République Algérienne Démocratique et Populaire
Ministère de l'enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale Polytechnique

D.E.R DE GENIE ELECTRIQUE ET INFORMATIQUE

FILIERE : ELECTRONIQUE

PROJET DE FIN D'ETUDES

Pour l'obtention du diplôme d'ingénieur d'état
en Electronique

Thème

ETUDE ET EVALUATION DES SYSTEMES DE COMMUNICATION NUMERIQUE UTILISANT LA MODULATION PAR CODES EN TREILLIS T C M

Proposé et Dirigé par :
M. Z. TERRA
M. B. DERRAS

Etudié par :
CHEMSA ALI
GHRISSI KAMEL

Promotion : *Octobre 1997*

E.N.P. 10, AVENUE HASSEN BADI - EL-HARRACH - ALGER

République Algérienne Démocratique et Populaire
Ministère de l'enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale Polytechnique

D.E.R DE GENIE ELECTRIQUE ET INFORMATIQUE

FILIERE : ELECTRONIQUE

PROJET DE FIN D'ETUDES

Pour l'obtention du diplôme d'ingénieur d'état
en Electronique

Thème

ETUDE ET EVALUATION DES SYSTEMES DE COMMUNICATION NUMERIQUE UTILISANT LA MODULATION PAR CODES EN TREILLIS T C M

Proposé et Dirigé par :
M. Z. TERRA
M. B. DERRAS

Etudié par :
CHEMSA ALI
GHRISSI KAMEL

Promotion : *Octobre 1997*

E.N.P. 10, AVENUE HASSEN BADI - EL-HARRACH - ALGER

ERRATA

PAGE	LIGNE	ERREUR	CORRECTION
1	17	les techniques	ces techniques
1	21	modélisation	modulation
4	FIG 1.1	distinataire	destinataire
4	7	issu du décodeur de source	issu du codeur de source
4	14	le décodeur de canal de source	les décodeurs, de canal, de source
5	22	le canal à l'entrée	le canal à entrée
7	12	un canal statonnaire	un canal stationnaire
7	20	un décodage	un codage
1	20	général 2^k.....et 2^k	général 2^m.....et 2^m
19	03	$\omega_c=\frac{2\pi n_v}{T}$	$\omega_c=\frac{2\pi n_c}{T}$
19	12	sont illustrées	sont illustrés
21	10	espacées	espacés
21	10	égale	égal
27	05	mobiles	mobile
27	11	trancodeur	transcodeur
27	14	trancodeur	transcodeur
27	21	trancodeur	transcodeur
27	26	de Ungerboeck	d' Ungerboeck
27	33	messages	message
29	11	specral	spectrale
29	12	le décodage	la décision
31	03	2 bit /T	2 bits /T
32	06	Quelque fois	Quelquefois
32	18	où d	où d_0
32	19	C0 et C1	C0 et C2
32	20	soit $d_2 = \sqrt{2}\ d_1 = d_0$	soit $d_2 = \sqrt{2}\ d_1 = 2d_0$
32	32	et sélectionnés par m'+1 bits codés	et sélectionnent un point dans le sous-ensemble sélectionné par les m'+1 bits codés
34	05	aux séquence	aux séquences
34	06	trés importante	trés important
34	13	est appliquée	est appliqué
34	23	illustrée	illustré
45	22	la même état	le même état
62	16	la séquence minimum	la séquence
62	24	parmi	parmis
62	28	tous événements	tous les événements
66	10	simples	simple
89	26	coséquence	conséquence

ملخص :

لقد درسنا في هذه المذكرة "الموائمة المشفرة بالشبكية" تتمثل هذه التقنية في دمج المشفر مع الموائم، و إعتبارهما كعنصر واحد لأن هدفها الوحيد هو تعظيم المسافة الإقليدية في فضاء الإشارات و هذا تجنبا لأخطاء الإتصال بقدر المستطاع.

إن إستعمال التشفير في هذه التقنية يظفي عليها ميزات جيدة حتى و لو كان عدد نقط السحابة كبير، و هذا ما يميزها عن التقنية التي تستعمل التشفير مستقل عن الموائمة.

لإجلاء ذلك دعمنا هذه الدراسة ببعض المحاكيات الرقمية، آخذين الأنظمة الغير مشفرة كمرجع للمقارنة.

المصطلحات الأساسية :

سحابة – محول – مشفر – شبكية – قائمة – مسافة طليقة أو حدية – مشفر ملتف – مدى التقرير.

RÉSUMÉ :

Dans cette mémoire, nous avons étudié la "La Modulation Codée en treillis (TCM)", cette dernière consiste à considérée le codage et la modulation comme une seule entité. Son objectif principale est la maximisation de la distance euclidienne minimale directement dans l'espace des signaux, pour éviter les erreurs de transmission.

L'utilisation du codage dans la TCM présente des résultats meilleurs même si le nombre des points de la constellation est important, ce qui caractérise de ce qui utilise un codage indépendant à les modulation.

Les résultats de la simulation vient pour confirmer les résultats théoriques.

Mots-Clés :

Constellation - Transcodeur - Codeur - Treillis - Alphabet - Distance libre ou limite - Codeur convolutif - Profondeur de décision

ABSTRACT :

In this work we had trited the "Trellis Coded Modulation (TCM)". In this technique, coding and modulation are considered as one entity. Its main goal is the maximisation of the Euclidean distance in signal space to avoid error transmission.

Using coding in the TCM, yields good performance ever if the points constellation number is importunate.

This latter characterises the studied technique from others using coding independent of modulation.

Simulation results gives support to theory results.

Key words :

Constellation - Mapper - Coder - Trellis - Alphabet - Free distance - Convolutional coder - Depth of decision.

الإهداء

أهدي عملي هذا ، إلى قرة عيني ، وولي نعمتي سيدي محمد البشير بن سيدي العيد التجاني .

"علي"

أهدي عملي المتواضع هذا . إلى والديّ العزيزين ، وإلى كل أولاد صدقاء البعيد و القريب .

"كمال"

شكرات

هذا العمل اقترح وتبع من طرف الأستاذين "دراس بلقاسم" و "زيدان تيرا" فلهم كل الشكر والتقدير على مجهوداتهم ومساعدتهم لنا طوال مدة التحضير لإنجاز هذا العمل.

كما نوجه شكرنا إلى كل أساتذة المدرسة الوطنية المتعددة التقنيات الذين ساهموا في تكويننا.

نوجه شكرنا كذلك إلى زملائنا الطلبة الذين أعدونا يد المساعدة وخصه بالذكر الزملاء:

تيجاني زكريا، تيجاني الصولي، تيجاني عبد الكريم، تيجاني عبد الله، تيجاني طه، متمنيا للجميع كل التوفيق والسداد.

على كمال

SOMMAIRE

INTRODUCTION

INTRODUCTION

Par définition une transmission numérique permet d'acheminer des messages de nature numérique entre une source et un destinataire . Cette source délivre, avec un certain rythme, une suite de symboles qui prennent leurs valeurs dans un ensemble fini appelé alphabet . Le caractère numérique de la transmission se manifeste de manière évidente dans le fonctionnement du récepteur lors de l'échantillonnage et de la prise de décision . Il est beaucoup moins apparent au niveau du signal transmis qui présente plus nettement le caractère discret de la source .

Par une convention préalable, le destinataire a la connaissance de l'alphabet utilisé par la source . Il peut donc interpréter l'information qu'il reçoit en fonction de cet alphabet. Il compare les signaux reçus (déformés et perturbés par la transmission dans le canal) à la liste de caractères possibles et il déduit par une « décision » lequel de ces caractères est le plus probablement à l'origine du signal reçu .

La transmission de l'information entre la source et le destinataire ne s'effectue pas sans risques d'erreurs à cause du bruit de distorsion du canal . Afin de remédier à cela et donner une grande fiabilité au système de transmission, il existe plusieurs techniques de protections de l'information se basant toutes sur le principe d'ajouter une certaine redondance au message original avant la transmission à l'aide d'un codeur qui maximise la distance de Hamming entre les mots-code. Mais le fait d'ajouter de la redondance à l'information , augmente la bande passante du signal original et diminue ainsi l'efficacité de l'utilisation de la bande de fréquence allouée à la transmission car les techniques traitent les fonctions de codage et de modulation comme deux entité séparées et indépendantes . L'optimisation du codage au sens de la distance de Hamming ne donnait pas alors de bons résultats lorsque la modulation n'était pas une modulation à deux ou à quatre états de phase. C'est G.Ungerboeck qui est arrivé à une solution pratique qui permet de tirer pleinement profit du codage avec de modélisation à grand nombre d'états. L'idée est de faire le codage directement dans l'espace des signaux et de l'optimiser au sens de la distance euclidienne . Les fonctions de codage et de modulation ne forment plus qu'une seule et unique entité dite *modulation codée en treillis* (Ang : Treillis Coded Modulation TCM) cette dernière, fait l'objet de notre travail.

Le premier chapitre consiste en des généralités sur le codage convolutif et la modulation .

Le second chapitre est réservé à la théorie des modulations codées en treillis . Le principe de ces modulations codées , les règles heuristiques de la partition d'ensemble et quelques notions de base sont exposés . Une approche du sujet y est exposée aussi . Cette approche est basée sur le fait qu'un codeur en treillis peut être vu comme une machine à nombre d'état fini (Ang: Finite State Machine FSM) dont l'évolution d'un état à un autre dépend de l'état de départ et l'entrée présente de la machine . A partir de cette approche, une caractéristique très connue d'une TCM y est tiré, c'est la fonction de transfert

qui joue un rôle important dans le calcul des performances. L'algorithme de décodage de Viterbi est détaillée, pour un décodeur à maximum de vraisemblance . Enfin quelques applications sont exposées.

Le troisième chapitre est consacré à la simulation et l'évaluation des performances de TCM , en fonction de leurs paramètres (comme le type de modulation). Pour la programmation on utilisera le MATLAB sous windows.

Deux annexes A et B contenant le détail de la théorie de la séquence pseudo-aléatoire et listing des programmes de la simulation réalisée respectivement, sont présentés à la fin de ce travail.

CHAPITRE I :
GENERALITES SUR LES CODES CONVOLUTIFS ET LAMODULATION

CHAPITRE I

GENERALITES SUR LES CODES CONVOLUTIFS ET LA MODULATION

1.1 Introduction

La tâche de la conception d'un système de communication numérique est de réaliser un système efficace raisonnable pour la transmission d'information à partir d'une source à un débit et un niveau de fiabilité qui sont acceptables pour l'utilisateur.

Les deux paramètres clés lors de la conception d'un tel système sont

- la puissance du signal transmis ;
- la bande passante du canal.

Ces deux paramètres ensemble avec la densité spectrale de puissance du bruit du récepteur, déterminent le rapport de l'énergie du signal par bit sur la densité spectrale de puissance du bruit E_b/N_0.

Pour un rapport E_b/N_0 fixé, la seule façon pratique d'améliorer la qualité de transmission est d'utiliser le codage de canal.

Une autre motivation pratique pour l'utilisation de ce type de codage est de réduire le rapport E_b/N_0 exigé, ce qui permet aussi de réduire la taille de l'antenne et par conséquent le coût de la transmission.

Le codage de canal consiste à ajouter au message à transmettre des symboles de contrôle (non informatifs) suivant une loi donnée et à venir vérifier au décodage que cette loi est respectée. Si c'est le cas, on considère qu'il n'y a pas eu d'erreurs lors de la transmission. Dans le cas contraire, on détecte la présence d'erreurs que l'on peut éventuellement corriger.

1.2 Modèle d'un système de communication numérique [6]

Le modèle est présenté par la figure 1.1.

La transmission d'information numérique actuelle à travers un canal réel est accomplie par un modem (modulateur et démodulateur), qui peut opérer à un débit d'information numérique R..

La probabilité d'erreur P_e à la réception, en utilisant seulement le modem, dépend du débit R et du rapport signal sur bruit. Cette probabilité dépasse une valeur inférieure prescrite chaque fois qu'on

augmente le débit R, d'où le besoin d'un recours à la conception des codeurs et des décodeurs de canaux afin d'obtenir des probabilités d'erreurs acceptables.

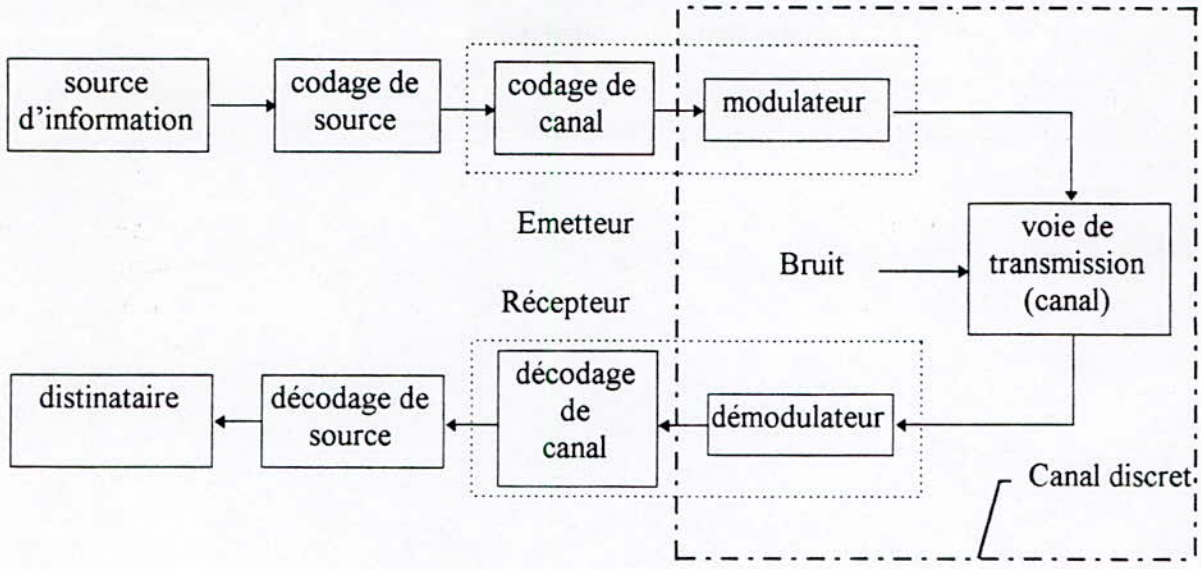


Figure 1.1 Modèle d'un système de communication numérique.

Nous décrivons ici brièvement les différentes fonctions d'un système de communication numérique

- La source d'information délivre le message contenant l'information à transmettre ;
- Le codeur de source supprime toute redondance dans le message ;
- Le codeur de canal introduit une redondance contrôlée dans le message issu du décodeur de source.

  Cette redondance connue aussi au niveau du récepteur, permettra la détection et/ou la correction des erreurs dues au bruit inévitable ;
- Le modulateur a pour fonction principale de modifier le signal message en une forme plus adaptée à la transmission à travers le canal ;
- Le canal représente le milieu physique entre l'émetteur et le récepteur. Il peut être l'espace libre, un câble coaxial ou une fibre optique ;
- Le démodulateur et le décodeur de canal de source réalisent, respectivement, les fonctions inverses du modulateur, du codeur de canal et du codeur de source afin de restituer le signal original.

1.3 Modélisation d'un canal [10]

Deux modèles de canal sont souvent adoptés selon que la sortie du canal sera prise après ou avant le circuit de décision du démodulateur et selon le type de bruit affectant le canal.

Dans le premier cas, l'entrée et la sortie du canal seront discrètes, alors on parlera *de canal discret (canal discret = modulateur + canal réel + démodulateur)*. Dans le second cas, lorsque le bruit est un bruit blanc gaussien, on parlera de canal gaussien.

1.3.1 Canal discret

Un canal discret peut être défini par ses probabilités de transition p_{ij}^k où

$$\text{avec} \quad \begin{aligned} & p_{ij}^k = Pr\left\{Y_k = y_j / X_k = x_i\right\} \\ & \sum_{j=0}^{m-1} p_{ij}^k = 1, \forall i \end{aligned} \tag{1.1}$$

où X_k et Y_k représentent respectivement l'entrée et la sortie du canal à l'instant k et prennent leurs valeurs dans les ensembles : $\left[x_0,\ldots,x_i,\ldots,x_{n-1}\right] et \left[y_0,\ldots,y_j,\ldots,y_{m-1}\right]$ et, p_{ij}^k est la probabilité d'avoir y_j en sortie lorsque x_i est introduit à l'entrée (est transmis).

Lorsque les probabilités de transition sont indépendantes du temps, le canal est *stationnaire*. Si un symbole à la sortie du canal à l'instant $t=kT$ (où T est la période d'échantillonnage) ne dépend que du symbole à l'entrée du canal à l'instant $t = kT$, le canal est dit *sans mémoire*.

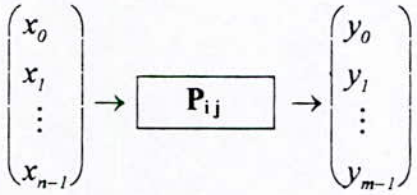


Figure 1.2 Canal discret sans mémoire.

Lorsque les symboles à l'entrée et à la sortie du canal sont binaires et que les probabilités de transition sont symétriques, c'est-à-dire

$$p_{ij} = p_{ji}, \ \forall i,j \in \{0,1\}$$

alors le canal est appelé *canal binaire symétrique*, et il est entièrement défini par sa probabilité d'erreur p d'un bit

$$p_{01} = p_{10} = p$$

$$p_{00} = p_{11} = 1 - p$$

1.3.2 Canal gaussien

Lorsque la perturbation prise en compte dans la transmission suit une loi de Gauss et que le démodulateur cohérent est synchronisé avec la porteuse et la phase de la porteuse de l'émetteur, les échantillons à l'entrée du circuit de décision suivent une loi de Gauss. Dans ce cas, le canal à l'entrée discrète et à sortie analogique est appelé *canal gaussien*. Il est défini par ses densités de probabilité conditionnelles

$$f(Y_k / X_k) = P_{Y_k / X_k = x_i}(Y_k)$$

Si le canal est stationnaire et sans mémoire, les densités de probabilité sont indépendantes du temps et la sortie du canal à l'instant $t = kT$ (où T est la période d'échantillonnage, généralement prise égale à l'unité du temps) ne dépend que du symbole d'entrée à l'instant $t = kT$.

1.4 Capacité du canal [10]

1.4.1 Canal discret stationnaire et sans mémoire

Pour définir la capacité d'un canal discret, on introduit la notion d'information mutuelle entre deux variables aléatoires. Considérons deux variables aléatoires dépendantes X et Y qui prennent, respectivement, leurs valeurs dans les ensembles suivants

$$X \in \{x_0,...,x_i,...,x_{n-1}\}$$

et

$$Y \in \{y_0,...,y_j,...,y_{m-1}\}$$

Lorsqu'il y a une certaine dépendance entre x_i et y_j la réalisation de l'événement $X = x_i$ apporte une certaine information sur la réalisation de l'événement $Y = y_j$ que l'on peut mesurer par la quantité

$$i(x_i,y_j) = log_2 \frac{P_r\{Y = y_j / X = x_i\}}{P_r\{Y = y_j\}} \qquad (1.2)$$

appelée information mutuelle entre x_i et y_j.

L'information mutuelle moyenne $I(x,y)$ est définie comme la moyenne de la quantité *i(X,Y)*

$$I(x,y) = \sum_{i=0}^{n-1}\sum_{j=0}^{m-1} Pr\{X = x_i, Y = y_j\} log_2 \frac{Pr\{Y = y_j / X = x_i\}}{Pr\{Y = y_j\}} \qquad (1.3)$$

L'information mutuelle moyenne est positive est s'annule lorsque les variables X et Y sont indépendantes. On peut montrer qu'elle peut s'exprimer en fonction de *H(X)* et de *H(X/Y)*

$$I(X,Y) = H(X) - H(X,Y)$$

où : $H(X) = -\sum_{i=1}^{n} P(x_i) log_2 P(x_i,y_j)$ est l'entropie du canal ;

et : $H(X / Y) = -\sum_{i=1}^{n}\sum_{j=1}^{m} P(x_i,y_j) log_2 P(x_i / y_j)$ est l'équivoque [16].

La *capacité* C d'un canal discret est définie comme la plus grande quantité d'information mutuelle moyenne qu'il peut transmettre.

$$C = \max_{p(x_i)} I(X,Y) \qquad (1.4)$$

où X et Y représentent respectivement l'entrée et la sortie du canal. La capacité d'un canal s'exprime en bit par seconde. Le maximum d'information mutuelle est à prendre par rapport à la distribution de probabilité

où
$$\{p_0,\ldots,p_{n-1}\}$$
$$p_i = Pr\{X_k = x_i\}$$

Pour un canal symétrique binaire, le maximum d'information mutuelle $I(X,Y)$ est obtenu lorsque les symboles 0 et 1 à l'entrée du canal sont équiprobables et sa capacité est égale à [10]

$$C = 1+(1-p)\cdot log_2(1-p)+p\cdot log_2 p \qquad (1.5)$$

où p est la probabilité d'avoir un 0 ou un 1.

1.4.2 Canal stationnaire gaussien sans mémoire

Pour un canal statonnaire gaussien sans mémoire, nous nous limiterons à donner l'expression de sa capacité C [10]

$$C = B\,T\,log_2\left(1+\frac{S}{N}\right) \qquad (1.6)$$

où B : Bande passante du canal ;

S/N : Rapport signal sur bruit en sortie du canal ;

$1/T$: Nombre de symboles transmis par unité de temps.

1.5 Théorèmes fondamentaux sur le codage du canal

1.5.1Théorème de codage de Shannon

Il est possible, en utilisant un décodage de canal approprié, de transmettre de l'information sous forme d'une suite de symboles discrets avec une probabilité d'erreur aussi faible que l'on veut si le débit d'information R est inférieure à la capacité du canal C.

Le théorème de Shannon affirme l'existence de codes dont la probabilité d'erreur est arbitrairement faible mais ne montre pas comment ces codes peuvent être construits.

Ce théorème affirme une chose toute à fait surprenante à savoir que, quelque soit le niveau des perturbations d'un canal, on peut toujours y passer des messages codés d'une manière appropriée avec une probabilité d'erreur aussi faible que l'on veut, c'est la raison pour laquelle ce théorème était la cause de l'énorme développement de la théorie des codes.

En pratique, dans tous les cas où $R < (1/2)C$, il existe des codes qui réalisent une probabilité d'erreur très faible, où R représente le débit d'information et C est la capacité du canal [16].

1.5.2 Théorème de Nyquist [16]

Le théorème de Nyquist affirme *qu'à travers un canal équivalent à un filtre passe-bas idéal avec une fréquence de coupure B, il est possible de transmettre des signaux binaires (impulsions) indépendants avec un débit de moments $R_s \leq 2B$ impulsions/seconde sans interférences entre symboles*, où R_s est donné par définition

$$R_s = \frac{nombre\ d'impulsions\ dans\ l'intervalle\ T}{T}$$

1.6 Types d' erreurs [6]

Les erreurs dans les systèmes de communication numériques sont dues au bruit du canal de transmission. Généralement, deux sortes de bruits peuvent être distinguées dans les canaux de transmission.

- Le bruit blanc gaussien : constitue le souci principal dans la conception des modulateurs, des codeurs, des décodeurs, et des démodulateurs pour la transmission de l'information numérique.
  Les erreurs de transmission introduites par le bruit blanc gaussien sont considérées comme des erreurs aléatoires.
- Le bruit impulsif : est caractérisé par des intervalles de courte durée. mais avec des amplitudes importantes du bruit. Ce type de bruit résulte de plusieurs causes naturelles ou artificielles telles que les interrupteurs de commutation, des centraux de communication , arcs électrique,...,etc. Quand un paquet d'erreurs apparaît, il affecte plusieurs symboles ou bits, et il y a d'habitude une dépendance des erreurs dans les symboles affectés. Ces erreurs apparaissent en paquets.

Les schémas de contrôle d'erreurs qui s'occupent d'erreurs aléatoires sont appelés codes correcteurs d'erreurs aléatoires, tandis que les schémas de codage conçus pour corriger les paquets d'erreurs sont appelés code correcteurs de paquets d'erreurs.

1.7 Types de codes [15]

Les codes de canal appelés aussi codes correcteurs d'erreurs sont répartis en deux catégories :
Les codes en blocs linéaires et les codes convolutifs.

Le codage en blocs consiste à associer, à chaque bloc de k bits d'information, un bloc de n bits (n > k) contenant n - k bits de redondance. Les 2^k blocs de n bits délivrés par un codeur sont appelés les mots-code.

Le rapport k/n est appelé le rendement du code.

Les opérations de codage et de décodage dans les codes en blocs se font à l'aide d'addition et de multiplications sur des éléments binaires. Ces dernières correspondant, respectivement, aux opérations logiques « OU » et « ET » exclusif .

Il existe un type spécial de code en blocs linéaires appelé codes cycliques. ce sont des codes en blocs linéaires vérifiants quelques propriétés supplémentaires [6], [11].

1.8 Codes convolutifs [15]

Un code convolutif binaire est un système à mémoire finie qui génère n bits chaque fois que l'on présente m bits d'information à son entrée. Toutefois, contrairement aux codes par blocs, les n bits de sortie ne dépendent pas seulement du bloc de m bits à l'entrée du codeur, mais aussi des D blocs précédants. Les codes convolutifs introduisent donc un effet mémoire d'ordre D, la quantité (D+1) s'appelle la longueur de contrainte du code. Le principe général du codage convolutif est illustré à la figure 1.3 [3].

Ainsi, le codeur est constitué d'un registre à m (D+1) étages qui mémorisent les derniers (D+1) blocs de m bits d'information, d'une logique combinatoire qui calcule les blocs de n bits fournis par le codeur et d'un convertisseur parallèle / série (P/S).

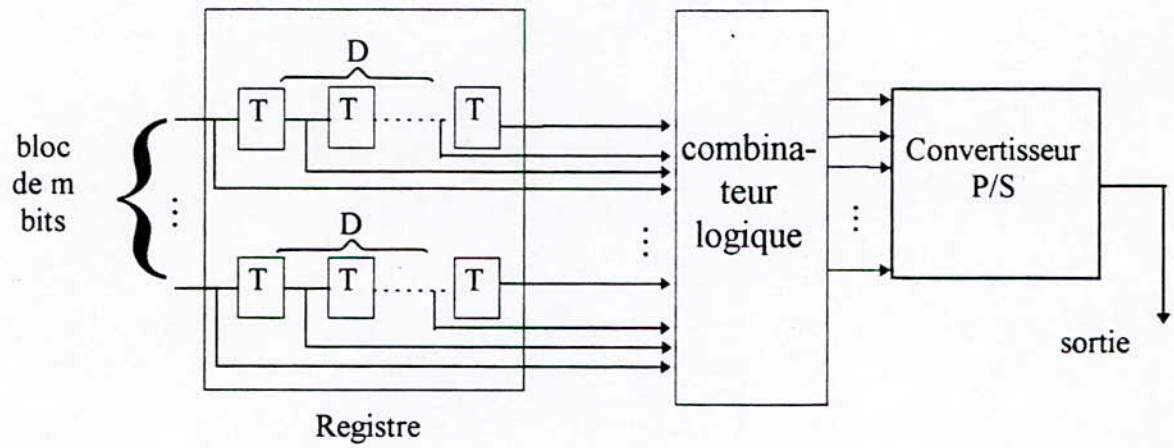


Figure 1.3 Schéma général d'un codeur convolutif.

Soit un codeur convolutif à m entrées et n sorties. Dans ce type de codes, une séquence de longueur L bits produit à la sortie une séquence codée de longueur n [(L/m) + D].

On définit le rendement (re) du codeur comme suit

$$re = \frac{L}{n[(L/m) + D]} \text{ bits / symbole.}$$

Généralement, on a L >> D, donc le rendement se simplifie à re ≈ m/n bits / symbole.

Nous allons illustrer cette famille de code en considérant un exemple de codeur convolutif de rendement re = 1/2 et de longueur de contrainte (D + 1) = 3. Son entrée est constituée par des blocs de m = 1 bit et sa sortie par des blocs de n = 2 bits.

Le caractère convolutif du codeur provient du fait que la sortie du codeur est le produit de convolution de son entrée avec sa réponse impulsionnelle combinatoire. Pour le codeur de la figure 1.4 les sorties b_k^1 *et* b_k^2 sont données par [15]

$$b_k^i = \sum_{j=0}^{2} g_{ij}\, a_{k-j}, \qquad g_{ij} \in \{0,1\}$$

Les deux séquences génératrices étant

et

$$g_1 = [g_{10}, g_{11}, g_{12}] = [1,1,1]$$
$$g_2 = [g_{20}, g_{21}, g_{22}] = [1,0,1]$$

Les séquences génératrices sont en général représentées sous une forme octale, ce qui donne $g_1 = 7$ et $g_2 = 5$ pour le code de la figure 1.4

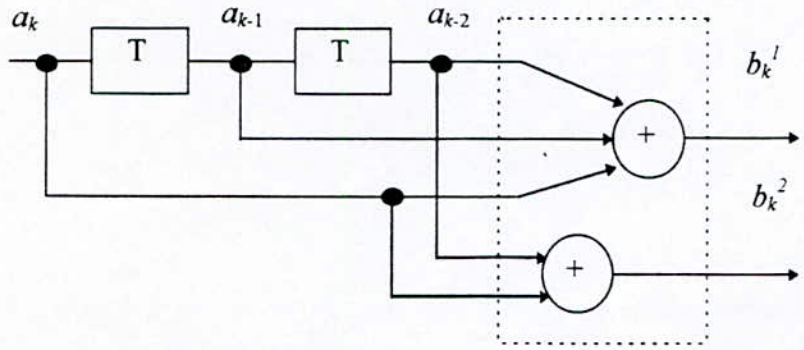


Figure 1.4 Exemple de codeur convolutif, D = 2, m = 1, et n = 2.

1.8.1 Représentation

Un codeur convolutif étant un système numérique à mémoire, il se prête mieux à une description par diagramme d'état qu'à une description algébrique qui fait intervenir une matrice génératrice. Il existe trois façons simples de décrire un code convolutif : la description par un diagramme en arbre , la description par diagramme en treillis, et enfin la description par diagramme d'état.

- ***Diagramme en arbre [15]***

Le diagramme en arbre associé au code de la figure 1.4 est illustré à la figure 1.5, où le temps s'écoule de gauche vers la droite.

Dans cet arbre, on suit une branche montante lorsque l'entrée du codeur est un 0 et une branche descendante lorsque l'entrée est un 1. Pour une séquence binaire à l'entrée du codeur, la séquence de sortie, correspondante est représentée par un chemin dans l'arbre constitué d'une suite de branches.

Chaque bloc de n = 2 bits en sortie du codeur dépend non seulement du bloc de m = 1 bit présent à son entrée, mais aussi des D = 2 blocs de m bits contenus dans sa mémoire.

Ces D·m = 2 bits définissent l'état du codeur $\sigma_k = (a_{k-1} , a_{k-2})$. Les quatre états possibles seront notés a = 00, b = 01, c = 10, et d = 11.

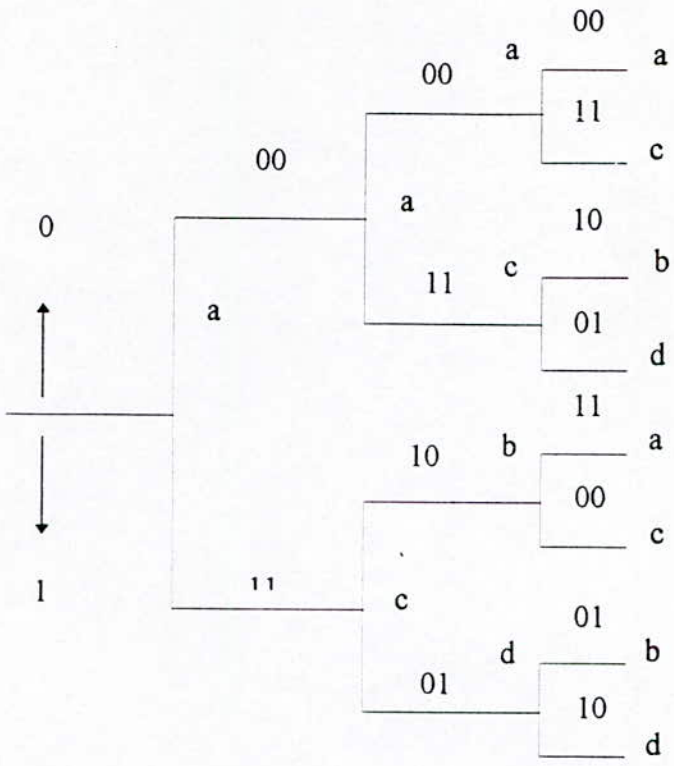


Figure 1.5 Diagramme en arbre du codeur convolutif de la figure 1.4.

Dans ce diagramme, les lettres au niveau des noeuds représentent l'état du codeur et les couples de bits sur les branches représentent la sortie. On suppose que le codeur démarre avec un état initial a = 00. Une entrée 0 fournit une sortie 00 et laisse le codeur à l'état a. Par contre, une entrée 1 fournit une sortie 11 et fait transiter le codeur à l'état c = 10. Ensuite, lorsque le codeur se trouve dans un état c, une entrée 0 fournit une sortie 10 et fait passer le codeur vers un état b et une entrée 1 fournit une sortie 01 et fait passer le codeur à l'état d.

Ainsi, au bout de trois décalage, tous les états du codeur sont atteints. Au-delà, l'arbre se répète et sa taille se multiplie par deux à chaque étage. La mémoire du codeur étant fini, il est évident que la représentation en arbre est redondante et il est possible de passer à une représentation en treillis à 4 états du codeur.

- ***Diagramme en treillis***

Le treillis à 4 états déduit de l'arbre de la figure 1.5 est donné à la figure 1.6. Comme dans la représentation en arbre, les deux bits indiqués sur les branches du treillis correspondent à la sortie du codeur.

Après (D+1) décalages, le motif du treillis se répète, quelque soit l'état initial du codeur.

Deux branches convergent vers chaque état et deux branches partent d'un état donné. D'une manière générale 2^k branches convergent vers chaque état et 2^k branches partent d'une état donné.

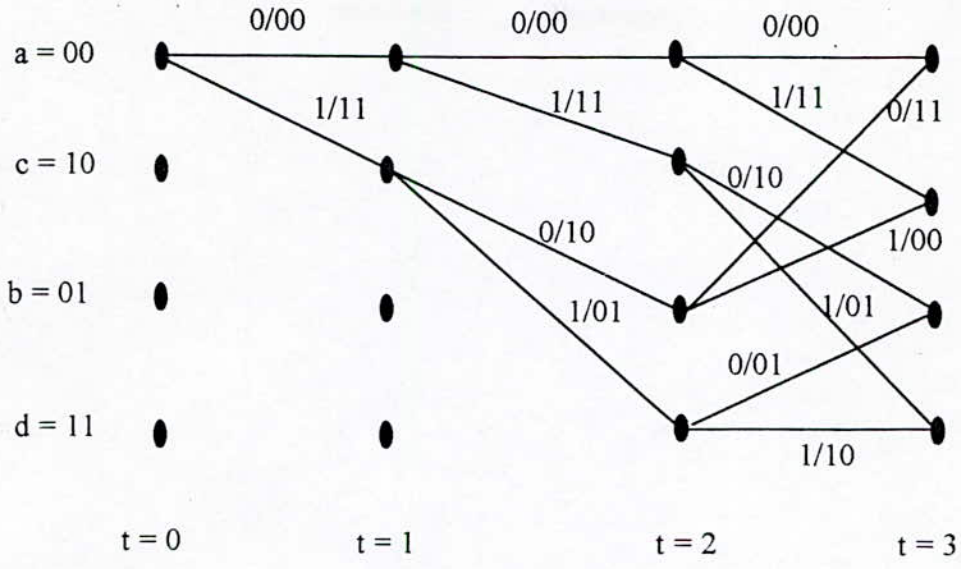


Figure 1.6 Représentation en treillis du codeur de la figure 1.4.

- ***Diagramme d'état [10]***

L'état d'un codeur est défini par le contenu de ses D mémoires. Dans l'exemple de la figure 1.4, le codeur possède quatre états que nous appelons a, b, c, et d (a = 00, b = 01, c = 10, et d = 11). Ces états sont définis par le couple $\sigma_k = (a_{k-1} , a_{k-2})$.

A chaque message constitué d'un symbole binaire a_k est associé un mot-code formé de deux symboles binaires b_k^1 *et* b_k^2. Le fonctionnement du codeur peut être représenté par le diagramme d'état de la figure 1.7. Sur ce diagramme, les états se communiquent par des branches sur lesquelles sont indiqués le bit d'entrée et le mot-code de sortie correspondant.

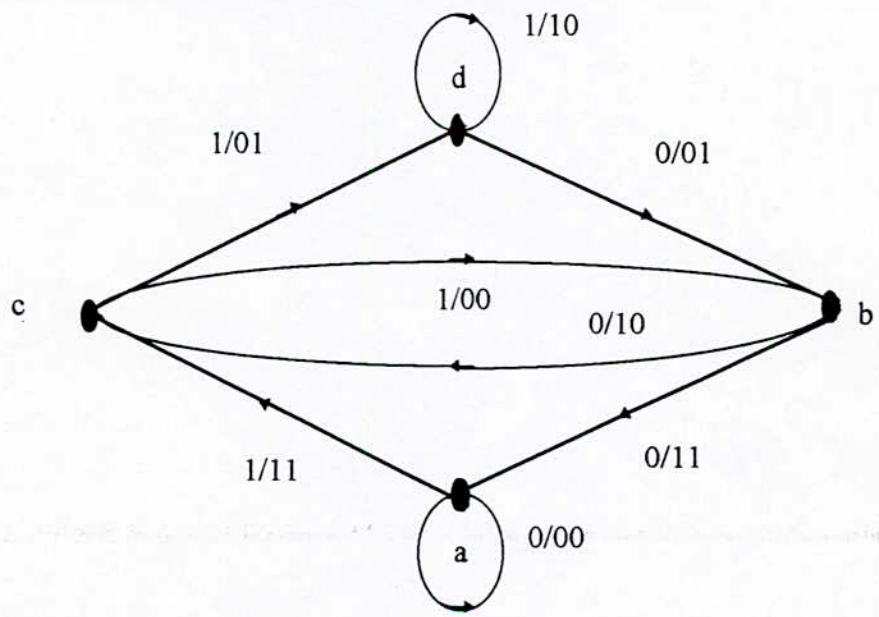


Figure 1.7 Diagramme d'état du codeur de la figure 1.4.

1.8.2 Fonction génératrice [6]

Nous pouvons calculer l'ensemble des poids de Hamming associés aux chemins du code, ou d'une façon équivalente, l'ensemble des distances entre le chemin associés à la suite codée formée par des « zéros » seulement et les autres chemins possibles, à l'aide du diagramme d'état.

Pour simplifier les choses, considérons l'exemple de la figure 1.4. Le diagramme d'état du codeur est montré sur la figure 1.7. Nous commençons par la modification du diagramme d'état de la figure 1.7 en un autre représenté par la figure 1.8.

Nous scindons en deux états l'état (a), un état initial (a_0) et un autre état final (a_1), et sa propre boucle est éliminée. Un autre changement, c'est que chaque branche est dénommée $D^dL^lI^i$, où l'exposant d sur les branches décrit le poids de Hamming de la séquence de sortie correspondant à cette branche, l'exposant i décrit le poids de Hamming de la séquence d'entrée correspondante. Donc pour l'entrée « zéro » nous avons : $I^0 = 1$, et pour l'entrée « un », nous avons : $I^l = I$. L'exposant l est toujours égal à « un » correspondant en fait à la longueur de chaque branche qui est « un ».

Par exemple, le chemin a_0 c b c d b a_1 est noté par $D^7L^6I^3$, c'est-à-dire que le poids de Hamming de la séquence de sortie correspondante est 7, la longueur du chemin est 6, et le poids de Hamming de l'entrée est 3.

Nous définissons un chemin fondamental comme le chemin qui commence en (a_0) et se termine en (a_1). Soit $T_{d,l,i}$ le nombre de chemins de (a_0) vers (a_1) noté $D^dL^lI^i$. La fonction génératrice est définie comme étant

$$T(D, L, I) = \sum_{d=1}^{\infty}\sum_{l=1}^{\infty}\sum_{i=1}^{\infty} T_{d,l,i} \cdot D^d\, L^l\, I^i \qquad (1.7)$$

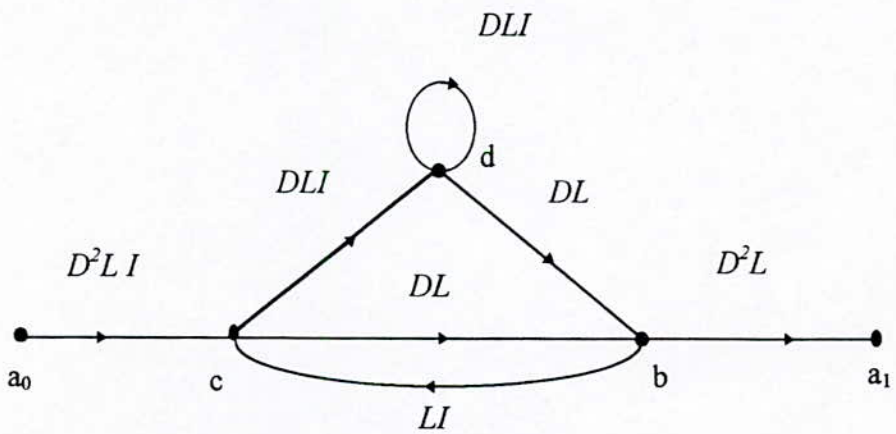


Figure 1.8 Diagramme d'état modifié.

Pour le codeur de notre exemple, nous pouvons calculer la fonction génératrice, en examinant le diagramme d'état modifié comme étant un graphe de fluence à une seule entrée et une seule sortie.

La fonction génératrice est considérée comme l'équivalent de la fonction de transfert du graphe de fluence du diagramme d'état ayant comme entrée a_0 et comme sortie a_1.

Ainsi nous traitons les noeuds comme des jonctions de sommation et les notations de branches comme des gains. Nous pouvons écrire

$$\left.\begin{aligned} X_c &= D^2 L I X_{a_0} + L I X_b \\ X_b &= D L X_c + D L X_d \\ X_d &= D L I X_c + D L I X_d \\ X_{a_1} &= D^2 I X_b \end{aligned}\right\} \qquad (1.8)$$

et

$$T(D,L,I) = X_{a_1} / X_{a_0} \qquad (1.9)$$

En résolvant le système (1.8), on trouve

$$T(D,L,I) = \frac{D^5 L^3 I}{1 - DLI(1+L)} \qquad (1.10)$$

En utilisant le développement en série de Maclaurin, on peut réécrire l'expression précédente en série de puissances

$$T(D,L,I) = D^5 L^3 I + D^6 L^4 I^2 (1+L) + D^7 L^5 I^3 (1+L)^2 + \ldots \qquad (1.11)$$

Nous pouvons faire les remarques suivantes :

- Il existe un seul chemin fondamental à la distance 5 du chemin nul ; il diverge de celui-ci sur les trois branches qui précèdent le noeud à partir duquel il se confonde avec lui. En plus, il diffère du chemin nul en un seul bit d'entrée .
- Il y a deux chemins fondamentaux à la distance 6 du chemin nul, dont l'un a divergé quatre branches en amont et l'autre 5, et les deux différent du chemin nul en deux bits d'entrée, et ainsi de suite.

En général, pour un code convolutif, la plus petite distance de Hamming de n'importe quel chemin fondamental par rapport au chemin nul est égal au distance minimale (free distance) du code. Ainsi, le code convolutif, dont la fonction génératrice est donnée par (1.10) à la distance minimale (d_{min}) qui est égale à 5, $d_{min} = 5$. Ceci entraîne que deux erreurs quelconques peuvent être corrigées, puisque la suite reçue est à la distance 2 de la suite émise, à la distance 3 au moins de toute autre suite codée possible.

