Algorithmes de décodage pour les systèmes multi-antennes à complexité réduite

Rym Ouertani

Thèse

présentée pour obtenir le grade de docteur
de TELECOM ParisTech
Spécialité : Électronique et Communications

Rym OUERTANI

Algorithmes de décodage pour les systèmes multi-antennes à complexité réduite

Soutenue le 26 Novembre 2009 devant le jury composé de

Pr. Hikmet Sari	**Président**
Pr. Joseph-Jean Boutros	**Rapporteurs**
Pr. Ramesh Pyndiah	
Pr. Ammar Bouallègue	**Examinateurs**
Dr. Nicolas Gresset	
Dr. Ghaya Rekaya-Ben Othman	**Directeurs de thèse**
Pr. Jean-Claude Belfiore	

A mes parents

Remerciements

Cette thèse a été effectuée au sein du département Communications et Électronique de l'école TELECOM ParisTech. Je tiens à remercier toute personne qui m'a aidée et soutenue pour mener à bien ce travail.

J'exprime tout d'abord ma gratitude à Monsieur Hikmet Sari, Professeur à Supélec, pour m'avoir fait l'honneur de présider le jury de ma thèse.

Je tiens également à remercier Monsieur Joseph-Jean Boutros, Professeur à l'université Texas AM University à Qatar, pour avoir accepté d'être rapporteur de ce travail et pour m'avoir fait part de ses remarques constructives ainsi que pour ses discussions fructueuses pendant la soutenance. J'exprime ma reconnaissance à Monsieur Ramesh Pyndiah, Professeur à Télécom Bretagne, et pour lequel j'exprime ma profonde estime. Je le remerice pour sa lecture attentive du mémoire et ses corrections du manuscrit.

Toute ma reconnaissance va également à Monsieur Ammar Bouallègue, Professeur à l'Ecole Nationale d'Ingénieurs de Tunis, pour m'avoir fait l'honneur d'être membre du jury de ma thèse, et qui m'a toujours soutenue et encouragée durant mes études universitaires. Je remercie Monsieur Nicolas Gresset, Ingénieur de recherche à Mitsubushi Electric RD à Rennes, pour avoir accepté de juger mon travail, pour ses conseils avant et pendant ma thèse et pour sa très grande gentillesse.

Mes plus vifs remerciements et ma plus grande gratitude vont à mes directeurs de thèse, Madame Ghaya Rekaya Ben-Othman, Maître de conférences à Télécom ParisTech et Monsieur Jean-Claude Belfiore, Professeur à Télécom ParisTech. Je ne pense pas avoir parvenu à accomplir ce travail sans les conseils, l'encadrement et les encouragements de Ghaya. Elle a toujours été disponible et à l'écoute à tout moment de la thèse. Je voudrais lui exprimer ma profonde reconnaissance non seulement pour son encadrement et toutes les connaissances scientifiques qu'elle m'a apprises mais également pour son soutien permanent, sa très grande sympathie et pour son amitié. Merci Ghaya pour ton aide précieuse et pour la confiance que tu m'as accordée. Je tiens à exprimer également ma profonde reconnaissance à Jean-Claude pour sa collaboration fructueuse, sa riche expérience et son immense compétence. Je te remercie Jean-Claude pour ta gentillesse et ta bonne humeur.

Je remercie également Professeur Henri Maitre, Directeur de la Recherche de Télécom ParisTech et Directeur adjoint de l'EDITE de Paris pour son aide précieuse tout au long de ma thèse.

Un grand merci à tous les membres du département Communications et Électronique pour leur écoute et leur collaboration, en particulier le responsable du département Monsieur Bruno

Thedrez et Mesdmaes Florence Besnard, Danielle Childz, Chantal Cadiat, Marie Baquero et Fabienne Lassausaie pour m'avoir facilité les tâches administratives, toujours avec un grand sourire et sympathie.

Je ne manquerais pas de remercier mes collègues et amis doctorants et postdocs qui ont le plus partagé mon quotidien à Télécom ParisTech et avec qui j'ai partagé de très bons moments : Ihsan Fsaifes, Anne-Laure Deleuze, Abdellatif Salah, Charlotte Hucher, Sami Mekki, Yang Liu, Lina Mroueh, Maya Badr, Eric Boutton, Juan Petit, Qing Xu, Ghassan M. Kraidy, Mireille Sarkiss, Fatma Kharrat et Laura Luzzi.

Je ne pourrais pas oublier d'exprimer toute ma gratitude et toute mon affection à mes parents pour leur amour et pour l'éducation qu'ils m'ont donnée ainsi qu'à mes frères pour leur soutien sans limites et leurs encouragements tout au long de mes études.

Résumé

Durant ces dernières années, un grand intérêt a été accordé aux systèmes de communication sans fil ayant plusieurs antennes en émission et en réception. Les codes espace-temps permettent d'exploiter tous les degrés de liberté de tels systèmes. Toutefois, le décodage de ces codes présente une grande complexité qui croit en fonction du nombre d'antennes déployées et de la taille de la constellation utilisée.

Nous proposons un nouveau décodeur, appelé SB-Stack (Spherical Bound-Stack decoder) basé sur un algorithme de recherche dans l'arbre. Ce décodeur combine la stratégie de recherche du décodeur séquentiel Stack (dit également décodeur à pile) et la région de recherche du décodeur par sphères. Nous montrons que ce décodeur présente une complexité moindre par rapport à tous les décodeurs existants tout en offrant des performances optimales. Une version paramétrée de ce décodeur est aussi proposée, offrant une multitude de performances allant du ZF-DFE au ML avec des complexités croissantes, ainsi plusieurs compromis performances-complexités sont obtenus.

Comme pour tous les systèmes de communication, les codes espace-temps pour les systèmes à antennes multiples peuvent être concaténés avec des codes correcteurs d'erreurs. Généralement, ces derniers sont décodés par des décodeurs à entrées et sorties souples. Ainsi, nous avons besoin de sorties souples fournies par le décodeur espace-temps qui seront utilisées comme entrées par les premiers décodeurs. Nous proposons alors une version modifiée du décodeur SB-Stack délivrant des sorties souples sous forme de taux de vraisemblance logarithmiques (Log-Likelihood Ratio - LLR).

Pour la mise en oeuvre pratique des décodeurs, il est important d'avoir une complexité faible mais avoir également une complexité constante est indispensable dans certaines applications. Nous proposons alors un décodeur adaptatif qui permet de sélectionner, parmi plusieurs algorithmes de décodage, celui qui est le plus adéquat selon la qualité du canal de transmission et la qualité de service souhaitée. Nous présentons une implémentation pratique du décodage adaptatif utilisant le décodeur SB-Stack paramétré.

Le décodage des codes espace-temps peut être amélioré en le précédant par une phase de pré-traitement. En sortie de cette phase, la matrice du canal équivalente est mieux conditionnée ce qui permet de réduire la complexité d'un décodeur optimal et d'améliorer les performances d'un décodeur sous-optimal. Nous présentons et nous étudions alors les performances d'une chaine complète de décodage utilisant diverses techniques de pré-traitement combinées avec les décodeurs espace-temps étudiés précédemment.

Abstract

In recent years, a great interest has been accorded to Multiple-Input Multiple-Output systems due to the large capacity they offer. Space-time codes are used to give a coding gain. A lattice representation of multi-antenna systems has been proposed, which make it possible to decode such systems using lattice decoders. Several decoders exist in the literature. Unfortunately, their complexity increases drastically with the lattice dimension and the constellation size.

In our work, we propose a new sequential algorithm which combines the stack decoder search strategy and the sphere decoder search region. The proposed decoder that we call the Spherical-Bound-Stack decoder (SB-Stack) can be used to resolve lattice and large size constellations decoding. Furthermore, it outperforms the existing decoders in term of complexity while achieving ML performance. By introducing a bias parameter, the SB-Stack gives a range of performances from ML to ZF-DFE with proportional complexities. So, different performance/complexity trade-offs could be obtained.

The proposed algorithm is given to decode space-time codes. However, when a channel coding is integrated to the transmission scheme, soft decoding becomes necessary. The SB-Stack is then extended to support soft-output detection over linear channels. A straightforward idea was to exploit internal nodes still stored in the stack at the end of hard decoding process to calculate LLR. We show that the potential gain of such method is rather large then classical soft decoders.

For practical implementation, the decoding complexity is a critical point, but also the big variation of the complexity between low and high SNR (Signal to Noise Ratio) is an additional problem because of the variable decoding time. We propose an adaptive decoder based on the SB-Stack. This one switches between optimal and sub-optimal decoders according to the channel realization and the system specifications. This decoder has an almost constant complexity while keeping good performance.

Lattice reduction is used to accelerate the decoding of infinite lattice. Using the MMSE-GDFE, it becomes possible to apply lattice reduction when finite constellations are considered. Therefore, interesting results are obtained when considering MIMO schemes combining the lattice reduction, the MMSE-GDFE process and the sequential decoders given previously.

Table des matières

Table des figures

Liste des tableaux

Liste des abréviations

AMC	Adaptive Modulation and Coding
AWGN	Additive White Gaussian Noise
BB	Branch and Bound
BeFS	Best First Search
BLAST	Bell Labs Layered Space-Time
BrFS	Breadth First Search
DAST	Diagonal Algebraic Space-Time
D-BLAST	Diagonal-BLAST
DFE	Decision Feedback Equalization
DFS	Depth First Search
GC	Golden Code
GDFE	Generalized Decision Feedback Equalizer
HSDPA	High Speed Downlink Packet Access
LDC	Linear Dispersion Code
LLR	Log Likelihood Ratio
LSD	List Sphere Decoder
MIMO	Multiple Input Multiple Output
ML	Maximum Likelihood
MMSE	Minimum Mean Square Error
QAM	Quadrature Amplitude Modulation
QoS	Quality of Service
QPSK	Quadrature Phase Shift Keying
QRD	QR Decomposition
RSB	Rapport Signal à Bruit
SB-Stack	Spherical Bound Stack decoder
SD	Sphere Decoder
SE	Schnorr-Euchner
SINR	Signal to Interference plus Noise Ratio
SISO	Single Input Single Output
SSD	Shifted Sphere Decoder
ST	Space-Time
STBC	Space Time Block Code
SVD	Singular Value Decomposition
TAST	Threaded Algebraic Space-Time
TEB	Taux d'Erreur par Bit
u.c	Utilisation canal
V-BLAST	Vertical-BLAST
ZF	Zero Forcing

Liste des notations

$\mathbb{N}$	Ensemble des entiers naturels
$\mathbb{Z}$	Ensemble des entiers relatifs
$\mathbb{R}$	Ensemble des réels
$\mathbb{C}$	Ensemble des nombres complexes
$\mathbb{R}^n$	Espace euclidien réel de dimension n
$\boldsymbol{M^c}$	Matrice à éléments complexes
$\boldsymbol{M}$	Matrice à éléments réels
$\boldsymbol{M_{m\times n}}$	Matrice à m lignes et n colonnes
$\boldsymbol{s^c}$	Vecteur complexe
x^*	conjugué de x
$\Re(x)$	Partie réelle de x
$\Im(x)$	Partie imaginaire de x
$sqrt(x)$	Racine carrée de x
$\|x\|$	Norme euclidienne du vecteur x
$\boldsymbol{M^T}$	Transposée d'une matrice
$\boldsymbol{M^\dagger}$	Transposée conjuguée d'une matrice
$\boldsymbol{I}$	Matrice identité

Introduction

L'intégration de nouvelles technologies telles que les applications multi-média dans la nouvelle génération des systèmes sans fil exigent des débits toujours plus élevés et des qualités de service plus importantes. Dans un système de communication classique utilisant une antenne à l'émission et une antenne à la réception (Single Input Single Output - SISO), la capacité réalisable est limitée par la puissance d'émission et la bande passante disponible. Afin de faire face à ces limites, l'utilisation de plusieurs antennes à chaque côté (Multiple Input Multiple Output - MIMO) a été suggérée. Les systèmes multi-antennes permettent théoriquement d'accroitre la capacité des liens de communication sans fil par rapport aux systèmes SISO. En effet, la capacité théorique du canal MIMO avec n_t antennes à l'émission et n_r antennes à la réception croit linéairement avec $\min(n_t, n_r)$. C'est pourquoi, nous assistons actuellement à un rapide développement de cette technologie avec des applications déjà envisagées dans les futurs standards de télécommunications tels que les réseaux locaux sans fil IEEE 802.11n pour les applications vidéo et les réseaux d'accès sans fil large bande IEEE 802.20 intégrant la technologie MIMO-OFDM.

Le codage espace-temps (ST) permet d'exploiter tous les degrés de liberté des systèmes MIMO, en introduisant une dépendance entre le domaine temporel et spatial, dans le but d'apporter une diversité spatiale et un gain de codage. Un décodage à Maximum de Vraisemblance permet l'obtention de performances optimales, cependant, il présente une grande complexité. Nous considérons les codes ST à dispersion linéaire. La représentation en réseaux de points de ces codes ST permet leur décodage par les décodeurs de réseaux de points.

Plusieurs algorithmes de décodage offrant les performances optimales ont été proposés dans la littérature. Nous distinguons en particulier les décodeurs basés sur la stratégie de recherche dans un arbre tels que le décodeur par sphères, l'algorithme de Schnorr-Euchner et les décodeurs séquentiels tels que le décodeur Fano et le décodeur Stack. Bien que ces derniers présentent une complexité plus faible par rapport au décodeur exhaustif, leur complexité est croissante en fonction du nombre d'antennes déployées et de la taille de la constellation utilisée, ce qui rend leur utilisation très limitée.

Dans ce travail, nous proposons dans un premier temps un nouvel algorithme de décodage séquentiel, appelé SB-Stack (Spherical Bound-Stack decoder). Celui-ci est basé à la fois sur le décodeur par sphères et le décodeur Stack. Le premier utilise la stratégie de recherche DeFS (Depth-first-search) dans une région finie du réseau qui consiste en une sphère de rayon finie centrée sur le point reçu. Le deuxième utilise une stratégie de recherche plus optimale, BeFS (Best-first-search), cependant, la recherche se fait dans toute la constellation. Le décodeur SB-Stack combine les avantages de l'un et l'autre des deux décodeurs. En fait, il utilise la stratégie de recherche du Stack et la région de recherche définie par la sphère centrée sur le point reçu. Par conséquent, le SB-Stack permet d'offrir de meilleures complexités que les décodeurs originaux

sans sacrifier les performances optimales. En outre, contrairement au décodeur Stack original, nous démontrons que le SB-Stack peut être utilisé pour décoder les constellations finies mais aussi les réseaux de points infinis.

Dans le but de réduire encore la complexité, une version paramétrée du décodeur SB-Stack est proposée. L'introduction d'un paramètre, appelé biais, permet de donner une multitude de performances allant du ZF-DFE au ML avec des complexités d'autant plus faibles que les performances sont sous-optimales. Plusieurs compromis performances-complexités sont alors obtenus.

Dans une chaine de transmission réelle, un code correcteur d'erreurs comme un code convolutif ou un turbo code est toujours placé en amont du codage espace-temps. Afin d'optimiser au mieux les performances globales de la chaine de transmission, le décodeur MIMO à sorties binaires (dures) doit être remplacé par un décodeur à sorties pondérées (souples). Celles-ci seront utilisées comme entrées par les décodeurs des codes correcteurs d'erreurs. Nous avons alors proposé une version modifiée du décodeur SB-Stack pour délivrer des sorties souples sous forme de taux de vraisemblance logarithmiques (Log-Likelihood Ratio - LLR).

Les performances mais aussi la complexité sont les deux critères à prendre en compte dans le choix d'un décodeur. En pratique, il est important dans plusieurs applications, telles que les applications temps réel, d'avoir un décodeur ayant une complexité et un temps de décodage constants. Afin de répondre à ce besoin, nous avons mis en place des techniques d'adaptation permettant de sélectionner le décodeur le plus approprié selon plusieurs critères de sélection. En général, les systèmes MIMO sont sensibles aux conditions radio, nous proposons alors un algorithme de décodage adaptatif permettant de choisir le décodeur adéquat en fonction de la qualité instantanée du canal de transmission ainsi que de la qualité de service souhaitée. Nous présentons une implémentation pratique du décodage adaptatif utilisant le décodeur SB-Stack paramétré.

Les performances des systèmes MIMO dépendent étroitement du décodeur utilisé. L'ajout d'une phase de pré-traitement peut améliorer le décodage, soit en diminuant la complexité de la phase de recherche, soit en améliorant les performances sous-optimales. Le but de cette phase est d'avoir une nouvelle matrice équivalente du canal qui soit mieux conditionnée. Il existe deux types de pré-traitement : le pré-traitement à gauche qui consiste à l'application du MMSE-GDFE et le pré-traitement à droite qui consiste en la réduction du réseau de points. L'utilisation du pré-traitement dans la chaine de transmission MIMO permet par conséquent de réduire la complexité d'un décodeur optimal et d'améliorer les performances d'un décodeur sous-optimal. Nous présentons et nous étudions les performances d'une chaine complète de décodage utilisant diverses techniques de pré-traitement combinées avec les décodeurs espace-temps étudiés précédemment.

Organisation de la thèse :

Dans le premier chapitre, nous allons donner le cadre de notre travail et dresser le socle théorique des chapitres suivants. Nous définissons dans un premier temps les différents paramètres d'une chaine de transmission MIMO et nous détaillons les différents modules qui la constituent. Ensuite, nous définissons les réseaux de points et nous donnons quelques outils mathématiques qui nous seront utiles par la suite. Nous présentons également la représentation en réseau de points d'un système MIMO employant un codage ST à dispersion linéaire.

Dans le deuxième chapitre, nous allons donner un état de l'art des différents décodeurs utilisés pour les systèmes MIMO. Nous distinguons les décodeurs sous-optimaux tels que les décodeurs linéaires et les décodeurs à retour de décision. En un deuxième temps, nous présentons les décodeurs optimaux tels que les décodeurs par sphère et Schnorr-Euchner et les décodeurs séquentiels Fano et Stack. Nous donnons aussi la version paramétrée des décodeurs séquentiels qui permet d'avoir une complexité moindre au prix des performances sous-optimales.

Dans le troisième chapitre, nous introduisons le nouvel algorithme de décodage séquentiel, le SB-Stack que nous proposons. Deux approches seront présentées : le décodeur SB-Stack à sorties dures et le décodeur SB-Stack à sorties souples. Nous montrons que le premier décodeur offre des performances optimales avec des complexités inférieures à tous les décodeurs existants et la deuxième version souple permet d'offrir un bon compromis performances-complexités par comparaison aux décodeurs à sorties souples connus. Nous illustrons les performances des décodeurs à travers les résultats de simulation.

Dans le quatrième chapitre, le décodeur adaptatif proposé sera détaillé. Nous commençons par présenter les méthodes adaptatives qui existent dans la littérature. Par la suite, nous détaillons les trois critères de sélection proposés afin de choisir le décodeur adéquat. Nous donnons finalement les performances du décodeur adaptatif implémenté en utilisant le décodeur SB-Stack.

Le cinquième chapitre sera consacré à l'étude des techniques de pré-traitement. Nous allons proposer d'appliquer les principales techniques utilisées dans la littérature dans notre chaine de transmission MIMO, à savoir le MMSE-GDFE comme pré-traitement à gauche et la réduction LLL et Seysen comme pré-traitement à droite. Nous cloturons ce chapitre par présenter et étudier les performances de la chaine complète de décodage combinant les diverses techniques de pré-traitement avec les différents décodeurs proposés.

Chapitre 1

Préliminaires

1.1 Introduction

Dans ce chapitre, nous définissons le cadre de notre travail et nous exposons les notions théoriques que nous utiliserons dans les chapitres suivants.

Nous nous intéressons aux systèmes de transmission à antennes multiples. Nous rappelons dans une première partie les caractéristiques ainsi que le modèle général de la chaine de transmission. Nous présentons deux schémas de transmission : un système sans codage utilisant un simple multiplexage spatial et un système avec codage employant un codage espace-temps.

Les réseaux de points sont les principaux outils nécessaires pour la représentation et l'étude des systèmes à antennes multiples. Nous présentons par la suite les principales définitions et caractéristiques relatives aux réseaux de points.

1.2 Généralités sur les transmissions MIMO

1.2.1 Le canal radio

Dans un canal radio, le signal transmis peut suivre plusieurs acheminements avant d'atteindre la destination. Ces acheminements ou trajets multiples sont causés par différents phénomènes telles que la diffraction, la réflexion, la dispersion, etc. De plus sur chaque trajet, le signal subit des évanouissements dont l'amplitude dépend du milieu de propagation, la présence d'obstacles ou encore de la distance parcourue entre l'émetteur et le récepteur.

Nous écrivons le signal reçu sous la forme :

$$y = h \cdot x + w \tag{1.1}$$

- h représente l'évanouissement causé par le canal de transmission.
- x est le signal transmis.
- w est le bruit modélisé par un bruit blanc additif Gaussien (Additive White Gaussian Noise AWGN) de moyenne nulle et de variance $\sigma_w^2 = N_0/2$ par dimension réelle. Le bruit résulte du bruit thermique des composants du système ainsi que de toute forme de perturbation telles que les interférences multi-utilisateurs.

Nous supposons les trajets suffisamment séparés pour que les évanouissements soient considérés comme des variables aléatoires indépendantes. D'après le théorème central limite, en présence d'un grand nombre d'évanouissements le canal peut être modélisé par un processus aléatoire Gaussien complexe [1]. Si la moyenne des évanouissements est nulle, l'enveloppe suit une loi de Rayleigh et le canal est dit **canal de Rayleigh**. C'est le modèle de canal que nous allons considérer. Si la moyenne des évanouissements n'est pas nulle, le canal est dit **canal de Rice**.

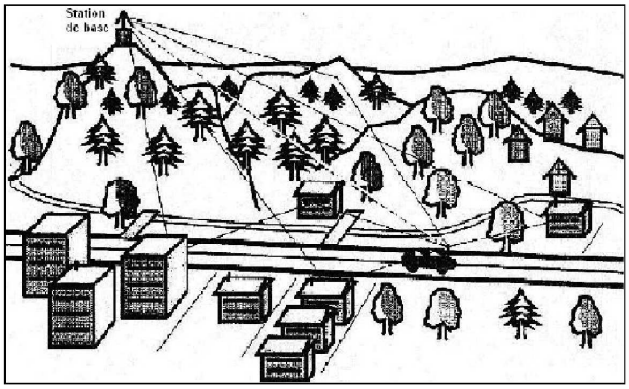


FIGURE 1.1 – Schéma de transmission sur un canal radio mobile

Le canal h est à entrées complexes Gaussiennes de moyenne nulle et de variance $\sigma^2 = 0.5$ par dimension réelle et son module $|h|$ suit une loi de Rayleigh dont la fonction de densité de probabilité est donnée par :

$$p(u) = \frac{u}{\sigma^2} e^{-\frac{u^2}{2\sigma^2}} \quad , \quad u \geq 0 \tag{1.2}$$

Nous appelons temps de cohérence du canal T_c l'intervalle de temps pendant lequel h reste constant.

- Lorsque le temps de cohérence est suffisamment petit, *i.e*, l'évanouissement varie d'un temps symbole à un autre, le canal est dit **ergodique** ($T_c = T_S$).
- Lorsque le temps de cohérence est plus grand que le temps de transmission d'une trame, *i.e*, le canal reste constant pendant la transmission d'une trame ($T_c = T_f$), le canal est dit **quasi-statique**.
- Le canal est **à évanouissement par blocs** s'il reste constant pendant la transmission de N trames ($T_c = N \cdot T_f$).

L'évanouissement et les interférences liés aux obstacles et aux multi-trajets représentent la principale cause de perte d'information pour les transmissions sur le canal radio. La transmission du signal sur un seul lien peut s'avérer insuffisante pour retrouver l'information à la réception. Pour faire face à ce problème, il serait intéressant d'envoyer plusieurs répliques du signal sur différents liens subissant des évanouissements indépendants : c'est la notion de diversité. Plusieurs systèmes utilisant cette technique ont été mis en place. Nous retrouvons entre autres les

systèmes à antennes multiples à l'émission et à la réception. Dans ce qui suit, nous donnons le modèle général des systèmes multi-antennes et nous présentons la chaine de transmission que nous considérons tout le long de notre travail.

1.2.2 Système de transmission sans fil MIMO

Pour un système MIMO, nous considérons la chaine de transmission avec n_t antennes à l'émission et n_r antennes à la réception présentée dans la figure 1.2.

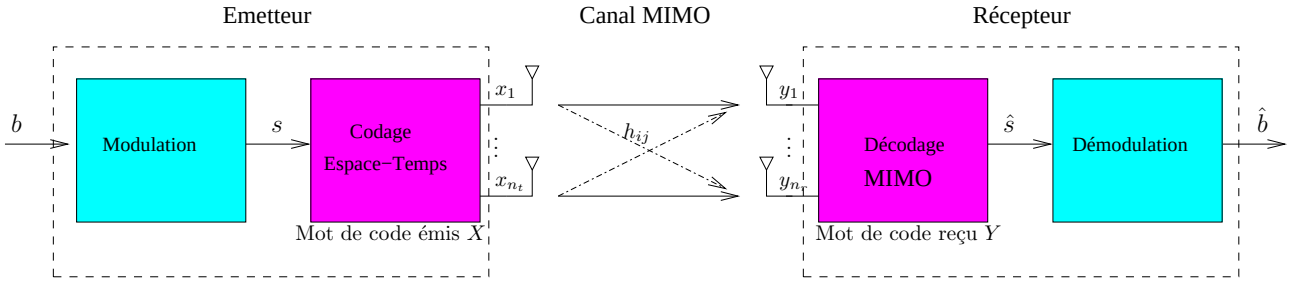


Figure 1.2 – Schéma de transmission MIMO

Une chaine de communication représente l'ensemble des traitements reliant une source (délivrant le message à transmettre) à un destinataire. Les trois éléments de base sont l'émetteur, le canal de transmission et le récepteur. L'émetteur convertit, sous une forme adaptée au canal, le flux d'information fourni par la source. Le récepteur effectue l'opération inverse et fournit le message au destinataire. Le schéma de transmission que nous considérons est un système à plusieurs antennes à l'émission et plusieurs antennes à la réception dans le cas d'une transmission sans fil.

Dans ce qui suit, nous allons décrire les différents modules de la chaine de transmission MIMO.

L'émetteur

A l'émission, une source délivre une séquence de bits d'information $\boldsymbol{b}$. Nous considérons ici une chaine de transmission simplifiée ne comportant pas de codage source ni de codage canal et nous supposons simplement que la séquence de bits d'informations est $i.i.d$ (indépendants identiquement distribués) sur $[0, 1]$. Ces bits sont convertis ensuite en une séquence de symboles complexes modulés, notée $\boldsymbol{s^c}$, tirés dans la constellation $\mathcal{C}_S$. Un symbole d'information est représenté par l'écriture complexe :

$$s_k^c = \Re\left(s_k^c\right) + j\Im\left(s_k^c\right) \tag{1.3}$$

Ces symboles d'information peuvent être pris dans un alphabet infini. On dit dans ce cas qu'ils appartiennent à un **réseau de points**, tel que $\mathbb{Z}[i]$ où $\Re\left(s_k^c\right)$ et $\Im\left(s_k^c\right)$ prennent leurs valeurs dans $\mathbb{Z}$.

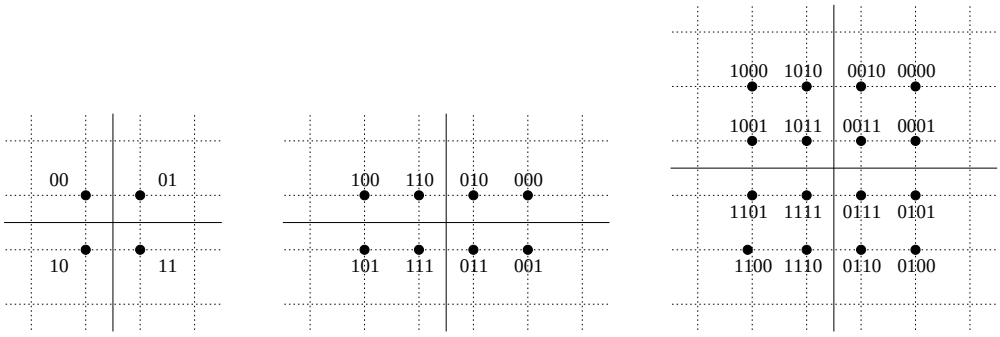


FIGURE 1.3 – Exemples de constellations q-QAM

Pour les schémas pratiques de transmission MIMO, les constellations les plus fréquemment utilisées sont les **q-QAM** (Modulation d'Amplitude en Quadrature) [2]. Une constellation QAM est un sous-ensemble fini de $\mathbb{Z}[i]$, obtenue en considérant des points équidistants contenus dans une région donnée de $\mathbb{Z}[i]$. Dans ce cas, $\Re(s_k^c)$ et $\Im(s_k^c)$ prennent leurs valeurs dans l'alphabet fini $\left\{\pm 1, \pm 3, \ldots, \pm\sqrt{q}-1\right\}$ donnant naissance à une modulation possédant q états ou symboles complexes.

Quelques exemples de constellations q-QAM, avec $q = 4, 8$ et 16 sont représentés dans la figure 1.3. La distance minimale entre deux points adjacents de la constellation est égale à 2. L'énergie moyenne d'un symbole de la modulation q-QAM est égale à :

$$E_s = \frac{2}{3}(q-1)$$

A la sortie du modulateur, les symboles complexes sont groupés par blocs de K symboles puis codés. $\boldsymbol{s^c} = [s_1^c, s_2^c, \ldots, s_K^c]^T$ de dimension $K \times 1$ est le vecteur symboles d'information.

En pratique, le codage associé aux systèmes MIMO est le codage espace-temps (Space-Time - ST). Le principe du codage espace-temps est d'émettre des symboles différents sur chacune des antennes pour augmenter le débit de transmission et améliorer la robustesse du lien [3].

Le codeur ST réalise l'application unique du vecteur $\boldsymbol{s^c}$ vers la matrice mot de code complexe $\boldsymbol{X^c}$ de dimension $n_t \times T$:

$$\boldsymbol{X^c} = \begin{bmatrix} x_{11}^c & \cdots & x_{1T}^c \\ \vdots & \ddots & \vdots \\ x_{n_t 1}^c & \cdots & x_{n_t T}^c \end{bmatrix} \tag{1.4}$$

où T est la longueur temporelle du code, c'est le nombre d'intervalles élémentaires pendant lesquels les symboles sont transmis avec cette matrice mot de code. Chaque composante de la matrice appartient à la constellation C_X qui peut être identique ou différente de la constellation des symboles d'information.

Puisque K symboles sont transmis sur chaque antenne pendant un temps T, le rendement de la transmission d'un code MIMO, donné en symboles par utilisation canal (u.c), est égal à :

$$R = \frac{K}{T}\ symboles/u.c$$

Le canal MIMO

Le signal codé est envoyé sur les différentes antennes émettrices. Chaque flux traverse un trajet différent et subit un évanouissement indépendant. D'après (1.1), le signal reçu sur l'antenne i à l'instant t est une combinaison linéaire des signaux issus des n_t antennes émettrices à laquelle s'ajoute le bruit :

$$y_{it}^c = \sum_{j=1}^{n_t} h_{ij}^c \cdot x_{jt}^c + w_{it}^c \tag{1.5}$$

où h_{ij}^c est le coefficient complexe d'évanouissement introduit par le canal entre l'antenne émettrice j et l'antenne réceptrice i à l'instant t et w_{it}^c est le bruit mesuré à l'antenne i à l'instant t.

Nous considérons le modèle du canal **quasi-statique**. La matrice globale des coefficients du canal MIMO, de dimension alors $n_r \times n_t$, est donnée par :

$$\boldsymbol{H^c} = \begin{bmatrix} h_{11}^c & \cdots & h_{1n_t}^c \\ \vdots & \ddots & \vdots \\ h_{n_r1}^c & \cdots & h_{n_rn_t}^c \end{bmatrix} \tag{1.6}$$

A partir de l'équation (1.5), le signal total reçu peut s'écrire sous la forme matricielle suivante :

$$\boldsymbol{Y_{n_r \times T}^c} = \boldsymbol{H_{n_r \times n_t}^c} \cdot \boldsymbol{X_{n_t \times T}^c} + \boldsymbol{W_{n_r \times T}^c} \tag{1.7}$$

où $\boldsymbol{Y^c}$ est la matrice à éléments complexes des signaux reçus sur toutes les antennes et $\boldsymbol{W^c}$ est la matrice de bruit additif blanc Gaussien.

Il existe deux grandes classes de systèmes de transmission suivant la connaissance apriori du canal de transmission au niveau du récepteur. On parle de **système cohérent**, si les coefficients du canal sont supposés être parfaitement connus au niveau du récepteur. Si les coefficients sont inconnus par le récepteur, deux cas sont possibles. Le premier correspond au **système non cohérent** qui utilise des techniques d'estimation des coefficients du canal de transmission, comme l'envoi d'une séquence d'apprentissage. Le deuxième correspond au **système différentiel** qui décode le signal reçu sans connaissance du canal.

Dans ce travail, nous allons considérer **les systèmes cohérents**.

Le récepteur

Sous l'hypothèse d'une parfaite connaissance du canal, le décodeur d'un code espace-temps a pour but de retrouver une estimation du mot de code émis $\hat{\boldsymbol{X}}$. Le décodage permet de minimiser la probabilité d'erreurs par paire pour une réalisation du canal $\boldsymbol{H^c}$, c'est-à-dire, la probabilité que le récepteur décode $\hat{\boldsymbol{X}}$ alors que $\boldsymbol{X}$ a été transmis $Pe = P\left\{\hat{\boldsymbol{X}} \neq \boldsymbol{X}/\boldsymbol{y^c}, \boldsymbol{H^c}\right\}$ [4].

Pour cela, différentes règles de décodage peuvent être considérées. Nous allons détailler cette partie dans le chapitre suivant. Une fois un mot de code $\hat{\boldsymbol{X}}$ est trouvé, la récupération des symboles d'information $\hat{\boldsymbol{s}}$ se fait en utilisant l'association unique entre $\hat{\boldsymbol{s}}$ et $\hat{\boldsymbol{X}}$. A la sortie du décodeur, les symboles seront démodulés. La démodulation est l'opération inverse de la modulation. En faisant l'étiquetage inverse, les bits d'information sont reconstitués.

1.3 Gain des systèmes MIMO

Les systèmes MIMO peuvent exploiter différents gains par rapport aux systèmes à une seule antenne en émission et une seule antenne en réception (SISO). Un gain de diversité et de multiplexage peuvent être obtenus. Nous allons présenter ces différents gains dans ce qui suit.

1.3.1 Gain de diversité

Il est évident qu'un signal transmis sur un canal radio subit des variations en fonction du temps, de la fréquence et de l'espace. Lorsque le signal est fortement affecté, on dit qu'il a subi un évanouissement. L'envoi d'une seule réplique du signal peut être insuffisante pour décoder l'information. Il serait donc intéressant de fournir au récepteur diverses répliques indépendantes : cette technique s'appelle la diversité. On parle alors de branches de diversité. Plus le nombre de branches augmente, plus la probabilité qu'au moins une des branches ne soit pas dans un évanouissement augmente.

Nous distinguons différents types de diversité :

- La diversité temporelle : c'est l'envoi du même signal en n instants différents. Les instants sont séparés d'au moins le temps de cohérence du canal afin d'assurer une bonne décorrélation des signaux.Ce type de diversité est intéressant pour les canaux ergodiques.
- La diversité fréquentielle : c'est l'envoi du même signal sur n fréquences différentes. Les fréquences sont séparées d'au moins la bande de cohérence du canal. La diversité fréquentielle est utilisée dans les systèmes OFDM (Orthogonal Frequency Division Multiplexing).

Avec les systèmes MIMO, on dispose d'une nouvelle forme de diversité, la diversité spatiale. Quand on utilise plusieurs antennes à l'émission, chacune devient une source d'information différente pour les antennes de réception.

- La diversité d'espace en émission : c'est l'envoi du même signal sur n_t antennes différentes séparées d'au moins dix fois la longueur d'onde. A la réception, cette diversité est perçue comme une diversité temporelle.
- La diversité d'espace en réception : c'est la réception du même signal sur n_r antennes différentes séparées d'au moins dix fois la longueur d'onde. L'ordre de diversité maximal est égal à n_r.

L'utilisation de systèmes à antennes multiples à l'émission et à la réception permet d'acquérir la diversité en réception. Il reste donc à récupérer la diversité en émission. En pratique, pour mettre à profit cette diversité, il est nécessaire d'utiliser un codage espace-temps.

La combinaison de plusieurs types de diversité permet d'obtenir des ordres de diversité élevés, l'ordre de diversité total étant le produit des ordres de diversités particuliers. Elle permet de combattre plus efficacement les effets des évanouissements. Le gain de diversité maximal possible est $n_t \times n_r$.

La diversité est perçue en particulier dans les performances d'un système MIMO et correspond à la pente de l'asymptote de la probabilité d'erreur P_e à fort rapport signal à bruit (RSB). $P_e \propto RSB^{-d}$ ou encore pour une diversité d'ordre d :

$$d = -\lim_{RSB \to \infty} \frac{\log P_e}{\log RSB} \tag{1.8}$$

1.3.2 Gain de multiplexage

Lorsque les évanouissements des différents trajets sur un lien radio sont indépendants, on peut montrer que la matrice du canal $\boldsymbol{H^c}$ est de rang plein avec une grande probabilité. Le canal MIMO peut alors être vu comme un ensemble de canaux SISO indépendants. En transmettant des flux d'information dans chacun de ces canaux, il est possible d'augmenter le débit d'information : c'est le gain de multiplexage. Ce gain est limité par $\min(n_t, n_r)$. Dans [5], un compromis entre le gain de diversité et le gain de multiplexage a été réalisé.

Il existe deux techniques pour exploiter le potentiel des systèmes MIMO : le multiplexage spatial et le codage espace-temps. Dans le multiplexage spatial, des flux de données indépendants sont transmis sur différentes antennes d'émission maximisant ainsi le débit transmis. Le codage espace-temps, quant à lui, offre une diversité et un gain de codage tout en améliorant l'efficacité spectrale.

1.4 Le multiplexage spatial : V-BLAST

Le multiplexage est l'opération qui consiste à diviser la séquence de données en plusieurs flux ou trames et de les transmettre sur des canaux indépendants, en temps, en fréquence ou en espace. Ce dernier type de multiplexage est le multiplexage par répartition en espace ou plus simplement le multiplexage spatial. Le système transmet alors n_t flux en un intervalle de temps. Le but est ainsi d'augmenter le débit d'information par rapport à un système SISO. Cette technique a été introduite sous le nom de BLAST (Bell Labs Layered Space-Time). Bien qu'il en existe différentes versions [6], la version la plus utilisée est le V-BLAST (Vertical BLAST) présentée par Foschini et *al.* dans [7], [6], [8].

Afin d'exploiter la dimension espace-temps du canal MIMO, le vecteur symboles d'information est divisé en n_t flux. L'idée ici est de répartir ces flux selon une trajectoire verticale de façon à ce que chacun soit transmis sur une antenne, d'où la structure en couches verticale. Cette architecture est très simple à réaliser puisqu'il s'agit uniquement d'une transformation série-parallèle (S/P).

A titre d'exemple, nous allons considérer le cas où $n_t = n_r = 2$, $K = 2$ et $T = 1$. La matrice du mot de code est la suivante :

$$X^c_{VBLAST,2} = \begin{bmatrix} s^c_1 \\ s^c_2 \end{bmatrix} \tag{1.9}$$

Il faut noter que dans ce cas, $\boldsymbol{X^c} = \boldsymbol{s^c}$. Le V-BLAST peut être vu comme une transmission non codée.

Le signal reçu se présente de la manière suivante :

$$\begin{bmatrix} y^c_1 \\ y^c_2 \end{bmatrix} = \boldsymbol{H^c} \cdot \begin{bmatrix} s^c_1 \\ s^c_2 \end{bmatrix} + \begin{bmatrix} w^c_1 \\ w^c_2 \end{bmatrix} \tag{1.10}$$

Les structures BLAST ont été conçues dans le but de maximiser le débit transmis sur le canal MIMO. Notons que le rendement d'un schéma V-BLAST est égal à n_t symboles/u.c. Par construction, le multiplexage spatial n'exploite pas la diversité spatiale en émission mais seulement celle en réception.

Pour retrouver les symboles à la réception, le système doit être bien posé, c'est-à-dire que le nombre de symboles transmis simultanément doit être inférieur au rang de la matrice de transfert du canal $\boldsymbol{H^c}$. Or le rang de la matrice du canal est borné par le minimum du nombre d'antennes en émission et en réception. Par la suite, une contrainte importante pour les schémas MIMO de type BLAST est que le nombre d'antennes en réception doit être supérieur ou égal au nombre d'antennes en émission : $\boldsymbol{n_t \leq n_r}$.

1.5 Le codage Espace-Temps

Les codes espace-temps ou spatio-temporels (ST) ont été proposés afin d'exploiter la diversité en émission et améliorer les performances des systèmes MIMO [3]. Le codage ST permet d'introduire une dépendance entre le domaine temporel et spatial. Nous distinguons deux grandes familles de codes ST : les codes ST en treillis (STT) [3], [4] et les codes ST en blocs (STBC) [9].

Les codes en treillis peuvent être vus comme la généralisation au cas MIMO des modulations codés en treillis développées auparavant pour le cas SISO. Bien que ces codes offrent de bonnes performances, ils souffrent en général d'une grande complexité de décodage due à l'utilisation de l'algorithme de Viterbi. Dans la suite, nous ne présentons que les codes ST en blocs qui sont plus intéressants sur le plan pratique.

1.5.1 Les codes ST en blocs

Plusieurs constructions des codes ST en blocs existent dans la littérature [10], [11], [12], [13], [14], [9], [15]. La plus célèbre parmi elles est celle du code d'Alamouti. Dans [16], S. Alamouti a proposé un schéma de codage orthogonal pour $n_t = 2$ et $n_r = 1$ dont le détecteur à maximum de vraisemblance (Maximum Likelihood ML) se réduit à une simple détection linéaire à forçage à zéro ZF (Zero Forcing). Grâce à cette simplicité d'implémentation, le code d'Alamouti a rapidement été adopté dans les normes. De plus, le code d'Alamouti est le seul code orthogonal à éléments complexes permettant d'atteindre la diversité maximale d'ordre 2 avec un rendement $R = 1$ symbole/u.c.

Dès 1999, Tarokh et *al.* ont présenté une généralisation du code d'Alamouti pour un nombre quelconque d'antennes à l'émission et une antenne à la réception proposant ainsi une nouvelle famille de codes ST en blocs [14], [17]. Malheureusement, ces constructions sont pénalisées par leurs rendements strictement inférieurs à 1 symbole/u.c. Pour atteindre un rendement de 1 pour plus de deux antennes en émission, il est nécessaire de sacrifier la propriété d'orthogonalité [18], [19], [20]. Des codes ST de rendement compris entre 1 et n_t ont été construits.

En général, nous notons par $\boldsymbol{X}$ un mot de code STBC. $\boldsymbol{X}$ est représenté par une matrice de taille $n_t \times T$ dont les composantes sont les symboles d'information ou des combinaisons linéaires des symboles d'information $\{s_1^c, s_2^c, \ldots, s_K^c\}$ et de leurs conjugués $\{s_1^c, s_2^c, \ldots, s_K^c\}^*$. A titre d'exemple, le mot de code d'Alamouti s'écrit :

$$\boldsymbol{X} = \begin{bmatrix} s_1^c & -s_2^{c*} \\ s_2^c & s_1^{c*} \end{bmatrix} \tag{1.11}$$

pour $n_t = 2$ et $K = 1$.

1.5.2 Les codes ST à dispersion linéaire (LDC)

Nous avons vu que le schéma V-BLAST ne permet pas d'exploiter la diversité en émission. Les codes ST en blocs, quant à eux, permettent de résoudre ce problème et d'atteindre une diversité optimale. L'idée des codes à dispersion linéaire est de proposer une construction commune pour les BLAST et les codes STBC combinant ainsi les avantages de l'une et l'autre des deux structures.

Les codes LDC ont été proposés par Hassibi dans [15]. Ils sont obtenus par combinaison linéaire de matrices, appelées matrices de dispersion. Ces codes possèdent la structure suivante :

$$\boldsymbol{X} = \sum_{k=1}^{K} \left(\Re\left(s_k^c\right) \boldsymbol{A_k} + j \Im\left(s_k^c\right) \boldsymbol{B_k} \right) \tag{1.12}$$

où les coefficients $\Re\left(s_k^c\right)$ et $\Im\left(s_k^c\right)$ sont des scalaires, ils représentent la partie réelle et imaginaire du symbole modulé s_k^c et $\boldsymbol{A_1}, \boldsymbol{A_2}, \ldots, \boldsymbol{A_K}, \boldsymbol{B_1}, \boldsymbol{B_2}, \ldots, \boldsymbol{B_K}$ sont des matrices de dispersion complexes de dimension $n_t \times T$.

