect à travers le treillis sélectionné par le décodeur. Pour réduire la complexité de l'algorithme MAP, deux approximations de ce même algorithme connus sous les noms de Max-Log-MAP et Log-MAP sont utilisées. Ceci constitue la première tentative de réduction de la complexité. La proposition la plus importante de ce modeste travail, est le couplage des deux algorithmes T-BCJR et Log-MAP pour obtenir un turbo décodeur à faible complexité. Nous avons appelé le turbo décodeur obtenu T-Log-MAP.

Mots Clés :

Turbo-codes, Codage convolutif, concaténation, entrelacement, L'algorithme MAP, L'algorithme Log-MAP, Techniques d'approximation, LLR, L'algorithme T-BCJR.

Abstract:

The objective of this thesis is to explore a convolutional coding technique with iterative decoding with interleaving in order to propose simple solutions to the problems of complexity and latency of conventional Turbo coding and decoding.

The MAP algorithm was the first algorithm used in an iterative decoding process. This algorithm is optimal in terms of minimizing the BER, unlike the Viterbi algorithm, which minimizes the probability of an incorrect path through the trellis selected by the decoder. To reduce the complexity of the MAP algorithm, two approximations of the same algorithm known as Max-Log-MAP and Log-MAP are used. This is the first attempt to reduce complexity. The most important proposal of this modest work is the coupling of the two algorithms T-BCJR and Log-MAP to obtain a low complexity turbo decoder. We called the resulting turbo decoder T-Log-MAP.

Keywords:

Turbo-codes, Convolutional coding, concatenation, interlacing, The MAP algorithm, The Log-MAP algorithm, Approximation techniques, LLR, The T-BCJR algorithm.

Liste des figures :

Listes des acronymes et abréviation :

AWGN	Additive White Gaussian Noise
BCJR	Bahl Cocke Jelinek Raviv
LDPC	Low-Density Parity-Check
LLR	Logarithm of Likelihood Ritio
Log-MAP	Logarithmic Maximum A posteriori Probability
LUT	Look-Up Table
MAP	Maximum A Posteriori
ML	Maximum Likelihood
RSC	Codes Systématiques Récursifs
SNR	Signal-to-Noise Ratio
SOVA	Soft-Output Viterbi Algorithm
T–BCJR	Treshold- Bahl-Cocke-Jelinek and Raviv

Sommaire:

Chapitre I : Codage canal

Chapitre II: Algorithmes de décodage pour les turbo codes

Chapitre III : Algorithmes à complexité réduite

Chapitre IV : résultats de Simulation

Introduction Générale

INTRODUCTION GENERALE :

La communication numérique mobile est en forte expansion. Ainsi, le codage de canal est devenu un élément indispensable puisqu'il permet d'assurer la protection de l'information transmise, en contrôlant les erreurs de transmission. Le codage convolutif est l'une des techniques principales du codage de canal. Les unités de codage et de décodage de canal font partie d'un système complexe, dont le but est d'assurer une transmission fiable des données. L'objectif principal du codage de canal est le contrôle des erreurs de transmission des symboles, c'est-à-dire, les erreurs causées principalement par les perturbations du canal et du récepteur.

Dans un système de communications, un système de codage ou un codeur ne peut pas fonctionner seul. Nous avons donc besoin d'un décodeur d'où la nécessite d'algorithmes de décodage complexes. Des différents algorithmes de décodage itératif des turbo-codes ont été proposés. On peut citer les algorithmes SOVA, MAP, Log-MAP et Max-Log-MAP. Il est à noter que toutes ces approches sont basées sur le calcul des probabilités des symboles transmis ou un rapport des Log-Rapport de Vraisemblance LRV (ou LLR Log-Likelihood Ratio).

Le but principal de ce travail est de réduire la complexité calculatoire des turbos décodeurs qui se basent sur le treillis du codeur. L'algorithme de réduction choisi est le TBCJR. Il est couplé à l'algorithme le plus répandu en pratique : L'algorithme Log-MAP. Nous avons appelé le nouveau récepteur obtenu : Turbo décodeur T-Log-MAP.

Le manuscrit est organisé en trois chapitres :

Le premier chapitre : est consacré à quelques généralités relatives au codage canal, des codes convolutifs et les turbo codes. Puis une petite description de l'algorithme de Viterbi est présentée.

Le deuxième chapitre : Porte sur le turbo décodage à faible complexité, les algorithmes de décodage proposés, plus particulièrement, les deux approximations Max-Log-MAP et Log-MAP.

Le troisième chapitre : porte sur les algorithmes à complexité réduite, les Algorithmes de décodage proposés sont : M-BCJR, T-BCJR, et Z-MAP.

Le dernier chapitre : expose les résultats de simulation et analyse les performances des turbo décodeurs utilisant les algorithmes à faible complexité.

Enfin, une conclusion générale sera donnée.

CHAPITR I

Codage canal

I.1.Introduction :

Près de la limite de Shannon, les performances des codes turbo ont attiré de nombreux chercheurs. Les principes de turbo codage et de décodage itératif ont trouvé des applications pratiques justes après leur découverte. Aujourd'hui, les turbo codes convolutifs, les codes LDPC (*Low Density Parity Check*) ou encore les codes polaires permettent de transmettre des signaux dont l'entropie approche la capacité du canal tout en restant décodables en temps réel.

Dans ce chapitre, on va voir le théorème et les Limites de Shannon, la Concaténation, l'entrelacement, ainsi que Les turbo codes et leur décodage.

I.2.Théorème de Shannon pour le codage de canal :

Théorème de Shannon de la capacité d'un canal de transmission : On peut transmettre l'information de façon fiable tant que le débit ne dépasse pas la capacité du canal. Le bruit présent dans le canal ne limite pas la qualité de la communication, mais uniquement le débit de transmission.

On définit le rendement (ou taux) d'un code binaire de cardinal *M* constitué par des mots de longueur *n* par :

$$R = \frac{\log_2 M}{n} \tag{I.1}$$

Un code aura un taux 1 lorsque M = 2^n c'est-à-dire lorsqu'aucune redondance n'aura été ajoutée. Un code de taux $1/2$ signifie que M = $2^{n/2}$, , c'est-à-dire que le code double la longueur des séquences binaires de la source [1].

Théorème : Soit C la capacité d'un canal. Si R $<$ C alors il existe un codage qui permet d'avoir une probabilité d'erreur de décodage évanescente lorsque n tend vers l'infini. Inversement, s'il existe un code dont la probabilité d'erreur de décodage est évanescente alors nécessairement R $\leq$ C [2].

Ce théorème comporte deux résultats. Le premier résultat du théorème de codage est appelé atteignabilité car on dit qu'un rendement de codage est atteignable lorsque la probabilité de décodage est aussi petite que l'on veut pour n suffisamment grand. Le second est appelé

réciproque et affirme qu'il ne peut y avoir de codage ayant un rendement supérieur à la capacité ayant une probabilité de décodage évanescente avec n.

Le second théorème de Shannon prédit l'existence de codes dont le rendement peut valoir la capacité, a priori différente de zéro, et qui pourtant produisent une probabilité d'erreur de décodage évanescente. Le théorème affirme donc bien que pour un degré de redondance limité il est possible d'éviter toute ambiguïté de décodage en réception.

I.3. Capacité :

Pour tout canal de transmission on définit une grandeur caractéristique appelée ***capacité du canal*** et que l'on peut interpréter comme la quantité maximale d'information pouvant transiter à travers le canal.

Dans le cadre de ce émémoire, seuls sont considérés des canaux de transmission causaux sans effet mémoire et stationnaires. En d'autres termes, la sortie du canal à l'instant $\boldsymbol{t}$ ne dépend que de son entrée en l'instant $\boldsymbol{t}$.

• La capacité d'un canal est définie par l'information mutuelle maximale entre la source X à valeurs sur l'alphabet d'entré du canal, et sa sortie correspondante Y sur le canal.

$$\mathrm{C} = Max\ [I(X;Y)] = 1 - \mathrm{H}\ (X/Y) \qquad \text{(I.2)}$$

$I(X;Y)$ Représente l'information transmise par le canal. Elle mesure aussi la dégradation de l'information transmise à travers le canal. La capacité représente la quantité d'information maximale que le canal peut transporter et son unité est le Shannon par symbole. D'après l'équation I.2, afin d'obtenir une communication efficace, il est nécessaire que H (X/Y) soit aussi petit que possible. Or, ce terme, dépend du canal. Plus le canal est bruité, plus ce terme est grand. Le message source doit alors être modifié pour que la communication satisfasse une contrainte de qualité. Ceci constitue le codage canal. Son principe est d'ajouter de la redondance en quantité suffisante pour éviter toute ambiguïté lors du décodage.

I.4.Limites de Shannon :

Pour un canal AWGN, la capacité est donnée par :

$$\mathrm{C} = \frac{1}{2}\log_2\ (1 + \frac{P}{N}) \qquad \text{(I.3)}$$

$\frac{P}{N}$ Est le rapport signal sur bruit.

La variance du bruit est donnée par

$$\sigma^2 = N_0/2 \tag{I.4}$$

Avec N_0 , la densité spectrale du bruit et comme $P = R \times E_b$(E_b est l'énergie moyenne par bit d'information), nous pouvons écrire :

$$\mathrm{C} = \frac{1}{2}\log_2\left(1 + \frac{2RE_b}{N_0}\right) \tag{I.5}$$

Selon le théorème de Shannon, des communications fiables sont possibles sur un canal de transmission si et seulement si $\mathrm{R} < \mathrm{C}$. Ainsi, $2\mathrm{R} < \log_2\left(1 + \frac{2RE_b}{N_0}\right)$, ce qui donne :

$$\frac{2^{2R}-1}{2R} < \frac{E_b}{N_0} \tag{I.6}$$

En faisant tendre ***R*** vers 0, nous avons

$$\frac{E_b}{N_0} > \ln 2 = -1{,}59\mathrm{dB} \tag{I.7}$$

En considérant un canal AWGN, aucun système ne peut transmettre de l'information de manière fiable pour un rapport signal à bruit inférieur à −1,6 dB [3].

I.5.Les codes convolutifs :

I.5.a Définition :

Les codes convolutifs en 1955 Elias inventé les codes convolutifs. Ils forment une classe extrêmement souple et efficace de codes correcteurs d'erreurs. Ce sont les codes les plus utilisés dans les systèmes de télécommunications fixes et mobiles, où une faible probabilité d'erreur par bit est requise. Théoriquement, ils ont les mêmes caractéristiques que les codes en blocs sauf pour la valeur de leur dimension et leur longueur. Les codes convolutifs s'appliquent sur des séquences infinies de symboles d'information et génèrent des séquences infinies de

symboles codés. Les codes convolutifs sont basés sur l'introduction de la redondance à l'aide de registres à décalage [3].

Les principaux inconvénients du codage en bloc peuvent être évités par une approche différente de la problématique du codage : C'est le cas du codage Convolutif (Convolutional Coding). Ce type de codage ajoute systématiquement de la redondance au message codé au fur et à mesure que les symboles du sont livrés au codeur. Le message codé se forme ainsi itérativement en utilisant un registre à décalage [3].

I.5.b Principe de codage convolutif :

Les codes convolutifs constituent la deuxième grande classe de codes correcteurs d'erreur en plus des codes en blocs. Le codage convolutif consiste à émettre, à partir de la source, des symboles d'information binaires et les transmettre en série dans des séquences assez longues. Au début du codage, toutes les bascules de l'encodeur sont initialisées à zéro. Dans le cas général, au niveau du codeur les symboles sont séquentiellement décalés dans k registres à décalage et chaque registre contient K=m+1 bascules. K est appelée la longueur de contrainte du code. Les K bascules de chaque registre sont connectées aux entrées d'un certain nombre V d'additionneurs modulo 2[3-4]. Les trois principaux paramètres qui caractérisent un codeur convolutionnel sont:

.• La longueur de contrainte • Le taux de codage .• Le polynôme générateur

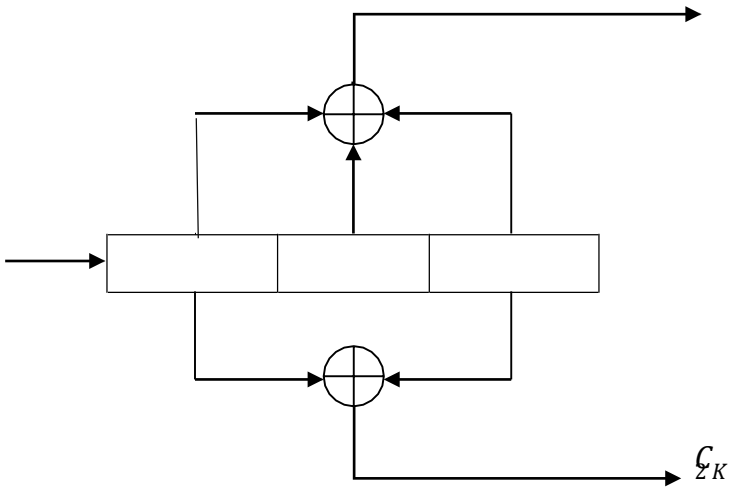

Figure I.1 : exemple de codeur convolutif [5/7] [5]

Le principe du codeur convolutif est illustré par le schéma de la figure suivante :

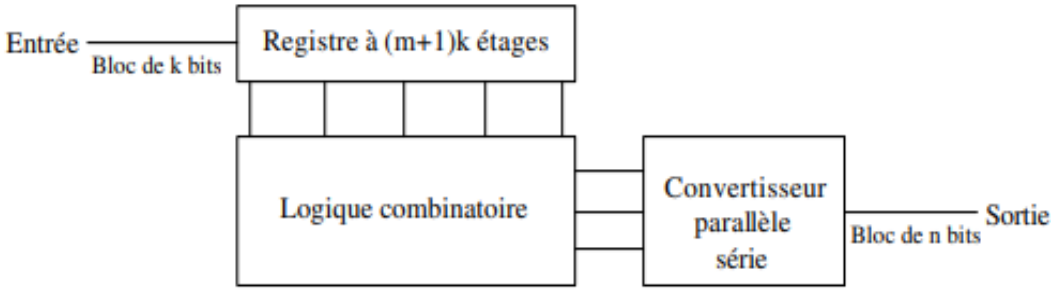

Figure I.2 : principe d'un codeur convolutif [3]

Chaque bloc de *n* bits en sortie du codeur dépend non seulement du bloc de *k* éléments binaires présents à son entrée mais aussi des *m* blocs présents précédemment. Par conséquent, les codes convolutifs introduisent un effet de mémoire d'ordre *m*. La quantité (*m+1*) s'appelle la longueur de contrainte du code. Un tel code est noté *C (n, k, m)*. Son rendement est donc té *R= k/n*. [3]

I.5.c Les types de code convolutif :

I.5.c.1 Codeur convolutif systématique non récursif :

Un codeur convolutif est dit systématique si les symboles codés comportent une réplique exacte des bits d'information de l'entrée. Autrement dit, il laisse passer directement un bit d'information de l'entrée à la sortie comme symbole systématique. Les autres symboles à la sortie de l'encodeur sont appelés symboles de parités (ou redondances) [6-3]. Le principe de fonctionnement d'un codeur convolutif systématique non récursif est illustré à la Figure I.3

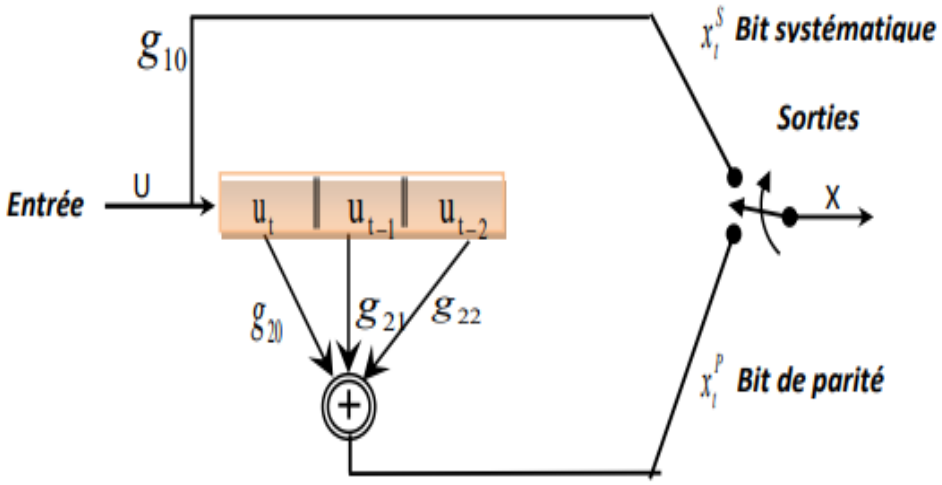

Figure I.3 : Codeur convolutif systématique non récursif [6-3].

I.5.c.2 Les Codes Systématiques Récursifs (CSR) :

Il existe un autre type de codeur, à savoir les codeurs récursifs et systématiques (RSC) qui sont obtenus par les deux propriétés : La récursivité est caractérisée par le fait que la sortie de l'encodeur sont fonctions des entrées et des sorties précédentes ; et la propriété systématique, que nous avons déjà mentionné, qu'il s'agissait de transmettre les bits d'information dans les symboles codés. Le principe de fonctionnement du codeur RSC est illustré dans la Figure I.4. Le contenu de la première cellule de registre à décalage ne dépend pas seulement du bit d'information à l'entrée mais aussi du contenu des cellules du registre.

Un code convolutif est dit récursif lorsque ses polynômes générateurs sont remplacés par les quotients de deux polynômes. Une partie de la sortie est alors réintroduite dans le registre à décalage selon les connexions définies par les polynômes situés aux dénominateurs [3].