1.9 Représentation vectorielle [11]

Soit $S(t)$ un signal donné défini sur $[0,T]$. On appelle une représentation vectorielle du signal $S(t)$ tout développement de la forme

$$S(t) = \sum_{k=1}^{n} \alpha_k \Phi_k(t), \quad t \in [0,T] \tag{1.12}$$

où $(\alpha_1, \alpha_2, ..., \alpha_n)^T$ constituent une représentation discrète du signal, donc $S(t)$ représente un point de coordonnées $(\alpha_1, \alpha_2, ..., \alpha_n)^T$ dans l'espace à n dimension, engendré par la base $\{\Phi_1(t), \Phi_2(t), ..., \Phi_n(t)\}$. Il est toujours préférable d'utiliser une base orthogonale ou orthonormale. Dans le cas d'une base orthonormale, on a

$$\langle \Phi_k(t), \Phi_l(t) \rangle = \int_0^T \Phi_k(t) \Phi_l^*(t) dt = \begin{cases} 1, & k = l \\ 0, & k \neq l \end{cases}, \quad k,l = 1,...,n \tag{1.13}$$

où : $\langle \cdots \rangle$ représente le produit scalaire, et les coefficients α_k sont donnés par

$$\alpha_k = \frac{\langle S(t), \Phi_k(t) \rangle}{\langle \Phi_k(t), \Phi_k(t) \rangle} = \int_0^T S(t) \Phi_k^*(t) dt \tag{1.14}$$

Dans le cas bidimensionnel avec la base $\left\{ \sqrt{\frac{2}{T}} cos(\omega_c t), \sqrt{\frac{2}{T}} sin(\omega_c t) \right\}$, tout signal $S(t)$ peut s'écrire sous la forme

$$S(t) = \alpha_1 \sqrt{\frac{2}{T}} cos(\omega_c t) + \alpha_2 \sqrt{\frac{2}{T}} sin(\omega_c t) \tag{1.15}$$

avec

$$\alpha_1 = \sqrt{\frac{2}{T}} \int_0^T S(t) cos(\omega_c t)\, dt$$

et

$$\alpha_2 = \sqrt{\frac{2}{T}} \int_0^T S(t) sin(\omega_c t)\, dt$$

Dans le cas où $\{\Phi_k(t)\}$ est une base quelconque, il est très utile de l'orthogonaliser (ou l'orthonormaliser) à l'aide de la procédure de Gram-Schmidt, car une telle opération nous permet d'avoir une représentation plus simple à manipuler lors de la détection au niveau du récepteur. De plus, la séparation des signaux représentant les symboles se fait d'une manière plus aisée au moyen d'un corrélateur ou d'un filtre adapté .

1.10 Modèle vectoriel d'un système de communication numérique

Soit le modèle suivant

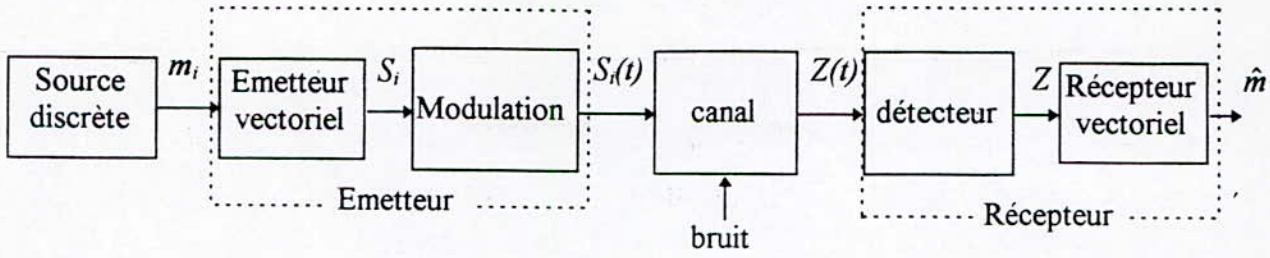


Figure 1.9 Modèle vectorielle d'un système de communication numérique.

La source discrète génère un des symboles m_i, $i=1,...,M$, chaque T seconde. On suppose que les M symboles de l'alphabet sont équiprobables, c'est-à-dire que

$$P_i = P(m_i \text{ é mis}) = \frac{1}{M}, \quad \forall i$$

Les symboles m_i est ensuite représenté par un vecteur

$$S_i = \begin{pmatrix} \alpha_{i1} \\ \alpha_{i2} \\ \vdots \\ \alpha_{in} \end{pmatrix}, \quad i = 1,...,M \qquad (1.16)$$

à l'aide du bloc émetteur vectoriel, où la dimension $n \leq M$ pour que les signaux $\Phi_k(t)$ soient linéairement indépendants. Ensuite à l'aide d'une base $\{\Phi_k(t)\}$ convenablement choisie, le bloc modulation affecte une représentation vectorielle au signal $S(t)$ dans un espace engendré par $\{\Phi_k(t)\}$.

Le signal $S_i(t)$ s'écrit donc sous la forme

$$S_i(t) = \sum_{k=1}^{n} \alpha_{ik} \Phi_k(t) \qquad (1.17)$$

et

$$\alpha_{ik} = \int_0^T S_i(t)\Phi_k(t)dt, \quad \begin{cases} i = 1,...,M \\ k = 1,...,n \end{cases} \qquad (1.18)$$

Donc chaque signal de l'ensemble $\{S_i(t)\}_{i=1}^{M}$ est complètement déterminé par les vecteurs

$$S_i = \begin{pmatrix} \alpha_{i1} \\ \vdots \\ \alpha_{ik} \end{pmatrix}, \quad i = 1,...,M$$

Dans un espace engendré par $\{\Phi_k(t)\}$, le vecteur S_i est un point message de coordonnés α_{i1}, α_{i2}, ... α_{in}. Un tel espace est appelé *espace des signaux*.

L'énergie du signal $S_i(t)$ est donnée par [11]

$$E_i = \int_0^T S_i^2(t)\,dt = \sum_{k=1}^{n} \alpha_{ik}^2 \qquad (1.19)$$

donc la quantité E_i est invariante lorsqu'on passe de $S_i(t)$ à sa représentation vectorielle.

On peut montrer aussi que [11]

$$\|S_i - S_k\|^2 = \sum_{j=1}^{n} \left(\alpha_{ij} - \alpha_{kj}\right)^2 \qquad (1.20)$$

qu'on appelle la *distance euclidienne* (ED) entre les deux signaux $S_i(t)$ et $S_k(t)$. Cette notion de distance est très importante pour la comparaison des signaux lors de la détection.

A la réception, on reçoit le signal $Z(t) = S_i(t) + W(t)$ (1.21.a)

où $W(t)$ est un bruit blanc gaussien ayant une moyenne nulle et une densité spectrale de puissance bilatérale $N_0/2$.

Le récepteur doit observer le signal $Z(t)$ et doit effectuer la meilleure estimation du signal transmis $S_i(t)$ ou d'une manière équivalente le symbole m_i .

Dans un espace euclidien de signaux, les signaux $S_i(t)$, $i=1,...,M$ peuvent être représentés par M points qu'on appelle « constellation du signal » ou les « points message ».

Lorsque la transmission est idéale, c'est-à-dire, $W(t) = 0$, il suffit d'appliquer $S_i(t)$ à un corrélateur [11] pour avoir le vecteur S_i qui correspond au symbole m_i. Toutefois, ceci n'est pas vrai en pratique car le signal reçu est toujours contaminé par un bruit $W(t)$ ce qui complique la décision. Dans ce cas, la sortie du corrélateur est donnée par le vecteur $Z = S_i + W$, $i = 1,...,M$. Le vecteur Z diffère de S_i par une quantité w appelée vecteur bruit.

On doit trouver à partir de Z une estimation $\hat{m}$ du symbole m_i de sorte que la probabilité moyenne de l'erreur de décision soit minimisée.

Le détecteur au sens du Maximum de Vraisemblance MV (ou ML : Maximum Likelihood) donne la solution à ce problème. Le ML postule qu'il faut chercher le point message le plus proche du point observé et ce au sens d'Euclid [11]. Dans ce cas une forme d'onde réceptionnée représentée par une point Z dans l'espace des signaux, la détection du symbole correspondant au sens ML peut être faite en calculant toutes les distances euclidiennes (voir (1.20)) $\|Z - S_i\|$,$i=1,...,M$, et choisi le symbole S_j pour lequel $\|Z - S_j\| < \|Z - S_i\|$ $i = 1,...,M$, $i \neq j$.

D'une manière équivalente et plus simple, cette détection peut se faire aussi en utilisant la règle suivante [11]

Choisir le symbole S_j si $\sum_{i=1}^{n} z_i \alpha_{ki} - \frac{1}{2} E_k$ est maximum pour $k = j$ (1.21.b)

où E_k est l'énergie du signal transmis $S_k(t)$ représentant le symbole m_k ($E_k = \sum_{i=1}^{n} S_{ki}^2$).

Notons ici que la règle (1.21.b) choisit le symbole S_j pour lequel la densité de vraisemblance est maximum ou d'une manière équivalente la distance euclidienne est minimum. Bien sûr ceci n'est vrai que pour un canal gaussien.

1.11 Techniques de modulation

La fonction de modulation a pour objectif d'adapter le spectre de signal à émettre au canal de transmission. Lorsqu'il s'agit d'une transmission numérique à travers un canal du type passe bande, il est nécessaire de transposer (ou de translater) le spectre de fréquence du signal à bande de base. Ceci peut se faire au moyen d'une porteuse qui est en général une fonction sinusoïdale pure. Cette opération s'appelle *modulation numérique* ou bien *modulation analogique discrète* car la porteuse est analogique et l'information (signal de base) est discrète.

En tout cas, on peut distinguer dans la modulation analogique discrète

- ASK : modulation par déplacement d'amplitude (Amplitude Shift Keying) ;
- PSK : modulation par déplacement de phase (Phase Shift Keying) ;
- FSK : modulation par déplacement de fréquence (Frequency Shift Keying)
- Ou une modulation composite ou hybride comme la modulation QAM qui est une combinaison des modulations ASK et PSK.

Ces modulations peuvent être considérées comme des cas spéciaux des modulation analogiques AM, PM, FM, et AM-PM respectivement.

Sur un canal gaussien, le choix d'une modulation se fait en considérant l'occupation spectrale, les performances et la complexité du couple modulateur/démodulateur. Il est à noter que la faible occupation spectrale et les performances du système sont deux contraintes antagonistes, ce qui nécessite en pratique un compromis lors du choix d'une modulation.

Nous allons maintenant présenter un rappel sur les types de modulations utilisées dans notre projet.

1.11.1 Modulation à déplacement d'amplitude ASK [15]

Les signaux ASK M -aire sont donnés par

$$S_i(t) = \sqrt{\frac{2E_0}{T}} a_i \cos(\omega_c t), \qquad i=1,\ldots,M \tag{1.22}$$

où les symboles a_i prennent leurs valeurs dans l'alphabet $\{\pm 1,\pm 3,\ldots,\pm(M-1)\}$, E_0 représente l'énergie du signal ayant la plus faible amplitude et T est la durée de chaque symbole m_i. La pulsation $\omega_c = \frac{2\pi n_v}{T}, (n_c \in N)$ est la fréquence porteuse.

On prend comme fonction de base

$$\Phi_1(t) = \sqrt{\frac{2}{T}} cos(\omega_c t), \qquad 0 \le t \le T \tag{1.23}$$

donc

$$S_i(t) = \sqrt{E_0} a_i \Phi_1(t)\ ,\ i = 1,\ldots,M \tag{1.23}$$

ce qui signifie que la modulation ASK est une modulation linéaire unidimensionnelle.

La figure 1.10 montre une représentation géométrique des signaux ASK M-aire avec les régions de décision.

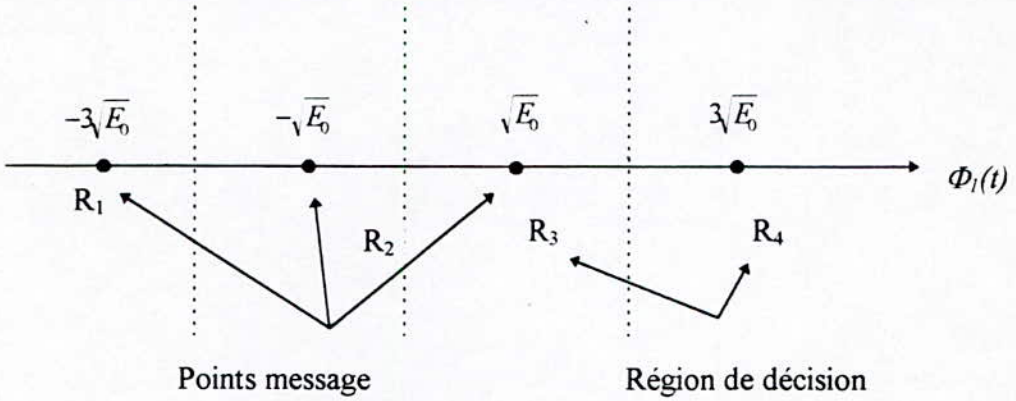
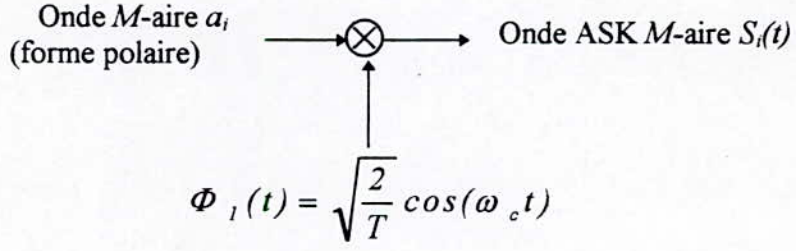
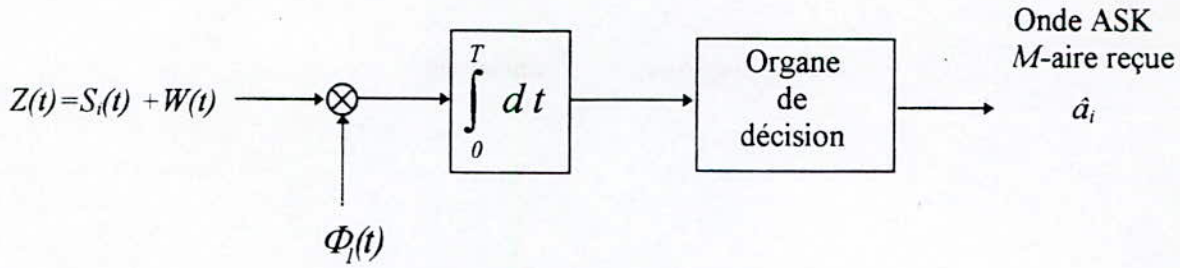


Figure 1.10 Les signaux ASK *M*-aire avec les régions de décision.

Les schémas synoptiques d'un modulateur et d'un démodulateur ASK sont illustrées à la figure1.11



a) Modulateur ASK M-aire



b) Démodulateur ASK *M*-aire

Figure 1.11 Modulateur et démodulateur ASK *M*-aire.

Le signal $Z(t) = S_i(t) - W(t)$ est multipliée par $\Phi_1(t)$(figure 1.11.b) et moyenné sur une durée T, ce qui donne

$$Z_1 = \sqrt{E_0}a_i + W, \quad i = 1,\ldots,M \tag{1.25}$$

où W est une variable aléatoire gaussienne de moyenne nulle et de variance $\sigma^2 = N_0/2$ représentant $w(t)$ dans l'espace engendré par $\Phi_1(t)$. La probabilité d'erreur par un symbole est donnée par la formule [15]

$$P_s(e) = \frac{M-1}{M} erfc\left(\sqrt{\frac{3}{(M^2-1)}\frac{E}{N_0}}\right) \tag{1.26}$$

où E/N_0 est le rapport de l'énergie par symbole sur la densité spectrale de puissance du bruit et

$$erfc(x) = \frac{2}{\sqrt{\pi}}\int_x^{\infty} exp(-Z^2)dZ. \tag{1.27}$$

Du point de vue pratique, c'est la probabilité d'erreur par bit qui est la plus importante à déterminer. Toutefois, celle-ci, que nous notons $P_b(e)$, ne peut être calculée d'une manière générale, mais elle peut être doublement bornée par [15]

$$\frac{1}{\log_2 M} P_s(e) \le P_b(e) \le P_s(e)$$

Remarque

On note que la fréquence baud (fréquence symboles) $R_s = 1/T$ est liée au débit binaire $R_b = 1/T_b$ (T_b: la durée d'un bit) par la relation

$$R_s = R_b / \log_2 M \tag{1.28}$$

1.11.2 Modulation à déplacement de phase PSK [11]

Les signaux PSK *M*-aire sont donnés par

$$S_i(t) = \sqrt{\frac{2E}{T}} cos(\omega_c t + \theta_i), \quad \theta_i = \frac{2i\pi}{M}, i = 0,1,...,M-1$$

où E représente l'énergie de chaque symbole et T sa durée. La pulsation $\omega_c = \frac{2\pi n_c}{T}$ ($n_c \in N$) représente la fréquence porteuse .

On peut mettre les signaux $S_i(t)$ sous la forme

$$S_i(t) = \sqrt{\frac{2E}{T}} cos(\omega_c t) cos\theta_i - \sqrt{\frac{2E}{T}} sin(\omega_c t) sin\theta_i \qquad (1.29)$$

On prend comme fonctions de base $\begin{cases} \Phi_1(t) = \sqrt{\frac{2}{T}} cos(\omega_c t), & 0 \le t \le T \\ \Phi_2(t) = \sqrt{\frac{2}{T}} sin(\omega_c t), & 0 \le t \le T \end{cases}$ (1.30)

Une représentation géométrique des signaux PSK M-aire est donnée à la figure 1.13. Les points de la constellation sont régulièrement espacées, sur un cercle de rayon $\sqrt{E}$, par un angle égale à $2\pi/M$.

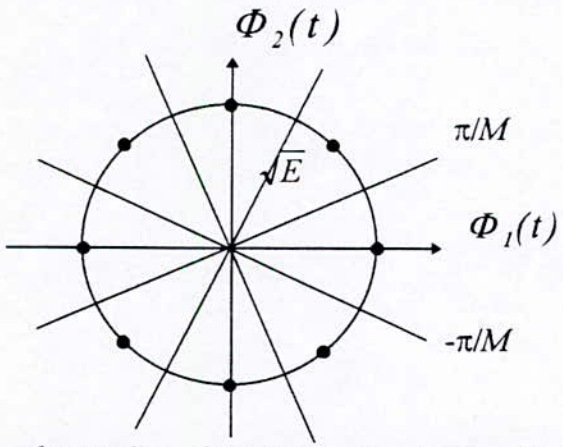


Figure 1.13 Représentation géométrique des signaux PSK *M*-aire ($M = 8$).

La démodulation du signal PSK M-aire se fait à l'aide du schéma suivant

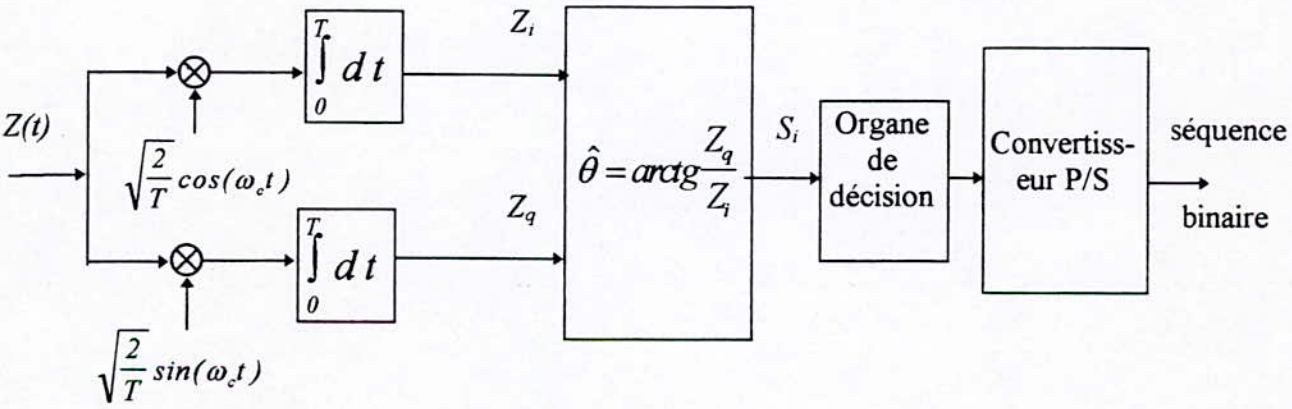


Figure 1.14 Démodulateur PSK *M*-aire.

Le signal $Z(t) = S_i(t)+W(t)$ est acheminé à travers deux canaux où il est multiplié par $cos(\omega_c t)$ et $sin(\omega_c t)$et moyenné ensuite sur une durée de T

$$\begin{aligned} Z_I(t) &= \sqrt{E}\cos\left(\frac{2\pi i}{M}\right)+W_I, \quad i=0,1,\ldots,M-1 \\ Z_Q(t) &= \sqrt{E}\sin\left(\frac{2\pi i}{M}\right)+W_Q, \quad i=0,1,\ldots,M-1 \end{aligned} \tag{1.31}$$

W_I et W_Q sont des variables aléatoires gaussiennes indépendantes de moyennes nulles et de variances $\sigma^2 = N_0/2$ chacun.

Il est simple de démontrer que la probabilité d'erreur par symbole pour un rapport $E/N_0 >> 1$ est donnée par la formule

$$P_s(e) \cong erfc\left(\sqrt{\frac{E_0}{N_0}}\sin(\frac{\pi}{M})\right), M \geq 4 \tag{1.32}$$

Pour $M = 2$, on a

$$P_s(e) = \frac{1}{2}erfc\left(\sqrt{\frac{E_0}{N_0}}\right) \tag{1.33}$$

1.11.3 Modulation d'amplitude à deux porteuses en quadrature QAM [11]

Pour un nombre de points M grand, ni l'ASK ni la PSK ne constituent une solution satisfaisante pour utiliser efficacement l'énergie émise.

La probabilité d'erreur étant fonction de la distance minimale entre points de la constellation, la meilleure modulation (pour le canal gaussien) est celle qui maximise cette distance pour une puissance moyenne donnée.

Un choix plus rationnel que l'ASK où les points de la constellation sont sur une droite et la PSK où les points sont sur un cercle, est de moduler en amplitude deux porteuses en quadratique.

Soient les signaux

$$S_i(t) = \sqrt{\frac{2E_0}{T}}a_i\cos(\omega_c t) - \sqrt{\frac{2E_0}{T}}b_i\sin(\omega_c t), \quad 0 \leq t \leq T \tag{1.34}$$

où E_0 est l'énergie du signal ayant la plus faible amplitude,a_i et b_i sont des entiers indépendants

$$\begin{aligned} \Phi_1(t) &= \sqrt{\frac{2}{T}}\cos\omega_c t \\ \Phi_2(t) &= \sqrt{\frac{2}{T}}\sin\omega_c t \end{aligned}, \quad 0 \leq t \leq T \tag{1.35}$$

Les coordonnées du points message m_i est $(a_i\sqrt{E_0}, b_i\sqrt{E_0})$ où le couple (a_i, b_i) est donné par la matrice suivante

$$(a_i, b_i) = \begin{pmatrix} (-L+1, L-1) & (-L+3, L-1) & \cdots & (L-1, L-1) \\ \vdots & \vdots & \ddots & \vdots \\ (-L+1, -L+1) & (-L+3, -L+1) & \cdots & (-L-1, -L+1) \end{pmatrix} \tag{1.36}$$

où $L = \sqrt{M}$.

Par exemple pour $L = 4$, $M = 16$ on a

$$(a_i, b_i) = \begin{pmatrix} (-3,+3) & (-1,+3) & (+1,+3) & (+3,+3) \\ (-3,+1) & (-1,+1) & (+1,+1) & (+3,+1) \\ (-3,-1) & (-1,-1) & (+1,-1) & (+3,-1) \\ (-3,-3) & (-1,-3) & (+1,-3) & (+3,-3) \end{pmatrix}$$

Une représentation géométrique des signaux QAM M-aire pour le cas M=16 est donnée à la figure 1.15.

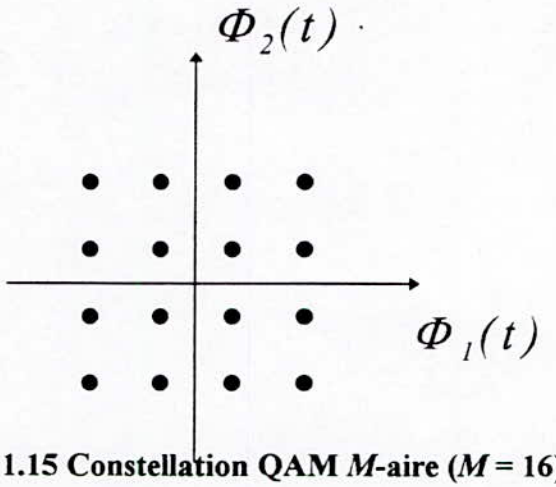
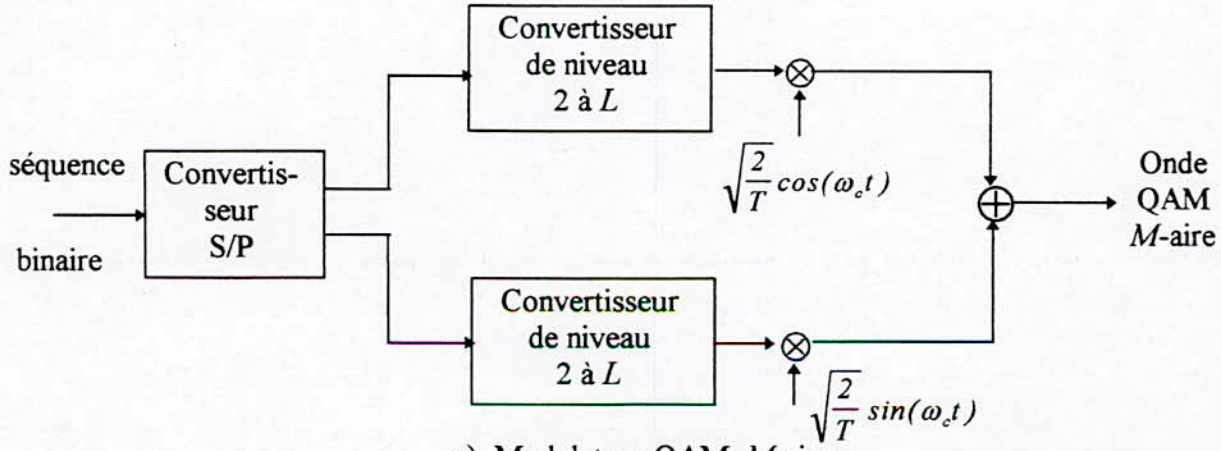
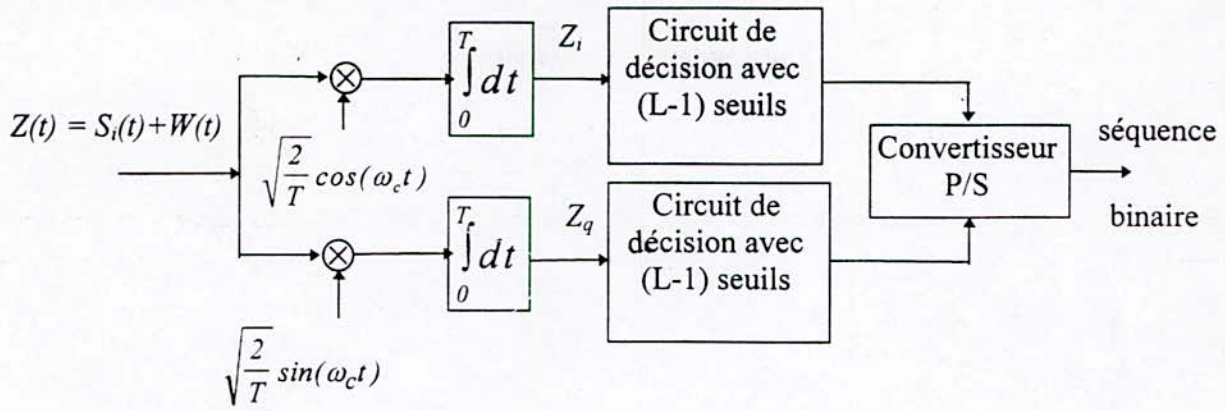


Figure 1.15 Constellation QAM *M*-aire (*M* = 16).

Les schémas synoptiques d'un modulateur et d'un démodulateur QAM sont illustrés à la figure (1.16)



a) Modulateur QAM *M*-aire.



b) Démodulateur QAM *M*-aire

Figure 1.15 Modulateur et démodulateur QAM *M*-aire.

On peut montrer facilement que la probabilité d'erreur par symbole est donné par la formule[11]

$$P_s(e) = 2\left(1-\frac{1}{\sqrt{M}}\right)erfc\left(\sqrt{\frac{3E_{av}}{2(M-1)N_0}}\right) \qquad (1.37)$$

où

$$E_{av} = \frac{2(M-1)E_0}{3} \qquad (1.38)$$

est l'énergie moyenne sur L points.

1.12 Conclusion

On a présenté dans ce chapitre quelques généralités sur des systèmes de communication numériques, en particulier les fonctions de codage convolutif et de la modulation.

Toutefois dans cette présentation, on a essayé de donner l'essentielle de ce thème pour ne pas être exhaustif. Dans la section 1.8.2, on a donné un exemple sur le calcul de la fonction génératrice d'un code de canal convolutif, qui sera très utilisée dans le prochain chapitre en particulier pour les calculs des probabilités d'erreur.

Dans la section 1.11, nous avons présenté les formules de probabilité d'erreur des modulations ASK, PSK, et QAM pour des formes d'ondes non codées. Ceci est très utile pour les comparaisons que nous devons faire entre les performances des systèmes utilisant des séquences codées et non codées.

Il est à noter que, les codeurs convolutifs conventionnels nécessitent un élargissement de la bande passante pour une performance meilleure car les deux opérations du codage et de modulation sont effectuées séparément. Un remède à ce problème (en particulier lorsqu'il y a des contraintes sur la bande de fréquence) est développé par G.Ungerboeck [17] et appelé codeur par TCM (Trellis Coded Modulation) qui combine ces deux dernières opérations et qui ne nécessite aucun sacrifice de la bande passante, ceci fait l'objet du chapitre suivant.

CAPITRE II :
LES MODULATIONS CODEES EN TREILLIS

CHAPITRE II

Modulations Codées en Treillis (TCM)

2.1. Introduction

La théorie du codage classique a été bâtie en considérant le canal symétrique binaire (BSC) comprenant un élément de décision. En sortie de ce canal, le problème est de détecter le signal et de corriger les erreurs dues aux perturbations lors de la transmission à travers le canal. Aussi bien pour les codes en blocs que pour les codes convolutifs, l'optimisation est faite en maximisant la distance de Hamming minimale entre les séquences codées.

L'inclusion d'un élément de décision pour aboutir à un canal BSC à partir d'un canal gaussien qui fournit un signal à valeurs continues peut aujourd'hui être considéré comme une erreur qui a empêché pendant longtemps la construction de bons codes pour les signaux transmis à l'aide de modulations a grande efficacité spectrale. La probabilité d'erreur pour ces modulations est fonction de la distance euclidienne minimale et la maximisation de la distance de Hamming n'implique pas la maximisation de la distance euclidienne.

Jusqu'au milieu des années 70, les fonctions de codage et de modulation étaient traitées comme deux entités séparées et indépendantes. L'optimisation du codage au sens de la distance de Hamming ne donnait pas alors de bons résultats lorsque la modulation n'était pas une modulation à deux ou à quatre états de phase.

C'est G.Ungerboeck [17] qui est alors arrivé à une solution pratique permettant de tirer pleinement profit du codage avec des modulations à grand nombre d'états sans sacrifier la bande de fréquence . L'idée est de faire le codage directement dans l'espace des signaux et l'optimiser au sens de la distance euclidienne . Cette méthode est connue sous l'appellation « Modulation codée en treillis » (Ang. Trellis Coded Modulation TCM) . Le terme treillis est utilisé par ce qu'un tel système de codage peut être décrit par un diagramme de transition d'état (Treillis) similaire au diagramme en treillis des codes binaires convolutifs . La différence est que dans les systèmes à TCM les branches de treillis sont attribuées aux signaux modulés non binaire.

En 1984, un système à TCM avec un gain de codage de 4 dB est adopté par CCITT(Comité Consultatif International Télégraphique et Téléphonique) pour être utilisé dans les nouveaux modems à haute vitesse dans la bande de la parole . Avant l'apparition de la TCM, une transmission non codée à 9.6Kbit/s à travers des canaux de bande de parole été souvent considéré en pratique comme une limite pratique pour les modems des données .

Depuis 1984, les modems de données sont apparus dans les marchés où en utilisant les TCM mais avec une amélioration de l'égalisation, synchronisation, annulation d'écho,...et ainsi de suite, pour transmettre des données à travers les canaux de bande de la parole avec un débit de 14.4 Kbit/s et plus. L'utilisation commune des techniques de TCM dans de telles applications, en satellite, en micro onde terrestre, et les communications mobile, pour augmenter le débit ou pour réaliser des opérations satisfaites en cas de faible rapport signal sur bruit, peuvent être prédite dans un futur proche .

2.2. Principe de codage en TCM [12]

La forme simple de codage en treillis proposée par G.Ungerboeck [17], utilise une constellation à deux dimensions, et est illustrée sur la figure 2.1. Nous limiterons notre étude aux codeurs en treillis linéaires dont le formalisme mathématique est détaillé dans [3]. Dans la figure 2.1.a, le symbole d'information PAM (Pulse Amplitude Modulation) S_k non codé est généré par un trancodeur (qui transforme des séquences binaires à des symboles sous forme d'ondes).

Remarque

On entend par « trancodeur » le bloc qui transforme des séquences binaires à des symboles sous forme d'ondes. On remarque que ce bloc est exprimé dans la littérature anglaise par « line coder » [12] (codeur de ligne) , et « mapper » [17] (application). Nous avons choisi le mot « transcodeur » pour exprimer cette equivalence.

La taille de la constellation de ce transcodeur est de 2^k , et le débit (Ang: rate) d'information binaire est k bits par symbole (système sans codage). Le codeur en treillis de la figure 2.1.b, est obtenu en modifiant la figure2.1.a par l'insertion d'un codeur de canal convolutif ayant un taux k/n (n>k), le trancodeur est modifié pour utiliser une constellation de taille 2^n .

Un gain de codage significatif peut être atteindre par cette méthode, car l'utilisation de codage peut réduire le RSB pour la même probabilité d'erreur ou de réduire la probabilité d'erreur pour le même RSB.

Exemple 2.1 [11]

La figure 2.2 illustre le schéma d'un codeur de Ungerboeck en treillis simple à modulation 8-PSK, pour une transmission de 2 bits / symbole. Ce codeur en treillis utilise un codeur convolutif de taux 1/2 avec une entrée non codée. On peut dire que ces deux dernières forment un codeur convolutif de taux 2/3 (c'est-à-dire 2 entées et 3 sorties). La constellation de ce codeur en treillis est de taille égale à $2^3 = 8$ points messages. Le treillis correspondant est illustré à la figure 2.2.b. Ce treillis contient 4 états, car le codeur convolutif contient deux éléments mémoire (Ang : Flip-flops).

Dans le bloc de transcodage, le codeur TCM réalise la correspondance entre les sorties binaires du codeur convolutif et les numéros de points messages.

Il est à noter que l'état du codeur en treillis est défini par le contenu des éléments mémoires tels que la lecture se fait de la gauche vers la droite.

Si on applique à l'entrée du codeur la séquence binaire 10 10 11 01, c'est-à-dire 1 1 1 0 à l'entrée x_2 et 0 0 1 1 à l'entrée x_1 , on reçoit, après les opérations d'utilisation du treillis de la figure 2.2.b et de la correspondance du transcodeur de la figure 2.2.a, à la sortie la séquence des symboles exprimée par les chiffres 2 3 5 7.

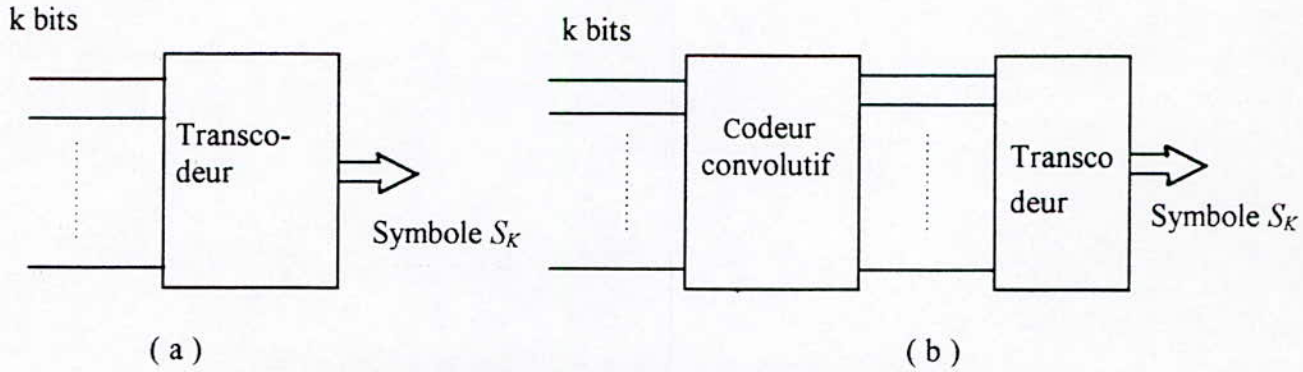


Figure 2.1. a. Système de transmission sans codage .
b. Système de transmission avec codage .

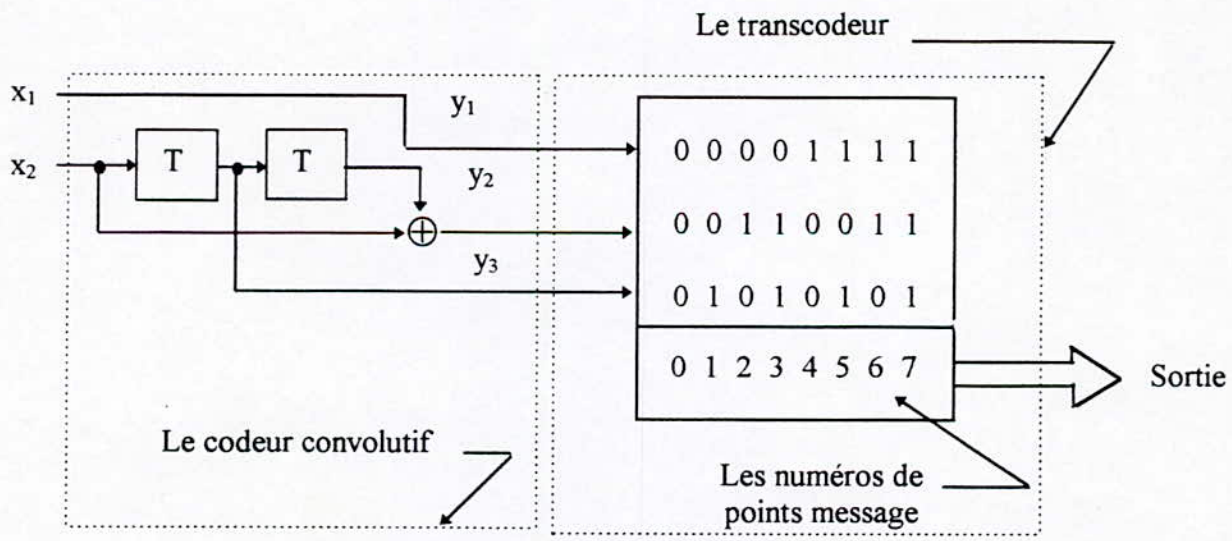


Figure 2.2.a Codeur en treillis de taux 2/3 et une modulation 8-PSK.

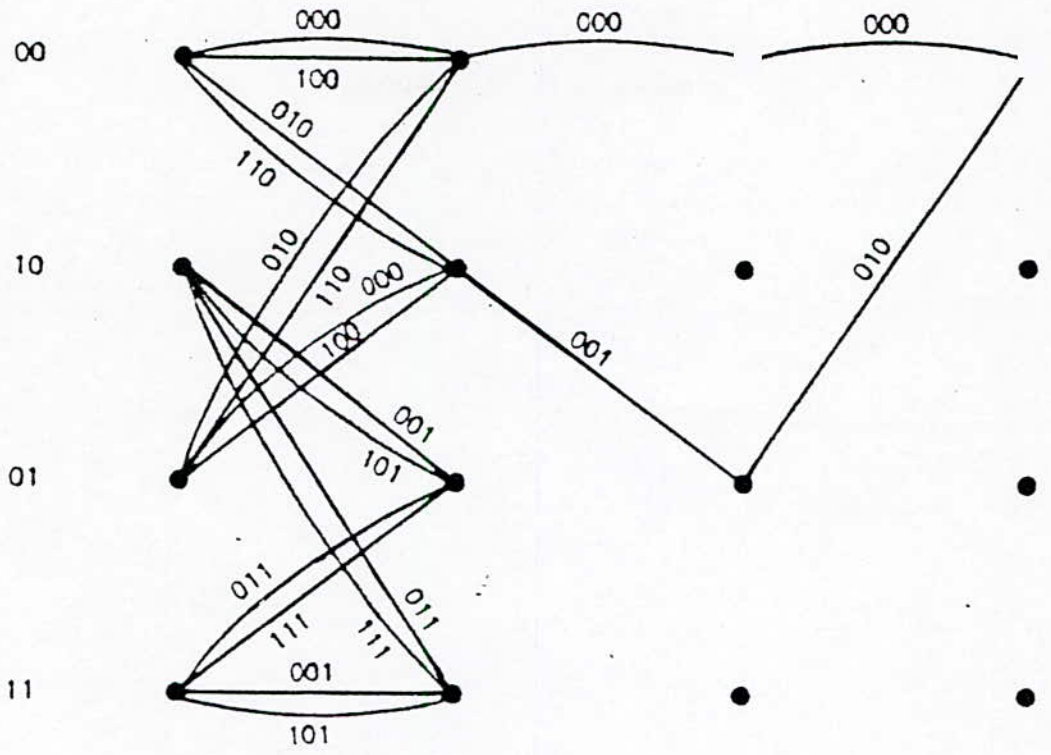


Figure 2.2.b Le treillis correspondant au codeur de la figure 2.2.a

Lorsque on traite le codage convolutif et la modulation comme des entités séparés, l'optimisation du codage au sens de Hamming ne donnait pas des bons résultats, lorsque la modulation n'était pas de modulation à 2 ou à 4 états de phase. Mais, lorsqu'on utilise le codage directement dans l'espace des signaux et l'optimiser au sens de la distance euclidienne, on peut arriver à des meilleurs performances, d'où le principe de la TCM.

2.2.1 - Modélisation et capacité de canal

Pour étudier la performance d'un système de communication numérique qui utilise la TCM, On considère le modèle du canal le plus simple qui est le canal gaussien, à modulation multiniveaux / phases, qui utilise un codeur convolutif linéaire, et qui est perturbé par un bruit blanc gaussien additif (AWGN) de moyenne nulle et de densité spectral monolatérale $N_0/2$.

Ayant un récepteur qui utilise le décodage fine (Ang : Soft decision) à maximum de vraisemblance (MV), et un canal gaussien à modulation multiniveaux / phases comme indiquer dans le paragraphe précédant.

On considère seulement les modulations à une seule dimension et à deux dimensions. On suppose qu'il y a interférence entre les symboles dans la bande limitée du canal considéré [17]. Avec la synchronisation de la phase de la porteuse et un réglage parfait, après échantillonnage la sortie du canal prend la forme (figure 1.9)

$$Z_n = S_n + W_n \qquad (2.1)$$

où S_n est une valeur réelle ou complexe du signal du canal discret, transmise à l'instant de modulation nT, et Wn est un bruit indépendant normalement distribué avec une moyenne nulle et une variance σ^2 sur chaque dimension, Z_n est le résultat d'un échantillonnage du signal reçu.