Le code est entièrement défini à partir de la donnée de K ainsi que des $\boldsymbol{A_k}$ et $\boldsymbol{B_k}$. Le choix des matrices de dispersion déterminent les performances du code considéré. Dans [4], [13], des critères de constructions de codes ST ont été introduits et sont applicables aux codes ST à dispersion linéaire, tels que le critère de maximisation de l'information mutuelle, le critère du rang permettant d'avoir une diversité maximale et le critère du déterminant visant à optimiser le gain de codage. Pour les codes LDC, le choix des matrices de dispersion est obtenu en maximisant l'information mutuelle entre les signaux émis et les signaux reçus. Toutefois, ce critère à lui seul ne garantit pas la diversité maximale. Les codes LDC construits par Hassibi sont obtenus par maximisation de l'information mutuelle [15].

Dans la littérature, plusieurs autres schémas de codage à dispersion linéaire ont été proposés. Nous pouvons citer par exemple, les codes construits à travers des structures algébriques comme

le DAST [10], le TAST [21] ou à partir d'algèbres cycliques de division tels que les Codes Parfaits [22]. Il est à noter également que le V-BLAST et Alamouti tels que définis respectivement dans (1.9) et (1.11) sont des exemples particuliers des codes LDC. Dans le cas du multiplexage spatial de type V-BLAST, le codage ST est réduit à une simple répartition des symboles sur les antennes d'émission.

Les systèmes MIMO employant un code LDC peuvent avoir une représentation en réseau de points, ce qui permet de les décoder par les décodeurs de réseaux de points en plus des procédés de décodage linéaires tels que le ZF, MMSE, ... Dans le paragraphe suivant, nous allons introduire la notion de réseaux de points et la représentation en réseau de points d'un système MIMO employant un codage à dispersion linéaire.

1.6 Notions de réseaux de points pour les systèmes MIMO

La théorie de réseaux de points est fondamentale pour le décodage des systèmes MIMO. Dans ce qui suit, nous commençons par donner la définition d'un réseau de points et quelques propriétés importantes [23], [24].

1.6.1 Définitions relatives aux réseau de points

Définition 1 : *Réseau de points*

Soient $\boldsymbol{v_1}, \boldsymbol{v_2}, \ldots, \boldsymbol{v_m}$ m vecteurs linéairement indépendants de $\mathbb{R}^n$. Nous appelons réseau de points Λ de $\mathbb{R}^n$, le réseau défini par :

$$\Lambda = \left\{ \boldsymbol{x} = \sum_{i=1}^{m} \alpha_i \boldsymbol{v}_i \, , \; \alpha_i \in \mathbb{Z} \right\} \tag{1.13}$$

Λ est constitué de toutes les combinaisons linéaires des vecteurs v_i, $1 \leq i \leq m$. m est appelé la dimension du réseau de points. $(\boldsymbol{v_1}, \boldsymbol{v_2}, \ldots, \boldsymbol{v_m})$ forme une base de Λ. Cette base n'est pas unique, nous pouvons trouver une infinité de bases pour le réseau de points. Pour le réseau de points de dimension 2 représenté dans la figure 1.4, (v_1, v_2) et $\left(v_1^{'}, v_2^{'}\right)$ constituent deux bases différentes de Λ.

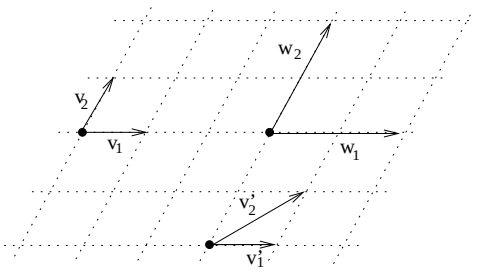


FIGURE 1.4 – Exemple de réseau de points en dimension 2

Définition 2 : *Matrice génératrice*

D'après la figure (1.4), un point du réseau s'écrit $\boldsymbol{x} = \alpha_1 \boldsymbol{v_1} + \alpha_2 \boldsymbol{v_2} + \ldots + \alpha_m \boldsymbol{v_m}$. $\boldsymbol{x}$ peut alors s'écrire sous la forme matricielle suivante :

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} v_{11} & v_{12} & \ldots & v_{1m} \\ v_{21} & v_{22} & \ldots & v_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ v_{n1} & v_{n2} & \ldots & v_{nm} \end{bmatrix} \cdot \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix} \tag{1.14}$$

La matrice $\boldsymbol{M} = \begin{bmatrix} v_{11} & v_{12} & \ldots & v_{1m} \\ v_{21} & v_{22} & \ldots & v_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ v_{n1} & v_{n2} & \ldots & v_{nm} \end{bmatrix}$ est appelée la matrice génératrice du réseau de points noté $\Lambda_{\boldsymbol{M}}$, le vecteur $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_m]^T$ est le vecteur coordonnée de $\boldsymbol{x}$ dans $\Lambda_{\boldsymbol{M}}$. Le réseau est complètement défini par la matrice génératrice $\boldsymbol{M}$. Un point du réseau peut être vu comme une transformation linéaire appliquée au réseau $\mathbb{Z}^m$:

$$\boldsymbol{x} = \boldsymbol{M} \cdot \boldsymbol{\alpha} \ , \ \alpha \in \mathbb{Z}^m \tag{1.15}$$

Définition 3 : *Matrice de Gram*

On appelle matrice de Gram de $\boldsymbol{M}$ ou du réseau $\Lambda_{\boldsymbol{M}}$, la matrice définie par $\boldsymbol{G} = \boldsymbol{M^T} \cdot \boldsymbol{M}$.

Définition 4 : *Sous-réseau*

Soit $\boldsymbol{U}$ une matrice à éléments dans $\mathbb{Z}$ de taille $n \times n$. $\Lambda_{\boldsymbol{M'}}$ est un sous-réseau de $\Lambda_{\boldsymbol{M}}$ défini par :

$$\Lambda_{\boldsymbol{M'}} = \{\boldsymbol{x} = \boldsymbol{U} \cdot \boldsymbol{M} \cdot \boldsymbol{\alpha} \ , \ , \boldsymbol{\alpha} \in \mathbb{Z}^m\}$$

Dans la figure 1.4, un exemple de réseau de points de $\mathbb{Z}^2$ est Λ de dimension 2 de base (v_1, v_2). Un sous-réseau de Λ est défini par la base (w_1, w_2).

Définition 5 : *Réseau équivalent*

Un réseau de points $\Lambda_{\boldsymbol{M}}$ de matrice génératrice $\boldsymbol{M}$ est équivalent à un réseau de points $\Lambda_{\boldsymbol{M'}}$ de matrice génératrice $\boldsymbol{M'}$ si :

$$\boldsymbol{M'} = c \cdot \boldsymbol{U} \cdot \boldsymbol{M} \cdot \boldsymbol{Q}$$

avec $c \in \mathbb{N} - \{0\}$ est un entier naturel non nul, $\boldsymbol{U}$ une matrice unimodulaire[1] et $\boldsymbol{Q}$ est une matrice orthogonale [2].

[1] Une matrice unimodulaire est une matrice carrée d'entiers naturels avec un déterminant égal à -1 ou $+1$.

[2] Une matrice orthogonale $\boldsymbol{Q}$ est tel que $\boldsymbol{Q^T} \cdot \boldsymbol{Q} = \boldsymbol{I}$

1.6.2 Représentation en réseaux de points des systèmes MIMO

La théorie de réseaux de points est un outil mathématique puissant permettant d'étudier géométriquement les propriétés d'un système. Dans [25], une représentation en réseaux de points des systèmes MIMO a été proposée. Cette représentation permet d'exploiter la théorie de réseaux pour proposer des algorithmes de décodage optimaux et sous-optimaux des systèmes MIMO. Nous illustrons ici la représentation en réseau de points d'un système MIMO que nous considérons dans toute la suite du rapport. Nous distinguons deux cas : un schéma de transmission employant le multiplexage spatial (V-BLAST) qui correspond à une transmission non codée, et un schéma de transmission employant un codage espace-temps à dispersion linéaire.

Cas non codé

Reprenons le système MIMO à n_t antennes à l'émission et n_r antennes à la réception. D'après (1.9), le système non codé est réduit en notation complexe à l'équation :

$$\boldsymbol{y^c} = \boldsymbol{H^c} \cdot \boldsymbol{s^c} + \boldsymbol{w^c} \tag{1.16}$$

avec $\boldsymbol{y^c}$ le vecteur reçu de dimension n_r, $\boldsymbol{H^c}$ la matrice du canal de dimension $n_r \times n_t$, $\boldsymbol{s^c}$ le vecteur symbole d'information transmis de dimension n_t et $\boldsymbol{w^c}$ le vecteur de bruit de dimension n_r.

Considérons la fonction suivante permettant de convertir un vecteur complexe de dimension n en un vecteur réel [26] :

$$\begin{array}{rcl} \Psi : \mathbb{C}^n & \rightarrow & \mathbb{R}^{2n} \\ \boldsymbol{v^c} & \rightarrow \boldsymbol{v} = & \begin{bmatrix} \mathfrak{Re}(\boldsymbol{v^c}) \\ \mathfrak{Im}(\boldsymbol{v^c}) \end{bmatrix} \end{array} \tag{1.17}$$

Pour les matrices, cette transformation est définie par :

$$\begin{array}{rcl} \Psi : \mathbb{C}^{n \times m} & \rightarrow & \mathbb{R}^{2n \times 2m} \\ \boldsymbol{M^c} & \rightarrow \boldsymbol{M} = & \begin{bmatrix} \Re(\boldsymbol{M^c}) & -\Im(\boldsymbol{M^c}) \\ \Im(\boldsymbol{M^c}) & \Re(\boldsymbol{M^c}) \end{bmatrix} \end{array} \tag{1.18}$$

En appliquant la fonction Ψ au système (1.16), nous obtenons le système suivant de dimension double :

$$\boldsymbol{y} = \boldsymbol{H} \cdot \boldsymbol{s} + \boldsymbol{w} \tag{1.19}$$

Les évanouissements de la matrice du canal $\boldsymbol{H^c}$ sont indépendants, la probabilité d'avoir deux colonnes de $\boldsymbol{H^c}$ dépendantes est nulle. La matrice est donc de rang plein. De même $\boldsymbol{H}$ est une matrice réelle de taille $2n_r \times 2n_t$ de rang plein.

D'après la définition du réseau de points donnée dans (1.15), le vecteur reçu $\boldsymbol{y}$ peut être vu comme un point du réseau $\Lambda_{\boldsymbol{H}}$ perturbé par un vecteur de bruit.

Cas codé

Pour le cas codé, reprenons le système défini par (1.7). Pour obtenir la représentation en réseau de points, nous commençons par réécrire le système sous forme vectorielle comme pour le cas non codé (1.16).

Dans [25],[27], les auteurs donnent une méthode permettant d'extraire le vecteur symbole d'information $\boldsymbol{s^c}$ de la matrice $\boldsymbol{X^c}$. Cette méthode est valable pour les codes LDC :

$$\boldsymbol{X^c} = \begin{pmatrix} \phi_{11} & \dots & \phi_{1,n_t \cdot T} \\ \vdots & \ddots & \vdots \\ \phi_{n_t \cdot T,1} & \dots & \phi_{n_t \cdot T, n_t \cdot T} \end{pmatrix} \cdot \begin{pmatrix} s_1^c \\ \vdots \\ s_{n_t \cdot T}^c \end{pmatrix}$$

où $\boldsymbol{\phi}$ est une matrice complexe de taille $n_t \cdot T \times n_t \cdot T$ appelée matrice génératrice du code.

En appliquant cette transformation à (1.7), nous obtenons une vectorisation du système matriciel précédent :

$$\begin{aligned} \boldsymbol{y^c} = & \begin{pmatrix} \boldsymbol{H^c} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \boldsymbol{H^c} \end{pmatrix} \cdot \begin{pmatrix} \phi_{11} & \dots & \phi_{1,n_t \cdot T} \\ \vdots & \ddots & \vdots \\ \phi_{n_t \cdot T,1} & \dots & \phi_{n_t \cdot T, n_t \cdot T} \end{pmatrix} \cdot \begin{pmatrix} s_1^c \\ \vdots \\ s_{n_t \cdot T}^c \end{pmatrix} + \begin{pmatrix} w_1^c \\ \vdots \\ w_{n_r \cdot T}^c \end{pmatrix} \quad (1.20) \\ = & \quad \boldsymbol{H_1} \cdot \boldsymbol{\phi} \cdot \boldsymbol{s} + \boldsymbol{w} \end{aligned}$$

La multiplication de $\boldsymbol{H_1}$ par $\boldsymbol{\phi}$ donne une matrice équivalente $\boldsymbol{M^c}$:

$$\boldsymbol{y^c} = \boldsymbol{M^c} \cdot \boldsymbol{s^c} + \boldsymbol{w^c} \quad (1.21)$$

Nous obtenons une représentation similaire que le système non codé. $\boldsymbol{M^c}$ de taille $n_r \cdot T \times n_t \cdot T$ est considérée comme la matrice du canal équivalente pour un système codé.

La deuxième étape consiste alors à transformer le système complexe obtenu en un système réel en utilisant (1.17) et (1.18). Nous obtenons ainsi :

$$\boldsymbol{y} = \boldsymbol{M} \cdot \boldsymbol{s} + \boldsymbol{w} \quad (1.22)$$

Dans ce cas, $\boldsymbol{M}$ est la matrice génératrice du réseau de points de dimension $2n_rT \times 2n_tT$.

Dans toute la suite, nous désignons par (1.19) et (1.22) respectivement la représentation en réseau de points d'un système MIMO non codé et codé.

1.6.3 Quelques outils mathématiques pour le décodage des réseaux de points

En utilisant la représentation en réseaux de points des systèmes MIMO, différents décodeurs optimaux et sous-optimaux peuvent à présent être mis en oeuvre. Parmi ces décodeurs, on trouve tout particulièrement les décodeurs de réseaux de points.
Ces derniers peuvent être vus comme des algorithmes de recherche dans l'arbre. Afin de construire l'arbre de recherche, nous avons besoin de transformer le système initial en utilisant des outils mathématiques de décomposition de matrices, telles que la décomposition QR, la factorisation de Cholesky, la décomposition en valeurs singulières (SVD), etc. Nous avons choisi de présenter ces trois décompositions qui seront utilisées dans la phase de décodage.

Décomposition QR

Principe

La décomposition QR permet la transformation d'une matrice quelconque $\boldsymbol{M}$ en une matrice équivalente présentée sous une forme canonique. Nous supposons que $\boldsymbol{M}$ est une matrice réelle de dimension $n \times m$, il existe une matrice orthogonale $\boldsymbol{Q}$ ($\boldsymbol{Q^T \cdot Q = I}$) de dimension $n \times n$ et une matrice triangulaire supérieure $\boldsymbol{R}$ de dimension $n \times m$ tel que :

$$\boldsymbol{M = Q \cdot R} \tag{1.23}$$

Application

En appliquant cette décomposition à la matrice du canal dans (1.19), le système MIMO devient :

$$\boldsymbol{y = Q \cdot R \cdot s + w} \tag{1.24}$$

$\boldsymbol{Q}$ est une matrice orthogonale, la multiplication de l'équation précédente par $\boldsymbol{Q^T}$ donne :

$$\boldsymbol{z = Q^T \cdot y = R \cdot s + Q^T \cdot w} \tag{1.25}$$

Nous obtenons un système équivalent puisque le bruit $\boldsymbol{Q^T \cdot w}$ reste blanc Gaussien. $\boldsymbol{R}$ est alors considérée comme la nouvelle matrice génératrice du réseau de points $\Lambda_{\boldsymbol{R}}$.

Il est à noter que le système obtenu représente une forme plus simplifiée du système initial. Cette phase de triangularisation de la matrice génératrice est utilisée dans plusieurs algorithmes de décodage. Nous l'appelons la phase de **pré-décodage**.

Complexité

La complexité introduite par la décomposition QR dépend des dimensions du réseau. Pour un réseau de dimension n, elle est de l'ordre de $\frac{2}{3}n^3$ [8] à laquelle s'ajoute la complexité due à la multiplication de la matrice $\boldsymbol{Q^T}$ par le vecteur reçu $\boldsymbol{y}$, égale à $n \cdot m$.

Algorithm 1 Décomposition QR

- Entrées : M, n
- Initialisation : Pour $i = 1 : n$, pour $j = 1 : n$, $R(i,j) = 0$.
- $R(1,1) = \sum_{i=1}^{n} M(i,1)$; $Q = M(:,1)/R(1,1)$;
- Pour $k = 2 : n$
  $R(1 : k-1, k) = Q(:, 1 : k-1)^T \cdot M(:,k)$;
  $Q(:,k) = M(:,k) - Q(:, 1 : k-1) \cdot R(1 : k-1, k)$;
  $R(k,k) = \sum_{i=1}^{n} Q(i,k)$
  $Q(:,k) = Q(:,k)/R(k,k)$

Factorisation de Cholesky

Principe

De manière similaire à la décomposition QR, la factorisation de Cholesky permet d'obtenir une représentation canonique plus simple de la matrice de départ.
La méthode de Cholesky s'applique à une matrice symétrique définie positive. Pour un réseau de points $\Lambda_{\boldsymbol{M}}$, elle peut être alors utilisée pour la factorisation de la matrice de Gram ($\boldsymbol{G} = \boldsymbol{M^T \cdot M}$). Elle consiste à déterminer une matrice triangulaire inférieure $\boldsymbol{L}$ dont les termes diagonaux sont réels positifs, tel que :

$$\boldsymbol{G = (LU) \cdot (LU)^T} \tag{1.26}$$

Ou plus simplement :

$$\boldsymbol{G = L \cdot L^T} \tag{1.27}$$

$\boldsymbol{U}$ étant une matrice unitaire (c'est-à-dire $\boldsymbol{U \cdot U^T = U^T \cdot U = I}$).

Application

Nous pouvons démontrer que la décomposition de Cholesky et la décomposition QR sont équivalentes. Supposons $\boldsymbol{M = Q \cdot R}$ est la décomposition QR de $\boldsymbol{M}$. La matrice de Gram peut s'écrire :

$$\begin{aligned} G &= \boldsymbol{M^T \cdot M} \\ &= \boldsymbol{(QR)^T \cdot (QR)} \\ &= \boldsymbol{R^T \cdot Q^T \cdot Q \cdot R} \\ &= \boldsymbol{R^T \cdot R} \\ &= \boldsymbol{L \cdot L^T} \end{aligned} \tag{1.28}$$

Nous pouvons voir que $\boldsymbol{R = L^T}$. $\boldsymbol{Q}$ est obtenu par l'opération $\boldsymbol{Q = M \cdot R^{-1}}$. Il est donc possible de passer de l'une à l'autre des représentations.

Une autre application de la décomposition de Cholesky est de calculer la matrice inverse de $\boldsymbol{G}$, de calculer son déterminant, etc. Ce dernier est égal au carré du produit des éléments diagonaux de $\boldsymbol{L}$. En effet :

$$\begin{aligned} \|\det(G)\| &= \left\|\det\left(LL^T\right)\right\| \\ &= \|\det(L)\| \quad \left\|\det\left(L^T\right)\right\| \\ &= \|\det(L)\|^2 \end{aligned} \tag{1.29}$$

Le déterminant d'une matrice triangulaire est le produit de ses éléments diagonaux, nous avons donc :

$$\|\det(G)\| = \left(\prod_{i=1}^{n} L_{ii}\right)^2 \tag{1.30}$$

Complexité

La complexité de la décomposition de Cholesky est donnée par $\frac{1}{6}n^3 + n$ [8]. Elle est inférieure à celle de la méthode QR. Toutefois, sachant que la décomposition de Cholesky se fait sur la matrice de Gram, cela rajoute n^3 opérations de multiplication. De plus, la décomposition QR est plus stable numériquement. Pour cela, nous avons choisi d'utiliser la décomposition QR dans la phase de pré-décodage dans notre travail.

Algorithm 2 Factorisation de Cholesky

- Entrées : $G(i,j)_{1<j<i<n}$ représentant la partie triangulaire inférieure de G, n.
- Sorties : $G(i,j)_{1<i<j<n}$ représentant L qui satisfait $G = L \cdot L^t$.
- $G(1,1) = sqrt(G(1,1))$.
- Pour $i = 2 : n$
  $G(i,1) = G(i,1)/G(1,1)$;
- Pour $k = 2 : n-1$
  $G(k,k) = sqrt(G(k,k) - \sum_{j=1}^{k-1} G^2(k,j))$;
  Pour $i = k+1 : n$
  $G(i,k) = (G(i,k) - \sum_{j=1}^{k-1} G(i,j) \cdot G(k,j))/G(k,k)$;
- $G(n,n) = sqrt(G(n,n) - \sum_{j=1}^{n-1} G^2(n,j))$;

Décomposition SVD

Principe

La décomposition en valeurs singulières (SVD) est appliquée pour une matrice quelconque de taille $m \times n$ à éléments réels ou complexes. Elle consiste à écrire la matrice $\boldsymbol{M}$ sous la forme :

$$\boldsymbol{M} = \boldsymbol{U} \cdot \boldsymbol{D} \cdot \boldsymbol{V^T} \tag{1.31}$$

avec $\boldsymbol{U}$ une matrice unitaire $m \times m$, $\boldsymbol{D}$ une matrice diagonale $m \times n$ à éléments positifs ou nuls et $\boldsymbol{V}$ est une matrice unitaire $n \times n$. La matrice $\boldsymbol{D}$ contient les valeurs singulières de $\boldsymbol{M}$.

Application

Cette méthode peut être utilisée dans le décodage des systèmes MIMO. Par exemple, en utilisant la décomposition SVD pour un système MIMO avec $n_t \leq n_r$, nous pouvons montrer que le décodage du système initial de dimension $n_r \times n_t$ peut être ramené à un décodage dans un réseau de dimension plus petite $n_t \times n_t$. Pour cela, reprenons l'équation :

$$\boldsymbol{y} = \boldsymbol{M} \cdot \boldsymbol{s} + \boldsymbol{w} \tag{1.32}$$

avec $\boldsymbol{y}$, $\boldsymbol{M}$, $\boldsymbol{s}$ et $\boldsymbol{w}$ de taille respectives $n_r \times 1$, $n_r \times n_t$, $n_t \times 1$ et $n_r \times 1$.

La décomposition SVD appliquée à ce système donne :

$$\boldsymbol{y} = \boldsymbol{U} \cdot \boldsymbol{D} \cdot \boldsymbol{V^T} \cdot \boldsymbol{s} + \boldsymbol{w} \tag{1.33}$$

La matrice $\boldsymbol{U}$ étant unitaire, la multiplication par $\boldsymbol{U^T}$ donne un système équivalent :

$$\begin{aligned} \boldsymbol{U^T} \cdot \boldsymbol{y} = &\quad \boldsymbol{D} \cdot \boldsymbol{V^T} \cdot \boldsymbol{s} &&+\boldsymbol{U^T} \cdot \boldsymbol{w} \\ \boldsymbol{y_1} = &\quad \boldsymbol{\Phi} \cdot \boldsymbol{s} &&+\boldsymbol{w_1} \end{aligned} \tag{1.34}$$

Rappelons que la matrice diagonale $\boldsymbol{D}$ contient les valeurs singulières de $\boldsymbol{M}$. Or, $\boldsymbol{M}$ possède au maximum n_t valeurs singulières. Par conséquent, les lignes de $\boldsymbol{D}$ d'indice $> n_t$ sont nulles.

$$\boldsymbol{D} = \begin{bmatrix} d_1 & & \\ & \ddots & \\ & & d_{n_t} \\ 0 & \cdots & 0 \\ \vdots & & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \Bigg\downarrow n_r$$

Dans le système (1.34), la multiplication de $\boldsymbol{D}$ par $\boldsymbol{V^T}$ donne une matrice $\boldsymbol{\Phi}$ nulle au-delà de la ligne n_t :

$$\begin{bmatrix} y_{1_1} \\ \vdots \\ \\ \\ \\ y_{1_{n_r}} \end{bmatrix} = \begin{bmatrix} \Phi_{11} & \cdots & \Phi_{1n_t} \\ \vdots & & \vdots \\ \Phi_{n_t1} & \cdots & \Phi_{n_tn_t} \\ 0 & \cdots & 0 \\ \vdots & & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} s_1 \\ \vdots \\ s_{n_t} \end{bmatrix} + \begin{bmatrix} w_{11} \\ \vdots \\ \\ \\ \\ w_{1_{n_r}} \end{bmatrix} \tag{1.35}$$

Par conséquent, nous pouvons estimer que les $(n_r - n_t)$ dernières lignes du vecteur $\boldsymbol{y_1}$ sont également nulles. Il est par la suite inutile d'appliquer le décodage jusqu'à l'ordre n_r, le réseau initial de dimension $n_r \times n_t$ est ainsi réduit à un réseau de dimension plus petite $n_t \times n_t$.

La décomposition SVD permet notamment de calculer le rang d'une matrice. Il a été démontré dans [8] que le rang de la matrice $\boldsymbol{M}$ est égal au nombre des valeurs singulières non nulles de $\boldsymbol{M}$ ou encore le nombre des éléments non nuls de $\boldsymbol{D}$.

La décomposition en valeurs singulières permet aussi de calculer le pseudo-inverse d'une matrice. En effet, le pseudo-inverse de $\boldsymbol{M}$ est donné par :

$$\boldsymbol{M^\ddagger} = \boldsymbol{V} \cdot \boldsymbol{D^+} \cdot \boldsymbol{U^T} \tag{1.36}$$

telle que $\boldsymbol{D^+}$ représente une matrice diagonale où tout composant non nul de $\boldsymbol{D}$ est remplacé par son inverse.

Complexité

Pour une matrice $\boldsymbol{M}$ de taille $m \times n$, la complexité de cette phase est de l'ordre de $8m^2n + 8mn^2 + 9n^3$, elle est largement supérieure à la complexité des décompositions QR et de Cholesky. Cependant, en appliquant la décomposition SVD comme phase de pré-décodage, nous pouvons réduire le problème initial à un réseau de dimension plus petite, d'où un gain significatif en complexité de décodage, en particulier pour les algorithmes de décodage itératif.

1.7 Conclusion

Dans ce premier chapitre, nous avons commencé par présenter le modèle du système de transmission MIMO ainsi que les hypothèses et les notations que nous allons considérer tout le long de ce mémoire. Nous avons ensuite présenté deux schémas de transmissions possibles : le schéma non codé donné par l'architecture V-BLAST, et le schéma codé utilisant les codes espace-temps à dispersion linéaire.

Dans la deuxième partie, nous avons rappelé quelques notions de la théorie des réseaux de points et nous avons démontré que les systèmes MIMO employant un codage à dispersion linéaire admettent une représentation en réseau de points. Cette propriété rend possible le décodage des systèmes MIMO par les décodeurs de réseau de points.

Dans le chapitre suivant, nous présentons un état de l'art des décodeurs optimaux et sous-optimaux les plus connus dans la littérature.

Chapitre 2

Détection pour systèmes MIMO : État de l'art

2.1 Introduction

Dans ce chapitre nous présentons un état de l'art des principaux algorithmes de décodage pour les systèmes MIMO qui existent dans la littérature. En un premier temps, nous allons présenter les décodeurs sous-optimaux les plus utilisés tels que le ZF, le MMSE, les décodeurs avec retour de décision, etc.

Par la suite, nous allons présenter les décodeurs optimaux pour les systèmes MIMO. En utilisant la représentation en réseaux de points, nous allons décrire les décodeurs optimaux basés sur des algorithmes de décodage séquentiels. Nous distinguons deux catégories : les décodeurs employant la stratégie de Pohst, tels que le décodeur par sphères et l'algorithme de Schnorr-Euchner et les décodeurs employant la stratégie de Dijkstra (ou encore dite *best-first-search*) tels que le décodeur Stack et le décodeur Fano. Nous montrons aussi que cette dernière catégorie de décodeurs peut être modifiée pour donner une complexité plus faible au prix de performances sous-optimales.

2.2 Décodage pour système MIMO

Pour la simplicité, nous considérons un schéma de transmission utilisant le multiplexage spatial V-BLAST. Les algorithmes de décodage que nous expliciterons restent vrais pour un système employant un codage espace-temps. Seules les dimensions du système changent.

Reprenons l'équation qui donne la représentation en réseaux de points d'un système MIMO à n_t antennes en émission et n_r antennes en réception. Nous nous plaçons dans le cas d'un canal quasi-statique et d'une transmission cohérente, c'est-à-dire que la matrice du canal $\boldsymbol{H}$ est connue par le récepteur.

$$\boldsymbol{y} = \boldsymbol{H} \cdot \boldsymbol{s} + \boldsymbol{w} \tag{2.1}$$

Le décodage a pour but de trouver une estimation du vecteur émis connaissant le signal reçu $\boldsymbol{y}$.

Le décodage optimal est un décodage à maximum de vraisemblance (ML). Il consiste à chercher le vecteur le plus proche de $\boldsymbol{s}$ minimisant la métrique :

$$\hat{\boldsymbol{s}} = \arg \min_{\boldsymbol{s} \in C_s} \|\boldsymbol{y} - \boldsymbol{H} \cdot \boldsymbol{s}\|^2 \tag{2.2}$$

Cependant, il est possible de considérer d'autres critères pour trouver le vecteur émis, comme l'annulation des interférences entre symboles (Forçage à zéro des interférences - ZF) ou la minimisation de l'erreur quadratique moyenne (Minimum Mean Square Error - MMSE). Dès lors, nous parlons de décodeurs sous-optimaux.

2.3 Décodeurs sous-optimaux

Il existe deux grandes classes de décodeurs sous-optimaux : les décodeurs linéaires et les décodeurs avec retour de décision. Pour les premiers décodeurs, le décodage consiste à une multiplication du vecteur reçu par un filtre, suivie d'une décision sur les symboles résultants. Le choix de la matrice de transfert du filtre $\boldsymbol{F}$ détermine le critère de décodage et ses performances.

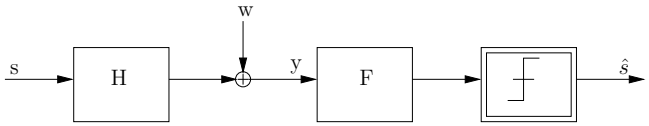


Figure 2.1 – Structure générale d'un décodeur linéaire

Pour les décodeurs avec retour de décision, le principe consiste après avoir mis le système sous une forme matricielle triangulaire supérieure, à estimer une première donnée puis à soustraire sa contribution avant d'estimer la suivante.

Dans ce qui suit, nous donnons des exemples de décodeurs sous-optimaux les plus utilisés dans la littérature.

2.3.1 Décodeur ZF

Pour un système MIMO, les symboles transmis par les autres antennes viennent s'ajouter au symbole transmis sur une antenne donnée. Par conséquent, l'utilisation d'antennes multiples crée de l'interférence entre symboles (IES) à la réception.

Le critère de forçage à zéro (Zero Forcing) consiste à appliquer un filtre $\boldsymbol{F}$ au signal reçu afin d'éliminer une partie des IES. Ceci est réalisé par la multiplication par l'inverse de la matrice du canal, ou plus généralement par le pseudo-inverse de la matrice du canal dans le cas où celle-ci est rectangulaire.

$$\boldsymbol{F_{ZF}} = \boldsymbol{H}^{\ddagger} = \left(\boldsymbol{H}^{\dagger}\boldsymbol{H}\right)^{-1} \boldsymbol{H}^{\dagger} \tag{2.3}$$

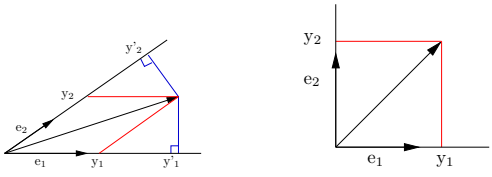
A la sortie du filtre ZF, le signal est représenté comme suit :

$$\boldsymbol{y_1} = \boldsymbol{F_{ZF}} \cdot \boldsymbol{y} = \boldsymbol{s} + \boldsymbol{F_{ZF}} \cdot \boldsymbol{w} \tag{2.4}$$

Le signal obtenue est le signal transmis auquel s'ajoute un bruit. Une simple détection à seuil permet par la suite de déterminer le signal transmis, le vecteur $\hat{\boldsymbol{s}}$ sera estimé par la quantification de $\boldsymbol{y_1}$ dans la constellation QAM utilisée :

$$\hat{\boldsymbol{s}} = Q_{QAM}\left\{\boldsymbol{F_{ZF}} \cdot \boldsymbol{y}\right\} \tag{2.5}$$

Le décodage ZF peut être vu comme une projection orthogonale du vecteur reçu sur la base formée de vecteurs lignes de $\boldsymbol{H}$.
La figure 2.2 montre un exemple de projection en dimension 2. Si la base n'est pas orthogonale alors la projection de $\boldsymbol{y}$ sur la base (e_1, e_2) engendre une erreur de décodage $(y'_1 - y_1, y'_2 - y_2)$ et correspond à des IES restantes. Si la base formée par les vecteurs de $\boldsymbol{H}$ est orthogonale, dans ce cas, la projection n'engendre pas d'erreurs et la solution obtenue est la solution ML.



(a) H quelconque (b) H orthogonale

Figure 2.2 – Projection orthogonale du vecteur reçu

En pratique, la matrice du canal n'est pas orthogonale. Des travaux existent dans la littérature permettant d'obtenir des bases équivalentes composées des vecteurs les plus courts et les plus orthogonaux possibles. Il s'agit des techniques de réduction [28]. De ce fait, l'application de ces techniques sur la matrice du canal comme phase de pré-traitement suivie d'un décodage ZF, permet d'obtenir des performances proches de ML. Ceci sera détaillé dans le dernier chapitre.

Calcul du bruit résultant

A la sortie du filtre ZF, le bruit résultant est $\tilde{\boldsymbol{w}} = \boldsymbol{F_{ZF}} \cdot \boldsymbol{w}$. Sa matrice de covariance est définie par :

$$\boldsymbol{R_{\tilde{w}\tilde{w}}} = E\left[(\tilde{\boldsymbol{w}}) \cdot (\tilde{\boldsymbol{w}})^{\dagger}\right] = \sigma_w^2 \left(\boldsymbol{H}^{\dagger} \cdot \boldsymbol{H}\right)^{-1} = \sigma_w^2 \boldsymbol{G}^{-1} \tag{2.6}$$

Ce résultat montre que le bruit n'est plus blanc, $\boldsymbol{R_{\tilde{w}\tilde{w}}} \neq \boldsymbol{R_{ww}} = \sigma_w^2 I$. De plus, si on applique la décomposition SVD à la matrice de Gram $\boldsymbol{G}$, nous pouvons écrire $\boldsymbol{G} = \boldsymbol{U} \cdot \boldsymbol{D} \cdot \boldsymbol{V}^{\dagger}$, avec $\boldsymbol{U}$ et $\boldsymbol{V}$ des matrices unitaires et $\boldsymbol{D}$ est la matrice diagonale contenant les valeurs singulières de $\boldsymbol{G}$.
En utilisant la propriété que les valeurs singulières de la matrice de Gram de $\boldsymbol{H}$ sont égales aux carrés des valeurs propres de $\boldsymbol{H}$, notées $\lambda_1, \ldots, \lambda_{n_r}$, la matrice de covariance de $\tilde{\boldsymbol{w}}$ est donnée comme suit :

$$\boldsymbol{R}_{\tilde{\boldsymbol{w}}\tilde{\boldsymbol{w}}} = \sigma_w^2 \boldsymbol{U} \cdot \begin{pmatrix} \sqrt{\lambda_1}^{-1} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sqrt{\lambda_{n_r}}^{-1} \end{pmatrix} \cdot \boldsymbol{V}^{\dagger} \tag{2.7}$$

Nous pouvons vérifier que le bruit est amplifié par les inverses des valeurs propres de $\boldsymbol{H}$. Ces valeurs sont d'autant plus grandes que la matrice $\boldsymbol{H}$ est mal conditionnée.
Ainsi, bien que la détection ZF permette de réduire les interférences, cette technique reste très limitée à cause de la coloration du bruit et le fait de ne pas tenir compte du bruit entraine une dégradation notoire des performances.

2.3.2 Décodeur MMSE

Le décodage MMSE est un exemple du décodage linéaire. Le filtre MMSE a été introduit pour remédier au problème précédent. Le but de ce décodeur est d'améliorer les performances globales du système en tenant compte à la fois des interférences et du bruit. On peut aussi définir ce filtre comme le filtre maximisant le rapport signal sur bruit plus interférences (Signal to interference plus Noise Ratio - SINR).

Le principe du MMSE est alors de trouver le filtre optimal $\boldsymbol{F}$ qui réduit l'erreur quadratique moyenne.

$$\begin{aligned} \boldsymbol{F_{MMSE}} &= \arg\min_{\boldsymbol{F}} \left(E\left\{ \|\hat{\boldsymbol{s}} - \boldsymbol{s}\|^2 \right\} \right) \\ &= \arg\min_{\boldsymbol{F}} \left(E\left\{ \|\boldsymbol{F} \cdot \boldsymbol{y} - \boldsymbol{s}\|^2 \right\} \right) \end{aligned} \tag{2.8}$$

Ceci est obtenu en prenant :

$$\boldsymbol{F_{MMSE}} = \boldsymbol{H}^{\dagger} \cdot \left(\boldsymbol{H}^{\dagger}\boldsymbol{H} + \frac{1}{\rho}\boldsymbol{I} \right)^{-1} \tag{2.9}$$

tels que $\rho = \frac{E_s}{\sigma_w^2}$ est le rapport signal sur bruit (RSB), E_s est l'énergie symbole et σ_w^2 est la variance du bruit. Le calcul détaillé du filtre MMSE est donné en annexe A.

Cas particuliers

- Si $\rho \to \infty$: Le bruit est négligeable. Dans ce cas, le décodeur traite uniquement les IES et le filtre MMSE devient équivalent au filtre ZF.
- Si $\rho \to 0$: $\frac{1}{\rho}\boldsymbol{I} >> \boldsymbol{H}^{\dagger}\boldsymbol{H}$, le bruit est prépondérant. Dans ce cas, le filtre MMSE est équivalent à : $\boldsymbol{F_{MMSE}} = \boldsymbol{\rho}\boldsymbol{H}^{\dagger}$. Le bruit est le paramètre à traiter en premier.

En général, le décodeur MMSE permet de réaliser un compromis entre l'élimination des interférences par la multiplication de $\boldsymbol{y}$ par $\boldsymbol{H}^{\dagger} \cdot \left(\boldsymbol{H}^{\dagger}\boldsymbol{H}\right)^{-1}$ et la réduction du bruit en considérant le deuxième terme $\rho \cdot \boldsymbol{I}$.

De plus, si on définit par $\boldsymbol{G}$ la matrice $\boldsymbol{H}^{\dagger}\boldsymbol{H} + \frac{1}{\rho}\boldsymbol{I}$, la décomposition SVD de la matrice $\boldsymbol{G}$ donne :

$$\boldsymbol{G} = \boldsymbol{U} \cdot \begin{pmatrix} \sqrt{\lambda_1} + \frac{1}{\rho} & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \sqrt{\lambda_{n_r}} + \frac{1}{\rho} \end{pmatrix} \cdot \boldsymbol{V}^{\dagger} \tag{2.10}$$

$\lambda_1, \dots, \lambda_{n_r}$ étant les valeurs propres de $\boldsymbol{H}$. La présence du terme $\frac{1}{\rho}$ permet de mieux équilibrer les coefficients de la diagonale en particulier dans le cas d'une matrice du canal mal conditionnée où les valeurs propres peuvent avoir des normes très différentes.

Le filtre MMSE peut être également un outil de pré-traitement dans la mesure où il offre une matrice du canal équivalente mieux conditionnée.

2.3.3 Décodeur avec retour de décision : ZF-DFE

Les décodeurs linéaires permettent une détection simultanée des flux transmis sur les différentes antennes en éliminant les IES. La méthode de décodage que nous présentons dans ce paragraphe permet néanmoins de décoder de manière successive les symboles d'information et ainsi d'éliminer l'interférence au fur et à mesure du décodage. De manière pratique, si un symbole $\hat{s}_k$ est estimé, le décodeur se sert de cette décision pour estimer les symboles précédents $\hat{s}_{k-1}, \hat{s}_{k-2}, \dots, \hat{s}_1$, d'où l'appellation de décodage avec retour de décision ou décodage non linéaire (DFE : Decision Feedback Equalization).

Le décodage du symbole $\hat{s}_k$ peut se faire selon le critère de forçage à zéro ZF, la minimisation de l'erreur quadratique MMSE, ... Dans ce qui suit, nous allons décrire un décodeur avec retour de décision employant le critère ZF.

Puisqu'il s'agit d'une détection successive des symboles, la décomposition QR de la matrice du canal s'avère utile dans ce cas :

$$\begin{aligned} \boldsymbol{y} &= \boldsymbol{H} \cdot \boldsymbol{s} + \boldsymbol{w} \\ &= \boldsymbol{Q}\boldsymbol{R} \cdot \boldsymbol{s} + \boldsymbol{w} \end{aligned} \tag{2.11}$$

Afin d'exploiter la forme triangulaire de la matrice $\boldsymbol{R}$, nous considérons le système équivalent suivant :

$$\begin{aligned} \boldsymbol{y_1} &= \boldsymbol{Q^T y} \\ &= \boldsymbol{R} \cdot \boldsymbol{s} + \boldsymbol{Q^T w} \end{aligned} \tag{2.12}$$

$\boldsymbol{R}$ étant triangulaire supérieure, à la première itération, le décodeur est capable d'estimer le symbole s_n à partir de l'équation suivante :

$$\hat{s}_n = Q_{QAM}\left\{\frac{y_{1n}}{r_{nn}}\right\} \tag{2.13}$$

Pour décoder le symbole d'information s_i, le décodeur utilise les symboles d'information déjà estimés $\hat{s}_j$, $j = i+1, \dots, n$ selon l'équation :

$$\hat{s}_i = Q_{QAM}\left\{\frac{1}{r_{ii}}\left(y_{1i} - \sum_{j=i+1}^{n} r_{ij}\cdot\hat{s}_j\right)\right\}\ ,\ 1 \leq i \leq n \tag{2.14}$$

Ce décodeur est très simple à mettre en oeuvre. Toutefois, l'inconvénient majeur de ce décodeur est la propagation d'erreurs. Si une erreur est commise pour les premiers symboles, elle sera propagée pour les symboles suivants.

2.3.4 Performances des décodeurs sous-optimaux

Dans la figure 2.3, nous comparons les performances et les complexitées obtenues pour les différents décodeurs sous-optimaux que nous avons présentés. Nous considérons un système MIMO 2×2 (avec deux antennes en émission et deux antennes en réception) utilisant un multiplexage spatial. La constellation employée est la constellation 4-QAM.

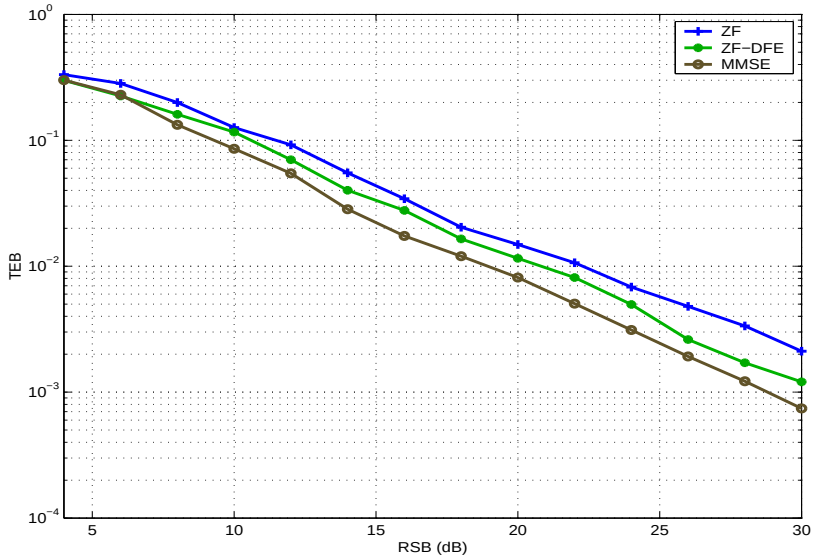
Les performances sont calculées en taux d'erreur par bit (TEB) en fonction du rapport signal à bruit en décibel (RSB). Sous ces hypothèses, le RSB est calculée par la formule suivante :

$$RSB = 10 \times log_{10}\left(\frac{n_r \sum_{i=1}^{n_t} E_{s_i}}{2log_2\left(q\right) N_0}\right) dB$$

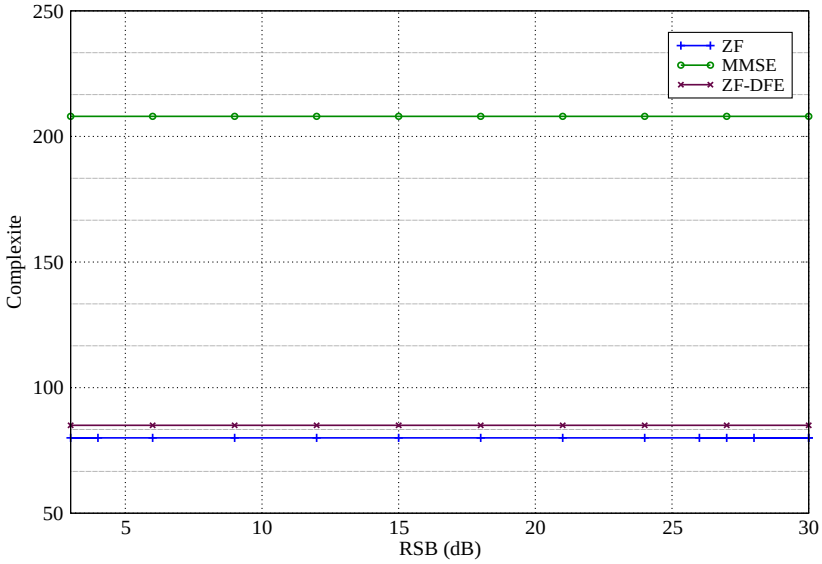
telle que E_{s_i} est l'énergie moyenne par dimension complexe du symbole d'information s_i appartenant à la constellation q-QAM.