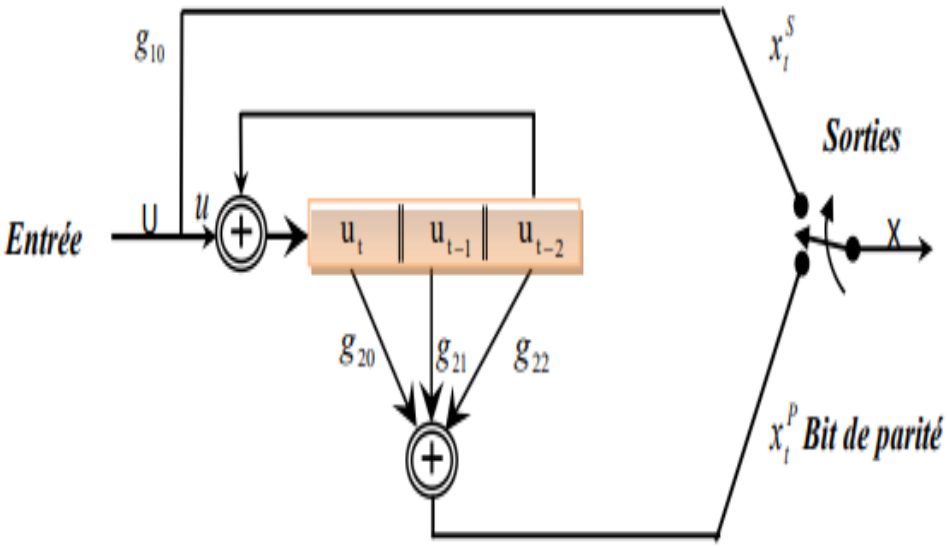

Figure I.4 : Codeur convolutitionnel systématique récursif [6-3].

I.6. Diagramme en treillis :

I.6.A Représentation en treillis :

Lorsqu'un code convolutif est suffisamment simple, il peut être représenté sous forme d'un **treillis**. Il s'agit d'une représentation graphique en deux dimensions, où chaque état du codeur est représenté par un point. L'état d'un codeur correspond aux valeurs des bits du registre du codeur, à l'exception des bits qui viennent de rentrer à l'instant correspondant. L'axe horizontal représente le temps, et l'axe vertical les différents états possibles. Enfin, des flèches indiquent les valeurs du ou des bits d'entrée du codeur [7].

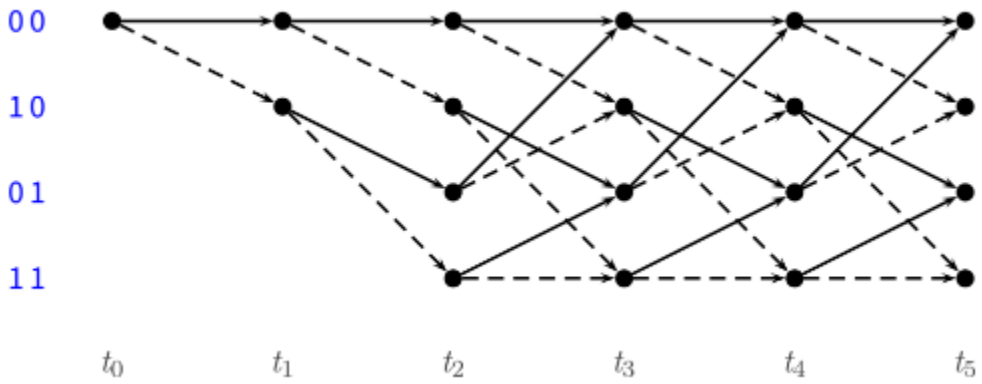

Figure I.5 : Représentation en treillis d'un code convolutif

Les quatre états du codeur sont en bleu, les flèches pleines et en pointillés représentent respectivement l'entrée d'un bit à 0 ou à 1.

La branche en traits pointillés correspondants à la présence d'un élément binaire d'information égale à 0 à l'entrée du codeur et les branche en trait plein correspondants à un élément binaires égale à 1. Le couple binaire mentionné sur chaque branche correspond à la sortie du codeur [3].

Après ($m+1$) décalage, quel que soit l'état initial du codeur, le motif du treillis se répète. De chaque nœud partent 2^k branches (ici 2), et en chaque nœud convergent 2^k $branches$ [3].

Partant de l'état α=00 à l'instant t=0 par exemple, nous voyons qu'il existe quatre chemins qui permettent d'atteindre l'état α=00 à l'instant t=4 :

00 00 00 00 chemin1

11 10 11 00 chemin2

11 01 01 11 chemin3

00 11 10 11 chemin4

I.7.Décodage des codes convolutifs :

Un avantage des codes convolutifs est la disponibilité d'algorithmes de décodage très efficaces fonctionnant sur le diagramme en treillis du code. Deux algorithmes sont principalement utilisés, soit l'algorithme de Viterbi et l'algorithme MAP ou BCJR (doit son nom à ses auteurs Bahl, Cocke, Jelinek et Raviv). Ce dernier sert essentiellement dans des algorithmes de décodage itératif.

Le décodage est l'opération la plus complexe puisqu'elle consiste à vérifier si le mot reçu est correct ou non. Dans le cas où le mot reçu est erroné, le décodeur doit corriger les erreurs puis extraire l'information utile.

Le décodage d'un code convolutif consiste à chercher, dans le treillis, la séquence binaire la plus proche de la séquence reçue. Cette séquence sera appelée séquence la plus vraisemblable. Les techniques de décodage des codes convolutifs tendent de plus en plus à être regroupées sous les deux termes génériques de : décodage séquentiel et décodage de Viterbi.

I.7.a Principe de l'algorithme de Viterbi :

Cet algorithme repose sur la représentation en treillis. L'algorithme de Viterbi est la méthode la plus couramment utilisée pour le décodage à maximum de vraisemblance (ML : Maximum Likelihood) des codes convolutifs de faible longueur de contrainte (typiquement ≤ 7). La distance de Hamming des séquences étant additive (Métrique cumulée). Il est possible de ne conserver en chaque nœud que les chemins les plus vraisemblables appelés survivants. Si deux chemins sont aussi vraisemblables, un seul chemin est toutefois conservé et le choix peut être arbitraire [8].

L'avantage de l'algorithme de Viterbi est qu'il n'est pas nécessaire de calculer les probabilités des branches le long du chemin complet pour tous les mots de code possibles. À chaque nœud du treillis, le meilleur chemin jusqu'ici, appelé chemin survivant (survivor path) pour ce nœud, est stocké et les chemins restants sont rejetés. Cela réduit considérablement la complexité de trouver le mot de code à vraisemblance maximale (ML) pour les codes convolutifs.

Le principe de cet algorithme consiste à estimer les transitions, qui se sont produites dans la

mémoire du codeur Convolutif pendant le codage. L'estimation se base sur le maximum de vraisemblance (Maximum Likelihood), fonction qui permet d'identifier le message globalement le plus probable. L'importante contribution apportée par l'algorithme de Viterbi est la possibilité d'exploiter tout le potentiel de correction d'erreurs, mis à disposition par le code, sans devoir contrôler individuellement chacun des chemins possibles [7].

I.7.b Décodage Séquentiel :

La méthode de décodage séquentielle est caractérisée par l'exécution séquentielle et bidirectionnelle de la recherche du chemin le plus prometteur. La conséquence de cette stratégie de recherche est tout d'abord un temps de décodage variable et fortement influencé par le nombre et le type d'erreurs de transmission. Ensuite, la qualité de protection est aussi fonction du réglage des paramètres du seuil dynamique de référence. D'autre part, la fonction de métrique doit être établie de manière à permettre une comparaison équitable entre chemins se situant à différents niveaux de profondeurs.

Elle utilise la structure en arbre du code convolutif pour chercher de façon séquentielle le chemin correspondant à la séquence la plus vraisemblable à celle émise par la source d'information. Ce type de décodage est plutôt réservé au décodage des codes convolutif a grande longueur de contrainte (>8). L'algorithme le plus répandu et utilisé est celui de Viterbi.

I.8. Concaténation Parallèle :

Les concaténations les plus connues sont la parallèle et la série. Les turbo codes série sont plus complexes à mettre en œuvre cependant ils apportent de plus grand gain en distance minimale et en entrelacement. Les turbo codes parallèle permettent d'approcher la limite de Shannon de manière efficace pour des rapports signal sur bruit très petits tandis que les turbo codes série permettent de diminuer considérablement le taux d'erreurs pour des rapports signal sur bruit faibles et moyens [9].

Les avantages tirés de l'utilisation de ce système sont accompagnés d'une complexité de décodage accrue. Il a fallu 30 ans pour expliquer ce que Berrou, Glaviewc et Thitimajshima [10] appellent aujourd'hui les chaînes parallèles. En fait, ce dernier a été le point de départ des turbo codes. Il a été introduit avec le concept de décodage itératif. Notez que le nombre de

codeurs peut être supérieur à un, qu'ils soient en série ou en parallèle. Le turbo code a été introduit en connectant deux codeurs convolutifs identiques en parallèle [10].

I.9.Entrelacement :

C'est une technique qui prend les symboles d'information d'un alphabet fixe à l'entrée et qui reproduit les mêmes symboles mais dans un ordre temporel différent. On peut dire alors qu'un entrelaceur est un système qui permute les éléments d'une séquence, sans aucune répétition. Cette technique permet d'améliorer la capacité de correction du code et de lutter contre les paquets d'erreurs [9].

Un entrelacement avec de bonnes propriétés de dispersion évite que deux bits proches l'un de l'autre avant d'être entrelacés (c'est-à-dire dans l'ordre naturel) restent proches après l'entrelacement. Ces aspects indésirables provoquent une dégradation des performances car elles introduisent une certaine corrélation dans le processus de décodage itératif. Les propriétés de dispersion d'un entrelacement peuvent être mesurées à travers sa distance spatiale cumulée, définie comme :

$$S_{min} = \min_{j_1, j_2 / j_1 \neq j_2} (|j_1 - j_2| + |\prod(j_1) - \prod(j_2)|) \quad \text{(I.8)}$$

Pour $j_1, j_2 = 1, \ldots, K$. Où K est la longueur de l'entrelaceur et $\prod$ désigne la fonction de permutation qui associe l'indice $\prod(j)$ dans l'ordre entrelacé à un indice j dans l'ordre naturel.

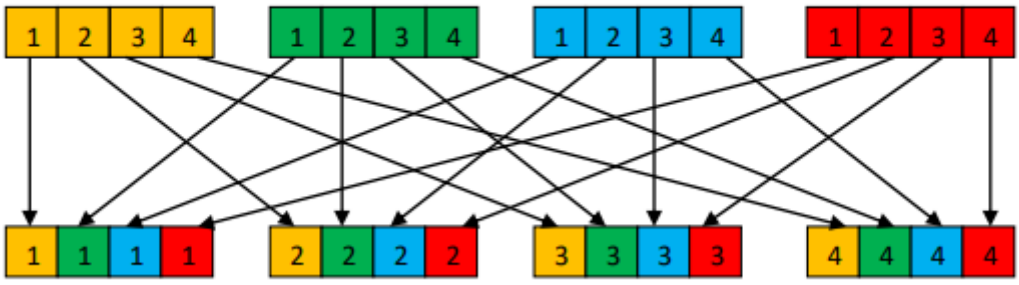

Figure I.6 : Illustration d'un exemple d'entrelaceur

Il existe deux types d'entrelaceur : l'entrelaceur bloc et entrelaceur convolutionnel.

I.9.a Entrelaceur bloc :

Entrelaceur bloc, vient du fait qu'il effectue une permutation sur un bloc de symboles La famille d'entrelacement bloc regroupe plusieurs types d'entrelacement comme l'entrelacement bloc classique, entrelacement pseudo-aléatoire ; entrelaceur en matrice, entrelaceur aléatoire.

L'effet de l'entrelacement est de répartir sur des mots différents les erreurs d'un même paquet d'erreurs, l'entrelacement n'augmente pas la capacité de correction d'un code, il ne fait qu'adapter la puissance de correction aux configurations d'erreurs les plus fréquentes [10].

I.10.Les turbo codes :

Les turbo codes sont une classe de codes correcteurs d'erreurs approchant la limite théorique de capacité formulée par Shannon. Conjointement à leurs excellentes performances de décodage, la complexité calculatoire modérée des turbos décodeurs a permis leur inclusion dans de nombreux standards de communications numériques. Les turbo codes utilisent la composition avec entrelacement de deux codes. Il s'agit souvent de codes systématiques récursifs (RSC) dans lesquels des bits de sortie sont réinjectés en entrée du codeur. Le décodage tire parti de cette propriété, d'où le nom turbo. Le fait de réinjecter des bits de sortie dans les bits d'entrée est un procédé itératif et la correction se fait alors par tour, dans chaque tour les erreurs sont de plus en plus corrigées. Plus précisément, les corrections portent seulement sur peu de bits sont effectuées à chaque tour. Le mot corrigé étant alors réinjecté par le turbo. A partir de ce dernier une nouvelle passe de correction est initiée. Comme le mot a été déjà corrigé, les anciennes erreurs peuvent dorénavant porter sur moins de bits et être mieux corrigées. Pratiquement le nombre d'erreurs mal corrigées ou même non détectées est drastiquement réduit. Si des mots très particuliers peuvent toujours échapper à la correction, le système d'entrelacement permet de repartir les erreurs de manière Pseudo – aléatoire, uniformément dans les mots code. Ainsi, le comportement en moyenne des turbo-codes est très bon.

Le codage turbo est basé sur deux idées fondamentales : La conception d'un code avec des propriétés aléatoires et la conception d'un décodeur qui exploite le décodage itératif facile à mettre en œuvre. Les turbo codes ont une performance exceptionnellement bonne, en particulier pour des longueurs de blocs importantes.

Le décodeur est constitué par la concaténation en série de deux décodeurs convolutifs qui sont séparés par deux entrelaceurs. Ces deux décodeurs partagent leurs informations de manière itérative afin de décoder le message reçu. Ce procédé itératif, introduit au niveau du décodage, permet d'obtenir des gains de performances considérables [10].

I.10.a Turbo code série :

David Forney a proposé durant sa thèse de doctorat en 1965 la concaténation série de deux codes convolutifs séparés par des entrelacements Π. En effet, il a montré que les codes concaténés en séries permettent d'obtenir des probabilités d'erreurs qui décroissent exponentiellement pour un décodeur dont la complexité calculatoire croît pronominalement avec la taille du code. Sur la figure I.7 qui décrits les codes concaténés en série, un code externe C1 ajoute de la redondance aux *K* bits d'information et le mot de code résultant est permuté par un entrelaceur. La séquence permutée servira comme message au second code C2, dit interne, qui génère le mot de code final.

Un turbo code convolutif série est obtenu par la concaténation série de codes convolutifs séparés par des entrelacements. Dans le cas d'un turbo code à deux niveaux le code en amont de l'entrelaceur est appelé code externe et celui en aval est le code interne[11].

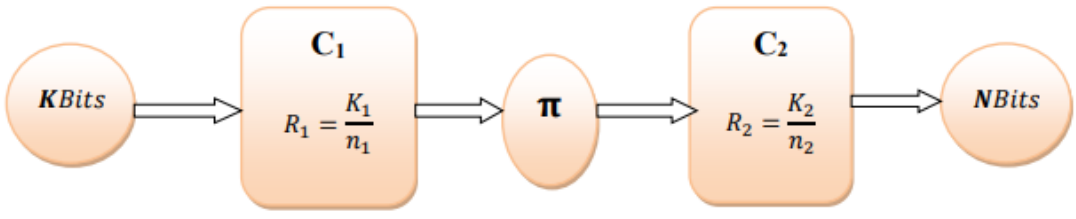

Figure I.7: Turbo code série.

$R_1 = \frac{k_1}{n_1}$ et $R_2 = \frac{k_2}{n_2}$ sont les rendements respectifs des code externe et interne C_1 et C_2 . K est le nombre de bits d'informations qui entrent dans le codeur. N est le nombre de bits codés en sortent du codeur. Le rendement global du turbo code convolutif concaténé en série est équivalent à :

$$R = \frac{K}{N} = R_1 \times R_2 \quad \text{(I.9)}$$

Le code C_1 génère L $= K/R_1$ bits et le code C_2 en génère N = L/R_2 . [11]

I.10.b Turbo codes parallèle :

La structure des turbo codes parallèle a été originalement introduite par Berrou, Glavieux et Thitimajshima. Le codeur turbo est constitué de deux ou plusieurs codeurs convolutifs systématiques récursifs (CSR) concaténés en parallèle et séparés par un ou plusieurs entrelaceurs. La différence avec la concaténation série, c'est que cette fois, les codeurs utilisés sont en parallèle. Le terme concaténation en parallèle provient du fait que les codeurs utilisent la même séquence d'information mais dans un ordre temporel différent alors que dans le cas de la concaténation en série, un codeur au moins doit utiliser la sortie codée de l'autre [11-16]. Les turbo codes que nous considérons dans ce mémoire sont constitués de deux codeurs identiques de type récursifs et systématiques séparés par un entrelaceur. Pour mieux illustrer le codeur turbo, nous allons expliquer ces derniers à la Figure (I.8).