Le RSB moyen est défini comme

$$RSB = \frac{E\{|S_n^2|\}}{E\{|W_n^2|\}} = \begin{cases} \dfrac{E\{|S_n^2|\}}{\sigma^2} \text{.....} (a) \text{ Pour une modulation à une dimension} \\ \dfrac{E\{|S_n^2|\}}{2\sigma^2} \text{.....} (b) \text{ Pour une modulation à deux dimension.} \end{cases} \qquad (2.2)$$

Sans perdre de généralité, on suppose que la puissance moyenne du signal $E\{|S_n^2|\} = 1$. Il est à noter que l'amélioration de la performance d'un système par l'utilisation de codage est généralement limitée par la capacité de canal de Shannon (capacité maximale).

L'extension de la forme de la capacité d'une canal discret sans mémoire dont les valeurs de sortie sont continues donne [17]

$$C = \max_{Q} \sum_{k=0}^{M-1} Q(k) \int_{-\infty}^{+\infty} P(Z/S_k) \log_2 \left\{ \frac{P(Z/S_k)}{\sum_{i=0}^{M-1} Q(i) P(Z/S_i)} \right\} dz \qquad (2.3)$$

est exprimé en bit / T. M est le nombre des signaux d'entrée du canal discret , $Q(k)$ représente la probabilité à priori associée à S_k .

A cause de la nature du canal AWGN, on peut remplacer dans (2.3)

$$P(Z/S_k) = \exp\left[-|Z - S_k|^2 / 2\sigma^2\right] \cdot \begin{cases} (2\pi\sigma^2)^{-1/2} \text{...} (a) \\ (2\pi\sigma^2)^{-1} \text{.....} (b) \end{cases} \qquad (2.4)$$

Si on suppose en plus que seulement les codes avec des entrées du canal à des réalisations équiprobables, sont intéressant, la maximisation à travers $Q(k)$ dans (2.3) peut être éliminée. L'équation (2.3) peut s'écrire sous la forme [17]

$$C^*_{Q(k)=\frac{1}{M}} = \log_2(M) - \frac{1}{M} \sum_{k=0}^{M-1} E\left\{ \log_2 \sum_{i=1}^{M-1} \exp\left[-\frac{|S_k - W - S_i|^2 - |W|^2}{2\sigma^2} \right] \right\} \qquad (2.5)$$

Dans l'équation (2.5) l'intégration est remplacé par l'espérance de la variable normalement distribuée W, qui est réel avec une variance σ^2 pour (a) et complexe avec une variance 2 σ^2 pour (b) .

Dans les figures 2.3.a et 2.3.b, la capacité C^* est tracée en fonction du RSB pour les types de modulations les plus connues comme AM, PSK, et QAM.

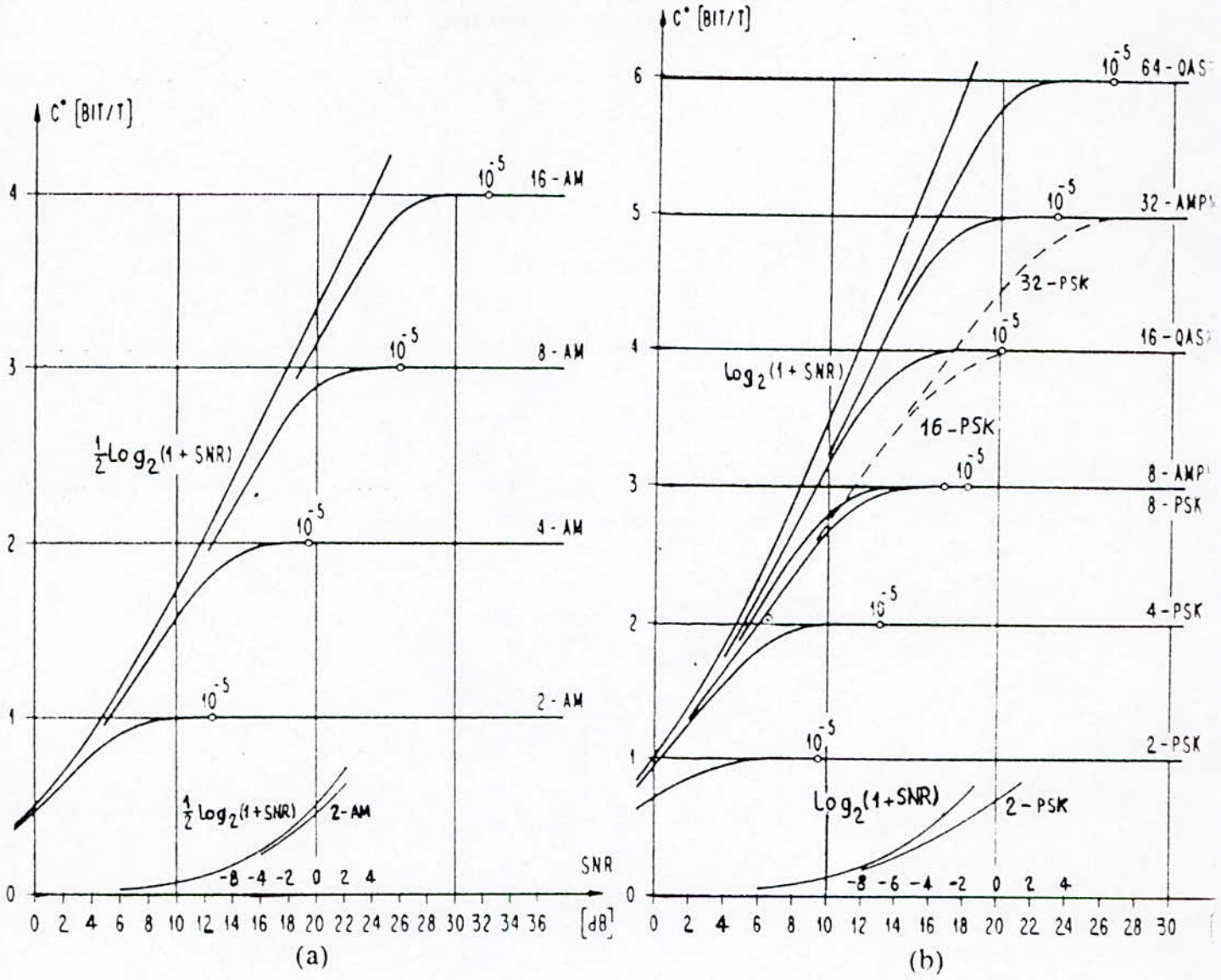


Figure 2.3 La capacité du canal à bande limités du type AWGN

(a) Modulation unidimensionnelle, (b) Modulation bidimensionnelle

Pour interpréter les figures 2.3.a et 2.3.b, nous considérons comme exemple la transmission de 2 bit / T pour une modulation 4 - PSK sans codage où la probabilité d'erreur par événement $P_e(e) = 10^{-5}$ est atteindre pour un RSB = 12.9 dB. Si on choisit une modulation 8-PSK, alors la même probabilité d'erreur $P_e(e) = 10^{-5}$ est théoriquement possible pour un RSB = 5.9 dB avec la même débit d'information 2 bit / T en supposant que la séquence est presque infinie. Au delà de cette valeur (c'est à dire au delà de 8 points message) on gagne seulement 1.2 dB.

Des propriétés similaires sont obtenues pour les autres types de modulations. On peut conclure que par le doublement du nombre des signaux du canal, presque tout est gagné en terme de capacité du canal, donc dans les codeurs en treillis, on utilise des codeurs convolutifs de taux $re = \frac{m}{m+1}$.

Pour la transmission à 2 bits / T, avec une modulation 8-PSK, un codeur convolutif de taux 2/3 à une distance de Harmming DH maximale, et avec le codage de Gray comme fonction de transcodage, il existe quelques problèmes [17]

i) Le codage de Gray ne transforme pas de façon monotone une DH élevée à une DE (distance euclidienne) élevée.

ii) Quelque fois, les permutations des sorties linéaire du codeur convolutif ont une influence inconnue sur la structure des codes résultants de 8-PSK.

Nous allons poursuivre une méthode différente qui vise à maximiser directement la DE limite (pour plus de détail sur la DE limite voir 2.3.2) . Cette méthode est basée sur un ensemble des règles dites *transcodage par la partition d'ensembles.*

2.3 Codage par la partition d'ensemble[15]

2.3.1 Principe

Les modulations codées sont basés sur le principe de partition de la constellation de l'alphabet qui a été introduit pas Ungerboeck [17]. La partition de la constellation est faite de manière à augmenter la distance euclidienne minimale à l'intérieur des sous-ensembles obtenus à partir de cette partition. Nous allons illustrer ce principe en considérant la partition de la constellation 16-QAM (Fig 2.4). La constellation 16-QAM , que nous notons A0, est partitionnée dans une première étape en deux sous-ensemble B0 et B1 dont la distance minimale d_1 est égale à $\sqrt{2}d_0$, où d désigne la distance minimale dans A0. Ensuite B0 est partitionné en C0 et C1 et de la même façon, B1 est partitionné en C1 et C3, tels que la distance minimale dans les sous-ensemble Ci (i=0,1,2,3) soit $d_2 = \sqrt{2}d_1 = d_0$.

Enfin, chacun des quatre sous-ensemble Ci ainsi obtenus est partitionné en deux sous ensemble D ayant chacun deux points. On voit ainsi que chaque étape de la partition d'alphabet augmente la distance minimale dans les sous-ensembles de 3 dB. A noter que cette propriété est valable pour toutes les constellations QAM, mais l'augmentation de la distance à chaque étape de la partition n'est pas régulière pour les modulation PSK (Figure 2.6) qui utilise le même principe de partition que la modulation QAM.

En général, les modulations codées n'utilisent pas la chaîne de partition jusqu'à la dernière étape.

Parmi les m bits d'information à l'entrée du codeur, m' bits entrent dans un codeur convolutif de rendement m'/(m'+1) et les (m'+1) bits de sortie de celui-ci sélectionnent un sous-ensemble de la constellation (voir la fig 2.5)

Les (m-m') bits restant ne sont pas codés et sélectionnés par m+1 bits codés. Le principe du codeur est illustré à la figure 2.5

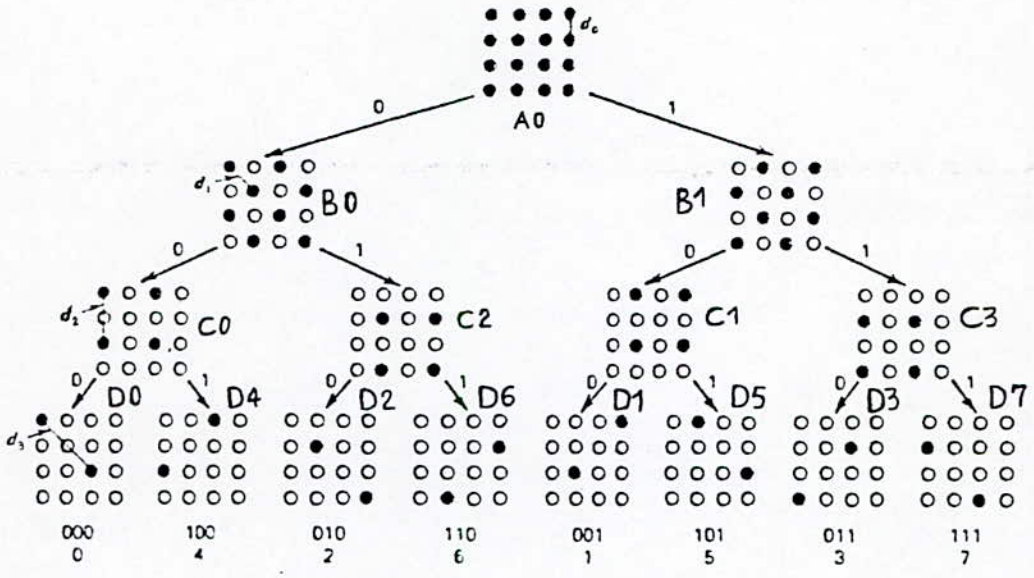


Figure 2.4 La partition de 16-QAM ($d_0<d_1<d_2<d_3$, $E\{|S_n|^2\}=1$)

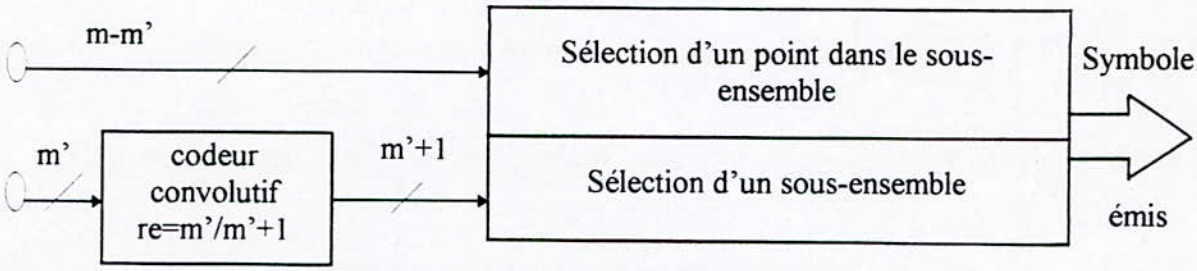


Figure 2.5 Schéma de principe de modulation codée en treillis.

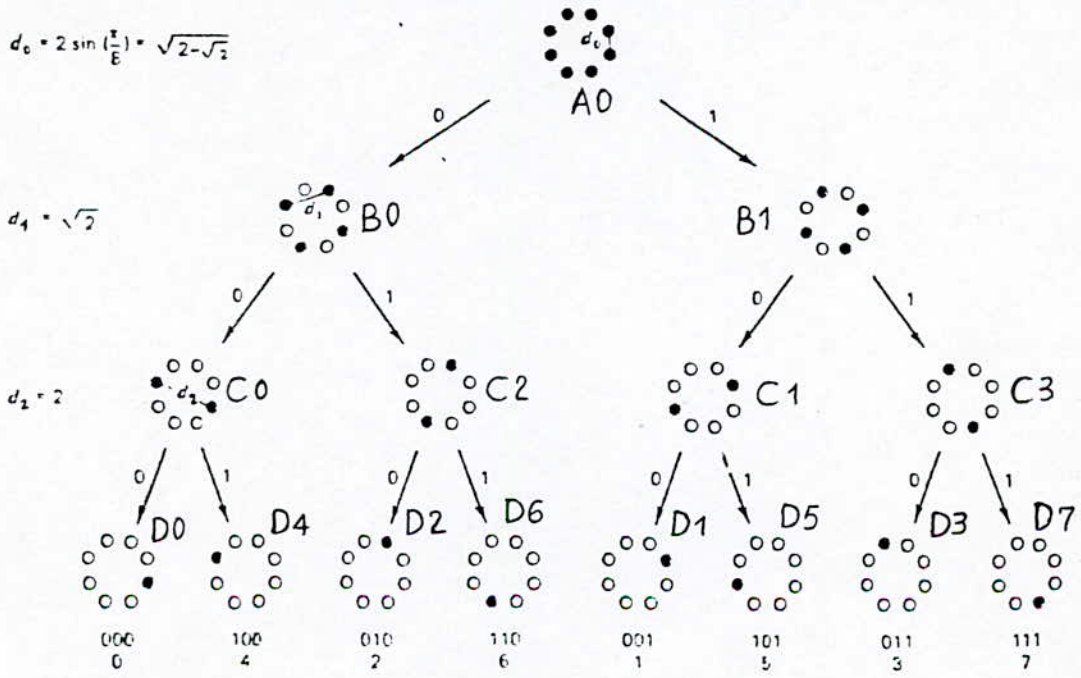


Figure 2.6 La partition de 8-PSK ($d_0<d_1<d_2<d_3$, $E\{|S_n|^2\}=1$)

2.3.2 Notion de la distance d'Euclid minimale [17], [3]

La distance euclidienne, entre deux séquences qui sont formées par des symboles généres par le codeur en treillis, est donnée par

$$d_E^2((S_t)_i,(S_t)_j) = \sum_E d_E^2(S_{it},S_{jt}) \qquad (2.6)$$

où S_{it} et S_{jt} sont des ondes appartient aux séquence $(S_t)_i$ et $(S_t)_j$ respectivement.

Un paramètre très importante d'un code en treillis qui est la distance limite (Ang: Free distance). Cette distance limite est donnée par

$$d_{free}^2 = \min_{i \neq j} \sum_t d_E^2(S_{it},S_{jt}) \qquad (2.7)$$

L'équation (2.7) est une équation générale, mais lorsque le treillis de la TCM contient des transitions parallèles, la distance minimale d_{free} peut être exprimée par

$$d_{free}^2 = min(d_p^2, d_t^2) \qquad (2.8)$$

où d_p^2 désigne la distance minimale entre les branches parallèles et d_t^2 désigne la distance du treillis lui même. L' importance de ce paramètre d_{free}^2 apparaît dans le calcul de la probabilité d'erreur.

Si un décodage à maximum de vraisemblance est appliquée, la probabilité d'événement d'erreur sera pratiquement approchée pour un RSB élevé, par une borne inférieure

$$P_e(e) \geq N(d_{free}) \cdot Q(d_{free} / 2\sigma) \qquad (2.9)$$

où $N(d_{free})$ dénote la multiplicité moyenne de l'erreur avec une distance d_{free} (c'est le nombre moyen des voisins à une distance d_{free}), et $Q(.)$ est la fonction de Gauss de la probabilité d'erreur définie par

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} exp(-y^2 / 2)dy \qquad (2.10)$$

A l'aide des formules qui sont données dans [3], on peut calculer numériquement les paramètres d_{free} et $N(d_{free})$.

Exemple 2.2 [12]

On considère un codeur convolutif de taux ½ suivi par un transcodeur qui utilise une modulation 4-PSK , illustrée à la figure 2.6. Le treillis correspond au codeur en treillis est montré dans la figure 2.7.a

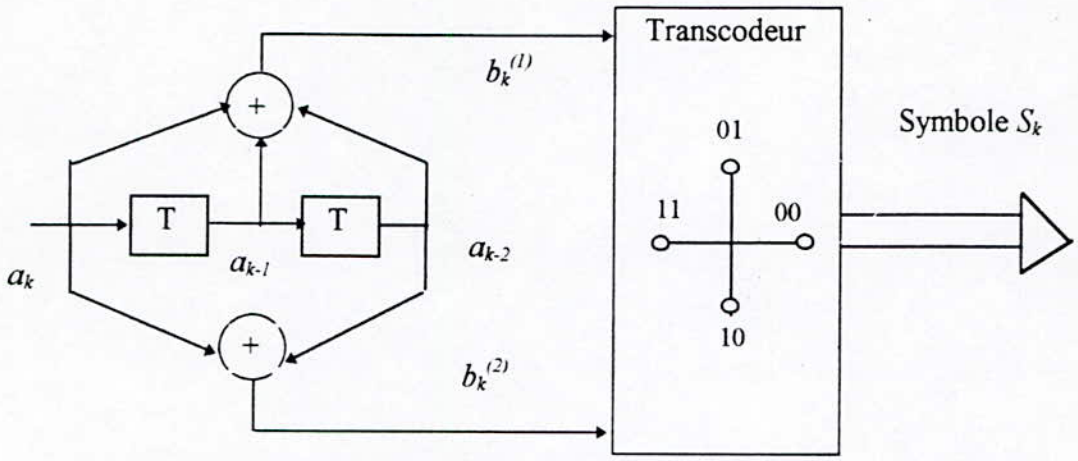
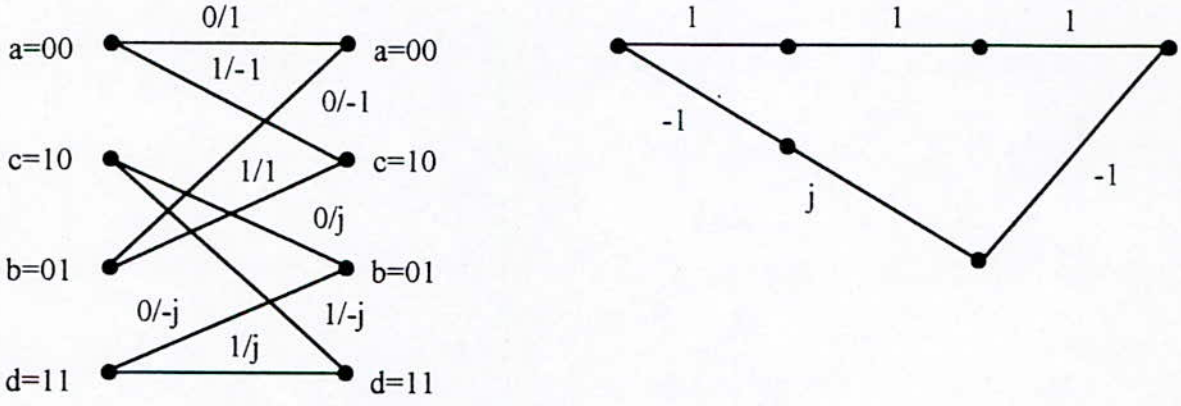


Figure 2.7 Codeur en treillis à modulation 4-PSK et de taux = 1/2.



(j : Le nombre complexe de module égale à 1 et d' argument égale à $\pi/2$)

Figure 2.8 a) Le treillis (portion de base) du codeur de la figure 2.7

b) Un événement ayant une distance minimale.

La seule différence entre le treillis de la figure 2.8.a et le treillis d'un codeur convolutif de taux 1/2 , est que les transitions sont étiquetées par a_k / S_k , où a_k représente l'entrée 1/2du codeur convolutif et S_k est le symbole de sortie. Par contre dans les codeurs convolutifs de taux , les transitions sont étiquetées par a_k / $b_k^{(1)}$ $b_k^{(2)}$ où a_k est l'entrée de codeur convolutif et $b_k^{(1)}$ $b_k^{(2)}$ représente la sortie codée du codeur. Dans la figure 2.8.b , nous avons représenté la trajectoire de l'événement d'erreur par rapport à celle de l'événement correcte (qui considérée l'événement de tous zéro); à partir de cette figure on peut déterminer la distance minimale d_{free} , en effet

$$d_{free}^2 = |1+1|^2 + |1-j|^2 + |1+1|^2 = 10$$

Dans ce cas on peut vérifier qu'il n'y a pas un événement d'erreur qui donne une distance inférieure à $\sqrt{10}$.

En effet, on observe que toutes les branches divergent à partir du même noeud ayant une distance égale à 2, et toutes les branches convergent vers le même noeud ayant une distance égale à 2, donc la distance minimale d_{free} est minorée par $\sqrt{8}$

$$d_{free} \geq \sqrt{2^2 + 2^2} = \sqrt{8}.$$

On peut déterminer au plus, qu'après deux chemins divergent, toutes les combinaisons possibles des branches ayant une distance$\sqrt{2}$, donc

$$d_{free}^2 \geq 2^2 + 2 + 2^2 = 10$$

d'où $d_{free}^2 = 10$

Exemple 2.3 [12]

Dans l'exemple précédent le codeur en treillis, ayant un seul bit d'information qui entre dans le codeur convolutif de débit 1/2 pour produire deux bits codés. Ces deux bits codés produisent un alphabet de taille égale 4. La technique est aisément étendue pour faire l'utilisation d'alphabet plus grand que 4. Une simple extension du codeur en treillis décrit dans l'exemple précédent, illustré à la figure 2.9.

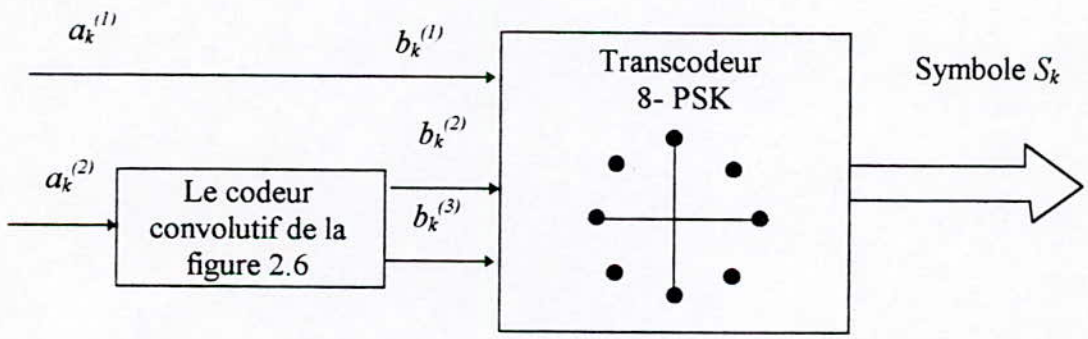


Figure 2.9 Codeur en treillis de 8-PSK avec un bit non codé.

On note que l'entrée $a_k^{(1)}$ produit ce qu'on appelle les transitions parallèles dans le treillis du codeur, car ceci n 'est pas codée. Les deux valeurs possibles de $a_k^{(1)}$ produisent deux transitions parallèles, tandis que l'entrée $a_k^{(2)}$ contrôle entièrement les couples des transitions qui lient les états entre eux.

D'aprés le principe de la TCM et la figure 2.6 , on peut conclure que ce codeur est basé sur la partition C de l'alphabet, c'est-à- dire qu' un sous- ensemble Ci est associé à chaque branche dans le treillis qui est donné à la figure 2.11, donc les symboles d 'une même branche sont contenus dans le même sous-ensemble, d'où les sorties du codeur convolutif $b_k^{(2)}$ et $b_k^{(3)}$ dans la figure 2.9,

sélectionnent l'un des sous-ensembles C0, C1, C2 ou C3 et $b_k^{(1)} = a_k^{(1)}$ sélectionne un point dans le sous-ensemble sélectionné par $b_k^{(2)}$ et $b_k^{(3)}$.

Maintenant, on attribue les sous-ensembles aux couples des transitions pour essayer de maximiser la distance minimale. Une heuristique simple est d'essayer de laisser la distance entre les branches divergeant ou convergeant, aussi large que possible, par exemple C0 est le sous-ensemble le plus éloigné de C2 ainsi les deux couples de transitions émergés dans le premier état doivent être attribuées aux C0 et C1 si on attribue aux deux couples de transitions partant du même état, C0 et C2 également.

On continue avec cette règle, on trouve le treillis correspondant au codeur de la figure 2.8 .

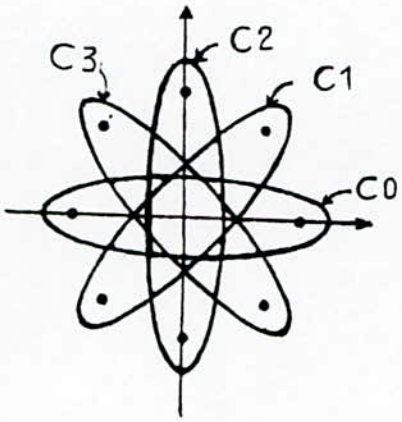


Figure 2.10 Les sous ensembles Ci obtenus à partir de la partition de 8-PSK de la figure 2.6

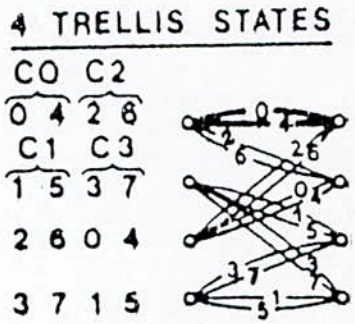
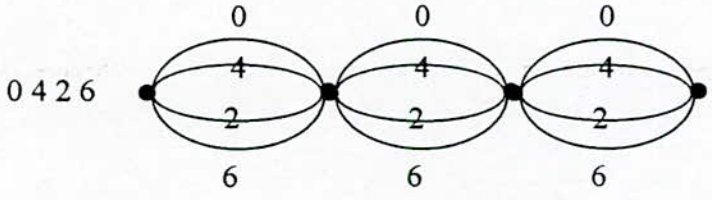


Figure 2.11 Le treillis du codeur de la figure 2.9

A l'aide de la figure 2.11, on peut déduire que la distance euclidienne entre les transitions parallèles est 2, et la distance minimale du treillis de la figure 2.10 est $\sqrt{6-\sqrt{2}} = 2.14 > 2$, donc d'après la formule (2.8) la distance minimale du codeur en treillis est $d_{free} = 2$ et $N(d_{free}) = 1$.

2.3.3 Les règles de partitionnement [17] , [12]

Tout d'abord, avant de donner les règles de partitionnement, on donne quelques exemples. Nous expliquons avec détail l'étude heuristique des codes 8-PSK pour un débit de transmission de 2 bits / T (T: la durée d'un symbole). La modulation non codée 4-PSK est utilisé comme système de référence.



$d_{fee}=d_{1\ 8\text{-}PSK}=1.414$

$P_e(e) \gtrsim 2Q(d_{fee}/2\sigma)$

Figure 2.12 Modulation non codée 4 -PSK .

Comme le montre la figure 2.12, on peut considérer la modulation 4-PSK non codée comme une modulation codée avec un treillis à un seul état, et quatre transitions parallèles auxquelles ces dernières sont assignées à partir des ensembles de la 8-PSK (de la figure 2.6) à quatre signaux, avec une grande distance minimale entre eux i.e. entre les signaux du sous-ensemble B0 (ou B1), le premier code illustré dans la figure 2.13 est le plus simple code à trouver. Les signaux des sous-ensemble B0 et B1 sont attribuées aux transitions d'origine du 1^{er} et $2^{éme}$ états respectivement, qui garantissent que la distance euclidienne libre est aux moins supérieure que celle de la modulation 4-PSK non codée. On remarque dans ce code que les symboles attribuées aux transitions sortent d'un état, ne sont pas les mêmes que celles attribués aux transitions entrant à la même état, et pour cela la gain en terme de la distance euclidienne libre à travers la 4-PSK reste limité à 1.1 dB.

Les autres codes 8-PSK, dans la figure 2.13 avec 4, 8 et 16 états, et ayant les gains en terme de la distance euclidienne limite 3dB, 3.6 dB et 4.1 dB respectivement exigent plus d'effort que ceux à 2 états. Donc *lorsque la largeur de la contrainte (ou le nombre des états) augmente la distance euclidienne limite augmente aussi,* si on applique les règles suivantes

i) Tous les signaux 8-PSK doivent arriver avec la même fréquence (i.e la même probabilité d'apparition);

ii) Les transitions qui proviennent du même état, reçoivent les signaux de l'un des sous-ensembles B0 ou B1;

iii) Les transitions qui se joignent au même état, reçoivent les signaux de l'un des sous-ensembles B0 ou B1;

iv) Les transitions parallèles reçoivent les signaux des sous-ensembles C0 ou C1 ou C2 ou C3.

La règle (i) reflète l'intuition qu'un bon code doit posséder une structure régulière, et les règles (ii), (iii), et (iv) nous garantissent que la distance euclidienne associée avec tous les événements d'erreur simples ou multiples augmente la distance euclidienne limite de la modulation 4-PSK non codée de 3dB au moins. On remarque qu'avec deux états seulement, les règles (ii) et (iii) ne sont pas vérifiées simultanément, puisque ceci n'est pas le seul code à 8-PSK, on devra donc plus s'intéresser aux codes à deux états.

Notons que les transitions parallèles impliquent, que les événements d'erreur simples peuvent se produire. Cela limite la distance euclidienne limite par la distance minimale dans les sous-ensembles des signaux attribués aux transitions parallèles. D'autre part, ces transitions parallèles réduisent la connectivité dans le treillis, et permettent ainsi l'extension de la longueur minimale des événements d'erreur multiples.

Ces idées peuvent être appliquées aussi aux autres formes de modulations. Par exemple, le cas de la transmission de 3 bits/T par la modulation codée 16-QAM. La modulation 8-PSK est considérée comme un système de référence pour cette dernière. D'après la discussion précédente et la partition de

la 16-QAM en sous-ensembles qui est illustrée dans la figure 2.4, on peut tirer facilement des codes à 16-QAM, on remplace tout simplement les signaux correspondants (D0 à D7) de la 16-QAM.

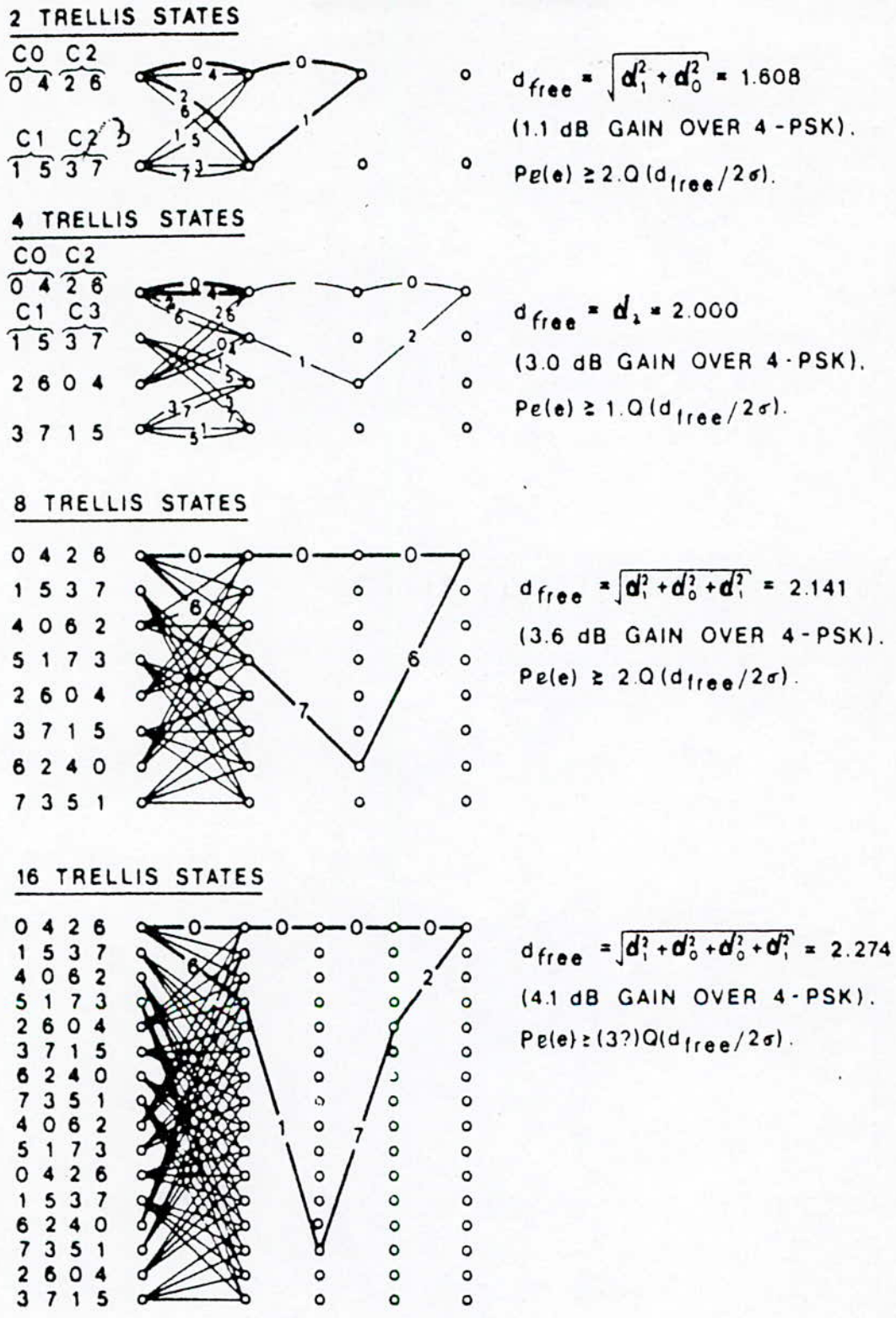


Figure 2.13 Modulation codée 8-PSK, 2 bit / *T*.

Un code à 8 états, 16-QAM obtenu par cette manière est présenté dans la figure 2.14, mais il faut toujours rendre tous les précautions envers cette approche. Il est à noter que la distance D0 de la modulation 16-QAM (figure 2.4) est calculée, en considérant que le pas entre deux points voisins dans

la même ligne égale à 2, ce qui donne une énergie moyenne de la constellation 16 QAM égale à 10, donc $d_0 = \frac{2}{\sqrt{10}}$.

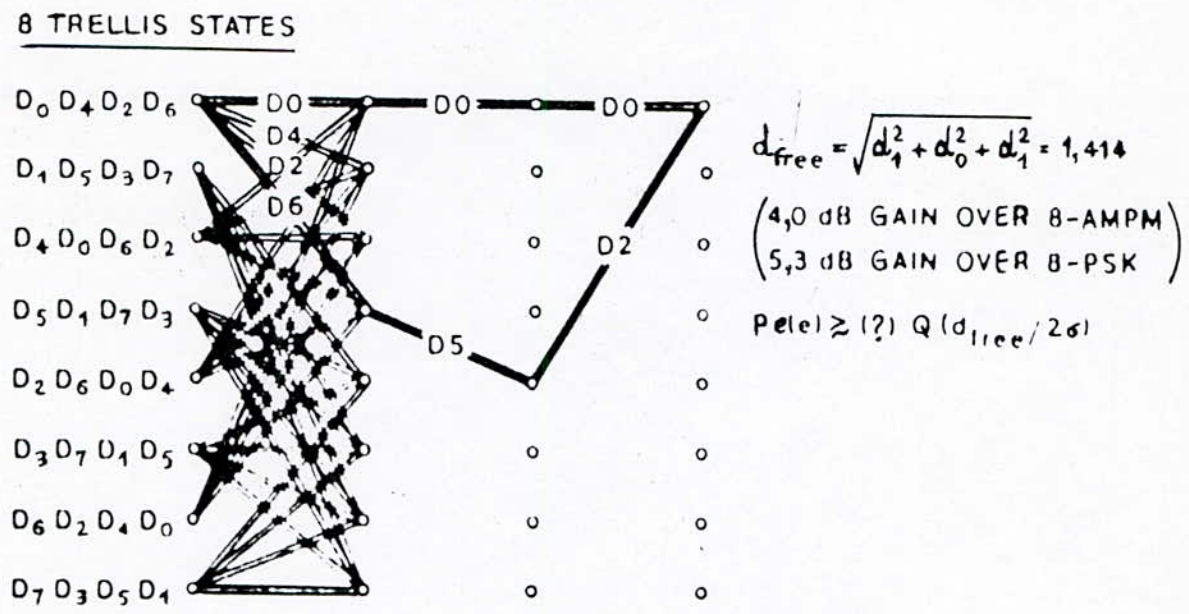


Figure 2.14 Modulation codée 16-QAM 3 bits / *T*.

La partition d'ensemble des signaux *M*-AM à une dimension produit des distances minimales entre les sous-ensembles $d_{i+1} = 2d_i$, i=0,1,... (6 dB par pas). Alors que pour l'ensemble des signaux *M*-QAM à deux dimensions nous avons $d_{i+1} = \sqrt{2}d_i$, i=0,1,... (3 dB par pas).

On remarque que la correspondance spécifiée, des bits codées en signaux du canal, indiquée dans les figures 2.3 et 2.4 n'est pas importante mais ce qui est important est que par des permutations des sous ensembles, on obtient les autres correspondances avec le même incrément des distances minimales des sous ensembles.

On peut donner Maintenant les règles de partitionnement d'une manière générale [12]

i) On maximise la distance entre les transitions parallèles;

ii) On maximise la distance entre les transitions qui partent du même état ou arrivent au même état;

iii) On utilise la même fréquence pour tous les symboles.

Ces règles sont heuristiques, et ne donnent pas nécessairement des codes optimales dans n'importe quel sens.

2.3.4 Gain de codage par rapport à une référence [3]

Le Gain de codage est définie comme étant la différence entre les deux valeurs du RSB en dB, nécessaire pour atteindre la même probabilité d'événement d'erreur $P_e(e)$ dans les systèmes sans codage et avec codage respectivement

$$g \hat{=} RSB_{sc}\Big|_{P_e(e)} - RSB_{ac}\Big|_{P_e(e)} \qquad (2.11)$$

On définit le gain asymptotique d'une TCM comme suit

$$g_\infty \hat{=} g\Big|_{RSB \longrightarrow \infty} = 10\log(d^2_{free} / d^2_{sc}) \qquad (2.12)$$

où d_{sc} est la distance minimale dans le système sans codage. L'équation (2.12) montre l'importance de la distance minimale d_{free}.

En considérant l'exemple de la figure 2.7, on a trouvé que $d_{free} = \sqrt{10}$. Le système sans codage (système de référence) est le système à 2-PSK, qui a une d_{free}=2 et qui transmet la même puissance qu'un système à 4-PSK. Le Gain asymptotique de codage est donc

$$g_\infty = 10\log(\frac{10}{4}) = 4dB$$

Le système avec codage est donc meilleur que celui sans codage de 4 dB approximativement .

2.4 Evaluation d'un codeur en treillis

Nous pouvons considérer un codeur en treillis comme une machine à nombre d'états fini, dont la structure peut être mise en évidence à l'aide de l'un de deux diagrammes : diagramme en treillis ou le diagramme d'état. Nous allons illustrer ces deux représentations à l'aide de l 'exemple de la figure 2.7.

2.4.1- Diagramme en treillis

Soit un codeur convolutif de taux 1/2 avec une longueur de contrainte K = D+1, où D est le nombre des éléments mémoires. Nous définissons l'état σ d'un codeur convolutif comme le nombre binaire des (K-1) plus anciens symboles binaires présents dans son registre, le plus récent de ces symboles étant écrit en premier.

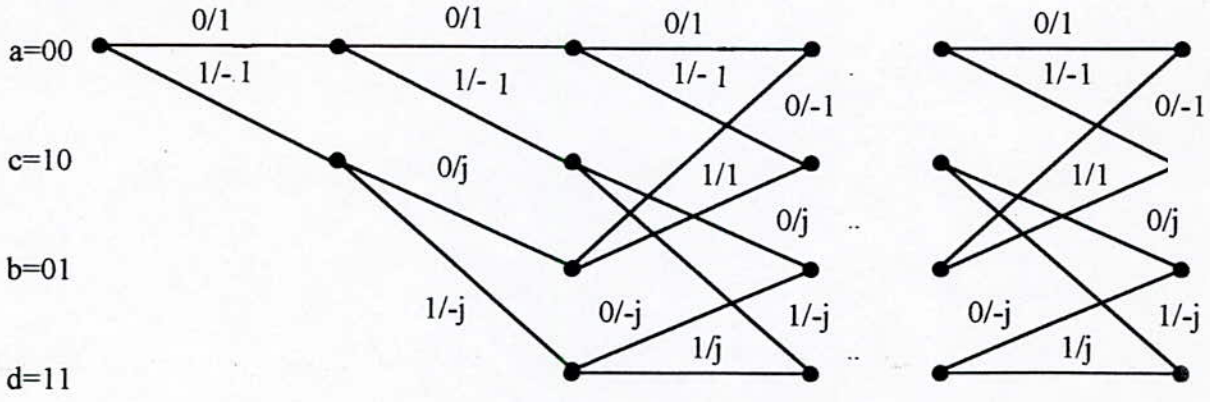


Figure 2.15 Diagramme en treillis associée au codeur de la figure 2.7.

Le diagramme en treillis contient (L+K) niveaux où K est la longueur de la contrainte du codeur

Les (K-1) niveaux correspondants au départ du codeur de l'état initial (a). Nous remarquons aussi que nous ne pouvons pas atteindre tous les états dans cette partie du treillis. Cependant dans la partie centrale du treillis, pour laquelle le niveau t reste borné par $K-1 \leq t \leq L$, tous les états du codeur peuvent être atteints . Nous pouvons aussi noter que la partie centrale du treillis conserve une structure fixe périodique .

Considérons la portion du treillis qui correspond à la transition entre les deux niveaux t et $t+1$ avec $t \geq 2$, de sorte que l'état le plus récent peut être l'un des quatre états a,b,c ou d. Cette portion est donnée par la figure 2.16 .