A partir de la figure 2.3.a, nous notons que le décodeur MMSE donne de meilleures performances que les décodeurs ZF et ZF-DFE. Ce gain est d'autant plus grand que le nombre d'antennes augmente. Nous pouvons voir que les courbes ont la même pente. Ceci est du au fait que les décodeurs ont le même ordre de diversité. Pour un système à n_t antennes en émission et n_r antennes en réception, il a été prouvé dans [29] que la diversité en réception apportée par le décodeur ZF est égale à $n_r - n_t + 1$. Ainsi, pour un système MIMO symétrique ($n_r = n_t$), la diversité apportée par ces décodeurs est égale à 1. Les performances du système restent donc faibles quel que soit le nombre d'antennes déployées.

Dans la figure 2.3.b, nous représentons la complexité relative aux trois décodeurs. Celle-ci est calculée en nombre d'opérations de multiplication moyennes nécessaires pour décoder un mot de code. Pour les décodeurs linéaires, le décodage est réduit à la multiplication de la séquence reçue par le filtre $\boldsymbol{F}$ et pour le décodeur ZF-DFE, la complexité de décodage correspond à la détection successive des symboles d'information. Dans les deux cas, les opérations de décodage sont des opérations matricielles appliquées sur le signal reçu indépendamment de la puissance du bruit. Ceci explique le fait que nous obtenons des complexités fixes pour toute valeur du RSB. Dans ce cas, elle n'est fonction que de la taille du système considéré.



(a)



(b)

Figure 2.3 – Comparaison des performances des décodeurs sous-optimaux pour un système MIMO 2×2 utilisant un multiplexage spatial et une constellation 4-QAM

Bien que les décodeurs sous-optimaux offrent une complexité faible et constante, leur utilisation ne permet pas d'exploiter les avantages apportés par les systèmes MIMO à cause de leur diversité faible. Toutefois, si le nombre d'antennes en réception est très élevé par rapport au nombre d'antennes en émission, il peut éventuellement être intéressant d'utiliser les décodeurs

sous-optimaux puisque la diversité apportée devient non négligeable.

Afin de récupérer la diversité totale offerte par les systèmes MIMO et les codes ST, il faut considérer les décodeurs optimaux. Les algorithmes de décodage séquentiels sont des exemples de décodeurs optimaux.

2.4 Le décodage séquentiel

Outre le décodeur exhaustif ML, le décodage optimal peut être obtenu en appliquant les décodeurs séquentiels. Ces décodeurs sont basés sur la stratégie de recherche dans un arbre. Dans la littérature, plusieurs versions de décodeurs séquentiels ont été proposées. Les plus connus sont le décodeur par sphères, la stratégie de Schnorr-Euchner ou récemment le décodeur Stack et le décodeur Fano. Ces décodeurs présentent les mêmes performances mais ont des stratégies de recherche différentes. Dans les paragraphes suivants, nous allons décrire chacun d'eux.

Pour appliquer le décodage séquentiel, nous commençons d'abord par exposer **la structure d'arbre** du système MIMO. Cette phase s'appelle la phase de **pré-décodage** et elle est commune à tous les décodeurs que nous allons présenter.

2.4.1 Phase de pré-décodage

Nous supposons un système MIMO symétrique, $n_t = n_r$. Reprenons l'équation qui donne la représentation réelle en réseau de points du système MIMO.

$$\boldsymbol{y} = \boldsymbol{H} \cdot \boldsymbol{s} + \boldsymbol{w} \tag{2.15}$$

Considérons le réseau Λ_H engendré par la matrice $\boldsymbol{H}$, de dimension $n = 2 \cdot n_r$. Soit $\boldsymbol{x} = \boldsymbol{H} \cdot \boldsymbol{s} \in \mathbb{R}^n$ le point du réseau émis sur le canal Gaussien. Pour un tel système, le signal reçu peut être vu comme un point du réseau Λ_H auquel s'ajoute le vecteur bruit. D'après (2.2), le décodage ML consiste à chercher le point du réseau le plus proche de $\boldsymbol{y}$ minimisant la métrique :

$$\min_{\boldsymbol{x} \in \Lambda_H} \|\boldsymbol{y} - \boldsymbol{x}\|^2 = \min_{\boldsymbol{s} \in C_s / \boldsymbol{H} \cdot \boldsymbol{s} \in \Lambda_H} \|\boldsymbol{y} - \boldsymbol{H} \cdot \boldsymbol{s}\|^2 \tag{2.16}$$

Pour avoir une structure arborescente, il faut transformer la base du réseau initial en une base triangulaire. Celle-ci est obtenue par la décomposition QR de la matrice génératrice du réseau, $\boldsymbol{H} = \boldsymbol{Q} \cdot \boldsymbol{R}$. Le système (2.15) devient :

$$\begin{aligned} \boldsymbol{y_1} &= \boldsymbol{Q^T} \cdot \boldsymbol{y} \\ &= \boldsymbol{R} \cdot \boldsymbol{s} + \boldsymbol{w_1} \end{aligned} \tag{2.17}$$

$\boldsymbol{Q}$ est une matrice orthogonale, la multiplication par $\boldsymbol{Q^T}$ ne modifie pas le système précédent. La métrique ML devient dans le nouveau réseau :

$$\min_{\boldsymbol{s} \in C_s / \boldsymbol{R} \cdot \boldsymbol{s} \in \Lambda_R} \|\boldsymbol{y_1} - \boldsymbol{R} \cdot \boldsymbol{s}\|^2 \tag{2.18}$$

La nouvelle base triangulaire du réseau définie par $\boldsymbol{R}$ ramène la recherche du point le plus proche à une recherche séquentielle dans un arbre (*tree-search*).

Nous définissons l'arbre comme présenté dans la figure 2.4.

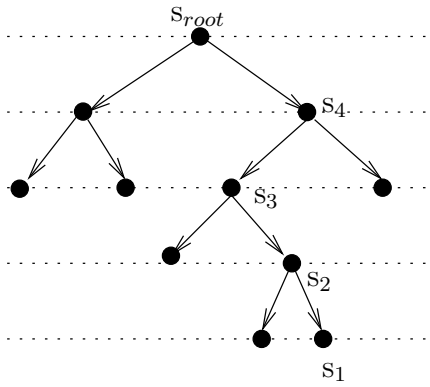


Figure 2.4 – Exemple d'arbre pour un réseau de dimension n=4

Les noeuds qui le constituent correspondent aux différentes valeurs possibles de s_i. Nous rappelons que s_i représentent les composantes réelles et imaginaires des symboles d'information modulés. Une branche de l'arbre est définie par deux noeuds consécutifs (s_{i+1}, s_i). Une métrique, appelée poids, est associée à chaque noeud s_i et est définie comme suit :

$$w_i(s_i) = \left| y_{1i} - \sum_{j=i}^{n} r_{i,j} s_j \right|^2 \tag{2.19}$$

Le poids représente la métrique de la branche (s_{i+1}, s_i). La matrice $\boldsymbol{R}$ étant triangulaire supérieure, la recherche commence alors par la composante s_n. Nous appelons noeud fils de s_i les composantes s_{i-1} et nous désignons par un chemin dans l'arbre à la profondeur i le vecteur de taille $n-i+1$ défini par $\boldsymbol{s^{(i)}} = (s_n, s_{n-1}, \ldots, s_i)$. On appellera noeud feuille, un noeud dans l'arbre de profondeur n. Le poids cumulé du noeud s_i représente la métrique du chemin $\boldsymbol{s^{(i)}}$. Il est alors égal à la somme des poids des différentes composantes qui le constituent :

$$w(s_i) = \sum_{j=i}^{n} w_j(s_j) \tag{2.20}$$

Pour un noeud feuille de taille n , le poids $w(s_1)$ correspond à la distance euclidienne entre le vecteur reçu $\boldsymbol{y_1}$ et le point $\boldsymbol{s^{(1)}}$ qui est égale à $\left|\boldsymbol{y_1} - \boldsymbol{R} \cdot \boldsymbol{s^{(1)}}\right|^2$.

D'après cette définition, la minimisation de la métrique ML (2.18) revient à chercher le chemin dans l'arbre ayant le plus faible poids cumulé :

$$\min_{\boldsymbol{s} \in C_s} \|\boldsymbol{y_1} - \boldsymbol{R} \cdot \boldsymbol{s}\|^2 \quad = \quad \min_{\boldsymbol{s} \in C_s} w(s_i) \tag{2.21}$$

Pour ce faire, plusieurs stratégies de recherche ont été proposées dans la littérature. Dans la suite, nous rappelons les principales stratégies.

2.4.2 Stratégies de recherche dans un arbre

Largeur d'abord : Breadth-First-Search (BrFS)

En partant du noeud racine s_{root}, cette méthode consiste à parcourir tous les noeuds se trouvant au niveau suivant (les noeuds fils). Pour chaque noeud trouvé, l'algorithme parcourt successivement tous ses noeuds fils, et ainsi de suite jusqu'à arriver à des noeuds feuilles. Cette méthode permet de parcourir l'arbre dans le sens de la largeur, i.e, l'algorithme explore tous les noeuds s_i avant de passer au niveau $i-1$ de l'arbre. De cette façon, toutes les combinaisons linéaires possibles $\boldsymbol{s^{(i)}} = (s_n, s_{n-1}, \ldots, s_i)$ sont calculées. Ainsi, l'algorithme correspond à une recherche exhaustive dans l'arbre.

Profondeur d'abord : Depth-First-Search (DFS)

En partant du noeud racine, cet algorithme définit le premier noeud fils s_n, puis son noeud fils s_{n-1} qui lui est directement relié, etc, jusqu'à arriver au noeud s_1. Une fois ce premier chemin trouvé, l'algorithme remonte au niveau précédent et explore le deuxième noeud fils de s_1. Une fois tous les chemins sont générés et leurs poids calculés, l'algorithme retourne le chemin le plus court. Similairement au BrFS, le DFS consiste en une recherche exhaustive, la solution ML est donc trouvée en dépit d'une très grande complexité.

Meilleur d'abord : Best-First-Search (BeFS)

La stratégie BeFS est une version optimisée de la stratégie BrFS. En effet, cette méthode permet de ne parcourir que les chemins ayant les plus faibles poids. En partant du noeud racine, l'algorithme explore tous les noeuds fils s_n et ne retient que le noeud ayant le plus faible poids. Les noeuds fils de celui-ci sont générés et leurs poids cumulés respectifs sont calculés. L'algorithme classe les noeuds explorés selon leurs poids et retient celui ayant le plus petit poids cumulé. Il est possible que l'algorithme remonte au niveau supérieur de l'arbre. L'algorithme continue ainsi jusqu'à trouver un noeud feuille $\boldsymbol{s^{(1)}} = (s_n, s_{n-1}, \ldots, s_1)$. Ainsi, l'algorithme permet d'explorer uniquement les chemins les plus "prometteurs" dans l'arbre qui constituent les métriques les plus courtes, réduisant ainsi la complexité par rapport aux autres algorithmes. La solution retournée est celle qui présente la plus faible métrique et correspond à la solution ML.

Stratégie de recherche par séparation et évaluation : Branch and Bound (BB)

Le principe de cette stratégie est de réduire la phase de recherche du point ML. Pour cela, cet algorithme introduit une contrainte, souvent liée à la métrique, afin de générer moins de noeuds. Les seuls chemins parcourus dans l'arbre sont ceux ayant les poids cumulés inférieurs ou égaux à une borne maximale fixée préalablement w_{max}. Ce principe peut être appliqué aux trois stratégies présentées dans les paragraphes précédents.

2.5 Décodeurs séquentiels basés sur la théorie de Pohst

Dans ce paragraphe, nous allons présenter deux décodeurs séquentiels utilisant la stratégie DFS. Ces décodeurs sont aussi des BB et utilisent comme contrainte une région de recherche finie.

Dans la littérature, on considère principalement deux régions de recherche. La première a été introduite par Pohst [30] et consiste à chercher le point le plus proche à l'intérieur d'une sphère de rayon fixé. La deuxième méthode a été proposée par Kannan et définit la région de recherche à l'intérieur d'un parallélotope [31]. Cependant, la surface du parallélotope étant supérieure à la surface de la sphère, par souci de complexité la méthode de Kannan a été abandonnée.

L'analyse mathématique de la méthode de Pohst a été présentée dans [32]. Dans [33], Viterbo et Boutros ont présenté une implémentation pratique de cette stratégie et l'ont appliquée au décodage pour un canal à évanouissements. Cet algorithme consiste à la recherche du point le plus proche dans une sphère centrée sur le point reçu $\boldsymbol{y_1}$, c'est le décodeur par sphères (SD).

Dans [34], une méthode basée sur le même principe a été proposée par Agrell *et al.* C'est la stratégie de Schnorr-Euchner. Cette dernière effectue des projections successives du point reçu sur les différents hyperplans constituant le réseau de points en respectant les bornes de la sphère afin de n'énumérer que les points du réseau à l'intérieur de la sphère.

Dans le paragraphe suivant, nous allons détailler ces deux algorithmes et comparer leurs propriétés respectives.

2.5.1 Le décodeur par sphères (SD)

Principe du SD

Reprenons le modèle du système MIMO représenté par l'équation (2.17). Notre objectif est de trouver le point le plus proche à l'intérieur d'une sphère de rayon C. Cela se traduit par l'inéquation suivante :

$$\min_{\boldsymbol{s}\in C_s} \|\boldsymbol{y_1} - \boldsymbol{R}\cdot\boldsymbol{s}\|^2 \leq C^2 \tag{2.22}$$

Soient les vecteurs ρ et ξ définis comme suit :

$$\begin{aligned} \boldsymbol{\rho} &= \boldsymbol{R^{-1}}\cdot\boldsymbol{y_1},\ (\rho_1,\ldots,\rho_n)\in\mathbb{R}^n \\ \boldsymbol{\xi} &= \boldsymbol{\rho}-\boldsymbol{s},\ (\xi_1,\ldots,\xi_n)\in\mathbb{R}^n \end{aligned} \tag{2.23}$$

$\boldsymbol{\rho}$ est le point ZF, il représente le coordonné de $\boldsymbol{y_1}$ dans le réseau Λ_R et les $\xi_i, 1\leq i\leq n$ définissent les coordonnées du vecteur $\boldsymbol{s}$ dans le nouveau repère. En remplaçant dans l'inéquation précédente, on obtient :

$$\|\boldsymbol{R}\cdot\boldsymbol{\xi}\|^2 = \sum_{i=1}^{n}(r_{ii}\xi_i + \sum_{j=i+1}^{n} r_{ij}\xi_j)^2 \leq C^2 \tag{2.24}$$

Dans le nouveau système de coordonnées défini par $\boldsymbol{\xi}$, la sphère de rayon C centrée sur le point reçu est transformée en un ellipsoïde centré sur l'origine comme défini par la forme bilinéaire dans l'équation précédente.

Afin de simplifier l'inéquation (2.24), nous posons :

$$\begin{aligned} q_{ii} &= r_{ii}^2 \quad , \quad i = 1, \ldots, n \\ q_{ij} &= \frac{r_{ij}}{r_{ii}} \quad , \quad j = i+1, \ldots, n \end{aligned} \tag{2.25}$$

L'inégalité précédente devient alors :

$$\|\boldsymbol{R} \cdot \boldsymbol{\xi}\|^2 = \sum_{i=1}^{n} q_{ii} (\xi_i + \sum_{j=i+1}^{n} q_{ij} \xi_j)^2 \leq C^2 \tag{2.26}$$

En exploitant la structure triangulaire de la matrice $\boldsymbol{Q} = (q_{ij})_{i,j=1,\ldots,n}$, nous commençons d'abord par traiter les composantes d'indice n :

$$q_{nn} \xi_n^2 \leq C^2 \Leftrightarrow -\frac{C}{\sqrt{q_{nn}}} \leq \xi_n \leq \frac{C}{\sqrt{q_{nn}}} \tag{2.27}$$

Par définition, $\xi_n = \rho_n - s_n$, la double inégalité (2.27) permet de déterminer l'intervalle dans lequel s_n appartient :

$$\left\lceil -\frac{C}{\sqrt{q_{nn}}} + \rho_n \right\rceil \leq s_n \leq \left\lfloor \frac{C}{\sqrt{q_{nn}}} + \rho_n \right\rfloor \tag{2.28}$$

où $\lceil x \rceil$ est le plus petit entier supérieur à x et $\lfloor x \rfloor$ est le plus grand entier inférieur à x. A partir de chaque valeur de s_n, nous pouvons déterminer de même l'intervalle pour s_{n-1} :

$$\left\lceil -\sqrt{\frac{C^2 - q_{n,n}\xi_n^2}{q_{n-1,n-1}}} + \rho_{n-1} + q_{n-1,n}\xi_n \right\rceil \leq s_{n-1} \leq \left\lfloor \sqrt{\frac{C^2 - q_{n,n}\xi_n^2}{q_{n-1,n-1}}} + \rho_{n-1} + q_{n-1,n}\xi_n \right\rfloor \tag{2.29}$$

De la même manière, nous déterminons la i^{me} composante :

$$\left\lceil -\sqrt{\frac{1}{q_{ii}} \left(C^2 - \sum_{l=i+1}^{n} q_{ll} (\xi_l + \sum_{j=l+1}^{n} q_{lj} \xi_j)^2 \right)} + \rho_i + \sum_{j=i+1}^{n} q_{ij} \xi_j \right\rceil \leq s_i$$

$$s_i \leq \left\lfloor \sqrt{\frac{1}{q_{ii}} \left(C^2 - \sum_{l=i+1}^{n} q_{ll} (\xi_l + \sum_{j=l+1}^{n} q_{lj} \xi_j)^2 \right)} + \rho_i + \sum_{j=i+1}^{n} q_{ij} \xi_j \right\rfloor \tag{2.30}$$

Afin de simplifier (2.30), nous posons les variables suivantes :

$$\begin{aligned} S_i &= \rho_i + \sum_{j=i+1}^{n} q_{ij} \xi_j \\ T_i &= C^2 - \sum_{l=i+1}^{n} q_{ll} (\xi_l + \sum_{j=l+1}^{n} q_{lj} \xi_j)^2 \\ &= T_{i-1} + q_{ii} (S_i - s_i)^2 \end{aligned} \tag{2.31}$$

Nous pouvons définir les bornes de chaque intervalle $I_i = [b_{inf,i}, b_{sup,i}]$ correspondant à la composante s_i comme suit :

$$b_{inf,i} = \left\lceil -\sqrt{\frac{T_i}{q_{ii}}} + S_i \right\rceil \leq s_i \leq \left\lfloor \sqrt{\frac{T_i}{q_{ii}}} + S_i \right\rfloor = b_{sup,i} \tag{2.32}$$

En partant de l'ordre n, l'algorithme calcule ainsi toutes les composantes jusqu'à arriver à la composante s_1. L'algorithme trouve alors un premier candidat $s^1 = (s_n, s_{n-1}, s_{n-2}, \ldots, s_1)$. La distance de ce point au point reçu est donnée par :

$$d^2 = C^2 - T_1 + q_{11}(S_1 - s_1)^2 \tag{2.33}$$

L'algorithme est alors recommencé en réduisant cette fois le rayon de la sphère à cette distance.

Ainsi l'algorithme du SD applique la stratégie de recherche DFS. Les noeuds s_i générés par l'algorithme sont compris dans l'intervalle $I_i = [b_{inf,i}, b_{sup,i}]$ ce qui met en oeuvre également la stratégie Branch and Bound. Il est donc inutile de parcourir les points du réseau au-delà de ces bornes.

L'algorithme définit pour chaque composante s_i du vecteur $\boldsymbol{s}$ l'intervalle I_i. Afin de trouver le point le plus proche, l'algorithme parcourt l'intervalle I_i en testant chaque composante et la distance d^2 est évaluée pour toute combinaison de s_i trouvée. Chaque point vérifiant $d^2 \leq C^2$ est stocké. La recherche continue jusqu'à ce que tous les points se trouvant à l'intérieur de la sphère soient parcourus.
A chaque fois qu'un point est trouvé dans la sphère, l'algorithme met à jour le rayon et par la suite les bornes $b_{inf,i}$ et $b_{sup,i}, i = 1, \ldots, n$. Ceci permet de réduire significativement la complexité de la phase de recherche en reprenant à chaque fois la recherche à l'intérieur d'une sphère plus petite.

Cependant, nous notons que la complexité dépend du choix du rayon initial. En effet, un rayon très grand implique un nombre très important de points à parcourir ce qui induit une complexité très élevée. Par contre, un rayon très petit implique une sphère qui risque d'être vide. Dans [35], Hassibi *et al.* ont proposé une formule pour le calcul du rayon optimal en se basant sur des calculs de probabilité. Ce rayon est fonction de la variance du bruit et de la dimension du réseau :

$$C = 2 \cdot n \cdot \sigma_w^2 \tag{2.34}$$

Cette formule permet de définir une sphère autour du point reçu tel qu'on ait au moins un point à l'intérieur de la sphère avec une probabilité égale à 99%. Dans nos simulations, nous considérons un rayon initial défini par cette formule.

Décodage des constellations par le SD

Le SD permet de ne considérer que les points à l'intérieur de la sphère et optimise ainsi la recherche du point le plus proche. Cependant, il ne garantit pas que ces points appartiennent à la constellation considérée. Il est donc impératif de tenir compte de cette condition.

Une idée simple est d'ajouter une routine qui permet de vérifier à chaque fois si le point trouvé appartient à la constellation. Ceci permet de résoudre le problème, toutefois entraine une

augmentation de la complexité.

Une solution possible consiste à imposer une contrainte sur l'intervalle de recherche I_i pour ne parcourir que les points appartenant à la constellation. Ceci se traduit par le calcul des bornes de l'intervalle I_i en tenant compte également des bornes de la constellation. Pour une constellation q-QAM, les symboles s_i appartiennent à l'intervalle $I_c = \left[\pm 1, \pm 3, \ldots, \pm(\sqrt{q}-1)\right]$. Dans ce cas, l'intervalle de recherche qu'il faut considérer est l'intersection entre I_i et I_c.

I_c est un sous-ensemble fini de $2\mathbb{Z}+1$. Afin de ramener la recherche dans $\mathbb{Z}$, nous considérons la transformation suivante :

$$u_i = \frac{s_i + (\sqrt{q}-1)}{2} \tag{2.35}$$

Pour la constellation 16-QAM par exemple, les symboles s_i appartiennent à l'intervalle $I_{c,2\mathbb{Z}+1} = [\pm 1, \pm 3]$. Le nouvel intervalle est $I_{c,\mathbb{Z}} = [0,3] = \{0,1,2,3\}$. On note c_{min} et c_{max} les bornes inférieure et supérieure de cet intervalle. Pour une constellation q-QAM quelconque et en considérant le changement de variable défini dans (2.35), les bornes inférieure et supérieure de la constellation sont définies par :

$$\begin{cases} c_{min} = & 0 \\ c_{max} = & \sqrt{q}-1 \end{cases} \tag{2.36}$$

La recherche du point le plus proche dans ce nouveau repère consiste à énumérer les points u_i dans l'intervalle :

$$I = [b_{inf,i}, b_{sup,i}] \cap I_{c,\mathbb{Z}} \tag{2.37}$$

Lorsque la solution est retournée par l'algorithme, un changement de variable inverse permet de retrouver le vecteur $\boldsymbol{s}$ décidé.

Dans toute la suite, nous sous-entendons ce changement de variable pour tous les décodeurs dans le cas des constellations QAM.

2.5.2 La stratégie de Schnorr-Euchner (SE)

Principe du SE

Agrell *et al.* ont présenté dans [36] une deuxième variante de décodeurs utilisant la stratégie de Pohst. A l'instar du SD, la stratégie de SE est une application de la stratégie de DFS utilisant la recherche en BB puisque la région de recherche est toujours une sphère.

Reprenons le système équivalent de l'équation (2.17). La forme triangulaire de la matrice $\boldsymbol{R}$ du réseau de points permet de voir le réseau comme une superposition de couches ou hyperplans. On se propose d'écrire la matrice $\boldsymbol{R}$ sous la forme suivante :

$$\boldsymbol{R} = \begin{bmatrix} \boldsymbol{R_1} \\ \boldsymbol{r_n} \end{bmatrix} \tag{2.38}$$

où $\boldsymbol{R_1}$ est composée des $n-1$ premières lignes de la matrice $\boldsymbol{R}$. La matrice $\boldsymbol{R}$ est triangulaire, le vecteur $\boldsymbol{r_n} = (0, \ldots, 0, r_{nn})$ est orthogonal à l'espace engendré par la matrice $\boldsymbol{R_1}$.

Le réseau de points $\Lambda_{\boldsymbol{R}}$ peut être vu comme la réunion infinie de sous-réseaux de dimensions $(n-1)$ engendré par la matrice $\boldsymbol{R_1}$:

$$\Lambda_{\boldsymbol{R}} = \cup \{\boldsymbol{c} + t_n \boldsymbol{r_n} / \boldsymbol{c} \in \Lambda_{\boldsymbol{R_1}}, t_n \in \mathbb{Z}\} \tag{2.39}$$

Une projection successive sur les différents hyperplans du réseau de points permet de trouver une première estimation du point le plus proche. Ce point est appelé **"Babai point"**. Il correspond au point ZF-DFE [37]. Une fois ce point trouvé, il constitue le point de départ pour parcourir tous les autres points du réseau. L'objectif de l'algorithme est de trouver le point du réseau le plus proche, il est inutile par la suite de considérer les points dont la distance au point reçu excède celle du point Babai. L'algorithme de SE correspond alors à une recherche à l'intérieur d'une sphère dont le rayon initial est la distance du point Babai au point reçu.

On se propose dans ce qui suit de trouver le point Babai par le calcul. La projection du point reçu $\boldsymbol{y_1}$ sur le vecteur $\boldsymbol{r_n}$ donne la n^{me} composante du point Babai :

$$\hat{s}_n = \frac{y_{1n}}{r_{nn}} \tag{2.40}$$

Soit $s_n = [\hat{s}_n]$ tel que $[x]$ est l'entier le plus proche de x. La distance qui sépare le point $\boldsymbol{y_1}$ de la couche n est donnée par :

$$d_n = |s_n - \hat{s}_n| \cdot |r_{nn}| \tag{2.41}$$

En considérant le vecteur ZF, $\boldsymbol{\rho} = \boldsymbol{R^{-1}} \cdot \boldsymbol{y_1}$, on a :

$$\begin{aligned} \hat{s}_n &= \rho_n \\ d_n &= |[\rho_n] - \rho_n| \cdot |r_{nn}| \end{aligned} \tag{2.42}$$

Pour retrouver $\hat{s}_{n-1}$, on peut partir de l'équation :

$$\boldsymbol{y_1} = \boldsymbol{R} \cdot \boldsymbol{\rho} \tag{2.43}$$

On peut donc écrire :

$$y_{1n-1} = \rho_{n-1} r_{n-1n-1} + \rho_n r_{n-1n} \tag{2.44}$$

La composante s_n ayant été trouvée dans l'étage précédent, nous pouvons utiliser cette estimation pour déduire ρ_{n-1}.

$$\rho_{n-1} = \frac{y_{1n-1} - r_{n-1n} \cdot s_n}{r_{n-1n-1}} \tag{2.45}$$

De la même manière que pour l'ordre n, la composante s_{n-1} est calculée par :

$$\begin{aligned} s_{n-1} &= [\rho_{n-1}] \\ d_{n-1} &= |[\rho_{n-1}] - \rho_{n-1}| \cdot |r_{n-1n-1}| \end{aligned} \tag{2.46}$$

Par substitutions successives, nous pouvons calculer toutes les composantes du point Babai :

$$\begin{aligned} s_i &= \left\lceil \frac{y_{1i} - \sum_{j \geq i} r_{ij} \cdot s_j}{r_{ii}} \right\rfloor \\ d_i &= |[\rho_i] - \rho_i| \cdot \|r_{ii}\| \end{aligned} \tag{2.47}$$

Lorsque le point Babai est trouvé, la distance de ce point au point reçu est donnée par :

$$d^2 = \sum_{i=1}^{n} d_i^2 \tag{2.48}$$

Le principe du SE est de chercher le point le plus proche dans une sphère centrée sur le point reçu en partant du point Babai. L'idée de l'algorithme est de zigzaguer autour de chaque composante du point Babai.
Pour chaque point du réseau trouvé, la distance d^2 est recalculée. Si la nouvelle distance est inférieure à celle trouvée précédemment, le point est stocké et d^2 est mise à jour. La recherche est par la suite recommencée.

Dans l'article original sur la stratégie de SE [36], les auteurs ont considéré un rayon de la sphère initialisé à l'infini, lorsque le point Babai est trouvé, le rayon est mis à jour et ajusté à la distance d^2. Toutefois, dans [38], il a été démontré qu'en considérant un rayon initial fini calculé de la même manière que pour le SD, cela permet d'accélérer le décodage de 30%.

Décodage des constellations par le SE

La stratégie de SE telle que présentée plus haut permet de décoder des réseaux de points avec $\boldsymbol{s} \in \mathbb{Z}^n$. Dans le cas d'utilisation des constellations finies, il faut s'assurer que la solution trouvée est un point de la constellation.

Soit $I_c = [c_{min}, c_{max}]$ l'intervalle auquel appartient les parties réelles et imaginaires des symboles de la constellation. Au niveau de l'algorithme, il faut vérifier que toutes les composantes du point Babai sont prises dans cet intervalle. Pour cela, il faut prendre $s_n = Q_{QAM}(\hat{s}_n)$ à la place de $s_n = [\hat{s}_n]$ tel que :

$$Q_{QAM}(\hat{s}_n) = \begin{cases} c_{min} & , si\, [\hat{s}_n] < c_{min} \\ [\hat{s}_n] & , si\ c_{min} \leq [\hat{s}_n] \leq c_{max} \\ c_{max} & , si\, [\hat{s}_n] > c_{max} \end{cases} \tag{2.49}$$

Nous sous-entendons ici le changement de variable donné dans (2.35) pour travailler dans $\mathbb{Z}^n$ au lieu de $2\mathbb{Z}^n$.

Pour chaque point parcouru, s'il se trouve dans la constellation, la recherche est poursuivie. Si aucun point n'est trouvé dans la constellation, dans ce cas l'algorithme remonte à la composante s_{i+1}.

2.5.3 Comparaison du SD et du SE

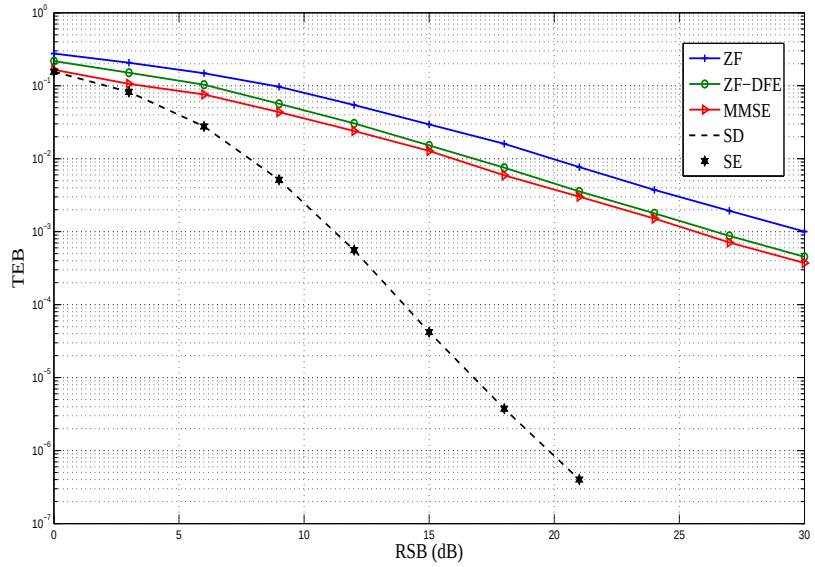
Dans la figure 2.5.a, nous présentons les performances d'un système MIMO 4×4 utilisant une constellation 4-QAM et un multiplexage spatial en taux d'erreur par bit en fonction du rapport signal à bruit. Nous vérifions bien que le SD et le SE permettent d'obtenir les performances optimales (ML). De plus, par comparaison aux décodeurs sous-optimaux, le SD et le SE permettent d'obtenir la diversité maximale égale à n_r.

Dans la figure 2.5.b, nous comparons les complexités, en terme du nombre moyen de multiplications, pour les deux décodeurs. Nous avons considéré ici la complexité de la phase de recherche,

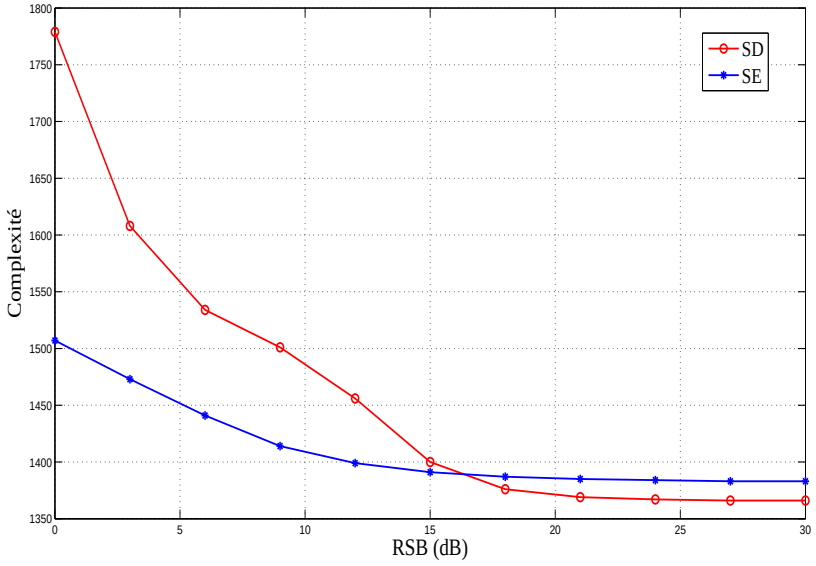
la phase de pré-décodage étant commune aux deux décodeurs. D'après ces résultats, nous pouvons constater que la complexité du SD est plus importante que celle du SE pour les RSB faibles.

La complexité du SD est étroitement liée au choix du rayon. Si celui-ci est très petit, le décodeur ne trouve aucun point et doit alors recommencer la recherche en choisissant un rayon plus grand. Toutefois, puisque le rayon est égal à l'infini pour le SE ou à une valeur assez grande pour renfermer au moins le point ZF-DFE, la convergence de l'algorithme est toujours assurée dès la première itération.

Pour les RSB élevés, la complexité des deux décodeurs convergent. Nous notons également que dans certains cas, la complexité du SD est légèrement plus faible, ceci s'explique par le fait que le point ZF-DFE est plus éloigné du point ML comme le prouve la courbe de performances du décodeur ZF-DFE, dans ce cas le SE qui zigzague autour de ce point, nécessite plus d'itérations pour trouver le point optimal.



(a)



(b)

Figure 2.5 – Comparaisons des performances du SD et du SE pour un système MIMO 4×4 utilisant un multiplexage spatial et une constellation 4-QAM

2.6 Décodeurs séquentiels : Stack et Fano

Une autre variante de décodeurs séquentiels existe dans la littérature. Ces décodeurs ont initialement été introduits pour le décodage des codes binaires en treillis [39]. Ils sont basés sur des algorithmes de recherche dans un arbre où les branches de l'arbre représentent les valeurs binaires du code. Deux principaux algorithmes ont été présentés : le décodeur Fano [40] et le décodeur à piles dit également décodeur "Stack" [41].

Dans [42], les auteurs ont redécouvert ces décodeurs et ils les ont appliqués aux systèmes MIMO représentés par les réseaux de points. La généralisation aux systèmes MIMO se base sur le fait que l'arbre de recherche n'est plus binaire comme dans le cas de l'article original mais formé par les différentes valeurs possibles des symboles s_i. Le principe de décodage reste néanmoins le même.

2.6.1 Le décodeur Fano

Principe du décodeur Fano

Les symboles modulés que nous considérons appartiennent à la constellation QAM. Par la suite les composantes réelles s_i sont prises dans l'intervalle $I_c = [c_{min}, c_{max}]$ (système 2.36). Pour une constellation 16-QAM par exemple, $s_i \in [0, 3]$. A chaque niveau de l'arbre correspond respectivement 4 noeuds. Les chemins dans l'arbre constituent toutes les combinaisons possibles que peut prendre le vecteur $\boldsymbol{s}$.

L'objectif du Fano est de trouver le vecteur ayant le plus petit poids cumulée en utilisant la stratégie BB avec la recherche BeFS. A chaque niveau i , le premier noeud à considérer est le noeud s_i avec la plus faible métrique. De plus, comme tous les algorithmes BB, il faut donc poser une contrainte sur la métrique. Pour le SD et le SE, cette contrainte est le rayon de la sphère. Pour le décodeur Fano, posons T cette contrainte. Les noeuds de l'arbre à considérer doivent donc répondre à la condition :

$$w(s_i) \leq T \tag{2.50}$$

tel que $w(s_i)$ est le poids cumulé au noeud s_i. Cette inéquation permet de définir un intervalle de définition pour les composantes s_i. Toutes les autres composantes ne satisfaisant pas (2.50) sont écartées. Au début de l'algorithme, T est initialisée à 0. Au début de l'algorothme, l'arbre està un noeud fictif s_{root} de poids $w(s_{root}) = 0$.

L'idée originale du Fano est de mettre à jour cette contrainte à chaque fois qu'une nouvelle composante s_i est trouvée, contrairement à l'algorithme classique BB où la contrainte reste fixe tout au long de l'algorithme. Ce principe rappelle l'algorithme du SD et du SE où le rayon de la sphère est mis à jour régulièrement.

Supposons qu'un noeud s_i a été trouvé ($w(s_i) \leq T$). Dans ce cas, la contrainte T est réajustée comme suit :

$$T = T - p \cdot \Delta \tag{2.51}$$

avec Δ le pas de décrémentation de T et $p \in \mathbb{N}$ est le plus grand entier tel que :

$$T - (p+1) \cdot \Delta \leq w(s_i) \leq T - p \cdot \Delta \tag{2.52}$$

Dans les articles originaux sur le Fano, Δ est choisi égal à 1. Toutefois, on peut prendre Δ dans $\mathbb{R}$. La valeur de ce paramètre peut influer sur la vitesse de décodage et par la suite sur la

complexité. L'idéal est de choisir un pas du même ordre de grandeur que les poids $w(s_i)$.

A partir de l'inéquation (2.52), la nouvelle contrainte est choisie de telle sorte que les noeuds suivants aient un poids inférieur ou égal au poids du noeud trouvé. C'est le principe de la recherche BeFS. Si aucun noeud s_{i-1} ne répond à la nouvelle contrainte, l'algorithme remonte au niveau $i+1$ et génère le meilleur noeud s_i autre que celui étant déjà été généré. Si aucun noeud n'est trouvé, cela veut dire que tous les noeuds ont un poids supérieur à T. Dans ce cas, T est incrémenté de Δ, $T = T + \Delta$ et la recherche se poursuit. L'algorithme prend fin lorsqu'un chemin $(s_n, s_{n-1}, s_{n-2}, \ldots, s_1)$ est trouvé.

Les différentes étapes de l'algorithme sont résumées dans l'organigramme ci-dessous.

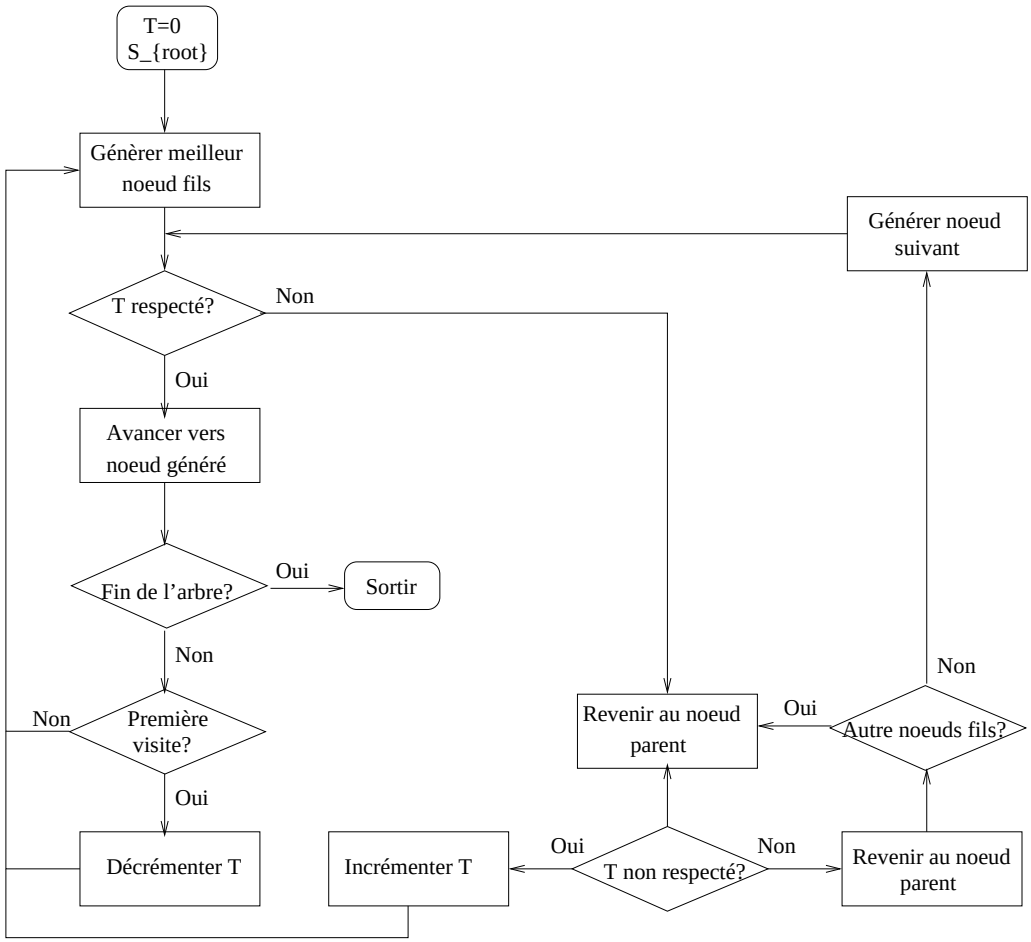


Figure 2.6 – Organigramme du décodeur Fano

Les décodeurs SD et SE procèdent à une mise à jour du rayon de la sphère une fois un vecteur $\boldsymbol{s}$ est trouvé, c'est-à-dire, un chemin complet a été trouvé dans l'arbre. Par contre, la contrainte du Fano est réajustée à chaque fois qu'un noeud intermédiaire s_i est parcouru.

En effet, le décodeur Fano consiste à sélectionner à chaque niveau le sous-chemin le plus court. L'algorithme permet donc de retourner la solution ML. Toutefois, l'inconvénient majeur de cet algorithme est qu'il ne permet de retenir que le chemin courant alors qu'il est possible qu'il remonte vers les niveaux plus hauts dans l'arbre. Il ne dispose pas de mémoire pour stocker les chemins parcourus. L'algorithme peut ainsi revenir plusieurs fois à un chemin ayant déjà été visité, ce qui ajoute une grande complexité et un temps de décodage très long.

2.6.2 Le décodeur Stack

Principe

En 1969, Zigangirov a présenté dans [41] un décodeur à pile (Stack) dans laquelle le décodeur stocke tous les noeuds parcourus dans l'arbre. Le décodeur Stack est une application de l'algorithme BeFS.

En partant du noeud initial s_{root}, l'algorithme génère tous les noeuds fils s_n, calcule leurs poids respectifs et les classe dans la pile selon un ordre croissant en fonction de leurs poids. Le noeud se trouvant en tête de pile est le noeud qui présente le plus faible poids. L'algorithme se place alors dans ce noeud et le remplace par tous ses noeuds fils s_{n-1} dans la pile. La pile est ensuite triée en tenant compte des nouveaux noeuds ajoutés et le nouveau noeud en tête de pile est traité de la même façon dans la prochaine étape.
Lorsqu'un noeud feuille s_1 se trouve en tête de pile, l'algorithme est terminé. Dans ce cas, le chemin de l'arbre $(s_n, s_{n-1}, \ldots, s_1)$ ayant abouti au noeud s_1 est la solution retournée. Elle correspond au vecteur $\boldsymbol{s}$ avec le plus faible poids cumulé. D'après (2.21), le vecteur $\boldsymbol{s}$ est donc la solution ML.

Ces différentes opérations sont résumées dans l'organigramme du décodeur Stack dans la figure 2.7.