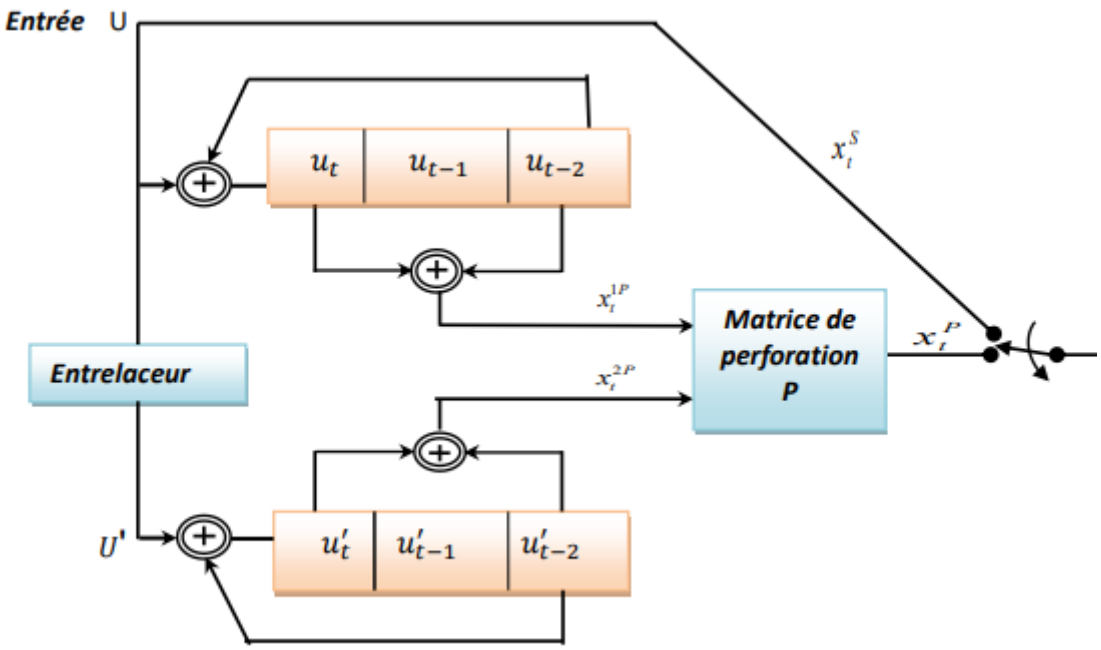

Figure I. 8: Schéma de principe d'un encodeur Turbo parallèle [11].

Nous observons que ce codeur possède deux étages. Le premier étage correspond à la transmission de l'information d'entrée, c'est-à-dire que les bits sont transmis sans aucun changement. Ce que nous appelons le deuxième étage correspond à la génération des symboles de parité permettant la correction des erreurs. Cet étage produit deux symboles de parité pour chaque symbole d'information transmis. Un turbo-code parallèle convolutif est construit par la concaténation parallèle de codes convolutifs séparés par des entrelacements. Chaque code constituant traite une version entrelacée différente de la même séquence d'information.

Le poinçonnage (ou la perforation) est un processus qui consiste à élimination systématique de certains symboles de la séquence codée à la sortie d'un codeur. De cette façon, le rapport du nombre des bits d'information qui entrent dans le codeur et le nombre des symboles codés qui subsistent après la perforation, présente un taux résultant supérieur au taux de codage du codeur non perforé. La perforation a été appliquée dans le cas des turbo codes pour introduire moins de redondance [12].

En examinant la figure I.8, un bloc de bits d'information qui se présente à l'entrée du codeur turbo (U) sera directement passé à sa sortie comme un bloc de symboles systématiques x_t^s . Le codeur supérieur du codeur turbo reçoit les symboles U, dont leur ordre initial, et génère les symboles de parité x_t^{1p} , tandis que le codeur inférieur reçoit des symboles U', issues de

l'entrelaceur, pour produire les symboles de parité x_t^{2p}. Si nous considérons la concaténation parallèle de deux codeurs systématiques dont les taux de codage sont

$R_1 = \frac{b}{n_1}$

Et

$R_2 = \frac{b}{n_2}$

Où:

- $n1$: est le nombre de bits à la sortie du 1^{er} codeur,
- $n2$: est le nombre de bits à la sortie du $2^{\text{éme}}$ codeur,
- b : le nombre de bits de l'information d'entrée.

Le taux de codage global du codeur turbo est :

$$R_t = \frac{b}{n_1 + n_2 - b} = \frac{b}{\frac{b}{R_1} + \frac{b}{R_2} - \text{b}} \quad \text{(I.10)}$$

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

$$\frac{1}{R} = \frac{1}{R_1} + \frac{1}{R_2} - 1 \quad \text{(I.11)}$$

Ou encore :

$$\text{R} = \frac{R_1 R_2}{R_1 + R_2 - R_1 R_2} \quad \text{(I.12)}$$

Le taux de codage global de la figure I.8 est égal à R = $\frac{1}{3}$

Le taux de codage après perforation de la figure I.8 est égal à R = $\frac{1}{2}$

I.11.Le décodage des turbo codes :

En sortie du canal, trois vecteurs bruitées $r^{(0)} = \tilde{u}$, $r^{(1)}$ et $r^{(2)}$sont observés. Afin de retrouver à partir de ces vecteurs l'information $\hat{u}$, le décodage suit le schéma représenté sur la figure I.9. Les grandeurs considérées à chaque nœud du montage sont des LLR, les opérations de décodage étant effectuées dans le domaine logarithmique. Deux décodeurs convolutifs sont placés en parallèle afin d'utiliser simultanément les informations données par chacun d'eux.

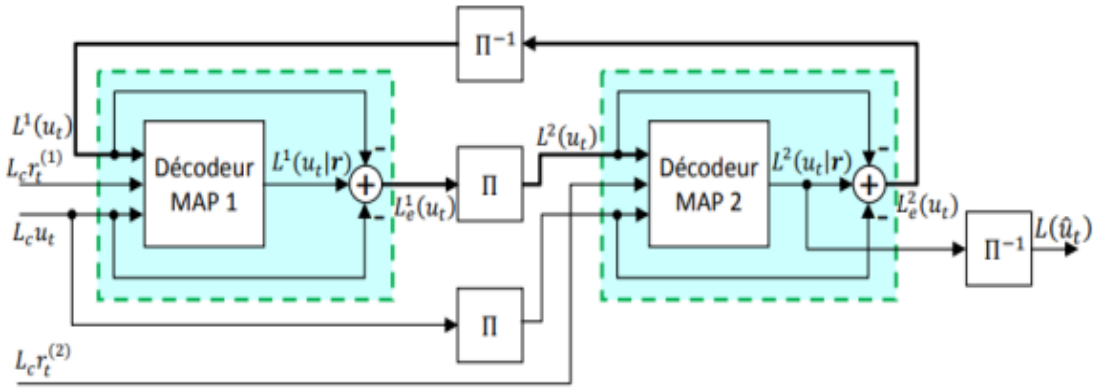

Figure I.9 : Schéma de décodage d'un turbo code (Turbo décodeur)

Comme on le voit sur la figure, chaque décodeur prend trois entrées: les bits de sortie de canal systématiques $\tilde{u}$, les bits de parité transmis par le codeur RSC associé $r^{(i)}$, i =1,2 et les informations provenant du décodeur de l'autre composant sur les valeurs probables des bits concernés.

Cette information provenant de l'autre décodeur est appelée information a priori. Les composants décodeurs doivent également fournir ce qu'on appelle les sorties souples (Soft-output) pour les bits décodés sous forme de LLR. La polarité du LLR détermine le signe du bit, tandis que son amplitude quantifie la probabilité d'une décision correcte.

I.12.Conclusions :

Ce chapitre a été un prétexte pour étudier le codage de canal, le théorème de Shannon de la capacité, la construction des turbo codes, et le décodage des codes convolutifs par le diagramme en treillis. On a aussi donné une idée des méthodes de concaténation ensuit on a étudié décodage des turbo codes.

Dans le chapitre suivant on va étudier les différents algorithmes de décodage pour les turbo codes et les techniques d'approximation de la fonction de correction.

CHAPITRE II

Algorithmes de décodage pour les turbos codes

II.1 Introduction :

L'opération de décodage consiste à retrouver la séquence d'information *s* en présence du bruit. Les deux principales solutions qui permettent la résolution de ce type de problème sont connues sous les noms d'algorithme de Viterbi et d'algorithme MAP, Log-MAP, Max-Log- MAP.

Le choix d'un algorithme de décodage est un problème majeur. En effet, deux algorithmes de décodage ont été proposés pour les Codes Turbo. Le premier est le SOVA (Soft Output Viterbi Algorithme) qui est une variante de l'algorithme de Viterbi.

L'avantage majeur de ces deux approches est qu'elles présentent de bonnes estimations des paramètres, ce qui permet aux décodeurs constitutifs de donner des informations extrinsèques plus fiables.

Le deuxième est celui que nous allons traiter dans ce chapitre. La caractéristique principale de ces d'algorithmes est qu'ils fournissent une mesure de fiabilité sous forme probabiliste sur chaque symbole décodé [13].

II.2 Algorithme MAP :

L'algorithme MAP [14] permet de calculer la probabilité a posteriori de chaque bit d'information ou de chaque symbole transmis.Le décodeur correspondant sélectionne à chaque instant le bit ou le symbole le plus probable. Cet algorithme est resté inutilisable jusqu'à la découverte des turbo codes car elle n'apporte pas d'amélioration de performance notable par rapport à l'algorithme de Viterbi pour le décodage des codes convolutifs et s'avère plus complexe à mettre en œuvre. En revanche, la situation a changé en 1993 car le décodage des turbocodes fait appel à des décodeurs élémentaires à sorties pondérées ou souples et l'algorithme MAP, contrairement à l'algorithme de Viterbi, permet d'associer naturellement une pondération à chaque décision [14].

Rappelons qu'un symbole d'information entrant est constitué de*k* symboles binaires, k=1 pour le cas binaire (notre cas).Considérons que le décodeur reçoit une séquence bruitée, ayant une longueur *N*.

Soit S_t l'état de l'encodeur à l'instant *t*. Le bit u_t est associé à la transition de l'instant t-1 à l'instant t. Les états du treillis au niveau t-1 et au niveau t sont respectivement indexés par les entiers S' et S. De plus, la transition $S_{t-1} \rightarrow S_t$ dépend seulement du symbole d'entrée u_t.

L'algorithme MAP donne, pour chaque bit décodé u_t, la probabilité que ce bit soit +1 ou - 1, sachant la séquence de symboles reçue *r* .Cela revient à trouver le logarithme de rapport de vraisemblance *LLR* (Log-Likelihood Ratio) a posteriori L(u_t/r), où :

$$L(u_t/r) = \ln\left(\frac{P(u_t=+1/r)}{P(u_t=-1/r)}\right) \quad \text{(II.1)}$$

La règle de Bayes nous permet de réécrire cette équation comme :

$$L(u_t/r) = \ln\left(\frac{P(u_t=+1,r)}{P(u_t=-1,r)}\right) \quad \text{(II.2)}$$

En utilisant l'algorithme MAP, nous pouvons réécrire le *LLR* :

$$L(u_t/r) = \ln\left(\frac{\sum_{(s',s)\Rightarrow u_t=+1} P(S_{t-1}=S',S_t=s,r)}{\sum_{(s',s)\Rightarrow u_t=-1} P(S_{t-1}=S',S_t=s,r)}\right) \quad \text{(II.3)}$$

Où $(s', s) \Rightarrow u_t = +1$ est l'ensemble des transitions de l'état précédent $S_{t-1} = s'$ à l'état actuel $S_t = s$ qui peut se produire si le bit d'entrée $u_t = +1$ et de même pour $(s', s) \Rightarrow u_t = -1$. Pour plus de brièveté, nous écrirons P ($S_{t-1} = s'$, $S_t = s$, r) comme P (s', s, $\boldsymbol{r}$).

En utilisant la loi de Bayes de P (a, b) = P ($a \mid b$)P(b), et en supposant un canal de transmission sans mémoire, alors la future séquence reçue $r_{j>t}$ ne dépendra que de l'état actuel $\boldsymbol{s}$ et non pas de l'état précédent $\boldsymbol{s}'$ ou des séquences présentes et précédents reçus du canal $r_t$et $r_{j>t}$. La probabilité conjointe P (s', s, r), peut être écrite comme le produit de trois probabilités [14] :

$$P(s', s, r) = P(s', r_{j<t}).\ P(s, r_t/s')\ .\ P(r_{j>t}/s)$$

$$= \underbrace{P(s', r_{j<t})}_{=\alpha_{t-1}(s')}.\ \underbrace{P(s/s').P(r_t/\{s',s\})}_{.\gamma_t(s',s)}\ .\ \underbrace{P(r_{j>t}/s)}_{.\beta_t(s)} \quad \text{(II.4)}$$

$\alpha_{t-1}(s')$: la probabilité que le treillis soit à l'état $\boldsymbol{S}'$ à l'instant $t-1$ et que la séquence reçue jusqu'à ce point soit $r_{j>t}$, ou tout simplement la métrique de nœud dans le sens Aller (forward).

$\beta_t(s)$: la probabilité que la future séquence reçue sera $r_{j>t}$, étant donné que le treillis est dans l'état $\boldsymbol{s}$ à l'instant $\boldsymbol{t}$, ou tout simplement la métrique de nœud dans le sens Retour (backward).

$\gamma_t(s', s)$: La probabilité que le treillis passe à l'état $\boldsymbol{s}$ et la séquence reçue pour cette transition est r_t , $\boldsymbol{s}'$ à l'instant à $t-1$, ou tout simplement la métrique de branche du treillis (transition).

Les probabilités récurrentes des métriques de nœuds dans le sens Aller et Retour de l'algorithme MAP sont calculées de la manière suivante [14] :

$$\alpha_t(s) = \sum_{s'} \gamma_t(s', s)\, \alpha_{t-1}(s') \qquad \text{(II.5)}$$

$$\beta_{t-1}(s') = \sum_{s} \gamma_t(s', s)\, \beta_t(s)$$

Ainsi, une fois que les valeurs de $\gamma_t(s', s)$ sont connues, les valeurs de $\alpha_t(s)$ et $\beta_{t-1}(s')$ peuvent être calculées récursivement. En supposant que le treillis à l'état initial $s_0 = 0$, de plus il est terminé à zéro (zero-tail), donc les conditions initiales de cette récursivité sont :

$\alpha_0(S_0 = 0) = \beta_N(S_N = 0) = 1$

$\alpha_0(S_0 = s) = \beta_N(S_N = s) = 0$ pour tous $s \neq 0$

Le *LLR* $L(u_t|\boldsymbol{r})$ s'écrira donc par :

$$L(u_t|\boldsymbol{r}) = \ln\left(\frac{\sum_{(s',s))\Rightarrow u_t=+1} \alpha_{t-1}(s')\, \gamma_t(s', s)\beta_t(s)}{\sum_{(s',s))\Rightarrow u_t=-1} \alpha_{t-1}(s')\, \gamma_t(s', s)\beta_t(s)}\right) \qquad \text{(II.6)}$$

Un décodeur qui implémente le critère MAP symbole par symbole, calcule pour tous les bits d'information la valeur binaire $\widehat{u_t}$ qui maximise la probabilité a posteriori P ($u_t = +1 \mid \boldsymbol{r}$). Par conséquent, un décodeur MAP qui procède symbole par symbole minimise réellement le taux d'erreurs binaire. Ainsi, nous pouvons formuler la règle de décodage MAP symbole par symbole en termes de rapport de vraisemblance $L(\widehat{u_t}) = L(u_t|\boldsymbol{r})$, et obtenir [14] :

$$\widehat{u_t} = \begin{cases} 0 \text{ si } L(\widehat{u_t}) \geq 0 \\ 1 \text{ si } L(\widehat{u_t}) < 0 \end{cases} \qquad \text{(II.7)}$$

Nous observons que la décision ferme de décodage MAP du symbole peut être basée sur le signe de la valeur LLR, et la quantité $|L(\widehat{u_t})|$ est la fiabilité de cette décision.

II.3 Algorithme Max-Log-MAP :

Le décodage selon le critère MAP est difficile à intégrer dans un circuit dû à la complexité de calcul, imposée notamment par les multiplications, les divisions et les opérations exponentielles. En pratique, il existe plusieurs algorithmes sous-optimaux qui ont pour objectif de faciliter l'implémentation.

Pour contourner les opérations arithmétiques très complexes citées au-dessus, l'une des méthodes efficaces permettant de simplifier ces opérations est de transférer le problème dans un espace logarithmique. Cette modification permet de transformer respectivement les multiplications, les divisions et les opérations exponentielles en additions et soustractions.

L'algorithme Max-Log-MAP était proposé au début par Koch et Baier [15] et Erfanian et al [16]. Cette technique apporte une simplification du MAP, en transformant la récursivité, dans le domaine logarithmique. Cette approximation réduit considérablement la complexité de cet algorithme, mais d'autre part, ces performances sont considérées comme étant sous optimales, en comparaison avec l'algorithme originel MAP. L'algorithme MAP calcule le *LLR* en utilisant les valeurs $\alpha_{k-1}(s')$, $\gamma_k(s', s)$ et $\beta_k(s)$, calculées de manière récursive. L'algorithme Max-Log-MAP simplifie ce calcul, en le transférant dans le domaine logarithmique par l'approximation [13] :

$$\log(\sum_i exp^{x_i}) \approx max(x_i) \tag{II.8}$$

$max(x_i)$ exprime la valeur maximale de x_i; par conséquent, on peut définir :

$$A_k(s) \cong \log\ (\alpha_k(s)) \tag{II.9}$$

$$B_k(s) \cong \log\ (\beta_k(s)) \tag{II.10}$$

$$\Gamma_k(s', s) \cong \log\ (\gamma_k(s', s)) \tag{II.11}$$

En appliquant cette nouvelle définition sur $\alpha_{k-1}(s')$, $\gamma_k(s', s)$, $\beta_k(s)$, on aura :

$$A_k(s) \cong \log\ (a_k(s)) \tag{II.12}$$

$$= \log\ (\sum_{s'} a_{k-1}(s')\ \gamma_k(s', s))$$

$$= \log\ (\sum_{s'} \exp[\, A_{k-1}(s') + \Gamma_k(s', s)])$$

$$\approx \max_s (A_{k-1}(s') + \Gamma_k(s', s))$$

Cette équation implique que pour chaque chemin dans le treillis, à partir de chaque état précédent, jusqu'à l'état présent $S_k = s$, l'algorithme additionne la valeur métrique $\Gamma_k(s', s)$, relative à la branche de transition entre les états s' et s avec $A_{k-1}(s')$, afin de donner une nouvelle estimation $\widetilde{A_k}(s)$ correspondant au chemin suivi dans le treillis. La nouvelle valeur de $A_k(s)$ est la valeur maximale de ses estimations $\widetilde{A_k}(s)$ Cette valeur représente la probabilité que le treillis est dans l'état $S_k = s$ en temps $\boldsymbol{k}$. A cause de l'approximation dans l'algorithme Max-Log-MAP, uniquement le chemin avec le maximum de probabilité (ML) parcourant l'état $S_k = s$ est considéré. Le calcul récursif des valeurs de $A_k(s)$ est pratiquement la même manière de calcul récursif pour l'algorithme de Viterbi (pour chaque paire de chemins fusionnant sur l'état $S_k = s$. Le chemin survivant (The Survivor) est déterminé en utilisant deux additions et une comparaison. De la même façon, on peut calculer récursivement les valeurs de $B_{k-1}(s')$, e t $\Gamma_k(s', s)$ [13] :

$$B_{k-1}(s') \cong \log\left(B_{k-1}(s')\right)$$

$$\max_{s} \left(B_k(s) + \Gamma_k(s', s)\right) \qquad \text{(II.13)}$$

Ceci est équivalent au calcul récursif utilisé dans Viterbi, sauf la différence que cette récursivité se fait de gauche à droite dans le treillis.