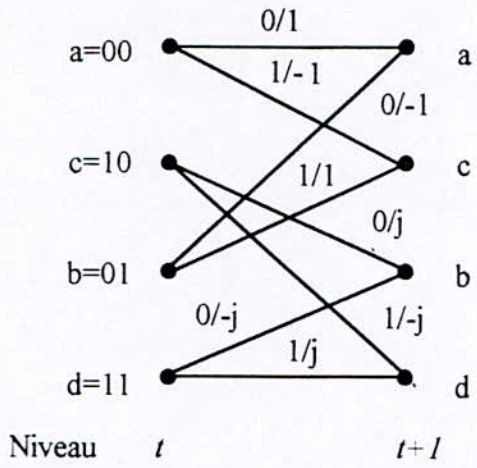


Figure 2.16. Portion centrale du treillis de la figure 2.6.

2.4.2. Diagramme d'état

En fusionnant les noeuds des niveaux t et $t+1$ de la figure 2.16, nous obtenons le diagramme d'état représenté sur la figure 2.16. Les noeuds de la figure 2.17 représentent les quatre états possibles du codeur . pour chaque noeud, on à deux branches sortantes . A l'aide du diagramme d'état on peut déterminer la sortie du codeur pour n'importe qu'elle entrée .

Considérons par exemple la séquence message (1, 0 , 0 , 1 , 1) . Pour cette entrée, on suit le chemin a c b a c d , d'ou la séquence de sortie (-1 , j , -1 , -1 , -j).

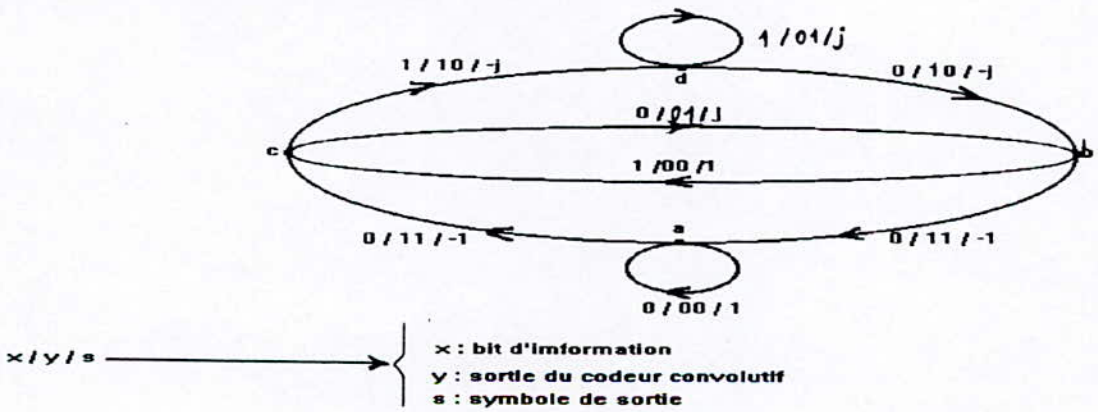


Figure2.17 Diagramme d'état associé au codeur de la figure 2.6.

Pour généraliser à des codes en treillis à taux d'émission m/(m+1), nous remarquons simplement que le diagramme en treillis comporte dans ce cas 2^m branches issues de chaque noeud. L'effet de la longueur de contrainte finie K est le même que ci-dessus, et donc au delà des K premières branches, les chemins commencent à converger par groupe de 2^m. Si le codeur en treillis ayant ν éléments mémoires et m' entrées à passer par ces éléments (ou m' entrées du codeur convolutif), alors [3]

* Le nombre d'états possibles est 2^{ν}.
* Le nombre des transitions à partir de chaque état (ou vers chaque état) est 2^m.
* Le nombre d'états accessibles à partir de chaque état est $2^{m'}$.
* Le nombre des transitions parallèles partent de chaque état est $2^{m-m'}$.

Par conséquent, le diagramme d'état aura aussi 2^{ν} noeuds, chacun ayant 2^m branches qui en partent et autant qui arrivent.

2.4.3 La fonction de transfert et propriété de distance des codes en treillis [6],[3]

Nous pouvons calculer l'ensemble des poids d'Euclid associés aux chemins du code, ou de façon équivalente, l'ensemble des distances entre le chemin de référence et tous ceux qui sont séparés à l'aide du diagramme d'état. Pour simplifier les choses, considérons le même exemple traité depuis le début de ce chapitre, celui de la figure 2.7. le diagramme d'état du codeur est montré sur la figure 2.17. Nous commençons par la modification de ce diagramme en un autre représenté par la figure 2.18. Puisque on s'intéresse au codeur en treillis linéaire, donc sans perdre de généralité, nous scindons en deux états, l'état (a) à un état initial (a_0) et un état final (a_1), et sa propre boucle est éliminée. Un autre changement est que chaque branche est dénommée $D^w L^l I^i$, où l'exposant w sur les branches décrit le poids d'Euclid de séquence de sortie correspondante à cette branche (Le poids d'Euclid est le carré de la distance euclidienne) , l'exposant l est toujours égal à 'un' correspondant en fait, que la longueur de chaque branche est 'un' , et l'exposant i décrit le poids de Hamming de séquence d'entrée correspondante. Donc pour l'entrée 'Zéro' nous avons $I^0 = 1$, et pour l'entrée 'Un' nous avons $I^1 = I$.
Par exemple, le chemin a_0 c b d d b a_1 , est noté par $D^{14} L^6 I^3$, c-à-d que le poids d'Euclid de la séquence de sortie correspondante est 14 , la longueur du chemin est 6 , et le poids d'Euclid de l'entrée est 3 .

Nous définissons un chemin fondamental comme le chemin qui commence en (a_0) et se termine en (a_1). Soit $T_{w,l,i}$ le nombre de chemins de (a_0) vers (a_1) notés $D^w L^l I^i$. La fonction génératrice est définie comme étant

$$T(D,L,I) = \sum_w \sum_l \sum_i T_{w,l,i} D^w L^l I^i \qquad (2.13)$$

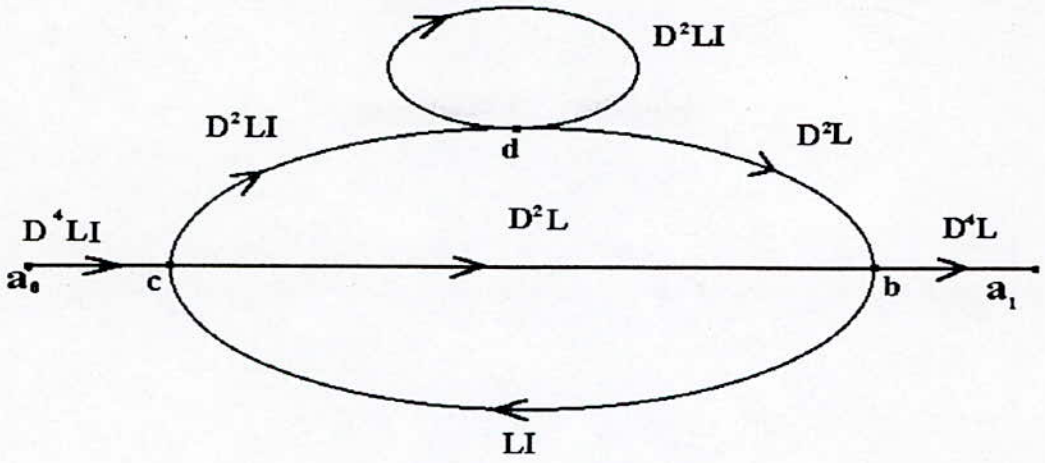


Figure 2.18 Diagramme d'état modifié.

Pour le codeur de notre exemple, nous pouvons calculer la fonction génératrice en considérant le diagramme d'état modifié comme étant un graphe de fluence à une seule entrée et une seule sortie.

La fonction génératrice peut être considérée comme la fonction de transfert du graphe de fluence . Ainsi nous traitons les noeuds comme des jonctions de sommation et les notations de branches comme des gains, nous pouvons écrire

$$\left.\begin{aligned} X_c &= D^4 LIX_{a_0} + LIX_b \\ X_b &= D^2 LX_c + D^2 LX_d \\ X_d &= D^2 LIX_c + D^2 LIX_d \\ X_{a_1} &= D^4 LX_b \end{aligned}\right\} \qquad (2.14)$$

et $$T(D,L,I) = X_{a_1} / X_{a_0} \qquad (2.15)$$

En résolvant le système (2.14) , On trouve

$$T(D,L,I) = \frac{D^{10}L^3 I}{1 - D^2 LI(1+L)} \qquad (2.16)$$

En utilisant le développement en série de Maclaurin, on peut réécrire l'expression (2.16) en série de puissance

$$T(D,L,I) = D^{10}LI + D^{12}L^4I^2(1+L) + D^{14}L^5I^3(1+L)^2 \qquad (2.17)$$

Nous pouvons faire les remarques suivantes

• Il existe un seul chemin fondamental a le carré de la distance euclidienne égale à 10 du chemin de référence qui est le chemin de tous les 1. Il diverge de celui-ci sur les trois branches qui précédent le noeud à partir duquel il se confond avec lui . en plus, il diffère du chemin de référence en un seul bit d'entrée .

• Il y a deux chemins fondamentaux ont le carré de la distance euclidienne égale à 12 du chemin de référence dont l'un a divergé quatre branches en amont et l'autre cinq, et les deux diffèrent du chemin de référence en deux bits d'entrée, et ainsi de suite .

En général, pour un code en treillis la plus petite distance euclidienne de n'importe quel chemin fondamental par rapport au chemin de référence est égal au distance limite du code (distance libre) . Ainsi, le code en treillis dont la fonction génératrice est donnée par (2.17) a la distance limite (d_{free}) qui est égale à $\sqrt{10}$.

2.4.4. Notion d'événement d'erreur et sa probabilité [22],[3]

Un événement d'erreur est formé par un couple de séquences distingues ayant le même état de départ et le même état d'arrivé dans le treillis après un certain nombre de pas. Avec plus de précision, un événement d'erreur de longueur l est défini par deux séquences

$$(S_t) = (S_t, \ldots, S_{t+l-1})$$

$$(S_t)' = (S_t, \ldots, S_{t+l-1})$$

tel que $\sigma_t = \sigma_t'$ et $\sigma_{t+l} = \sigma_{t-l}'$

$$\sigma_i \neq \sigma_j \quad i = t+1, \ldots, t-l-1$$

où σ_t représente l'état du codeur à l'instant t.

On considère que la probabilité d'événement d'erreur est notée par $P_e(e)$. le schéma de la figure 2.19 représente deux chemins divergent de la même état σ_t et convergent vers le même état σ_{t+l} .

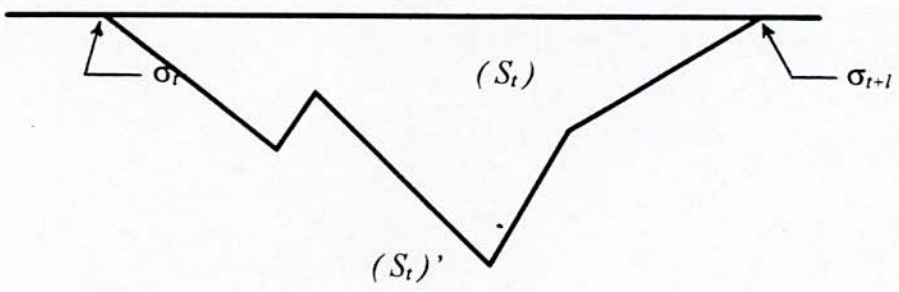


Figure 2.19 Exemple sur un événement d'erreur.

Sans perdre de généralité, nous pouvons prendre comme chemin correct celui de Haut et supposer que celui de bas est choisi par le décodeur de vraisemblance maximale. Pour que cela se produit , il faut que la différence de poids d'Euclid

$$\Delta W\left((S_j),(S_j)'\right) = d_E^2\left((S_j),(S_j)\right) - d_E^2\left((S_j)',(S_j)\right) \geq 0 \qquad (2.18)$$

tel que (S_t) est la séquence correcte, et $(S_t)'$ est la séquence incorrecte qui est choisie par le décodeur de vraisemblance maximale. (S_t) est la séquence reçue . Nous dirons alors qu'il y a un *événement d'erreur au* noeud $j = t / T$ où T est la période d'échantillonnage qui est prise généralement égale à l'unité du temps .

La probabilité d'erreur est notée $P_e(j)$. Nous pouvons donc borner supérieurement la probabilité d'erreur au noeud j par la probabilité qu'un chemin se sépare en j du chemin correct, qui a une distance euclidienne inférieure à celle du chemin correct par rapport à la séquence reçue dans la partie ou les deux chemins sont distincts.

$$P_e(j) \leq Pr\left[\bigcup_{(S_t)'\in\prod((S_t))}\left\{\Delta W((S_t),(S_t)') \geq 0 / (S_t)\right\}\right] \qquad (2.19.a)$$

où $\prod((S_t))$ est l'ensemble de tous les chemins incorrects qui peuvent être choisis par le décodeur de vraisemblance maximale au lieu de la séquence correcte (S_t). On fait la moyenne sur tous les séquences qui peuvent être produises au noeud j , sur les deux membres de l'inéquation (2.19.a) on obtient

$$P_e(j) \leq \sum_{(S_t)} Pr((S_t))\, Pr\left[\bigcup_{(S_t)'\in\prod((S_t))}\left\{\Delta W((S_t),(S_t)') \geq 0 / (S_t)\right\}\right] \qquad (2.19.b)$$

(2.19.b) donne

$$P_e(j) \leq \sum_{(S_t)} Pr((S_t)) \sum_{(S_t)'\in\prod((S_t))} Pr\left[\Delta W((S_t),(S_t)') \geq 0 / (S_t)\right] \qquad (2.20)$$

Pour simplifier le calcul , on fait une permutation entre les deux opérateurs de la sommation de (2.20). Maintenant, le deuxième membre de (2.20) peut être calculé par la moyenne sur tous les chemins corrects possibles, pour une séquence d'erreur $(S_j)'$ donnée, puis avec la sommation sur toutes les séquences incorrectes possibles ε , qui peuvent être choisies par le décodeur , on obtient

$$\begin{aligned} P_e(j) &\leq \sum_{(S_t)'}\sum_{(S_t)} Pr((S_t))\, Pr\left[\Delta W((S_t),(S_t)') \geq 0 / (S_t)\right] \\ &= \sum_{(S_t)} Pr((S_t))\, Pr\left[\Delta W((S_t),(S_t)') \geq 0\right] \\ &= \sum_{\varepsilon} P_e(\varepsilon) \end{aligned} \qquad (2.21)$$

Chaque terme de cette sommation est la probabilité d'erreur pour un couple de vecteur du code (ou séquences) dans la partie ou les deux chemins sont distincts. Pour un canal gaussien perturbé par un bruit blanc gaussien additif de variance σ^2 la probabilité de sélectionner une séquence $(S_t)'$ au lieu de la séquence correcte (S_t) est

$$P(d) = P((S_t) \rightarrow (S_t)') = \frac{1}{2} erfc(\frac{d}{\sqrt{8}\sigma}) \qquad (2.22)$$

où d est la distance euclidienne entre les deux séquences, et la fonction *erfc* est donnée par (1.27) . On peut partitionner l'ensemble des événements totaux ε en sous-ensembles ε_i tels que chaque ensemble ε_i contient les événements qui possèdent la même distance euclidienne d_i . Si on note par $N(d_i)$ au cardinal d'un sous -ensemble ε_i , alors l'inéquation (2.21) donne

$$P_e(1) \leq \sum_i N(d_i) P(d_i) \qquad (2.23)$$

Car $P_e(e)$ est la probabilité qu'une erreur se produise à l'instant $t=jT$, au noeud j . La probabilité d'événement d'erreur a un instant t permet au décodeur d'identifier correctement l'état transmis à cette instant .

- ***Borne inférieur BI de $P_e(e)$***

IL est claire que la probabilité d'événement d'erreur est minorée par la probabilité d'erreur mutuelle donnée dans (2.22) pour $d = d_{free}$, ce qui donne une première borne inférieure

(BI) $$P_e(e) \rangle \frac{1}{2} erfc\left(\frac{d_{free}}{\sqrt{8}\sigma}\right) \qquad (2.24)$$

Soit l'équation (2.22). Pour un RSB très grand, les quantités $\left(\frac{d_i}{\sqrt{8}\sigma}\right)$ tendent vers l'infini , donc on peut négliger les termes faibles dans (2.23), et on obtient alors

$$P_e(e) = \frac{1}{2} N(d_{free}) erfc\left(\frac{d_{free}}{\sqrt{8}\sigma}\right) + \varepsilon(\sigma) \qquad (2.25)$$

où $\varepsilon(\sigma)$ est une quantité positive très faible . Donc

$$P_e(e) \geq \frac{1}{2} N(d_{free}) erfc\left(\frac{d_{free}}{\sqrt{8}\sigma}\right) \qquad (2.26)$$

d'où l'estimation asymptotique de $P_e(e)$.

- ***Borne supérieur (BS) de $P_e(e)$***

Remplaçons $P(d_i)$ de l'équation (2.22) par leur expression dans (2.23) on obtient

$$P_e(e) \le \sum_i \frac{1}{2} N(d_i) erfc\left(\frac{d_i}{\sqrt{8\sigma}}\right) \qquad (2.27)$$

D'après la définition des coefficients $N(d_i)$ qui est donnée ci-dessus , on peut les calculées à partir de la fonction de transfert du codeur en treillis, car ces coefficients ne sont autres que les coefficients T_w de l'équation (2.13) avec (L=1, I=1). On sait bien que

$$erfc\left(\sqrt{x+y}\right) \le erfc\left(\sqrt{x}\right) e^{-y} \ldots\ldots\ldots x \ge 0, y \ge 0 \qquad (2.28)$$

donc

$$erfc\left(\frac{d_i}{\sqrt{8\sigma}}\right) \le erfc\left(\frac{d_{free}}{\sqrt{8\sigma}}\right) . e^{-\left(d_i^2 - d_{free}^2\right)/8\sigma^2}$$

et (2.27) dévient

$$P_e(e) \le \frac{1}{2} erfc\left(\frac{d_{free}}{\sqrt{8\sigma}}\right) . e^{\frac{d_{free}^2}{8\sigma^2}} \sum_i N(d_i) . e^{-\frac{d_i^2}{8\sigma^2}}$$

D'après (2.13) (avec L=1 , I=1) on obtient

(BS) $$P_e(e) \le \frac{1}{2} erfc\left(\frac{d_{free}}{\sqrt{8\sigma}}\right) . e^{\frac{d_{free}^2}{8\sigma^2}} T(D)_{/D=e^{-1/8\sigma^2}} \qquad (2.29)$$

d'où une borne supérieure de la probabilité $P_e(e)$.

2.4.5 Probabilité d'erreur par bit $P_b(e)$

Ce paramètre est le plus important dans l'utilisation, car il est considéré comme la mesure la plus utile de la qualité du système. Toutefois, la probabilité d'erreur par bit ne peut être calculer d'une manière aisée car il dépend du codage utilisé pour affecter les bits à émettre aux points de la constellation. On peut démontrer facilement comme dans le paragraphe précédent, que

$$P_b(e) \le \frac{1}{m} \sum_i \sum_j N(d_i) b_j P(d_i) \qquad (2.30)$$

où m est le nombre des entrées du codeur et b_j est le nombre des bits erronés (dans la séquence d'entrée, après le décodage) dans un événement d'erreur ayant une distance d_i donc les coefficients b_j ne sont autres que les puissances de la variable I dans l'équation (2.13). Donc avec la même procédure que précédemment, on obtient

$$P_b(e) \le \frac{1}{2m} erfc\left(\frac{d_{free}}{\sqrt{8}\sigma}\right) e^{\frac{d_{free}^2}{8\sigma^2}} \left.\frac{\partial T(D,I)}{\partial I}\right|_{I=1, D=e^{-1/8\sigma^2}} \qquad (2.31)$$

d'où une borne supérieure de la probabilité $P_b(e)$

2.5 Décodage des TCM

2.5.1 Introduction

Il existe plusieurs algorithmes de décodage (détection) de TCM, nous nous limiterons à présenter le plus utilisé : L'algorithme de Viterbi. Cet algorithme suit la règle de décision du maximum de vraisemblance, et recherche parmi toutes les séquences possibles de symboles que le codeur peut émettre, celle qui a la distance minimale de la séquence fourni par le démodulateur.

Pour présenter cet algorithme, supposons que le canal de transmission soit de type stationnaire gaussien et sans mémoire.

2.5.2 Décodage à maximum de vraisemblance

Avant d'exposer le principe de l'algorithme de Viterbi, nous allons montrer que dans le cas d'un canal gaussien, le décodeur à vraisemblance maximale se réduit au décodage à distance euclidienne minimale.

Soit la figure 1.9. Supposons que lorsque le vecteur Z est observé, on décide en faveur de m_i, c'est-à-dire : $\hat{m} = m_i$, la probabilité moyenne de l'erreur pour cette décision est

$$\begin{aligned} P_e(m_i, Z) &= P(m_i \ n'est \ pas \ transmis / Z) \\ &= 1 - P(m_i \ transmis / Z) \end{aligned} \qquad (2.32)$$

Puisqu'on doit minimiser $P_e(m_i, Z)$, il est équivalent à maximiser $P_e(m_i \ transmis / Z)$. Donc la règle de la décision optimale est

On décide pour $\hat{m} = m_i$, si

$$P(m_i \text{ émis/Z}) \ge P(m_e \text{ émis/Z}), \quad \forall k \ne i, \ k = 1,...,M$$

Cette règle de décision est appelée décision au sens des maximum à posteriori où (MAP). En utilisant la loi de Bays [16], on peut écrire la règle précédente comme suit

On décide pour $\hat{m} = m_i$, si

$$\frac{P_k f_Z(Z/m_k)}{f_Z(Z)} \quad \text{est maximum pour } k = i.$$

mais pour des symboles équiprobables $P_k = \frac{1}{M}$.

D'autre part $f_z(Z)$ est indépendant du signal émis. Donc la règle de décision devient

On décide pour $\hat{m} = m_i$, si

$$f_Z(Z/m_k) \text{ esr maximum pour } k = i,$$

où $f_Z(Z/m_k)$ est appelée fonction de vraisemblance qu'on remplace souvent par son logarithme qui donne :

On décide pour $\hat{m} = m_i$, si

$$ln\left[f_Z(Z/m_k)\right] \text{ est maximum pour } k = i.$$

Une telle décision est appelée « décision au sens du maximum de vraisemblance ». Donc cette règle peur être interprétée de la manière suivante

On calcule $ln\left[f_Z(Z/m_k)\right]$, $i = 1,2,...,M$, ensuite on compare et on décide en faveur du maximum donnée par $i = k$.

Il est préférable de donner une interprétation graphique de la décision de maximum de vraisemblance. Soit R_1, R_2,..., R_M, les zones correspondantes aux points message $S_i = (\alpha_{i1}\ \alpha_{i2}...\alpha_{in})^T$ représentées par les signaux $S_i(t)$, $i = 1,2,...,M$, et comme indiqué à la figure suivante

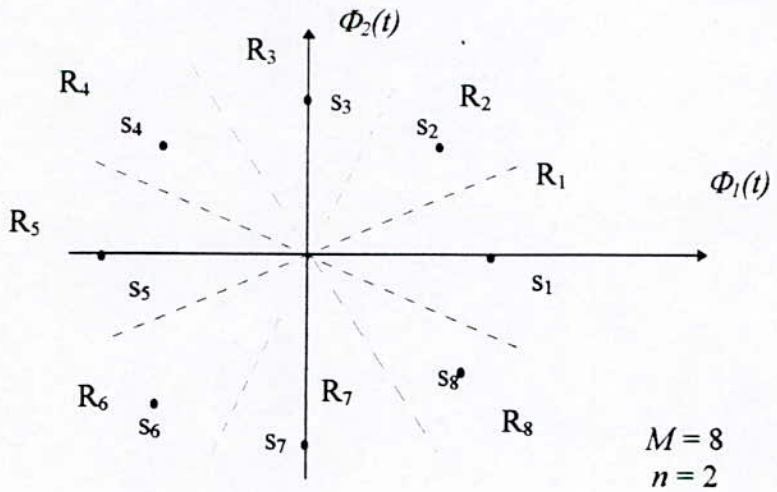


Figure 2.20 Huit zones de décision (R_1, R_2,..., R_8)

La décision MV peut être formulée de la manière suivante :

Le point d'observation représenté par le vecteur Z se trouve dans la région R_i (c'est-à-dire on décide en faveur de m_{ii}) si
$ln\left[f_Z(Z/m_k)\right]$ est maximum pour $k = i$.

Pour un canal gaussien

$$f_Z(Z/m_k) = (\pi N_0)^{-\frac{n}{2}} \cdot e^{-\frac{1}{N_0}\sum_{j=1}^{n}\left(z_j - \alpha_{kj}\right)^2},\ k = 1,...,M \qquad (2.33)$$

$$\ln f_Z(Z/m_k) = -\frac{n}{2}\ln(\pi N_0) - \frac{1}{N_0}\sum_{j=1}^{n}\left(z_j - \alpha_{kj}\right)^2, i = 1,\ldots,M \qquad (2.34)$$

La fonction $\ln\left[f_Z(Z/m_k)\right]$ est maximum lorsque $\sum_{j=1}^{n}\left(z_j - \alpha_{kj}\right)^2$ est maximum.

Donc la règle de décision, donnée ci-dessus devient

On décide en fonction du symbole m_i, (c'est-à-dire Z se trouve dans la région R_i)si

$$\|Z - S_k\|^2 = \sum_{j=1}^{n}(z_j - \alpha_{kj})^2 \quad \text{est minimum pour } k = i.$$

On remarque $\|Z - S_k\|^2$ n'est que la distance euclidienne entre les valeurs z_i et α_{ki} qui représente la distance entre le point d'observation et le point message. Dans la dernière règle donnée ci-dessus, on cherche le point message le plus proche du point observé et ce au sens d'Euclid.

2.5.3 Algorithme de V iterbi

Concéderons l'exemple de la TCM de la figure 2.7 dont la représentation en treillis est donnée par la figure 2.7, le décodage à maximum de vraisemblance consiste à choisir le chemin correspondant à la séquence des symboles qui diffère de la séquence reçue en une distance euclidienne minimale.

L 'algorithme peut être partagé en deux parties :

- la première s'étend du début du treillis jusqu'au niveau t = K - 1, où K est la longueur de contrainte. Le décodage consiste à attribuer une distance nulle à l'état initial, et à chaque autre état σ, atteint par la transition d'un état σ`, un carré de distance, qui est la somme des carrés de la distance à l'état σ` et la distance de transition. La distance de transition à un niveau t du treillis entre deux états σ et σ`, est la distance euclidienne entre le symbole correspondant à cette transition du codeur et le symbole de la séquence reçue.
- La deuxième partie s'étend du niveau t = K jusqu'au niveau t= [L / m] + K – 1 où L est la largeur du message et de m le nombre des entrées du codeur.

Le décodage consiste à choisir la branche pour laquelle la distance au noeud σ sera minimale et éliminer toutes les autres. La branche choisie est appelée *survivant*. Si le choix de plusieurs branches donnera la même distance minimale au noeud σ, alors peu importe, on choisira une de ces branches. Cette opération est répété avec tous les 2^{ν} états , et cela pour tous les niveaux de cette partie . A la fin de cette partie nous obtiendrons 2^{ν} chemins, desquels le décodeur choisira celui qui a la distance la plus proche, au sens d'Euclid de la séquence émise.

En résumé, l'algorithme de Viterbi peut être mis en œuvre de façon commode à partir de diagramme en treillis en appliquant les règles suivantes :

1.A partir du $K^{ème}$ *étape (niveau) du diagramme où* 2^m *branches arrivent à chaque état, calculer pour chacun des états, la distance euclidienne de chacun des* 2^m *chemins qui arrivent à l'état.Pour chaque*

état, conserver le chemin dont la distance euclidienne est la plus faible, (le survivant) et éliminer les autres.

2.Répéter l'opération pour chaque étape t du treillis tant que $t \leq [L / m] + K - 1$.

3.Choisir le chemin dont la distance euclidienne est la plus petite et éliminer tous les autres.

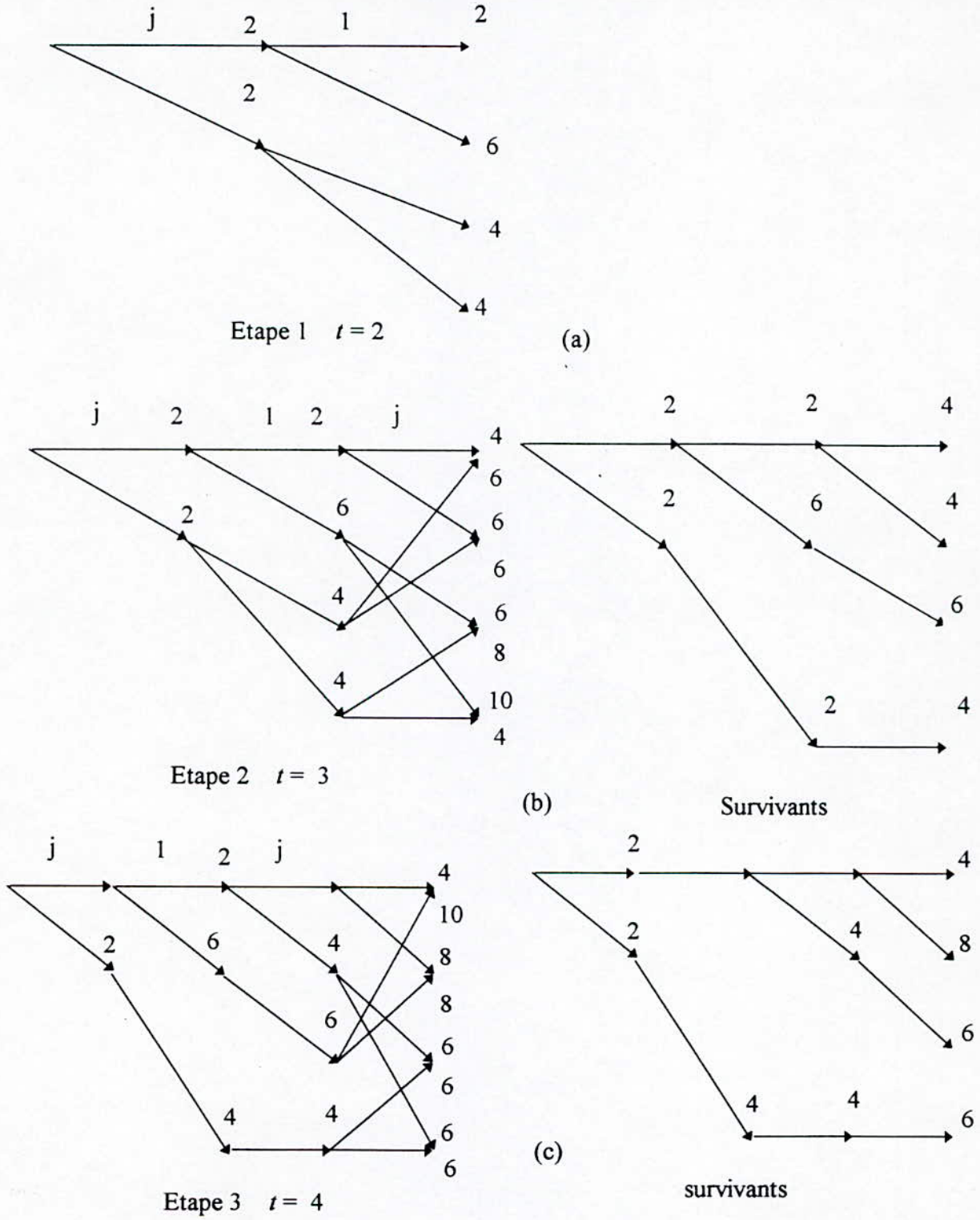


Figure 2.21 Exemple de décodage utilisant l'algorithme de Viterbi .

En notant, qu'il est nécessaire de stoker 2^{ν} chemins jusqu'à la fin , où la décision sera prise, on se rend compte que le décodeur devient très complexe et nécessite une capacité mémoire importante pour une longueur de contrainte importante.

Toutefois , lorsqu'on examine les différents survivants à une étape t , on remarque qu'avec une grande probabilité , ils ont tous le même passé entre les étapes 0 et t-dp où dp est la profondeur de décision .

On peut par conséquent décider définitivement à l'étape t toutes les données émises jusqu'à l'étape t -dp. Comme règle empirique , on peut fixer le paramètre dp égal à 5 où 6 fois le nombre d'états du décodeur . Avec ces valeurs les performances du décodeur sont quasiment les mêmes que lorsque la décision se fait en fin de la séquence.

Cette stratégie permet de réduire la complexité du décodeur et le retard de décision . La complexité d'un décodeur de Viterbi est proportionnelle au nombre d'états de treillis.

Celui-ci croit exponentiellement avec la longueur de contrainte. Pour cette raison les codes utilisés en pratique ont une longueur de contrainte inférieure à 10.

Exemple 2.4

Revenons à notre codeur de la figure 2.7. Supposons que ce dernier génère une séquence de n symboles tous des 1, laquelle après transmission à travers le canal donne la séquence reçue (j 1 j 1) comprenant deux erreurs. La figure 2.21.a. montre la première partie de décodage. La distance euclidienne à chaque état y est indiquée.

Les figures 2.21.b,c montre la deuxième partie, chacune de ces figures compte deux treillis, dans celui de gauche où à conserver tous les chemins aboutissant à chaque noeud , tandis que dans celui de droite on a gardé que les survivants.

Le chemin conservé après la décision finale est celui dont l'erreur cumulée est la plus petite , c'est à dire le chemin (a a a a) (a : c'est l'ètat 00) correspondant à la séquence des 1.

2.6 Applications

Après avoir exposé la TCM dans les systèmes de transmission numérique, nous allons maintenant donner quelques applications de la TCM dans certaines domaines de la communication..

2.6.1 Modems téléphoniques

La transmission de données sur le canal téléphonique a été pendant les années 60 et 70 à l'origine du développement d'un grand nombre de techniques de traitement du signal en télécommunications . La transmission d'un grand débit sur le canal téléphonique (sur une bande de fréquence de 3500 Hz environ) a nécessité la mise en oeuvre de modulations à grand nombre d'états

comme la 16 - QAM , la 32 -QAM et la 128- QAM ainsi que le développement d'ègaliseurs adaptatifs [1] pour compenser la réponse imparfaite du canal de transmission .

Jusqu'au milieu des années 70, l'effort en transmission de données sur le canal téléphonique a été fortement orienté vers l'égalisation et l'annulation d'échos adaptatives pour compenser les imperfections du milieu de transmission. Il était alors couramment admis que le débit ne peut dépasser 9.6 Kbit / s à cause du rapport RSB limité des liaisons . La théorie classique du codage correcteurs d'erreurs ne semblait pas bien adapté aux modulations utilisées et ne permettait pas un saut technologique en matière de débit que l'on peut atteindre .

C'est alors que le décodage en treillis a été introduit par Ungerboeck, ce qui a permis de franchir un saut spectaculaire en matière de qualité . Dés 1984, un code en treillis à 8 à états, associé à une modulation 32 - QAM, a été standardisé par le CCITT pour les liaisons a 9.6 Kbit /s sur le réseau téléphonique commuté . Le même code mais associé à une modulation 128 - QAM, a également été adopté comme standard pour les modems à 14.4 Bit/ s sur des liaisons 4 fils privées .

A peine quelques années après l'adoption de ces standards, des modems téléphoniques à 19.2 Kbit / s ont été développés par certains constructeurs utilisant des codes en treillis multi -dimensionnels .

L'utilisation de codes en treillis puissants et des techniques de filtrages adaptatifs sophistiquées ont ainsi permis d'augmenter le débit d' abord à 9.6 Kbit/ s et ensuite jusqu'au 19.2 Kbit /s , ce dernier débit ayant pendant des années été considéré comme le débit ultime que l'on ne peut dépasser sur le canal téléphonique .

Il se trouve qu'en matière de débit les modems téléphoniques n'ont toujours pas atteint leur limite et on assiste aujourd'hui au développement de modems V. Fast dont le débit peut atteindre 28 Kbit /s sur des liaisons de bonne qualité . Le standard V. Fast en cours de standardisation par les instances internationales fait appel à de multiples techniques de traitement de signal comme le codage en treillis multidimensionnels , le précodage , la prédistortion , l'égalisation , adaptative , ainsi que la mise en forme de constellation de type treillis (treillis shapping).

2.6.2 Faisceaux hertiziens

Au début de la numérisation, les faisceaux hertiziens faisaient usage de modulations simples comme 4 –PSK, mais l'utilisation efficace du spectre radioélectrique disponible a nécessité le développement de faisceaux hertziens utilisant des modulations à grand nombre d'état comme la 16 - QAM et la 64 - QAM . C'est la modulation 16 - QAM qui à permis la transmission d'un débit de 140 Mbit / s dans des plans de fréquence à espacement de 40 Mhz entre canaux adjacents en polarisations croisées .

Plus tard , c'est grâce à l'utilisation de la modulation 64 - QAM que la transmission d'un débit de 140 Mbit / s est devenue possible dans des plans de fréquence avec espacement de 30 Mhz entre

canaux adjacents . Aujourd'hui, il existe même des faisceaux hertziens utilisant la modulation 256 - QAM qui offre la capacité double de la 16 - QAM ou encore une capacité 33% au dessus de celle offerte par la 64 - QAM .

Le problème principal dans les faisceaux hertziens numériques est la propagation par trajets multiples qui dégrade sérieusement la qualité et limite la disponibilité des liaisons à grande capacité

L'effet de la propagation par multitrajets est autant plus important que le nombre d'états de la modulation est grand . Les premiers faisceaux hertziens à grande capacité faisant usage d'egaliseurs simples, placés en général à l'étage de fréquence intermédiaire (FI) du du récepteur et dont l'adaptation est basée sur des critères spectraux . L'objectif était simplement d'amplifier le signal dans les bandes de fréquence ou le spectre était fortement atténué par superposition de structures des trajets multiples, ou encore de créer une pente dans la bande du signal de manière à réduire l'effet des évanouissements sélectifs hors bande.

Dans cette approche, la compensation des distorsions spectrales n'est que très approximative et la distorsion du temps de groupe restait incompensée . L'utilisation de modulation à grand nombre d'états dans les faisceaux hertziens numériques de deuxième génération a nécessité la mise en oeuvre d'égalisateurs adaptatifs en bande de base optimisés selon le critère de forçage à zéro ou la minimisation de l'EQM (Erreur Quadratique Moyenne).

Depuis maintenant une dizaine d'années (1985), le codage correcteur d'erreur est presque systématiquement utilisé dans les faisceaux hertziens. Les modulations codées en treillis ont aussi fait leur entrée dans les faisceaux hertziens à grande capacité . La transmission du débit STM - 1 (155 Mbit /s) dans des plans de fréquence de 28 ou 30 Mhz utilise aujourd'hui une modulation 128 - QAM avec un codage treillis de dimension 4.

2.7 Conclusion

Nous avons présenté avec détail, la théorie des TCM basée sur la combinaison de la modulation et du codage linéaire binaire, pour des constellations à de dimensions inférieures ou égales à 2, nous avons justifié le choix des codeurs convolutifs de taux m / (m + 1) et le gain porté par ces derniers.

Pour un codeur en treillis à m entrées et ν éléments mémoires il y a $[(2M)! / M!]^N$ où ($N = 2^{\nu}$ et $M = 2^m$) codes possibles, le choix des codes utilisés en pratique parmis ces codes est basé sur la performance réalisée (moins d'erreur des transmission) et la simplicité des structures (implémentation non complexe des codeurs et décodeurs). Les codeurs convolutifs utilisés dans la simulation satisfont ces critères et sont par conséquent des codeurs optimaux.

Pour le transcodage, nous avons présenté en détail la partition d'ensemble avec ses règles , qui peuvent minimiser la distance limite. Cette distance est le paramètre le plus important dans la TCM, car

il détermine ses performances qui sont basées sur la fonction de transfert. Cette dernière est utilisée pour trouver deux bornes inférieures aux probabilités $P_e(e)$ et $P_b(e)$.

La probabilité d'événement d'erreur $P_e(e)$, à un instant donné, permet au décodeur d'identifier l'état transmis à cet instant. Ce décodeur utilise l'algorithme de Viterbi avec le principe de maximum de vraisemblance. Dans la TCM à deux dimensions, si on veut affecter plus d'informations à un point message, il faut augmenter la taille de constellation, donc les erreurs de transmission augmente aussi pour un RSB donné . Un remède à ce problème est d'augmenter la dimension de la constellation c'est-à-dire d'utiliser la TCM multidimensionnelle [15].

Enfin, on a présenté quelques applications sur les TCM à une seule et à deux dimensions. Le prochain chapitre sera la mise en oeuvre d'une simulation afin d'évaluer les performance des systèmes de communication utilisant cette technique de codage et de modulation.

CHAPITRE III :
SIMULATION, RESULTATS ET INTERPRETATIONS

CHAPITRE III

SIMULATION , RESULTATS ET INTERPRETATIONS

3.1. Introduction

La simulation permet l'évaluation des performances des systèmes de communication numériques, et constitue aussi un outil puissant qui aide à l'étude et la conception de tels systèmes. nous citons quelques raisons montrant l'importance de cet outil :

- Quelques niveaux d'analyse ne sont pas faisables avec les outils mathématiques traditionnels, surtout qu'on est en face d'un modèle de canal et d'un système complexe .

- Il est très facile de passer d'un modèle de simulation d'un système à son implémentation matérielle, et pour cela la distinction entre un modèle de simulation et un prototype d'un système donné devient minimale, d'ou l'intérêt économique de la simulation qui réduit considérablement le temps et le coût de réalisation d'un système , et qui se prête plus facilement à la correction et à l'amélioration qu'un prototype .

L'évaluation des performances se fait en calculant la probabilité d'erreur pour un système de communication numérique donné (figure 1.1) et un bruit donné.

3.2. Modèle de simulation

Le modèle utilisé lors de notre simulation est représenté par la figure 3.1. Il comporte :

- une source d'information binaire
- un codeur en treillis (ou une TCM)
- un canal gaussien supposé idéal (sans distorsions)
- une source de bruit
- un décodeur à maximum de vraisemblance , utilisant l'algorithme de Viterbi,
- un bloc de calcul de la probabilité d'événement d'erreur moyenne $P_e(e)$
- un comparateur pour calculer le nombre de bits erronés après le décodage .

Le décodage est détaillé dans le chapitre précédent.

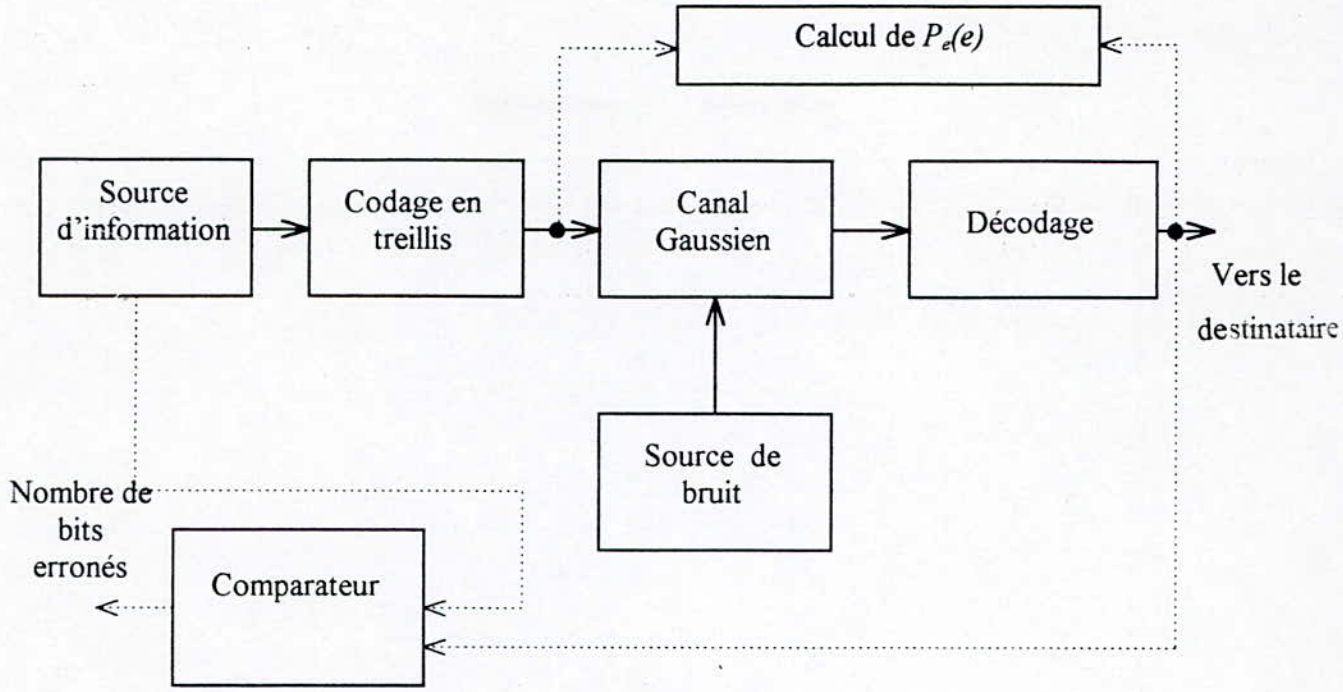


Figure 3.1. Modèle de simulation.