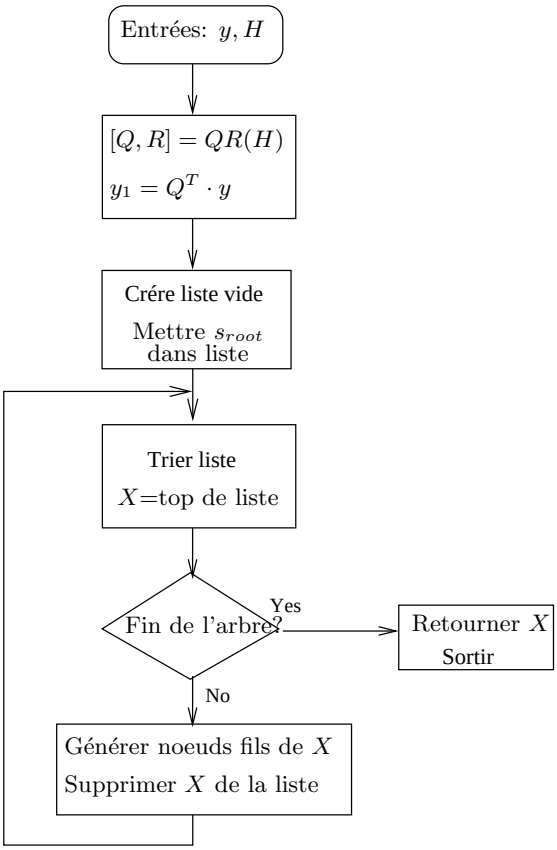


Figure 2.7 – Organigramme du décodeur Stack original

2.6.3 Comparaison du décodeur Fano et du décodeur Stack

A l'instar du décodeur Fano, le Stack peut revenir aux niveaux plus hauts de l'arbre. Cependant, puisqu'il stocke tous les chemins parcourus dans une pile, il ne repasse jamais par le même chemin ce qui évite de regénèrer des noeuds ayant déjà été parcourus. Le prix à payer est une mémoire plus importante, toutefois la complexité ainsi que le délai de décodage sont largement réduits par rapport au décodeur Fano.
Cependant, il est à noter que la complexité des deux décodeurs dépend étroitement de l'arbre considéré. En effet, plus l'arbre est dense plus la complexité est grande.

Dans la figure 2.8, nous avons représenté un système MIMO 4×4 utilisant les constellations 16-QAM et 64-QAM et nous avons tracé les complexités correspondantes calculées en nombre d'opérations de multiplication. Comme il est clair que la complexité du décodeur Fano est plus élevée que celle du décodeur Stack, nous avons choisi de comparer la complexité du SD et du décodeur Stack.

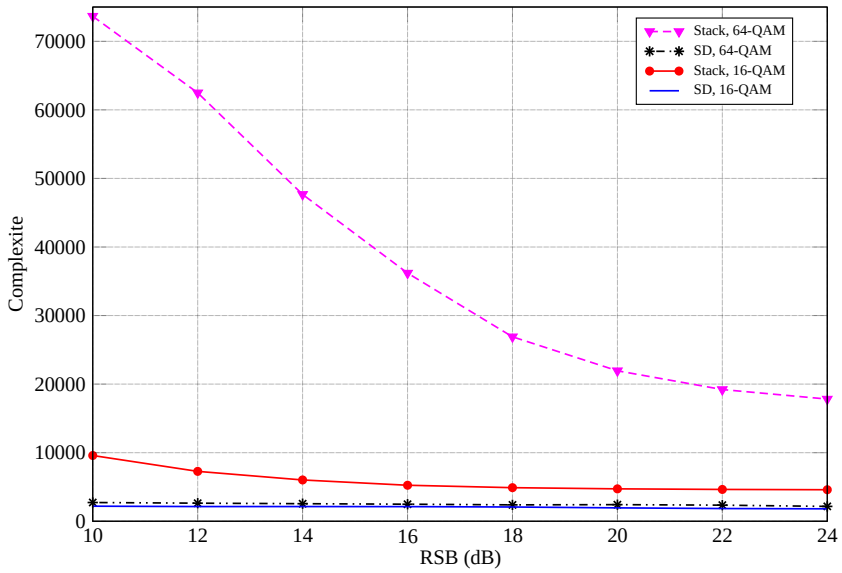


Figure 2.8 – Comparaisons des complexités du décodeur Stack et du SD pour différentes constellations utilisées

A partir de la courbe, nous vérifions par ces résultats qu'en augmentant la taille de la constellation, la complexité du décodeur Stack devient de plus en plus importante. En effet, en considérant la constellation 16-QAM, les symboles s_i appartiennent à l'intervalle $I_{c,16-QAM} = \{0, 1, 2, 3\}$. Chaque noeud de l'arbre dispose alors de 4 noeuds fils. Pour une constellation 64-QAM, les symboles s_i appartiennent dans ce cas à l'intervalle $I_{c,64-QAM} = \{0, 1, 2, 3, 4, 5, 6, 7\}$ et chaque noeud possède 8 fils. L'arbre est alors plus grand et la recherche de la solution dans l'arbre est par la suite plus complexe.

Par comparaison au SD, nous notons qu'en utilisant de petites constellations, la complexité des deux décodeurs sont proches. Par contre, en considérant des constellations de plus grandes dimensions, la complexité du Stack devient très importante par rapport au SD. Nous pouvons conclure que l'utilisation du Stack dans sa version actuelle est non pratique pour le décodage des systèmes MIMO.

2.7 Paramétrisation des décodeurs séquentiels

Les décodeurs séquentiels présentés précédemment, en particulier les décodeurs Stack et Fano sont des décodeurs optimaux. Ceux-ci utilisant la stratégie BeFS, permettent donc de trouver la solution ML dans l'arbre. Nous montrons que ces algorithmes peuvent avoir des performances sous-optimales avec des complexités plus faibles en introduisant un paramètre appelé biais.

Considérons l'équation (2.20) donnant la métrique à minimiser pour tout chemin $s^{(i)}$. La recherche consiste alors à trouver le chemin $\boldsymbol{s^{(1)}} = (s_n, s_{n-1}, \ldots, s_1)$ qui minimise une nouvelle métrique qui dépend du bias.

Nous définissons cette métrique par :

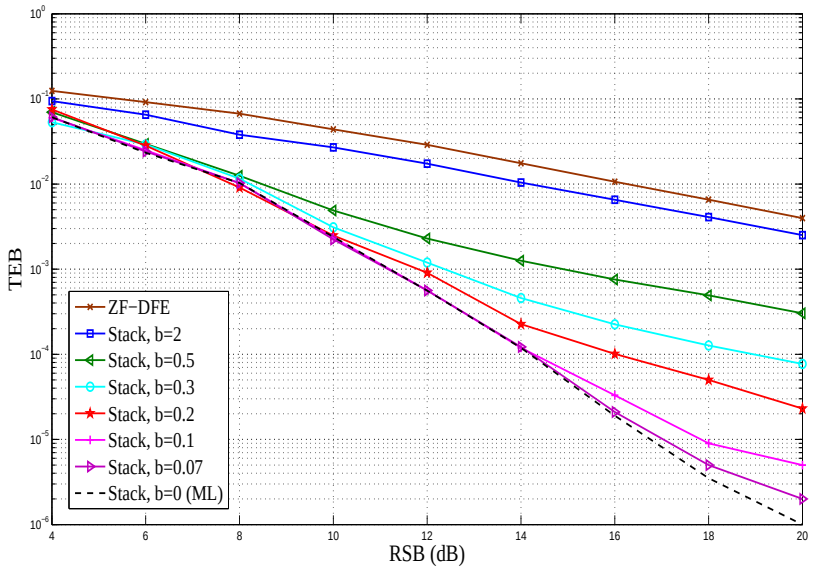
$$w(\boldsymbol{s_i}) = \sum_{j=i}^{n} \quad w_j(s_j) \quad -b \cdot i \ , \ b \in \mathbb{R}^+ \tag{2.53}$$

où b est le biais, chaque chemin dans l'arbre est alors pondéré par la quantité $(-b \cdot i)$. Dans ce cas, l'algorithme favorise les chemins les plus profonds dans l'arbre puisque les chemins les plus courts correspondent à ceux de plus grandes profondeurs i. Par conséquent, le décodeur parcourt moins de noeuds que le décodeur original et la complexité est alors réduite. Toutefois, la solution trouvée ne correspond pas à la solution ML et les performances obtenues sont sous-optimales. Celles-ci dépendent de la valeur choisie pour le biais.

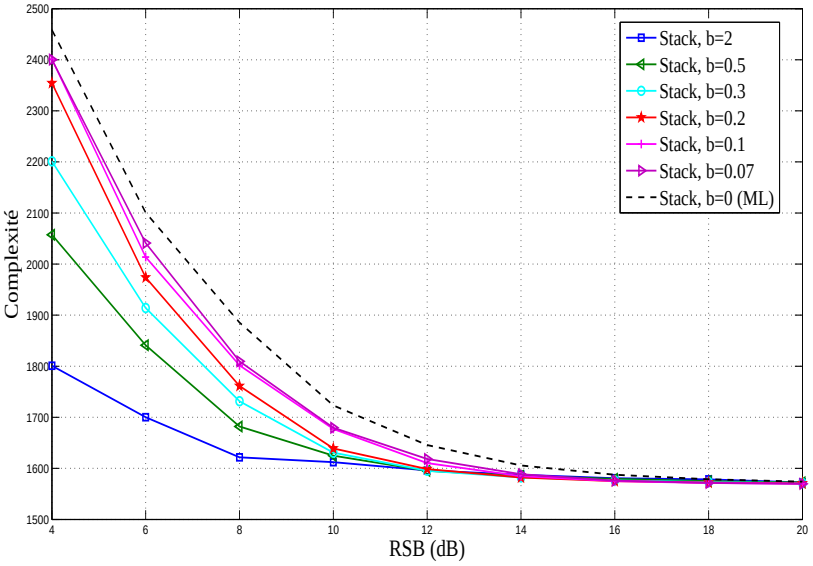
Dans la figure 2.9, nous avons tracé les performances en taux d'erreur par bit ainsi que les complexités obtenues en nombre d'opérations de multiplications pour différentes valeurs de b pour un système MIMO 4×4 utilisant une constellation 4-QAM et un décodage Stack. Nous constatons que plus b est grand, plus les performances sont dégradées, par contre plus la complexité est réduite.

En effet pour des valeurs importantes du biais, le facteur de pondération $(b \cdot i)$ est élevé, l'algorithme parcourt l'arbre quasiment dans le sens de la profondeur jusqu'au noeud final. Autrement dit, pour une valeur élevée de b, l'algorithme parcourt pratiquement un seul chemin dans l'arbre qui correspond au chemin ZF-DFE, les performances sont alors proches de celles du ZF-DFE et la complexité est faible.
Par contre, pour des valeurs plus petites de b, la quantité $(b \cdot i)$ est négligeable et le poids biaisé (2.53) est presque équivalent au poids non biaisé (2.20) qui correspond à la métrique ML. Dans ce cas nous avons des performances proches de ML. Les performances ML sont obtenues pour $b = 0$.

Ainsi, la paramétrisation de l'algorithme de recherche dans un arbre permet de réduire la complexité par rapport à l'algorithme original en offrant une panoplie de courbes de performances variant entre le ZF-DFE et le ML avec des complexités réduites. Par conséquent, différents compromis performances-complexités peuvent être obtenus.



(a)



(b)

Figure 2.9 – Performances et complexités du décodeur Stack paramétré en fonction du biais

2.8 Conclusion

Dans ce chapitre, nous avons passé en revue les principaux décodeurs des systèmes MIMO. Nous avons vu que ces décodeurs peuvent être classés en deux classes : les décodeurs optimaux et les décodeurs sous-optimaux. Nous avons présenté les différents décodeurs les plus connus pour les deux classes. Cependant, nous avons montré que les décodeurs optimaux souffrent d'une grande complexité. Cette complexité est croissante en fonction de la dimension du réseau et de la taille de la constellation utilisée, et rend non pratique leur utilisation pour les systèmes MIMO avec un nombre élevé d'antennes.

Nous avons présenté une version paramétrée de l'algorithme Stack en introduisant un paramètre appelé le biais. L'algorithme paramétré permet de réduire la complexité en fonction de la valeur du biais, toutefois il conduit à des performances sous-optimales.

Pour faire face à ce problème, nous proposons dans le chapitre suivant un nouveau décodeur séquentiel basé sur l'algorithme Stack et le SD. Nous allons montrer que cet algorithme présente une complexité inférieure à tous les décodeurs ML connus, tout en gardant des performances optimales.

Chapitre 3

Décodeur Stack modifié

3.1 Introduction

Dans le deuxième chapitre, un état de l'art des décodeurs les plus utilisés pour les systèmes MIMO a été établi. Les performances mais aussi la complexité sont les deux critères majeurs pour le choix du décodeur. Par exemple, le décodeur ML exhaustif engendre une complexité exponentielle en fonction de la taille du réseau ce qui rend non pratique son utilisation. Les décodeurs séquentiels représentent alors une meilleure alternative.

La simplicité de l'algorithme Stack et sa faible complexité pour les petites constellations nous ont particulièrement motivés à étudier de plus près cet algorithme. Nous proposons dans ce chapitre des modifications du décodeur Stack afin de mieux l'adapter à notre problématique, à savoir le décodage des systèmes MIMO.

Le premier algorithme proposé, appelé M-Stack, combine la méthode de recherche du Schnorr-Euchner avec la stratégie de recherche du Stack. Le deuxième algorithme proposé, appelé, SB-Stack combine la région de recherche du décodeur par sphères avec la stratégie du Stack. Par la suite, nous proposons une modification du SB-Stack afin d'avoir un décodeur à sorties souples qui sera utilisé lors de la concaténation du code espace-temps avec un code correcteur d'erreurs.

3.2 Le décodeur M-Stack

3.2.1 Principe

Comme vu dans le chapitre précédent, les décodeurs originaux Stack et Fano génèrent à chaque niveau de l'arbre tous les noeuds possibles appartenant à la constellation utilisée. De ce fait, la taille de l'arbre et par la suite la complexité de recherche sont d'autant plus grandes que la taille de la constellation augmente. Pour les constellations de grandes tailles, telles que la 64-QAM ou la 256-QAM, la recherche dans l'arbre est équivalente à une recherche dans un réseau infini.

Notre but est alors de diminuer la complexité de recherche. Une solution est de ne considérer qu'une partie de l'arbre et donc de tronquer l'arbre initial pour parcourir un nombre limité de noeuds. Pour avoir les meilleures performances possibles, toute la difficulté réside dans le choix de la région de recherche ou encore la partie de l'arbre à conserver.

Dans la littérature, plusieurs algorithmes ont été proposées tels que les décodeurs QRD-M et le QRD-Stack. Dans la suite, nous proposons un nouveau décodeur, appelé M-Stack. Nous montrons que celui-ci peut également être utilisé pour décoder les réseaux de points infinis.
Nous commençons d'abord par présenter les algorithmes existants.

3.2.2 Le décodeur QRD-M

Le QRD-M (QR Decomposition-M) est un algorithme proposé par Yue et *al.* dans [43] afin de réduire la complexité du décodeur exhaustif. Ce dernier parcourt tous les points de la constellation afin de trouver le point minimisant la métrique ML. Ceci se fait en générant tous les noeuds fils à chaque niveau de l'arbre.

Le décodeur QRD-M permet de restreindre la recherche à un nombre limité de noeuds en ne retenant que les M meilleurs noeuds parmi tous les noeuds générés.
En partant du noeud racine, l'algorithme génère toutes les valeurs possibles de s_n. Leurs métriques respectives sont calculées et classées en ordre croissant. L'algorithme retient alors les M noeuds s_n correspondant aux plus faibles métriques. Ces M noeuds seront par la suite étendus et toutes les valeurs possibles de leurs noeuds fils s_{n-1} générées. Parmi tous les noeuds générés dans le niveau n et niveau $n-1$, M noeuds seront retenus et le reste est écarté. La même procédure est appliquée dans les niveaux suivants jusqu'à atteindre le niveau 1 de l'arbre.

L'algorithme QRD-M peut être résumé comme suit :

Algorithm 3 Algorithme QRD-M

1. $i = n$. Algorithme au noeud racine.
2. Générer tous les noeuds fils possibles de tous les noeuds retenus.
3. Calculer les métriques et les classer selon un ordre croissant dans la pile.
4. Retenir M noeuds ayant les plus faibles métriques.
5. Si $i = 1$ aller à 6. Sinon $i = i - 1$, aller à 2.
6. Retourner le chemin avec la plus faible métrique cumulée. Fin de l'algorithme.

3.2.3 Le décodeur QRD-Stack

Le décodeur QRD-Stack a été présenté dans [44]. Il applique le même algorithme que le Stack original mais en limitant la taille de la pile.
L'algorithme QRD-Stack est différent de l'algorithme QRD-M. Le QRD-Stack commence par générer tous les fils du noeud racine, les classe selon leurs métriques respectives dans une pile et ne retient que les M meilleures valeurs. Le noeud en tête de la pile sera ensuite étendue en noeuds fils. Ceux-ci seront stockés dans la pile en respectant les métriques croissantes. De la même manière, seuls M noeuds seront retenus parmi tous les noeuds de la pile. L'algorithme procède ainsi jusqu'à atteindre la fin de l'arbre.

Ces différentes étapes sont représentées comme suit :

Algorithm 4 Algorithme QRD-Stack

1. $i = n$. Algorithme au noeud racine.
2. Générer tous les noeuds fils possibles du noeud en tête de pile et enlever le noeud parent.
3. Calculer les métriques et les classer selon un ordre croissant dans la pile.
4. Retenir M noeuds ayant les plus faibles métriques. i=niveau auquel appartient le noeud en tête de la pile.
5. Si $i = 1$ aller à 6. Sinon $i = i - 1$, aller à 2.
6. Retourner le noeud en tête de la pile. Fin de l'algorithme.

Par comparaison au décodeur Stack, ce décodeur permet de ne retenir que M noeuds dans la pile et limite ainsi la complexité de la phase de recherche. Par comparaison au décodeur QRD-M, l'algorithme QRD-Stack ne génère pas tous les noeuds fils de chaque noeud retenu mais seulement ceux du noeud en tête de la pile. De plus, le QRD-M permet de générer à la fois tous les noeuds possibles d'un niveau donné de l'arbre avant de passer au niveau suivant. Par contre, le QRD-Stack peut revenir aux niveaux supérieurs (selon le noeud en tête) pour y rajouter des noeuds supplémentaires.
D'un autre côté, il est clair que la complexité du décodeur QRD-Stack est réduite par rapport au décodeur QRD-M en ne considérant que le noeud en tête à chaque fois au lieu de tous les M noeuds de la pile.

Dans [44], il a été démontré par les résultats de simulation que les deux algorithmes présentent les mêmes performances en considérant le même nombre M. Toutefois, en comparant les complexités, il a été observé que le QRD-Stack présente une complexité moindre. De plus, la complexité de ce dernier décroit lorsque le RSB augmente contrairement au QRD-M qui présente une complexité constante.

Dans la suite, nous proposons un nouvel algorithme appelé M-Stack.

3.2.4 Le décodeur M-Stack proposé

Les décodeurs précédents conviennent parfaitement aux cas des constellations finies. Cependant, bien qu'ils limitent la recherche dans l'arbre, l'algorithme doit d'abord générer toutes les valeurs possibles de la constellation pour retenir les M meilleurs noeuds. L'application de tels algorithmes devient alors limitée pour les constellations de grandes tailles et pour les réseaux de points infinis. Dans ce cas particulier, les noeuds prennent leurs valeurs dans $\mathbb{Z}$, l'arbre de recherche possède une infinité de noeuds à chaque niveau (voir figure 3.1). Il est donc impossible de savoir quelles valeurs considérer.

Dans cette approche, nous proposons de combiner la stratégie de Schnorr-Euchner et le Stack original.

Le principe de la stratégie de Schnorr-Euchner est de zigzaguer autour de chaque composante du point Babai (le point ZF-DFE). Nous proposons ici de construire l'arbre de recherche en utilisant la stratégie BeFS. A chaque niveau, on génère M noeuds fils appartenant à un intervalle

centré sur la composante du point Babai. La stratégie BB est appliquée de la même manière que pour le Stack. A savoir, à chaque itération, les noeuds engendrés sont stockés dans une pile, celle-ci est par la suite triée et le noeud se trouvant en tête est remplacée par ses noeuds fils.

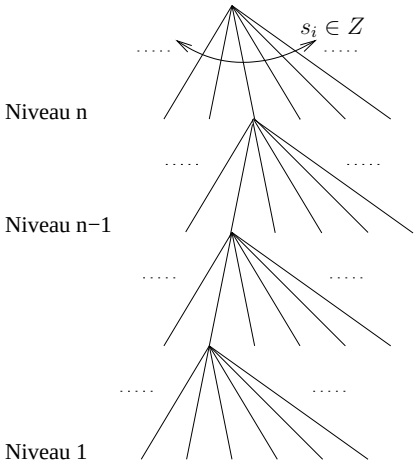


Figure 3.1 – Arbre généré dans le cas d'un réseau de points

Pour mieux expliciter cet algorithme, supposons initialement qu'il est au noeud racine. Le point reçu $\boldsymbol{y_1}$ est projeté sur la n^{me} couche de la matrice $\boldsymbol{R}$ comme suit :

$$z_n = \left[\frac{y_{1n}}{r_{nn}}\right] \tag{3.1}$$

$[x]$ est l'entier le plus proche de x. La composante z_n obtenue est la n^{me} composante du point Babai. L'algorithme énumère M noeuds autour de z_n tels que :

$$s_i = \{z_i, z_i \pm 1, z_i \pm 2, \ldots, z_i \pm ((M-1)/2)\} \tag{3.2}$$

Les composantes s_i ainsi calculées au niveau n seront stockées dans la pile selon un ordre croissant de leurs métriques. La composante en tête sera alors étendue à ses noeuds fils. Pour cela, nous calculons d'abord la composante z_{n-1} en tenant compte de la valeur de s_n trouvée :

$$z_{n-1} = \left[\frac{y_{1n-1} - r_{n-1,n}s_n}{r_{n-1,n-1}}\right] \tag{3.3}$$

Il est à noter que z_{n-1} ne correspond pas forcément à la $(n-1)^{me}$ composante du point Babai puisque le noeud s_n peut être différent du noeud z_n. Les noeuds s_{n-1} sont calculés de la même manière qu'en (3.2) et stockés dans la pile. Les M premiers noeuds sont ensuite retenus et les autres sont rejetés. L'algorithme applique la même démarche pour les niveaux suivants.

Dans la figure 3.2, nous représentons un exemple de construction d'arbre pour $M = 3$.

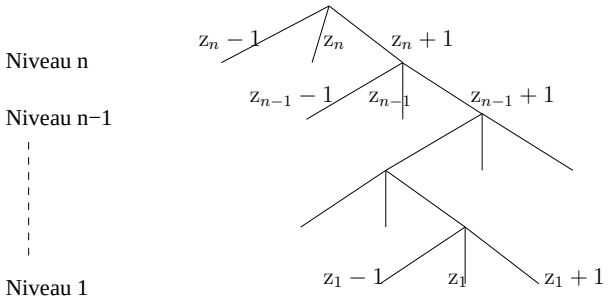


Figure 3.2 – Exemple de construction d'arbre par l'algorithme M-Stack

L'algorithme M-Stack peut être résumé comme suit :

Algorithm 5 Algorithme M-Stack

1. $i = n$. Algorithme au noeud racine. Pile vide.
2. Calculer $z_i = \left\lfloor \frac{y_{1i} - \sum_{j=i+1}^{n} r_{i,j} s_j}{r_{ii}} \right\rceil$. Énumérer M noeuds autour de z_i.
3. Calculer les métriques et les classer selon un ordre croissant dans la pile.
4. Retenir M noeuds ayant les plus faibles métriques. i =niveau auquel appartient le noeud en tête de la pile.
5. Si $i = 1$ aller à 6. Sinon $i = i - 1$, aller à 2.
6. Retourner le noeud en tête de la pile. Fin de l'algorithme.

3.2.5 Décodage des constellations

L'algorithme M-Stack permet de ramener la recherche dans un arbre fini, il reste alors valable dans le cas des constellations finies. Pour cela, il faut ajouter une routine qui permet de tester si la composante calculée appartient à l'intervalle de la constellation QAM (noté I_c). Pour ce faire, il suffit de remplacer l'équation à l'étape 2 de l'algorithme M-Stack ci-dessus par :

$$z_i = QAM\left(\frac{y_{1i} - \sum_{j=i+1}^{n} r_{i,j} s_j}{r_{ii}}\right) \tag{3.4}$$

telle que la fonction $QAM\,(x)$ donne l'entier le plus proche de x dans la constellation. Le zigzag autour de z_i risque de sortir de l'intervalle I_c. Afin de garantir l'aboutissement sur un point de

la constellation, il faudra s'assurer que les tous les noeuds engendrés appartiennent bien à I_c. L'algorithme énumère alors les composantes s_i tel que :

$$s_i = \{z_i, z_i \pm 1, z_i \pm 2, \ldots\} \bigcap I_c \tag{3.5}$$

jusqu'à obtenir M composantes valides.

3.2.6 Choix du nombre de noeuds M

L'algorithme proposé permet de ne parcourir que M noeuds à chaque niveau. De plus, pour les très grandes constellations telle que la 256-QAM, l'arbre devient très dense ce qui rend impossible l'utilisation du décodeur Stack original. La délimitation de la région de recherche permet de réduire le nombre de noeuds parcourus. Cependant, les performances et les complexités dépendent du nombre M choisi. Si M est faible, le point ML n'est pas nécessairement inclus dans l'arbre, la solution retournée est donc sous-optimale. Pour atteindre la solution ML, il faut donc élargir la région de recherche au prix d'une complexité croissante.

Nous avons tracé dans la figure 3.3.a le taux d'erreur par bit pour un système MIMO 4×4 employant un multiplexage spatial pour différentes valeurs de M dans le cas d'un réseau de points. Les symboles d'information sont définis dans l'alphabet infini $\mathbb{Z}^n$.

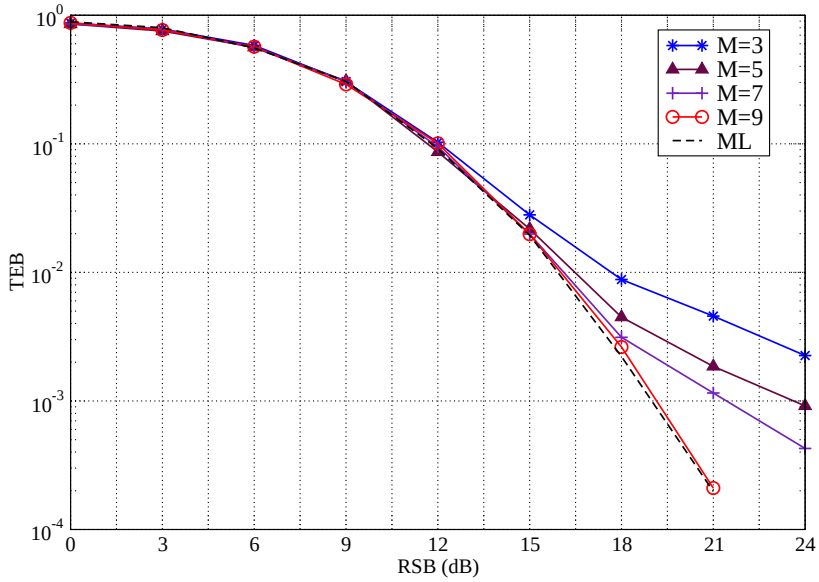
Les courbes montrent une amélioration des performances en augmentant M. Plus M augmente, plus on couvre une région plus importante du réseau et l'algorithme a plus de chance de trouver le point ML. Par contre, la complexité est d'autant plus importante que les performances sont proches du ML comme illustré dans la figure 3.3.b. Nous pouvons établir, dans ce cas, différents compromis performances-complexités.

Dans la figure 3.4.a, nous avons tracé les performances du décodeur M-Stack dans le cas d'une constellation 16-QAM pour un système MIMO 4×4 et nous l'avons comparé au décodeur QRD-Stack. Pour une valeur suffisamment grande de M, les deux algorithmes représentent les mêmes performances et peuvent atteindre les performances ML offertes par le décodeur Stack oiginal. Cependant, pour une valeur faible, le QRD-Stack offre un meilleur taux d'erreur. En effet, à chaque itération, il génère tous les noeuds fils du noeud en tête et en choisit les meilleurs alors que le M-Stack génère simplement les noeuds voisins du point Babai. Les noeuds considérés par le premier algorithme sont alors de meilleurs candidats que ceux produits par l'algorithme M-Stack. Par contre, il est clair que la complexité de l'algorithme QRD-Stack est plus importante puisqu'il commence par calculer tous les noeuds de la constellation à chaque niveau avant de sélectionner les M noeuds nécessaires et rejeter le reste.

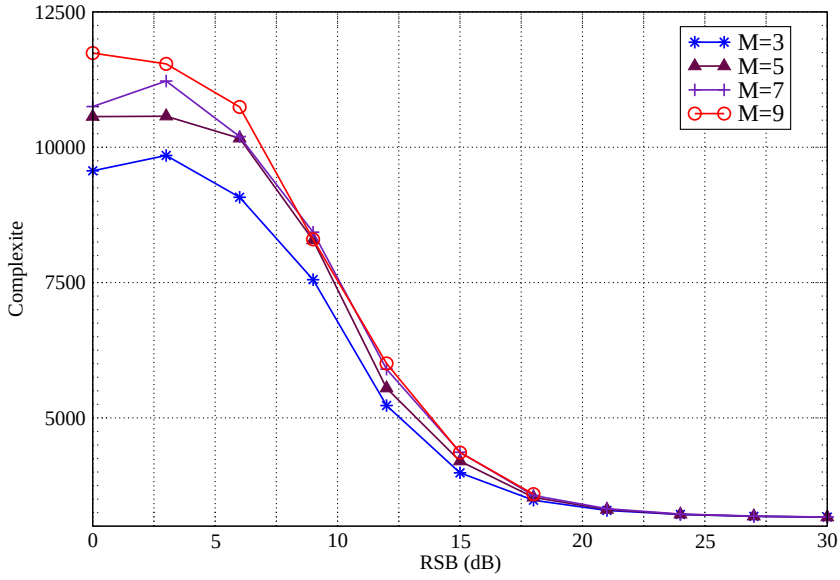
A l'instar du QRD-Stack et du QRD-M, l'avantage de l'algorithme M-Stack est qu'il a la possibilité d'avoir une complexité bornée. En effet, nous pouvons fixer M pour ne pas dépasser un certain seuil pour la complexité ce qui est très intéressant de point de vue pratique. Cependant, l'avantage essentiel du M-Stack par rapport aux premiers décodeurs est qu'il peut être utilisé pour décoder les constellations finies mais aussi un alphabet infini.

Toutefois, pour les faibles RSB, le point Babai sur lequel nous nous basons pour la construction de l'arbre se trouve très éloigné du point ML. Dans ce cas, pour se rapprocher des performances ML, il faut augmenter M. Cependant, il en résulte une augmentation de la complexité.

Le fait de centrer la recherche autour du point Babai n'est pas alors la meilleure solution.

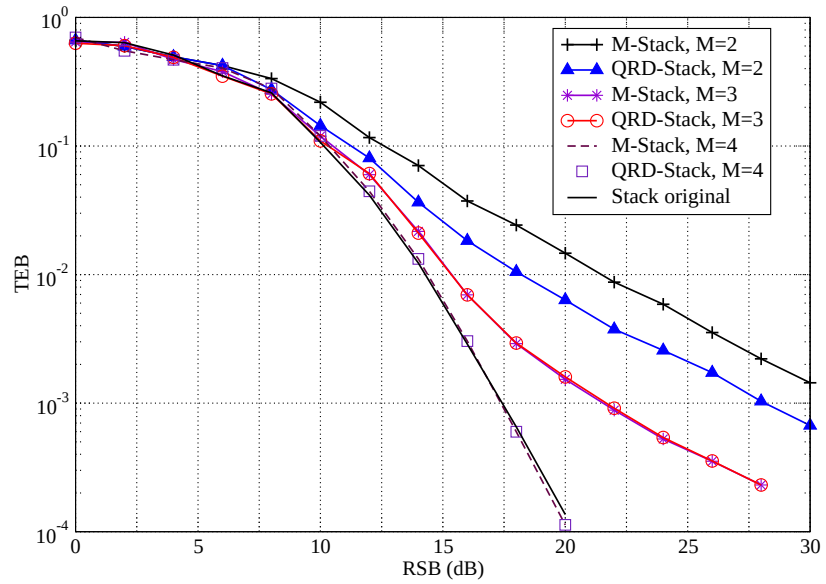


(a)

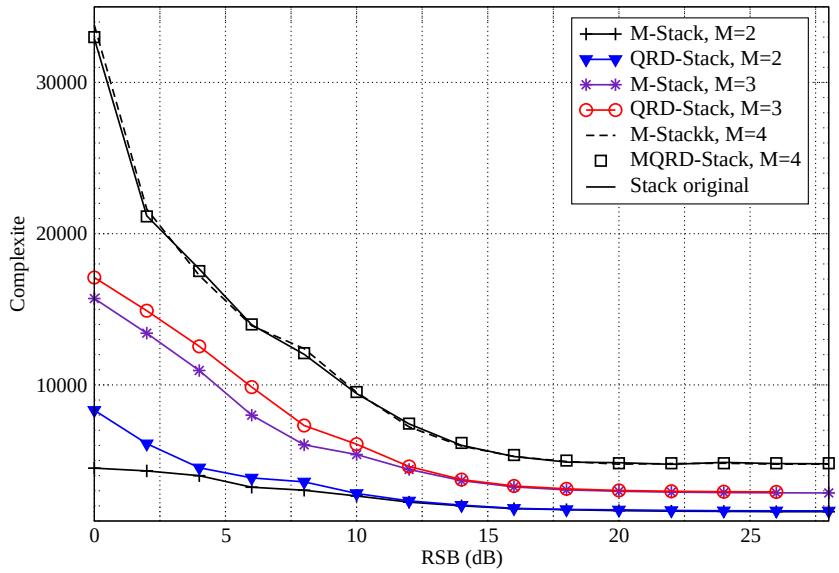


(b)

Figure 3.3 – Performances du décodeur M-Stack pour différentes valeurs de M dans le cas de réseaux de points



(a)



(b)

Figure 3.4 – Performances du décodeur M-Stack pour différentes valeurs de M dans le cas d'une constellation 16-QAM

Dans la suite, nous proposons une deuxième approche de décodage de réseaux de points par l'algorithme séquentiel Stack en considérant une région de recherche plus optimale.

3.3 Le décodeur SB-Stack

3.3.1 Principe de l'algorithme

Le décodage ML consiste à chercher le point le plus proche du point reçu appartenant au réseau de points. La recherche exhaustive qui consiste à parcourir tous les points du réseau est très complexe. Pour décoder le vecteur reçu, il est nécessaire de définir une région finie du réseau. Dans le paragraphe précèdent, nous avons considéré la région du réseau définie autour du point Babai. Notre objectif était de considérer ce point comme une première estimation de la solution, puis d'améliorer cette estimation en cherchant le point le plus proche du point reçu parmi les voisins du point Babai.

Nous proposons ici un nouvel algorithme qui combine la région de recherche du décodeur par sphères (SD) et la stratégie de recherche du Stack. Nous avons appelé cet algorithme le SB-Stack (Spherical Bound-Stack). Pour cela, nous considérons une sphère de rayon fixé construite autour du point reçu. Les points à considérer sont les points du réseau contenus dans cette région. La recherche dans l'arbre sera donc restreinte aux noeuds correspondant à ces points. On aura donc un arbre fini et l'algorithme Stack original peut s'appliquer.

3.3.2 Décodage de réseaux de points par l'algorithme SB-Stack

La solution est à chercher à l'intérieur d'une sphère de rayon C. En faisant le même calcul que dans le SD, nous définissons les intervalles de recherche pour les composantes s_i donnés par :

$$b_{inf,i} = \left\lceil -\sqrt{\frac{T_i}{q_{ii}}} + S_i \right\rceil \leq s_i \leq \left\lfloor \sqrt{\frac{T_i}{q_{ii}}} + S_i \right\rfloor = b_{sup,i} \tag{3.6}$$

Nous rappelons que $\boldsymbol{T}$ et $\boldsymbol{S}$ sont donnés par les équations (2.31) comme suit :

$$\begin{aligned} S_i &= \rho_i + \sum_{j=i+1}^{n} q_{ij}\xi_j \\ T_i &= C^2 - \sum_{l=i+1}^{n} q_{ll}(\xi_l + \sum_{j=l+1}^{n} q_{lj}\xi_j)^2 \\ &= T_{i-1} + q_{ii}(S_i - s_i)^2 \end{aligned} \tag{3.7}$$

En partant du noeud racine, l'algorithme du SB-Stack commence par calculer les bornes $b_{inf,n}$ et $b_{sup,n}$ selon l'équation (3.6). Tous les noeuds s_n de l'arbre définis à l'intérieur de cet intervalle sont alors générés. Chaque composante s_n est calculée avec son poids respectif et stockée dans la pile selon un ordre croissant en fonction des poids. Pour générer les noeuds fils s_{n-1} du noeud se trouvant en tête de la pile, nous commençons par calculer les bornes $b_{inf,n-1}$ et $b_{sup,n-1}$. Les chemins correspondants aux couples (s_n, s_{n-1}) sont alors stockés dans la liste avec leurs poids cumulés respectifs. Ces opérations sont répétées jusqu'à obtenir un chemin complet de longueur

n en tête de la pile. La solution retournée est alors le vecteur $(s_n, s_{n-1}, \ldots, s_1)$.

D'après l'équation (3.7), nous notons que les bornes à un niveau i dépendent des bornes du niveau précédent. Par conséquent, l'algorithme SB-Stack stocke, en plus des poids, les entités T_i et S_i de chaque noeud parent ce qui permet un calcul plus rapide des intervalles de recherche dans les prochaines itérations.

Remarque1
Lorsqu'à un niveau donné, l'algorithme ne trouve aucun noeud valide s_i, c'est-à-dire que $b_{inf,i} \geq b_{sup,i}$, ceci implique que le chemin défini à ce niveau par $\boldsymbol{s^{(i)}} = (s_n, s_{n-1}, \ldots, s_{i+1}, s_i)$ n'existe pas dans l'arbre et que ce noeud ne correspond pas à un point dans la sphère. Dans ce cas, l'algorithme retire le noeud prédécesseur s_{i+1} de la pile et l'algorithme est repris en considérant le nouveau noeud en tête de la pile.

Remarque2
Si la pile est vide, ceci implique qu'aucun chemin valide n'a été trouvé dans l'arbre. Autrement, aucun point n'a été trouvé dans la sphère. Dans ce cas, le rayon de la sphère est augmenté et la recherche est recommencée.
Pour assurer la convergence de l'algorithme, le rayon initial est choisi selon la formule donnée dans (2.34). Ce rayon garantit d'avoir au moins un point dans la sphère.

Remarque 3
Contrairement au SD, le rayon de la sphère ne peut pas être ajusté au cours de la recherche. Étant donné que le décodeur SB-Stack utilise la stratégie BeFS, l'algorithme peut retourner aux niveaux précédents dans l'arbre. Au cours de la recherche, nous n'obtenons pas un chemin complet qui peut représenter une estimation de la solution retournée. De ce fait, nous n'avons aucune information sur la distance de cette solution au point reçu, le rayon initial est donc maintenu jusqu'à la fin de l'algorithme. Par contre, le SD applique la stratégie DFS qui cherche d'abord à trouver un chemin complet dans l'arbre, nous avons donc un premier candidat dont nous évaluons la distance au point reçu. L'algorithme est donc poursuivi en ne tenant compte que des points du réseau qui sont plus proches de ce premier candidat. Autrement dit, nous cherchons des candidats éventuels dans une nouvelle sphère de rayon maximal ajusté à la distance de ce point au point reçu.

3.3.3 Organigramme du SB-Stack

L'organigramme suivant résume l'algorithme du SB-Stack proposé plus haut.

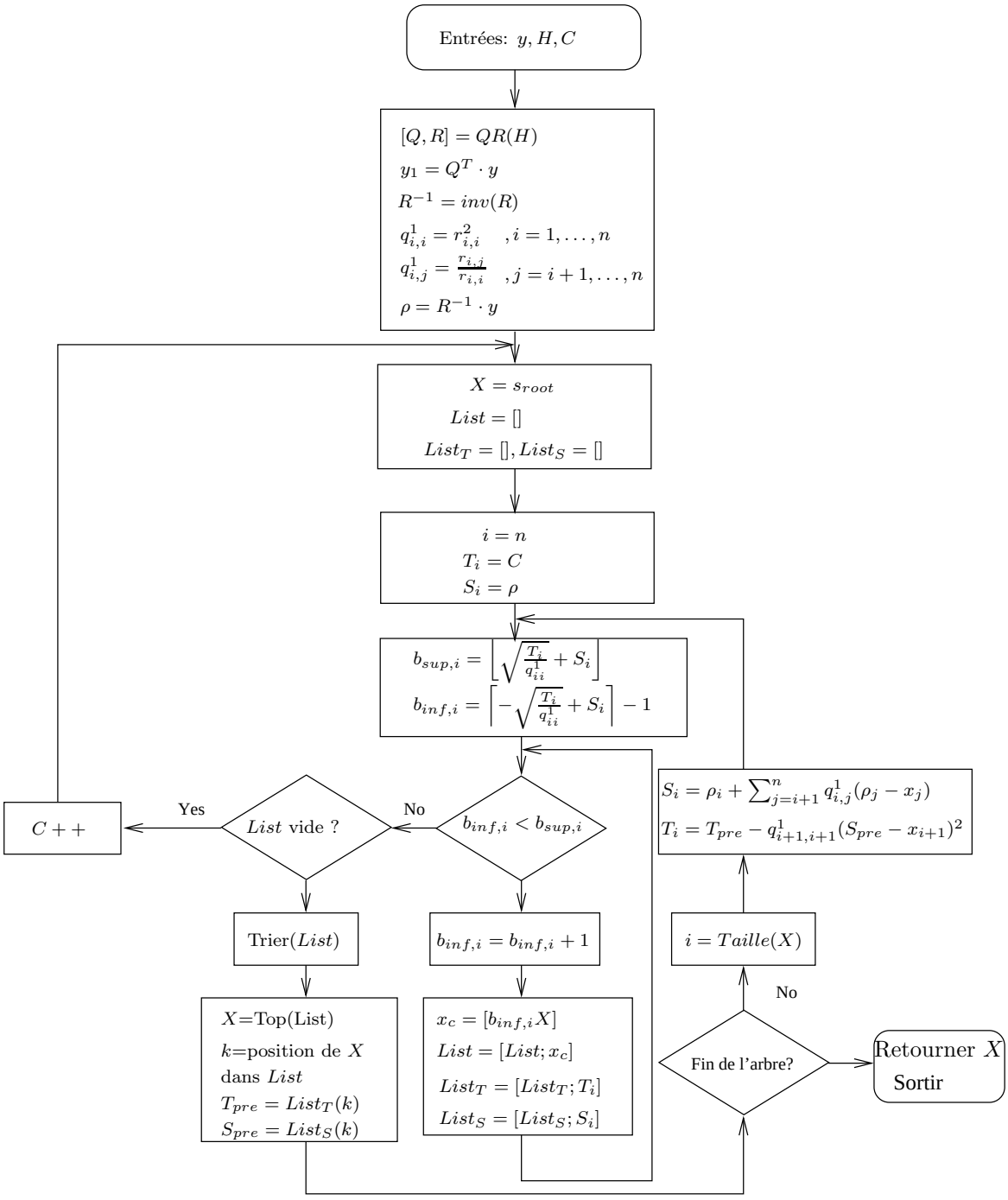


FIGURE 3.5 – Organigramme du SB-Stack

3.3.4 Décodage des constellations par l'algorithme SB-Stack

Dans l'algorithme SB-Stack cité plus haut, aucune contrainte n'est faite sur les symboles d'information $\boldsymbol{s}$. Le décodeur SB-Stack peut donc être appliqué pour le décodage des ensembles infinis

comme pour le décodage des sous-parties finies du réseau.

Dans le cas de constellations finies, les points $\boldsymbol{s}$ considérés par le décodeur sont définis par les points du réseau qui appartiennent à l'intersection de la sphère avec la constellation choisie. Pour cela, une contrainte est ajoutée dans le calcul des bornes de $\boldsymbol{s}$. En tenant compte de la constellation et en considérant le changement de variable définie dans l'équation (2.35) pour s_i, les symboles à considérer dans l'arbre de recherche sont alors définis par :

$$b_{inf,i} = \sup\left(\left\lceil -\sqrt{\frac{T_i}{q_{ii}}} + S_i \right\rceil, c_{min}\right) \leq s_i \leq \inf\left(\left\lfloor \sqrt{\frac{T_i}{q_{ii}}} + S_i \right\rfloor, c_{max}\right) = b_{sup,i} \tag{3.8}$$

où c_{min} et c_{max} représentent les bornes de la constellation définies dans (2.36).