Pour le calcul des valeurs de $\Gamma_k(s', s)$ on a ;

$$\Gamma_k(s', s) \cong \log\left(\gamma_k(s', s)\right)$$

$$= \frac{1}{2} u_k L(u_k) + \frac{L_c}{2} \sum_{i=1}^{n} y_{kl}\, x_{kl} \qquad \text{(II.14)}$$

la valeur métrique $\Gamma_k(s', s)$ est la même utilisée par Viterbi avec, en plus, une information a priori $u_k L(u_k)$. C'est pourquoi le terme de corrélation $\sum_{i=1}^{n} y_{kl}\, x_{kl}$ est pondéré par l'indice de fiabilité du canal L_c .

Finalement, on peut écrire que l'information a posteriori $L(u_k/y)$ que calcule l'algorithme Max-Log-MAP :

$$L(u_k/y) = \log\left(\frac{\sum_{u_k=+1} \alpha_{k-1}(s')\, \gamma_k(s', s) \beta_k(s)}{\sum_{u_k=-1} \alpha_{k-1}(s')\, \gamma_k(s', s) \beta_k(s)}\right) \qquad \text{(II.15)}$$

$$\approx \max_{(s', s) u_k=+1} \left(A_{k-1}(s') + \Gamma_k(s', s) + \beta_k(s)\right) - \max_{(s', s) u_k=-1} \left(A_{k-1}(s') + \Gamma_k(s', s) + \beta_k(s)\right) \qquad \text{(II.16)}$$

La complexité de l'algorithme Max-Log-MAP n'est pas significativement élevée par rapport à celle de Viterbi. Il est à noter que cela nécessite deux récursivités au lieu d'une pour l'algorithme de Viterbi. La valeur métrique par branche (Branch Metric), nécessite l'ajout d'un terme relatif à l'information en priori u_kL (u_k) . L'algorithme Max-Log-MAP est à peine trois fois plus complexes que l'algorithme de Viterbi. L'espace mémoire nécessaire pour l'algorithme Max-Log-MAP serait identique à celui du Viterbi [13].

II.4 Algorithme Log-MAP :

L'algorithme Max-Log-MAP donne une légère dégradation des performances par rapport à celle du MAP. Elle résulte en une chute de performances de l'ordre de 0.35 dB [13] durant un décodage itératif. Cependant, cette approximation peut être ajustée par l'utilisation de l'algorithme Jacobien.

L'algorithme Log-MAP est un algorithme MAP qui est implémenté dans le domaine logarithmique pour réduire la complexité calculatoire. C'est un algorithme optimal pour le décodage itératif des Turbo codes.

La réduction de la complexité de calcul de l'algorithme Max-Log-MAP, obtenue à l'aide de l'utilisation des approximations, est contre balancée par la dégradation de la qualité de l'estimation du *LLR*. Pour pallier à ce problème, l'algorithme Log-MAP utilise la notion du 'Jacobian logarithm' de manière à améliorer la qualité des approximations.

L'algorithme jacobien est décrit par [13][17]:

$$\log(e^{x_1}+e^{x_2}) = \max(x_1, x_2) + \log(1 + e^{-|x_2-x_1|}) \qquad \text{(II.17)}$$

$$=\max(x_1, x_2) + f_c(|x_2 - x_1|) \qquad \text{(II.18)}$$

Dans ce cas, la quantité $A_m(k)$ devient [25] :

$$A_m(k) = \log\left(\sum_{m'=1}^{M_{tr}} e^{A_{m'}(k-1)+T_{m'm}(k)}\right) \qquad \text{(II.19)}$$

$$= \max^*_{m'=1,\ldots,M_{tr}} \left(A_{m'}(k-1) + \Gamma_{m'm}(k)\right)$$

De la même manière, Beta peut être calculé avec complexité par :

$$\beta_{m'}(k) = \underset{m'=1,\ldots,M_{tr}}{max^*} \left(\beta_{m'}(k+1) + \Gamma_{m'm}(k+1) \right) \quad \text{(II.20)}$$

En appliquant l'algorithme Log-MAP au décodage turbo, Le logarithme supprimera les exponentielles. Pour un codeur qui a un taux R=1/2, $\Gamma_{m'm}$ peut donc être calculé simplement par :

$$T_{m'm}(k) = \frac{1}{2}\text{u(k) La (u(k))} + \frac{L_c}{2} y_u\text{(k) u(k)} + \frac{L_c}{2}\, y_1(k)\, x_1(k) \quad \text{(II.21)}$$

Enfin, le *LLR* peut être calculé par [13][17]:

$$\text{L (u(k))} = \underset{m',m/u(k)=1}{max^*} \left[A_{m'}(k-1) + \Gamma_{m'm}(k) + B_m(k) \right] \quad \text{(II.22)}$$

$$- \underset{m',m/u(k)=0}{max^*} \left[A_{m'}(k-1) + \Gamma_{m'm}(k) + B_m(k) \right]$$

II.5 Techniques d'approximation de la fonction de correction :

Cette section donne un bref aperçu des algorithmes existants qui approxime la fonction de correction afin d'obtenir une mise en œuvre simple avec des performances améliorées par rapport au Max-Log-MAP.

II.5.a La technique d'approximation de Robertson :

Robertson a proposé de quantifiée $| x_1 - x_2 |$ en 8 valeurs qui varient entre 0 et 5. Une table LUT (Look Up Table) qui contient huit '8' valeurs suffit. Il s'agit donc de quantifier les valeurs 0 à 5 en utilisant une quantification uniforme [17].

II.5.b Algorithme Constant Log-MAP:

Pour cet algorithme, proposé par [18], la fonction de correction f c (x) est approchée avec la règle suivante :

$$f_c(x) = \begin{cases} \frac{3}{8}, & -2 \leq x < 2 \\ 0, & \text{ailleurs} \end{cases} \quad \text{(II.23)}$$

L'algorithme Constant Log-MAP offre une implémentation hardware simple mais avec un compromis en termes de performances.

II.5.c Algorithme Linear Log-MAP:

Dans [19], l'auteur propose une approximation linéaire de la fonction de correction en utilisant l'expansion de la série MacLaurin. On constate que la fonction de correction est efficace lors

que $f_c(x)$ est autour de zéro. Par conséquent, la série de Maclaurin peut être exploitée pour approximer la fonction de correction autour de zéro. En négligeant les séries d'ordre deux et supérieures de Maclaurin, l'approximation du terme de correction est donnée par :

$$f_c(x) \approx \max(0, \ln 2 - \frac{1}{2}\mathrm{x}) \qquad \text{(II.24)}$$

Cette approximation offre de meilleures performances que l'algorithme Constant Log-MAP et ne nécessite qu'un simple calcul linéaire [13][19].

II.5.d Algorithme Multistep Log-MAP:

Une solution plus précise et élégante pour approximer la fonction de correction est donnée par [20]. Le rapprochement avec le terme de correction tel que suggéré par [29] est donné par :

$$f_c(x) \approx \frac{\ln 2}{2^{[x+0.5]}} \qquad \text{(II.25)}$$

Où [x+0.5] désigne le plus grand entier qui est plus petit ou égal à x + 0.5.

Notez que la division par 2 peut être facilement effectuée dans les systèmes numériques en implémentant [x + 0.5] décalages binaires. L'algorithme utilise des registres à décalage stockant la constante ln(2) effectué la division [30]. Cependant, afin de faciliter le calcul rapide, un registre à décalage est nécessaire pour cet algorithme [13][20].

La figure suivante représente l'approximation de la fonction de correction [13].

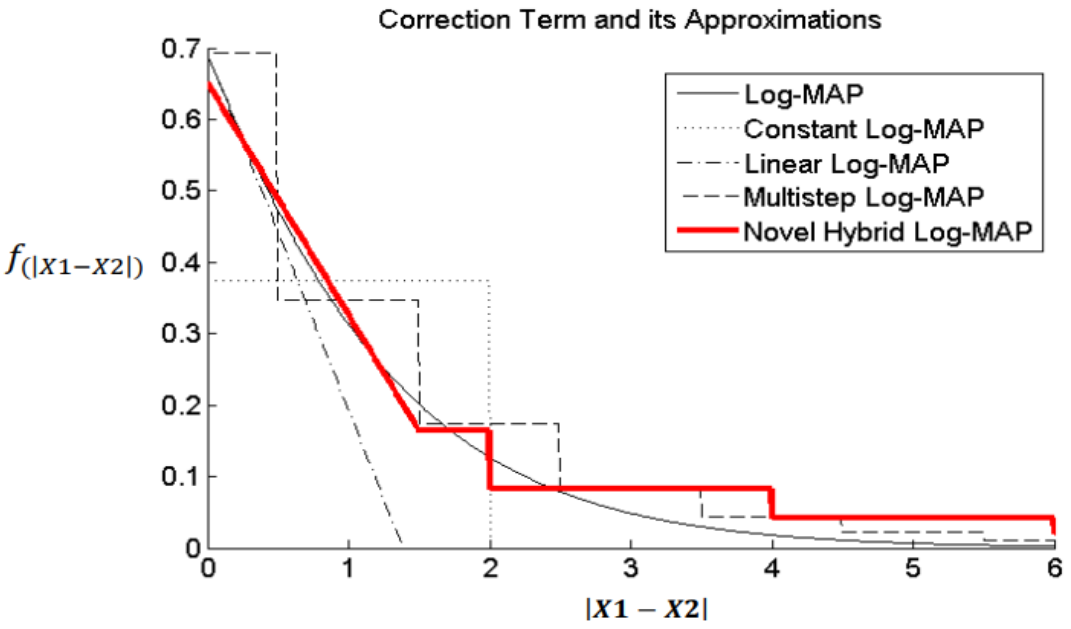

Figure II.1: Approximations de la fonction de correction [13]

II.6 Conclusion :

Ce dernier chapitre de la première partie donne une description détaillée des algorithmes de décodage itératif des turbo codes, c'est-à-dire, les algorithmes MAP, Max-Log-MAP et Log-MAP. L'algorithme MAP est un algorithme optimal mais complexe point de vu calcul. Les algorithmes Log-MAP et Max-Log-MAP sont des versions simplifiées de l'algorithme MAP.

L'algorithme le plus utilisé en pratique est l'algorithme Log-MAP. Il garantit un bon compromis complexité – performance en exploitant le logarithme jacobien et en utilisant une approximation de la fonction de correction.

Dans le chapitre suivant, on va étudier les algorithmes à complexité réduite.

CHAPITRE III

Algorithmes à complexité réduite

III.1. Introduction :

L'efficacité d'un algorithme se mesure par le temps d'exécution en fonction de la taille des données d'entrée et l'espace mémoire nécessaire pour exécuter l'algorithme en fonction de la taille des données d'entrée. Ces deux fonctions sont appelées la complexité de l'algorithme. La détermination de ces fonctions est appelée analyse de complexité.

Pour mesurer le temps d'exécution en fonction de la taille des données, il est nécessaire de se donner une unité de mesure. Nous traitons les opérations de base effectuées (arithmétique, opérations logiques, affectation, etc.) comme un temps constant. L'analyse consiste à exprimer le nombre d'opérations effectuées en fonction de la taille des données d'entrée.

III.2 Algorithme BCJR :

Considérons un codeur convolutif décrit par un treillis, et une séquence x = $x_1 x_2 \ldots x_N$ où x_K est le symbole généré par le codeur à l'instant k. Le bit d'information ou d'entrée de message correspondant, u_K, peut prendre les valeurs 0 ou 1 avec une probabilité a priori P(u_K), à partir de laquelle on peut définir le rapport dit de log-vraisemblance (LLR) [14].

$$L(u_K) = ln \frac{p(u_K=+1)}{p(u_K=-1)} \qquad \text{(III.1)}$$

Ce rapport log-vraisemblance est égal à zéro avec des bits d'entrée équiprobables.

Supposons que la séquence codée x soit transmise sur un canal de bruit blanc gaussien additif sans mémoire (AWGN) et reçue sous la forme de la séquence de y = $y_1 y_2 \ldots y_N$, comme illustré à la Figure III.1. Cette séquence est transmise au décodeur et utilisé par le BCJR [14], ou tout autre algorithme, afin d'estimer la séquence de bits d'origine u_K .pour lequel l'algorithme calcule a posteriori le rapport de log-vraisemblance L (u_K/ y), un nombre réel défini par le rapport

$$L(u_K|y) = \ln \frac{p(u_K=+1|y)}{p(u_K=-1|y)} \qquad \text{(III.2)}$$

Le numérateur et le dénominateur de l'Eq (III.2) contiennent des probabilités conditionnelles a posteriori, c'est-à-dire des probabilités calculées après que nous connaissons y. Le signe positif

ou négatif de L (u_K/ y) indique quel bit, 1 ou 0, a été codé à l'instant k. Son module est une mesure de fiabilité de que nous avons faite. Cette information fournie par L (u_K y) peut alors être transférée vers un autre bloc de décodage, s'il y en a un, ou simplement convertie en une valeur de bit par une décision dure [14].

Comme il a été dit précédemment, si L (u_K y) < 0 le décodeur estimera que le bit u_K = -1 a été envoyé. De même, il estimera u_K= +1 si L (u_K y) > 0 [14].

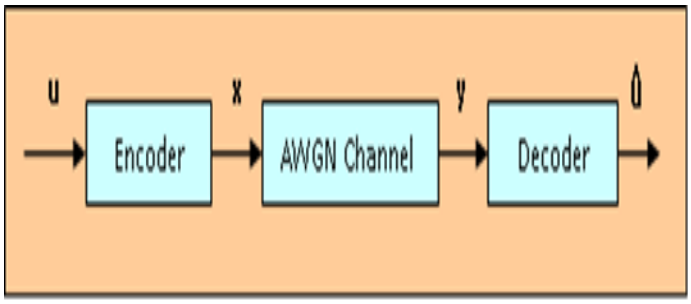

Figure III.1 : Un schéma fonctionnel simplifié du système considéré [14]

Il est pratique de travailler avec des treillis. Admettons que nous ayons un taux ½ avec M = 4 états S = {0, 1, 2, 3} et une section en treillis comme celle présentée sur la Figure III.2. Chaque branche est étiquetée avec le mot de code à deux bits associéx_K, où, pour simplifier, 0 et 1 correspondent respectivement à -1 et +1. [14]

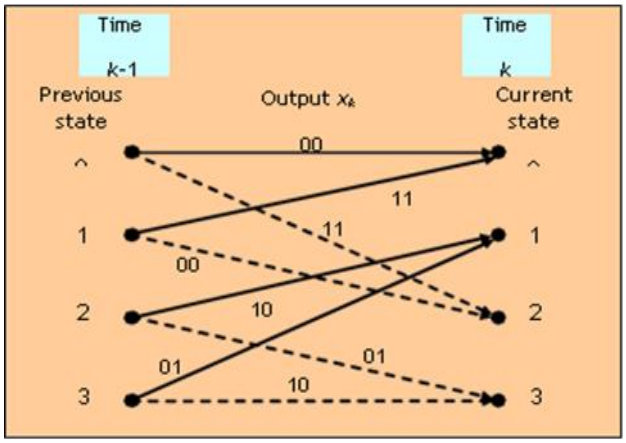

Figure III.2 : Le treillis de code convolutif M = 4

III.3. Algorithme M-BCJR :

Le M-BCJR [21] est une version à faible complexité de l'algorithme MAP-BCJR (Maximum A Posteriori). Il consiste à n'utiliser que les M meilleures états du treillis. Les autres états ne sont pas évalués à l'étape suivante, ce qui réduit la complexité. Le M-BCJR trouve ses applications dans la turbo-égalisation et le turbo décodage des turbo codes convolutifs. Il peut réduire le treillis des détecteurs/décodeursjusqu'à M états, mais lorsque M diminue, les performances seront dégradées. Notons que pour la turbo égalisation, certaines conceptions utilisent des complexités très faibles (deux états « 2 » seulement) sans utiliser la stratégie M-BCJR [22].

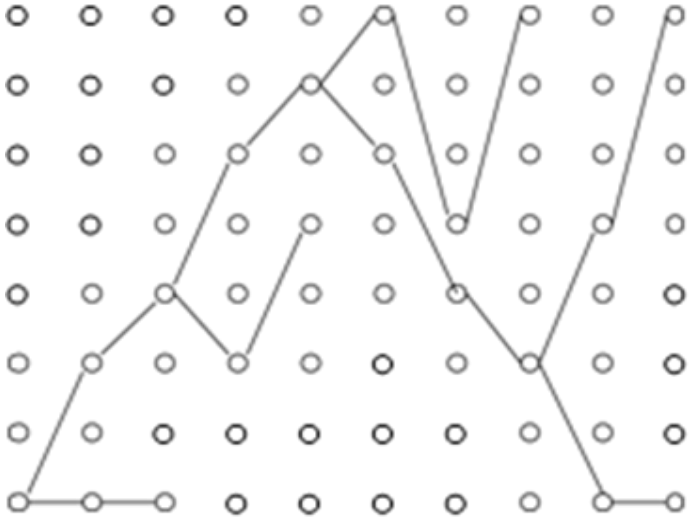

Figure III.3 : Modèle de calcul idéalisé dans l'algorithme M-BCJR sur un treillis à 8 états.