3.2.1. Source d'information

Pour que l'évaluation soit effectuée dans des conditions proches de ce qui est rencontre en exploitation, on doit choisir une séquence d'information qui simule le mieux possible le trafic réel tout en permettant une mesure simple .

La méthode universellement adoptée par la CCITT, consiste à utiliser une séquence dite pseudo-aléatoire, séquence périodique dont les propriétés statistiques sont « voisines » de celle d'un trafic réel « aléatoire » qui est généralement le résultat d'un brassage. Le générateur d'une telle séquence est constitué d'un registre à décalage comportant m étages, et des additionneurs modulo-2 (figure 3.2). Le choix de m détermine la période de la séquence, qui est égal à 2^m-1.

Les connexions aux additionneurs modulo-2 sont déterminées par des polynômes primitifs de degré m ayant la forme

$$h(x) = \sum_{j=0}^{m} h_j x^j \qquad \text{avec} \qquad \begin{cases} h_0 = h_m = 1 \\ h_j = 0 \text{ ou } 1 \end{cases} \qquad \text{pour } j=1,\ldots,m-1.$$

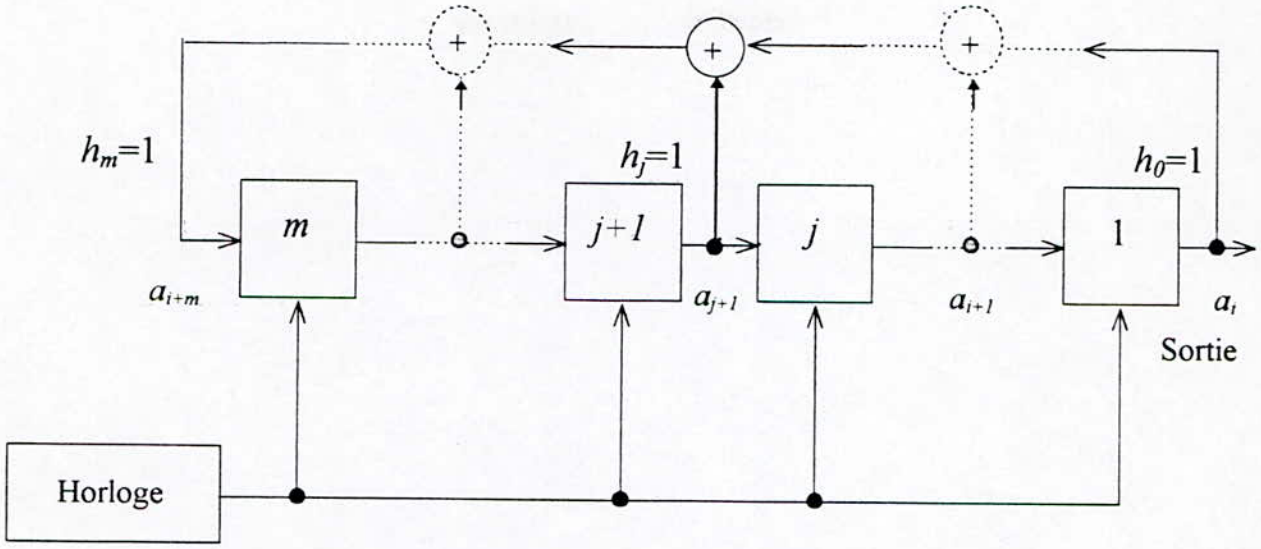


Figure 3.2. Générateur pseudo-aléatoire

Un polynôme primitif de degré m est un polynôme irréductible qui devise $x^{2m-1}+1$ et ne devise pas $x^n + 1$ pour $n < 2^m - 1$.

Chaque coefficient h_j égal à 1 correspond à une connexion .

La séquence générée comporte 2^{m-1} éléments égaux à 1 et à 2^{m-1}-1 égaux à 0 ce qui donne une répartition presque uniforme des 1 et des 0 .

Les principales propriétés de ces séquences sont rappelées à l'annexe A [13].

• ***Choix de la longueur de la séquence utilisée***

Le choix de la longueur de la séquence à utiliser dans la simulation, c'est -à - dire la longueur du registre qui l'engendre, est basé sur le fait qu'avec de faibles longueurs de registre (3 ou 4) on risque d'obtenir des résultats différents de la réalité, car une séquence courte ne représente pas un nombre statistiquement suffisant de configurations, donc il est recommandé d'utiliser des séquences relativement longues, supérieurs à 15 [4]. A cause du facteur de temps et de la mémoire limitée de l'ordinateur utilisé, nous avons choisis m= 16 (2^m-1= 65535).

3.2.2 Codage en treillis

• ***Codes convolutifs utilisés***

Dans l'utilisation , on évite les codes convolutifs « presque catastrophique » . Ces codes sont tels que l'exposant de I dans l'équation (1.7) est élevé pour un exposant de D voisin de d_{min} . Ces codes ne sont pas très performants puisqu'ils causent des séquences d'erreurs longues après décodage avec une probabilité non négligeable [4] . Un algorithme de recherche détaillé dans [17] permet d'obtenir des codes convolutifs très performants.

Dans notre simulation, on a choisi les codeurs convolutifs [17] illustrés à la figure 3.3 .

• ***Modulations utilisées***

Pour la modulation utilisée dans la simulation, notre choix reste limité aux modulations à deux dimensions. Les modulations et codeurs utilisés sont les suivants :

a) 4-PSK utilisée avec le codeur de la figure 3.3.a ;

b) 8-PSK utilisée avec le codeur de la figure 3.3.b ;

c) 8-PSK utilisée avec le codeur de la figure 3.3 .c ;

d) 8-PSK utilisée avec le codeur de la figure 3.3.d ;

e) 16-PSK qui est utilisée avec le codeur de la figure 3.3.e ;

f) 8-AM-PM qui est utilisée avec le codeur de la figure 3.3.b ;

g) 16-QAM qui est utilisée avec le codeur de la figure 3.3.e .

Pour le transcodage, on utilise bien sûr les règles de partitionnement, cette correspondance est exprimé à l'aide d'une formule mathématique . Par exemple pour l'exemple a),on a la correspondance suivante :

y_1	y_2	⟶	S_k
0	0		1
0	1		j
1	0		-j
1	1		-1

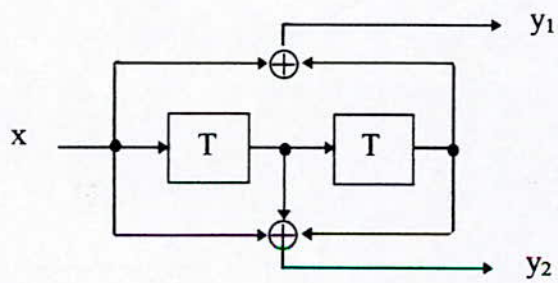
(où j est le nombre complexe de module égale à 1 et de phase égale à $\pi/2$).

La formule qui peut exprimer cette correspondance est donnée par

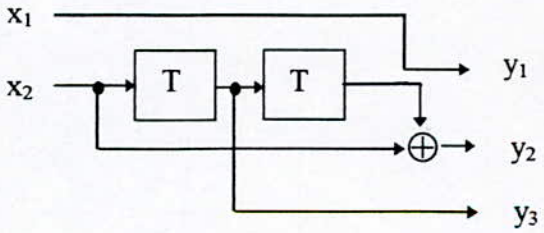
$$S = (1 - y_1)(1 - y_2) + j(1 - y_1)y_2 - j\, y_1(1 - y_2) - y_1 y_2$$

$$S = 1 - (y_1 + y_2) + j (y_2 - y_1)$$

ce qui montre que le codeur en treillis utilisé est linéaire .



(a)



(b)

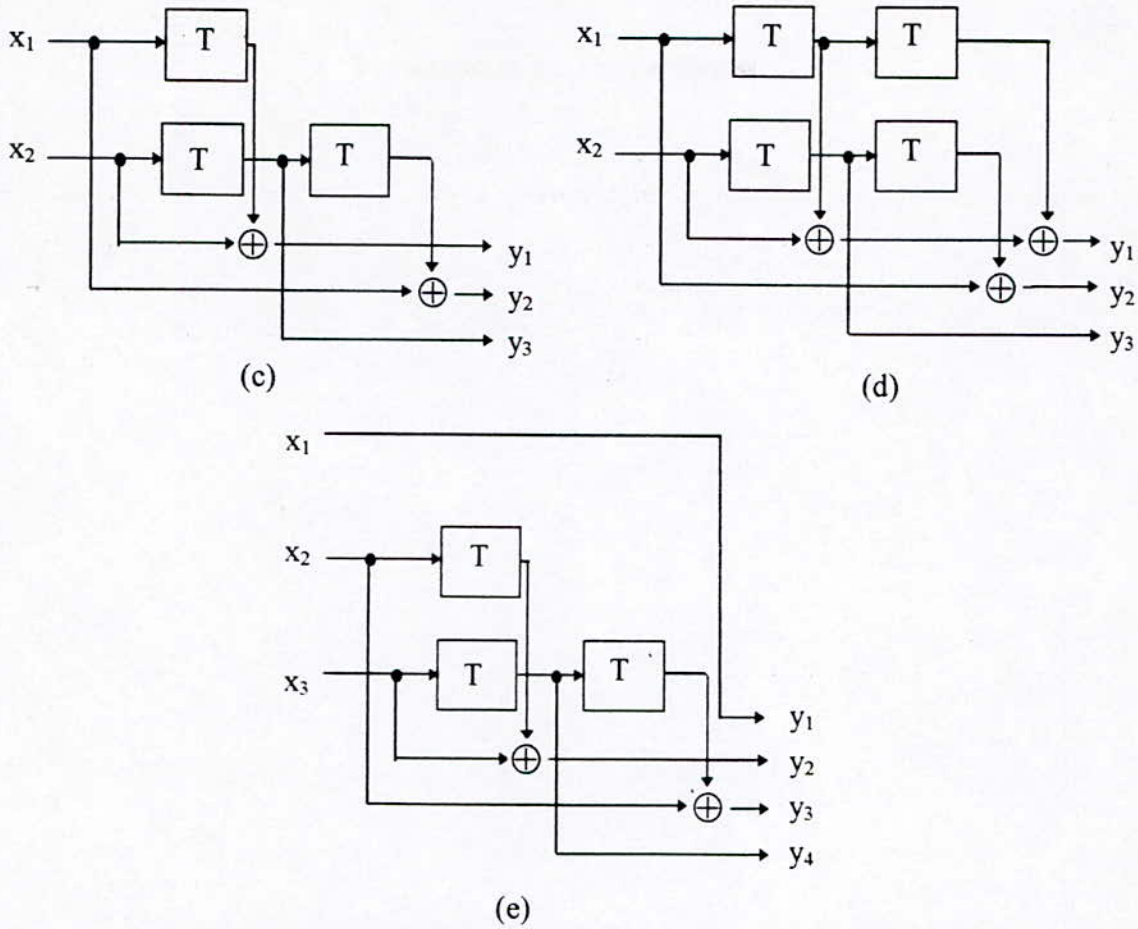


Figure 3.3 Les codeurs convolutifs utilisés dans la simulation.

(a) Codeur convolutif à 4 états et de taux 1/2, (b) Codeur convolutif à 4 états et de taux 2/3, (c) Codeur convolutif à 8 états et de taux 2/3, (d)Codeur convolutif à 16 états et de taux 2/3, (e) Codeur convolutif à 8 états et de taux 2/3.

3.2.3. Source de bruit

Le bruit envisagé est (AWGN)

- blanc
- additif
- gaussien.

On n'a pas pris en compte de l'effet du filtre de réception, c'est-à-dire que le bruit reste AWGN à la réception (car le filtrage rend ce bruit coloré) .

Pour tenir compte du bruit on a appliqué dans la simulation, la méthode directe (Monté-Carlo) qui fournit une suite d'échantillons de bruit qu'on ajoute aux symboles générés par le transcodeur .

Dans le MATLAB (Version sous Windows), on peut générer un bruit blanc gaussien

- de variance égale à l'unité dans chaque dimension (car on a deux dimensions)
- de moyenne nulle.

Pour avoir un bruit de variance σ^2 dans chaque dimension , on multiplions les échantillons générés du bruit par σ^2.

Soient p la probabilité et $\hat{p}$ la valeur estimée par la simulation . Dans la méthode de Monté-Carlo , Si on désire connaître la précision relative r pour un nombre d'essais N, et une confiance 1- c , on applique la formule [4]

$$r^2 \geq 2(erfc^{-1}(c))^2 \frac{1}{N}(\frac{1}{p} - 1) \qquad (3.1)$$

L'inconvénient majeur de cette méthode réside dans le fait que le nombre d'essais N, croît rapidement lorsque les probabilités d'erreur sont faibles . Malheureusement, on a pas pu calculer les précisions r dans notre simulation, à cause du facteur de temps !

3.2.4. Calcul de la probabilité P_e (e) et le nombre des bits erronés

La séquence d'information pseudo-aléatoire générée par la source est de longueur $2^{16} - 1 = 65535$ bits, mais dans notre simulation elle est limitée à 60.000 bits .

On calcule la séquence correcte correspondante à l'information pseudo- aléatoire, et son trajectoire dans le treillis, puis on calcule la séquence incorrecte qui est choisie par le décodeur à maximum de vraisemblance ainsi que sa trajectoire. La trajectoire d'une séquence est complètement définie par les états décrits par la séquence minimum elle même.

En tirant à partir de ces deux séquence et de leurs trajectoires, les événements d'erreur, puis on calcul les probabilités d'erreur correspondantes à ces événements à l 'aide de la formule(2.22).

La probabilité d'événement d'erreur est donnée par

$$P_e(e) = \frac{1}{fr / N(d_{free})} \sum_{i=1}^{nev} P_i \qquad (3.2)$$

où

* *nev* est le nombre des événements d'erreur sur toute la longueur du treillis ;

* *fr* est la fréquence des événements ayant une distance euclidienne égale à d_{free}. Si on suppose que ces événement sont équiprobables, alors chaque événement parmi ces dernières a une fréquence égale à $fr / N(d_{free})$;

* P_i est la i ème probabilité d'événement d'erreur, qui est calculée à l'aide de la relation (2.22). $N(d_{free})$ peut être calculer à la main si le codeur est simple , et avec l'ordinateur dans le cas contraire.

Si tous événements sont équiprobables, donc ils ont la même fréquence $fr/N(d_{free})$, la formule (3.2) devient dans ce cas un résultat direct du principe d'union.

Lorsque la longueur du treillis (ou la longueur de la séquence d'information) tend vers l'infini, $P_e(e)$ tend vers sa valeur exacte .

Si on peut calculer le nombre des bits erronés b_i dans l'événement d'erreur numéro i (i=1, ... , nev) la probabilité d'erreur par bit est donnée par

$$P_b(e) = \frac{1}{m \times fr / N(d_{free})} \sum_{i=1}^{nev} b_i P_i \qquad (3.3)$$

où m est le nombre des entrées du codeur .

Généralement le calcul de $P_b(e)$ est difficile, donc il faut chercher un autre paramètre facile à calculer et qui peut remplacer $P_b(e)$ même d'une manière asymptotique .

La profondeur de décision dp (depth) est pratiquement choisi supérieur ou égal à 6 fois le nombre des éléments mémoires du codeur. Dans notre simulation nous avons pris dp=2000, pour que les resultats soient les meilleurs possibles.

Nous rappelons pour un codeur en treillis de m entrées , qu'on a 2^m branches partent de chaque état. Dans notre programmation, on a donné à chaque branche un numéro, tel que la numérotation se fait de haut en bas , et de 0 jusqu'à 2^m - 1. La conversion décimal-binaire de ce numéro , qui est écrite en m digits , donne l'entrée binaire du codeur , tel que le bit le plus fort correspond à l'entrée la plus haute. Par exemple pour m = 2 , si on ce trouve dans la branche numéro 2 , l'entrée du codeur est automatiquement $x_1 = 1$ et $x_2 = 0$.

A partir de la trajectoire de la séquence incorrecte , et avec cette idée , on peut compter les bits erronés, donc le pourcentage de ces bits , qui est le nombre de bits erronés sur le nombre total des bits. Ce pourcentage n'est autre que la probabilité des bits erronés, conditionnée par l'apparition de séquence incorrecte, donc on peut calculer la probabilité Pb

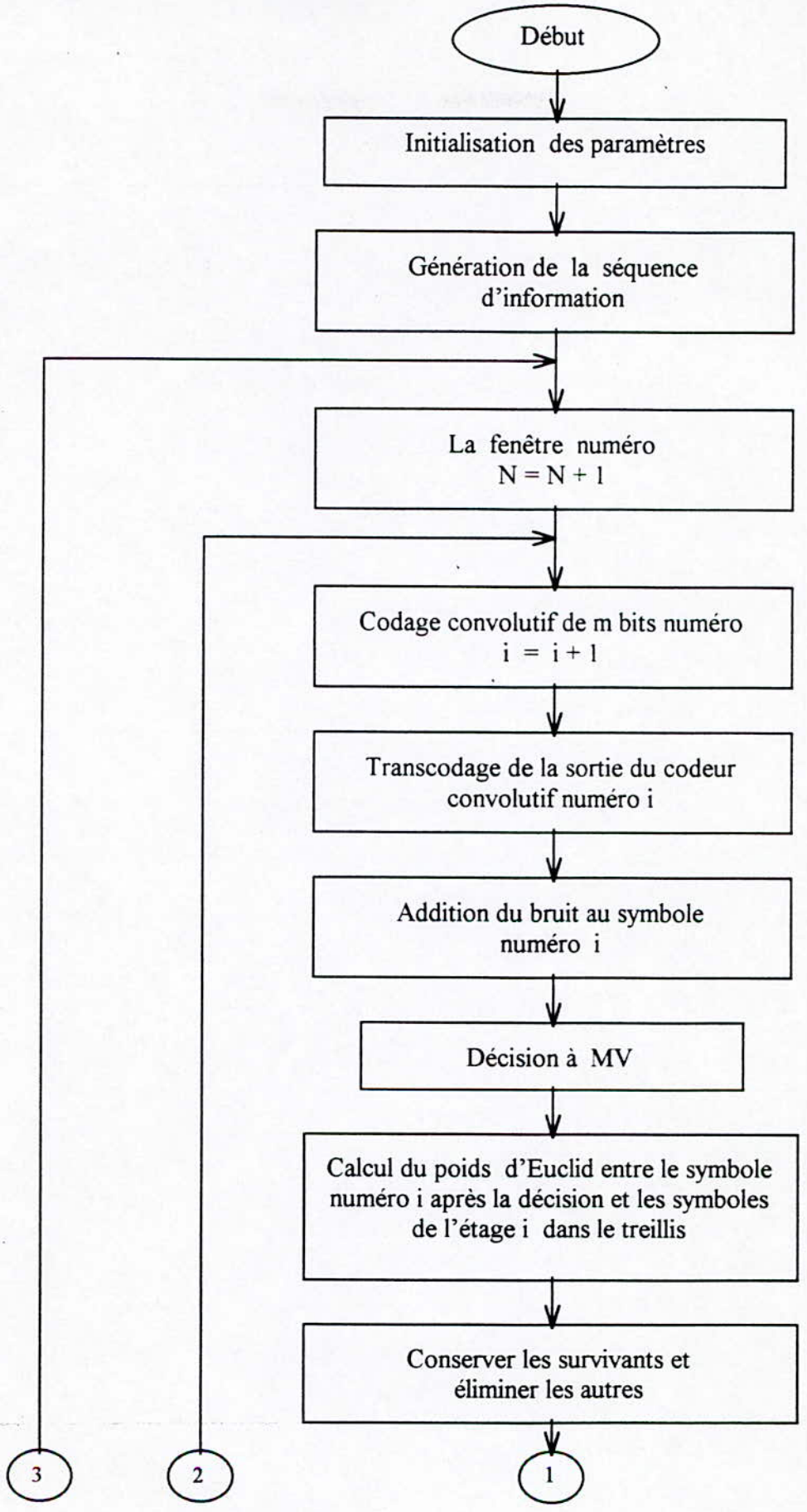
$$Pb = P_e(e) \times \text{le pourcentage de bits erronés} \qquad (3.4)$$

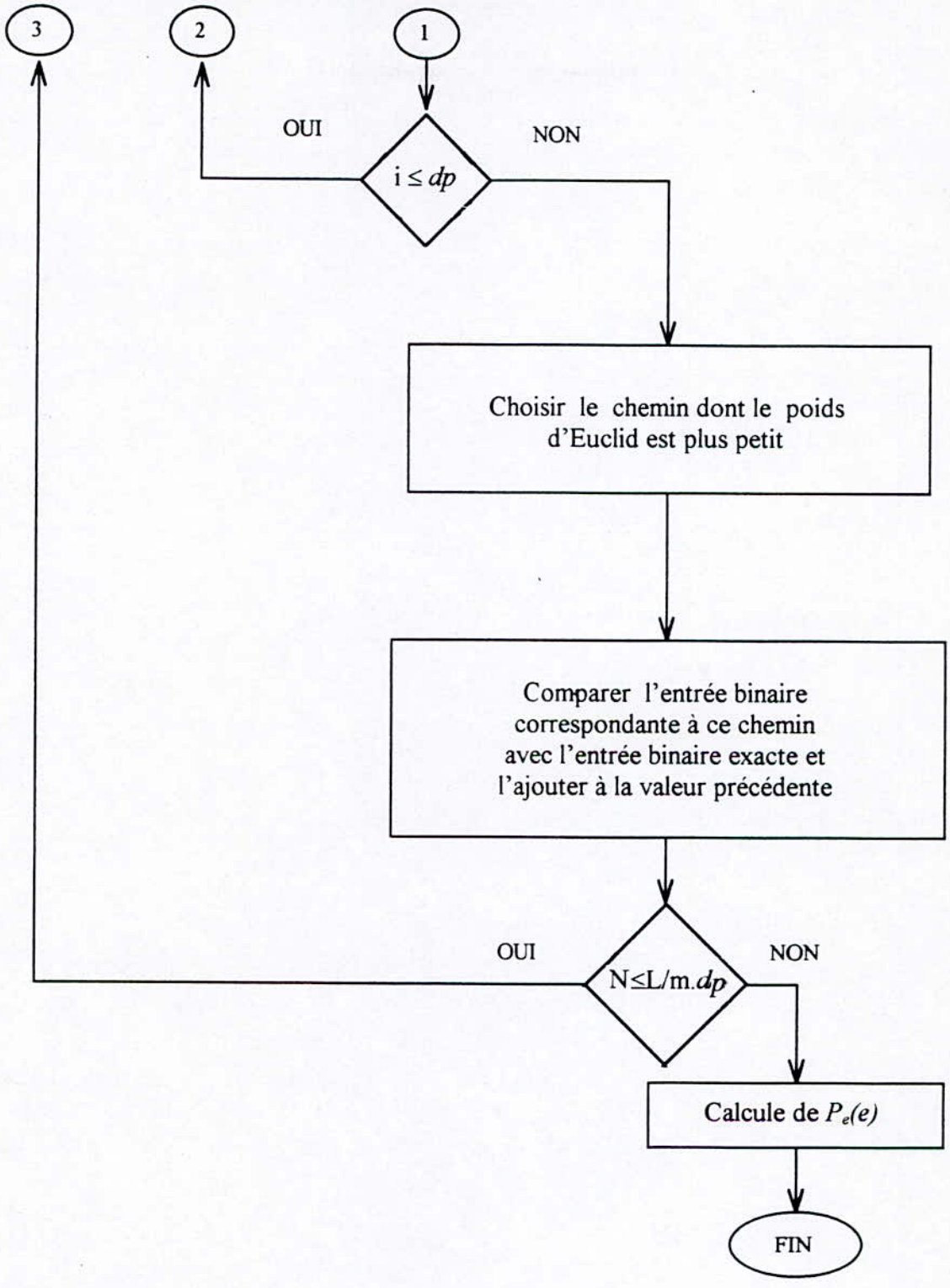
qui est donc la probabilité des bits erronés sans conditionnement . Nous allons utiliser ce paramètre pour la comparaison entre les codeurs choisis.

A l'aide des deux relations (3.3) et (3.4) et avec quelques hypothèses statistiques , on peut montrer que pour un RSB grand , on a

$$Pb \approx \frac{m}{L} \times P_b(e) \qquad (3.5)$$

où L est la longueur de séquence d'information. L'organigramme de programmation , pour un codeur en treillis donné, est illustré à la figure 3.4 .


Début
Initialisation des paramètres
Génération de la séquence d'information
La fenêtre numéro N = N + 1
Codage convolutif de m bits numéro i = i + 1
Transcodage de la sortie du codeur convolutif numéro i
Addition du bruit au symbole numéro i
Décision à MV
Calcul du poids d'Euclid entre le symbole numéro i après la décision et les symboles de l'étage i dans le treillis
Conserver les survivants et éliminer les autres
3
2
1



L : la longueur de l'information
m : le nombre des entrées du codeur
dp : le depth

Figure 3.4. Organigramme représentant le plan de simulation.

3.3 Résultats et Commentaires

3.3.1 Programmation

Pour la mise en œuvre de notre simulation, nous avons opter pour la réalisation des progammes par le Matlab sous Windows pour les raisons suivantes

- le Matlab contient des fonctions prêtes à utiliser, telles que la génération du bruit, la fonction d'erreur, fonction d'autocorrection, etc. Ces fonctions permettent une réduction des programmes.
- Un gain en espace mémoire du micro-ordinateur, car on n'a pu générer et exécuter avec Matlab des séquences d'information de taille de 2^{17} bits, par contre, la taille atteinte dans une programmation en Pascal était de 2047 (voir [6]).
- Le Matlab contient un graphisme qui permet une visualisation et un traçage simples des courbes.

Nous avons réalisé donc, à l'aide du Matlab des programmes de simulation pour des systèmes cités dans le tableau 3.1.

ORDRE	SYSTEME		PROGRAMME REALISE
	MODULATION	CODEUR	
a	4-PSK	FIGURE 3.3.a	TCM1
b	8-PSK	FIGURE 3.3.b	TCM2
c	8-PSK	FIGURE 3.3.c	TCM3
d	8-PSK	FIGURE 3.3.d	TCM4
e	16-PSK	FIGURE 3.3.e	TCM5
f	8-AM-PM	FIGURE 3.3.b	TCM6
g	16-QAM	FIGURE 3.3.e	TCM7

Tableau 3.1

Le listing de ces programmes sont présentés dans l'annexe B. Nous signalons que la partie décodage par l'utilisation de l'algorithme de Viterbi était la partie la plus difficile à mettre en œuvre du fait de sa complexité.

Pour illustrer la simulation, nous donnons à titre d'exemple quelques résultats de la simulation, TCM2, à la sortie de quelques blocs de la chaîne, pour une séquence binaire générée par la source

x = 0 0 0 1 0 0 1 1 0 1 0 1 1 1 1 0 0 0 1 0 0 1 1 0

Ces résultats sont présentés dans le tableau 3.2.

Sorties du codeur convolutif y1 y2 y3	0 0 0 1 0 0 1 1 0 1 0 1 0 1 0 0 1 0 0 1 1 0 1 0 0 0 1 0 1 1 1 1 0 0 0 1
La sortie du transcodeur	1 3 2 5 4 2 6 8 3 5 3 6
La séquence détectée pour un RSB=7 dB	1 2 3 5 4 2 6 8 4 6 2 7
La séquence détectée pour un RSB=12.6 dB	1 3 2 5 4 2 6 8 3 5 3 6

Tableau 3.2.

3.3.2 Probabilité d'événement d'erreur $P_e(e)$

La probabilité d'événement d'erreur $P_e(e)$ mesure la performance des systèmes TCM. Notre simulation a permis de calculer ce paramètre pour l'ensemble des systèmes étudiés et pour différentes valeurs de bruit. La figure 3.5.a. montre la probabilité d'événement d'erreur pour deux systèmes en fonction du rapport signal sur bruit.

- Pour le système 4-PSK sans codage sa probabilité $P_e(e)$ est bien approximée par des expressions (3.2) et représentée par la courbe supérieure.
- Pour le système 8-PSK codé en treillis ses équations expriment une limite inférieure qui est asymptotiquement atteinte pour des valeurs de RSB grandes. Les résultats de la simulation pour une distance supérieure à la distance limite sont inclus dans la figure, et représentés par des points.

L'analyse des figures 3.5.a. jusqu'à 3.11.a, permet de faire les constatations et les remarques suivantes

1) La courbe obtenue par simulation est située au-dessous de la courbe de probabilité d'événement d'erreur du système sans codage, et pour un RSB grand cette courbe converge vers sa courbe asymptotique. Ceci confirme bien qu'un système avec codage en treillis est plus performant que celui sans codage.

2) La probabilité $P_e(e)$ est inversement proportionnelle au RSB, ce qui confirme davantage la théorie, puisque pour un RSB grand la séquence émise n'est affectée que peut par le bruit. Donc, le codeur de vraisemblance maximale aura à son entrée une séquence avec un nombre faible de bits erronés (nombre des événements d'erreur faible), d'où une probabilité faible, et vice versa.

3) La comparaison des résultats obtenus par notre simulation et pour le même codeur (4 états, 8-PSK), figure 3.6.a, et les résultats donnés par Ungerboeck [18], figure 3.6.c, montre qu'ils sont très

proches, et se situent dans le même intervalle toléré par la théorie (formule (3.1)). Le tableau 3.3 donne trois valeurs comparées. Ceci donne un poids à notre travail.

	$P_e(e)$	
RSB	Resultats de simulation	Resultats de Ungerboeck [18]
7.5 dB	5.10^{-2}	7.10^{-2}
8 dB	9.10^{-2}	10^{-3}
9 dB	9.10^{-3}	2.10^{-4}

Tableau 3.3.

3.3.3 Performance en fonctions du nombre d'états et modulations

- Pour la même modulation 8-PSK, les codeurs en treillis à 16 états, figure 3.12, sont plus performants que les codeurs à 8 états. Ces derniers sont à leur tour plus performants que ceux de 4 états. On peut conclure, que lorsque le nombre d'états du codeur en treillis augmente, la performance devient meilleure.
- On remarque de la figure 3.14, que les codeurs de mêmes états 16, ceux avec une modulation 16-QAM sont plus performants que ceux avec modulation 16-PSK. En effet, lorsque le nombre de points augmente pour la modulation PSK la distance entre les points de la constellation sur le cercle unité diminue, ce qui influe négativement sur la distance minimale du codeur, donc diminution de la performance. Par contre, dans la modulation QAM, l'augmentation du nombre de points de la constellation dans le plan de l'espace des signaux n'influe pas sur la distance minimale du codeur.
- La figure3.13 montre en revanche qu'un codeur en treillis à 8 états et 8-PSK est plus performant qu'un codeur 8 états et 8-AM-PM.

Ces deux dernières remarques permettent de conclure que lorsque le nombre de points de la constellation du codeur est grand $M \geq 16$, la modulation QAM donne des résultats meilleurs par rapport à la modulation PSK.

3.3.4 Transmission sans erreurs

Pour chaque type de modulation, il y a une valeur limite avec une certaine marge du RSB dans laquelle il n'y a pas pratiquement des erreurs de transmission, quelque soit le codeur en treillis utilisé.

Cette valeur dépend uniquement de la forme, le nombre des points de la constellation et le type de bruit. Par exemple pour le 8-PSK, cette valeur égale à 12.6 dB, figure 3.15.b, au-dessous de cette valeur on remarque qu'il y a vraiment un chevauchement entre les points voisins, figure 3.15.a , et même entre tous les points si la valeur du RSB est faible. Au-dessus de la valeur limite, il n'y a pas d'erreurs de transmission , figure 3.15.c .

3.3.5 Probabilité de bits erronés *Pb*

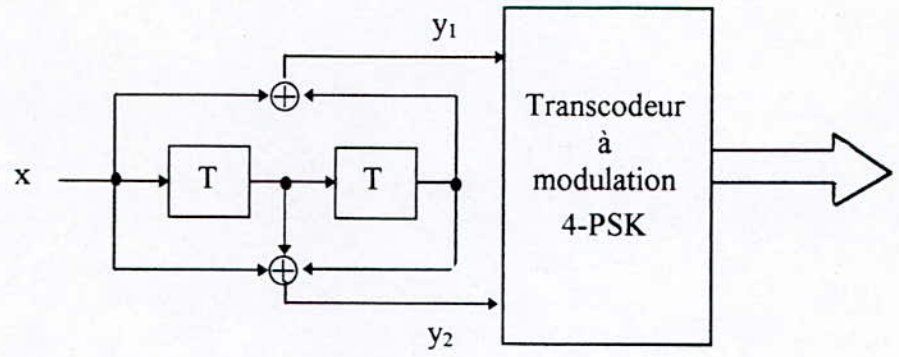
La probabilité de bits erronés *Pb* permet la comparaison entre les codeurs de même probabilité d'événement d'erreur, dans le but de choisir un codeur optimal de structure facile à implémenter, car la probabilité d'événement d'erreur ne dépend pas de la structure des codeurs convolutifs.[3]

Notre simulation a permis de calculer cette probabilité en fonction du bruit, les figures 3.5.b jusqu'à 3.11.b, confirment les mêmes conclusions concernant la probabilité d'événement d'erreur 3.3.3.

Performance fixée: $Pb=10^{-5}$	RSB	
débit	PSK	QAM
1 bit/*T*	3.5 dB	3.5 dB
2 bit/*T*	8 dB	8.5 dB
3 bit/*T*	13.5 dB	11.5 dB

Tableau 3.4.

La probabilité de bits erronés *Pb*, nous renseigne pour un débit bien défini sur la modulation à utiliser pour obtenir un gain en puissance comme l'indique le tableau 3.4 et permet ainsi de trouver un compromis entre le débit et la puissance.



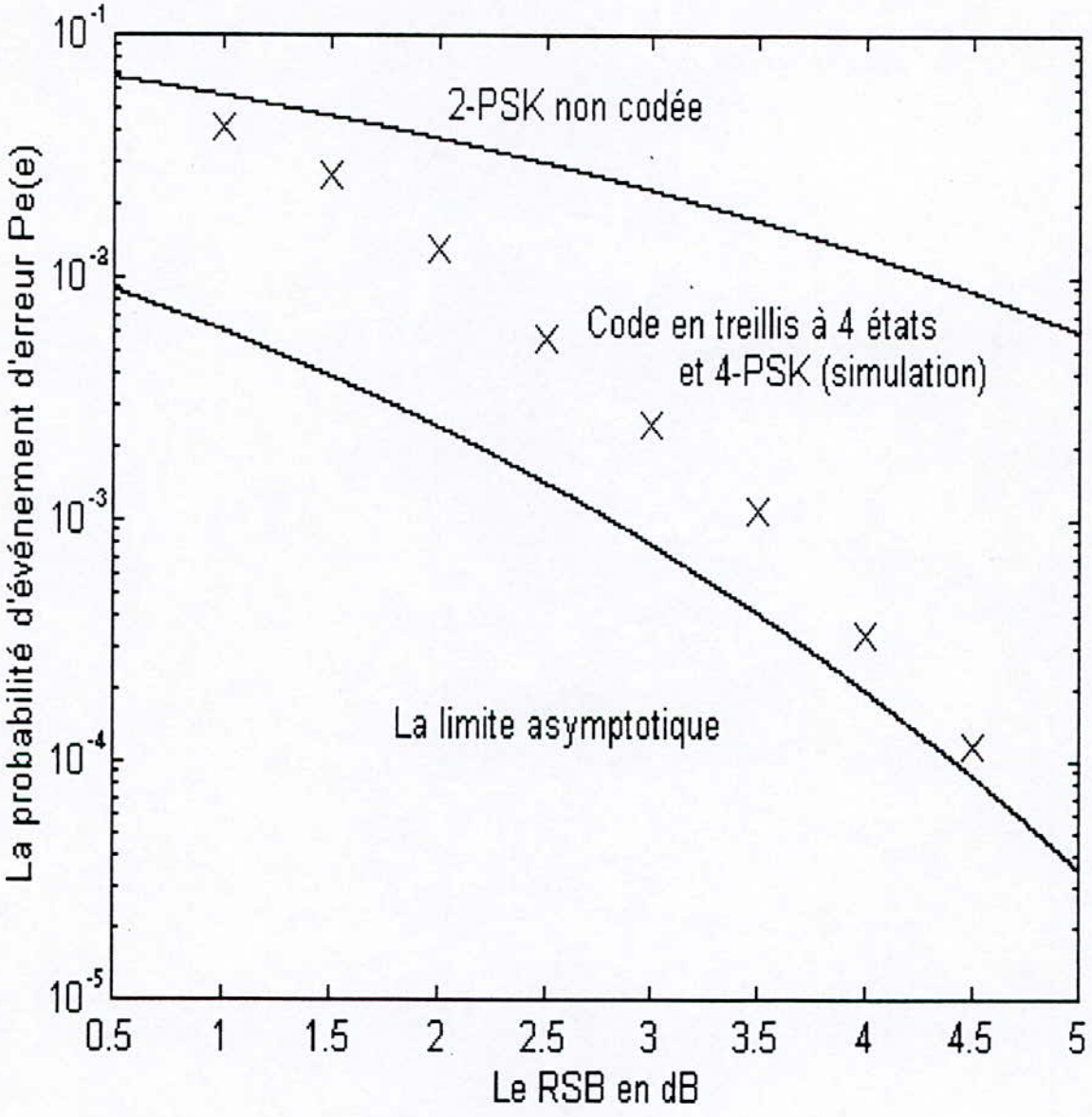
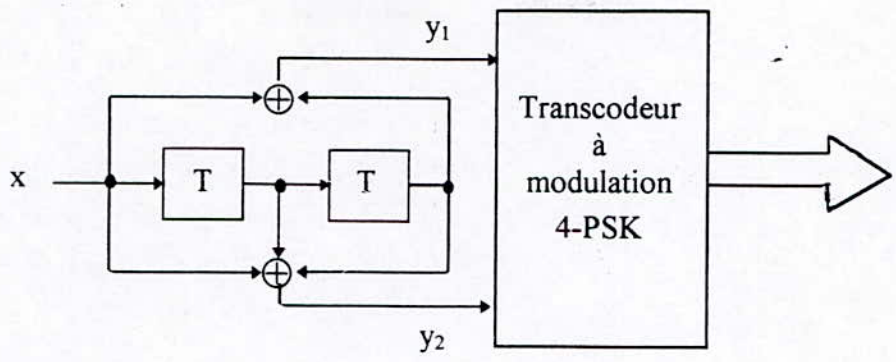


Figure 3.5.a La probabilité d'événement d'erreur d'un code en treillis à 4 états et 4-PSK en fonction du RSB.



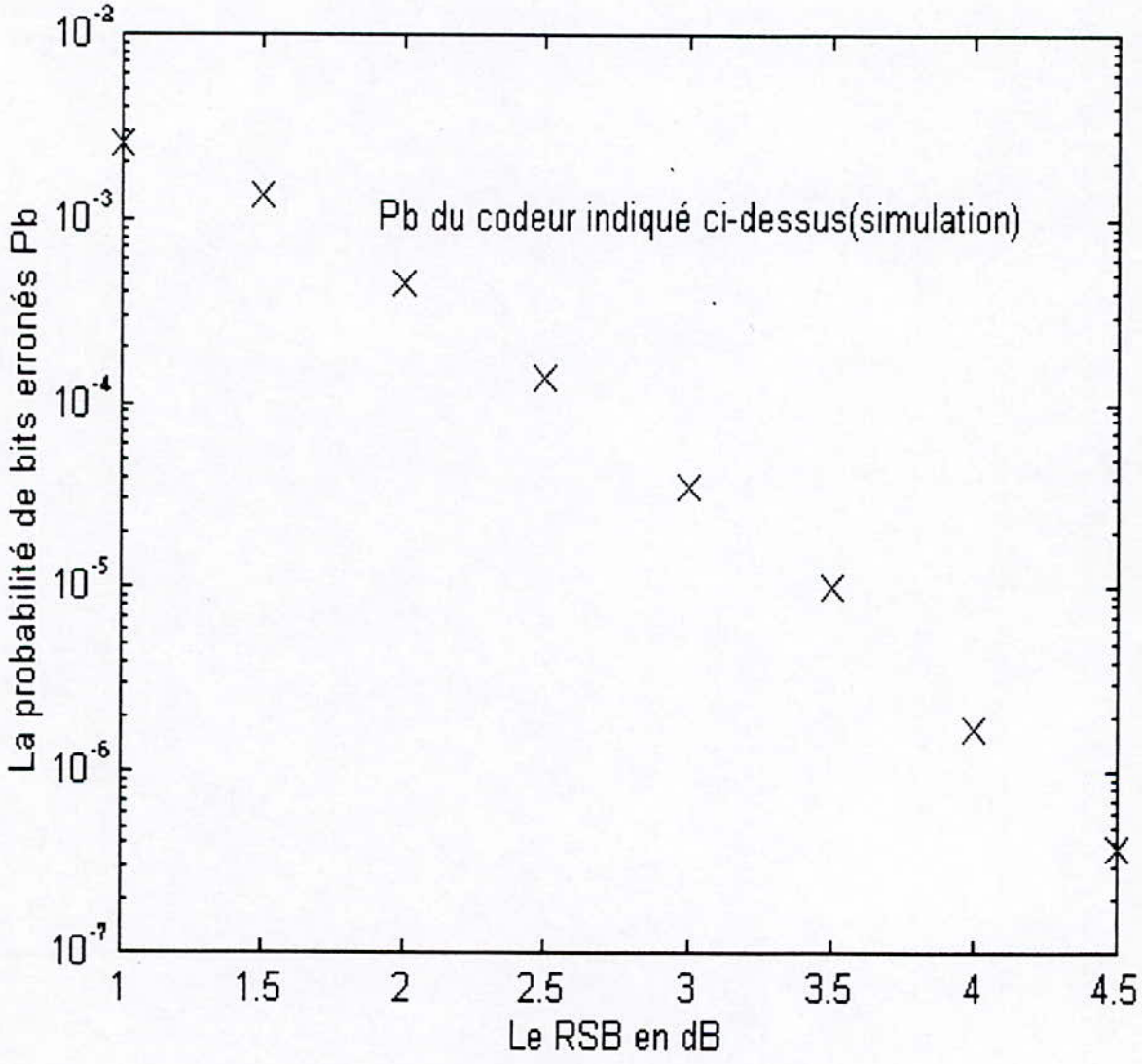
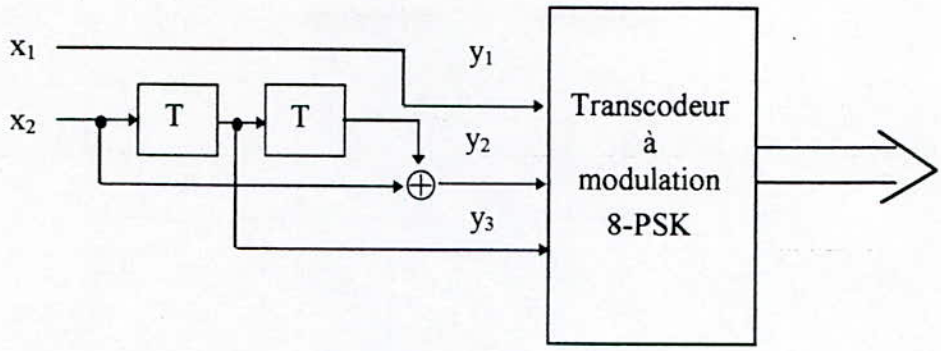


Figure 3.5.b La probabilité de bits erronés du codeur indiqué ci-dessus en fonction du RSB.