3.3.5 Paramétrisation du décodeur SB-Stack

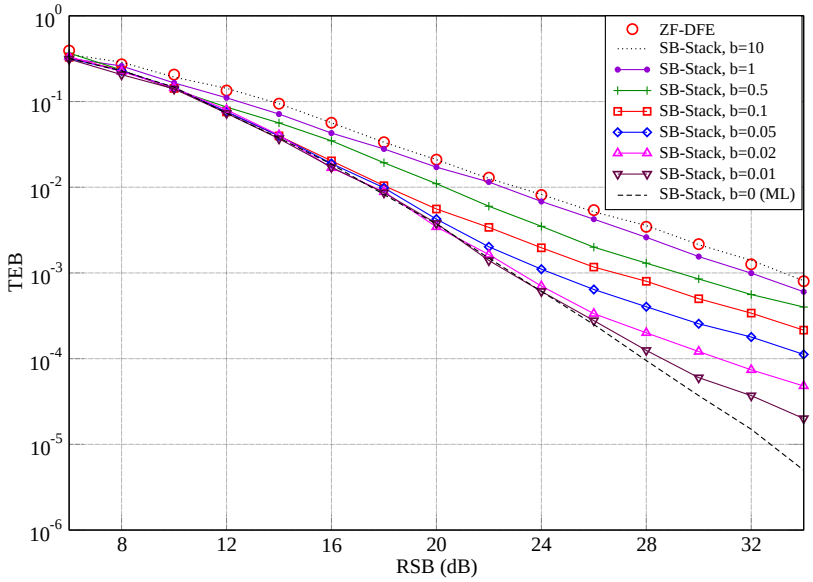
Tel que présenté plus haut, le décodeur SB-Stack proposé utilise la même stratégie de recherche que le Stack original. A l'instar de ce dernier, une paramétrisation peut alors être appliquée à l'algorithme SB-Stack afin de réduire la complexité. Ceci se fait de la même manière qu'en section 2.7. en introduisant un biais dans le calcul des métriques. Le calcul des bornes inférieures et supérieures dans les équations (3.6) et (3.8) reste inchangé.

Nous avons tracé dans la figure 3.6 les performances et les complexités du décodeur SB-Stack paramétré pour différentes valeurs du biais pour un système MIMO 2×2 utilisant une constellation 16-QAM et un multiplexage spatial.

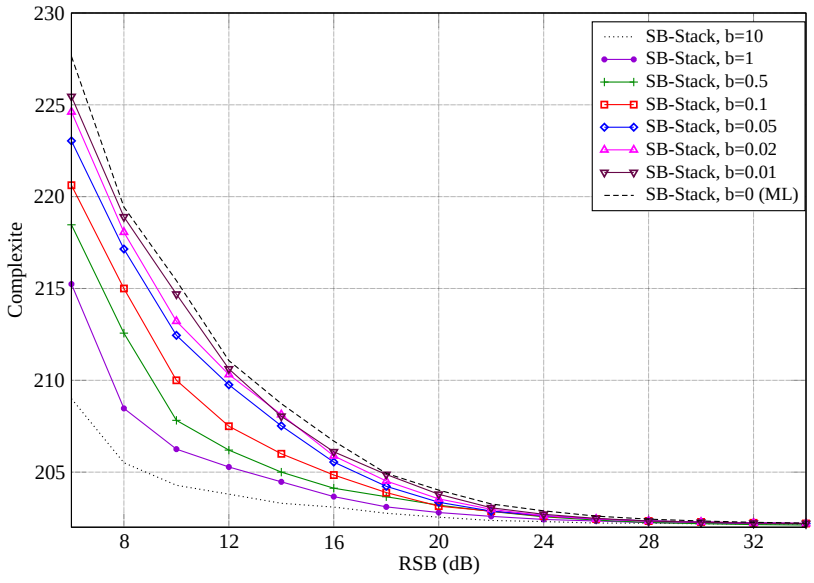
Lorsque la valeur du biais est égale à zéro, le décodeur SB-Stack proposé cherche le point le plus proche dans la sphère en minimisant la métrique ML. La solution retournée par l'algorithme est forcément la solution ML et les performances obtenues sont par la suite optimales. Le décodeur SB-Stack est par conséquent équivalent au décodeur ML pour un biais nul.

Lorsque la valeur du biais est très grande, le SB-Stack parcourt l'arbre dans le sens de la profondeur en passant du niveau i au niveau $i-1$ jusqu'à atteindre la base. La solution trouvée est donc la solution ZF-DFE. Le SB-Stack correspond dans ce cas au décodeur ZF-DFE.

Pour des valeurs intermédiaires du bias, nous vérifions une fois encore que plus b décroit, plus les performances sont améliorées, au prix d'une plus grande complexité. Une multitude de courbes avec un écart de 1dB peuvent être obtenues.
La complexité du SB-Stack paramétré est cependant plus faible que le Stack original grâce à la délimitation de la région de recherche à l'intérieur de la sphère.



(a)



(b)

Figure 3.6 – Performances et complexités du décodeur SB-Stack en fonction du biais

3.4 Comparaison du SB-Stack avec les décodeurs de réseaux de points

Après avoir présenté l'algorithme SB-Stack, nous nous proposons dans la suite de le comparer aux décodeurs existants dans la littérature. Nous avons vu que cet algorithme utilise la région de recherche du SD tout en gardant la stratégie de recherche du Stack original. Nous nous basons alors dans notre étude sur ces différents décodeurs.

3.4.1 Comparaison du SB-Stack et du décodeur Stack original

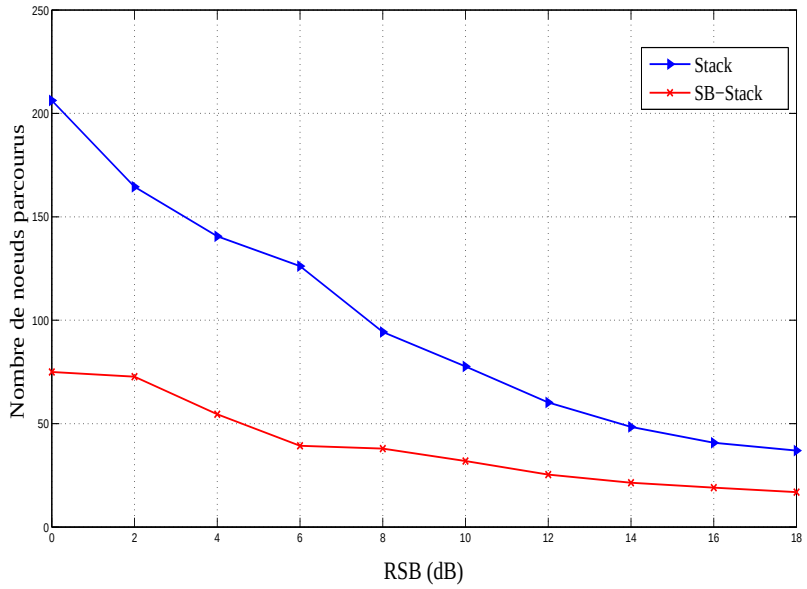
Comparaison en nombre de noeuds parcourus

Le principe du décodeur Stack original et le décodeur SB-Stack est de chercher le point le plus proche du point reçu. Comme nous l'avons vu précédemment, la stratégie de recherche dans l'arbre reste la même. De point de vue performances, tous les deux permettent de retrouver la solution ML. Toutefois, la différence majeure entre ces algorithmes est l'arbre considéré.
En effet, pour le décodeur Stack, l'arbre est constitué par toutes les combinaisons possibles des symboles d'informations $\boldsymbol{s}$ de la constellation considérée. A chaque niveau de l'arbre, les composantes du vecteur sont définis par :

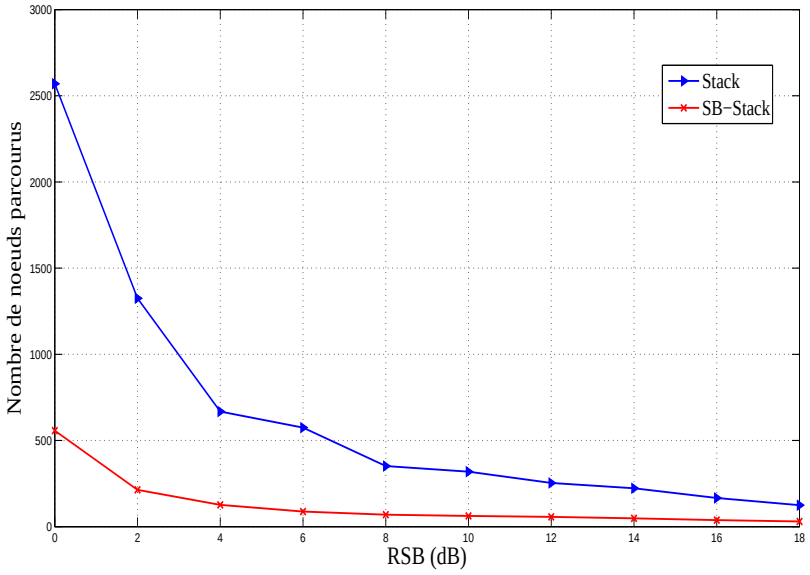
$$c_{min} \leq s_i \leq c_{max} \tag{3.9}$$

La taille de l'arbre et par conséquent la complexité de la phase de recherche dépend de la constellation choisie. Par exemple, pour une constellation $q - QAM$, chaque noeud de l'arbre génère $k = \log_2(q)$ noeuds fils. Par contre, le décodeur SB-Stack permet de réduire la recherche dans une sphère au lieu de chercher dans toute la constellation. Les composantes sont définies dans ce cas dans l'intervalle (3.8). En tenant compte de cette nouvelle contrainte, l'arbre est uniquement constitué des points les plus proches du vecteur reçu. L'algorithme SB-Stack permet d'écarter tous les points qui se trouvent à une distance supérieure au rayon de la sphère. Pour la même constellation utilisée, l'arbre est ici moins dense et la recherche est donc moins complexe.

Dans la figure 3.7, nous avons tracé le nombre total de noeuds parcourus par les deux algorithmes en fonction du rapport signal à bruit pour un système MIMO 4×4 utilisant le multiplexage spatial et les constellations $16 - QAM$ et $64 - QAM$. Nous pouvons vérifier pour les deux cas que le décodeur SB-Stack permet de parcourir moins de noeuds que le Stack original. Le gain en nombre de noeuds visités est en moyenne de l'ordre de 60% pour la 16-QAM. Pour une 64-QAM, il est de l'ordre de 80%. On déduit donc que plus la taille de la constellation augmente, plus il est intéressant d'utiliser le décodeur SB-Stack.



(a) 16-QAM



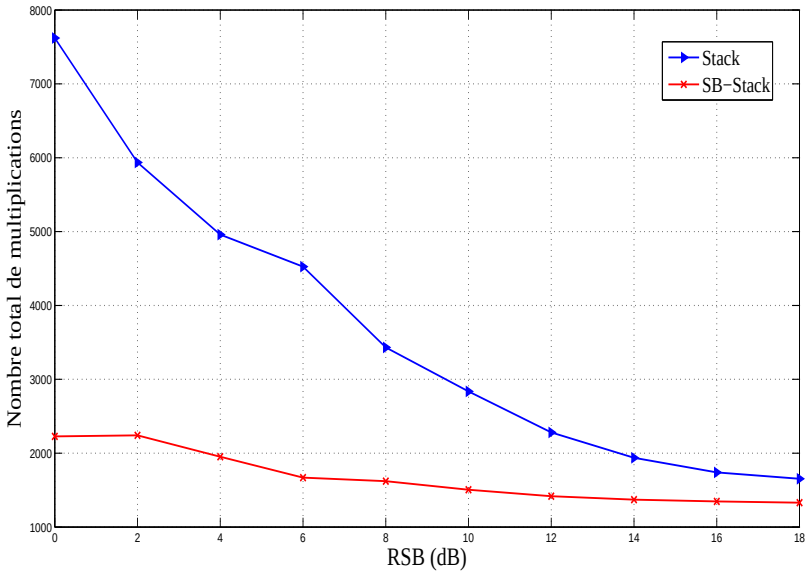
(b) 64-QAM

Figure 3.7 – Comparaison du nombre de noeuds parcourus par les décodeurs Stack et SB-Stack

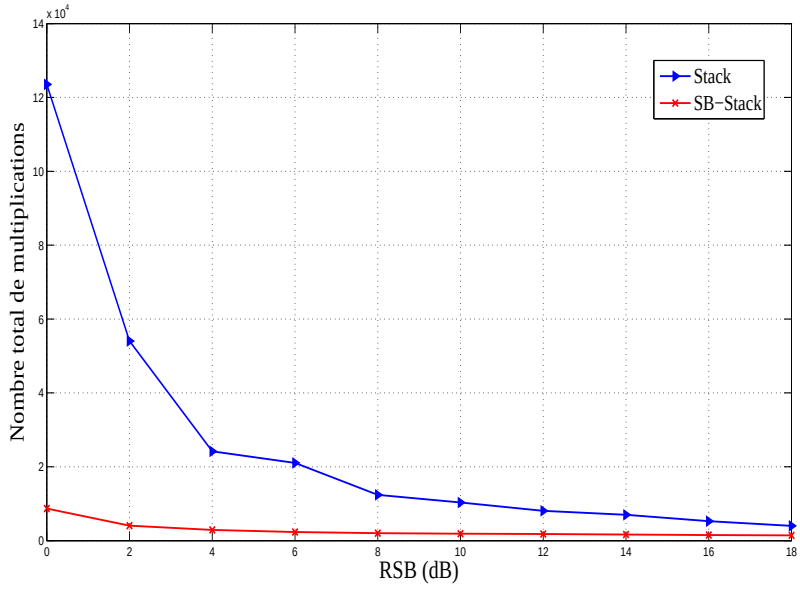
Comparaison en complexité de calcul

En ajoutant la contrainte que les noeuds parcourus doivent appartenir à la sphère, le décodeur SB-Stack doit calculer les bornes inférieures et supérieures de chaque noeud généré. Le calcul du noeud dans l'algorithme SB-Stack nécessite plus d'opérations que dans le décodeur Stack. En outre, ces bornes sont stockées dans la pile pour calculer les intervalles de recherche pour les niveaux suivants. Le décodeur SB-Stack stocke alors plus d'informations et requiert plus de mémoire.
Toutefois, la réduction importante du nombre de noeuds visités permet de compenser cette complexité rajoutée.

En considérant le même système que précédemment, nous avons représenté dans la figure 3.8, la complexité définie par le nombre d'opérations de multiplication nécessairs dans les deux algorithmes.
Nous notons que le gain en complexité du SB-Stack par rapport au décodeur Stack est de 46% dans le cas d'une constellation $16-QAM$ et de 78% pour la constellation $64-QAM$.
Ce pourcentage est la moyenne des gains obtenus pour les différents RSB considérés. Nous notons qu'il est légèrement inférieur au gain calculé pour le nombre de noeuds parcourus donné précédemment.



(a) 16-QAM



(b) 64-QAM

Figure 3.8 – Comparaison des complexités des décodeurs Stack et SB-Stack

3.4.2 Comparaison du décodeur SB-Stack et du SD

Similairement au SD, l'algorithme SB-Stack réduit la recherche du point le plus proche à l'intérieur d'une sphère centrée sur le point reçu. Les deux algorithmes considèrent alors les mêmes points du réseau. Toutefois, ils différent dans la stratégie de parcours de ces points.

Stratégies de parcours

Pour chaque composante s_i du vecteur $\boldsymbol{s}$, le SB-Stack parcourt l'intervalle $I_i = [b_{inf,i}, b_{sup,i}]$ de bout en bout. Le noeud dans la pile avec la plus faible métrique est sélectionné et tous ses noeuds fils générés. Cette stratégie correspond donc à la stratégie de recherche BeFS. Le noeud sélectionné peut être à un niveau différent du niveau courant i et l'algorithme peut retourner à un niveau supérieur dans l'arbre.

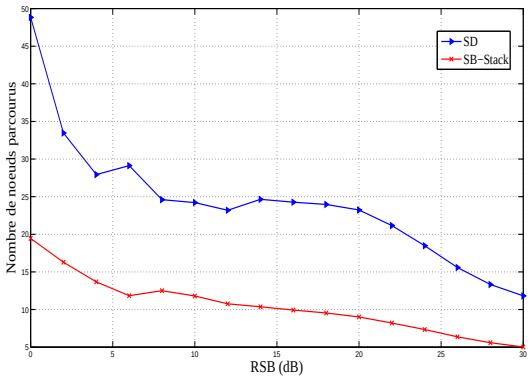
Le SD parcourt, pour chaque composante, le même intervalle I_i mais sélectionne le premier noeud valide s_i correspondant à $b_{inf,i}$. Celui-ci sera alors étendu à la prochaine itération et tous ses noeuds fils calculés jusqu'à atteindre la base de l'arbre (niveau 1). Une fois un chemin $s^{(1)} = (s_n, s_{n-1}, \ldots, s_1)$ est trouvé, l'algorithme met à jour le rayon de la sphère et l'algorithme est repris en considérant une sphère plus petite. Cette stratégie correspond à la stratégie DFS. L'ordre de parcours n'est donc pas le même pour les deux algorithmes.

Comparaison des complexités

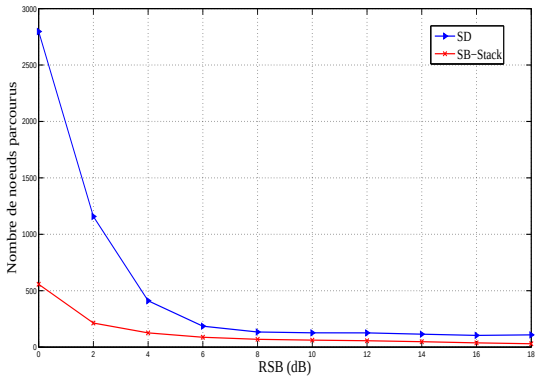
La méthode de recherche BeFS étant plus optimale que la méthode DFS, nous pouvons noter que le décodeur SB-Stack permet, grâce à sa stratégie de recherche, de parcourir moins de noeuds que le SD. Bien que les deux décodeurs considèrent la même région de recherche, le SB-Stack permet d'avoir des complexités inférieures à celles du SD.

Dans la figure 3.9, nous avons tracé la complexité en nombre de noeuds parcourus pour un système MIMO utilisant une constellation $64 - QAM$ en considérant différentes dimensions du système. Nous constatons que le SB-Stack permet d'avoir un gain moyen minimal de 57% par rapport au SD. Nous notons que ce gain est d'autant plus important que la dimension du système augmente. Il est égal à 57%, 63% et 70% respectivement pour les systèmes MIMO 2×2, 4×4 et 6×6.

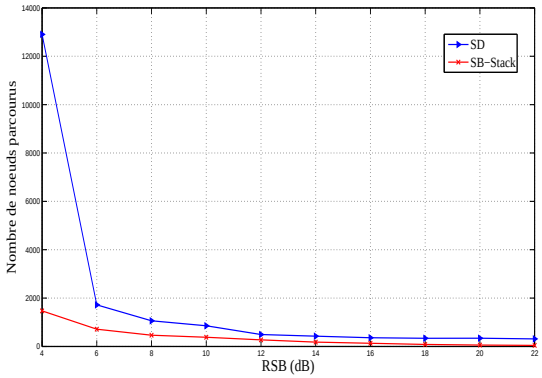
Ce gain est en particulier observé à faibles RSB. En effet, les noeuds ayant des poids très proches, les deux algorithmes testent alors plus de noeuds. Par conséquent, le gain cumulé est plus important.



(a) 2x2



(b) 4x4

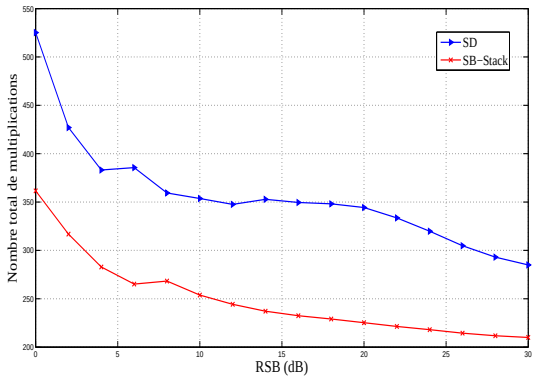


(c) 6x6

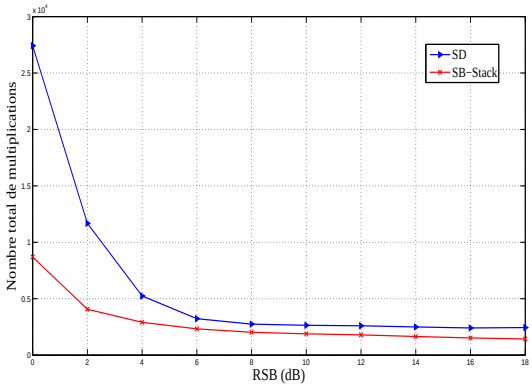
Figure 3.9 – Comparaison du nombre de noeuds parcourus par le SD et le SB-Stack pour différentes dimensions du système MIMO

Cependant, pour un noeud généré, le décodeur SB-Stack effectue plus d'opérations arithmétiques que le SD. Toutefois, la complexité totale reste inférieure à celle du SD et ceci grâce à la réduction du nombre de noeuds parcourus.

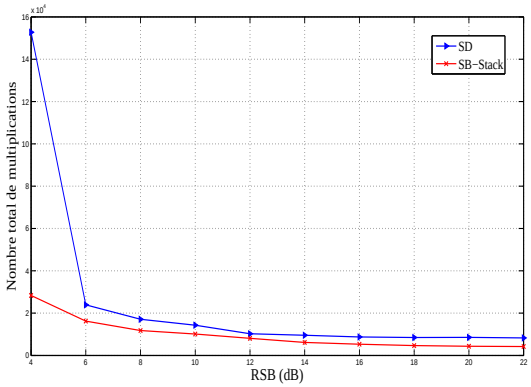
Ce résultat est vérifié dans la figure 3.10 où nous avons présenté la complexité calculée en nombre d'opérations de multiplication. Le SB-Stack permet d'obtenir un gain moyen d'au moins 30% par rapport au SD pour le cas d'un système MIMO 2×2. Nous vérifions également que ce gain augmente avec la taille du réseau.



(a) 2x2



(b) 4x4



(c) 6x6

Figure 3.10 – Comparaison des complexités des décodeurs SD et SB-Stack pour différentes dimensions du système MIMO

3.5 Application du décodeur SB-Stack au décodage à sorties souples

Nous avons proposé précédemment un nouvel algorithme séquentiel pour le décodage des systèmes MIMO. Par comparaison aux décodeurs existants, nous avons montré que cet algorithme présente une complexité plus réduite tout en gardant les performances ML. Ce décodeur SB-Stack a été proposé dans le cas d'un décodage à sorties dures (Hard decoding).

Toutefois, pour un système utilisant un codage de canal, tels que les Turbo codes [45], les codes LDPC [46] et les codes convolutifs [47], le décodeur MIMO à sorties binaires (dures) est remplacé par un décodeur de codes correcteurs d'erreurs à sorties pondérées (souples) afin d'augmenter le gain apporté par le codage. Le décodeur SB-Stack utilisé en amont doit dans ce cas être adapté afin de fournir des sorties souples. Nous nous proposons dans cette partie

d'apporter les modifications nécessaires au décodeur SB-Stack précédent pour satisfaire cette contrainte. Nous commençons d'abord par donner le principe du décodage à sorties souples.

3.5.1 Principe du décodage à sorties souples

Le décodage « hard » ou à sorties dures consiste à retourner à la sortie du décodeur une valeur estimée du symbole transmis en utilisant un seuillage. Le décodage souple permet, quant à lui, de retourner une information sur la probabilité de réalisation de chaque bit. En général, le décodage souple peut être réalisé en utilisant la notion de probabilité aposteriori (APP : aposteriori probability). Le principe consiste à maximiser la probabilité d'un bit donné et le APP est souvent exprimé en taux de vraisemblance logarithmique (LLR : Log-Likelihood-Rate). Par la suite, un seuillage est réalisé sur le signe du LLR calculé. Si le LLR est positif, le bit décodé b_i est alors égal à 1, sinon le bit décodé est égal à 0.

Le taux de vraisemblance logarithmique LLR d'un bit donné est défini par :

$$LLR(b_i) = \log \frac{\Pr\left(b_i = +1/\boldsymbol{y}, \boldsymbol{M}\right)}{\Pr\left(b_i = 0/\boldsymbol{y}, \boldsymbol{M}\right)} \tag{3.10}$$

$\boldsymbol{y}$ étant le vecteur reçu et $\boldsymbol{M}$ la matrice du canal équivalente (équation 1.22).
Le décodeur SB-Stack à sorties dures sera modifié afin de donner en sortie une liste de solutions au lieu d'une seule solution.

Définissons par $\mathcal{D}_{i,+1}$ la liste des points trouvés tels que le i^{me} bit vaut 1 et $\mathcal{D}_{i,0}$ la liste des points trouvés tels que le i^{me} bit vaut 0. Chaque point trouvé dans la liste est démodulé pour obtenir le vecteur de bits $\boldsymbol{b}$. En utilisant le théorème de Bayes et en assumant que les bits b_i sont équiprobables, l'expression précédente peut s'écrire sous la forme :

$$LLR(b_i) = \log \frac{\sum_{\boldsymbol{b} \in \mathcal{D}_{i,+1}} \Pr\left(\boldsymbol{y}/\boldsymbol{b}, \boldsymbol{M}\right)}{\sum_{\boldsymbol{b} \in \mathcal{D}_{i,0}} \Pr\left(\boldsymbol{y}/\boldsymbol{b}, \boldsymbol{M}\right)} \tag{3.11}$$

En considérant un bruit blanc de distribution Gaussienne, nous pouvons réécrire l'équation précédente en :

$$LLR(b_i) = \log \frac{\sum_{\boldsymbol{b} \in \mathcal{D}_{i,+1}} e^{-\frac{1}{\sigma^2}\|\boldsymbol{y} - \boldsymbol{M}\cdot\boldsymbol{s}(\boldsymbol{b})\|^2}}{\sum_{\boldsymbol{b} \in \mathcal{D}_{i,0}} e^{-\frac{1}{\sigma^2}\|\boldsymbol{y} - \boldsymbol{M}\cdot\boldsymbol{s}(\boldsymbol{b})\|^2}} \tag{3.12}$$

$\boldsymbol{s}(\boldsymbol{b})$ est le vecteur symbole d'information qui contient 1 ou 0 à la composante i. Étant donné que $\boldsymbol{w}$ est un bruit blanc Gaussien, nous pouvons appliquer l'approximation *max-log*[48] :

$$\begin{aligned} LLR(b_i) &\approx \max_{\boldsymbol{b} \in \mathcal{D}_{i,+1}} \left\{ -\frac{1}{\sigma^2} \|\boldsymbol{y} - \boldsymbol{M}\cdot\boldsymbol{s}(\boldsymbol{b})\|^2 \right\} - \max_{\boldsymbol{b} \in \mathcal{D}_{i,0}} \left\{ -\frac{1}{\sigma^2} \|\boldsymbol{y} - \boldsymbol{M}\cdot\boldsymbol{s}(\boldsymbol{b})\|^2 \right\} \\ &= \frac{1}{\sigma^2} \left[\min_{\boldsymbol{b} \in \mathcal{D}_{i,0}} \|\boldsymbol{y} - \boldsymbol{M}\cdot\boldsymbol{s}(\boldsymbol{b})\|^2 - \min_{\boldsymbol{b} \in \mathcal{D}_{i,+1}} \|\boldsymbol{y} - \boldsymbol{M}\cdot\boldsymbol{s}(\boldsymbol{b})\|^2 \right] \end{aligned} \tag{3.13}$$

En se basant sur la maximisation de (3.13), plusieurs algorithmes de décodage ont été proposés. Dans la suite, nous citons les algorithmes les plus utilisés.

3.5.2 Principaux algorithmes de décodage à sorties souples

Algorithme de SD à sorties souples (List Sphere Decoder - LSD)

A l'instar du SD à sorties dures, le principe de cet algorithme est de considérer les points du réseau à l'intérieur d'une sphère de rayon fixé C [49]. Toutefois, le décodeur à sorties dures retourne une seule solution qui correspond au point de la sphère le plus proche du vecteur reçu. Par contre, le décodeur LSD retourne une liste comprenant tous les points se trouvant à l'intérieur de la sphère. Cette sphère contient nécessairement la solution ML du décodeur Hard.

Toutefois, la taille de la liste dépend de la valeur du rayon choisi. Si le rayon est très petit, la liste est vide, si le rayon est grand la liste contient tous les points à la distance C. La taille de la liste n'est donc pas stable, ce qui peut affecter le calcul des LLR.
Dans [49], une solution a été introduite. Elle consiste à fixer le nombre de points retournés, noté N_p. Le rayon qui permet de générer ces N_p points est donné par la formule :

$$C = 2\sigma^2 \zeta N_p - \boldsymbol{y}^\dagger \left(I - \boldsymbol{H} \left(\boldsymbol{H}^\dagger \boldsymbol{H} \right)^{-1} \boldsymbol{H}^\dagger \right) \boldsymbol{y} \tag{3.14}$$

tel que $\zeta > 1$ est calculé par un intervalle de confiance de façon à avoir le point correct dans la sphère de rayon C. La taille de la liste est donc constante à chaque itération et elle contient les N_p points les plus proches parmi tous les points de la sphère. Pour chaque liste, l'algorithme calcule le LLR correspondant à chaque bit selon (3.13).

Algorithme modifié du SD à sorties souples (Shifted SD - SSD)

Cet algorithme commence par chercher la solution ML (celle-ci peut être donnée par le SD à sorties dures) et retourne tous les points du réseau autour du point ML. Pratiquement, ceci consiste à centrer la sphère autour du point ML au lieu du vecteur reçu.

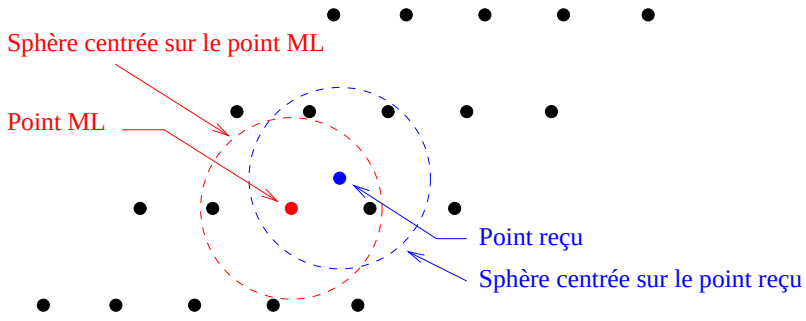


Figure 3.11 – Sphères centrées sur le point reçu et sur le point ML

En général, le point reçu est en dehors de la constellation. Le SD centré sur le point reçu parcourt plusieurs points n'appartenant pas à la constellation. En centrant la sphère sur le point ML, l'algorithme permet d'énumérer de meilleurs candidats et ainsi d'obtenir une meilleure estimation du LLR.

De la même manière que le LSD, un rayon est fixé afin d'avoir une taille stable de la liste des points trouvés. Ce rayon est donné par la formule [50] :

$$C \approx \left(\frac{N_p \times vol(\Lambda)}{V} \right)^{\frac{1}{n}} \tag{3.15}$$

avec $vol(\Lambda) = |\det(\boldsymbol{M})|$, $\boldsymbol{M}$ étant la matrice génératrice du réseau Λ et $V = \frac{\pi^n}{n}$.

Il est clair que le décodage à sorties souples nécessite une complexité nettement plus importante et un temps de décodage plus grand. Dans la suite, nous proposons un algorithme basé sur le SB-Stack offrant un décodage à sorties souples sans pour autant ajouter une grande complexité au décodage par rapport au décodeur à sorties dures.

3.5.3 Algorithme SB-Stack à sorties souples

Cet algorithme est une extension de l'algorithme SB-Stack proposé dans la première partie. Nous avons vu que le principe des décodeurs à sorties souples consiste à générer une liste de candidats pour pouvoir calculer les LLR.

Le décodeur SB-Stack à sorties dures permet grâce à l'utilisation d'une pile, de stocker tous les points parcourus dans l'arbre. En observant la pile, nous avons noté que l'algorithme a généré plusieurs candidats (un candidat est un chemin complet de longueur n dans l'arbre $\boldsymbol{s} = (s_n, \ldots, s_1)$) avant de retourner la solution ML correspondant à la plus faible métrique.
Il est alors judicieux d'exploiter tous ces points. A chaque fois qu'un candidat est trouvé, il est stocké dans une deuxième pile. A la fin de l'algorithme, le point en tête de la deuxième pile représente alors le point ML et les points suivants sont successivement les points les plus proches.

Nous obtenons ainsi une liste de candidats au lieu d'une seule solution sur lesquels nous pouvons calculer les LLR. Ainsi, l'algorithme à sorties dures peut facilement être modifié pour produire des sorties souples par le simple ajout d'une deuxième pile.

Nous proposons deux conditions d'arrêt pour l'algorithme SB-Stack :

- une condition sur la taille de la liste retournée : dans ce cas, nous fixons un nombre N_p de candidats souhaités. N_p est choisi au début de l'algorithme. A la sortie du décodeur SB-Stack à sorties dures, si la deuxième pile contient N_p points, l'algorithme prend fin. Sinon, l'algorithme SB-Stack est repris jusqu'à avoir N_p candidats.

- une condition sur le poids des candidats retournés : nous fixons une borne supérieure pour les poids des candidats retenus dans la liste : l'algorithme ne doit générer que les points ayant des poids inférieurs à cette borne. Dans ce cas, l'algorithme s'arrête lorsque le poids du noeud en tête de la première pile est supérieur à cette borne. Il faut noter cependant que cette deuxième méthode dépend de la contrainte choisie et ne retourne pas nécessairement le même nombre de points à chaque itération.

Dans l'objectif d'avoir la complexité la plus stable possible, nous allons utiliser la première méthode pour la construction de la liste des candidats.

3.5.4 Comparaison du SB-Stack à sorties souples avec les décodeurs à sorties souples existants

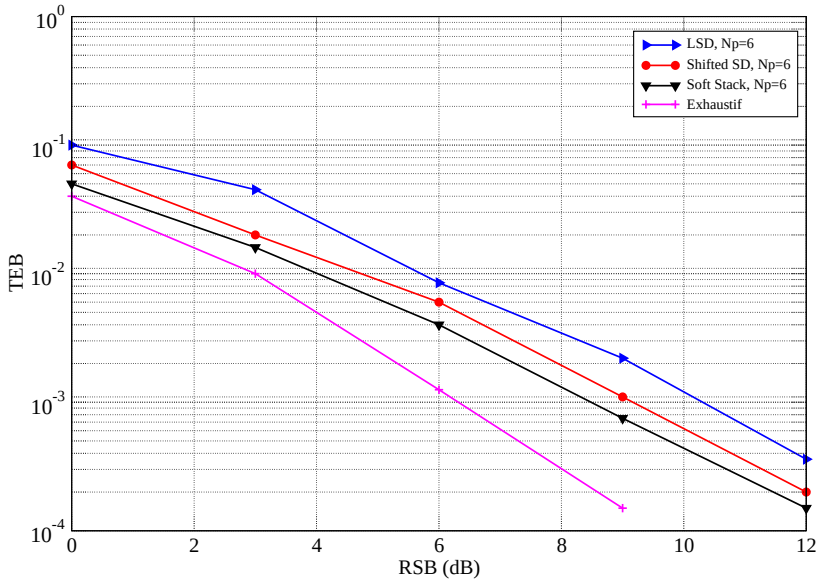
L'avantage du décodeur SB-Stack proposé est qu'il permet de générer la liste de points en continuant l'exécution de l'algorithme à sorties dures, ce qui permet de ne pas rajouter une grande complexité pour construire la liste. Par contre, les décodeurs LSD et SSD sont recommencés à chaque fois pour former cette liste.

En outre, nous pouvons noter que la liste retournée par l'algorithme SB-Stack contient les meilleurs candidats puisque l'algorithme retourne une liste ordonnée. Les N_p points générés sont successivement les points les plus proches du point ML, ce qui n'est pas le cas pour les décodeurs LSD et SSD. Ainsi, le SB-Stack permet d'obtenir les performances les plus optimales parmi ces décodeurs.

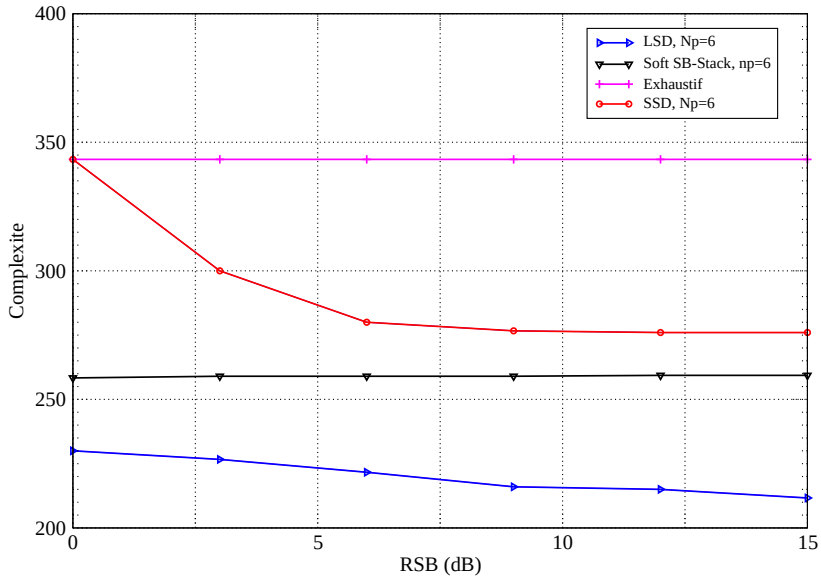
Pour illustrer ceci, nous considérons un système 2×2 utilisant un code convolutif de rendement $R = \frac{1}{2}$ de polynôme générateur (7,5) suivi d'un multiplexage spatial. Nous considérons une trame formée par 200 bits modulés par une modulation 4-QAM. A la réception, un décodage de Viterbi est appliqué à la sortie du décodeur MIMO considéré. Les différents décodeurs MIMO que nous comparons sont le LSD, le SSD et le SB-Stack souple.

Dans la figure 3.12.a, nous traçons le taux d'erreur par bit en utilisant les différents décodeurs à sorties souples cités plus haut. Nous considérons pour ceci que ces derniers génèrent une liste de $N_p = 6$ candidats. Nous pouvons noter que le décodeur SB-Stack permet d'avoir de meilleures performances que les décodeurs SSD et LSD et un gain moyen de 1dB et 2dB respectivement. Les performances que nous obtenons sont sous-optimales, pour avoir les performances du décodeur exhaustif, il suffit de considérer une liste plus grande.

Dans la figure 3.12.b, nous traçons la complexité des différents algorithmes en nombre d'opérations de multiplication. Nous observons que le décodeur SB-Stack permet d'avoir un gain en complexité par rapport au décodeur SSD. Toutefois, le LSD présente la complexité la plus faible mais les performances les moins bonnes. Cependant, il est à noter que le décodeur proposé présente une complexité quasi-constante par rapport aux autres décodeurs ce qui est intéressant d'un point de vue pratique afin d'avoir un délai de décodage stable.



(a)



(b)

Figure 3.12 – Comparaison des performances du SB-Stack et des principaux décodeurs à sorties souples existants

3.6 Conclusion

Dans ce chapitre, nous avons proposé d'appliquer le décodage séquentiel utilisant l'algorithme Stack pour le décodage des réseaux de points. Nous avons commencé par donner une première approche en se basant sur le calcul du point Babai. Nous avons vu que celle-ci présente des complexités d'autant plus importantes que les performances sont meilleures. Nous avons alors proposé un deuxième algorithme que nous avons appelé SB-Stack. Ce dernier permet de combiner la région de recherche du SD avec la stratégie de recherche du décodeur Stack. Nous avons montré que le SB-Stack permet d'avoir une complexité réduite tout en ayant les performances ML.

Par la suite, nous avons proposé une deuxième version du décodeur SB-Stack. Ce décodeur permet grâce à la stratégie de recherche du Stack de générer des sorties souples tout en gardant une complexité raisonnable. De plus, nous avons montré que celui-ci permet d'atteindre les meilleures performances par rapport aux différents décodeurs à sorties souples existants.

Chapitre 4

Décodage adaptatif pour les systèmes MIMO

4.1 Introduction

Comme présenté dans les chapitres précédents, plusieurs décodeurs ont été proposés pour le décodage des systèmes MIMO. Ces décodeurs peuvent être classés en deux groupes : les décodeurs optimaux tels que le décodeur par sphères, la stratégie de Schnorr-Euchner, le décodeur Stack et le SB-Stack et les décodeurs sous-optimaux tels que le ZF, MMSE, DFE.

La complexité des algorithmes de décodage représente un critère majeur quant à leur implémentation pratique. Un deuxième critère est aussi important en pratique, à savoir la complexité variable en fonction du RSB et de la réalisation du canal. Nous nous proposons dans ce chapitre de traiter ce critère. Pour cela, nous proposons un décodeur adaptatif.

Dans la littérature, il existe plusieurs travaux utilisant l'idée de l'adaptation. Nous commençons alors par présenter différents schémas d'adaptation existants. Puis, nous définissons le décodeur adaptatif que nous proposons ainsi que les critères de sélection considérés. Une implémentation pratique de ce décodeur sera donnée par la suite et ses performances étudiées.

4.2 Principaux schémas d'adaptation existants

Les systèmes MIMO permettent théoriquement d'accroitre la capacité et le gain de diversité par rapport aux systèmes SISO [51], [52]. Toutefois, les performances des systèmes MIMO dépendent des conditions du canal de transmission. Pour maintenir une qualité de service (QoS), les paramètres de transmission (tels que la puissance à l'émission, le débit binaire, la modulation, etc) sont ajustés au cas du canal le plus défavorable. Cependant, ceci conduit à une sous-utilisation des ressources du canal dans les cas de transmissions favorables.

Pour palier à ce problème, des techniques d'adaptation ont été proposées dans la littérature. Les plus utilisées sont le contrôle de puissance, la modulation et le codage adaptatifs ainsi que l'adaptation par sélection d'antennes.

4.2.1 Contrôle de puissance

C'est l'une des techniques majeures introduites dans le WCDMA [53]. L'idée est d'augmenter la puissance de transmission quand la qualité du signal reçu est faible et de diminuer la puissance de transmission quand la qualité en réception atteint un seuil donné. Ceci permet une communication fiable entre l'émetteur et le récepteur. Ainsi la technique de contrôle de puissance réduit les interférences causées par une puissance trop importante et la capacité du système est ainsi augmentée.

4.2.2 Modulation et codage adaptatifs (AMC)

Au lieu de garder une qualité de signal constante au niveau du récepteur, on peut changer les paramètres du système (le type de modulation et le taux de codage) de telle façon que le plus d'informations soit transmis lorsque l'état du canal est bon et le moins possible lorsque le canal est dégradé. Cette technique est appelée la modulation et codage adaptatifs. Elle est utilisée notamment en WIMAX et en UMTS.

Par exemple, dans le cas des nouveaux terminaux mobiles supportant la technologie HSDPA [54], les modulations disponibles sont la QPSK et la 16-QAM. La modulation 16-QAM présente 4 bits/symbole au lieu de 2 bits/symbole pour la modulation QPSK. Le débit est alors augmenté de façon significative. Toutefois, il faut noter que l'utilisation d'une modulation de grande taille est accompagnée d'une plus grande complexité dans les terminaux, qui doivent estimer l'amplitude des symboles reçus. Cette estimation d'amplitude est nécessaire pour séparer tous les points de la constellation. Étant donné que cette estimation devient plus difficile lorsque la qualité du signal reçu est mauvaise, il est alors plus judicieux d'utiliser la modulation QPSK dont la constellation est moins dense.

La décision d'une transmission en 16-QAM ou en QPSK est faite dans le réseau en utilisant un indicateur de qualité de canal (CQI : Channel Quality Indicator) mesuré par le mobile et transmis à la station de base. La sélection de l'une ou l'autre des modulations se fait en fonction de la valeur du CQI :

- Si le canal est marqué comme bon, on utilisera la modulation 16-QAM qui offre un meilleur débit mais plus de tolérance aux erreurs.

- Si le canal est dégradé, on utilisera la modulation QPSK, plus robuste contre les erreurs.

De plus, le terminal peut se voir attribuer un taux de codage de 1/4 à 3/4 selon le code utilisé. En combinant les différents types de modulation, le taux de codage, plusieurs combinaisons, appelées également schémas de modulation et de codage (Modulation and Coding Scheme - MCS) sont possibles.
Par exemple, pour un utilisateur proche de la station de base, les effets d'évanouissements dus au canal de transmission sont minimes. La station lui alloue alors la meilleure combinaison, à savoir, une modulation 16-QAM avec un taux de codage de 3/4. Dans ce cas, il bénéficie du débit maximal de 10.7 Mbits/s. Cette combinaison peut varier en fonction du signal reçu. Par conséquent, la station de base a la responsabilité de sélectionner l'algorithme de modulation et de codage approprié.

Le tableau (4.1) illustre différents MCS proposés par le 3GPP dans la Release 5 (UMTS) pour le lien descendant (station de base vers terminal mobile) utilisant la technologie HSDPA.