Une ligne reliant deux nœuds indique que le nœud gauche a été utilisé pour calculer le nœud [21-25].

III.4. L'algorithme T–BCJR :

La complexité est l'un des facteurs qui déterminent le choix de la version de l'algorithme de décodage utilisé. Notons que pour l'algorithme MAP, en général, pour chaque transition dans

le treillis, on doit calculer et stocker ($2^m\alpha$), ($2^m\beta$) et (2^m.M γ), plus les probabilités APP p_0(t) et p_1(t), tel que $\boldsymbol{m}$ est la mémoire du codeur et $\boldsymbol{M}$ est le nombre d'états de la modulation utilisée [21][25].

Cela signifie que la complexité de calcul est énorme pour des codes convolutifs qui ont une grande longueur de contrainte.Ceci affaiblit l'importance de l'algorithme MAP et de ses applications. D'autres variantes avec une complexité réduite doivent être trouvées. La première version vise à réduire la complexité et les méthodes de calcul de (α), (β) et (γ). Ce sont les algorithmes Log-MAP et Max-Log-MAP [25].

Notez que ces versions évaluent tous les états du treillis. Les seules variantes du MAP qui réduisent le nombre d'états sont les algorithmes T-BCJR et M-BCJR [25].

La contribution proposée par Franz [21] exploite le fait que certaines probabilités dans le fonctionnement de l'algorithme BCJR sont très faibles. L'idée est d'ignorer ces faibles probabilités et leurs calculs associés sans dégraderles performances. Elle consiste à forcer les faibles états(α faibles) à 0 et réduire donc considérablement la complexité.

III.5. Principe de l'algorithme T-BCJR :

Dans le T-BCJR simplifié (T en anglais veut dire Threshold, c'est-à-dire seuil), les probabilités des états a_t sont mises à zéro chaque fois qu'elles tombent en dessous du seuil T. À chaque instant, les a_t trouvés sont normalisés à l'unité. Pour calculer les a_t, seules les composantes de a_{t-1} qui dépassent le seuil sont utilisées. Les autres états sont forçés à zéro. Il s'agit du T-BCJR simplifié [21].

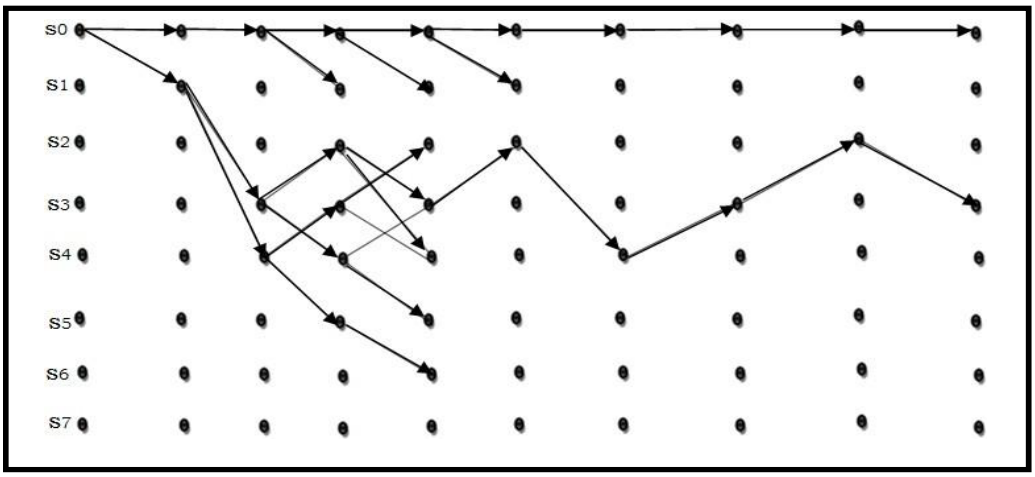

Figure III.4: Un exemple d'application de l'algorithme T-BCJR [21].

La figure (III.4) montre le treillis d'un codeur qui a une longueur de contrainte l =4. Une ligne reliant deux nœuds indique que la valeur du nœud gauche a été utilisée pour calculer la valeur du nœud droit, et que la valeur droite a survécu à la réduction.

III.6.Application de l'algorithme T-BCJR sur un système avec concaténation série :

Nous présentons un exemple d'une concaténation série de deux codes, avec un décodage réduit de type T-BCJR.

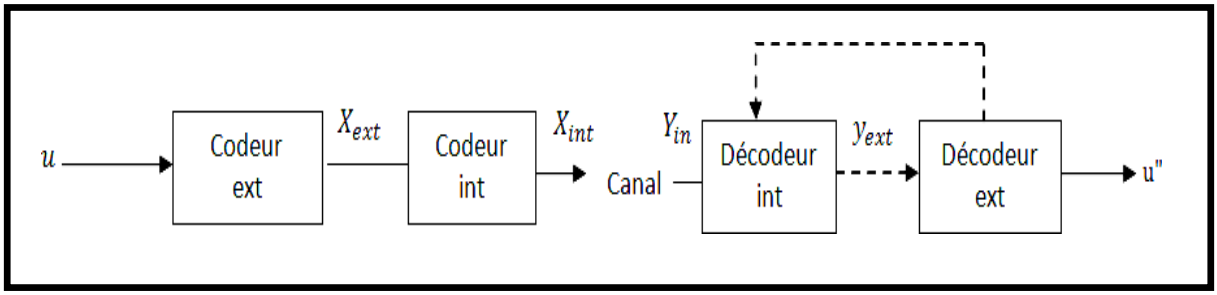

Figure III.5 : Concaténation série [21].

Le codeur externe a un taux de codage 2/3. Le codeur intérieur est un codeur de taux 1/2.

Puisqu'on à un système de concaténation série, donc le rendement de ce système est la multiplication du rendement des deux codes :

$$R = R_1 \times R_2 = \frac{1}{2} \times \frac{2}{3} = \frac{1}{3} \qquad \text{(III.3)}$$

La taille de trame est de 212 bits. Un entrelaceur aléatoire est inséré entre les deux codeurs. Le décodeur externe est un décodeur SOVA (Soft Output Viterbi Algorithm). Le système fonctionne mieux lorsque le décodeur interne utilise l'algorithme BCJR et le décodeur externe est l'algorithme SOVA. Lorsque le décodeur interne est un SOVA, les performances chutent à environ 0,2 dB.

Nous présentons quelques résultats tirés de [21]. Le système comprend un décodeur interne T-BCJR, un décodeur externe SOVA. Les résultats de simulation de ce système sont présentés sur la figure III.6.

Notons que les seuils T utilisés sont : T=0.05, 0.01, 0.005, 0.001, 0.0001

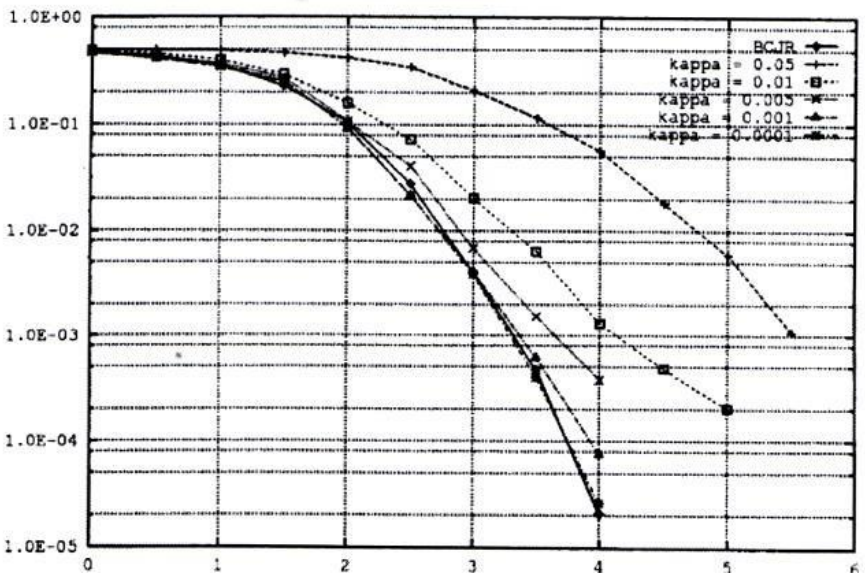

Figure III.6 : Performances du turbo décodeur T-BCJR [21]

Avec un seuil de 0. 0001, la performance du BER est équivalente à celle du BCJR.

Le nombre moyen d'états qui sont resté au-dessus du seuil est montré dans la figure III.7, est en fonction du rapport E_b /N_0, pour les différents seuils.

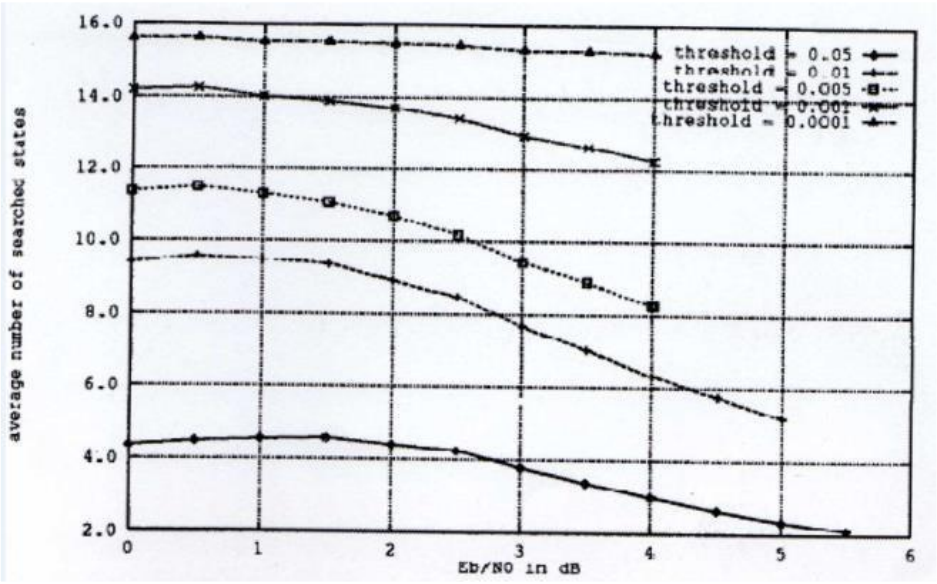

Figure III.7 : Nombre moyen des états survivants en fonction du E_b /N_0en dB pour plusieurs seuils et une concaténation série, décodage T-BCJR [21]

III.7. Algorithme Z-MAP :

III.7.a Introduction : Comportement de l'algorithme M-BCJR :

Dans cette partie nous présentons un troisième algorithme à faible complexité qui constitue une version optimale de l'algorithme M-BCJR. Montrons rapidement le principe de ce dernier (M-BCJR). Il consiste simplement à conserver les M valeurs significatives de alpha $\alpha_t(m)$ à l'avant (beta $\beta_t(m)$ à l'arrière). Les autres états sont déclarés morts et mis à 0. Cela implique une réduction : des états calculés (alpha, beta forcés à zéro), des branches sortantes de chaque état (donc gamma, également forcé à zéro) car [25]:

$$\alpha_i(m)=\sum_{m'=0}^{M-1} \alpha_{i-1}(m')\gamma_t(m',m) \qquad \text{(III.4)}$$

Et

$$\beta_i(m)=\sum_{m'=0}^{M-1} \beta_{i+1}(m')\gamma_t(m',m) \qquad \text{(III.5)}$$

Les observations du comportement de l'algorithme M-BCJR montrent que sa faiblesse est l'amplification des erreurs. La nature de l'algorithme M-BCJR provoque de mauvaises estimations de alpha $\alpha_t(m)$ en raison du forçage à zéro de certains états. Cela provoque des paquets d'erreurs. Aux moments erronés, on observe un ou plusieurs concurrents avec l'état le plus probable. La seule façon de corriger ce mauvais comportement est d'augmenter le nombre d'états. Ce mécanisme est appelé 'l'algorithme Z-MAP'. [25]

III.7.b Moments d'erreur :

Dans le décodage M-BCJR, nous observons que les instants corrects sont caractérisés par une impulsion avec une valeur alpha élevée pour l'état le plus probable (Figure III.8). Les autres alphas ont des niveaux (probabilités) très faibles. La présence d'erreur est généralement caractérisée par la présence d'un ou plusieurs concurrents (Figure III.9). Ce phénomène peut être utilisé comme un critère facile pour localiser la possibilité d'erreur.

Prenons par exemple un treillis à « 4 » états (Figure III.10) d'un codeur qui a une longueur de contrainte égale à « 3 ». [25]

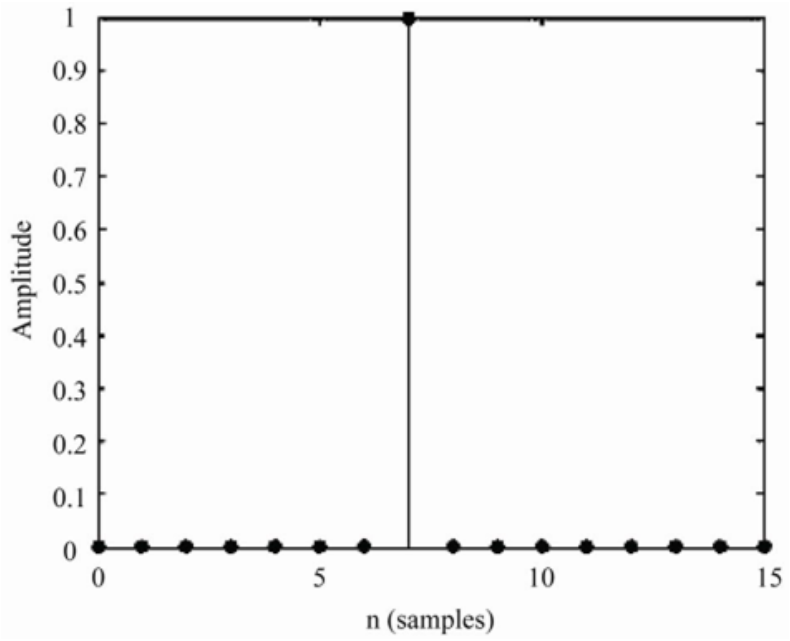

Figure III.8 : L'état le plus probable à l'instant correct [25]

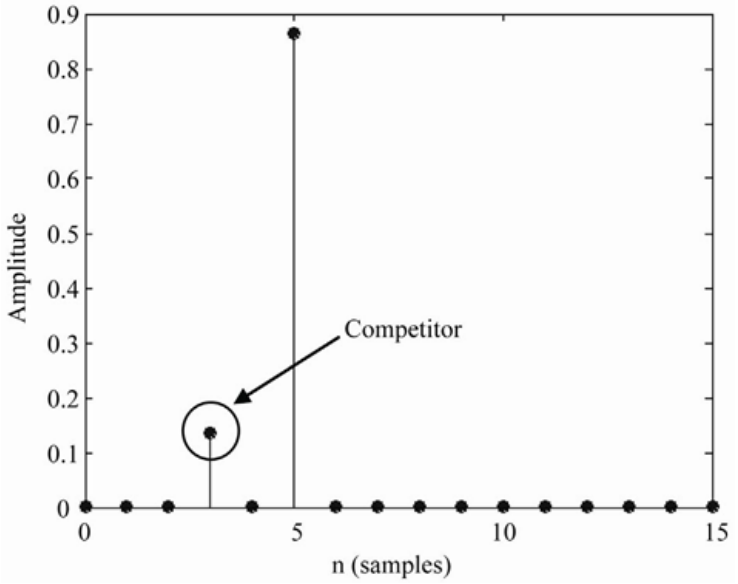

Figure III.9 : Présence de concurrent dans l'instant d'erreur [25].

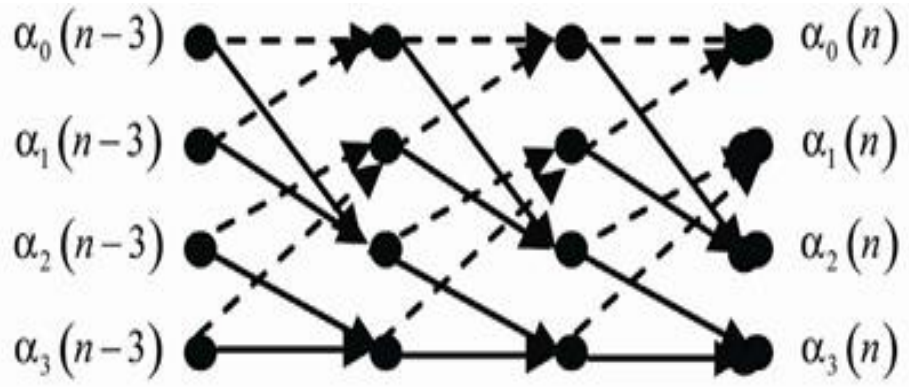

Figure III.10 : Treillis à quatre états [25].

On établit une relation entre l'instant « n » et le passé. Prenons ce passé « n - 3 ». Dans ce cas, alpha de l'état « 0 » est donné par :

$$\begin{aligned}\alpha_0(n)=&\lfloor\lceil \gamma_{00}(n-2)\gamma_{00}(n-1)\gamma_{00}(n)\\&+\gamma_{02}(n-2)\gamma_{21}(n-1)\gamma_{10}(n)\rceil\rfloor\alpha_0(n-3)\\&+\lfloor\lceil \gamma_{10}(n-2)\gamma_{00}(n-1)\gamma_{00}(n)\\&+\gamma_{12}(n-2)\gamma_{21}(n-1)\gamma_{10}(n)\rceil\rfloor\alpha_1(n-3)\\&+\lceil \gamma(n-2)\gamma(n-2)\gamma\quad(n)\\&\lfloor_{21}\qquad\qquad 10\qquad\qquad 00\end{aligned}$$

$$\begin{aligned}&+\gamma_{23}(n-2)\gamma_{31}(n-1)\gamma_{10}(n)\rceil\rfloor\alpha_2\,(n-3)\\&+\lceil\lfloor \gamma_{31}(n-2)\gamma_{10}(n-1)\gamma_{00}(n)\\&+\gamma_{33}(n-2)\gamma_{31}(n-1)\gamma_{10}(n)\rceil\rfloor\alpha_3(n-3).\end{aligned}\qquad\text{(III.6)}$$

Nous supposons que à l'instant « n -3 », α_3 (n-3) est forcé à zéro avec le mécanisme M-BCJR. Cet événement change les quatre alphas α_0 (n), α_1 (n), α_2 (n) and, α_3 (n) avec des quantités différentes. Ce phénomène génère un concurrent dans le décodage M-BCJR [25].