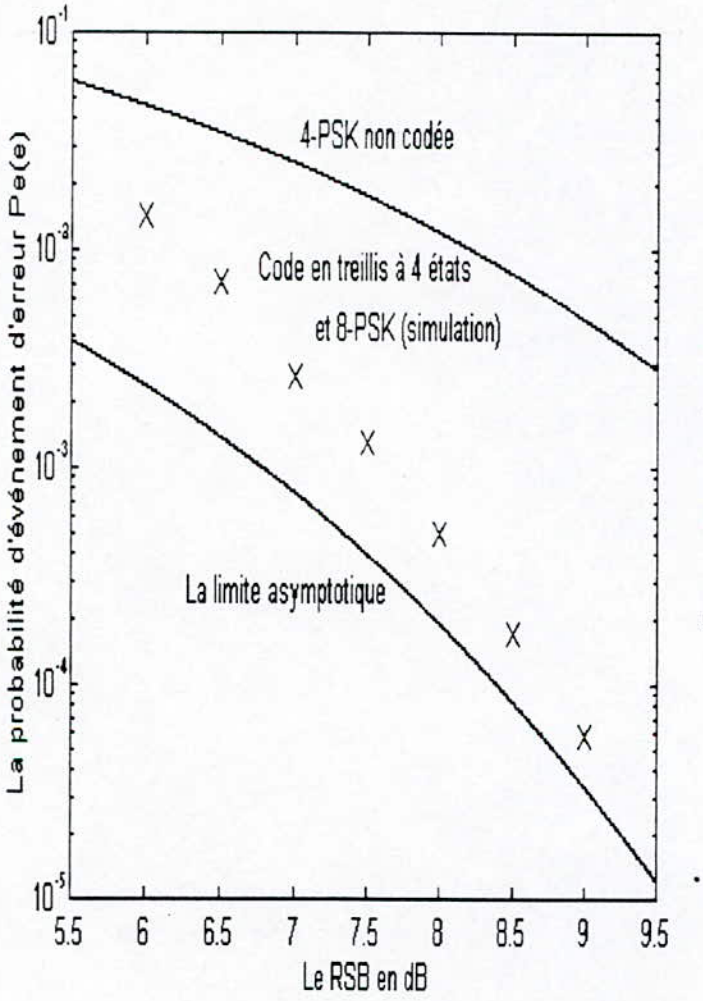


Figure 3.6.a La probabilité d'événement d'erreur d'un code en treillis à 4 états et 8-PSK en fonction du RSB.

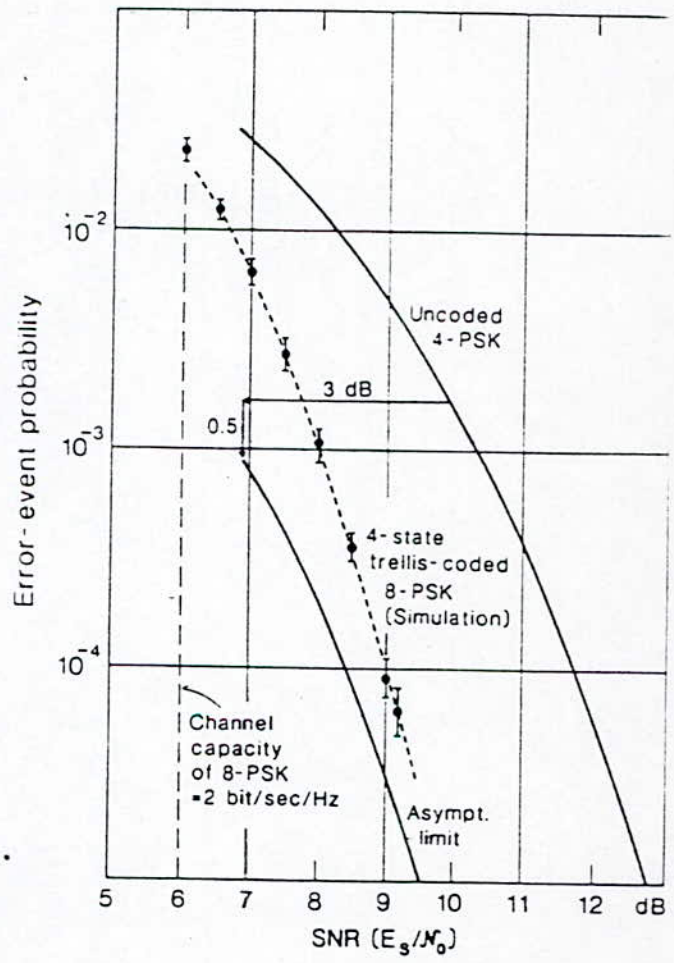
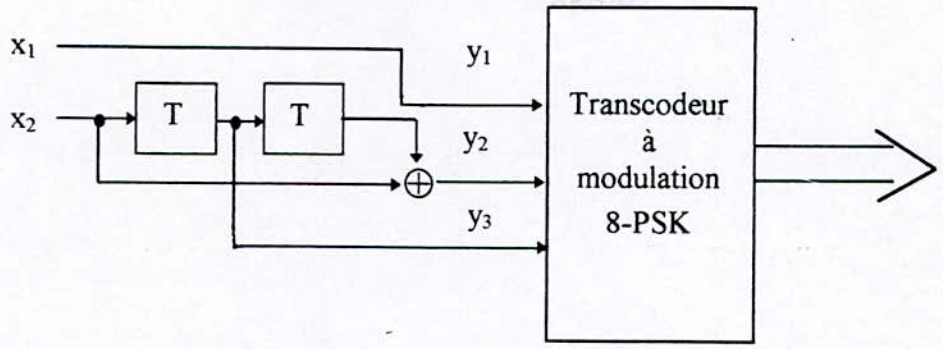


Figure 3.6.c La simulation d'Ungerboeck pour un code en treillis à 4 états et 8-PSK.



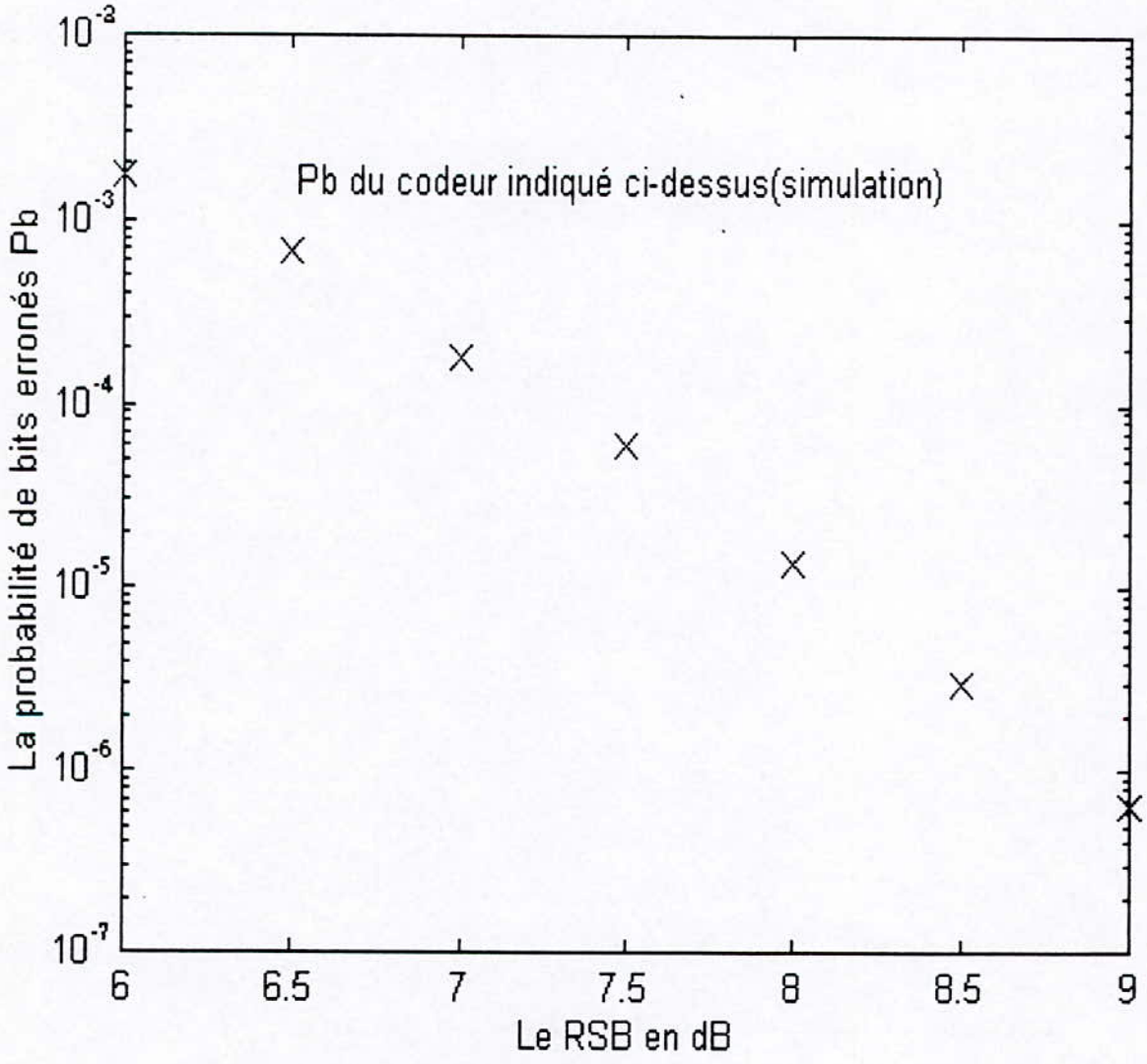
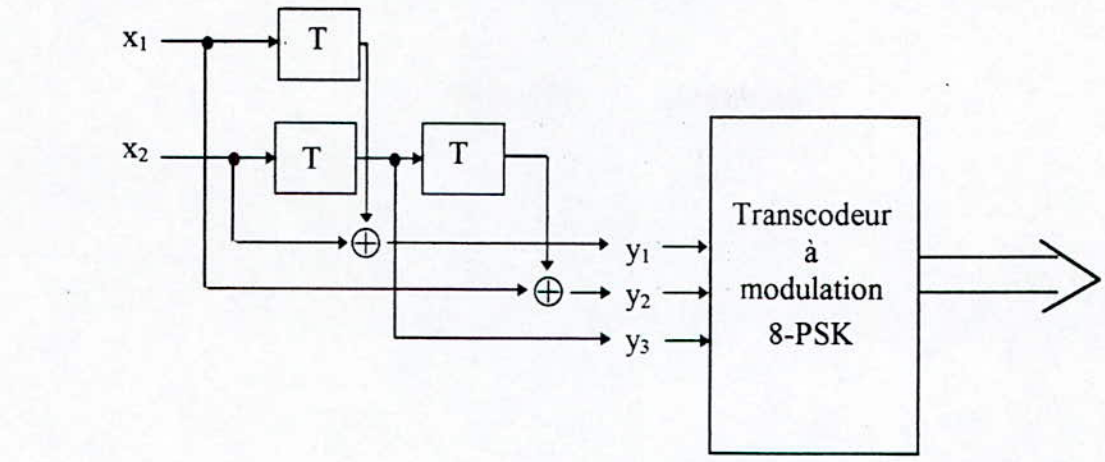


Figure 3.6.b La probabilité de bits erronés du codeur en treillis indiqué ci-dessus en fonction du RSB.



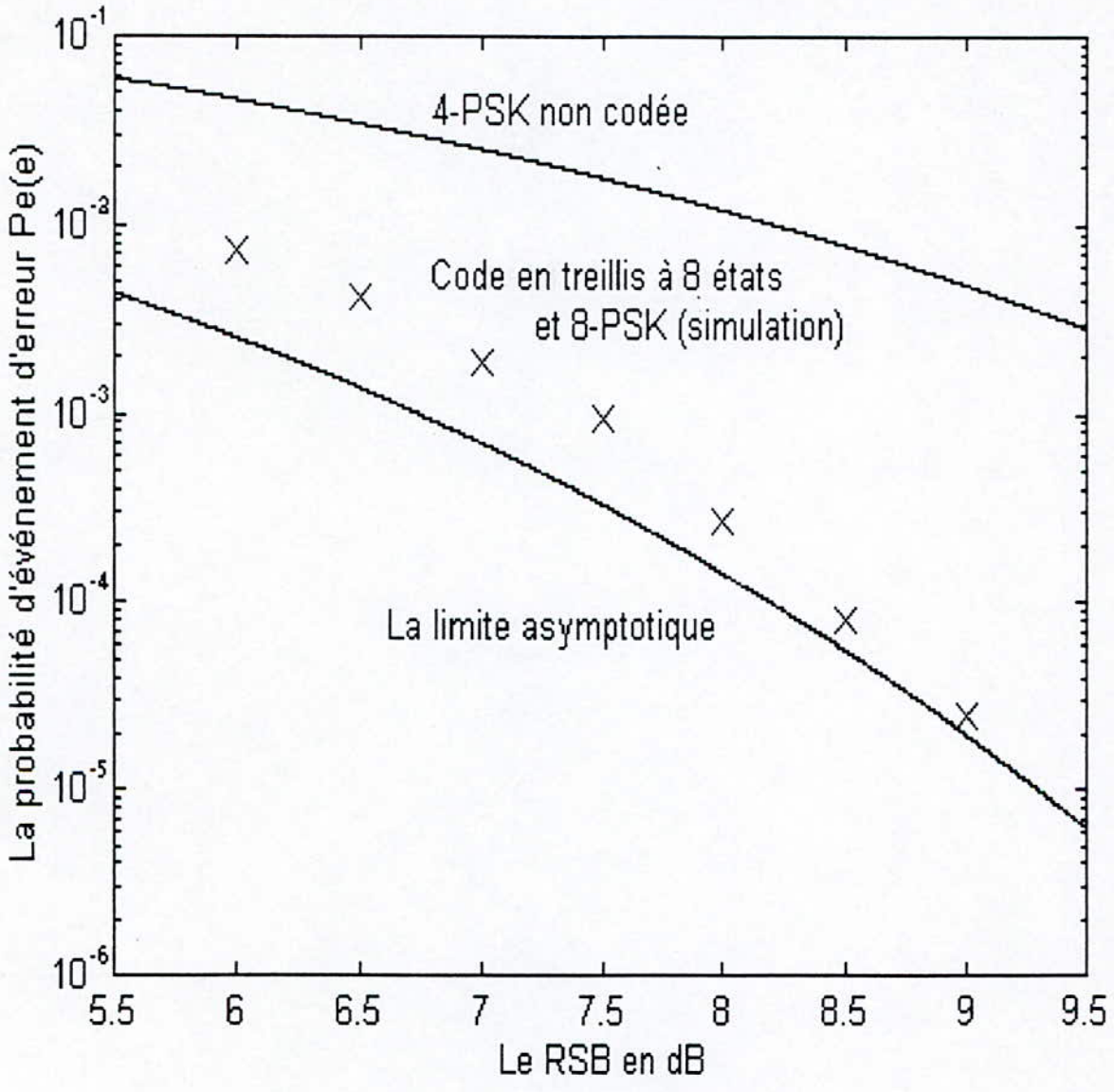
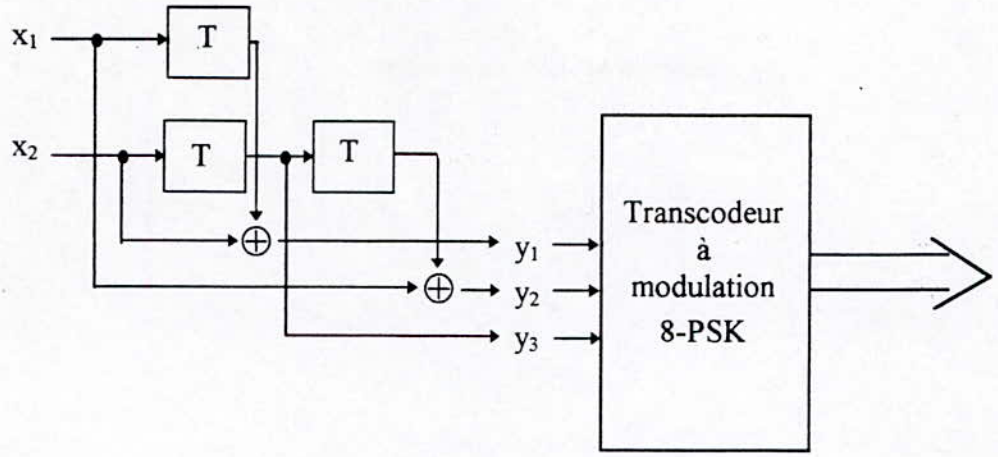


Figure 3.7.a La probabilité d'événement d'erreur d'un code en treillis à 8 états et 8-PSK en fonction du RSB.



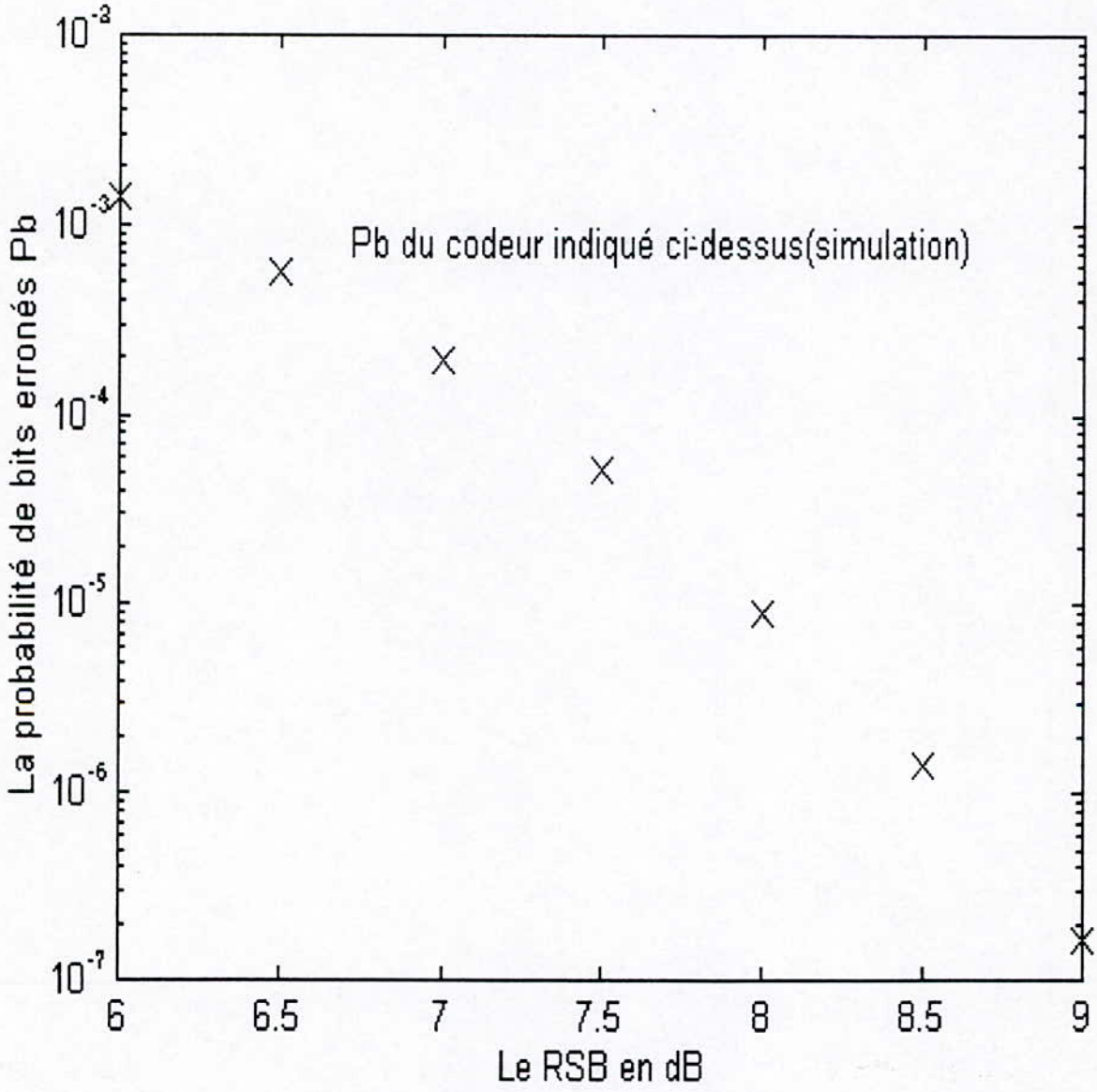
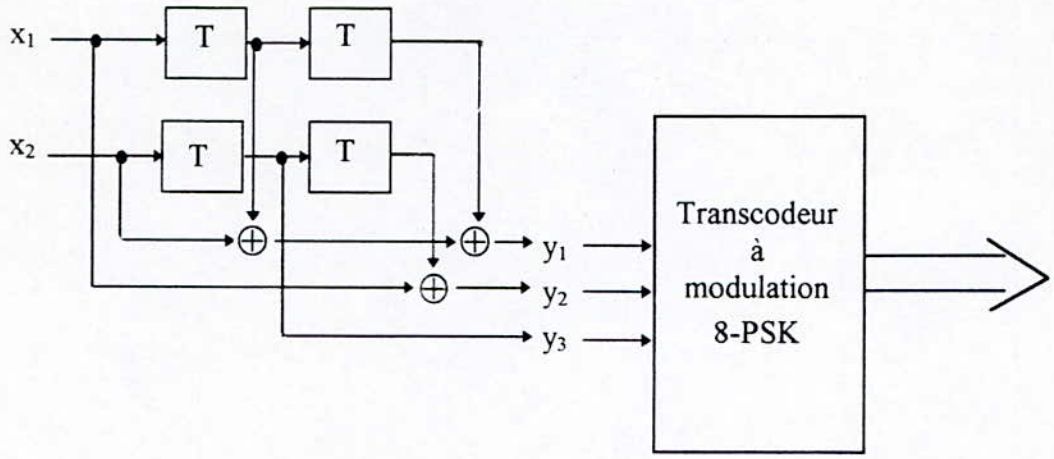


Figure 3.7.b La probabilité de bits erronés du codeur en treillis indiqué ci-dessus en fonction du RSB.



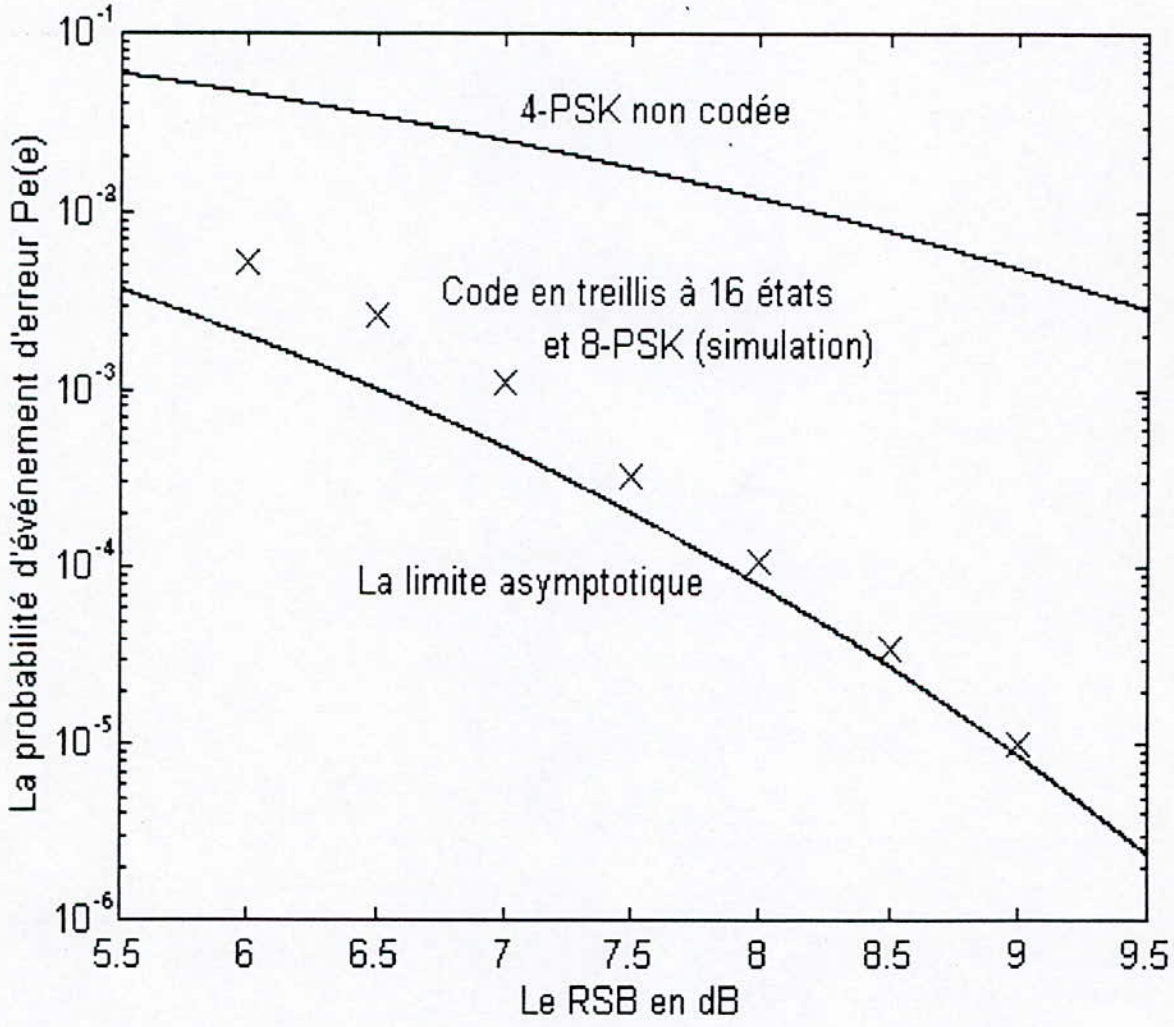
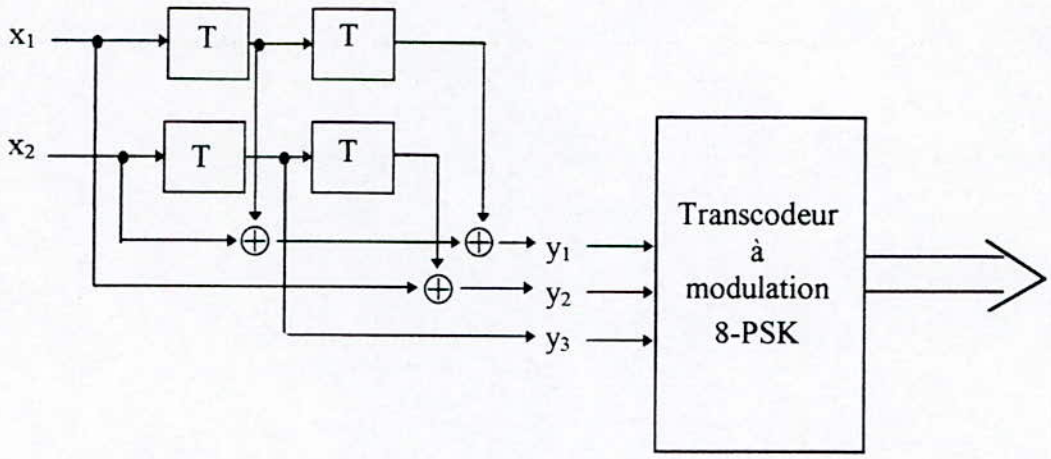


Figure 3.8.a La probabilité d'événement d'erreur d'un code en treillis à 16 états et 8-PSK en fonction du RSB.



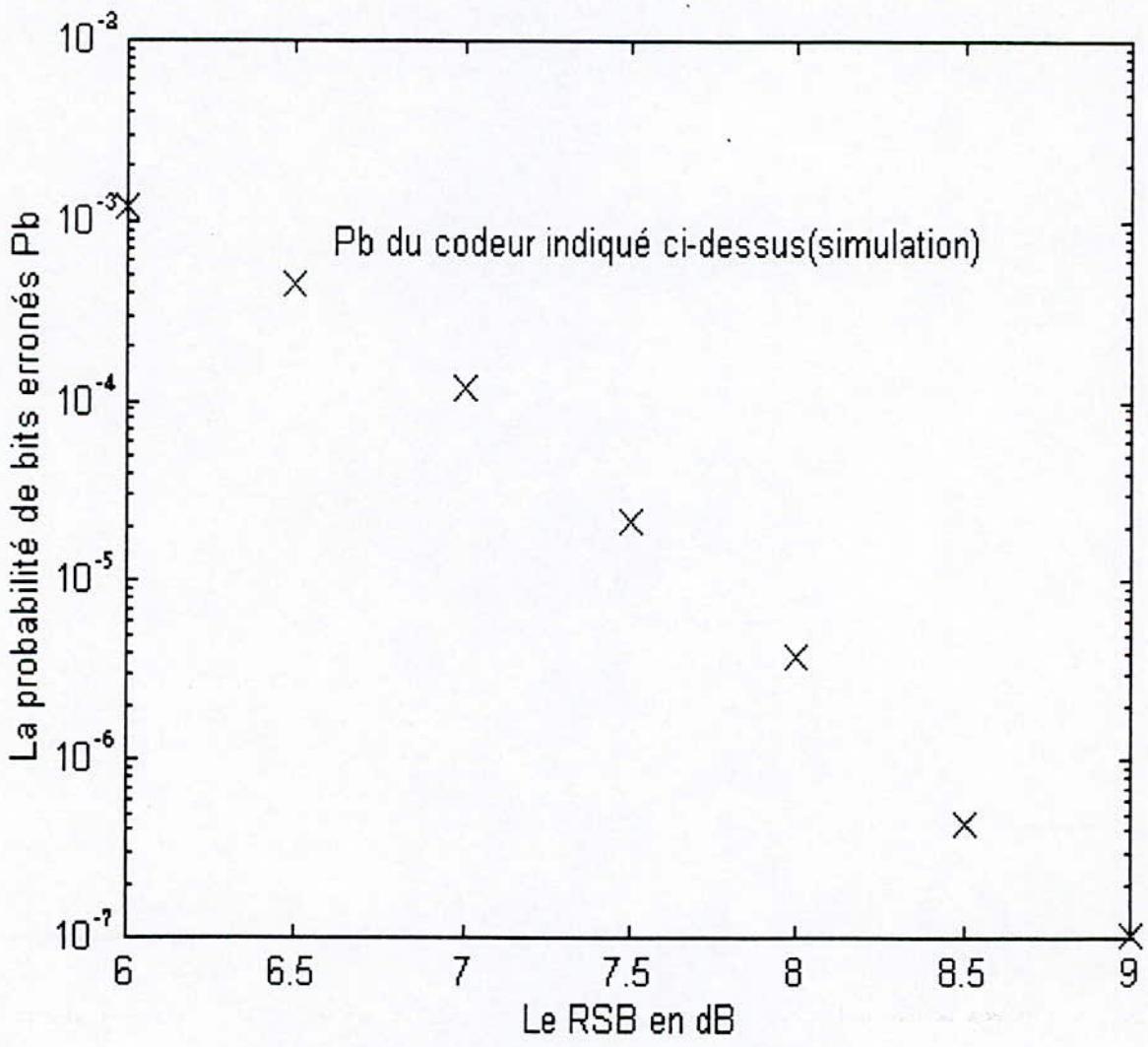


Figure 3.8.b La probabilité de bits erronés du codeur en treillis indiqué ci-dessus en fonction du RSB.

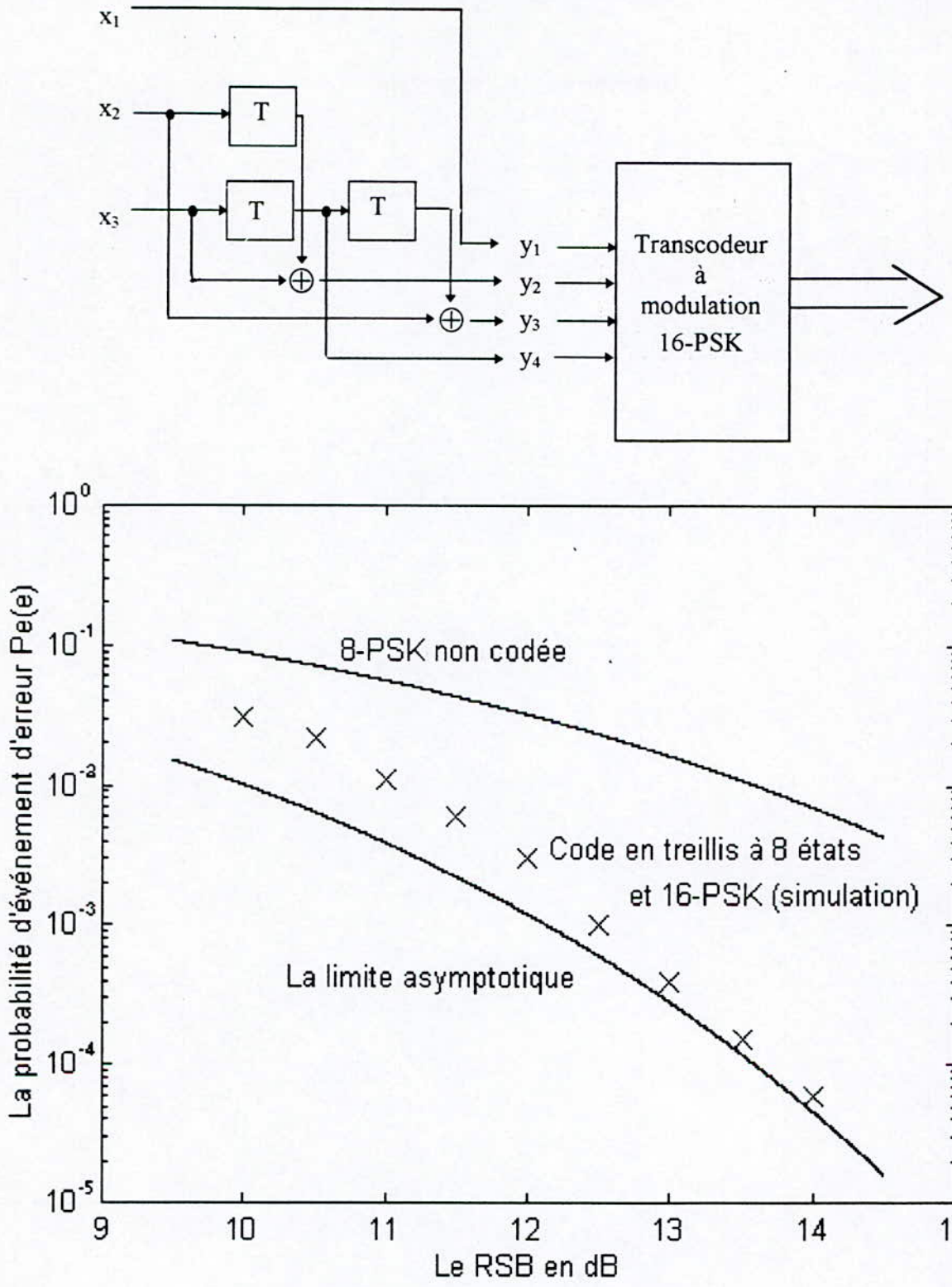
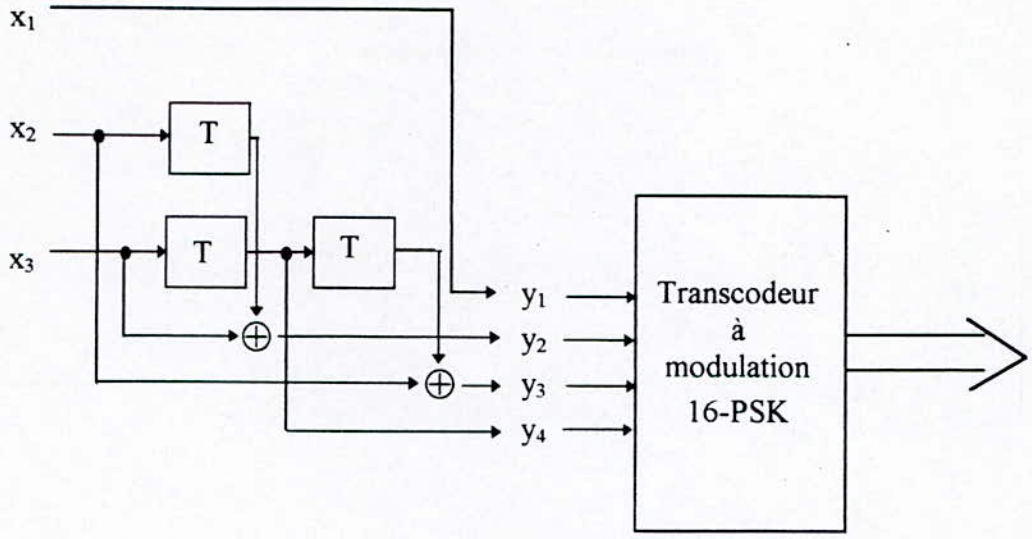


Figure 3.9.a La probabilité d'événement d'erreur d'un code en treillis à 8 états et 16-PSK en fonction du RSB.



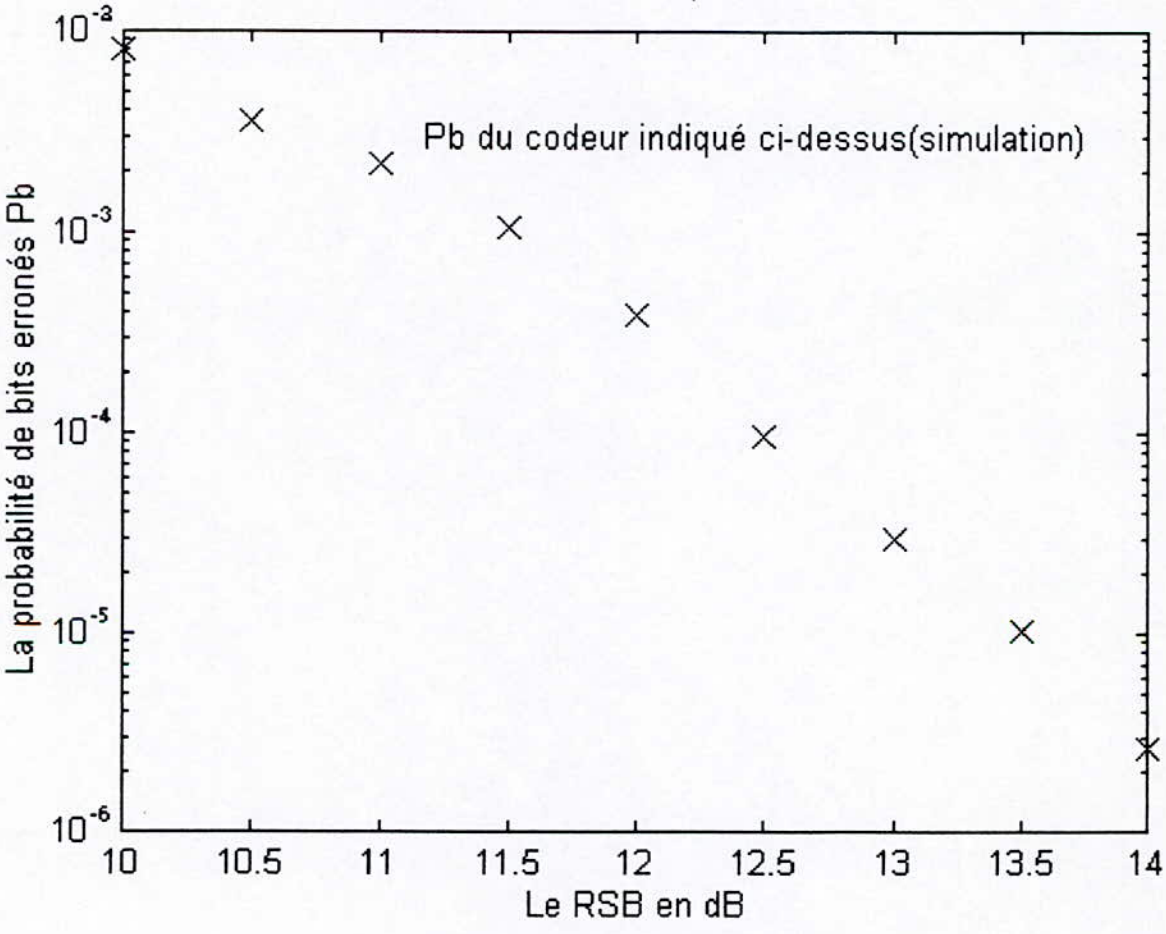


Figure 3.9.b La probabilité de bits erronés du codeur en treillis indiqué ci-dessus en fonction du RSB.

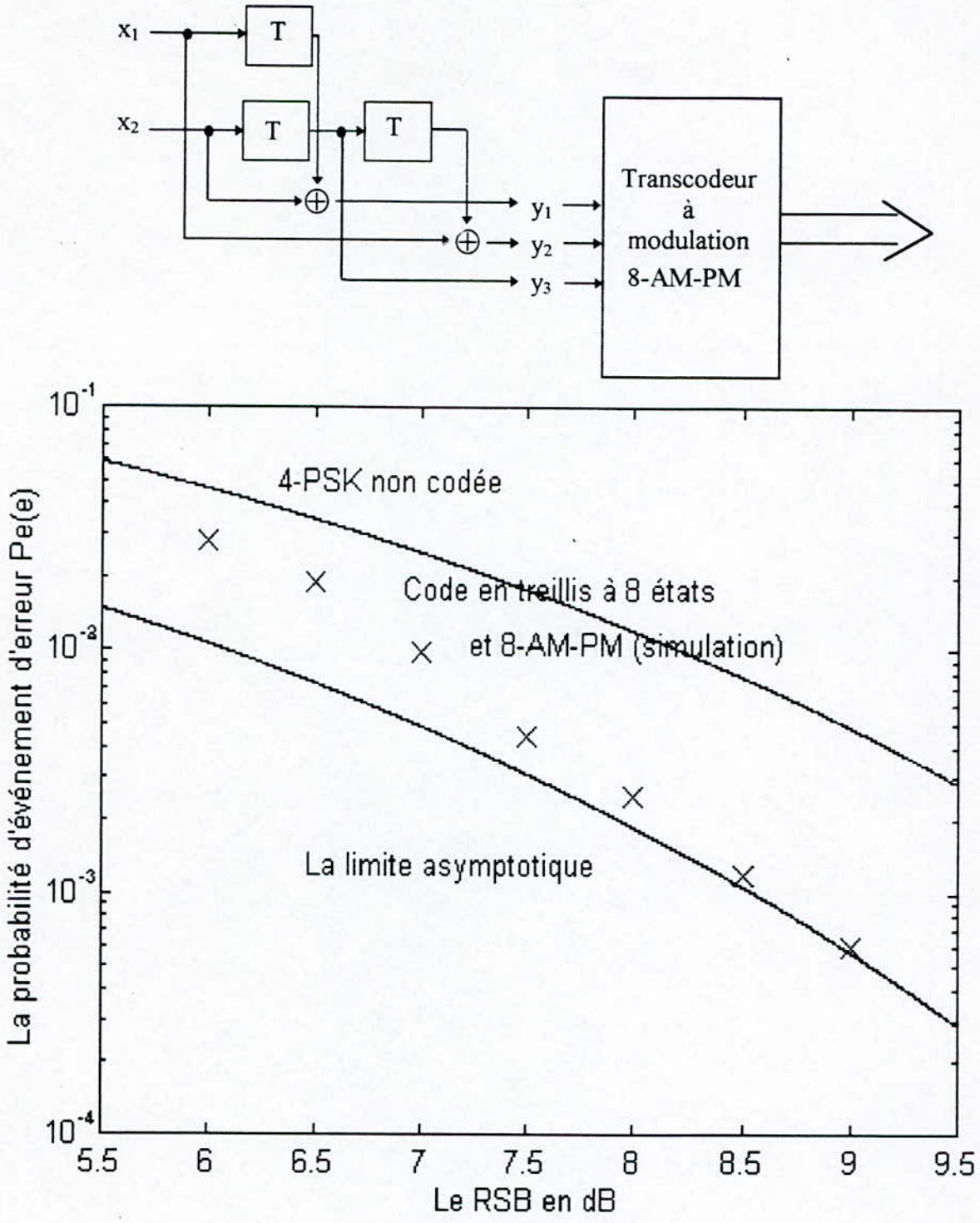
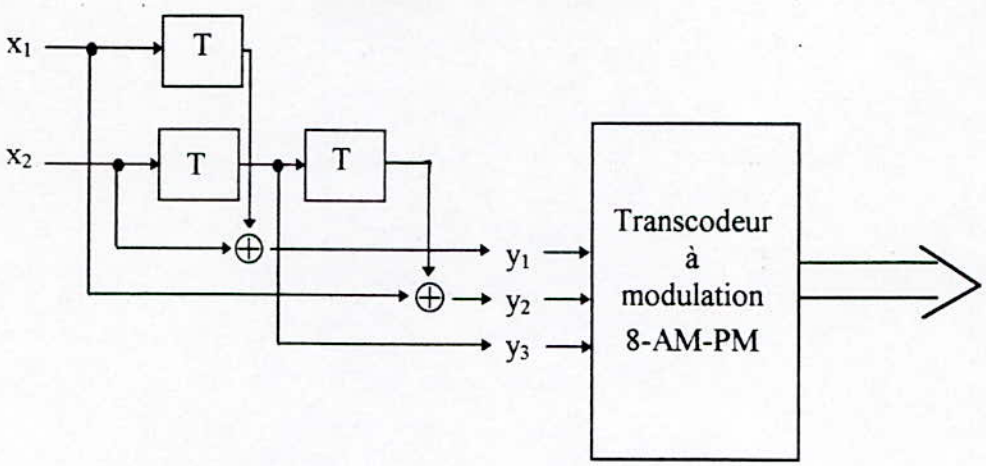


Figure 3.10.a La probabilité d'événement d'erreur d'un code en treillis à 8 états et 8-AM-PM en fonction du RSB.