MCS	Modulation	Taux de codage	Débit maximal		
			5 codes	10 codes	15 codes
1	QPSK	1/4	600 kbps	1.2 Mbps	1.8 Mbps
2		2/4	1.2 Mbps	2.4 Mbps	3.6 Mbps
3		3/4	1.8 Mbps	3.6 Mbps	5.4 Mbps
4	16−QAM	2/4	2.4 Mbps	4.8 Mbps	7.2 Mbps
5		3/4	3.6 Mbps	7.2 Mbps	10.7 Mbps

Table 4.1 – Schémas de modulation et de codage MCS pour le lien descendant dans la technologie HSDPA

4.2.3 Sélection d'antennes

L'utilisation de plusieurs antennes à l'émission et à la réception implique une plus grande complexité au niveau du récepteur afin de séparer les différents flux issus des diverses antennes.
La sélection d'antennes est une technique qui permet de réduire la complexité et le coût des systèmes MIMO tout en gardant le gain de diversité apporté par l'utilisation de toutes les antennes [55].

Le principe de cette technique est de sélectionner un ensemble d'antennes émettrices et réceptrices parmi toutes les antennes disponibles. Seules ces antennes sont activées, les autres restent inactives. Pour un système à M antennes en émission et N antennes en réception, notons respectivement par m et n le nombre d'antennes émettrices et réceptrices actives et p le nombre total d'antennes actives. Dans ce cas, la canal associé est noté par $\boldsymbol{H_p = H\,(1:n, 1:m)}$.

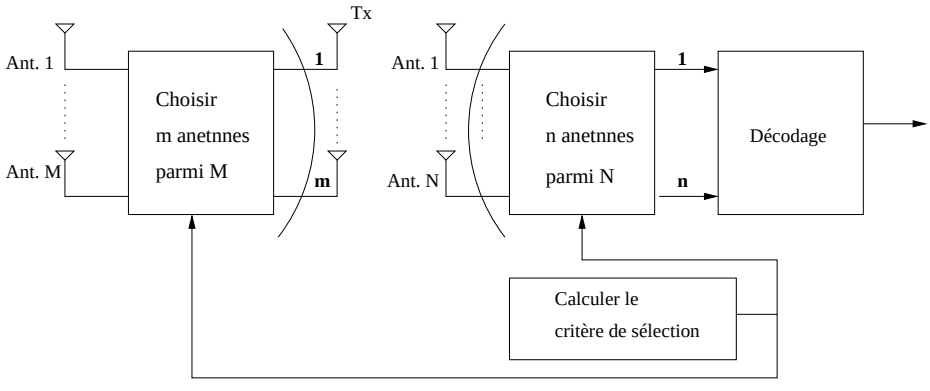


Figure 4.1 – Technique de sélection d'antennes dans un système MIMO

La sélection peut se faire selon différents critères en fonction de la qualité du canal de transmission.

Critère de minimisation de la probabilité d'erreur

Pour chaque combinaison de p antennes, calculer la probabilité d'erreur associée et choisir la meilleure combinaison d'antennes qui minimise cette probabilité.

Critère de maximisation du RSB

Dans [56], l'adaptation est appliquée en sélectionnant l'ensemble d'antennes de transmission qui maximise le rapport signal à bruit. Ce schéma d'adaptation est proposé dans le cas d'utilisation d'un décodeur linéaire (MMSE ou ZF). Le nombre d'antennes en réception est constant.

Notons par $\boldsymbol{G}$ la matrice d'égalisation considérée. Elle est égale à la matrice du filtre MMSE ou ZF. Le RSB est calculé selon l'expression :

$$RSB_k = \frac{E_s \left| g_k^* h_k \right|^2}{mN_0 \left\| g_k \right\|^2 + E_s \sum\limits_{j \neq k} \left| g_k^* h_j \right|^2}$$

avec g_k est la k^{me} ligne de $\boldsymbol{G}$ et h_k est la k^{me} colonne de $\boldsymbol{H_p}$, E_s représente l'énergie totale transmise et N_0 est la variance du bruit.

Pour tout ensemble d'antennes de transmission, l'algorithme d'adaptation calcule la matrice $\boldsymbol{G}$ et le RSB correspondant et choisit l'ensemble qui maximise le RSB minimal.

Critère de la capacité

Ce critère tient compte de la capacité instantanée du canal. Pour tout ensemble de p antennes, calculer la capacité instantanée donnée par [52] :

$$C_{inst}^p = \log \det \left(\boldsymbol{I_m} + \frac{E_s}{mN_0} \boldsymbol{H_p^\dagger H_p} \right)$$

Dans ce cas, le meilleur ensemble est celui qui donne la capacité la plus grande [56].

Critère de minimisation de la distance euclidienne

Ce critère a été présenté dans [57]. Les auteurs ont considéré que la probabilité d'erreur du système MIMO utilisant une modulation q-QAM est bornée par :

$$P_e \leq \left(2^{nb} - 1 \right) Q \left(\frac{d_{min}^c}{2\sqrt{N_0}} \right)$$

où $b = \log_2 (q)$ est le nombre de bits par symbole. Dans ce cas, le critère vise à minimiser cette borne supérieure de la probabilité. Autrement, ce critère consiste à choisir l'ensemble d'antennes maximisant la distance euclidienne de la constellation, notée par d_{min}^c. Ceci implique de calculer $2^{n \cdot b - 1} \left(2^{n \cdot b} - 1 \right)$ distances. Les résultats de simulations montrent que ce critère permet d'avoir de meilleures performances que le critère précédent [57], toutefois, il présente une complexité très élevée. Par exemple, pour un système MIMO à 2 antennes à l'émission utilisant une constellation 16-QAM, 65280 distances doivent être calculées.

4.2.4 Décodage adaptatif

Dans la plupart des schémas existants, l'adaptation est appliquée au niveau de l'émetteur afin d'augmenter l'efficacité spectrale et de garantir une certaine qualité de service (QoS). Dans tous ces schémas, le récepteur reste inchangé. Toutefois, le choix du décodeur adéquat permet également d'optimiser les performances globales du système.

Récemment, deux récepteurs adaptatifs ont été proposés. Tous deux sont définis dans le cas d'une transmission multi-utilisateurs. Le premier est basé sur le calcul du rapport signal plus interférences sur bruit ($SINR$) et le deuxième est basé sur le calcul du nombre de conditionnement de la matrice du canal choisie.

Décodage adaptatif selon le SINR

Cet algorithme est présenté dans [58], [59] dans le contexte d'une transmission DS-CDMA pour le lien descendant. Il permet de basculer entre le décodeur RAKE et le décodeur LMMSE (Linear MMSE).
L'algorithme permet de calculer le $SINR$ correspondant à l'utilisation de chaque décodeur selon la formule donnée en [58]. Les décodeurs dont le $SINR$ est inférieur à une valeur seuil $SINR_{target}$ sont écartés. Le décodeur sélectionné est le décodeur le moins complexe parmi ceux retenus.

Ce décodeur est très intéressant puisqu'il permet à la fois de répondre à la QoS souhaitée avec la plus faible complexité possible offrant un compromis performance-complexité.

Décodage adaptatif selon le nombre de conditionnement

Afin de mesurer la qualité du canal de transmission, plusieurs critères ont été proposés. Dans [58], cette mesure est donnée par l'estimation du rapport $SINR$. Récemment dans [60], les auteurs ont introduit une nouvelle mesure qui consiste au nombre de conditionnement. Pour une matrice du canal $\boldsymbol{H}$, le nombre de conditionnement est égal au rapport de la plus grande valeur propre sur la plus petite valeur propre de la matrice :

$$\delta\left(\boldsymbol{H}\right) = \frac{\lambda_{max}}{\lambda_{min}},$$

Si δ est proche de 1, on dit que la matrice de canal est bien-conditionnée. Par contre, si $\delta >> 1$, on dit que la matrice de canal est mal conditionnée ce qui correspond à un canal très perturbé.

Le nombre de conditionnement peut alors indiquer sur la fiabilité de transmission sur un canal donné. Dans [60], les auteurs ont démontré que la minimisation du nombre de conditionnement est équivalente au critère de minimisation de l'erreur quadratique moyenne. Toutefois, il est difficile de statuer sur la qualité du canal pour des valeurs intermédiaires de δ.

Dans [61], un algorithme de décodage est basé sur la mesure du canal de transmission. Si le canal est perturbé, on utilise un décodeur puissant pour avoir les performances souhaitées. si le canal est bon, un décodeur peu robuste est alors utilisé. Le but est de garder une QoS constante.

4.3 Décodage adaptatif proposé

4.3.1 Principe

Nous proposons ici d'appliquer le même principe que les décodeurs adaptatifs. Disposant de plusieurs décodeurs dans le terminal récepteur, notre but est de sélectionner le décodeur offrant la QoS souhaitée avec le minimum de complexité.

Pour les systèmes pratiques, il est aussi important d'avoir une complexité et un temps de décodage constants en particulier pour les applications temps réel telle que la TV numérique. Pour des cas de mauvaises transmissions, le décodage nécessite cependant un délai de décodage long. L'idée ici est de détecter ces cas défavorables et appliquer le décodeur approprié permettant de réduire ce délai.

Contrairement aux décodeurs adaptatifs que nous avons cités précédemment où le but est de garder une QoS constante, notre objectif est d'avoir une complexité constante.
Pour appliquer le décodage adaptatif, nous avons d'abord besoin de définir les critères de sélection qui permettent de choisir entre l'un ou l'autre des décodeurs disponibles. Dans la suite, nous proposons trois critères de sélection.

4.3.2 Critères de sélection

Afin de sélectionner le décodeur le plus adéquat, nous devons à la fois tenir compte de la qualité du canal de transmission et des performances globales souhaitées. Dans la suite, le premier critère que nous proposons permet de répondre à la première exigence et le second permet de garantir la QoS demandée. Afin d'améliorer la sélection, nous proposons de combiner ensuite ces deux critères.

Sélection selon la qualité du canal de transmission

En considérant le décodage ML tels que le SD et le décodeur SB-Stack, nous avons remarqué que dans certains cas, la phase de recherche nécessite un temps très long pour converger sans pour autant donner la bonne solution. Ces cas correspondent à de mauvaises réalisations du canal de transmission. Comme première approche, on peut arrêter le décodage ML après un certain délai seuil fixé et le compléter par une simple détection ZF ou ZF-DFE. Ceci engendre cependant de mauvaises performances.

Afin d'éviter ce problème, nous proposons dans ces cas d'utiliser le décodage sous-optimal afin d'éviter un temps de décodage long et par la suite une complexité importante. Pour détecter ces cas défavorables de réalisations du canal, une mesure fiable de la qualité de transmission peut être donnée par la capacité instantanée du canal. Celle-ci correspond à la quantité d'information qui peut être transmise sur le canal $\boldsymbol{H}$ sans erreurs [62]. Elle est définie par :

$$C(\boldsymbol{H}) = \log_2 \det \left(\boldsymbol{I}_n + \frac{RSB}{n} \boldsymbol{H}\boldsymbol{H}^t \right)$$

Dans la figure 4.2, nous avons tracé la capacité instantanée du canal pour différents rapports signal sur bruit.

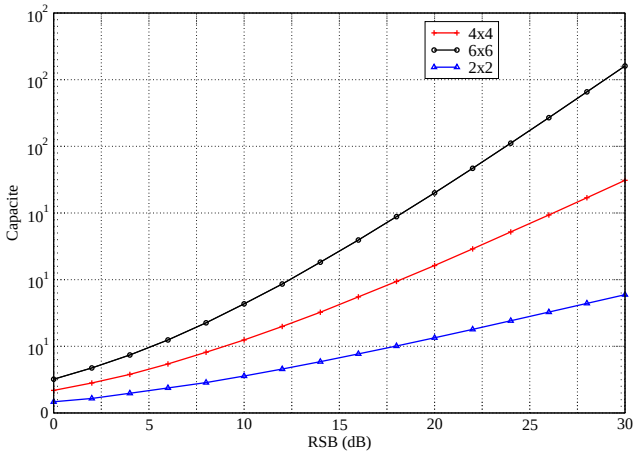


Figure 4.2 – Capacité d'un système MIMO utilisant une constellation 4-QAM

Nous appelons probabilité de coupure du canal P_{out} la probabilité donnée par :

$$P_{out}(R) = \Pr\{C(\boldsymbol{H}) < R\}$$

tel que R représente le rendement du système. Nous distinguons deux cas :

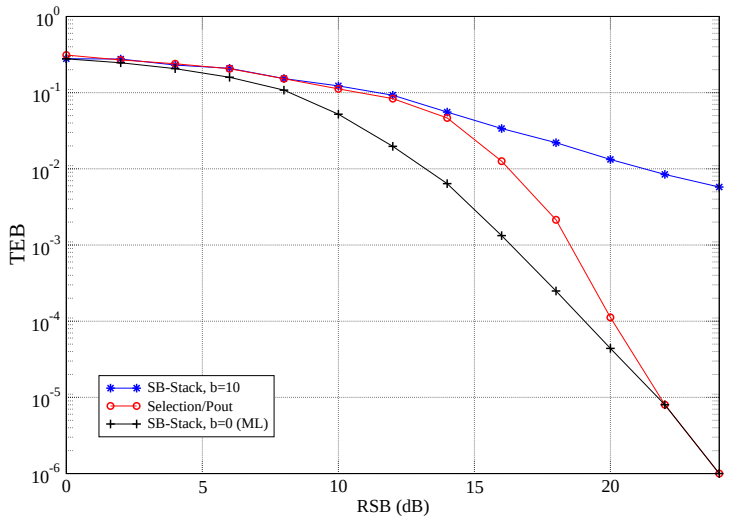
- Si $C(\boldsymbol{H}) < R$: le canal est dans la région de coupure. Dans ce cas, le canal est qualifié de mauvais, il est donc impossible de décoder le vecteur transmis sans erreurs même en utilisant un décodeur ML. Pour avoir une complexité faible, il est plus judicieux par conséquent d'utiliser un décodeur sous-optimal ayant une faible complexité.
- Si $C(\boldsymbol{H}) \geq R$: le canal peut transmettre le signal sans erreurs. Afin d'augmenter les performances du système, nous utilisons un décodeur plus robuste, dans ce cas le décodeur ML.

Par la suite, à chaque transmission, le décodeur adaptatif calcule la capacité instantanée $C(\boldsymbol{H})$ et sélectionne le décodeur adéquat selon que l'on est ou pas dans la région de coupure.

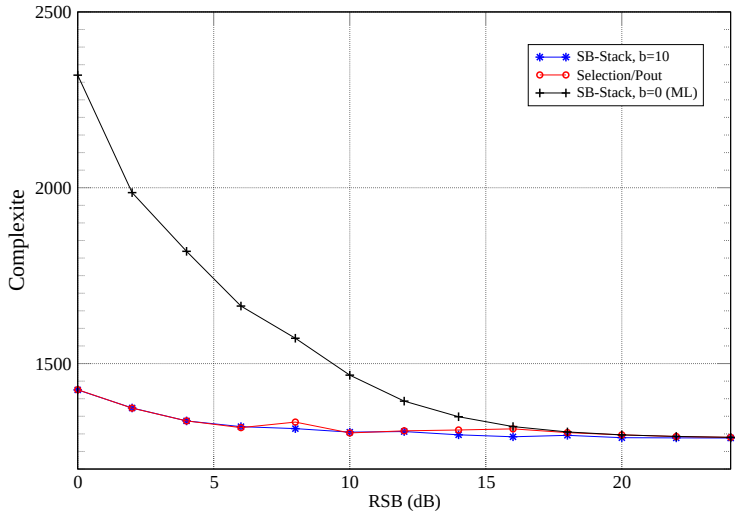
Dans la figure 4.3.a, nous avons représenté respectivement les performances du décodeur adaptatif (noté Selection/Pout) appliqué au système MIMO 4×4 utilisant une constellation 16-QAM. Dans ce cas, le décodeur adaptatif bascule entre le décodeur SB-Stack (qui représente dans ce cas le décodeur ML) et le décodeur sous-optimal ZF-DFE (équivalent au SB-Stack avec une grande valeur du biais, exemple b=10). Nous observons que le décodeur adaptatif permet d'avoir des performances qui sont à 3dB au maximum de ML. Toutefois, à partir d'un RSB égal à 20dB, les performances obtenues sont très proches du ML et convergent vers ML. En effet, à partir de cette valeur, le canal est rarement dans la région de coupure, le décodeur ML est donc quasiment utilisé tout le temps.

Cependant, nous observons à partir de la courbe 4.3.b, que la complexité obtenue par le décodeur adaptatif est largement réduite par rapport au ML, en particulier pour les RSB faibles ($<$10 dB) alors que les pertes en taux d'erreur par bit ne dépassent pas 2dB. De plus, cette complexité est quasi-constante. Cela signifie que le décodage adaptatif proposé présente une

caractéristique très intéressante de point de vue pratique ; il permet d'avoir quasiment le même délai de décodage quelle que soit la valeur du RSB considérée.



(a)



(b)

Figure 4.3 – Performance et complexité du décodeur adaptatif basé sur le critère de la qualité du canal de transmission pour un système MIMO 4×4 utilisant une 16-QAM

Ceci est vérifié par la figure 4.4, où nous avons tracé le temps de décodage pour le décodeur ML et celui du décodeur adaptatif en fonction des réalisations du canal pour un RSB égal à 12dB. Nous obtenons un délai de décodage quasi-constant lorsqu'on applique l'adaptation alors

qu'il est très variable dans le cas du ML. En effet, ce qui explique ce délai constant est que l'adaptation permet d'éliminer tous les cas de transmissions défavorables où $C(\boldsymbol{H}) < R$. Toutefois, les quelques pics qui persistent sont dus aux cas limites où la valeur de $C(\boldsymbol{H})$ est égale ou proche de R.

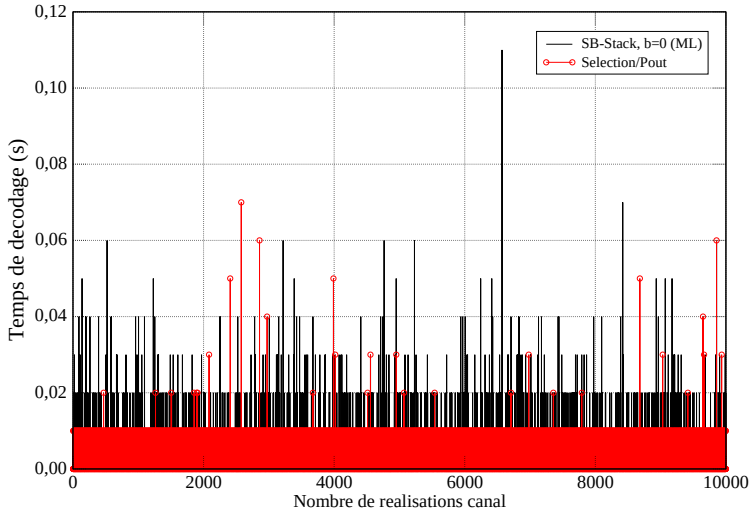


Figure 4.4 – Temps de décodage pour différentes réalisations du canal (10000 itérations) pour un système MIMO 4×4 utilisant une 16-QAM à RSB=12dB

Sélection selon les spécifications système

Afin de garantir une bonne QoS comme dans les schémas adaptatifs existants, nous proposons de sélectionner le décodeur qui permet de répondre aux spécifications système souhaitées.

Nous établissons pour cela une table de probabilité d'erreurs seuils pour chaque plage de RSB. Dans le tableau ci-dessous, nous illustrons un exemple de spécifications système qui correspondent à ces probabilités d'erreurs souhaitées. Le décodeur sélectionné est alors celui qui permet d'atteindre ces performances parmi tous les décodeurs disponibles. Si plusieurs décodeurs permettent d'atteindre le taux d'erreur demandé, le décodeur sélectionné est celui qui offre la plus faible complexité.

Le but de ce critère est de garantir une qualité de service seuil. Toutefois, ce critère d'adaptation ne permet pas de sélectionner de manière instantanée le décodeur adéquat mais, globalement dans une plage donnée de RSB, le décodeur le plus convenable.

$RSB(dB)$	$< 5dB$	$< 10dB$	$< 15dB$	$< 20dB$
P_e	$< 10^{0}$	$< 10^{-1}$	$< 10^{-2}$	$< 10^{-3}$

Table 4.2 – Exemple de spécifications système

Sélection combinée

Le premier schéma d'adaptation proposé tient compte de la qualité du canal et permet de sélectionner le décodeur approprié à chaque réalisation du canal. Le deuxième schéma permet de garantir une QoS souhaitée sans pour autant se soucier de l'état courant du canal.

Pour profiter des avantages offerts par les deux critères de sélection, nous proposons un troisième critère permettant de les combiner. Dans ce cas, nous établissons la table des performances souhaitées telles qu'illustrées dans le tableau 4.2. En utilisant le deuxième critère, nous sélectionnons le décodeur qui permet d'atteindre la QoS requise dans chaque plage de RSB. Par la suite, nous appliquons le premier critère. Nous calculons la capacité instantanée $C(\boldsymbol{H})$ dans chacune de ces plages. Si le canal est en coupure, nous appliquons le décodeur déjà sélectionné par le deuxième critère, sinon nous appliquons le décodeur ML.

Il est évident de voir qu'en combinant les deux critères, nous obtenons de meilleures performances. En effet, contrairement au premier critère où le taux d'erreur n'est pas prédictible, ce schéma permet de répondre aux spécifications système en garantissant un taux d'erreur seuil. Et par rapport au deuxième critère, nous obtenons un gain en performance puisque le décodeur ML est plus souvent utilisé dans ce cas.

Dans ce qui suit, nous proposons un exemple pratique d'implémentation du décodeur adaptatif.

4.4 Exemple d'implémentation pratique du décodage adaptatif

L'utilisation du décodage adaptatif sous-entend l'utilisation d'au moins deux décodeurs à la réception. Toutefois, l'implémentation de plusieurs décodeurs ajoute une complexité significative au récepteur, ce qui va à l'encontre des exigences actuelles où les systèmes doivent être de plus en plus simples et de moins en moins encombrants.

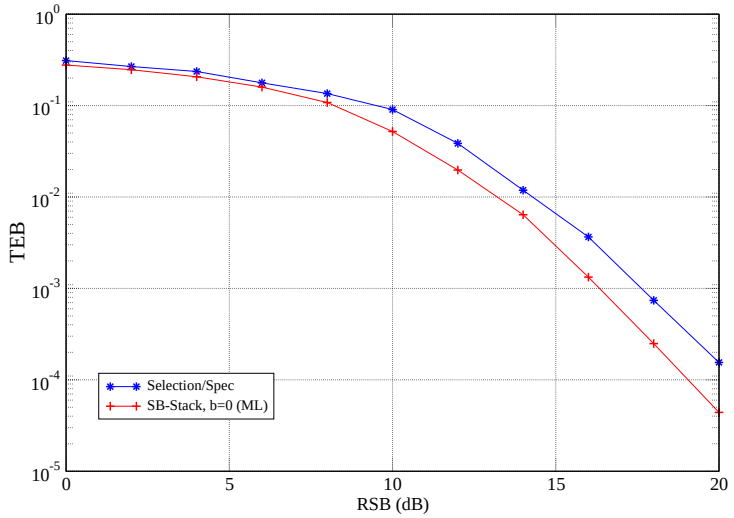
Nous avons vu dans le paragraphe 3.3.5 du chapitre 3, que le décodeur SB-Stack permet d'avoir différentes performances et complexités selon la valeur du biais utilisé. Nous rappelons que ces performances vont du ML au ZF-DFE avec des complexités décroissantes. Ainsi, une modification de la valeur du biais est équivalente à basculer d'un décodeur à un autre.
Nous proposons dans ce cas, d'utiliser le SB-Stack pour réaliser un décodage adaptatif. L'adaptation se réduit ainsi à un simple ajustement du biais, l'utilisation du SB-Stack nous permet alors de gagner en complexité d'implémentation.

Par exemple, nous nous proposons d'appliquer le deuxième critère de sélection en utilisant le décodage par SB-Stack. Dans ce cas, il suffit de choisir le biais qui répond à la spécification dans le tableau 4.2. Les différents biais nécessaires sont représentés dans le tableau 4.3.

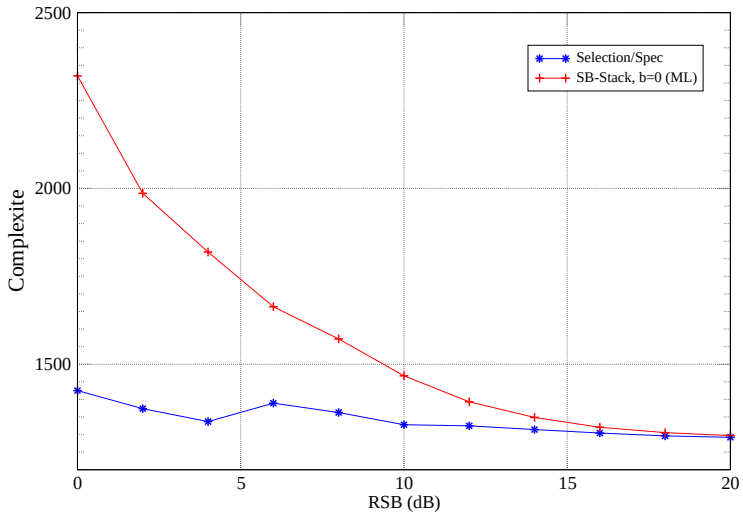
$RSB(dB)$	$< 5dB$	$< 10dB$	$< 15dB$	$< 20dB$
P_e	$< 10^{0}$	$< 10^{-1}$	$< 10^{-2}$	$< 10^{-3}$
$Biais$	10	1	0.5	0.15

Table 4.3 – Exemple de spécifications système dans le cas d'un décodeur adaptatif utilisant le SB-Stack

Dans la figure 4.5, nous représentons les performances correspondantes. Nous observons que la courbe obtenue est une courbe en morceaux. En effet, ceci s'explique par le fait que la sélection est appliquée par intervalles de RSB et non pas de façon instantanée. La complexité obtenue est quasi-constante, cependant, il est possible d'avoir des pics de complexité pour certaines réalisations du canal.



(a)

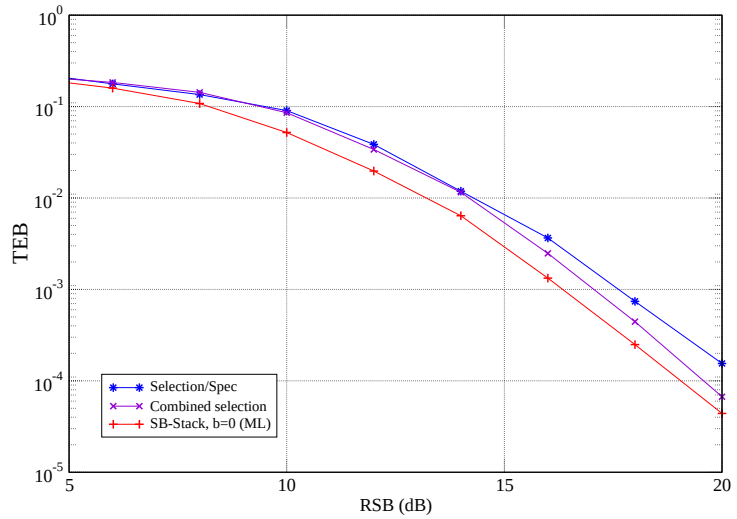


(b)

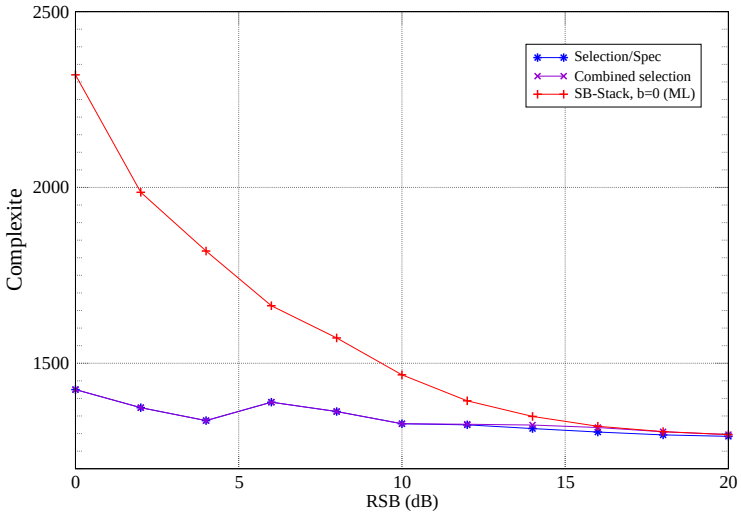
Figure 4.5 – Performance et complexité du décodeur adaptatif basé sur le critère de spécifications pour un système MIMO 4×4 utilisant une 16-QAM

En appliquant le troisième critère de sélection (noté Combined selection) sur le même système, nous obtenons des performances améliorées par rapport au cas précédent en particulier dans le cas des forts RSB (figure 4.6). Toutefois, la complexité pour les trois critères de sélection reste pratiquement inchangée et elle est peu variable en fonction du RSB. En effet, pour les faibles RSB, l'adaptation permet de détecter toutes les mauvaises réalisations du canal et de remplacer dans ce cas le décodeur ML par un décodeur sous-optimal ayant une plus faible complexité. Pour les grands RSB, où les conditions de transmission sont plus souvent favorables, la complexité des décodeurs optimaux et sous-optimaux se rejoignent et sont pratiquement égales, telles qu'illustrées dans la figure 4.6. Par la suite, l'utilisation de décodeurs plus robustes (ML ou proches de ML), n'ajoute pas de complexité supplémentaire au système.

Le schéma de décodage adaptatif proposé peut être d'un grand intérêt dans les applications pratiques, étant donné qu'il offre une complexité quasi-constante. De plus, l'utilisation du SB-Stack paramétré permet de n'implémenter qu'un seul décodeur et par la suite de ne pas augmenter la complexité lors de la mise en place du récepteur.



(a)



(b)

Figure 4.6 – Performance et complexité du décodeur adaptatif basé sur la sélection combinée pour un système MIMO 4×4 utilisant une 16-QAM

4.5 Conclusion

Dans ce chapitre, nous avons introduit la notion d'adaptation dans le décodage des systèmes MIMO. Le but est de choisir, pour chaque transmission, le décodeur le plus approprié. Nous avons donc défini trois critères d'adaptation.

Le premier critère est basé sur la mesure de la qualité du canal et permet d'avoir une complexité constante en fonction du RSB et la réalisation du canal. Le deuxième critère permet de sélectionner pour chaque RSB donné le décodeur permettant d'atteindre la QoS souhaitée. Finalement, le troisième critère est une combinaison des deux critères précédents et permet de regrouper ces deux avantages.

Pour appliquer cette méthode d'adaptation, il est nécessaire de disposer de plusieurs décodeurs implémentés au niveau du récepteur. Une solution pratique est de remplacer l'implémentation de différents décodeurs par le seul décodeur SB-Stack paramétré proposé dans le troisième chapitre puisque celui-ci permet d'avoir des performances et des complexités variables allant du ML au ZF-DFE par simple ajustement de la valeur du biais.

Chapitre 5

Chaine complète de décodage MIMO

5.1 Introduction

Dans les chapitres précédents, nous avons étudié les décodeurs pour les systèmes MIMO. L'utilisation d'un décodage sous-optimal donne en général des performances faibles et le décodage optimal présente une complexité de décodage élevée qui est d'autant plus complexe que la dimension du réseau augmente. Le but est d'avoir un décodeur offrant des performances acceptables avec des complexités faibles. Dans les chapitres précédents, nous avons présenté des schémas de décodage répondant à cette question. Toutefois, l'étude s'est restreinte à la phase de recherche du signal émis. Nous allons à présent nous intéresser à la phase de pré-traitement afin d'avoir une chaine de décodage complète.

Il existe deux techniques de pré-traitement : le pré-traitement à gauche, MMSE-GDFE, qui permet d'avoir une matrice du canal mieux conditionnée et le pré-traitement à droite, qui consiste en une réduction et permet d'avoir la matrice du canal la plus orthogonale possible.

Ainsi, le pré-traitement suivi d'un décodage sous-optimal permettra d'avoir de meilleures performances. Dans le cas d'un décodage optimal, le pré-traitement permettra d'accélérer le décodage. Nous présentons alors une chaine complète de décodage utilisant les techniques de pré-traitement associées à différents décodeurs pour un schéma de transmission MIMO.

Dans la première partie de ce chapitre, nous définissons le MMSE-GDFE et nous montrons l'utilité de son application. Nous donnons une méthode simplifiée de calcul du MMSE-GDFE dans le cas du Golden code. Dans la deuxième partie, nous présentons les différents algorithmes de réduction qui existent dans la littérature, parmi lesquels, nous distinguons la réduction LLL et la réduction Seysen. Une étude comparative de ces deux techniques est ensuite présentée. La dernière partie sera dédiée aux résultats de simulation de la chaine de transmission MIMO avec un schéma de décodage complet comportant le pré-traitement et le décodage.

5.2 Pré-traitement à gauche : MMSE-GDFE

5.2.1 Définition

Reprenons le système MIMO défini par :

$$\boldsymbol{y} = \boldsymbol{H} \cdot \boldsymbol{s} + \boldsymbol{w} \tag{5.1}$$

avec $\boldsymbol{H} \in \mathbb{C}^{n \times m}$ est la matrice du canal. Une transformation linéaire du système précédent définie par la matrice $\boldsymbol{H}^{\dagger}$ donne en sortie :

$$\boldsymbol{y_{mf}} = \boldsymbol{H}^{\dagger}\boldsymbol{H} \cdot \boldsymbol{s} + \boldsymbol{w_{mf}} \tag{5.2}$$

Le décodeur de réseaux de points peut être un décodeur sous-optimal tels que le ZF et le MMSE, ou optimal tels que le SD et le décodeur SB-Stack.
Le MMSE-GDFE (Minimum Mean Square Error-Generalized Decision Feedback Equalizer) est un décodeur à retour de décision qui permet de retrouver les symboles d'information sous la forme[63] :

$$\boldsymbol{z} = \boldsymbol{F_{mf}}\boldsymbol{y_{mf}} - (\boldsymbol{B_{mf}} - \boldsymbol{I})\hat{\boldsymbol{s}} \tag{5.3}$$

où $\boldsymbol{F_{mf}}$ et $\boldsymbol{B_{mf}} \in \mathbb{C}^{m \times m}$, et $\boldsymbol{B_{mf}}$ est triangulaire supérieure à éléments diagonaux tous égaux à l'unité. $\boldsymbol{F_{mf}}$ et $\boldsymbol{B_{mf}}$ sont respectivement les filtres direct et de retour, dits également *Forward* et *Backward*, $\boldsymbol{z}$ est la sortie du décodeur MMSE-GDFE et le vecteur $\hat{\boldsymbol{s}}$ représente l'estimation du vecteur transmis. Grâce à la structure triangulaire de la matrice $\boldsymbol{B_{mf}} - \boldsymbol{I}$, les symboles d'information sont récursivement calculés en partant de la m^{me} composante.

Contrairement aux autres cas où le décodeur MMSE-GDFE est utilisé, nous l'utilisons uniquement comme étape de pré-traitement du système initial (5.1). L'idée est donc d'obtenir un système équivalent qui dépend des filtres $\boldsymbol{F_{mf}}$ et $\boldsymbol{B_{mf}}$, un décodeur de réseaux de points sera par la suite utilisé pour calculer $\hat{\boldsymbol{s}}$.

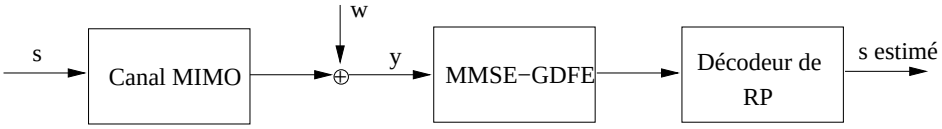


Figure 5.1 – Pré-traitement à gauche suivi d'un décodeur de réseaux de points

A partir de (5.3), $\boldsymbol{F_{mf}}$ et $\boldsymbol{B_{mf}}$ sont calculés de façon à réduire l'erreur quadratique moyenne $\mathbb{E}\left[\boldsymbol{e}\boldsymbol{e}^{\dagger}\right]$ tel que $\boldsymbol{e} = \boldsymbol{z} - \boldsymbol{s}$. Cela se fait en deux étapes. La première consiste à trouver le filtre $\boldsymbol{F_{mf}}$ en fonction de $\boldsymbol{B_{mf}}$, ce dernier sera considéré comme constant. La deuxième étape consiste à trouver le filtre optimal $\boldsymbol{B_{mf}}$ en respectant la contrainte de la forme (c'est-à-dire que $\boldsymbol{B_{mf}}$ doit être triangulaire supérieure avec $b_{ii} = 1, i = 1, ..., m$).

En se basant sur le calcul donné dans le chapitre 2, le filtre direct est le filtre MMSE égal à :

$$\boldsymbol{F_{mf}} = \boldsymbol{B_{mf}} \cdot \left(\boldsymbol{H}^{\dagger}\boldsymbol{H} + \frac{1}{\rho}\boldsymbol{I}\right)^{-1} = \boldsymbol{B_{mf}} \cdot \boldsymbol{\Sigma^{-1}} \tag{5.4}$$

avec $\rho = \frac{E_s}{\sigma^2_{w'}}$ est le rapport signal sur bruit et $\boldsymbol{\Sigma} = \boldsymbol{H}^{\dagger}\boldsymbol{H} + \frac{1}{\rho}\boldsymbol{I}$.

En remplaçant dans l'expression de $\boldsymbol{e}$, l'erreur quadratique minimale est donnée en fonction du filtre retour comme suit :

$$\begin{aligned} \boldsymbol{e_{min,F}} &= \boldsymbol{B_{mf}\Sigma^{-1}y_{mf} - B_{mf}\cdot s} \\ &= \boldsymbol{B_{mf}\cdot d} \end{aligned} \tag{5.5}$$

où $\boldsymbol{d = \Sigma^{-1}y_{mf} - s}$. La matrice $\boldsymbol{B_{mf}}$ étant triangulaire supérieure, l'erreur quadratique peut être vue comme une erreur de prédiction. Pour une composante d'ordre k, l'erreur s'écrit :

$$e_k = d_k + \sum_{j=k+1}^{m} b_{k,j}d_j \tag{5.6}$$

Par conséquent, $-\sum_{j=k+1}^{m} b_{k,j}d_j$ est l'estimation de d_k à partir des échantillons $d_{k+1}, ..., d_n$. $\boldsymbol{d}$ n'est plus un bruit blanc, sa matrice de covariance est égale à $\mathbb{E}\left[\boldsymbol{dd^\dagger}\right] = \boldsymbol{\Sigma^{-1}}$. Afin de blanchir le bruit résultant, il faut choisir le filtre $\boldsymbol{B_{mf}}$ telle que $\mathbb{E}\left[\boldsymbol{ee^\dagger}\right]$ est égale à une matrice diagonale. Si on écrit la décomposition de Cholesky de $\boldsymbol{\Sigma}$ comme suit :

$$\boldsymbol{\Sigma = B_{mf}^\dagger \Delta B_{mf}} \tag{5.7}$$

où Δ est diagonale, on peut vérifier que le bruit obtenu est blanc, avec :

$$\mathbb{E}\left[\boldsymbol{ee^\dagger}\right] = \mathbb{E}\left[\boldsymbol{B_{mf}dd^\dagger B_{mf}^\dagger}\right] = \boldsymbol{\Delta^{-1}} \tag{5.8}$$

En remplaçant dans (5.4), le filtre direct est donc égal à :

$$\boldsymbol{F_{mf} = \Delta^{-1}B_{mf}^{-\dagger}} \tag{5.9}$$

Les filtres $\boldsymbol{F_{mf}}$ et $\boldsymbol{B_{mf}}$ du MMSE-GDFE ne sont pas uniques. Une multiplication par une matrice diagonale non singulière donne un égaliseur MMSE-GDFE équivalent. En particulier, on multiplie par $\boldsymbol{\Delta^{1/2}}$ afin d'obtenir en sortie une erreur d'estimation de la matrice de covariance égale à la matrice identité $\boldsymbol{I}$. Le MMSE-GDFE appliqué à (5.3) donne :

$$\boldsymbol{z = Fy - B\hat{s}} \tag{5.10}$$

avec $\boldsymbol{B = \Delta^{1/2}B_{mf}}$ et $\boldsymbol{F = B^{-\dagger}H^\dagger}$.

5.2.2 Calcul simplifié du MMSE-GDFE

Les filtres forward et backward du MMSE-GDFE peuvent être directement obtenus en considérant la matrice de canal $\tilde{\boldsymbol{H}}$, tel que :

$$\tilde{\boldsymbol{H}} \triangleq \begin{bmatrix} \boldsymbol{H} \\ \frac{1}{\sqrt{\rho}}\boldsymbol{I} \end{bmatrix}$$

La décomposition QR de $\tilde{\boldsymbol{H}}$ donne :

$$\tilde{\boldsymbol{H}} = \tilde{\boldsymbol{Q}}\cdot\boldsymbol{R} = \begin{bmatrix} \boldsymbol{Q_1} \\ \boldsymbol{Q_2} \end{bmatrix}\cdot\boldsymbol{R}$$

avec $\tilde{\boldsymbol{Q}} \in \mathbb{C}^{(n+m)\times m}$, $\boldsymbol{R} \in \mathbb{C}^{m\times m}$, $\boldsymbol{Q_1} \in \mathbb{C}^{n\times m}$ représente la partie supérieure de $\tilde{\boldsymbol{Q}}$.
En prenant :

$$\begin{aligned} \boldsymbol{B} &= \boldsymbol{R} \\ \boldsymbol{F} &= \boldsymbol{Q_1^\dagger} \end{aligned} \tag{5.11}$$

Il est facile de vérifier que $\boldsymbol{B^\dagger B} = \tilde{\boldsymbol{H}}^\dagger \tilde{\boldsymbol{H}} = \boldsymbol{H^\dagger H} + \frac{1}{\rho}\boldsymbol{I} = \boldsymbol{\Sigma}$ et $\boldsymbol{F = B^{-\dagger} H^\dagger}$. Le nouveau système devient :

$$\begin{aligned} \boldsymbol{y'} &= \boldsymbol{Fy} \\ &= \boldsymbol{FHs + Fw} \\ &= \boldsymbol{FHs + Fw + Bs - Bs} \\ &= \boldsymbol{Bs + Fw - (B - FH)\, s} \\ &= \boldsymbol{Bs + w'} \\ &= \boldsymbol{Rs + w'} \end{aligned} \tag{5.12}$$

Il est à noter que le signal transformé $\boldsymbol{y'}$ n'est pas équivalent au signal initial $\boldsymbol{y}$. En général $\boldsymbol{Q_1}$ n'est pas une matrice orthogonale, de plus le bruit résultant $\boldsymbol{w' = Fw - (B - FH)\, s = Q_1^\dagger w - \left(R - Q_1^\dagger H\right) s}$ contient des composantes Gaussiennes données par $\boldsymbol{Q_1^\dagger w}$ et des composantes non Gaussiennes données par $\boldsymbol{-\left(R - Q_1^\dagger H\right) s}$ et dépendantes du symbole d'information. Toutefois, $\boldsymbol{w'}$ est un bruit blanc, ce qui n'affecte pas les performances du système [63].

Preuve :

$$\boldsymbol{E\left[w' w'^\dagger\right] = (B - FH)\,(B - FH)^\dagger + FF^\dagger}$$

Calculons d'abord $\boldsymbol{FH}$:

$$\begin{aligned} \boldsymbol{FH} &= \boldsymbol{Q_1^\dagger H} \\ &= \boldsymbol{B^{-\dagger} H^\dagger H} \\ &= \boldsymbol{B^{-\dagger}\left(H^\dagger H + I\right) - B^{-\dagger}} \\ &= \boldsymbol{B - B^{-\dagger}} \end{aligned}$$

Et :

$$\boldsymbol{FF^\dagger = B^{-\dagger} H^\dagger H B^{-1}}$$

Ainsi, la matrice de covariance du bruit $\boldsymbol{w'}$ est égale à :

$$\begin{aligned} \boldsymbol{E\left[w' w'^\dagger\right]} &= \boldsymbol{B^{-\dagger} B^{-1} + B^{-\dagger} H^\dagger H B^{-1}} \\ &= \boldsymbol{B^{-\dagger}\left(H^\dagger H + I\right) B^{-1}} \\ &= \tfrac{1}{\rho}\ \boldsymbol{I} \end{aligned}$$

A la sortie du MMSE-GDFE, le système à résoudre est $\boldsymbol{y' = Rs + w'}$. Le décodage se fait donc en fonction de la nouvelle matrice $\boldsymbol{R}$ qui est toujours de rang plein. De plus, $\boldsymbol{R}$ est mieux

conditionnée que la matrice originale $\boldsymbol{H}$. En effet puisque $\boldsymbol{R}^\dagger\boldsymbol{R} = \boldsymbol{H}^\dagger\boldsymbol{H} + \frac{1}{\rho}\boldsymbol{I}$, nous pouvons écrire :

$$\lambda^i_{R^\dagger R} = \lambda^i_{H^\dagger H} + \frac{1}{\rho} \tag{5.13}$$

tels que λ^i_M représentent les valeurs propres de $\boldsymbol{M}$. En considérant la première matrice $\boldsymbol{H}$, les valeurs propres peuvent avoir des valeurs très différentes. La multiplication de la constellation par la matrice du canal $\boldsymbol{H}$ risque d'avoir un réseau qui est plus étiré le long de quelques axes que d'autres et de transformer la structure de la constellation. Ainsi, il sera plus difficile de décoder un point de la constellation. Cependant, en considérant le nouveau système, le second terme en $\frac{1}{\rho}$ permet de mieux équilibrer les valeurs propres de $\boldsymbol{R}$ et mieux conserver la forme initiale de la constellation.