La figure III.11 montre un exemple de la différence $\Delta\alpha_m$(n) entre les valeurs alpha réelles du MAP-BCJR complet et celles de l'algorithme M-BCJR pour le treillis de la figure III.3. Nous utilisons SNR (E_b/N_0) égal à 0 dB. Le α_3(100) est forcé à zéro. Il affecte ensuite près de 15 alphas pour tous les états avec des amplitudes différentes. Ces fluctuations sont la raison de l'apparition des concurents.

Ces remarques sur la possibilité d'erreur donnent une idée sur la prédiction des moments d'erreur lors du décodage : La présence des concurrents indique la possibilité d'une décision erronée. Le critère de prédiction peut être : si le concurrent le plus probable dépasse un seuil « Th », alors, on peut appliquer l'algorithme Z-MAP en augmentant la complexité autour de cette instant. [25]

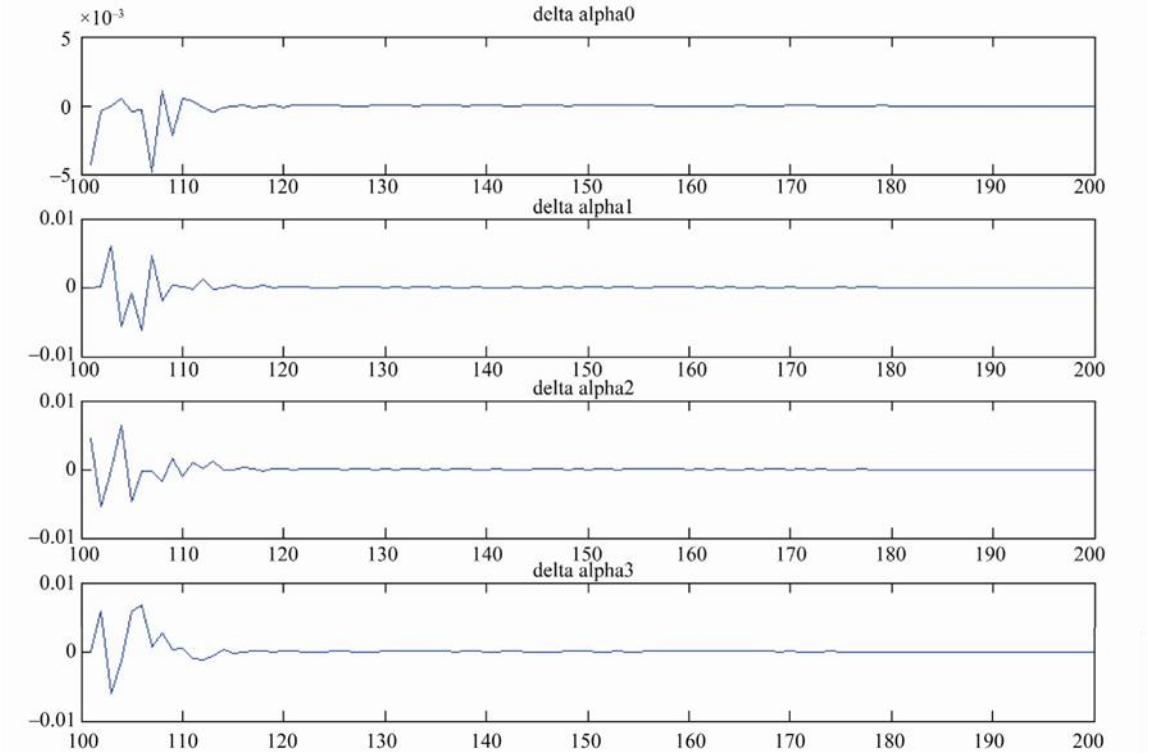

Figure III.11 : Différences entre les valeurs alpha réelles du MAP-BCJR complet et celles du M-BCJR [25].

III.7.c L'algorithme Z-MAP :

Lorsque nous localisons les instants d'erreurs, nous pouvant supprimer la raison de cette dégradation. L'algorithme Z-MAP [25] consiste à augmenter le nombre d'états autour de cet instant.

On doit alors fixer un seuil "Th" pour détecter l'événement de présence d'erreur et un retour "r" en arrière. L'algorithme Z-MAP, consiste à :

1) Localiser le moment d'erreur lors de la progression du M-BCJR dans une direction. Prenons par exemple le sens direct, c'est-à-dire lors du calcul des alphas. Prenons"i " ce moment.

2) Faire un retour avec "i-r" dans le treillis.

3) Augmentez M à M ', avec M'> M.

4) Cette augmentation ne doit pas dépasser "2 * r".

L'exemple suivant (Figure III.12(a)) montre les états survivants lors de l'exécution du 2-BCJR d'un treillis à 4 états.

Supposons qu'une erreur soit localisée à l'instant 5, et prenons r = 2. Nous montrons en seconde partie (Figure III.12(b)) la procédure Z-MAP [25].

0

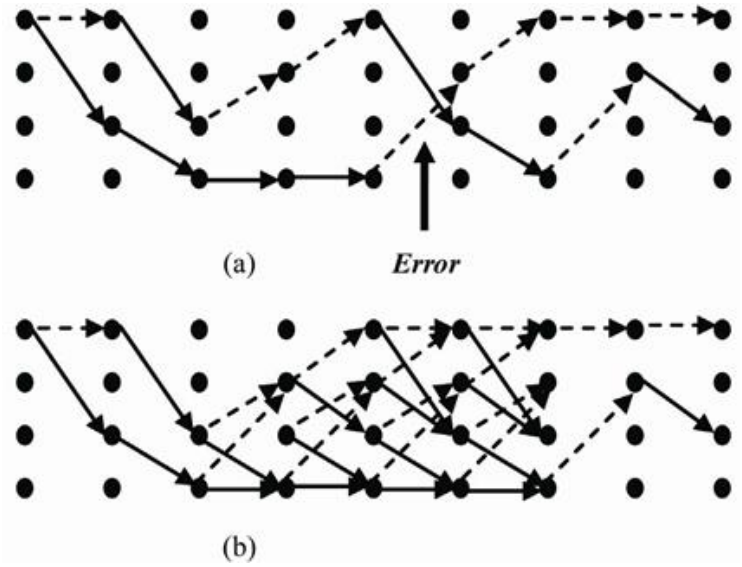

Figure III.12 : Principe de l'algorithme Z-MAP [2

CHAPITR IV

Résultats de Simulation

IV.1.Introduction :

Les algorithmes de décodage sont surtout caractérisés par leur taux d'erreur binaire TEB. Le but de ce chapitre est de montrer la capacité de correction des algorithmes utilisés pour réduire la complexité. Il est important de noter que les simulations sont exécutées pour un canal à bruit blanc gaussien additif AWGN. Cette étude nous permet d'analyser et d'évaluer l'influence ou l'effet du décodeur utilisant l'algorithme Log-MAP (version simplifiée de l'algorithme MAP) basé sur la stratégie de réduction de la complexité T-BCJR.

IV.2. Outil informatique utilisé :

IV.2.1 Matlab :

En ingénierie, la simulation est un moyen efficace et économique, couramment utilisé pour faire des études préliminaires et/ou comparatives, tant au stade du développement (Conception), qu'au cours du fonctionnement normal des systèmes. Actuellement, plusieurs outils de simulation, parmi lesquels MATLAB/SIMULINK, sont utilisés dans l'industrie et dans les milieux universitaires et de recherche. Dans ce travail, nous présenterons l'évaluation des performances des Turbo décodeurs T-Log-MAP (TBCJR-Log MAP) dans une chaine de transmission numérique.

IV.2.2. Les algorithmes utilisés :

A l'aide d'une simulation sur Matlab, il nous sera possible d'étudier les performances sur un canal gaussien. Nous avons implémenté les algorithmes de décodage MAP, Log-MAP, Max-Log-MAP et T-BCJR utilisés en turbo traitement, en vue d'évaluer leur capacité de décodage, leurs limites de performances par simulation ainsi que l'influence des éléments de la structure de turbo décodage, sur l'évolution de ces performances. L'algorithme Log-MAP présente le meilleur compromis, performance-complexité de calcul, apporté par la simplification de l'algorithme optimal d'origine MAP.

IV.3.Performances d'un Turbo décodeur [1, 5/7] :

IV.3.1.Condition de simulation :

Nous avons utilisés les paramètres suivants :

Turbo codeur : $[1, 5/7]_{oct}$

Rendement R=1/3.

Nombre de paquets = 5000.

RSB minimal = 0 dB.

RSB maximal = 1.4 dB.

Longueur des trames L=1000.

Algorithme de décodage : MAP [14].

IV.3.2.Résultats de simulation :

Les performances du turbo décodeur sont largement déterminées par l'ensemble de paramètres suivant : le nombre de paquet (trame), la longueur de trame (nombre de bits transmis), le rendement, la longueur de contrainte et l'algorithme de décodage. Afin d'analyser les performances de ces paramètres, une suite de simulations a été réalisées afin d'évaluer le taux d'erreur binaire en fonction du rapport signal à bruit SNR (E_b/N_0).

La figure suivante montre les performances en termes de Taux d'Erreur Binaire TEB du turbo décodeur MAP qui utilise 7 itérations. Le turbo codeur utilisé est formé de deux codeur récursifs systématiques RSC de séquences génératrices $[1, 5/7]_{oct}$ en parallèle.

En observant la figure ci-dessous, nous constatons une amélioration du taux d'erreur binaire TEB après chaque itération. Le TEB atteint $4 \ \ 10^{-5}$ au rapport signal sur bruit RSB = 1.4 dB.

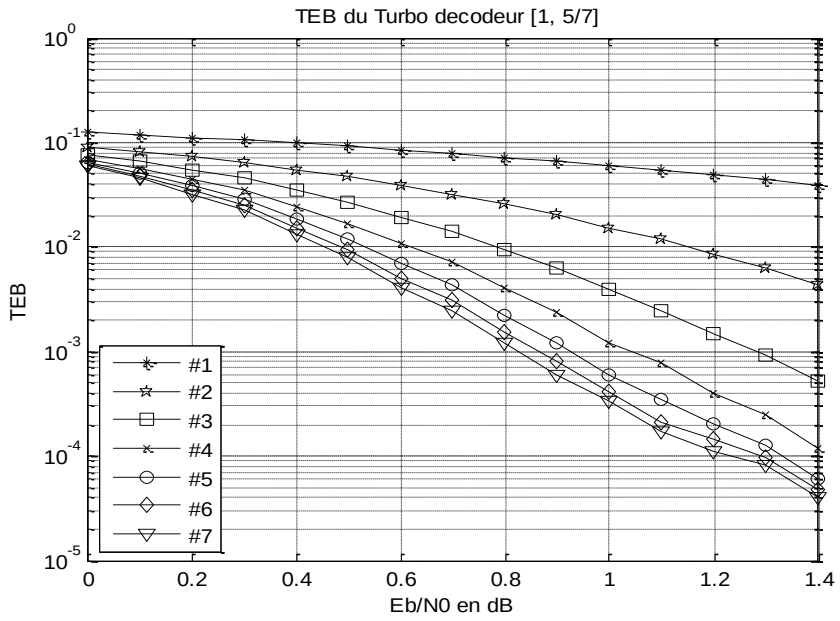

Figure IV.1 : Performances d'un Turbo décodeur MAP [1, 5/7].

IV.3.3 Interprétation des résultats :

Figure IV.1. Montre les performances du turbo décodeur MAP avec un entrelaceur pseudo aléatoire (de MATLAB) de taille 1000. La courbe de performances montre le taux d'erreurs binaire TEB en fonction du rapport signal sur bruit RSB (E_b/N_0).

Au fil des itérations, les estimations des bits d'informations sont améliorées, plus d'erreurs sont corrigées et les performances en termes de taux d'erreurs binaires 'TEB' sont devenues meilleures.

Aux forts rapports signal sur bruit RSB, les courbes sont devenues plates et la vitesse de convergence a diminué. Ce phénomène est appelé 'Error Floor' ou plancher d'erreurs. Cette zone peut empêcher l'utilisation de turbo codes dans des contextes nécessitant de très faibles taux d'erreurs.

IV.4. Turbo décodeur Log-MAP :

Dans ce cas, nous utilisons l'algorithme Log-MAP (Chapitre 2) dans les deux décodeurs Turbo d'une itération. En observant la figure ci-dessous, nous constatons que la dégradation par rapport au turbo décodeur MAP (Figure IV.1.) est très faible. L'algorithme Log-MAP assure un bon compromis Qualité-Complexité. Il assure les mêmes performances de l'algorithme MAP avec une faible complexité calculatoire.

IV.4.1.Conditions de simulation :

Turbo codeur : [1, 5/7]

Rendement R=1/3.

Nombre de paquets = 5000.0.

RSB minimal = 0 dB.

RSB maximal = 1.4 dB.

Longueur des trames L=1000.

Algorithme de décodage : Log-MAP.

IV.4.2.Performance d'un Turbo Décodeur Log-MAP [1, 5/7] :

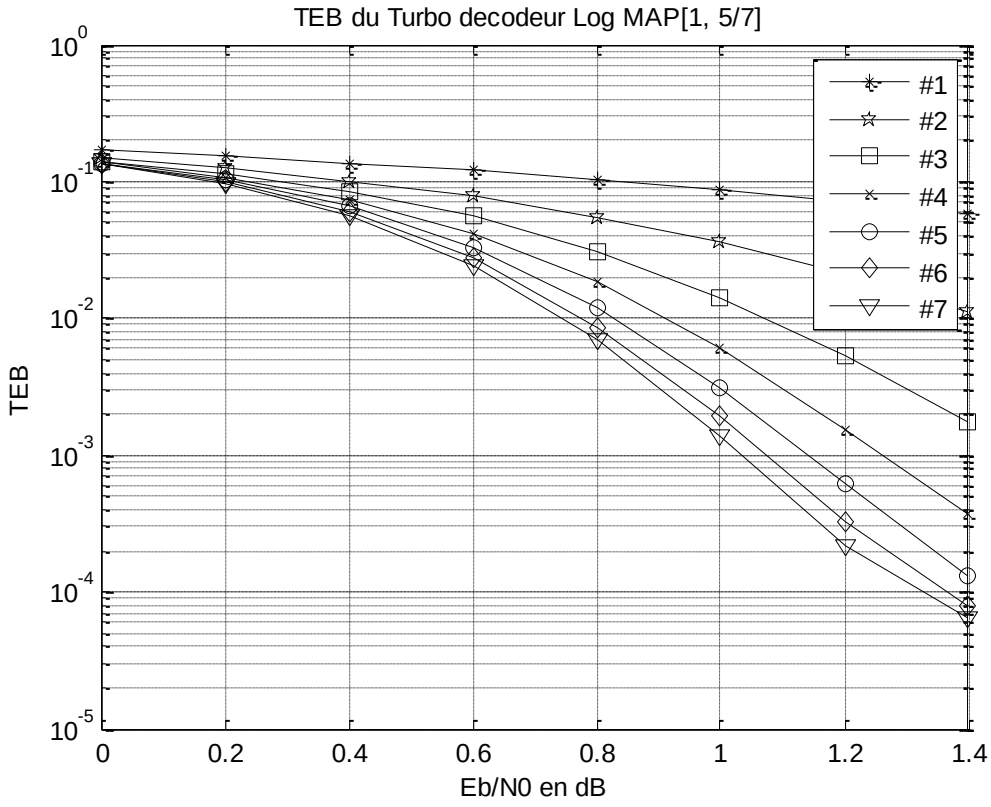

Figure IV.2: Performance d'un turbo-décodeur Log-MAP.

Commentaire :

Dans cette Figure IV.2, nous remarquons le déclenchement du Turbo décodage au point de RSB = 0.2 dB. A partir de ce point, et après chaque itération, le taux d'erreur binaire TEB diminue. Il atteint TEB = $7*10^{-5}$au RSB = 1.4 après la 7ème itération.

IV.5. Turbo décodeur T-Log-MAP (TBCJR-Log-MAP) :

IV.5.1.Condition de simulation :

Turbo codeur : [1, 5/7] oct

Rendement R=1/3.

Nombre de paquets = 5000.

RSB minimal = 0 dB.

RSB maximal = 1.2 dB.

Longueur des trames L=1000.