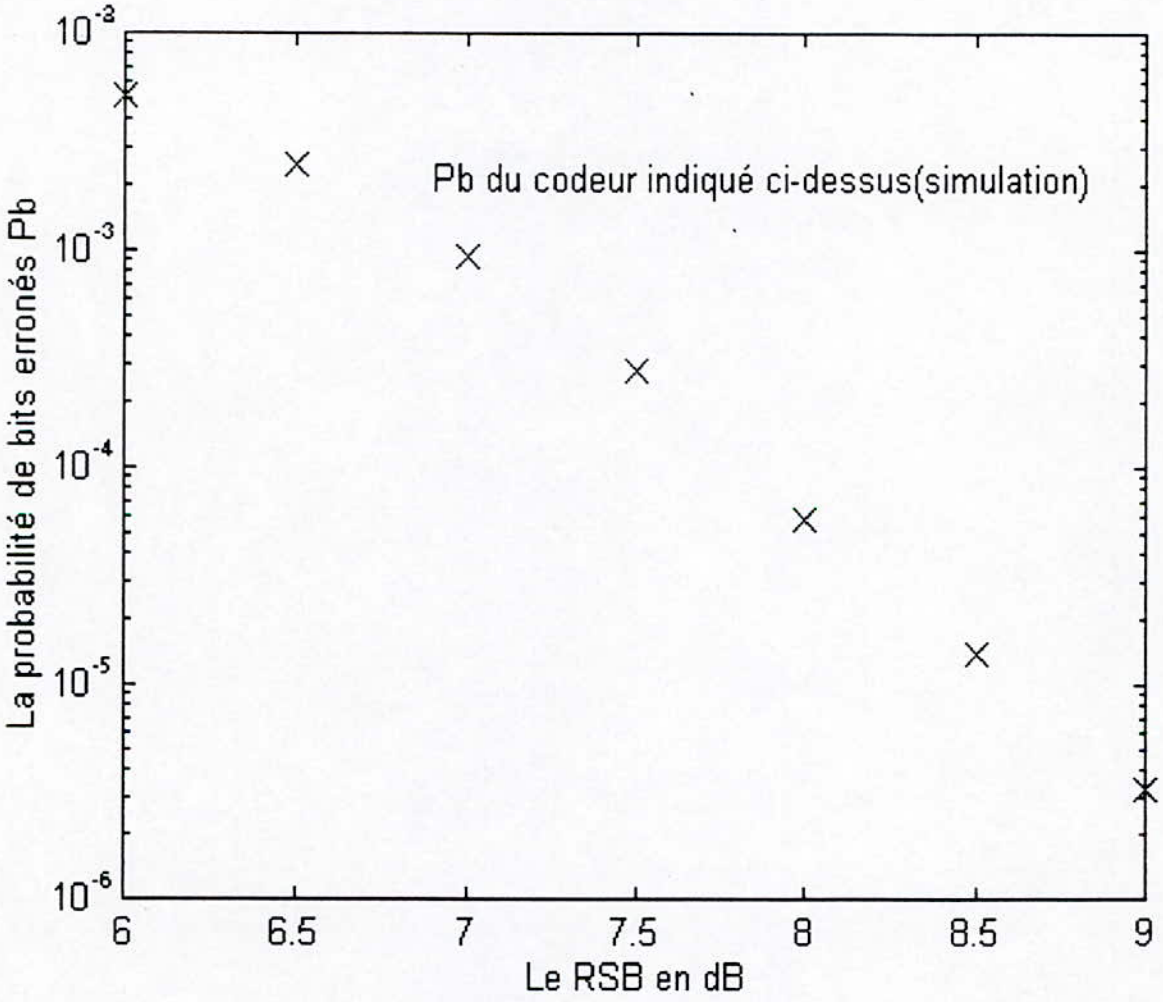
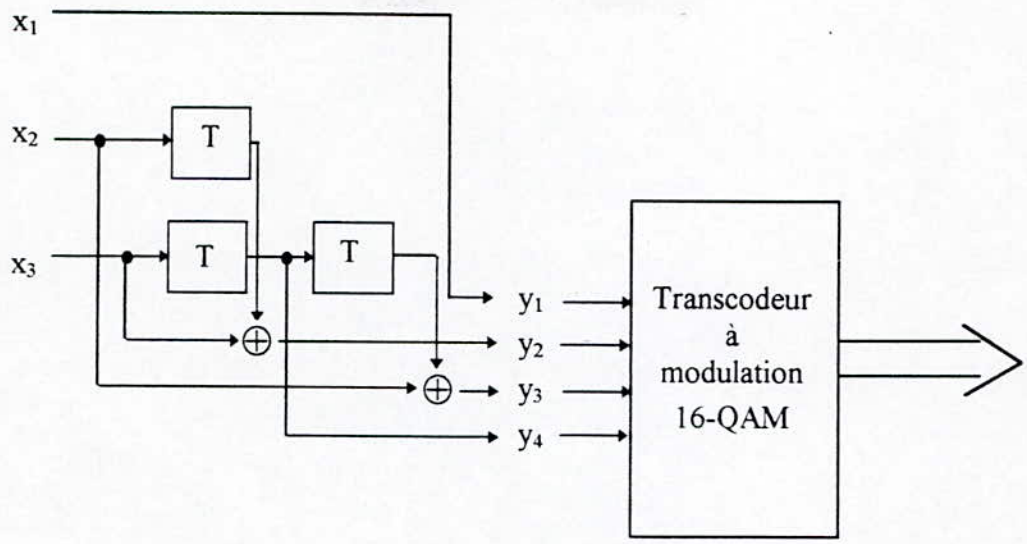


Figure 3.10.b La probabilité de bits erronés du codeur en treillis indiqué ci-dessus en fonction du RSB.



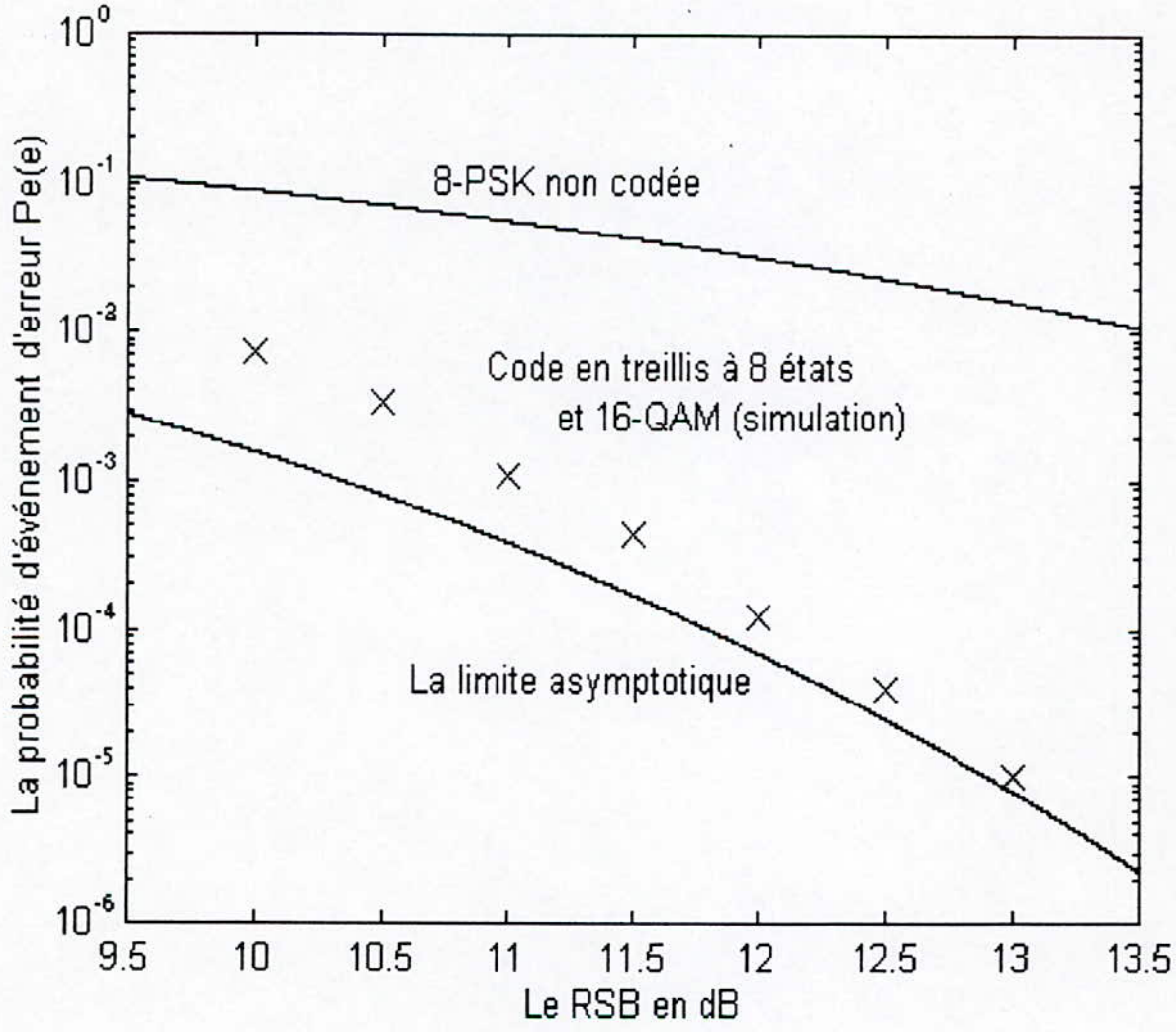


Figure 3.11.a La probabilité d'événement d'erreur d'un code en treillis à 8 états et 16-QAM en fonction du RSB.

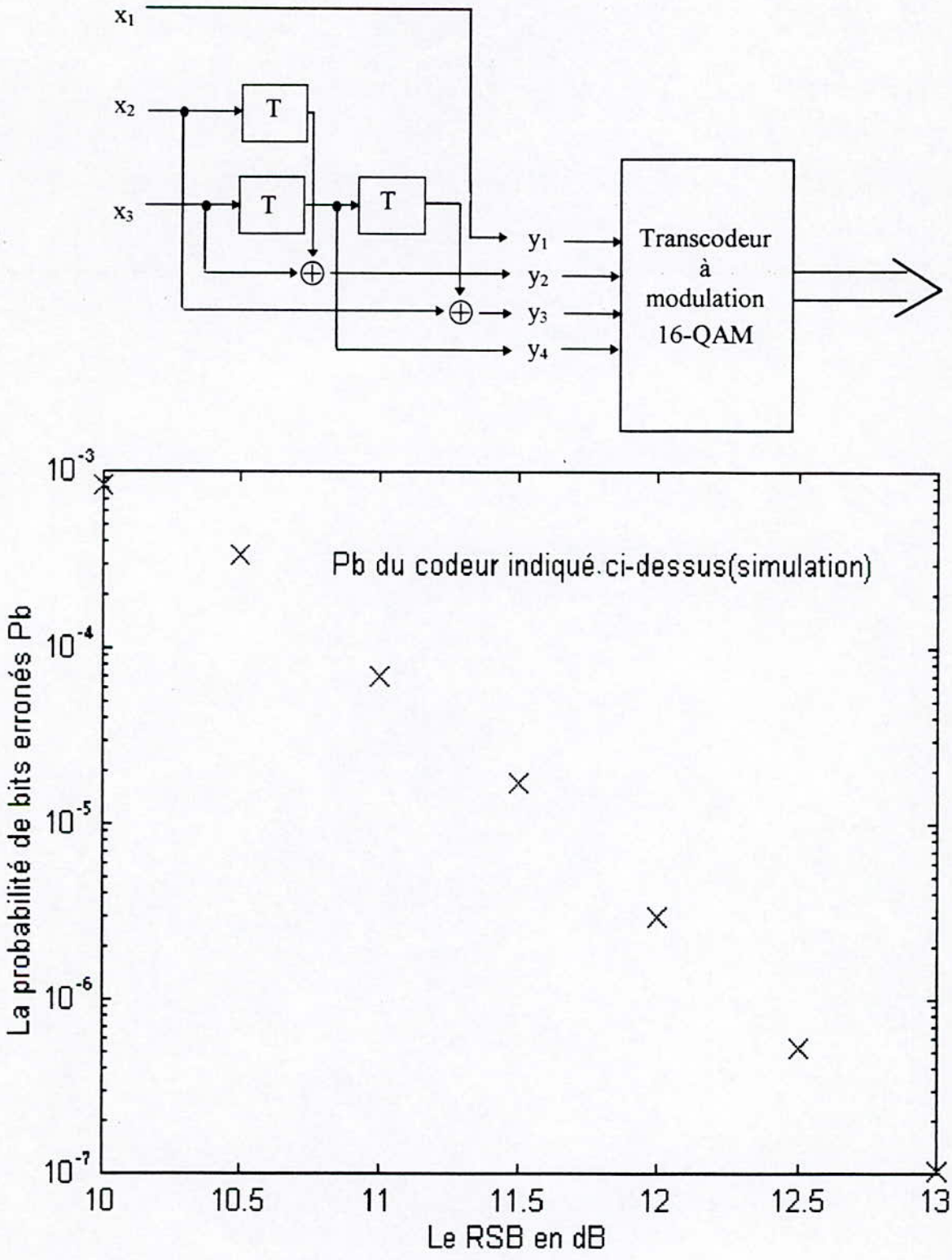


Figure 3.11.b La probabilité de bits erronés du codeur en treillis indiqué ci-dessus en fonction du RSB.

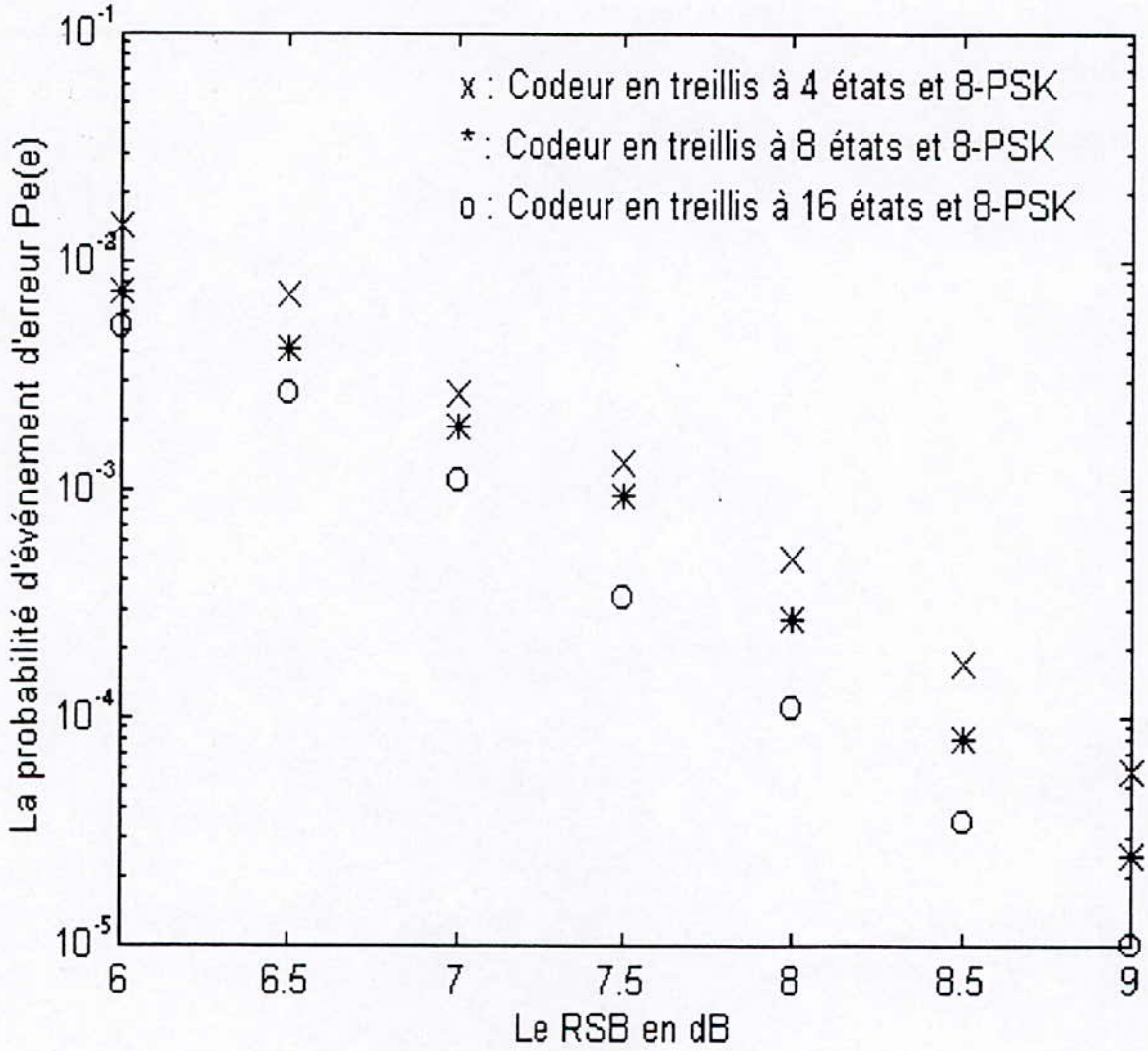


Figure 3.12 Comparaison entre les performances des codeurs

i) codeur en treillis à 4 états et 8-PSK;

ii) codeur en treillis à 8 états et 8-PSK; et

iii) codeur en treillis à 16 états et 8-PSK.

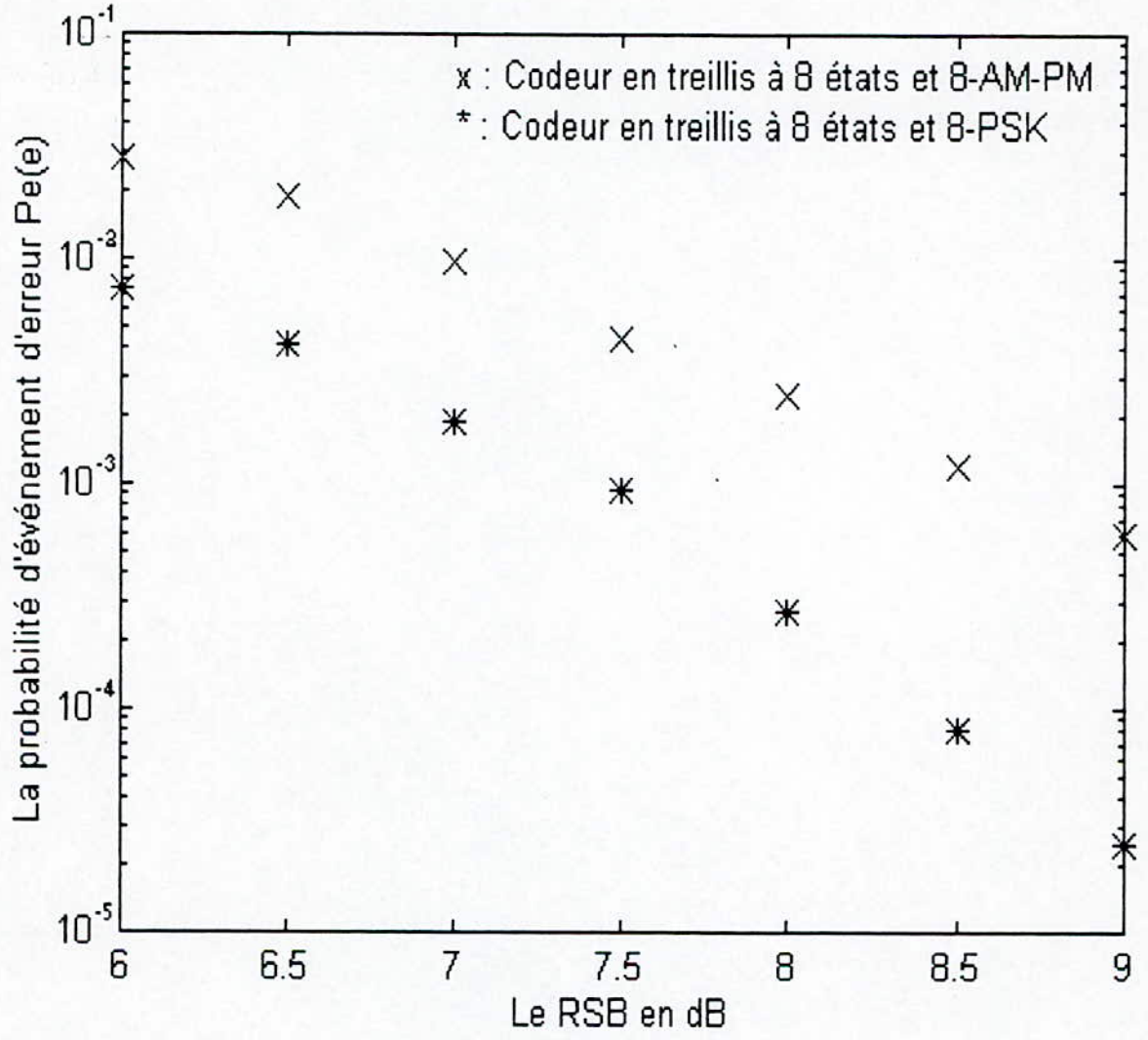


Figure 3.13 Comparaison entre les performances des codeurs

i) codeur en treillis à 8 états et 8-AM-PM; et

ii) codeur en treillis à 8 états et 8-PSK.

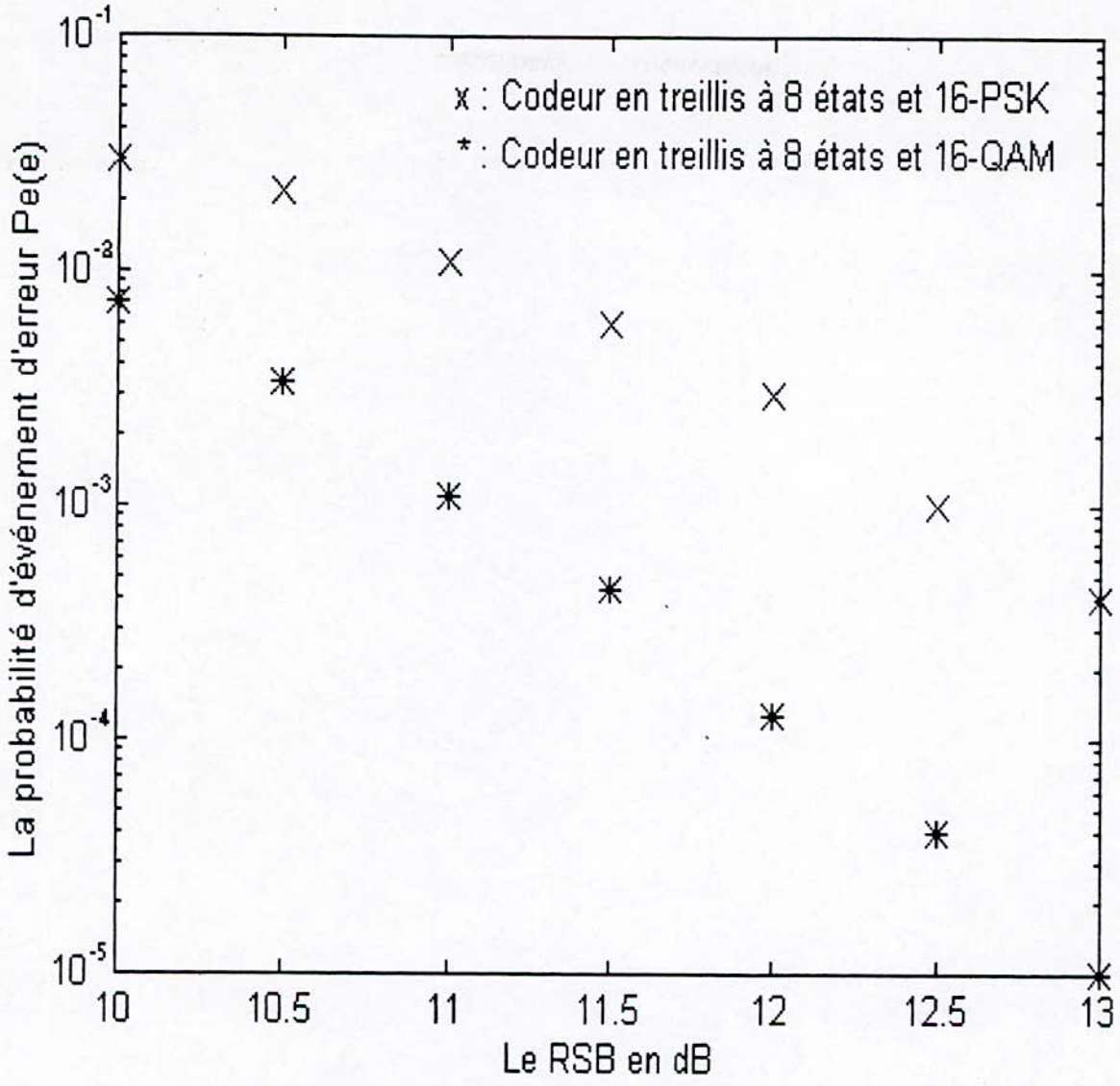
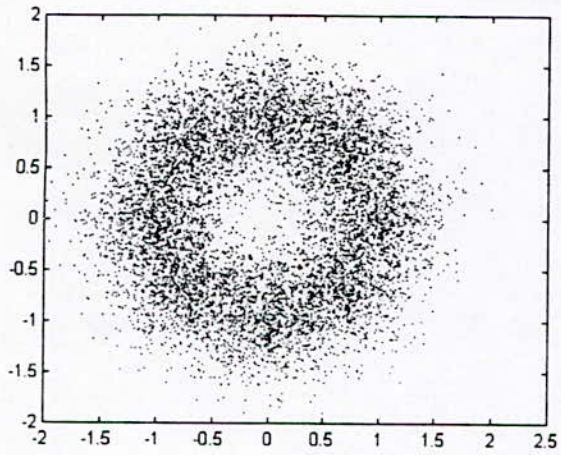


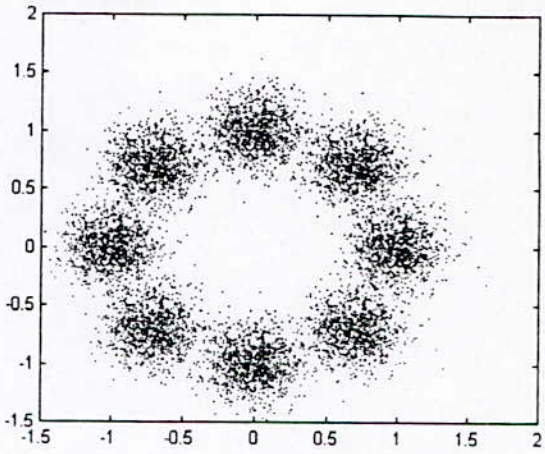
Figure 3.14 Comparaison entre les performances des codeurs

i) codeur en treillis à 8 états et 16-PSK; et

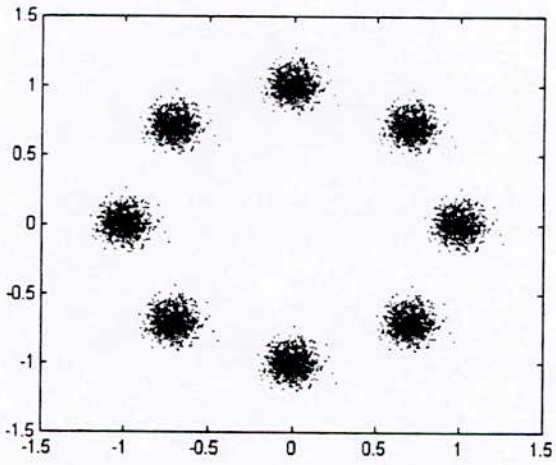
ii) codeur en treillis à 8 états et 16-QAM..



(a) Pour un RSB=7 dB.



(b) Pour un RSB=12.6 dB.



(c) Pour un RSB=20 dB.

Figure 3.15 L'influence du bruit sur la détection de la modulation 8-PSK.

3.4 Conclusion

Notre simulation a permis de calculer les performances des systèmes TCM. Les résultats obtenus sont en conformité avec la théorie et son très proches des résultats d'Ungerboeck.

Notre travail s'est limite bien sur aux deux types de modulations, PSK et QAM les plus utilisées dans le domaine. Il existe d'autres modulations qui peuvent donner des performances meilleurs surtout lorsque le nombre des points messages est grand.

On peut citer quelques constellations telles que la constellation hexagone et la constellation étoile. Elles peuvent êtres développées.

Pour la transmission des informations avec des grands débits, et avec une performances acceptable (satisfaisante), les codeurs convolutifs à grand nombre d'états son les plus performants et la modulation QAM est la plus utilisée.

CONCLUSION

Conclusion

Selon le cahier de charge exposé, notre travail concernant la modulation codée en treillis (TCM), a englobé deux parties : une théorique et l'autre une simulation.

La partie théorique a pour but de dégager les fondements théoriques de ces codeurs de canaux, à savoir : les théorèmes fondamentaux du codage de canal, le principe des TCM et les règles du transcodage par la partition d'ensemble. Nous avons détaillé l'algorithme de Viterbi et le principe du maximum de vraisemblance. Les premiers chapitres ont été consacrés à cette partie.

La deuxième partie était la mise en oeuvre d'une simulation permettant d'évaluer les performances des TCM, et cela en estimant la probabilité d'événement d'erreur $P_e(e)$ de ces modulations en fonction du RSB, le décodage étant réalisé à l'aide de l'algorithme de Viterbi . Cette simulation est réalisée pour différants codeurs en treillis utilisant des codeurs convolutifs de nombre d'états différent et des types de modulations QAM ou PSK.

Les moyens informatiques disponibles dans notre laboratoire ont limité la génération de la séquence d'information pseudo-aléatoire à une période 2^{16}-1=65535, et les calculs des intervalles d'incertitudes dans les courbes obtenues (temps d'éxécution des programmes est trés grand).

Cette simulation a permis de calculer les paramètres d'évaluation des performances des systèmes TCM, tels que la probabilité d'événement d'erreur $P_e(e)$, la probabilité de bits erronés Pb, etc. L'allure des courbes obtenues correspond bien à la théorie, puisque $P_e(e)$ diminue avec l'augmentation du RSB, converge supérieurement vers sa limite asymptotique, et reste toujours inférieure à la probabilité d'événement d'erreur du système sans codage (système de référence). Les résultats obtenus sont en conformité avec la théorie et sont très proches des résultats d'Ungerboeck [17], [18].

Notre travail s'est limite bien sur aux deux types de modulations, PSK et QAM les plus utilisées dans le domaine. Il existe d'autres modulations qui peuvent donner des performances meilleurs surtout lorsque le nombre des points messages est grand (constellations hexagonale et étoile...)

Dans la TCM à deux dimensions, pour affecter plus d'informations à un point message, il faut augmenter la taille de la constellation, donc augmentation des erreurs de transmission en coséquence pour un RSB donné. Pour transmettre une information avec un grand débit, et de faibles erreurs de transmission, il faut augmenter le RSB à une valeur très grande, car les TCM à deux dimensions utilisent une constellation de 2^{m+1} points pour transmettre m bits d'information par symbole et cette expansion d'alphabet coûte 3 dB. Une solution à ces problèmes est d'utiliser des modulations codées en treillis multidimensionnelles. Ces modulations seront l'objet d'un projet PFE pour l'année 97/98 . Ces derniéres ont pour but principal de réduire cette perte en utilisant le facteur d'expansion d'alphabet. (ce facteur d'expansion est $2^{1/n}$ pour une TCM de dimension 2n).

BIBLIOGRAPHIE

BIBLIOGRAPHIE

[1] J. B. Anderson and S. Mohan, *Source and Channel Coding*, Kluwer Academic Publishers, New York, 1991.

[2] S. Bellag, M. Guellai, « Simulation, Conception et Evaluation des Systèmes de Communication Numérique, » *Projet de Fin d'Etude*, ENP Département Electronique, 1995

[3] S. Benedetto, M. A.Marsan, Albertengo and E. Giachin, « Combined Coding and Modulation: Theory and Applications, » *IEEE Trans.On Information Theory*, Vol. IT-34, pp. 223-236, March 1988.

[4] J.C. Bic, D. Duponteil, J. C. Lmbeaux, *Elements de Communications Numérioques: Transmission sur Fréquence Porteuse*, Tome II, Bordas et C.N.E.T.E.N.S.T, Paris, 1986.

[5] A.R. Calderbank and J. Mazo, « A New Description of Trellis Codes, » *IEEE Trans. On Information Theory*, Vol.IT-30, pp. 784-791, Novem, 1984.

[6] F. Flitti et Y. Omarouyoub, « Codes Covolutifs: Etude, Simulation et valuation , », *Projet de Fin d'Etude*, ENP, Département d'Electronique , Sept. 1996.

[7] G. D. Forney, Jr. R.G. Gallager, G. R. Lang, F. M. Longstaff and S.U. Qureshi, «Efficient Modulation for Band-Limited Channels, », *IEEE Journal on Selected Areas in Communication*, Vol. SAC-2, pp. 632-647, Sept. 1984.

[8] G. D. Forney, JR. , « The Viterbi Algorithm, » *Proceedings of the IEEE*, Vol. IT-61, pp. 268-278, March 1973.

[9] G. D. Forney, JR. , « Convolutional Codes I: Algebraic Structure, » , *IEEE Trans. on Information Theory*, Vol. IT- 6, pp. 720-738, Novem, 1970.

[10] A. Glavieux, « Codage de l'Information et Modulation des Signaux, » *Techniques de l'Ingénieur, Traité Mesure et Contrôle* , R 308-1,1996.

[11] S. Haykin, *Digital Communication*, John Wiley & Sons, New York,1988.

[12] E. A. Lee and D. G. Messershmitt, *Digital Communication*, Kluwer Academic Publishers, New York, USA, 1994.

[13] F. J. MacWilliams and N. J. A. Sloane , « Pseudo-Random Sequences and Arrays, » *Proceedings of the IEEE*, Vol. IT-64, pp. 1715-1728, Decem. 1976.

[14] J. G. Proakis, *Digital Communication*, McGraw Hill, New York. 1989.

[15] H. Sari, « Transmission des Signaux Numériques, » *Techniques de l'Ingénieur, Traité Electronique*,E 7100, 1996.

[16] A. Spâtaru, *Fondements de la Théorie de la Transmission l'Information*, Presses Polytechnique Romandes. Lausanne, Suisse, 1987.

[17] G. Ungerboeck, « Channel Coding with Multilevel / Phase Signals, » *IEEE Trans. on Information Theory*, Vol. IT-28, pp.55-67, Janu 1982.

[18] G. Ungerboeck, « Trellis-Coded Modulation with Redundant Signal Sets Part I: Introduction, » *IEEE Communication Magazine*, Vol. 25, pp.5-11, Febru 1987.

[19] G. Ungerboeck, « Trellis-Coded Modulation with Redundant Signal Sets Part II: State of the Art, » *IEEE Communication Magazine*, Vol. 25, pp.12- 21, Febru 1987.

[20] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding*, New York: Mac Graw-Hill, 1979.

[21] L. F. Wei, « Trellis-Coded Modulation with Multidimensional Constellations, » *IEEE Trans. on Information Theory*, Vol. IT-33, July 1987.

[22] E. Zehavi and J. K. Wolf, « On the Performance Evaluation of Trellis Codes » *IEEE Trans. on Information Theory*, Vol. IT-33, pp. 196-202, March 1987.

ANNEXES A, B

ANNEXE A

LES SEQUENCES PSEUDO-ALEATOIRES

A.1 Introduction

Les séquences pseudo-aléatoires (qui sont appelées aussi les séquences pseudo-bruit (PN), séquences à registre à décalage de longueur maximale, ou *m*- séquences)sont des séquences binaires avec une longueur $n = 2^m - 1$.

Ils ont plusieurs propriétés pratiques, une de ces propriétés, est la fonction d'autocorilation (périodique) qui est donnée par

$$\rho(0) = 1,\ \rho(i) = -1/n \quad pour \quad 1 \le i \le n-1 . \qquad \text{(A-1)}$$

A.2 Les séquences pseudo-aléatoires

A.2.a Le registre à décalage

Pour construire une séquence pseudo-aléatoire avec un longueur $n = 2^m - 1$, on a besoin seulement du polynôme primitif $h(x)$ de degré m. Ce terme est défini ci-après.

Soit le polynôme

$$h(x) = x^4 + x + 1 \qquad \text{(A-2)}$$

ayant un degré $m = 4$ par exemple. Ce polynôme spécifie un registre à décalage avec un feed-back comme indiquer dans la figure A.1 . En général le registre à décalage est constitué par m éléments , chaque élément contient un 0 ou 1. A chaque unité de temps les contenus de ces éléments sont décalés d'une seule place vers la droite , et les éléments correspondants aux termes dans $h(x)$ sont additionnés et injectés dans l'élément mémoire numéro 4 (la somme est calculée en modulo-2).

Dans l'exemple ci- dessus, si le registre contient a_{i+3}, a_{i+2}, a_{i+1}, a_i à l'instant i, alors à l'instant $i+1$ il contient

$$a_{i+4} = a_{i+1} + a_i, \quad a_{i+3}, \quad a_{i+2}, \quad a_{i+1},$$

Comme indiquer dans la figure A.2 . Autrement dit, ce registre à décalage avec m feed-back génère une séquence infinie $a_0, a_1, a_2, \ldots, a_i, \ldots$ qui satisfait à la récurrence

$$a_{i+4} = a_{i+1} + a_i \qquad i = 0,1,... \tag{A-3}$$

où + représente l'addition modulo-2, et il font spécifier les valeurs initiales $a_0,\ldots,a_{m-1}$.

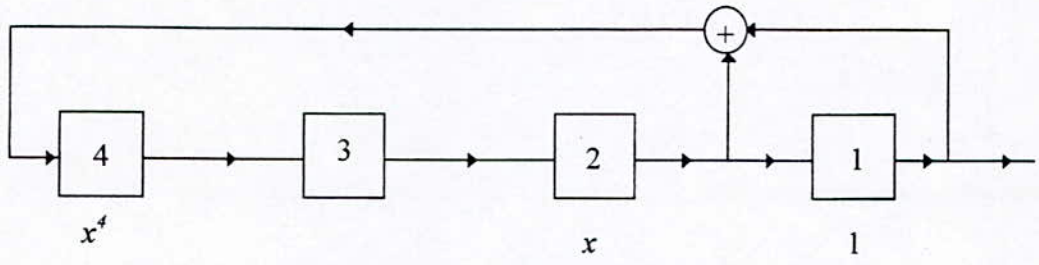


Figure A.1 Registre à décalage avec un feed-back correspondant *à* $x^4 + x + 1$

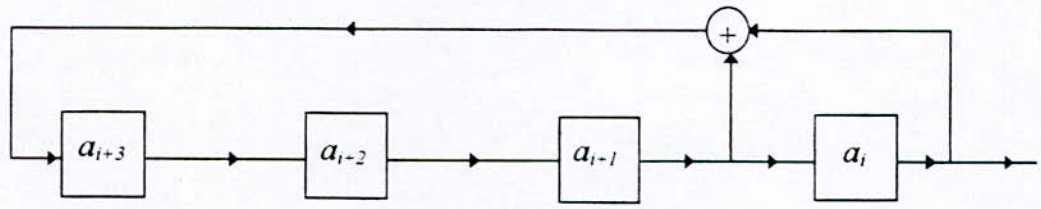


Figure A.2 Registre à décalage spécifiés à une relation de récurrence.

A.2 .b Séquences pseudo-aléatoires

Puisque chaque élément mémoire parmi les m éléments mémoires contient un 1 ou 0, donc il y'a 2^n états possibles .Ainsi la séquence a_0, a_1,... doit être périodique. Mais l'état 00...0 ne peut pas se produire à moins que la séquence est toute zéros, donc la période maximale possible est 2^m-1.

Maintenant nous pouvons définir le polynôme primitif $h(x)$. La séquence de sortie dans la figure A.1 ayant une période $2^4-1=15$, le polynôme primitif est $x^4 + x + 1$.

On accepte que pour chaque m il existe un polynôme primitif de degré m. Dans la figure A.3, on donne un tableau des polynômes primitifs pour $m \leq 40$. Si $h(x)$ un polynôme primitif de degré m , le registre à décalage passe par tous les états distinctes (non nulles) avant la répétition, et produit une séquence de sortie de période 2^m-1. En chaque segment

$$a_i, a_{i+1}, \ldots, a_{i+2^m-2}, \tag{A-4}$$

de longueur 2^m-1 une séquence pseudo-aléatoire. Il y'a 2^m-1 séquences pseudo-aléatoires différentes.

deg m	$h(x)$	deg m	$h(x)$
1	$x+1$	21	$x^{21}+x^2+1$
2	x^2+x+1	22	$x^{22}+x+1$
3	x^3+x+1	23	$x^{23}+x^5+1$
4	x^4+x+1	24	$x^{24}+x^4+x^3+x+1$
5	x^5+x^2+1	25	$x^{25}+x^3+1$
6	x^6+x+1	26	$x^{26}+x^8+x^7+x+1$
7	x^7+x+1	27	$x^{27}+x^8+x^7+x+1$
8	$x^8+x^6+x^5+x+1$	28	$x^{28}+x^3+1$
9	x^9+x^4+1	29	$x^{29}+x^2+1$
10	$x^{10}+x^3+1$	30	$x^{30}+x^{16}+x^{15}+x+1$
11	$x^{11}+x^2+1$	31	$x^{31}+x^3+1$
12	$x^{12}+x^7+x^4+x^3+1$	32	$x^{32}+x^{28}+x^{27}+x+1$
13	$x^{13}+x^4+x^3+x+1$	33	$x^{33}+x^{13}+1$
14	$x^{14}+x^{12}+x^{11}+x+1$	34	$x^{34}+x^{15}+x^{14}+x+1$
15	$x^{15}+x+1$	35	$x^{35}+x^2+1$
16	$x^{16}+x^5+x^3+x^2+1$	36	$x^{36}+x^{11}+1$
17	$x^{17}+x^3+1$	37	$x^{37}+x^{12}+x^{10}+x^2+1$
18	$x^{18}+x^7+1$	38	$x^{38}+x^6+x^5+x+1$
19	$x^{19}+x^6+x^5+x+1$	39	$x^{39}+x^4+1$
20	$x^{20}+x^3+1$	40	$x^{40}+x^{21}+x^{19}+x^2+1$

Figure A.3 Polynômes primitifs de $m \leq 40$.