Soit la décomposition SVD de la matrice initiale $\boldsymbol{H}$:

$$\begin{aligned} \boldsymbol{y} &= \boldsymbol{H}\cdot\boldsymbol{s} + \boldsymbol{w} \\ &= \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^\dagger\cdot\boldsymbol{s} + \boldsymbol{w} \end{aligned}$$

telles que $\boldsymbol{U}$ et $\boldsymbol{V}$ sont unitaires avec des dimensions respectives $n\times n$ et $m\times m$ et $\boldsymbol{D}$ diagonale de dimension $n\times m$. Les éléments de $\boldsymbol{D}$ représentent les valeurs singulières de $\boldsymbol{H}$ ou encore les racines carrées des valeurs propres de $\boldsymbol{H}^\dagger\boldsymbol{H}$, $\sqrt{\lambda^i}, i = 1, ..., p$ avec $p = rang\left(\boldsymbol{H}^\dagger\boldsymbol{H}\right) \leq min(n_r, n_t)$. Une multiplication par $\boldsymbol{V}$ appliquée au signal en émission permet d'écrire :

$$\begin{aligned} \boldsymbol{U}^\dagger\boldsymbol{y} &= \boldsymbol{U}^\dagger(\boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^\dagger)\boldsymbol{V}\cdot\boldsymbol{s} + \boldsymbol{U}^\dagger\boldsymbol{w}\boldsymbol{V} \\ \tilde{\boldsymbol{y}} &= \boldsymbol{D}\cdot\boldsymbol{s} + \tilde{\boldsymbol{w}} \end{aligned}$$

Ou encore pour une composante i :

$$\tilde{y}_i = \sqrt{\lambda^i}\ \cdot s_i + \tilde{w}_i \tag{5.14}$$

Le système MIMO peut être vu comme p sous-canaux parallèles tels que les valeurs propres sont considérées comme les évanouissements. Les sous-canaux représentent le nombre de symboles qui peuvent être transmis simultanément. A partir de (5.14), une valeur propre nulle λ^i engendre une perte d'information du symbole correspondant s_i et si la valeur propre est très faible, le canal est très mauvais.

En considérant le pré-traitement MMSE-GDFE, l'équation précédente devient :

$$\tilde{y}_i = \sqrt{\lambda^i_{R^T R}}\cdot s_i + \tilde{w}_i \tag{5.15}$$

$\boldsymbol{R}$ est de rang plein, de plus d'après (5.13), les $\lambda^i_{R^T R}$ n'étant jamais nulles pour $i = 1, ..., n_t$, aucune perte d'information n'est alors possible. De plus, pour une valeur propre initiale faible, la nouvelle valeur propre obtenue est nécessairement plus grande, le canal est donc mieux conditionné.

5.2.3 MMSE-GDFE pour les codes ST algébriques

Le calcul du MMSE-GDFE présenté dans le paragraphe précédent est valable aussi bien pour un système MIMO non codé que pour un système codé. Mais l'utilisation ou non de codes reste transparente et donc inexploitée.

Nous allons à présent considérer un système MIMO employant les codes ST algébriques tels que les codes parfaits [22]. Ces codes sont caractérisés par une matrice génératrice unitaire. Nous allons exploiter cette propriété pour proposer un pré-traitement MMSE-GDFE moins complexe. Nous allons illustrer le calcul sur l'exemple du Golden code [64].

Le Golden code (GC) :

Le GC fait partie de la famille des codes parfaits [22]. Ces codes représentent plusieurs avantages par rapport aux codes connus jusqu'à présent. Ils offrent un rendement plein, une diversité maximale et des déterminaux minimaux ne s'évanouissant pas lorsque l'efficacité spectrale augmente. De plus, les codes parfaits présentent une bonne efficacité énergétique qui se traduit par une répartition uniforme de l'énergie dans le mot de code et des constellations ne présentant aucune perte de forme par rapport à celles émises.

Le GC est défini pour les systèmes à 2 antennes en émission et 2 antennes en réception, un mot de code s'écrit :

$$\boldsymbol{X_{2\times 2}} = \frac{1}{\sqrt{5}} \begin{bmatrix} \alpha(s_1 + \theta s_2) & \alpha(s_3 + \theta s_4) \\ i\bar{\alpha}(s_3 + \bar{\theta} s_4) & \bar{\alpha}(s_1 + \bar{\theta} s_2) \end{bmatrix} \tag{5.16}$$

avec $\theta = \frac{1+\sqrt{5}}{2}$ est appelé le nombre d'or (Golden number) , $\bar{\theta} = \frac{1-\sqrt{5}}{2}$, $\alpha = 1 + i - i\theta$ et $\bar{\alpha} = 1 + i - i\bar{\theta}$. On pose $n = T = 2$, T étant la longueur temporelle du code. Le système codé s'écrit alors :

$$\boldsymbol{y_{2\times 2}} = \boldsymbol{H_{2\times 2}}.\boldsymbol{X_{2\times 2}} + \boldsymbol{w_{2\times 2}} \tag{5.17}$$

Afin de décoder le signal reçu, on doit d'abord mettre le système sous la forme d'une représentation en réseau de points telle que explicitée dans le paragraphe 6.2 du chapitre 1. Pour cela, on applique une vectorisation du GC comme suit [64] :

$$\begin{aligned} y &= \frac{1}{\sqrt{5}} \begin{bmatrix} H_{2\times 2} & 0 \\ 0 & H_{2\times 2} \end{bmatrix} \begin{bmatrix} \alpha & \alpha\theta & 0 & 0 \\ 0 & 0 & i\bar{\alpha} & i\bar{\alpha}\bar{\theta} \\ 0 & 0 & \alpha & \alpha\theta \\ \bar{\alpha} & \bar{\alpha}\bar{\theta} & 0 & 0 \end{bmatrix} . \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{bmatrix} + w \\ &= \frac{1}{\sqrt{5}} \begin{bmatrix} H_{2\times 2} & 0 \\ 0 & H_{2\times 2} \end{bmatrix} \begin{bmatrix} \phi_1 & i\phi_2 \\ \phi_2 & \phi_1 \end{bmatrix} + w \\ &= \boldsymbol{H_1}.\boldsymbol{\phi}.\boldsymbol{s} + \boldsymbol{w} \end{aligned} \tag{5.18}$$

$\boldsymbol{H_1}$ est la matrice du canal équivalente et $\boldsymbol{\phi}$ est la matrice de codage unitaire à éléments dans $\mathbb{C}$, tels que $\boldsymbol{\phi_1} = \begin{bmatrix} \alpha & \alpha\theta \\ 0 & 0 \end{bmatrix}$ et $\boldsymbol{\phi_2} = \begin{bmatrix} 0 & 0 \\ \bar{\alpha} & \bar{\alpha}\bar{\theta} \end{bmatrix}$. La matrice génératrice du réseau de points est égale à $\boldsymbol{M} = \boldsymbol{H_1}.\boldsymbol{\phi}$.

Calcul du MMSE-GDFE pour le GC :

En partant de la formule (5.7), le filtre $\boldsymbol{B}$ est calculé par la décomposition de Cholesky de $\boldsymbol{\Sigma} = \boldsymbol{M}^\dagger\boldsymbol{M} + \frac{1}{\rho}\boldsymbol{I_4}$. En explicitant cette formule, on obtient :

$$\begin{aligned} \boldsymbol{\Sigma} &= \boldsymbol{M}^\dagger\boldsymbol{M} + \frac{1}{\rho}\boldsymbol{I_4} \\ &= (\boldsymbol{H_1}.\boldsymbol{\phi})^\dagger(\boldsymbol{H_1}.\boldsymbol{\phi}) + \frac{1}{\rho}\boldsymbol{I_4} \\ &= \boldsymbol{\phi}^\dagger\boldsymbol{H_1}^\dagger\boldsymbol{H_1}\boldsymbol{\phi} + \frac{1}{\rho}\boldsymbol{I_4} \end{aligned} \tag{5.19}$$

La matrice $\boldsymbol{\phi}$ étant unitaire, on peut écrire $\boldsymbol{\phi^\dagger \phi = I_4}$. En remplaçant dans (5.19), on obtient :

$$\begin{aligned}\mathbf{\Sigma} &= \phi^\dagger . \begin{bmatrix} H_{2\times2}^\dagger . H_{2\times2} & 0 \\ 0 & H_{2\times2}^\dagger . H_{2\times2} \end{bmatrix} . \phi + \frac{1}{\rho}\phi^\dagger . \begin{bmatrix} I_2 & 0 \\ 0 & I_2 \end{bmatrix} . \phi \\ &= \phi^\dagger . \begin{bmatrix} H_{2\times2}^\dagger . H_{2\times2} + \frac{1}{\rho} I_2 & 0 \\ 0 & H_{2\times2}^\dagger . H_{2\times2} + \frac{1}{\rho} I_2 \end{bmatrix} . \phi \end{aligned} \tag{5.20}$$

La décomposition de Cholesky de la sous-matrice $\boldsymbol{H_{2\times2}^\dagger H_{2\times2} + \frac{1}{\rho} I_2}$ donne :

$$\boldsymbol{H_{2\times2}^\dagger H_{2\times2} + \frac{1}{\rho} I_2 = R_{2\times2}^\dagger R_{2\times2}} \tag{5.21}$$

$\boldsymbol{R_{2\times2}}$ est une matrice triangulaire supérieure de dimension 2×2. La matrice $\mathbf{\Sigma}$ peut donc se mettre sous la forme suivante :

$$\begin{aligned}\Sigma &= \phi^\dagger \begin{bmatrix} H_{2\times2}^\dagger H_{2\times2} + \frac{1}{\rho} I_2 & 0 \\ 0 & H_{2\times2}^\dagger H_2 + \frac{1}{\rho} I_2 \end{bmatrix} \phi \\ &= \phi^\dagger \begin{bmatrix} R_{2\times2}^\dagger & 0 \\ 0 & R_{2\times2}^\dagger \end{bmatrix} \begin{bmatrix} R_{2\times2} & 0 \\ 0 & R_{2\times2} \end{bmatrix} \phi \\ &= \begin{bmatrix} \phi_1 & i\phi_2 \\ \phi_2 & \phi_1 \end{bmatrix}^\dagger \begin{bmatrix} R_{2\times2}^\dagger & 0 \\ 0 & R_{2\times2}^\dagger \end{bmatrix} \begin{bmatrix} R_{2\times2} & 0 \\ 0 & R_{2\times2} \end{bmatrix} \begin{bmatrix} \phi_1 & i\phi_2 \\ \phi_2 & \phi_1 \end{bmatrix} \\ &= \phi^\dagger R_{4\times4}^\dagger R_{4\times4} \phi \\ &= (R_{4\times4}\phi)^\dagger (R_{4\times4}\phi) \\ &= \begin{bmatrix} R_{2\times2}\phi_1 & iR_{2\times2}\phi_2 \\ R_{2\times2}\phi_2 & R_{2\times2}\phi_1 \end{bmatrix}^\dagger \begin{bmatrix} R_{2\times2}\phi_1 & iR_{2\times2}\phi_2 \\ R_{2\times2}\phi_2 & R_{2\times2}\phi_1 \end{bmatrix} \\ &= \boldsymbol{B^\dagger . B} \end{aligned} \tag{5.22}$$

Il est à noter que le filtre $\boldsymbol{B}$ obtenu n'est pas triangulaire supérieur à cause de la multiplication par la matrice de codage. Le filtre $\boldsymbol{F}$ est calculé par $\boldsymbol{F = B^{-\dagger} M^\dagger}$. Alors, on déduit :

$$\begin{aligned}\boldsymbol{F} &= (\boldsymbol{R_{4\times4}\phi})^{-\dagger} (\boldsymbol{H_1 . \phi})^\dagger \\ &= \boldsymbol{R_{4\times4}^{-\dagger} H_1^\dagger} \\ &= \begin{bmatrix} R_{2\times2}^{-\dagger} H_{2\times2}^\dagger & 0 \\ 0 & R_{2\times2}^{-\dagger} H_{2\times2}^\dagger \end{bmatrix} \end{aligned} \tag{5.23}$$

A partir des expressions de $\boldsymbol{B}$ et $\boldsymbol{F}$ trouvées, on peut voir que le calcul de ces deux filtres peut se réduire à un calcul matriciel sur des matrices de dimensions 2×2 au lieu de la dimension 4×4 du système équivalent, ce qui représente un gain important en complexité.

Calcul des complexités de la phase de pré-traitement pour le GC :

Pour le GC, n=2. La complexité de la phase de pré-traitement, définie en nombre d'opérations de multiplication, est calculée pour les trois méthodes présentées ci-haut (respectivement dans les sections 5.2.1, 5.2.2 et 5.2.3) comme suit :

Méthode directe

- Calcul de $\boldsymbol{\Sigma} = \boldsymbol{M}^{\dagger}\boldsymbol{M} + \frac{1}{\rho}\boldsymbol{I}$: $(2n)^3 + (2n)$ opérations
- Décomposition de Cholesky de $\boldsymbol{\Sigma}$, $\boldsymbol{\Sigma} = \boldsymbol{B}^{\dagger}B$: $\frac{1}{6}(2n)^3 + (2n)$ opérations
- Inversion de $\boldsymbol{B}$: $(2n)^3$ opérations
- Calcul de $\boldsymbol{F} = \boldsymbol{B}^{-\dagger}\boldsymbol{M}^{\dagger}$: $(2n)^3$ opérations

Méthode simplifiée

- Calcul de la matrice étendue $\tilde{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{M} \\ \frac{1}{\sqrt{\rho}}\boldsymbol{I} \end{bmatrix}$: $(2n)$ opérations
- Décomposition QR de $\tilde{\boldsymbol{M}}$: $\frac{2}{3}(4n)^3$ opérations

Nouvelle méthode

- Calcul de $\boldsymbol{H}_{\mathbf{2\times 2}}^{\dagger}\boldsymbol{H}_{\mathbf{2\times 2}} + \frac{\mathbf{1}}{\boldsymbol{\rho}}\boldsymbol{I}_{\mathbf{2}}$: $n^3 + n$ opérations
- Décomposition de Cholesky de $\boldsymbol{H}_{\mathbf{2\times 2}}^{\dagger}\boldsymbol{H}_{\mathbf{2\times 2}} + \frac{\mathbf{1}}{\boldsymbol{\rho}}\boldsymbol{I}_{\mathbf{2}}$: $\frac{1}{6}n^3 + n$ opérations
- Calcul de $\boldsymbol{R}_{\mathbf{2\times 2}}\boldsymbol{\phi}_{\mathbf{1}}$ et $\boldsymbol{R}_{\mathbf{2\times 2}}\boldsymbol{\phi}_{\mathbf{2}}$: $n^3 + n^3$ opérations
- Calcul de $\boldsymbol{R}_{\mathbf{2\times 2}}^{-\dagger}\boldsymbol{H}_{\mathbf{2\times 2}}^{\dagger}$: $n^3 + n^3$ opérations

Les complexités totales pour ces trois méthodes sont respectivement égales à $\frac{76}{3}n^3 + 4n$, $\frac{128}{3}n^3 + 2n$ et $\frac{31}{6}n^3 + 2n$.

Il est clair que la troisième méthode est la plus optimale en terme de complexité de calcul. La phase de pré-traitement pour les codes ST algébriques est largement réduite grâce à leurs propriétés de construction.

5.3 Pré-traitement à droite : la réduction

5.3.1 Définition

Soit Λ un réseau de points de $\mathbb{R}^n$. Un réseau de points possède une infinité de bases, soit $\boldsymbol{B_1}$ une base de Λ. $\boldsymbol{B_2}$ est une autre base du réseau si $\boldsymbol{B_2} = \boldsymbol{B_1}.\boldsymbol{U}$, avec $\boldsymbol{U}$ une matrice unitaire. Dans la figure 5.2, $\boldsymbol{B_1} = (\boldsymbol{u_1}, \boldsymbol{u_2}) = ((\mathbf{3},\mathbf{2}),(\mathbf{2},\mathbf{1}))$ et $\boldsymbol{B_2} = (\boldsymbol{v_1}, \boldsymbol{v_2}) = ((\mathbf{1},\mathbf{0}),(\mathbf{0},\mathbf{1}))$ sont deux bases de $\mathbb{Z}^2$. La relation entre les deux bases est donnée par la transformation suivante :

$$\boldsymbol{B_2} = \boldsymbol{B_1}.\boldsymbol{T} \tag{5.24}$$

Avec :

$$\boldsymbol{T} = \begin{bmatrix} -1 & 2 \\ 2 & -3 \end{bmatrix}$$

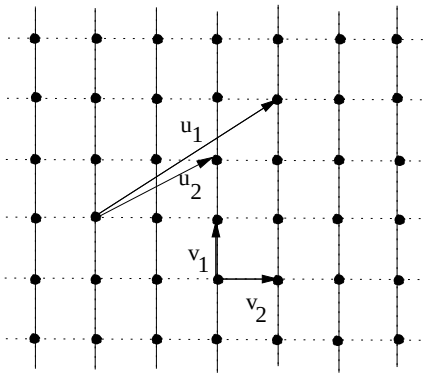


Figure 5.2 – Exemples de bases dans $\mathbb{Z}^2$

$\boldsymbol{B_2}$ est une meilleure base que $\boldsymbol{B_1}$, on dit que $\boldsymbol{B_1}$ est réduite en $\boldsymbol{B_2}$. Une base est considérée comme optimale si les vecteurs qui la composent sont les plus courts et les plus orthogonaux possibles. En général, la réduction de réseaux de points consiste à choisir la base optimale d'un réseau de points donné. La matrice $\boldsymbol{T}$ qui permet de trouver la base réduite est appelée matrice de réduction.

Le choix de la base est très important pour le décodage. Dans les paragraphes 2.3.1 du chapitre 2, nous avons illustré le cas d'une base orthogonale et d'une base non orthogonale avec le décodeur ZF. Il est clair que pour le premier cas le décodeur ZF est optimal. Plus la base est orthogonale, meilleures sont les performances du ZF. Ceci est aussi vrai pour les autres décodeurs. D'une manière générale, si la base est composée de vecteurs les plus courts et les plus orthogonaux possibles, alors :

- le décodage sous-optimal présente de meilleures performances
- le décodage optimal converge plus rapidement.

La réduction consiste à transformer une matrice du canal $\boldsymbol{H}$ donnée en une matrice $\tilde{\boldsymbol{H}}$ ayant des vecteurs plus courts et plus orthogonaux.

$$\boldsymbol{H} = \tilde{\boldsymbol{H}}\boldsymbol{T}^{-1} \tag{5.25}$$

Dans la littérature, plusieurs mesures d'orthogonalité ont été définies. Nous présentons dans ce qui suit les plus utilisées parmi elles.

5.3.2 Mesures d'orthogonalité

"Orthogonality defect" :

Soit une base $\boldsymbol{B}$ de réseaux de points de dimension n. L'inégalité de Hadamard donne :

$$|det\,(\boldsymbol{B})| \leq \|\boldsymbol{b}_1\| \ldots \|\boldsymbol{b}_n\|$$

tels que $\boldsymbol{b}_1, ..., \boldsymbol{b}_n$ sont les vecteurs colonnes de $\boldsymbol{B}$. Le cas d'égalité est obtenu si un des vecteurs

est nul ou encore si tous les vecteurs de $\boldsymbol{B}$ sont orthogonaux. Ainsi, nous pouvons définir la mesure d'orthogonalité (*orthogonality defect*) par la quantité :

$$\delta(\boldsymbol{B}) \triangleq \frac{\|\boldsymbol{b}_1\|^2 \dots \|\boldsymbol{b}_n\|^2}{|det\,(\boldsymbol{B}^\dagger \boldsymbol{B})|} \tag{5.26}$$

La matrice est d'autant plus orthogonale que cette mesure est petite.

A partir de l'équation (5.24), le déterminant de $\boldsymbol{B}^\dagger \boldsymbol{B}$ reste inchangé après réduction ($det(\boldsymbol{T}) = \pm 1$), $\delta(\boldsymbol{B})$ est alors proportionnel à la longueur des vecteurs colonnes de la matrice. La recherche de la quantité $\delta(\boldsymbol{B})$ optimale, ou encore la recherche de la base orthogonale, revient à la recherche des vecteurs les plus courts possibles [28], [65].

Mesure Seysen :

Soit $S(\boldsymbol{B})$ la quantité définie dans [66] par :

$$S\,(\boldsymbol{B}) \triangleq \sum_{i=1}^{n} \|\boldsymbol{b_i}\|^2 \|\boldsymbol{b_i^\star}\|^2 \tag{5.27}$$

$\boldsymbol{B^\star}$ est la base duale de $\boldsymbol{B}$. $S(\boldsymbol{B}) = n$ si et seulement si $\boldsymbol{B}$ est formée de vecteurs orthogonaux, sinon $S(\boldsymbol{B}) > n$. Une valeur petite de $S(\boldsymbol{B})$ indique que $\boldsymbol{B}$ et $\boldsymbol{B^\star}$ sont formées par des vecteurs courts. Cette mesure, appelée Seysen, permet à la fois de réduire la base $\boldsymbol{B}$ et sa base duale.

D'après [67], $\boldsymbol{B^\star}$ est définie tel que :

$$\left(\boldsymbol{b_i}, \boldsymbol{b_j^\star}\right) = \delta_{ij} \tag{5.28}$$

avec $\delta_{ij} = 1$ si $i = j$ et $\delta_{ij} = 0$ pour $i \neq j$. Il correspond au produit scalaire des vecteurs $\boldsymbol{b_i}$ et $\boldsymbol{b_j^\star}$. Pour un i fixé, nous avons $(\boldsymbol{b_i}, \boldsymbol{b_i^\star}) = 1$ avec :

$$\|\boldsymbol{b_i}\|\,\|\boldsymbol{b_i^\star}\|\,cos(\alpha) = 1 \tag{5.29}$$

où α est l'angle entre $\boldsymbol{b_i}$ et $\boldsymbol{b_i^\star}$. Soit S_i l'hyperplan de dimension $n-1$ formé par les vecteurs $(\boldsymbol{b}_1, \dots, \boldsymbol{b}_{i-1}, \boldsymbol{b}_{i+1}, \dots, \boldsymbol{b}_n)$. Par définition, $\boldsymbol{b_i^\star}$ est perpendiculaire à S_i. Ainsi, l'angle géométrique entre $\boldsymbol{b_i}$ et $\boldsymbol{b_i^\star}$ est égal à $\beta = \frac{\pi}{2} + \alpha$ [67]. La relation précédente devient :

$$\|\boldsymbol{b_i}\|\,\|\boldsymbol{b_i^\star}\| = \frac{1}{sin(\alpha)} \tag{5.30}$$

Si un vecteur $\boldsymbol{b_i}$ est indépendant des autres vecteurs de la base $\boldsymbol{b_j}$, $j \neq i$, l'angle entre $\boldsymbol{b_i}$ et l'hyperplan S_i est petit et la quantité $\frac{1}{sin(\alpha)}$ est par conséquent grande. Autrement, β est proche de $\frac{\pi}{2}$ et $\|\boldsymbol{b_i}\|\,\|\boldsymbol{b_i^\star}\| \cong 1$.

Nombre de conditionnement :

Nous rappelons la définition du nombre de conditionnement :

$$\kappa(\boldsymbol{H}) \triangleq \frac{\lambda_{max}}{\lambda_{min}} \tag{5.31}$$

telles que λ_{max} et λ_{min} sont respectivement les valeurs propres maximale et minimale de $\boldsymbol{H}$.

Une matrice mal-conditionnée présente un nombre $\kappa(\boldsymbol{H})$ très grand et une matrice bien-conditionnée a un nombre de conditionnement égal à 1.

$\kappa(\boldsymbol{H})$ mesure l'impact de la matrice de canal sur la puissance du bruit dans le cas de décodeurs linéaires. Si la matrice réduite $\tilde{\boldsymbol{H}}$ est orthogonale, $\kappa(\tilde{\boldsymbol{H}}) = 1$, par conséquent le bruit n'est plus amplifié. Ainsi, le décodage linéaire par rapport à la nouvelle matrice présente de meilleures performances que le système initial.

5.4 Méthodes de réduction existantes

En se basant sur les définitions des différentes mesures d'orthogonalité présentées plus haut, plusieurs algorithmes de réduction ont été proposés. Dans la littérature, il existe principalement trois méthodes de réduction : la réduction de Minkowski, la réduction Korkine-Zolotareff et la réduction Lenstra-Lenstra-Lovasz (LLL). Les deux premières permettent d'obtenir une base réduite optimale, cependant leur complexité est non-polynomiale ce qui rend peu pratique leur utilisation. La réduction LLL produit une base de vecteurs relativement courts et raisonnablement orthogonaux avec une complexité plus faible. Récemment, une nouvelle méthode de réduction a été proposée basée sur la minimisation de la mesure Seysen. Dans la suite, nous allons présenter le principe de chacune de ces techniques.

5.4.1 Réduction Minkowski

Le but de la réduction de Minkowski est de trouver une base de vecteurs les plus courts du réseau de points [68].

Soient Λ un réseau de points et $\boldsymbol{B}$ sa base. Cette base est la base réduite si elle vérifie les conditions suivantes :

1. $\boldsymbol{b_1}$ est le vecteur le plus court de Λ.
2. Pour $i = 1, ..., n$, $\boldsymbol{b_i}$ est le vecteur le plus court dans Λ indépendant des vecteurs $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{i-1}$ tel que l'ensemble $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{i-1}, \boldsymbol{b}_i)$ représente des vecteurs linéairement indépendants et peut être complété pour former une base du réseau.

De ce fait, la recherche de la base optimale se fait de manière progressive en considérant toutes les combinaisons linéaires possibles. Il est évident que la base trouvée contient les vecteurs les plus courts du réseau de points. Bien que la réduction de Minkowski a été intégrée dans des travaux sur la théorie des nombres, son application reste limitée à cause de sa grande complexité.

5.4.2 Réduction Korkine-Zolotareff

La réduction KZ est une variante de la réduction de Minkowski. Elle permet de trouver les vecteurs les plus courts des réseaux de points orthogonaux complémentaires engendrés par les vecteurs de la base déjà trouvés [69], [70].

La base réduite $\boldsymbol{B}$ vérifie :

1. $\boldsymbol{b_1}$ est le vecteur le plus court de Λ.
2. Soit Λ_i le réseau de points obtenu par projection de Λ_{i-1} sur le sous-espace $\mathbb{R}^{n-(i-1)}$ perpendiculaire à $\boldsymbol{b_{i-1}}$. $\boldsymbol{b_i}$ est le vecteur le plus court de Λ_i.

5.4.3 Réduction LLL

La réduction LLL est la réduction la plus classique et la plus utilisée jusqu'à maintenant. Contrairement aux techniques précédentes, elle ne produit pas la base optimale du réseau de points mais une base suffisamment optimale avec une complexité polynomiale [28], ce qui lui a valu d'être largement utilisée dans diverses applications actuelles telles que la théorie des nombres et la cryptographie.

Dans [71] et [34], un algorithme pratique de la réduction LLL, basé sur la technique d'orthogonalisation de Gram-Shmidt, a été proposé. Soit $\boldsymbol{B^\star} = (\boldsymbol{b_1^\star}, \ldots, \boldsymbol{b_n^\star})$ une base orthogonale obtenue par ce procédé. Les vecteurs $\boldsymbol{b_1^\star}, \ldots, \boldsymbol{b_n^\star}$ sont calculés par les formules suivantes :

$$\boldsymbol{b_i^\star} = \boldsymbol{b_i} - \sum_{j=1}^{i-1} \mu_{ij} \boldsymbol{b_j^\star} \tag{5.32}$$

$$\mu_{ij} = \frac{\left\langle \boldsymbol{b_i}, \boldsymbol{b_j^\star} \right\rangle}{\left\langle \boldsymbol{b_j^\star}, \boldsymbol{b_j^\star} \right\rangle} \tag{5.33}$$

La base $\boldsymbol{B^\star}$ est orthogonale mais elle n'est pas orthonormale.

Considérons par exemple un réseau de points de dimension 2. Sa base réduite est formée de deux vecteurs $\boldsymbol{B} = (\boldsymbol{b_1}, \boldsymbol{b_2})$. A partir de (5.32) et (5.33), $\boldsymbol{b_1} = \boldsymbol{b_1^\star}$ et $\boldsymbol{b_2} = \mu_{21}\boldsymbol{b_1^\star} + \boldsymbol{b_2^\star}$ vérifiant les conditions suivantes :

$$\frac{\langle \boldsymbol{b_1}, \boldsymbol{b_2} \rangle}{\|\boldsymbol{b_1}\|^2} \leq \frac{1}{2} \tag{5.34}$$

$$\|\boldsymbol{b_1}\|^2 \leq \|\boldsymbol{b_2}\|^2 \tag{5.35}$$

Afin d'accélérer la recherche de la base réduite, la deuxième inégalité est remplacée par une contrainte plus sélective :

$$\|\boldsymbol{b_1}\|^2 \leq \frac{4}{3} \|\boldsymbol{b_2}\|^2 \tag{5.36}$$

En général, pour un réseau de points Λ de dimension n, on définit par Λ_i le réseau de points engendré par les vecteurs $(\boldsymbol{b_i}, \boldsymbol{b_{i+1}})$. Une projection orthogonale de Λ_i sur l'hyperplan $(\boldsymbol{b_1}, ..., \boldsymbol{b_{i+1}})$ donne un réseau de base $(\boldsymbol{b_i}, \boldsymbol{b_{i+1}})$ tels que :

$$\boldsymbol{b_i}(i) = \boldsymbol{b_i^\star} \tag{5.37}$$

$$\boldsymbol{b_{i+1}}(i) = \boldsymbol{b_{i+1}^\star} + \boldsymbol{\mu_{i+1,i}} \boldsymbol{b_i^\star} \tag{5.38}$$

En appliquant les contraintes (5.34) et (5.36) sur $(\boldsymbol{b_i}, \boldsymbol{b_{i+1}})$, la base réduite $\boldsymbol{B}$ doit vérifier les conditions suivantes :

$$|\mu_{i,j}| \leq \frac{1}{2}, \qquad 1 \leq i, j \leq n \tag{5.39}$$

$$\left\|\boldsymbol{b_{i+1}^\star}\right\|^2 \leq \frac{4}{3} \left\|\boldsymbol{b_{i+1}^\star} + \mu_{i+1,i} \boldsymbol{b_i^\star}\right\|^2 \tag{5.40}$$

Ou encore :

$$\left\|\boldsymbol{b_{i+1}^\star}\right\|^2 \geq \left(\frac{3}{4} - \mu_{i+1,i}^2\right) \left\|\boldsymbol{b_i^\star}\right\|^2 \tag{5.41}$$

Nous notons que l'algorithme LLL applique le procédé d'orthogonalisation *"localement"* en considérant à chaque fois deux vecteurs adjacents. Par conséquent, la base réduite est quasi-orthogonale mais les vecteurs qui la constituent sont orthogonaux deux par deux. Par contre les bases engendrées par les réductions Minkowski et KZ sont parfaitement orthogonales cependant au prix d'une complexité très élevée.

En général, la réduction est appliquée sur la matrice de canal $\boldsymbol{H}$ comme définie en (5.25). Récemment, il a été démontré dans [72] que la réduction appliquée à $\boldsymbol{H}^{-\dagger}$ présente de meilleures performances comparées à la réduction LLL de $\boldsymbol{H}$. Afin d'illustrer concrètement l'intérêt de cette dernière méthode, nous allons comparer la détection ZF avec les deux types de réduction.

LLL type I

C'est la réduction LLL classique appliquée sur la matrice du canal. La matrice réduite $\tilde{\boldsymbol{H}}$ s'écrit :

$$\tilde{\boldsymbol{H}} = \boldsymbol{HT}$$

Le système (5.1) devient alors :

$$\boldsymbol{y} = \tilde{\boldsymbol{H}} \boldsymbol{T^{-1}} \cdot \boldsymbol{s} + \boldsymbol{w} \tag{5.42}$$

La détection ZF se fait en deux étapes. Nous calculons d'abord :

$$\hat{\boldsymbol{\rho}} = \left[\tilde{\boldsymbol{H}}^{-1} \boldsymbol{y}\right] \tag{5.43}$$

où $[x]$ est l'entier le plus proche de x. $\boldsymbol{T}$ étant unimodulaire, $\boldsymbol{T^{-1}}$ est également unimodulaire à éléments dans $\mathbb{Z}$. Le signal transmis est estimé par :

$$\hat{\boldsymbol{s}} = \boldsymbol{T} \hat{\boldsymbol{\rho}} \tag{5.44}$$

Le bruit résultant est égal à :

$$\tilde{\boldsymbol{w}} = \tilde{\boldsymbol{H}}^{-1} \boldsymbol{w} \tag{5.45}$$

En procédant par une décomposition SVD de la matrice $\tilde{\boldsymbol{H}}^{-1}$, le bruit est amplifié par les inverses des valeurs singulières de $\tilde{\boldsymbol{H}}$, λ_i^{-1}.

LLL type II

Dans ce cas, la réduction est appliquée sur $\boldsymbol{H}^{-\dagger}$. Soit $\boldsymbol{B}$ la matrice réduite, le système précédent s'écrit :

$$\boldsymbol{y} \quad = \boldsymbol{B}^{-\dagger} \quad \boldsymbol{T}^{\dagger} \cdot \boldsymbol{s} + \boldsymbol{w} \tag{5.46}$$

De la même façon, nous commençons par calculer :

$$\hat{\boldsymbol{\rho}} = \left[\boldsymbol{B}^{\dagger}\boldsymbol{y}\right] \tag{5.47}$$

On en déduit par la suite :

$$\hat{\boldsymbol{s}} = \boldsymbol{T}^{-\dagger}\hat{\boldsymbol{\rho}} \tag{5.48}$$

Par la suite, le bruit résultant est donné par :

$$\tilde{\boldsymbol{w}} = \boldsymbol{B}^{\dagger}\boldsymbol{w} \tag{5.49}$$

Le bruit est maintenant amplifié par les valeurs singulières de la matrice réduite. La réduction de $\boldsymbol{H}^{-\dagger}$ au lieu de $\boldsymbol{H}$ permet ainsi de réduire le bruit effectif et par conséquent d'avoir de meilleures performances. Nous allons donner dans la section suivante des exemples de simulations de ces deux méthodes.

Cependant il a été démontré dans [72] que les deux types de réduction LLL suivis par un décodage sous-optimal tel que ZF, permettent de récupérer la diversité en réception. Pour un système MIMO codé, un tel décodage permettra ainsi de bénéficier de la diversité en réception mais malheureusement pas celle en émission.

5.4.4 Réduction Seysen

La réduction Seysen a été proposée par Martin Seysen en 1993 [73]. Contrairement au LLL qui considère les vecteurs de la base deux par deux, la réduction Seysen permet de réduire tous les vecteurs de la base simultanément. Toutefois, la réduction se fait par rapport à la mesure Seysen définie dans (5.27) et non plus à la mesure d'orthogonalité (5.26).

Rappelons que la mesure Seysen tient compte à la fois de la base du réseau et de sa base duale. L'algorithme Seysen consiste ainsi à réduire simultanément ces deux bases. Avant de présenter la réduction Seysen, nous introduisons quelques définitions.

Forme quadratique :
Soit une base $\boldsymbol{B}$ d'un réseau de points défini sur $\mathbb{R}^n$. Nous appelons forme quadratique $\boldsymbol{A}$ associée à la base $\boldsymbol{B}$ la forme qui à chaque vecteur $\boldsymbol{x}$ associe :

$$\boldsymbol{x^T A x} = \boldsymbol{x^T B^T B x} \tag{5.50}$$

$\boldsymbol{A} = \boldsymbol{B^T B}$ est symétrique définie positive, $\boldsymbol{A^{-1}}$ l'est également. Nous désignons par a^*_{ij} la composante de la matrice $\boldsymbol{A^{-1}}$ à la ligne i et la colonne j.
Il est clair que $a_{ii} = \|\boldsymbol{b_i}\|^2$ et $a^*_{ii} = \|\boldsymbol{b^*_i}\|^2$ telle que $\boldsymbol{B^*}$ représente la base duale de $\boldsymbol{B}$ définie en (5.28).

La transformation linéaire appliquée à la base du réseau $\boldsymbol{B} \longmapsto \boldsymbol{B.T}$, $\boldsymbol{T} \in \mathbb{Z}^n$ est équivalente à la transformation associée à la forme quadratique suivante [67] :

$$\boldsymbol{A} \longmapsto \boldsymbol{T^T.A.T} \tag{5.51}$$

Réduction Seysen :

Pour toute forme quadratique $\boldsymbol{A} = (a_{ij})$, nous définissons la mesure Seysen par :

$$S(\boldsymbol{A}) = \sum_{i=1}^{n} a_{ii}.a_{ii}^{\star} \tag{5.52}$$

La base $\boldsymbol{B}$ du réseau et sa forme quadratique associée sont le produit de la réduction Seysen (*S-reduced*) si elles vérifient la condition suivante :

$$S(\boldsymbol{A}) \leq S(\boldsymbol{T^T.A.T}) \tag{5.53}$$

pour toute matrice $\boldsymbol{T} \in \mathbb{Z}^n$.

Cependant, il est difficile de trouver la matrice de transformation optimale dans tout l'espace $\mathbb{Z}^n$ pour une base donnée. Afin de garantir une complexité polynomiale de l'algorithme de réduction, une contrainte est imposée sur les matrices de transformation [66]. Les matrices de transformation à considérer sont de la forme :

$$\boldsymbol{T_{ij}^{\lambda}} = \boldsymbol{I_n} + \lambda \boldsymbol{U_{ij}}, \qquad avec\, i \neq j,\ \lambda \in \mathbb{Z} \tag{5.54}$$

telles que $\boldsymbol{I_n}$ est la matrice identité de dimension n et $\boldsymbol{U_{ij}}$ est la matrice définie par :

$$\boldsymbol{U_{ij}} \triangleq [u_{kl}], \qquad u_{kl} = \begin{cases} 1 & si\, k = i \quad et\, l = j \\ 0 & si\, k \neq i \quad et\, l \neq j \end{cases} \tag{5.55}$$

Par construction, la $\boldsymbol{T_{ij}^{\lambda}}$ a des éléments diagonaux égaux à 1 et un élément non nul en dehors de la diagonale. La multiplication à droite de $\boldsymbol{B}$ par la matrice de transformation ajoute simplement λ fois la i^{me} colonne de $\boldsymbol{B}$ à sa j^{me} colonne. Soit $\boldsymbol{B} = (\boldsymbol{b_1}, \ldots, \boldsymbol{b_n})$, alors la base réduite s'écrit :

$$\boldsymbol{B}\boldsymbol{T_{ij}^{\lambda}} = \boldsymbol{B'} \tag{5.56}$$

$$= (\boldsymbol{b_1}, \ldots, \boldsymbol{b_{j-1}}, \boldsymbol{b_j} + \boldsymbol{\lambda b_i}, \boldsymbol{b_{j+1}}, \ldots, \boldsymbol{b_n}) \tag{5.57}$$

Une itération de l'algorithme consiste alors à calculer la matrice $\boldsymbol{T_{ij}^{\lambda}}$. On dit que la forme quadratique $\boldsymbol{A}$ est le produit de la réduction Seysen (S_2-*reduced*) si elle vérifie :

$$S(\boldsymbol{A}) \leq S(\boldsymbol{T_{ji}^{\lambda}.A.T_{ij}^{\lambda}}), \qquad 1 \leq i, j \leq n,\ \lambda \in \mathbb{Z} \tag{5.58}$$

Choix des vecteurs $(\boldsymbol{b_i}, \boldsymbol{b_j})$:

A chaque itération, un couple de vecteurs $(\boldsymbol{b_i}, \boldsymbol{b_j})$ doit être réduit. Il est nécessaire de trouver une paire d'indices (i, j) pour lesquels il existe une matrice $\boldsymbol{T_{ij}^{\lambda}}, \lambda \neq 0$ répondant à la condition précédente.
Dans [67], deux approches ont été proposées afin de choisir le couple de vecteurs optimal à chaque itération :

1. La première considère le premier couple de vecteurs disponibles $(\boldsymbol{b_i}, \boldsymbol{b_j})$ tel que $\lambda_{ij} \neq 0$ est donnée par :

$$\lambda_{ij} = \left\lceil \frac{1}{2} \left(\frac{a_{ij}^{\star}}{a_{jj}^{\star}} - \frac{a_{ij}}{a_{ii}} \right) \right\rfloor \tag{5.59}$$

   Une fois un couple de vecteurs est trouvé, la matrice de transformation $\boldsymbol{T_{ij}^{\lambda}}$ est calculée et appliquée à la base du réseau selon la formule (5.57). L'algorithme reprend la recherche d'un nouveau couple $(\boldsymbol{b_i}, \boldsymbol{b_j})$. Cette méthode s'appelle *Lazy Selection*.

2. Une autre méthode possible est de calculer pour chaque $(\boldsymbol{b_i}, \boldsymbol{b_j})$ la quantité $\Delta(i,j,\lambda)$ définie par :

$$\Delta(i,j,\lambda) = S\left(\boldsymbol{T_{ji}^{\lambda}}.\boldsymbol{A}.\boldsymbol{T_{ij}^{\lambda}}\right) - S\left(\boldsymbol{A}\right) \tag{5.60}$$

La matrice de transformation $\boldsymbol{T_{ij}^{\lambda}}$ doit avoir $\Delta(i,j,\lambda) < 0$. Le couple de vecteurs à choisir est donc celui qui minimise $\Delta(i,j,\lambda)$. Cette méthode s'appelle *Greedy Selection*. Il est clair que la complexité de la réduction dépend de la valeur de λ. Dans [67], il a été démontré que $\lambda = \left\lceil \frac{1}{2}\left(\frac{a_{ij}^{\star}}{a_{jj}^{\star}} - \frac{a_{ij}}{a_{ii}}\right)\right\rceil$ donne la réduction la plus optimale.

Dans la suite, nous utilisons l'algorithme de réduction Seysen en utilisant la deuxième approche.

5.4.5 Comparaison de la réduction LLL et de la réduction Seysen

Après avoir présenté les deux algorithmes de réduction LLL et Seysen, nous nous proposons de les comparer afin de choisir celui qui convient le mieux à notre système de transmission.

Il existe différents critères de comparaison. Le but de la réduction est d'avoir la base la plus orthogonale possible. Nous nous intéressons alors à la complexité de la réduction et de la qualité de la base retournée.

Nous avons calculé dans le tableau suivant la complexité de chacun des deux algorithmes. Ceux-ci effectuent essentiellement des opérations sur les colonnes de la base initiale, la complexité calcule le nombre moyen d'opérations matricielles pour les différentes dimensions considérées du système MIMO.

Système MIMO	4×4	6×6	8×8
LLL	13	28	43
Seysen *(Lazy)*	21	44	69
Seysen *(Greedy)*	12	24	36

Table 5.1 – Complexité de la réduction LLL et de la réduction Seysen

Il est clair que la deuxième approche Seysen est moins complexe que la première grâce à l'ajout de la contrainte (5.60). Nous pouvons voir aussi que celle-ci a une complexité équivalente, voire légèrement inférieure à celle du LLL.

De plus, la réduction Seysen engendre une base plus orthogonale que celle obtenue par la réduction LLL. En effet, cette dernière permet d'avoir une base avec des vecteurs orthogonaux deux par deux, alors que la réduction Seysen permet d'avoir une matrice *'globalement'* orthogonale.

L'utilisation de la réduction Seysen avec un décodeur sous-optimal donnera ainsi de meilleures performances. Dans la section suivante, nous nous proposons d'étudier les performances d'une chaine de transmission MIMO intégrant toutes les techniques de pré-traitement que nous avons présentées suivies de différents décodeurs optimaux et sous-optimaux.

5.5 Performances du pré-traitement pour une transmission MIMO

5.5.1 Schéma de transmission complet

Le schéma de transmission que nous allons considérer est le système MIMO employant un multiplexage spatial. A la réception, la chaine de décodage comporte un pré-traitement à gauche suivi d'un pré-traitement à droite et d'un décodeur de réseaux de points.

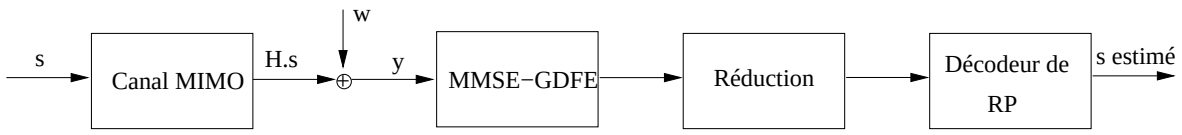


Figure 5.3 – Schéma de transmission MIMO avec une chaine complète de décodage

5.5.2 Performances du pré-traitement à gauche : le MMSE-GDFE

Nous commençons par étudier l'apport du MMSE-GDFE avec un décodage sous-optimal et avec un décodage optimal.