T= log (0.01) / T= log (0.001) / T= log (0.0001)

Algorithme de décodage : T-Log-MAP (TBCJR-Log-MAP)

IV.5.2.Performance d'un Turbo Décodeur T-Log-MAP [1, 5/7] :

IV.5.2.A l'algorithme T-Log MAP [1, 5/7] avec seuil T=log (0.01) :

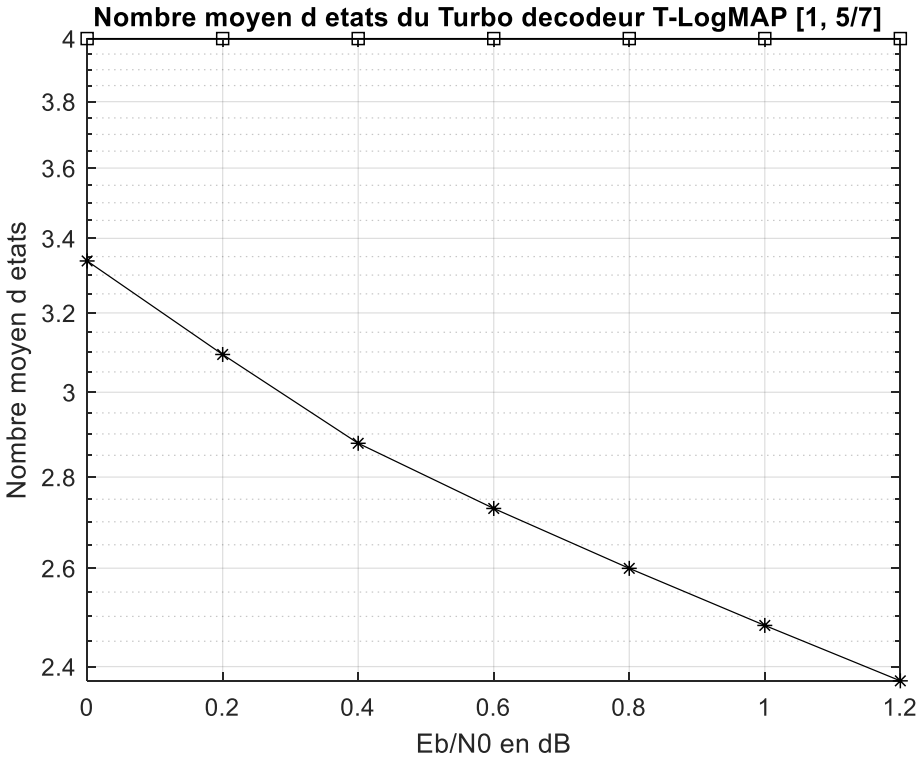

Figure IV.3: Performance d'un turbo–décodeur T-Log-MAP [1, 5/7], R=1/3, N=5000, T= log (0.01).

Commentaire :

La (Fig IV.3) présente le nombre moyen d'états du turbo décodeur T-Log MAP [1, 5/7]. On remarque sur cette figure que le nombre moyen d'états décroit avec l'augmentation du Rapport Signal sur Bruit RSB en dB et atteint 2.4 pour le rapport signal sur bruit RSB (E_b/N_0) dB = 1.2 dB.

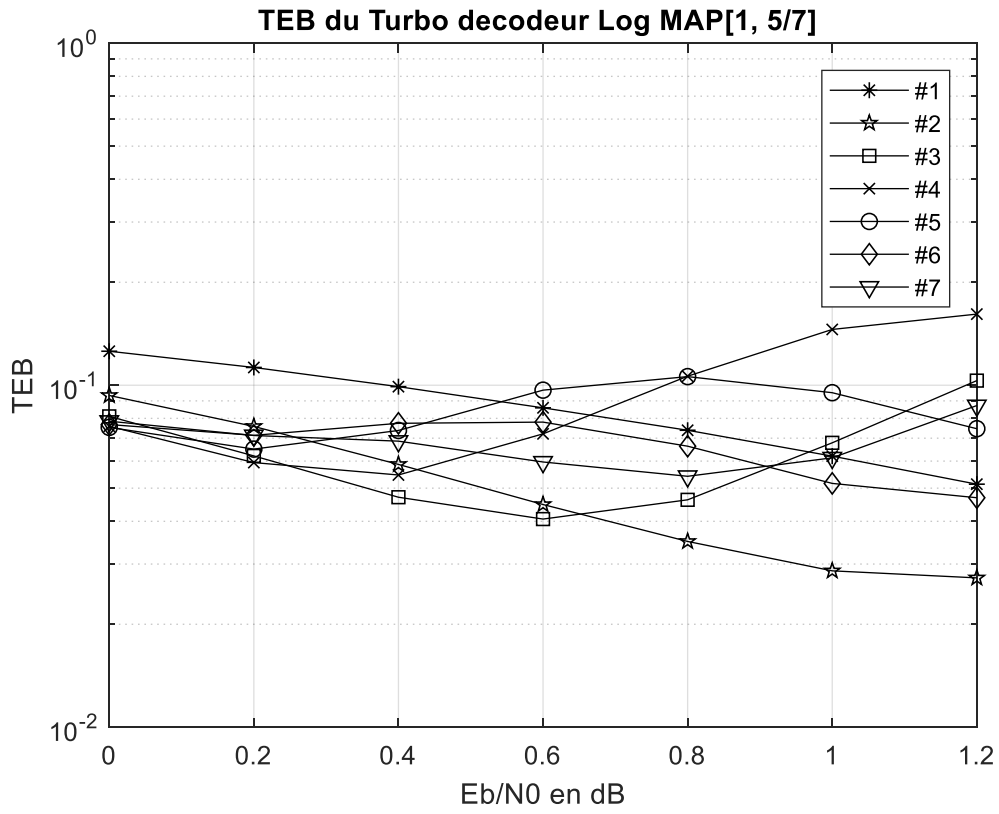

Figure IV.4: Performance d'un turbo–décodeur T-Log-MAP [1, 5/7], R=1/3, N=5000, T= log (0.01)

Commentaire :

La (Fig IV.4) présente le TEB du turbo décodeur Log MAP [1, 5/7] Cette figure montre que le turbo décodage est bloqué et les taux d'erreurs binaires TEB sont supérieurs à $2\ 10^{-2}$. La qualité de décodage est donc dégradée.

IV.5.2.B l'algorithme T-BCJR [1, 5/7] avec seuil T=log(0.001) :

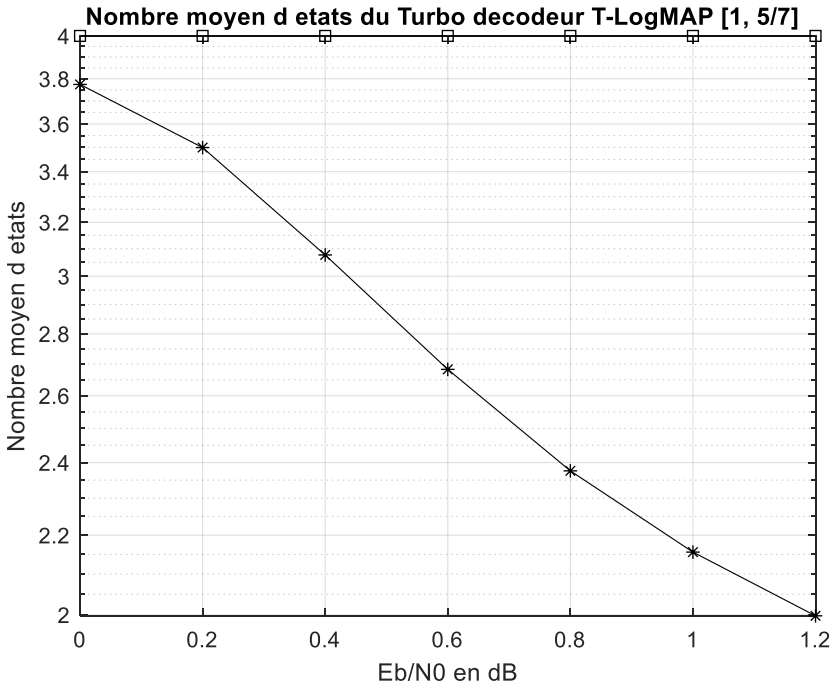

Figure IV.5 : Performance d'un turbo–décodeur T-Log-MAP [1, 5/7], R=1/3,N=5000, T= log (0.001).

Commentaire :

La (Fig IV.5) présente le nombre d'états du turbo décodeur T-Log MAP [1, 5/7]. On remarque sur La Figure IV.5 que le nombre moyen d'états décroit avec l'augmentation du Rapport Signal sur Bruit RSB en dB et atteint 2 pour le rapport signal sur bruit RSB (E_b/N_0) dB = 1.2 dB.

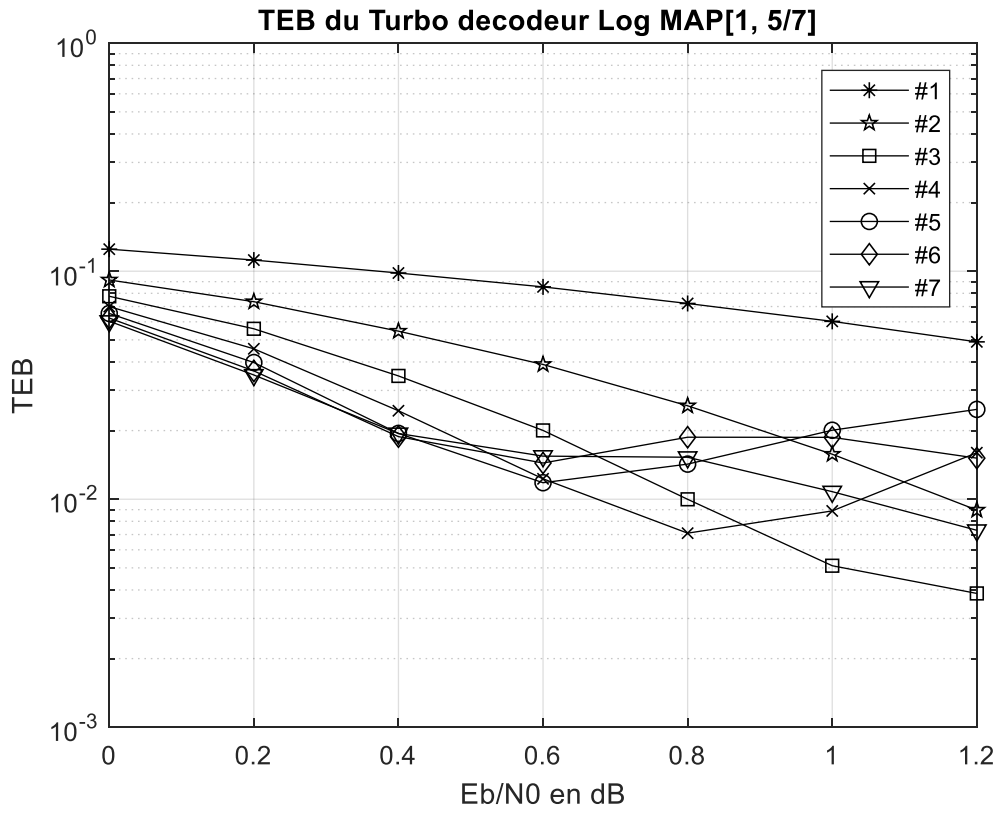

Figure IV.6: Performance d'un turbo–décodeur T-Log-MAP [1, 5/7], R=1/3,N=5000, T= log (0.001).

Commentaire :

La (Fig. IV.6) présente le TEB du turbo décodeur Log MAP [1, 5/7] Cette figure montreuse amélioration jusqu'à 3ème itération.

IV.5.2.C l'algorithme T-BCJR [1, 5/7] avec seuil T=log (0.0001) :

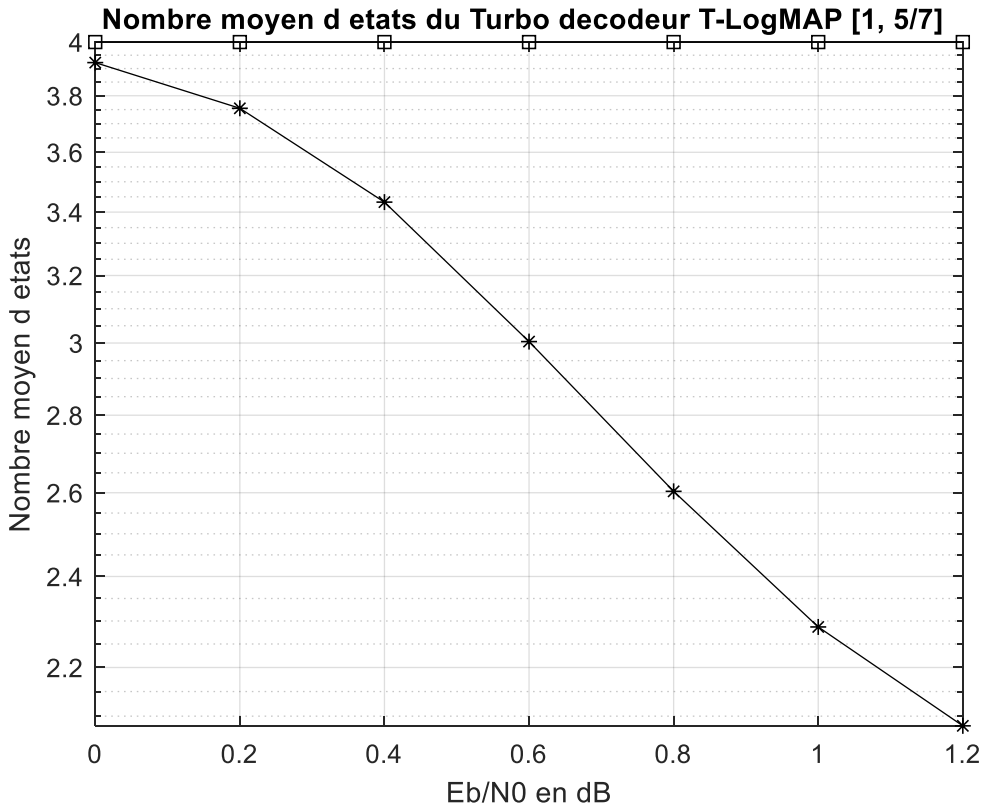

Figure IV.7: Performance d'un turbo–décodeur T-Log-MAP [1, 5/7], R=1/3,N=5000, T= log (0.0001).

Commentaire :

On remarque sur La Figure IV.7 que le nombre moyen d'états décroit avec l'augmentation du Rapport Signal sur Bruit RSB en dB et atteint la moyenne 2.25 états pour le rapport signal sur bruit RSB (E_b/N_0) dB = 1 dB.

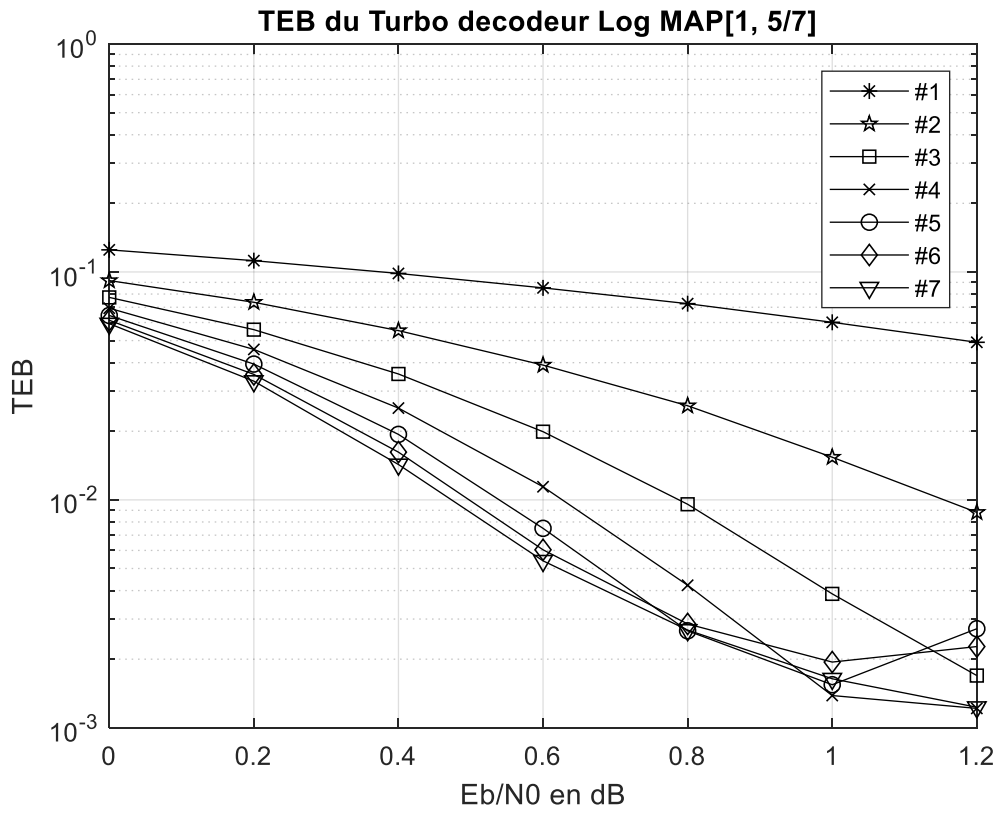

Figure IV.8: Performance d'un turbo–décodeur T-Log-MAP [1, 5/7], R=1/3,N=5000, T= log (0.0001).

Commentaire :

La (Fig. IV.6) présente le TEB du turbo décodeur T-Log MAP [1, 5/7] Cette figure montre le blocage du décodage Turbo après le 5ème itération à cause de la réduction de la complexité par le principe du TBCJR.

IV.5.3.Performance d'un Turbo Décodeur T-Log MAP [1, 17/15] :

IV.5.3.A L'algorithme T-BCJR [1, 17/15] avec seuil T=0.01 :

La longueur de contrainte (taille du registre à décalage) du codeur [1, 17/15] est égale à 4. Le nombre d'états des treillis du décodeur est égal à 2^{4-1} = 8 états (pour la modulation 2-PSK, M = 2).