Remarque

Il est possible d'atteindre une période 2^m (au lieu de 2^m-1) on utilise un registre à décalage non linéaire.

A.2.c Propriétés des séquences pseudo- aléatoires

Soit $h(x)$ un polynôme primitif de degré m et soit δ_m l'ensemble constitué de ces séquences pseudo-aléatoires générées à partir de $h(x)$.

Propriété 1- La propriété de décalage : si $b_0, b_1, \ldots, b_{2^m-2}$ est une séquence pseudo-aléatoire quelconque dans δ_m, alors tout décalage cyclique de b, noté $b_j\, b_{j+1} \ldots b_{2^m-2}\, b_0 \ldots b_{j-1}$ est aussi dans δ_m.

Propriété 2- La récurrence : On suppose que $h(x)=\sum_{i=0}^{m} h_i x^i$ avec $h_0=h_m=1$, $h_i=0$ ou 1 pour $0<i<m$. Toute séquence pseudo-aléatoire $b \in \delta_m$ satisfaite la récurrence

$$b_{i+m}=h_{m-1}b_{i+m-1} + h_{m-2}b_{i+m-2} +.....+ h_1 b_{i+1} + b_1 \qquad \text{pour } i= 0,1,... \qquad \text{(A-5)}$$

Réciproquement toute solution de (A-5) est dans δ_m. Il y'a m solutions de (A-5) linéaires indépendantes, d'où m séquences linéaires indépendantes dans δ_m.

Propriété 3-Propriété de fenêtre : toute fenêtre de longueur *m* (glissée en avant) d'une séquence pseudo-aléatoire dans δ_m est apparaître exactement une seule fois.(figure A.4)

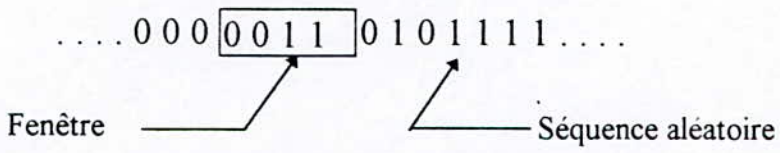


Figure A.4 La propriété de fenêtre.

Propriété 4 - Demi 0 et demi 1 : Chaque séquence pseudo-aléatoire dans δ_m contient 2^{m-1} « *1* » et $2^{m-1}-1$ « *0* ».

Propriété 5 - Propriété du décalage et de l'addition : La somme de deux séquences dans δ_m est une autre séquence dans δ_m (pour la somme de deux solutions de (A-5) est une autre solution).

Propriété 6 - Propriété du décalage et de l'addition : La somme d'un séquence pseudo-aléatoire et un décalage cyclique de lui même est une autre séquence pseudo-aléatoire et *m* décalage cyclique de lui même est une autre séquence pseudo-aléatoire (à partir de propriétés (1) et (2)).

A.2.d. La fonction d'autocorélation

On arrive maintenant à la propriété la plus importante, *la fonction d'autocorelation* $\rho(i)$ d'une séquence réel (complexe) $s_0\, s_1\, ...s_{n-1}$ de longueur *n* est définie par

$$\rho(i)=\frac{1}{n}\sum_{j=0}^{n-1} s_j \bar{s}_{i+j} \quad i=0,\pm1,\pm2,.... \qquad \text{(A-6)}$$

C'est une fonction périodique : $\rho(i) = \rho(i+n)$. La fonction d'autocorrelation d'une séquence binaire $a_0\ a_1\ a_2\ .....a_{n-1}$ doit être égale à la fonction d'autocorrelation de séquence réel $(-1)^{a_0},.....,(-1)^{a_{n-1}}$ obtenu, on remplaçant les « 1 » par « -1 » et les « 0 » par « +1 », ainsi

$$\rho(i) = \frac{1}{n}\sum_{j=0}^{n-1}(-1)^{a_j + a_{i+j}} \qquad \text{(A-7)}$$

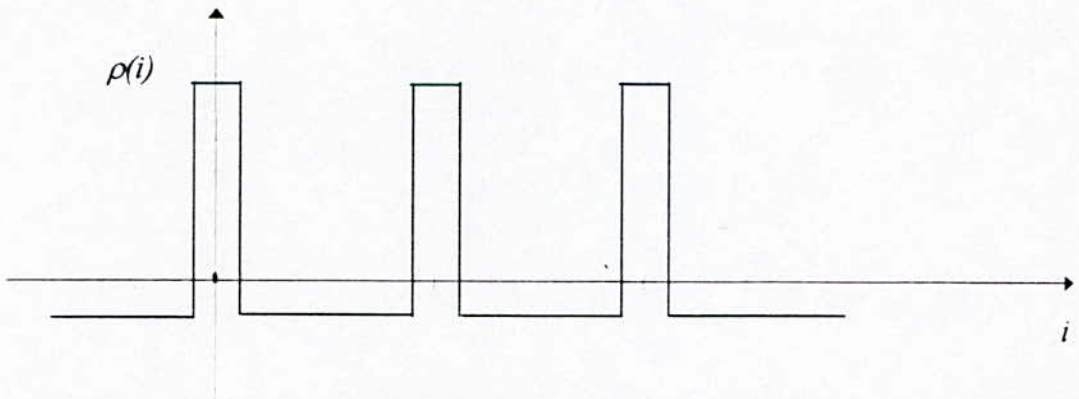


Figure A .5 Fonction d'autocorrelation d'une séquence pseudo-aléatoire

Propriété 7 - fonction d'autocorrelation : La fonction d'autocorrelation d'une séquence pseudo-aléatoire de longueur $n = 2^m - 1$ est donnée par :

$$\rho(0) = 1$$

$$\rho(i) = -1/n \text{ pour } i = 1,..2^m - 2$$

A.2.e Séries

On défini une séries comme chaîne maximale des symboles consécutives identiques. Par exemple, pour la séquence 000100110101111 contient des séries de quatre 1, trois 0, deux 1, deux 0, deux séries simples de 1 et deux séries simples de 0, la totale est de 8 séries.

Propriété 8- Séries : Dans chaque séquence pseudo-aléatoire 1/2 de série ayant longueur 1, 1/4 ayant longueur 2,ayant 1/8 ayant longueur 3, et ainsi de suite, jusqu'à $1/2^{m-1}$.
Dans chaque cas, le nombre de séries de 0 est égale au nombre de séries de 1.

A.2.f Séquences pseudo- aléatoires *q*-aires

Les séquences pseudo-aléatoires peuvent être définies sur des corps de Galois q-éléments avec q premier, ou sur des corps de Galois étendus.
Ces séquences sont utilisable dans les simulations de transmissions numériques à q-états.

ANNEXE B

LES PROGRAMMES DE LA SIMULATION

Le programme du codeur de la figure 3.3.a à modulation 4-PSK (TCM 1)

```
clear
RSB=input('donner la valeur du rapport signal sur bruit en dB ')
load D4;lm=60000;dp=2000;
ss=[1 2 3 4;3 4 1 2];
load x ;
nex(dp+1)=1;ph=0;nef=1;fr=0;
for N=1:lm/dp
fne=fix((nef-1)/2)+1;
rne=rem(nef-1,2)+1;
ref=2*rne-1;
nex(1)=nex(dp+1);
for k=1:dp
kk=(N-1)*dp+k;
%********** Le codage:
y1=rem(x(kk-2)+x(kk),2);
y2=rem(x(kk-2)+x(kk+1)+x(kk),2);
nex(k+1)=1+x(kk+2)+2*rem(nex(k)-1,2);
%********** Le mapping:
iss(k)=1+3*y1+y2-2*y1*y2;
s=1-(y1+y2)-(y1-y2)*i;
%s=exp((iss(k)-1)*i*pi/2);
%********** Les symboles bruités:
r=randn(1,2);
z=s+10.^(-RSB/20)*0.707*(r(1)+i*r(2));
%********** La décision:
[z is]=min(abs((exp((0:3)*i*pi/2)-z)'));
%********** Le décodage:
if k==1
ls1=[D4(ss(1,nef),is)  D4(ss(2,nef),is)];
f(1,ref:ref+1)=[fne fne];
elseif k==2
ls2=[D4(ss(1,ref),is)  D4(ss(2,ref),is)]+ls1(1);
ls1=[D4(ss(1,ref+1),is)  D4(ss(2,ref+1),is)]+ls1(2);
f(2,1:4)=[rne rne rne rne];
elseif k>2
ls4=[D4(1,is) D4(3,is)]+ls2(1);
ls3=[D4(2,is) D4(4,is)]+ls2(2);
ls2=[D4(3,is) D4(1,is)]+ls1(1);
ls1=[D4(4,is) D4(2,is)]+ls1(2);
[ls2 f(k,1:2)]=min([ls4;ls2]);
[ls1 f(k,3:4)]=min([ls3;ls1]);
end
```

```
end
[dmin nef]=min([ls2 ls1]');ne(dp+1)=nef;
for k=dp:-1:1
kk=(N-1)*dp+k;
itg=f(k,ne(k+1));
ne(k)=fix((ne(k+1)-1)/2)+2*itg-1;
itd(k)=rem(ne(k+1)-1,2);
%********** Le poids de Hamming 'ph':
ph=itd(k)+x(kk+2)-2*x(kk+2)*itd(k)+ph;
end
p=0;
for k=1:dp-1
if (ne(k)==nex(k) & ne(k+1)~=nex(k+1))
t1=k;
end
if (ne(k+1)~=nex(k+1) & ne(k+2)==nex(k+2))
t2=k+1;
d=0;
for l=t1:t2
d=d+D4(iss(l),ss(itd(l)+1,ne(l)));
end
if d>=10
p=p+0.5*erfc(sqrt(d)/(2*10.^(-RSB/20)));
end
if d==10
fr=fr+1;
end
end
end
end
pe=p/fr;
```

Le programme du codeur de la figure 3.3.b à modulation 8-PSK (TCM 2)

```
clear
RSB=input('donner la valeur du rapport signal sur bruit en dB ')
load D8;
%******** La matrice caractéristique de mapping :
ss=[1 2 3 4;5 6 7 8;3 4 1 2;7 8 5 6];
load x;
dp=2000;lm=60000;fr=0 ;
nef=1;ph=0;
nex(dp+1)=1;
for N=1:lm/(2*dp)
nex(1)=nex(dp+1);
ref=2*rem(nef-1,2)+1;
fne=2*fix((nef-1)/2);
for k=1:dp
kk=2*((N-1)*dp+k);
nex(k+1)=3-2*rem(nex(k),2)+x(kk+4) ;
%*********Sorties du codeur convolutif
y1=x(kk+3);
```

Chargement de l'information pseudo-aléatoire.

lm: la longueur du message, et dp : le depth

Changement de variable .

Ce paramètre permet de donner le trajectoire correspond à l'information dans le treillis.

```
y2=rem(x(kk+4)+x(kk),2);
y3=x(kk+2);
%********** Le mapping:
s=exp((y3+2*y2+4*y1)*pi*i/4) ;          La sortie du transcodeur.
%********** Le numéro du symbole qui sort du transcodeur est donné par
iss(k)=1+y3+2*y2+4*y1;
r=randn(1,2);
z=s+10.^(-RSB/20)*0.707*(r(1)+i*r(2)) ;     L'interférance du bruit.
%********** La décision:
[z is]=min((abs(exp((0:7)*pi*i/4)-z)') );     Le numéro du symbole réçu.
%********** Le décodage
if k==1
ls1=[D8(ss(1,nef),is) D8(ss(3,nef),is);D8(ss(2,nef),is) D8(ss(4,nef),is)];
[ls1 f(1,ref:ref+1)]=min(ls1);f(1,ref:ref+1)=f(1,ref:ref+1)+fne;
elseif k==2
ls2=[D8(ss(1,ref),is) D8(ss(3,ref),is);D8(ss(2,ref),is) D8(ss(4,ref),is)]+ls1(1);
[ls2 f(2,1:2)]=min(ls2);f(2,1:2)=f(2,1:2)+ref-1;
ls1=[D8(ss(1,ref+1),is) D8(ss(3,ref+1),is);D8(ss(2,ref+1),is) D8(ss(4,ref+1),is)]+ls1(2);
[ls1 f(2,3:4)]=min(ls1);f(2,3:4)=f(2,3:4)+ref-1;
elseif k>2
ls4=[D8(1,is) D8(3,is);D8(5,is) D8(7,is)]+ls2(1);
ls3=[D8(2,is) D8(4,is);D8(6,is) D8(8,is)]+ls2(2);
ls2=[D8(3,is) D8(1,is);D8(7,is) D8(5,is)]+ls1(1);
ls1=[D8(4,is) D8(2,is);D8(8,is) D8(6,is)]+ls1(2);
[ls2 f(k,1:2)]=min([ls4;ls2]);
[ls1 f(k,3:4)]=min([ls3;ls1]);
end
end
[dmin ne(dp+1)]=min([ls2 ls1]');
nef=ne(dp+1) ;          nef : le numéro d'état final dans chaque fenétre
for k=dp:-1:1                                          (ou depth).
kk=2*((N-1)*dp+k);
%********** L'Indice de Transition Gauche
itg=f(k,ne(k+1));
%********** L' indice de Transition Droite
itd(k)=rem(itg-1,2)+2*rem(ne(k+1)-1,2);
%********** Le paramètre qui donne le trajectoire de l'information réçue.
ne(k)=1+2*fix((itg-1)/2)+fix((ne(k+1)-1)/2);
%********** Le poids de Hamming 'ph':
ph=rem(itd(k),2)+x(kk+4)+fix(itd(k)/2)+x(kk+3)-
2*(x(kk+3)*rem(itd(k),2)+fix(itd(k)/2)*x(kk+4))+ph;
end
%********** Le calcul de la probabilité d'événement d'erreur
p=0;
for k=1:dp
if ne(k)==nex(k)
t1=k;
end
if ne(k+1)==nex(k+1)
t2=k;
d=0;
for l=t1:t2
d=d+D8(iss(l),ss(itd(l)+1,ne(l)));
```

```
end
if d >=4
p=p+0.5*erfc(sqrt(d)/(2*10.^(-RSB/20)));
end
if d==4
fr=fr+1;
end
end
end
end
pe=p/fr;
```

Le programme du codeur de la figure 3.3.c à modulation 8-PSK (TCM 3)

```
clear
RSB=input('Donner la valeur du rapport signal sur bruit en dB ')
load D8; lm=60000; dp=2000;
ss=[1 2 5 6 3 4 7 8;5 6 1 2 7 8 3 4;3 4 7 8 1 2 5 6;7 8 3 4 5 6 1 2];
load x;
nex(dp+1)=1;ph=0;nef=1;fr=0;
for N=1:lm/(2*dp)
nex(1)=nex(dp+1);
ref=rem(nef-1,2)-1;
fne=fix((nef-1)/2)-1;
for k=1:dp
kk=2*((N-1)*dp+k);
%********** Le codage:
y1=rem(x(kk+4)+x(kk+1),2);
y2=rem(x(kk+3)+x(kk),2);
y3=x(kk+2);
nex(k+1)=1+x(kk-4)+2*x(kk+3)+4*rem(nex(k)-1,2);
%********** Le mapping:
iss(k)=1+y3+2*y2+4*y1;
s=exp((iss(k)-1)*pi*i/4);
%********** Les symboles bruités:
r=randn(1,2);
z=s+10.^(-RSB/20)*0.707*(r(1)+i*r(2));
%********** La décision:
[z is]=min((abs(exp((0:7)*pi*i/4)-z)'));
%********** Le décodage:
if k==1
ls1=[D8(ss(1,nef),is) D8(ss(2,nef),is) D8(ss(3,nef),is) D8(ss(4,nef),is)];
f(1,4*ref-3:4*ref)=[fne fne fne fne];
elseif k==2
ls4=[D8(ss(1,4*ref-3),is) D8(ss(2,4*ref-3),is) D8(ss(3,4*ref-3),is) D8(ss(4,4*ref-3),is)]+ls1(1);
ls3=[D8(ss(1,4*ref-2),is) D8(ss(2,4*ref-2),is) D8(ss(3,4*ref-2),is) D8(ss(4,4*ref-2),is)]+ls1(2);
ls2=[D8(ss(1,4*ref-1),is) D8(ss(2,4*ref-1),is) D8(ss(3,4*ref-1),is) D8(ss(4,4*ref-1),is)]+ls1(3);
ls1=[D8(ss(1,4*ref ),is) D8(ss(2,4*ref ),is) D8(ss(3,4*ref ),is) D8(ss(4,4*ref ),is)]+ls1(4);
[ls2 f(2,1:4)]=min([ls4;ls2]);f(2,1:4)=f(2,1:4)+2*(ref-1);
[ls1 f(2,5:8)]=min([ls3;ls1]);f(2,5:8)=f(2,5:8)+2*(ref-1);
elseif k > 2
ls8=[D8(1,is) D8(5,is) D8(3,is) D8(7,is)]+ls2(1);
```

```
ls7=[D8(2,is) D8(6,is) D8(4,is) D8(8,is)]+ls2(2);
ls6=[D8(5,is) D8(1,is) D8(7,is) D8(3,is)]+ls2(3);
ls5=[D8(6,is) D8(2,is) D8(8,is) D8(4,is)]+ls2(4);
ls4=[D8(3,is) D8(7,is) D8(1,is) D8(5,is)]+ls1(1);
ls3=[D8(4,is) D8(8,is) D8(2,is) D8(6,is)]+ls1(2);
ls2=[D8(7,is) D8(3,is) D8(5,is) D8(1,is)]+ls1(3);
ls1=[D8(8,is) D8(4,is) D8(6,is) D8(2,is)]+ls1(4);
[ls2 f(k,1:4)]=min([ls8;ls6;ls4;ls2]);
[ls1 f(k,5:8)]=min([ls7;ls5;ls3;ls1]);
end
end
[dmin nef]=min([ls2 ls1]');ne(dp+1)=nef;
for k=dp:-1:1
kk=2*((N-1)*dp+k);
itg=f(k,ne(k+1));
itd(k)=rem(ne(k+1)-1,4);
ne(k)=2*itg+fix((ne(k+1)-1)/4)-1;
%********** Le poids de Hamming 'ph':
ph=ph+rem(itd(k),2)+x(kk+4)+fix(itd(k)/2)+x(kk+3)-2*(rem(itd(k),2)*x(kk+4)...
+fix(itd(k)/2)*x(kk+3));
end
p=0;
for k=1:dp-1
if (ne(k)==nex(k) & ne(k+1)~=nex(k+1))
t1=k;
end
if (ne(k+1)~=nex(k+1) & ne(k+2)==nex(k+2))
t2=k+1;
d=0;
for l=t1:t2
d=d+D8(iss(l),ss(itd(l)+1,ne(l)));
end
if d>=(4+4*(sin(pi/8)).^2)
p=p+0.5*erfc(sqrt(d)/(2*10.^(-RSB/20)));
end
if d==4+4*(sin(pi/8)).^2
fr=fr+1;
end
end
end
end
pe=2*p/fr;
```

Le programme du codeur de la figure 3.3.d à modulation 8-PSK (TCM 4)

```
clear
RSB=input('Donner la valeur du rapport signal sur bruit en dB ')
load D8; lm=60000; dp=2000;
nef=1;ph=0;nex(dp+1)=1;fr=0;
load x;

ss=[1 2 5 6 3 4 7 8 5 6 1 2 7 8 3 4;5 6 1 2 7 8 3 4 1 2 5 6 3 4 7 8;
```

```
   3 4 7 8 1 2 5 6 7 8 3 4 5 6 1 2;7 8 3 4 5 6 1 2 3 4 7 8 1 2 5 6];
for N=1:lm/(2*dp)
nex(1)=nex(dp+1);
ref=rem(nef-1,4)+1;
fne=fix((nef-1)/4)+1;
for k=1:dp
kk=2*((N-1)*dp+k);
y1=rem(x(kk+4)+x(kk+1)+x(kk-1),2);
y2=rem(x(kk+3)+x(kk),2);
y3=x(kk+2);
nex(k+1)=1+x(kk+4)+2*x(kk+3)+4*rem(nex(k)-1,2);
%********** Le Mapping:
iss(k)=1+y3+2*y2+4*y1;
s=exp((iss(k)-1)*pi*i/4);
%********** les symboles bruités:
r=randn(1,2);
z=s+10.^(-RSB/20)*0.707*(r(1)+i*r(2));
%********** La décision:
[z is]=min((abs(exp((0:7)*pi*i/4)-z)'));
%********** Le décodage:
if k==1
ls1=[D8(ss(1,nef),is) D8(ss(2,nef),is) D8(ss(3,nef),is) D8(ss(4,nef),is)];
f(1,4*ref-3:4*ref)=[fne fne fne fne];
elseif k==2
ls4=[D8(ss(1,4*ref-3),is) D8(ss(2,4*ref-3),is) D8(ss(3,4*ref-3),is) D8(ss(4,4*ref-3),is)]+ls1(1);
ls3=[D8(ss(1,4*ref-2),is) D8(ss(2,4*ref-2),is) D8(ss(3,4*ref-2),is) D8(ss(4,4*ref-2),is)]+ls1(2);
ls2=[D8(ss(1,4*ref-1),is) D8(ss(2,4*ref-1),is) D8(ss(3,4*ref-1),is) D8(ss(4,4*ref-1),is)]+ls1(3);
ls1=[D8(ss(1,4*ref ),is) D8(ss(2,4*ref ),is) D8(ss(3,4*ref ),is) D8(ss(4,4*ref ),is)]+ls1(4);
f(2,1:16)=zeros(1,16)+ref;
elseif k>2
ls16=[D8(1,is) D8(5,is) D8(3,is) D8(7,is)]+ls4(1);
ls15=[D8(2,is) D8(6,is) D8(4,is) D8(8,is)]+ls4(2);
ls14=[D8(5,is) D8(1,is) D8(7,is) D8(3,is)]+ls4(3);
ls13=[D8(6,is) D8(2,is) D8(8,is) D8(4,is)]+ls4(4);
ls12=[D8(3,is) D8(7,is) D8(1,is) D8(5,is)]+ls3(1);
ls11=[D8(4,is) D8(8,is) D8(2,is) D8(6,is)]+ls3(2);
ls10=[D8(7,is) D8(3,is) D8(5,is) D8(1,is)]+ls3(3);
ls9 =[D8(8,is) D8(4,is) D8(6,is) D8(2,is)]+ls3(4);
ls8 =[D8(5,is) D8(1,is) D8(7,is) D8(3,is)]+ls2(1);
ls7 =[D8(6,is) D8(2,is) D8(8,is) D8(4,is)]+ls2(2);
ls6 =[D8(1,is) D8(5,is) D8(3,is) D8(7,is)]+ls2(3);
ls5 =[D8(2,is) D8(6,is) D8(4,is) D8(8,is)]+ls2(4);
ls4 =[D8(7,is) D8(3,is) D8(5,is) D8(1,is)]+ls1(1);
ls3 =[D8(8,is) D8(4,is) D8(6,is) D8(2,is)]+ls1(2);
ls2 =[D8(3,is) D8(7,is) D8(1,is) D8(5,is)]+ls1(3);
ls1 =[D8(4,is) D8(8,is) D8(2,is) D8(6,is)]+ls1(4);
[ls4 f(k,1:4)]=min([ls16;ls12;ls8;ls4]);
[ls3 f(k,5:8)]=min([ls15;ls11;ls7;ls3]);
[ls2 f(k,9:12)]=min([ls14;ls10;ls6;ls2]);
[ls1 f(k,13:16)]=min([ls13;ls9;ls5;ls1]);
end
end
[dmin nef]=min([ls4 ls3 ls2 ls1]');ne(dp+1)=nef;
```

```
for k=dp:-1:1
kk=2*((N-1)*dp+k);
itg=f(k,ne(k+1));
itd(k)=rem(ne(k+1)-1,4);
ne(k)=4*itg+fix((ne(k+1)-1)/4)-3;
%********** Le poids de Hamming 'ph':
ph=ph+rem(itd(k),2)+x(kk+4)+fix(itd(k)/2)+x(kk+3)-2*(rem(itd(k),2)*x(kk+4)...
+fix(itd(k)/2)*x(kk+3));
end
p=0;
for k=1:dp-1
if (ne(k)==nex(k) & ne(k+1)~=nex(k+1))
t1=k;
end
if (ne(k+1)~=nex(k+1) & ne(k+2)==nex(k+2))
t2=k+1;
d=0;
for l=t1:t2
d=d+D8(iss(l),ss(itd(l)+1,ne(l)));
end
if d>=(4+8*(sin(pi/8)).^2)
p=p+0.5*erfc(sqrt(d)/(2*10.^(-RSB/20)));
end
if d==4+8*(sin(pi/8)).^2
fr=fr+1;
end
end
end
end
pe=3*p/fr;
```

Le programme du codeur de la figure 3.3.e à modulation 16-PSK (TCM5)

```
clear
RSB=input('Donner la valeur du rapport signal sur bruit en dB ')
load D16; lm=60000; dp=2000;fr=0;
ss=[1 2 5 6 3 4 7 8;9 10 13 14 11 12 15 16;5 6 1 2 7 8 3 4;13 14 9 10 15 16 11 12;
   3 4 7 8 1 2 5 6;11 12 15 16 9 10 13 14;7 8 3 4 5 6 1 2;15 16 11 12 13 14 9 10];
load x;
nex(dp+1)=1;ph=0;nef=1;
for N=1:lm/(3*dp)
nex(1)=nex(dp+1);
ref=4*rem(nef-1,2)+1;
fne=2*fix((nef-1)/2);
for k=1:dp
kk=3*((N-1)*dp+k);
%********** Le codage:
y1=x(kk+2);
y2=rem(x(kk+4)+x(kk),2);
y3=rem(x(kk+3)+x(kk-2),2);
y4=x(kk+1);
nex(k+1)=1+x(kk+4)+2*x(kk+3)+4*rem(nex(k)-1,2);
```

```
%********** Le mapping:
iss(k)=1+y4+2*y3+4*y2+8*y1;
s=exp((iss(k)-1)*pi*i/8);
%********** Les symboles bruités:
r=randn(1,2);
z=s+10.^(-RSB/20)*0.707*(r(1)+i*r(2));
%********** La décision:
[z is]=min((abs(exp((0:15)*pi*i/8)-z)'));
%********** Le décodage:
if k==1
ls1=[D16(ss(1,nef),is) D16(ss(3,nef),is) D16(ss(5,nef),is) D16(ss(7,nef),is);
   D16(ss(2,nef),is) D16(ss(4,nef),is) D16(ss(6,nef),is) D16(ss(8,nef),is)];
[ls1 f(1,ref:ref+3)]=min(ls1);
f(1,ref:ref+3)=f(1,ref:ref+3)-fne;
elseif k==2
ls4=[D16(ss(1,ref),is) D16(ss(3,ref),is) D16(ss(5,ref),is) D16(ss(7,ref),is);
   D16(ss(2,ref),is) D16(ss(4,ref),is) D16(ss(6,ref),is) D16(ss(8,ref),is)]+ls1(1);
ls3=[D16(ss(1,ref+1),is) D16(ss(3,ref-1),is) D16(ss(5,ref-1),is) D16(ss(7,ref+1),is);
   D16(ss(2,ref+1),is) D16(ss(4,ref+1),is) D16(ss(6,ref-1),is) D16(ss(8,ref+1),is)]+ls1(2);
ls2=[D16(ss(1,ref+2),is) D16(ss(3,ref-2),is) D16(ss(5,ref-2),is) D16(ss(7,ref+2),is);
   D16(ss(2,ref+2),is) D16(ss(4,ref+2),is) D16(ss(6,ref+2),is) D16(ss(8,ref+2),is)]+ls1(3);
ls1=[D16(ss(1,ref+3),is) D16(ss(3,ref-3),is) D16(ss(5,ref-3),is) D16(ss(7,ref+3),is);
   D16(ss(2,ref+3),is) D16(ss(4,ref+3),is) D16(ss(6,ref+3),is) D16(ss(8,ref+3),is)]+ls1(4);
[ls2 f(2,1:4)]=min([ls4;ls2]);f(2,1:4)=f(2,1:4)+ref-1;
[ls1 f(2,5:8)]=min([ls3;ls1]);f(2,5:8)=f(2,5:8)+ref-1;
elseif k > 2
ls8=[D16(1,is) D16(5,is) D16(3,is) D16(7,is);
   D16(9,is) D16(13,is) D16(11,is) D16(15,is)]+ls2(1);
ls7=[D16(2,is) D16(6,is) D16(4,is) D16(8,is);
   D16(10,is) D16(14,is) D16(12,is) D16(16,is)]+ls2(2);
ls6=[D16(5,is) D16(1,is) D16(7,is) D16(3,is);
   D16(13,is) D16(9,is) D16(15,is) D16(11,is)]+ls2(3);
ls5=[D16(6,is) D16(2,is) D16(8,is) D16(4,is);
   D16(14,is) D16(10,is) D16(16,is) D16(12,is)]+ls2(4);
ls4=[D16(3,is) D16(7,is) D16(1,is) D16(5,is);
   D16(11,is) D16(15,is) D16(9,is) D16(13,is)]+ls1(1);
ls3=[D16(4,is) D16(8,is) D16(2,is) D16(6,is);
   D16(12,is) D16(16,is) D16(10,is) D16(14,is)]+ls1(2);
ls2=[D16(7,is) D16(3,is) D16(5,is) D16(1,is);
   D16(15,is) D16(11,is) D16(13,is) D16(9,is)]+ls1(3);
ls1=[D16(8,is) D16(4,is) D16(6,is) D16(2,is);
   D16(16,is) D16(12,is) D16(14,is) D16(10,is)]+ls1(4);
[ls2 f(k,1:4)]=min([ls8;ls6;ls4;ls2]);
[ls1 f(k,5:8)]=min([ls7;ls5;ls3;ls1]);
end
end
[dmin nef]=min([ls2 ls1]');ne(dp+1)=nef;
for k=dp:-1:1
kk=3*((N-1)*dp+k);
itg=f(k,ne(k+1));
itd(k)=2*rem(ne(k+1)-1,4)+rem(itg-1,2);
ne(k)=2*fix((itg-1)/2)+fix((ne(k+1)-1)/4)+1;
%********** Le poids de Hamming 'ph':
```

```
it1=rem(itd(k),2);it2=rem((itd(k)-it1)/2,2);it3=fix(itd(k)/4);
if it1~=x(kk+2)
ph=ph+1;
end
if it2~=x(kk+4)
ph=ph+1;
end
if it3~=x(kk+3)
ph=ph+1;
end
end
p=0;
for k=1:dp-1
if ne(k)==nex(k)
t1=k;
end
if ne(k+1)==nex(k+1)
t2=k+1;
d=0;
for l=t1:t2
d=d+D16(iss(l),ss(itd(l)+1,ne(l)));
end
if d>=1.3238
p=p+0.5*erfc(sqrt(d)/(2*10.^(-RSB/20)));
end
if d==1.3238
fr=fr+1;
end
end
end
end
pe=2*/fr;
```

Le programme du codeur de la figure 3.3.b à modulation 8-AM-PM (TCM6)

```
clear
RSB=input('Donner la valeur du rapport signal sur bruit en dB ')
load Q8;load q;
lm=60000; dp=2000;
ss=[1 2 5 6 3 4 7 8;5 6 1 2 7 8 3 4;3 4 7 8 1 2 5 6;7 8 3 4 5 6 1 2];
load x;
nex(dp+1)=1;ph=0;nef=1;fr=fr+1;
for N=1:lm/(2*dp)
nex(1)=nex(dp+1);
ref=rem(nef-1,2)+1;
fne=fix((nef-1)/2)+1;
for k=1:dp
kk=2*((N-1)*dp+k);
%********** Le codage:
y1=rem(x(kk+4)+x(kk+1),2);
y2=rem(x(kk+3)+x(kk),2);
```

```
y3=x(kk+2);
nex(k+1)=1+x(kk+4)+2*x(kk+3)+4*rem(nex(k)-1,2);
%********** Le mapping:
iss(k)=1+y3+2*y2+4*y1;
s=q(iss(k));
%********** Les symboles bruités:
r=randn(1,2);
z=s+10.^(-RSB/20)*1.2247*(r(1)+i*r(2));
%********** La décision:
[z is]=min((abs(q-z)));
%********** Le décodage:
if k==1
ls1=[Q8(ss(1,nef),is) Q8(ss(2,nef),is) Q8(ss(3,nef),is) Q8(ss(4,nef),is)];
f(1,4*ref-3:4*ref)=[fne fne fne fne];
elseif k==2
ls4=[Q8(ss(1,4*ref-3),is) Q8(ss(2,4*ref-3),is) Q8(ss(3,4*ref-3),is) Q8(ss(4,4*ref-3),is)]+ls1(1);
ls3=[Q8(ss(1,4*ref-2),is) Q8(ss(2,4*ref-2),is) Q8(ss(3,4*ref-2),is) Q8(ss(4,4*ref-2),is)]+ls1(2);
ls2=[Q8(ss(1,4*ref-1),is) Q8(ss(2,4*ref-1),is) Q8(ss(3,4*ref-1),is) Q8(ss(4,4*ref-1),is)]+ls1(3);
ls1=[Q8(ss(1,4*ref ),is) Q8(ss(2,4*ref ),is) Q8(ss(3,4*ref ),is) Q8(ss(4,4*ref ),is)]+ls1(4);
[ls2 f(2,1:4)]=min([ls4;ls2]);f(2,1:4)=f(2,1:4)+2*(ref-1);
[ls1 f(2,5:8)]=min([ls3;ls1]);f(2,5:8)=f(2,5:8)+2*(ref-1);
elseif k > 2
ls8=[Q8(1,is) Q8(5,is) Q8(3,is) Q8(7,is)]+ls2(1);
ls7=[Q8(2,is) Q8(6,is) Q8(4,is) Q8(8,is)]+ls2(2);
ls6=[Q8(5,is) Q8(1,is) Q8(7,is) Q8(3,is)]+ls2(3);
ls5=[Q8(6,is) Q8(2,is) Q8(8,is) Q8(4,is)]+ls2(4);
ls4=[Q8(3,is) Q8(7,is) Q8(1,is) Q8(5,is)]+ls1(1);
ls3=[Q8(4,is) Q8(8,is) Q8(2,is) Q8(6,is)]+ls1(2);
ls2=[Q8(7,is) Q8(3,is) Q8(5,is) Q8(1,is)]+ls1(3);
ls1=[Q8(8,is) Q8(4,is) Q8(6,is) Q8(2,is)]+ls1(4);
[ls2 f(k,1:4)]=min([ls8;ls6;ls4;ls2]);
[ls1 f(k,5:8)]=min([ls7;ls5;ls3;ls1]);
end
end
[dmin nef]=min([ls2 ls1]');ne(dp+1)=nef;
for k=dp:-1:1
kk=2*((N-1)*dp+k);
itg=f(k,ne(k+1));
itd(k)=rem(ne(k+1)-1,4);
ne(k)=2*itg+fix((ne(k+1)-1)/4)-1;
%********** Le poids de Hamming 'ph':
ph=ph+rem(itd(k),2)+x(kk+4)+fix(itd(k)/2)+x(kk+3)-2*(rem(itd(k),2)*x(kk+4)...
+fix(itd(k)/2)*x(kk+3));
end
p=0;
for k=1:dp-1
if (ne(k)==nex(k) & ne(k+1)~=nex(k+1))
t1=k;
end
if (ne(k+1)~=nex(k+1) & ne(k+2)==nex(k+2))
t2=k+1;
d=0;
for l=t1:t2
```

```
d=d+Q8(iss(l),ss(itd(l)+1,ne(l)));
end
if d>=8
p=p+0.5*erfc(sqrt(d)/(2*1.732*10.^(-RSB/20)));
end
if d==8
fr=fr+1;
end
end
end
end
pe=p/fr;
```

Le programme du codeur de la figure 3.3.e à modulation 16-QAM (TCM7)

```
clear
RSB=input('Donner la valeur du rapport signal sur bruit en dB ')
load Q16;load qq;
lm=60000; dp=2000;
ss=[1 2 5 6 3 4 7 8;9 10 13 14 11 12 15 16;
   5 6 1 2 7 8 3 4;13 14 9 10 15 16 11 12;
   3 4 7 8 1 2 5 6;11 12 15 16 9 10 13 14;
   7 8 3 4 5 6 1 2;15 16 11 12 13 14 9 10];
load x;
nex(dp+1)=1;ph=0;nef=1;fr=0;
for N=1:lm/(3*dp)
fne=2*fix((nef-1)/2);
nex(1)=nex(dp+1);
ref=4*rem(nef-1,2)+1;
fne=2*fix((nef-1)/2);
for k=1:dp
kk=3*((N-1)*dp+k);
%********** Le codage:
y1=x(kk+2);
y2=rem(x(kk+4)+x(kk),2);
y3=rem(x(kk+3)+x(kk-2),2);
y4=x(kk+1);
nex(k+1)=1+x(kk+4)-2*x(kk+3)+4*rem(nex(k)-1,2);
%********** Le mapping:
iss(k)=1+y4+2*y3+4*y2+8*y1;
s=qq(iss(k));
%********** Les symboles bruités:
r=randn(1,2);
z=s+10.^(-RSB/20)*1.5811*(r(1)+i*r(2));
%********** La décision:
[z is]=min(abs(qq-z));
%********** Le décodage:
if k==1
ls1=[Q16(ss(1,nef),is) Q16(ss(3,nef),is) Q16(ss(5,nef),is) Q16(ss(7,nef),is);
   Q16(ss(2,nef),is) Q16(ss(4,nef),is) Q16(ss(6,nef),is) Q16(ss(8,nef),is)];
[ls1 f(1,ref:ref+3)]=min(ls1);
```

```
f(1,ref:ref+3)=f(1,ref:ref+3)-fne;
elseif k==2
ls4=[Q16(ss(1,ref),is) Q16(ss(3,ref),is) Q16(ss(5,ref),is) Q16(ss(7,ref),is);
    Q16(ss(2,ref),is) Q16(ss(4,ref),is) Q16(ss(6,ref),is) Q16(ss(8,ref),is)]+ls1(1);
ls3=[Q16(ss(1,ref+1),is) Q16(ss(3,ref+1),is) Q16(ss(5,ref+1),is) Q16(ss(7,ref+1),is);
    Q16(ss(2,ref+1),is) Q16(ss(4,ref+1),is) Q16(ss(6,ref+1),is) Q16(ss(8,ref+1),is)]+ls1(2);
ls2=[Q16(ss(1,ref+2),is) Q16(ss(3,ref+2),is) Q16(ss(5,ref+2),is) Q16(ss(7,ref+2),is);
    Q16(ss(2,ref+2),is) Q16(ss(4,ref+2),is) Q16(ss(6,ref+2),is) Q16(ss(8,ref+2),is)]+ls1(3);
ls1=[Q16(ss(1,ref+3),is) Q16(ss(3,ref+3),is) Q16(ss(5,ref+3),is) Q16(ss(7,ref+3),is);
    Q16(ss(2,ref+3),is) Q16(ss(4,ref+3),is) Q16(ss(6,ref+3),is) Q16(ss(8,ref+3),is)]+ls1(4);
[ls2 f(2,1:4)]=min([ls4;ls2]);f(2,1:4)=f(2,1:4)+ref-1;
[ls1 f(2,5:8)]=min([ls3;ls1]);f(2,5:8)=f(2,5:8)+ref-1;
elseif k > 2
ls8=[Q16(1,is) Q16(5,is) Q16(3,is) Q16(7,is);
    Q16(9,is) Q16(13,is) Q16(11,is) Q16(15,is)]+ls2(1);
ls7=[Q16(2,is) Q16(6,is) Q16(4,is) Q16(8,is);
    Q16(10,is) Q16(14,is) Q16(12,is) Q16(16,is)]+ls2(2);
ls6=[Q16(5,is) Q16(1,is) Q16(7,is) Q16(3,is);
    Q16(13,is) Q16(9,is) Q16(15,is) Q16(11,is)]+ls2(3);
ls5=[Q16(6,is) Q16(2,is) Q16(8,is) Q16(4,is);
    Q16(14,is) Q16(10,is) Q16(16,is) Q16(12,is)]+ls2(4);
ls4=[Q16(3,is) Q16(7,is) Q16(1,is) Q16(5,is);
    Q16(11,is) Q16(15,is) Q16(9,is) Q16(13,is)]+ls1(1);
ls3=[Q16(4,is) Q16(8,is) Q16(2,is) Q16(6,is);
    Q16(12,is) Q16(16,is) Q16(10,is) Q16(14,is)]+ls1(2);
ls2=[Q16(7,is) Q16(3,is) Q16(5,is) Q16(1,is);
    Q16(15,is) Q16(11,is) Q16(13,is) Q16(9,is)]+ls1(3);
ls1=[Q16(8,is) Q16(4,is) Q16(6,is) Q16(2,is);
    Q16(16,is) Q16(12,is) Q16(14,is) Q16(10,is)]+ls1(4);
[ls2 f(k,1:4)]=min([ls8;ls6;ls4;ls2]);
[ls1 f(k,5:8)]=min([ls7;ls5;ls3;ls1]);
end
end
[dmin nef]=min([ls2 ls1]');ne(dp+1)=nef;
for k=dp:-1:1
kk=3*((N-1)*dp+k);
itg=f(k,ne(k+1));
itd(k)=2*rem(ne(k+1)-1,4)+rem(itg-1,2);
ne(k)=2*fix((itg-1)/2)+fix((ne(k+1)-1)/4)+1;
%********** Le poids de Hamming 'ph':
it1=rem(itd(k),2);it2=rem((itd(k)-it1)/2,2);it3=fix(itd(k)/4);
if it1~=x(kk+2)
ph=ph+1;
end
if it2~=x(kk+4)
ph=ph+1;
end
if it3~=x(kk+3)
ph=ph+1;
end
end
p=0;
for k=1:dp-1
```

```
if ne(k)==nex(k)
t1=k;
end
if ne(k+1)==nex(k+1)
t2=k+1;
d=0;
for l=t1:t2
d=d+Q16(iss(l),ss(itd(l)+1,ne(l)));
end
if d>=10
p=p+0.5*erfc(sqrt(d)/(2*2.236*10.^(-RSB/20)));
end
if d==10
fr=fr+1;
end
end
end
end
pe=2*p/fr;
```

Remarques

- *D4 : la matrice distance de la modulation 4-PSK.*
- *D8 : la matrice distance de la modulation 8-PSK.*
- *D16: la matrice distance de la modulation 16-PSK.*
- *Q8 : la matrice distance de la modulation 8-AM-PM.*
- *Q16: la matrice distance de la modulation 16-QAM.*
- *q : la matrice qui contient les points de la 8-AM-PM, par ordre de numérotation croissante.*
- *qq : la matrice qui contient les points de la 16-QAM, par ordre de numérotation croissante.*
- *La formule générale de*

itg

$itg=f(k,ne(k+1))$ où f est la matrice qui contient tous les survivants.

itd

$itd=2^{m-m'} \times rem(ne(k-1)-1,2^{m'})+ rem(itg(k),2^{m-m'})$.

ne

$ne=2^{v-m'} \times fix((itg(k)-1)/ 2^{m-m'}) + fix((ne(k+1)-1)/2^{m'})+1.$

fne

$fne=2^{m-m'} \times fix((nef-1)/ 2^{v-m'})$

ref

$ref=2^{m'} \times rem(nef-1, 2^{v-m'}) + 1.$