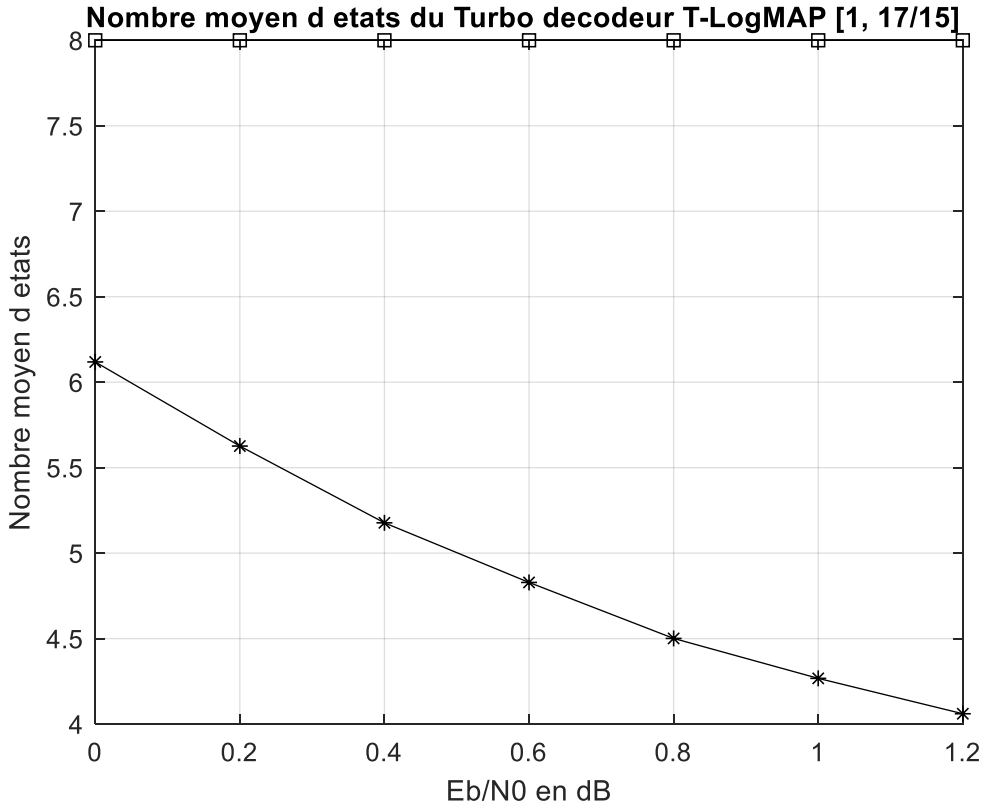

Figure IV.9: Performance d'un turbo–décodeur T-Log-MAP [1, 17/15], R=1/3,N=5000, T= log (0.01).

Commentaire :

La (Fig. IV.9) présente le nombre moyen d'états du turbo décodeur T-Log MAP [1, 17/15] On remarque sur cette figure que le nombre d'états décroit avec l'augmentation du Rapport Signal sur Bruit RSB en dB et atteint 4 pour le rapport signal sur bruit RSB (E_b/N_0) dB = 1.2 dB. On a réduit la complexité de 8 états jusqu'à 4 états.

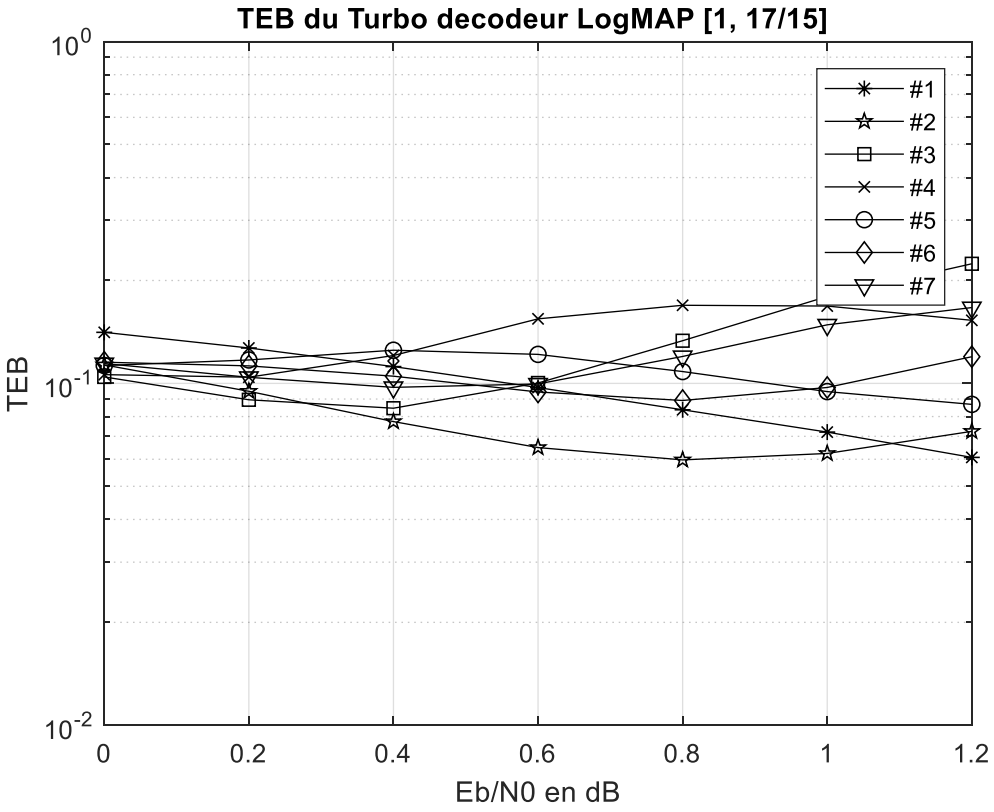

Figure IV.10: Performance d'un turbo–décodeur T-Log-MAP [1, 17/15], R=1/3,N=5000, T= log (0.01).

Commentaire :

Le processus Turbo est bloqué à cause du grand seuil log (0.01). Pour améliorer les performances on doit réduire le seuil T. Les figures suivantes montrent les performances TEB en utilisant les seuils T= log(0.001) et T = log (0.0001).

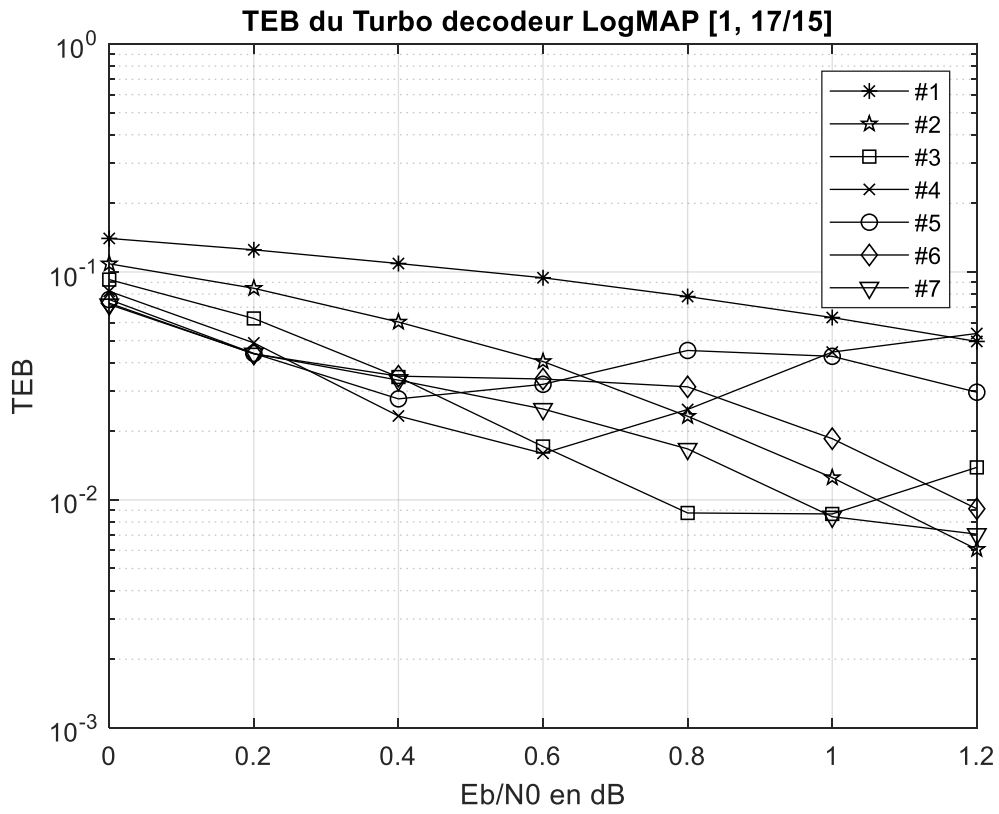

Figure IV.11: Performance d'un turbo–décodeur T-Log-MAP [1, 17/15], R=1/3,N=5000, T= log (0.001).

Le turbo décodage est bloqué après la 3ème itération et les TEB sont supérieurs à 5 10^{-3}.

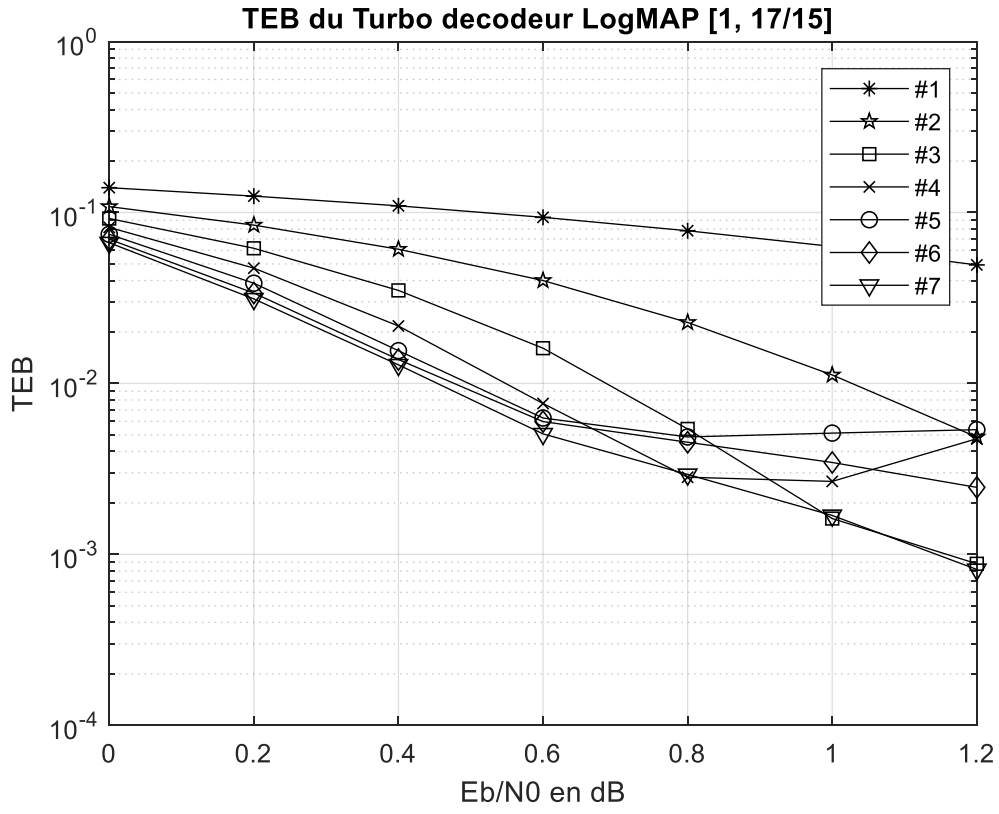

Figure IV.12: Performance d'un turbo–décodeur T-Log MAP [1, 17/15], R=1/3, N=5000, T= log (0.0001).

Le turbo décodage est bloqué après la 4ème itération. Le taux d'erreur binaire atteint 10^{-3} au rapport signal sur bruit 1.2 dB.

Conclusion Générale

Conclusion générale :

Dans un système de communications numériques, les codes correcteurs d'erreurs englobent des concepts très divers. Des codes linéaires aux turbos codes, la mise en avant de différences structurelles permet de mieux comprendre toutes les difficultés liées au décodage, qui rendent l'approche de la limite de Shannon à la fois surprenante et inespérée. Ce mémoire a été consacré à l'étude des codes turbo qui sont actuellement parmi les codes les plus performants connus à ce jour. Dans un premier temps, nous avons concentré notre étude sur des généralités et des rappels de la théorie du codage convolutif. Plus particulièrement, nous nous sommes intéressés tout d'abord aux notions fondamentales qui se rattachent aux codes Turbo. Nous avons présenté les codeurs convolutionnels systématiques récursifs ou non récursifs et leurs propriétés respectives. Par la suite, nous avons examiné les différents éléments qui composent la partie codage des codes turbo. La concaténation parallèle des codes convolutifs systématiques et récursifs, à la base des codes turbo, a été introduite. Nous avons présenté ce qui fait la spécificité des codes turbo : le décodage itératif turbo. Par la suite nous avons examiné l'importance des deux approximations proposées notamment Max Log-MAP et Log-MAP. L'algorithme TBCJR constitue une solution souple et efficace pour réduire le nombre d'états des treillis traités.

Le traitement turbo a montré que les taux d'erreur binaire TEB diminuent après chaque itération. La qualité de la transmission est améliorée en augmentant la complexité de calcul (itérations). Cependant, on ne peut pas utiliser un nombre infini d'itérations. Un mécanisme d'arrêt du traitement est donc nécessaire.

Les résultats de simulation obtenus montrent que le turbo décodeur proposé T-Log-MAP, qui associe deux techniques de réduction de complexité Log-MAP et TBCJR, fonctionne bien lorsque le seuil utilisé est bien choisi. Un mauvais seuil peut bloquer le décodage et dégrade les taux d'erreurs binaires TEB à la sortie des décodeurs.

Perspectives : Nous proposons d'appliquer l'algorithme Z-MAP au turbo décodage Log-

MAP et vérifier si cette association corrige le problème de blocage du turbo décodage.

Références Bibliographie

[1] D. Le Ruyet, et M. Pischella. Digital Communications 1: Source and Channel Coding. John Wiley & Sons, 2015.

[2] E. A. Glavieux, Channel Coding in Communication Networks: From Theory to Turbo codes. John Wiley& Sons, 2007.

[3] Claude Berrou, « code et turbo-codes », livre, école nationale supérieure des télécommunications de Bretagne, Edition Springer, France2007.

[4] Amin Zribi,« Décodage conjoint source/canal des codes entropiques.Application à la transmission d'images »,Thèse de Doctorat,Université européenne de Bretagne,07 Décembre 2010

[5] IfteneEssedik ,étude des structure d'entrelaceurs pour le codage turbo du canal pour l'optimisation des systèmes de communication par satellite ,Diplome de magister , université des science et la technologie d'oranMohmed Boudiaf.2016.

[6] Grégoy Rover,« Évaluation des entrelaceursau sein des Codes Turbo par simulations »,DIPLÔME DE MATRISE ES SCIENCES APPLIQUÉE, Université de Montréal,NOVEMBRE 2000

[7] J. G. Proakis. "Digital Communications", (3rd edition), McGraw-Hill, 1995.

[8] Michel Joindot, Alain Glavieux, « introduction au communication numérique» , IUT Ecole d'ingénieure ,Edition DUNOD , novembre 2007 .

[9] E. Biglieri, D. Declerq, et M. Fossorier, Channel coding: theory, algorithms, and applications, 1st ed. Oxford: AcademicPress, 2014.

[10] P. Thitimajshima, "Les codes Convolutifs Récursifs Systématiques et leur application à la concaténation parallèle," Thèse de doctorat, Université de Bretagne Occidentale, Brest, Décembre 1993.

[11] Hadj GuenaouiHafsa, slamaamina,« performance d'un trubo- code pralléle à rendements élevé», projet de fin d'études, Université Saad Dahleb Blida, département d'électronique, juin 2011.

[12] Thibaud Tonnellier, « Contribution à l'amélioration des performances de décodage des turbo codes :algorithmes et architecture » 'Université de Bordeaux, Électronique,5 Juillet 2017

[13] AMAMRA Imed, « Codage Canal et Techniques Efficaces de Décodage Itératif », Thèse présent Thèse Présentée pour l'Obtention du Diplôme de Doctorat, Université 20 Août 1955 de Skikda,2017 – 2018.

[14] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," IEEE Transactions on Information Theory, vol.20, no. 2, pp. 284–287, Mar 1974.

[15] W. Koch and A. Baier, "Optimum and Sub-Optimum Detection of Coded Data Disturbed by Time-Varying Inter-Symbol Interference," IEEE Globecom, pp. 1679-1684, Dec. 1990

[16] J. A. Erfanian, S. Pasupathy, and G. Gulak, "Reduced Complexity Symbol Detectors with Parallel Structures Fir ISI Channels," IEEE Trans. Commun., vol. 42, pp. 1661-1671, 1994.

[17] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," Proceedings IEEE International Conference on Communications ICC '95, Seattle, WA, USA, 1995. vol. 2, pp. 1009- 1013. doi:10.1109/ICC.1995.524253

[18] W. J. Gross and P. G. Gulak, "Simplified MAP algorithm suitable for implementation of turbo decoders," Electronic Letters, IET Journal, vol. 34, pp. 1577 – 1578, Aug 1998.

[19] S. Talakoub, L. Sabeti, B. Shahrrava, and M. Ahmadi, "A linear log-MAP algorithm for turbo decoding and turbo equalization," IEEE International Conference on Wireless And Mobile Computing, Networking And Communications, vol. 1, pp. 182 – 186, Aug 2005.

[20] Hao Wang, Hongwen Yang, and Dacheng Yang, "Improved log-MAP decoding algorithm for turbo-like codes," IEEE Communications Letters, vol. 10, pp. 186 – 188, Mar 2006.

[21] V. Franz and J. B. Anderson, "Concatenated decoding with a reduced-search BCJR algorithm," IEEE J. Select. Areas Commun., vol. 16, pp. 186–195, Feb. 1998.

[22] A.Ouardi, A. Djebbari and B. S. Bouazza, “Partial Turbo Detection with Reduced Complexity,” International Jour- nal of Electronics, Vol. 98, No. 6, 2011, pp. 825-831. doi:10.1080/00207217.2011.567040

[23] H. Yakoubi., F. Abdellaoui., « Décodage MAP à complexité réduite: AlgorithmeTBCJR », Université Dr. Tahar Moulay Saida, 2008.

[24] K. R. Narayanan, R. V. Tamma, E. Kurtas and X. Yang, “Performance Limits of Turbo Equalization Using M-BCJR Algorithm,” Proceedings of the 3rd International Symposium on Turbo Codes, 2003, pp. 2031-2035

[25] A. Ouardi, A. Djebbari, and B. S. Bouazza, “Optimal M-BCJR turbo decoding: The Z-MAP algorithm,” Wireless Engineering and Technology, vol. 2, no. 4, pp. 230–234, 2011. doi:10.4236/wet.2011.24031.