MMSE-GDFE + ZF-DFE

Nous allons appliquer le décodeur ZF-DFE avec la matrice du canal équivalente obtenue à la sortie du MMSE-GDFE. Ceci revient à appliquer le décodeur ZF-DFE sur le nouveau système obtenu dans l'équation (5.12). Dans la figure 5.4, nous avons tracé le taux d'erreur par bit pour un système MIMO 4×4 utilisant une constellation 4-QAM. Nous pouvons voir que les performances sont nettement améliorées en employant le pré-traitement à gauche. Pour un taux d'erreur égal à 10^{-3}, nous constatons un gain de 4dB. Cependant, le MMSE-GDFE n'apporte pas de gain de diversité.

MMSE-GDFE + ML

Le MMSE-GDFE permet d'avoir une matrice équivalente mieux conditionnée. Ceci permet d'améliorer les performances du décodeur sous-optimal. Pour un décodeur optimal, la solution ML est systématiquement trouvée. Le rôle du MMSE-GDFE dans ce cas est d'accélérer la phase de recherche de cette solution. Afin d'illustrer cette propriété, nous avons tracé dans la figure 5.5 la complexité de la phase de recherche (calculée en nombre d'opérations de multiplications) pour un système 2×2 employant une 16-QAM décodé avec un décodeur ML (exemple le décodeur par sphères).

Grâce à une matrice mieux conditionnée, le MMSE-GDFE permet de réduire le nombre d'itérations nécessaires de l'algorithme ML pour trouver la solution. Le gain est observé en particulier pour les RSB faibles. Pour les RSB forts, la puissance du signal est suffisamment grande que le bruit est négligé. Le décodage est aisément effectué quelle que soit la matrice du canal, ce qui explique que les complexités du décodage avec et sans pré-traitement se rejoignent. Le pré-traitement à gauche est par conséquent utile en particulier pour les RSB faibles.

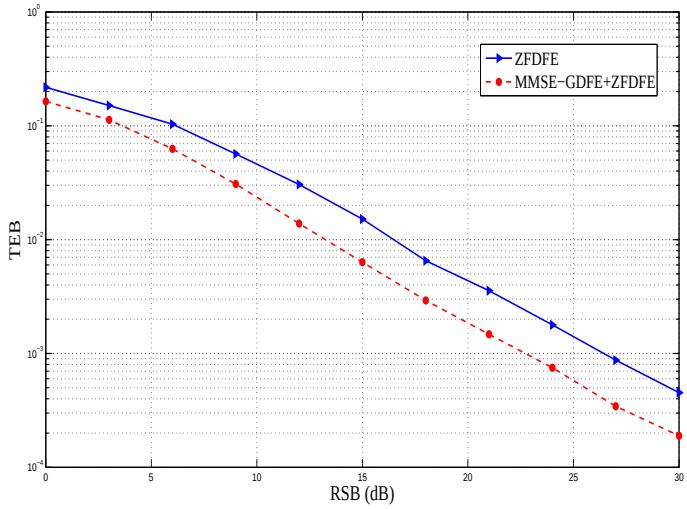


Figure 5.4 – Performances du MMSE-GDFE appliqué à un décodeur sous-optimal

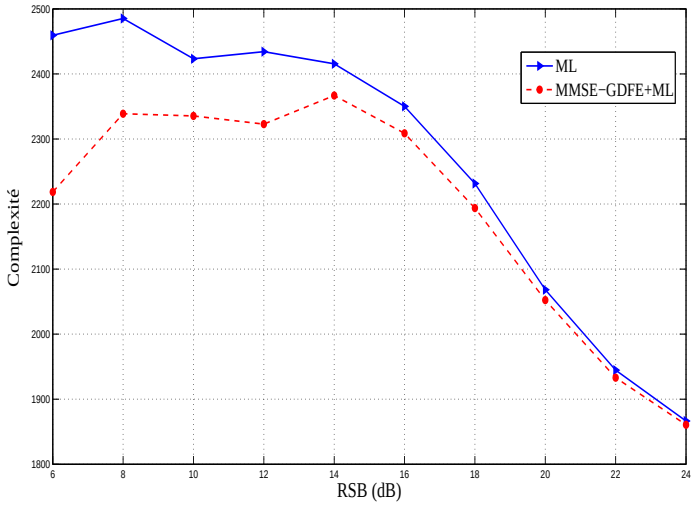


Figure 5.5 – Complexité du MMSE-GDFE appliqué à un décodeur ML

5.5.3 Performances du pré-traitement à droite : la réduction

Réduction LLL *vs* réduction Seysen

Nous nous proposons ici d'étudier les performances de la réduction avec et sans pré-traitement MMSE-GDFE. Nous commençons par comparer les réductions LLL et Seysen.

Dans la figure 5.6, nous pouvons voir que la réduction Seysen est meilleure que LLL. Ce gain est du au fait que l'algorithme de réduction LLL donne une base avec des vecteurs orthogonaux deux par deux alors que l'algorithme Seysen permet de procurer une base globalement orthogonale. Ces réductions offrent toutes les deux la diversité en réception qui est égale dans ce cas à la diversité maximale puisque nous considérons le cas d'un système MIMO non codé.

Cependant, il existe deux variantes de la réduction LLL. Nous évaluons dans ce qui suit les performances correspondant aux deux types de réductions LLL.

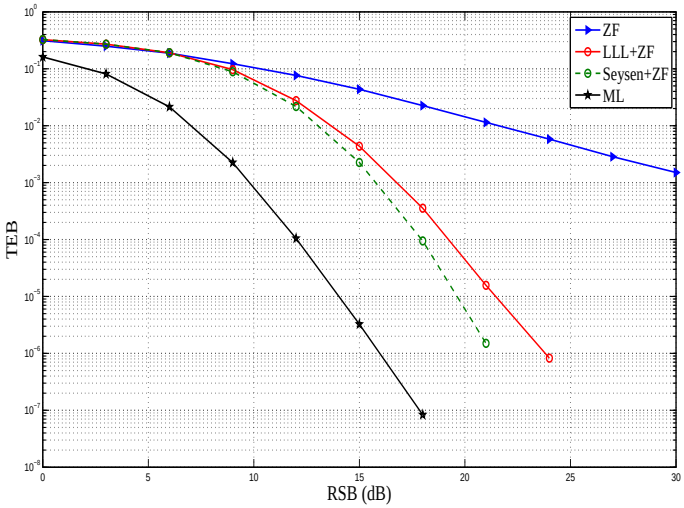


Figure 5.6 – Performances de la réduction LLL et la réduction Seysen pour un système MIMO 6×6 employant un multiplexage spatial et une 4-QAM

Réduction LLL type I *vs* LLL type II

Dans la figure 5.7, nous précédons la réduction par le traitement MMSE-GDFE et nous traçons le taux d'erreur par bit obtenu pour les deux types de réduction LLL appliqués à un décodage ZF. Nous rappelons que la première réduction permet de réduire la matrice du canal alors que la deuxième réduction LLL est appliquée sur l'inverse transposée de la matrice du canal.

A partir des simulations, nous vérifions que la réduction LLL de type II est plus performante que la réduction LLL de type I. En effet, celle-ci permet de réduire le bruit effectif résultant ($\boldsymbol{B^H w}$) qui est plus faible dans ce cas que le bruit résultant de la réduction de la matrice du canal. Les deux méthodes présentent néanmoins le même ordre de diversité.

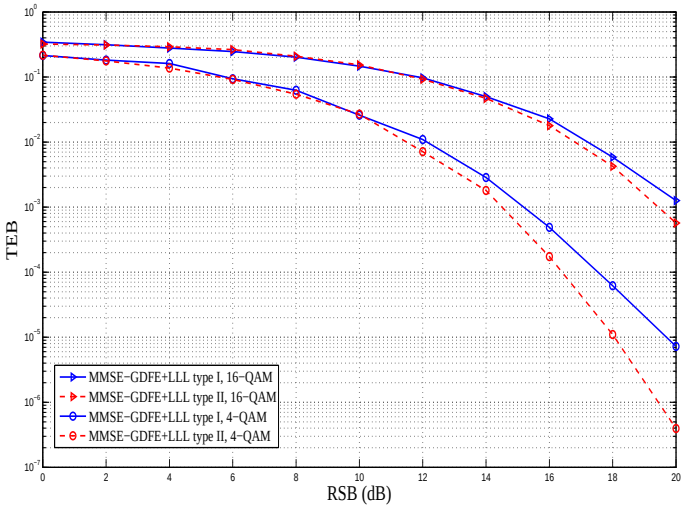


Figure 5.7 – Performances de la réduction LLL suivie d'un décodeur sous-optimal pour un système MIMO 6×6 employant un multiplexage spatial

5.5.4 Combinaison des deux techniques de pré-traitement

Nous présentons dans la figure 5.8 les résultats de simulation obtenus en considérant une chaine de transmission complète. Nous considérons ici un système MIMO 6×6 employant un multiplexage spatial. Les symboles d'informations sont modulés par la modulation 4-QAM. Nous appliquons un pré-traitement à gauche pour transformer le système initial en un système mieux conditionné et un pré-traitement à droite pour obtenir une meilleure base construite de vecteurs les plus orthogonaux possibles.

En précédant la réduction par le pré-traitement MMSE-GDFE, les performances sont améliorées de 2dB comparées à celles obtenues en utilisant uniquement la réduction. Grâce à l'application du MMSE-GDFE, la constellation finie va être vue comme un réseau infini, ce qui permet de résoudre le problème de *Shaping* lorsqu'on utilise ensuite la réduction. En effet, la réduction modifie les bornes de la constellation, le fait de la précéder par le MMSE-GDFE ramène le problème à la résolution d'un réseau de points infini.

Nous observons cependant, que lorsqu'on utilise le décodeur ZF-DFE, nous obtenons les mêmes performances en considérant une réduction LLL ou une réduction Seysen. Les résultats obtenus montrent alors l'apport de la réduction Seysen par rapport à la réduction LLL particulièrement dans le cas de décodeurs linéaires. Ces performances sont à 3dB du ML mais obtenues avec une plus faible complexité. Toutefois, puisque l'algorithme Seysen est moins complexe que l'algorithme de LLL, il est plus pratique d'utiliser cette première réduction.

Les mêmes observations sont vraies dans le cas de systèmes de plus grandes dimensions. Nous illustrons ces résultats dans la figure 5.9 par des simulations obtenues pour un système MIMO de dimension 8×8.

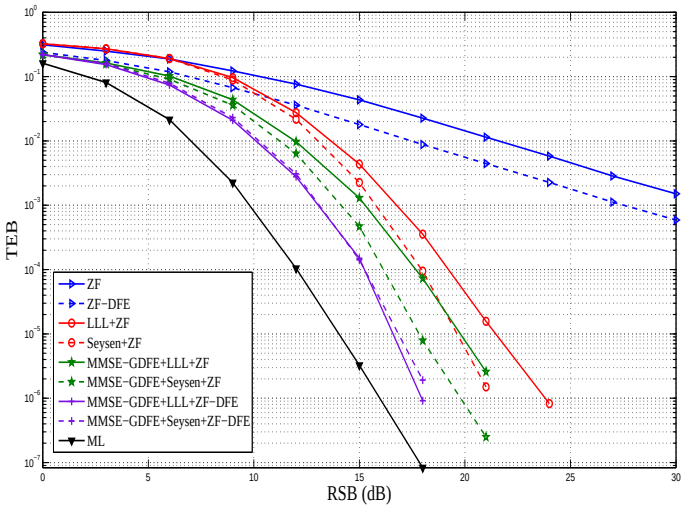


Figure 5.8 – Combinaison des deux techniques de pré-traitement MMSE-GDFE et réduction pour un système MIMO 6×6 employant un multiplexage spatial et une 4-QAM

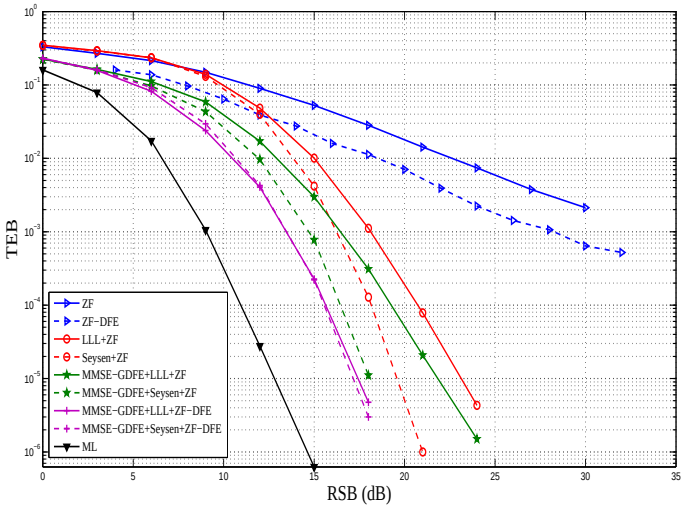


Figure 5.9 – Combinaison des deux techniques de pré-traitement MMSE-GDFE et réduction pour un système MIMO 8×8 employant un multiplexage spatial et une 4-QAM

5.5.5 Décodage adaptatif + pré-traitement

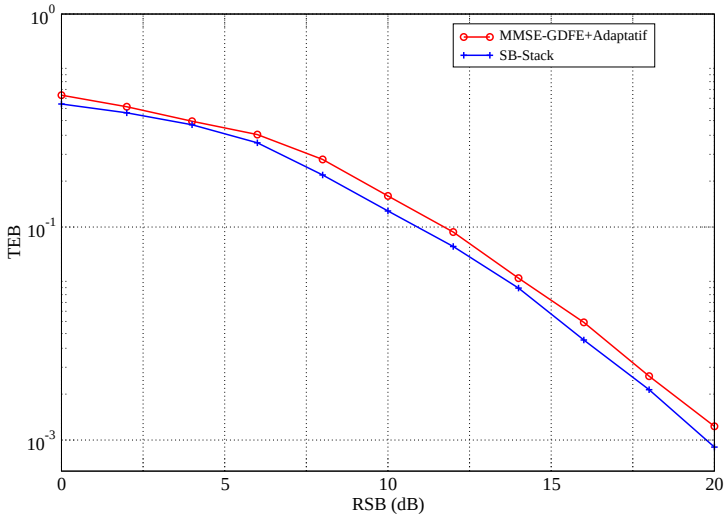
L'objectif des schémas de décodage précédents est soit d'améliorer les performances soit de réduire la complexité. Le décodeur adaptatif tel que présenté dans le chapitre 4 permet de choisir à chaque transmission le décodeur le plus approprié parmi plusieurs. Afin d'avoir une complexité constante tout en garantissant de bonnes performances, nous proposons d'employer le pré-traitement avec le décodeur adaptatif.

Le schéma de décodage adaptatif que nous considérons dans ce cas sélectionne soit le MMSE-GDFE suivi du détecteur ZF soit le MMSE-GDFE suivi du décodeur ML (représenté ici par le SB-Stack). Le critère de sélection que nous considérons est la probabilité de coupure du canal présentée dans la section 4.3.2 du chapitre 4.

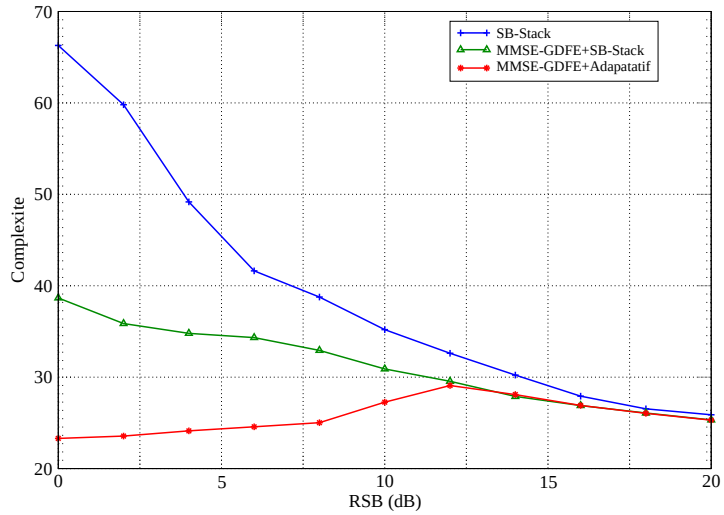
Le pré-traitement permet d'avoir une meilleure matrice du canal. Ainsi, le canal se trouve moins souvent dans la région de coupure ce qui permet de sélectionner plus souvent le décodeur ML. Par conséquent, les performances obtenues sont proches des performances optimales et elles sont à 0.5 dB seulement du ML (figure 5.10.a).

Le recours au décodeur ML risque d'augmenter la complexité. Cependant, l'utilisation du pré-traitement à gauche permet au décodeur de converger plus rapidement vers la solution et de diminuer ainsi sa complexité de décodage. Ceci peut être vu à partir de la figure 5.10.b où le pré-traitement appliqué au décodeur présente un gain de complexité sans pour autant sacrifier les performances.

De plus, l'utilisation du pré-traitement avec les techniques d'adaptation du décodage permet de diminuer encore cette complexité au prix d'une faible dégradation des performances. Pour les grands RSB, les complexités sont similaires. En effet, les conditions de transmission sont bonnes, par conséquent la matrice du canal initiale est bien conditionnée, le système se comporte pratiquement de la même façon que l'on applique ou pas un pré-traitement ou une adaptation.



(a)



(b)

FIGURE 5.10 – Combinaison des techniques de pré-traitement et d'adaptation pour un système MIMO 2×2 employant un multiplexage spatial et une 16-QAM

5.6 Conclusion

Le pré-traitement est un outil très performant qui permet de mieux conditionner la matrice du canal. Dans ce chapitre, nous avons passé en revue les techniques de pré-traitement qui existent dans la littérature. Nous avons distingué deux types de pré-traitement : le pré-traitement à gauche qui consiste à l'application du MMSE-GDFE et le pré-traitement à droite qui consiste à

la réduction de la matrice du canal.

Nous avons proposé d'appliquer le pré-traitement aux systèmes de transmission MIMO employant un codage espace-temps. En considérant les codes ST algébriques, nous avons proposé une méthode simplifiée du calcul du MMSE-GDFE en se basant sur la structure en blocs qui caractérise ces codes. Ensuite, nous avons étudié les algorithmes de réduction les plus utilisées. Nous avons conclu que la réduction Seysen présente une meilleure complexité et offre une meilleure qualité de réduction que la réduction LLL.

L'intégration des techniques de pré-traitement dans la chaine de décodage offre un gain significatif en performances et/ou en complexité par rapport à une chaine comportant uniquement un décodeur, quel que soit le décodeur optimal ou sous-optimal employé.

Conclusions et perspectives

Conclusions

Dans ce mémoire, nous avons étudié le décodage des systèmes MIMO utilisant un codage espace-temps à dispersion linéaire, dans le cas d'une transmission cohérente et un canal quasi-statique.

La représentation en réseaux de points de ces codes permet de les décoder par les algorithmes de réseaux de points. Les plus connus d'entre eux sont le décodeur par sphères (SD) et la stratégie de Schnorr-Euchner (SE). Récemment, des décodeurs basés sur des algorithmes de recherche séquentiels dans un arbre ont été généralisés pour le décodage des systèmes MIMO, à savoir les décodeurs Fano et Stack. Ce dernier permet, grâce à l'utilisation d'une pile, de stocker tous les noeuds parcourus dans l'arbre.

Dans le cas de réseaux de points, où les symboles appartiennent au réseau infini $\mathbb{Z}$, ou bien dans le cas de constellations de grandes tailles, l'application du décodeur Stack original et très complexe voire impossible. Pour cela, nous avons proposé une première approche combinant les stratégies de recherche du Stack et du SE, appelée M-Stack. Cet algorithme présente des performances sous-optimales. Nous avons alors proposé une deuxième approche qui combine la stratégie de recherche du Stack et la région de recherche du SD, que nous avons appelée SB-Stack (Spherical Bound-Stack decoder). Le décodeur proposé permet d'avoir les performances ML avec une complexité plus faible que les décodeurs originaux, présentant un gain moyen de 30% par rapport au SD et 70% par rapport au décodeur Stack.

Par la suite, nous avons donné une version paramétrée du décodeur SB-Stack, qui consiste à modifier la métrique ML cumulée en fonction d'un biais, ce qui permet d'offrir une multitude de performances allant du ZF-DFE au ML avec des complexités croissantes.

Dans les schémas de transmission pratiques, le code ST est généralement utilisé conjointement avec un code correcteur d'erreurs. Celui-ci est décodé par les décodeurs à entrées et sorties souples. Pour cela, nous avons modifié le décodeur SB-Stack afin de générer des sorties sous forme de taux de vraisemblance logarithmiques (LLR). Nous avons vérifié que ce dernier offre les meilleures performances par rapport aux décodeurs à sorties souples existants, tels que le *Shifted Sphere Decoder* (SSD) ou le *List Sphere Decoder* (LSD). De plus, le décodeur proposé permet de contrôler parfaitement la taille de la liste des solutions retournées.

Nous avons constaté que les décodeurs précédents présentent une complexité très variable et des temps de décodage très différents pour les faibles RSB et les forts RSB. Cependant, il est important en pratique d'avoir un temps de décodage constant. Une solution simple est d'arrêter

le système après un certain délai fixé, cependant cette solution ne présente aucune garantie sur les performances retournées.

Pour pallier à ce problème, nous avons proposé un décodeur adaptatif qui permet de sélectionner le décodeur le plus adéquat à chaque réalisation du canal et pour toute valeur du RSB. Le décodeur adaptatif permet d'obtenir les performances souhaitées avec une complexité quasi-constante. Par la suite, nous avons présenté un schéma pratique d'implémentation du décodeur adaptatif utilisant le SB-Stack paramétré.

Pour clore notre étude sur les décodeurs MIMO, nous nous sommes intéressés à la phase de pré-traitement. A la sortie de cette phase, la matrice du canal équivalente est mieux conditionnée ce qui permet de réduire la complexité du décodeur optimal et améliorer les performances du décodeur sous-optimal. Nous avons présenté et étudié les performances d'une chaine complète de décodage utilisant diverses techniques de pré-traitement combinées avec les décodeurs ST proposés précédemment. Nous avons d'abord donné une nouvelle méthode de calcul du MMSE-GDFE moins complexe et adaptée à la structure spécifique du Golden code. Nous avons aussi appliqué le pré-traitement au décodeur adaptatif que nous avons proposé, ce qui nous a permis d'améliorer les performances et obtenir une complexité quasi-constante et encore plus faible.

Perspectives

Le pré-traitement est un outil très efficace pour améliorer les performances des systèmes MIMO en gardant une complexité raisonnable. Pour le MMSE-GDFE, nous avons proposé une méthode de calcul simplifiée pour les codes ST algébriques, en particulier le Golden code. Il serait également intéressant d'exploiter la structure spécifique de ces codes dans le but d'optimiser la complexité d'implémentation du pré-traitement.

Nous avons montré par les résultats de simulation que la réduction Seysen permet d'offrir la diversité en réception. Nous nous proposons alors d'étudier de plus près la diversité de Seysen dans le cas des systèmes MIMO non codés et codés et d'apporter la preuve théorique sur l'ordre de diversité qu'elle offre.

Les décodeurs que nous avons proposés peuvent être appliqués à tout code ST à dispersion linéaire. Toutefois, nous nous proposons dans le futur d'exploiter les structures de certains codes ST afin de concevoir des décodeurs adaptés moins complexes et plus performants. Une autre idée toute aussi intéressante consiste à proposer des codes ST plus facilement décodables tels que les codes Silver et ses variantes [74], [75], [76] et leurs décodeurs appropriés.

Annexe A : Calcul du filtre MMSE

Nous reprenons le système défini dans le chapitre 2. Le principe du MMSE est de trouver le filtre optimal F qui réduit l'erreur quadratique moyenne donnée par :

$$F_{opt} = \arg\min_{F} \left(E\left\{ \|F \cdot y - s\|^2 \right\} \right) \tag{5.61}$$

Notons par $\varepsilon^2 = E\left\{ \|F \cdot y - s\|^2 \right\}$ l'erreur quadratique moyenne. En développant l'expression de ε^2, nous obtenons :

$$\begin{aligned}
\varepsilon^2 &= E\left\{ \|F \cdot y - s\|^2 \right\} \\
&= E\left\{ \|F \cdot (H \cdot s + w) - s\|^2 \right\} \\
&= E\left\{ ((F \cdot H - I) \cdot s + F \cdot w) \cdot ((F \cdot H - I) \cdot s + F \cdot w)^\dagger \right\} \\
&= E\left\{ (F \cdot H - I) \cdot s \cdot s^\dagger \cdot (F \cdot H - I)^\dagger \right\} + E\left\{ F \cdot w \cdot w^\dagger \cdot F^\dagger \right\} \\
&= E_s \cdot (F \cdot H - I) \cdot (F \cdot H - I)^\dagger + \sigma_w^2 \cdot F \cdot F^\dagger \\
&= E_s \cdot F \cdot (H \cdot H^\dagger + \sigma_w^2 \cdot I) \cdot F^\dagger (F \cdot H - I)^\dagger - E_s \cdot F \cdot H - E_s \cdot (F \cdot H)^\dagger + E_s \cdot I
\end{aligned}$$

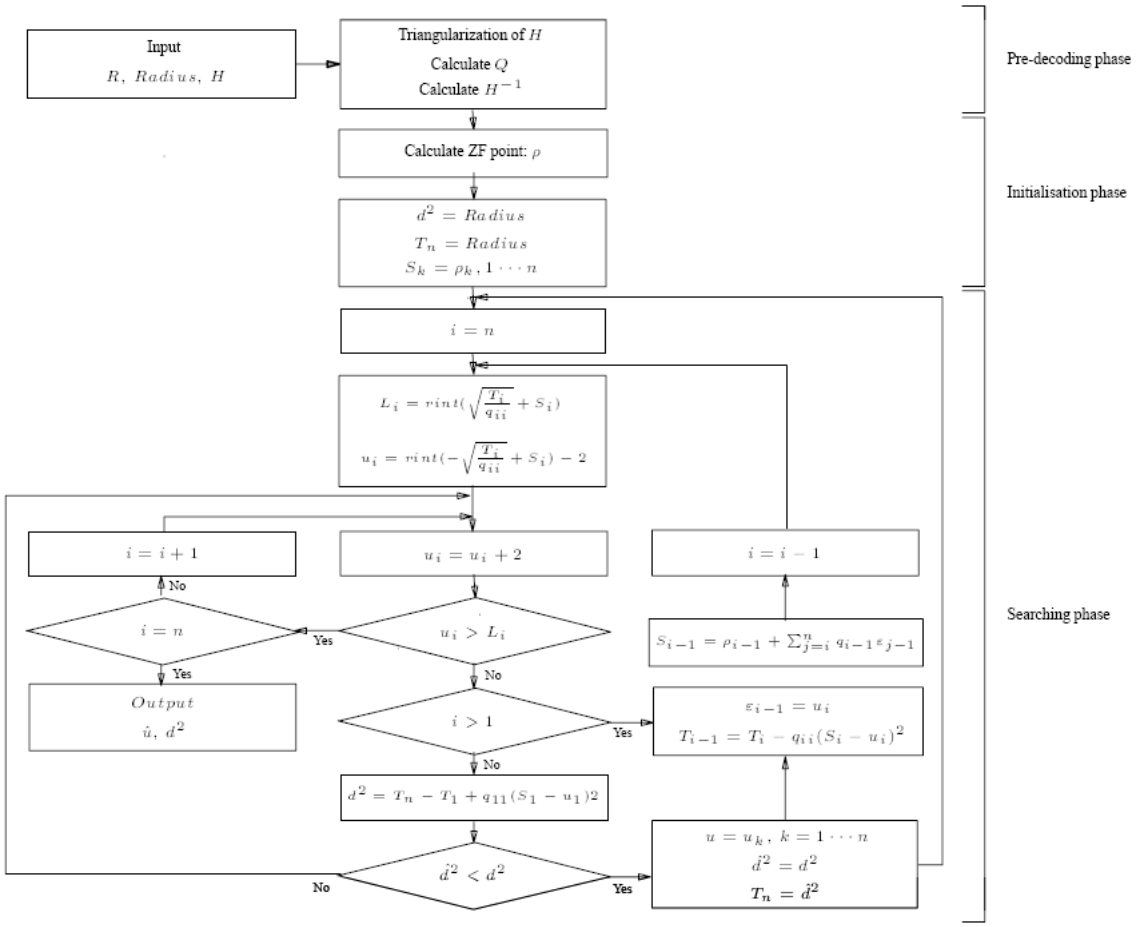
En posant $G = H \cdot H^\dagger + \left(\sigma_w^2 / E_s\right) \cdot I$, l'erreur quadratique devient :

$$\frac{\varepsilon^2}{E_s} = \left(F \cdot G - H^\dagger (G^\dagger)^{-1} \right) \cdot \left(F \cdot G - H^\dagger (G^\dagger)^{-1} \right)^\dagger + I - \left(G^{-1} \cdot H \right)^\dagger \left(G^{-1} \cdot H \right) \tag{5.62}$$

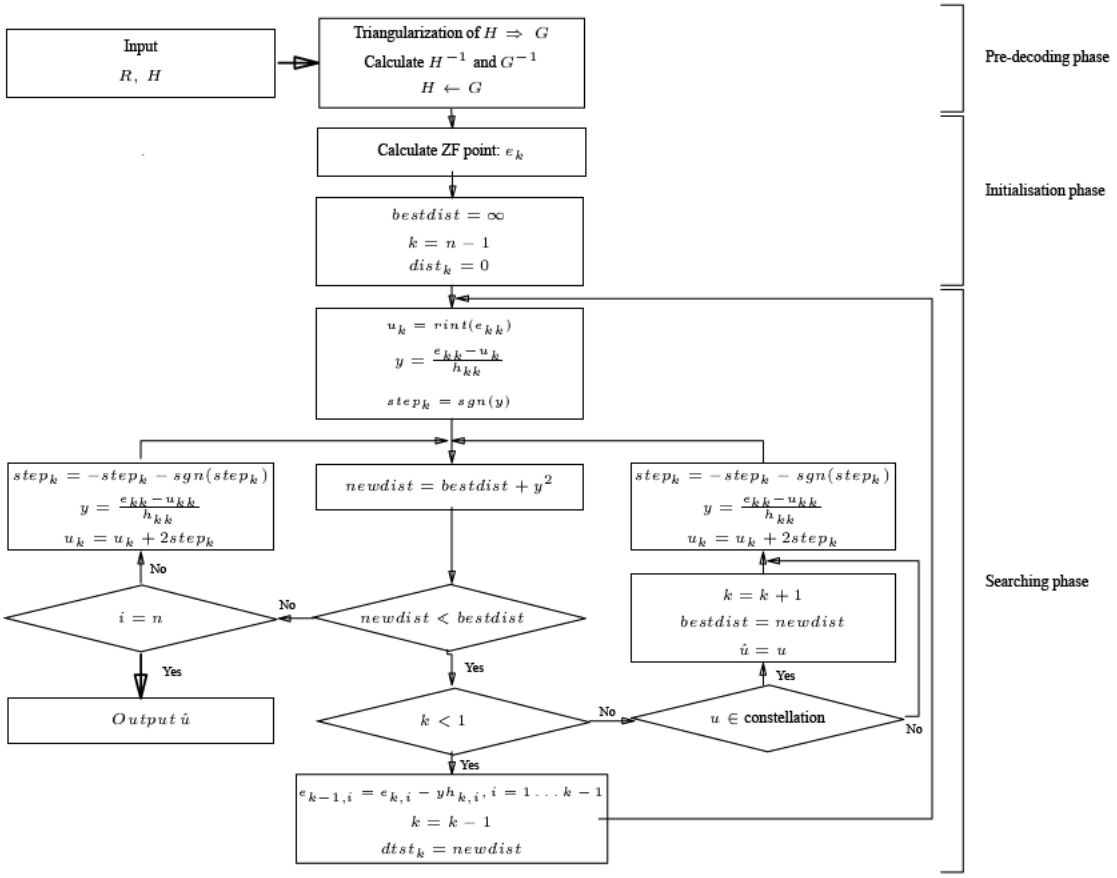
Etant donné une matrice de canal constante, la minimisation de l'erreur quadratique par rapport à F conduit à prendre

$$F = H^\dagger \cdot \left(H^\dagger H \cdot + \left(\sigma_w^2 / E_s\right) \cdot I \right)^{-1} \tag{5.63}$$

Annexe B : Organigramme du SD



Annexe C : Organigramme du SE



Annexe D : Algorithme LLL

Algorithm 6 Algorithme de réduction LLL

Entrée : $\boldsymbol{B} = (\boldsymbol{b_1}, ..., \boldsymbol{b_n})$

1. $\boldsymbol{b_1^\star} \leftarrow \boldsymbol{b_1}$, $\boldsymbol{B_1} \leftarrow \langle \boldsymbol{b_1^\star}, \boldsymbol{b_1^\star} \rangle$
2. Pour $i = 1, ..., n$
   2.1 $\boldsymbol{b_i^\star} \leftarrow \boldsymbol{b_i}$
   2.1 Pour $j = 1, ..., i-1$ $\mu_{i,j} = \langle \boldsymbol{b_1^\star}, \boldsymbol{b_1^\star} \rangle / \boldsymbol{B_j}$, $\boldsymbol{b_i^\star} \leftarrow \boldsymbol{b_i^\star} - \boldsymbol{\mu_{i,j} b_j^\star}$
   2.1 $\boldsymbol{B_i} \leftarrow \langle \boldsymbol{b_i^\star}, \boldsymbol{b_i^\star} \rangle$
3. $k = 2$
4. Exécuter procédure $RED(k, k-1)$
5. Si $\boldsymbol{b_k} < \left(\frac{3}{4} - \mu_{k,k-1}^2\right) \boldsymbol{b_{k-1}}$
   5.1 $\mu \leftarrow \mu_{k,k-1}$, $\boldsymbol{b} \leftarrow \boldsymbol{b_k} + \mu^2 \boldsymbol{b_{k-1}}$, $\mu_{k,k-1} = \mu \boldsymbol{b_{k-1}} / \boldsymbol{b}$, $\boldsymbol{b_k} \leftarrow \boldsymbol{b_{k-1} b_k} / \boldsymbol{b}$, $\boldsymbol{b_{k-1}} \leftarrow \boldsymbol{b}$
   5.2 Échanger $\boldsymbol{b_k}$ et $\boldsymbol{b_{k-1}}$
   5.3 Si $k > 2$
   Pour $j = 1, ..., k-2$ Échanger $\mu_{k,j}$ et $\mu_{k-1,j}$
   5.4 Pour $i = k+1, ..., n$
   $t \leftarrow \mu_{i,k}$, $\mu_{i,k} \leftarrow \mu_{i,k-1} - \mu t$ et $\mu_{i,k-1} \leftarrow t + \mu_{k,k-1} \mu_{i,k}$
   5.5 $k \leftarrow \max(2, k-1)$
   5.6 Aller à 4.
   Sinon
   Pour $l = k-2, k-3, \ldots, 1$ $RED(k, l)$
   $k \leftarrow k+1$
6. Si $k \leq n$ Aller à 4.
   Sinon retourner $(\boldsymbol{b_1}, \boldsymbol{b_2}, \ldots, \boldsymbol{b_n})$

Procédure $RED(k, l)$:
Si $|\mu_{k,l}| > \frac{1}{2}$
1. $r \leftarrow \lfloor 0.5 + \mu_{k,l} \rfloor$, $\boldsymbol{b_k} \leftarrow \boldsymbol{b_k} - r \boldsymbol{b_l}$
2. Pour $j = 1, ..., l-1$, $\mu_{k,j} \leftarrow \mu_{k,j} - r \mu_{l,j}$
3. $\mu_{k,l} \leftarrow \mu_{k,l} - r$

Annexe E : Algorithme Seysen

Algorithm 7 Algorithme de réduction Seysen

Entrée : $\boldsymbol{B} = (\boldsymbol{b_1}, ..., \boldsymbol{b_n})$

1. $\boldsymbol{B^\star = B^{-T}}$
2. Pour $i, j = 1, ..., n$, $a_{ij} = \langle \boldsymbol{b_i}, \boldsymbol{b_j} \rangle$, $a^\star_{ij} = \left\langle \boldsymbol{b^\star_i}, \boldsymbol{b^\star_j} \right\rangle$
3. Pour $i, j = 1, ..., n$, $\lambda_{ij} = \left\lceil \frac{1}{2} \left(\frac{a^\star_{ij}}{a^\star_{jj}} - \frac{a_{ij}}{a_{ii}} \right) \right\rfloor$
4. Tant que $(S_2 - Reduced(\lambda) \neq 1)$
   - 4.1 Pour $i = 1, ..., n$
     - Pour $j = 1, ..., n$
       - Si $i \neq j$ et $\lambda_{ij} \neq 0$ alors
         1. $\boldsymbol{T_{ij}^{\lambda_{ij}}} = T_{ij}(i, j)$
         2. $\boldsymbol{b_j} \leftarrow \boldsymbol{b_j} + \lambda_{ij} \boldsymbol{b_i}$, $\boldsymbol{b^\star_i} \leftarrow \boldsymbol{b^\star_i} - \lambda_{ij} \boldsymbol{b^\star_j}$
         3. $\boldsymbol{A} = \boldsymbol{T_{ji}^{\lambda_{ji}}}.\boldsymbol{A}.\boldsymbol{T_{ij}^{\lambda_{ij}}}$, $\boldsymbol{A^{-1}} = \left(\boldsymbol{T_{ij}^{\lambda_{ij}}}\right)^{-1}.\boldsymbol{A^{-1}}.\left(\boldsymbol{T_{ji}^{\lambda_{ji}}}\right)^{-1}$
         4. $\boldsymbol{T} = \prod \boldsymbol{T_{ij}^{\lambda_{ij}}}$
         5. $\lambda_{pq} = \left\lceil \frac{1}{2} \left(\frac{a^\star_{pq}}{a^\star_{qq}} - \frac{a_{pq}}{a_{pp}} \right) \right\rfloor$, pour $p, q = 1, ..., n$
       - Fin (Si)

   Fin (Tant que)

Procédure $S_2 - Reduced(\lambda)$: λ est la matrice ayant pour composantes λ_{ij}

- Pour $i = 1, ..., n$
  - Pour $j = 1, ..., n$
    - Si $i \neq j$ et $\lambda_{ij} \neq 0$ alors
      - $i = n, j = n$. Retourner 0
    - Sinon retourner 1
    - Fin (Si)

Procédure $T_{ij}(s, v)$:

1. $\boldsymbol{V^{sv}} = V_{ij}(s, v)$
2. $\boldsymbol{T_{ij}^{\lambda_{ij}}} = \boldsymbol{I_n} + \boldsymbol{V^{sv}}$

Procédure $V_{ij}(s, v)$:

1. Pour $i = 1, ..., n$
   - Pour $j = 1, ..., n$
     - Si $i = s$ et $j = v$ alors
       - $V_{ij}^{sv} = 1$
     - Sinon $V_{ij}^{sv} = 0$
     - Fin(Si)

Bibliographie

[1] J. G. Proakis, « Digital Communications », *MCGraw-Hill Series in Electrical and Computer Engineering*, 4th edition 1995.

[2] G. Forney, R. Gallager, G. Lang, F. Longstaff et S. Qureshi, « Efficient Modulation for Band-Limited Channels », *IEEE Journal on Selected Areas in Communications*, vol. 2, p. 632–647, September 1984.

[3] A. Naguib, V. Tarokh, N. Seshadri, et A. R. Calderbank, « A space-time coding modem for high-data-rate wireless communications », *IEEE Journal on Selected Areas in Communications*, vol. 16, p. 1459–1478, 4th edition 1998.

[4] V.Tarokh, N. Seshadri et A. R. Calderbank, « Space-time codes for rate wireless communication : performance criterion and code construction », *IEEE Transactions On Information Theory*, vol. 44, p. 744–765, March 1998.

[5] D. Tse et L. Zheng, « Diversity and multiplexing : a fundamental tradeoff in multiple antenna channels », *IEEE Transactions on Information Theory*, vol. 49, p. 1073–1096, May 2003.

[6] G. J. Foschini, « Layerd space-time architecture for wireless communication in a fading environment when using multiple antennas », *Bell Laboratories Technical Journal*, vol. 1, no. 2, p. 41–59, 1996.

[7] G. J. Foschini, G. D. Golden, R. Valenzuela et Wolniansky, « Simplified processing for high spectral efficiency wireless communication employing multi-element arrays », *IEEE Journal on Selected Areas on Communications*, vol. 17, p. 1841Ũ1852, 1999.

[8] P. W. Wolniansky, G. J. Foschini, G. D. Golden et R. A. Valenzuela, « V-BLAST : an architecture for realizing very high data rates over the rich-scattering wireless channel », Bell Labs, Lucent Technologies, Crawford Hill Laboratory 791 Holmdel-Keyport RD., Holmdel, NJ07733.

[9] O. Tirkkonen et A. Hottinen, « Square-matrix embeddable space-time block codes for complex signal constellations », *IEEE Transactions on Information Theory*, vol. 48, p. 384–395, February 2002.

[10] M. Damen, K. Abed-Meraim et J.-C. Belfiore, « Diagonal algebraic space-time block codes », *IEEE Transactions on Information Theory*, vol. 48, p. 628–636, March 2002.

[11] H. ElGamal et M. O. Damen, « Universal space-time coding », *IEEE Transactions On Information Theory*, May 2003.

[12] M. Damen, A. Tewfik et J.-C. Belfiore, « A construction of a space-time code based on number theory », *IEEE Transactions on Information Theory*, vol. 48, p. 753–760, March 2002.

[13] D. Aktas, H. E. Gamal et M. P. Fitz, « Towards optimal space-time coding », *in Proceedings of Asilomar Conference on Signals, Systems and Computers*, vol. 2, p. 1137–1141, November 2002.

[14] V.Tarokh, H. Jafarkhani et R. A. Calderbank, « Space-time block codes from orthogonal designs », *IEEE Transactions on Information Theory*, vol. 45, p. 1456–1467, July 1999.

[15] B. Hassibi et B. M. Hochwald, « High-rate codes that are linear in space and time », *IEEE Transactions on Information Theory*, vol. 48, p. 1804–1824, July 2002.

[16] S. Alamouti, « Space-time block coding : A simple transmitter diversity technique for wireless communications », *IEEE Journal On Select Areas In Communications*, vol. 16, p. 1451–1458, October 1998.

[17] O. Tirkkonen et A. Hottinen, « Complex space-time block codes for four Tx », *in Proceedings of IEEE Global Telecommunications Conference*, vol. 2, p. 1005–1009, November 2000.

[18] H. Jafarkhani, « A quasi-orthogonal space-time block code », *IEEE Transaction on Communication*, vol. 49, p. 1–4, 2000.

[19] M. Uysal et C. Georghiades, « Non-Orthogonal Space-Time Bmock Codes for 3-TX Antennas », *IEEE Electronics Letters*, vol. 38, p. 1689–1691, December 2002.

[20] N. Sharma et C. Papadias, « Full-rate full-diversity linear quasi-orthogonal space-time codes for any number of transmit antennas », *EURASIP Journal on Applied Signal Processing*, vol. 2004, p. 1246–1256, January 2004.

[21] H. ElGamal et M. O. Damen, « Threaded Algebraic Space-Time Codes », *ISWC, Victoria, Canada*, September 2002.

[22] F. Oggier, G. Rekaya, J.-C. Belfiore et E. Viterbo, « Perfect space-time block codes », *IEEE Transctions on Information Theory*, vol. 52, p. 3885–3902, September 2006.

[23] M.-A. Knus, A. Merkurje et J.-P. Tignol, « The book of involutions », *American Mathematical Society Colloquium Publications*, 1998.

[24] D. Morris, « Introduction to arithmetic groups », 2003.

[25] M. O. Damen, A. Chkeif et J.-C. Belfiore, « Lattice code decoder for space-time codes », *IEEE Communications Letters*, vol. 4, p. 161–163, May 2000.

[26] T. Cui et C. Tellambura, « Joint data detection and channel estimation for OFDM systems », *IEEE Transactions On Communications*, vol. 54, p. 670–679, April 2006.

[27] G. Rekaya et J.-C. Belfiore, « On the complexity of ML lattice decoders for decoding linear full-rate space-time codes », *in Proceedings in IEEE International Symposium on Information Theory*, p. 206, July 2003.

[28] A. K. Lenstra, H. Lenstra et L. Lovasz, « Factoring polynomials with rational coefficients », *Matematische Annalen*, vol. 261, p. 515–534, 1982.

[29] F. Kharrat-Kammoun, *Techniques adaptatives et classification des canaux à antennes multiples*. Thèse doctorat, Ecole Nationale Superieure des Telecommunications-Motorola Labs Saclay, 2006.

[30] M. Pohst, « On the computation of lattice vectors of minimal length, successive minima and reduced basis with applications », *ACMSIGSAMBulletin*, vol. 15, p. 37Ũ44, February 1981.

[31] R. Kannan, « Improved algorithms for integer programming and related lattice problems », *in Proceedings of the ACM Symposium on Theory of computing*, p. 193–206, April 1983.

[32] U. Fincke et M. Pohst, « Improved methods for calculating vectors of short length in a lattice, including a complexity analysis », *Mathematics of computation*, vol. 44, p. 463Ũ471, April 1985.

[33] J. Boutros et E. Viterbo, « A universal lattice code decoder for fading channels », *IEEE Transactions On Information Theory*, vol. 45, p. 1639–1642, July 1999.

[34] C. Schnorr et M. Euchner, « Lattice basis reduction : improved practical algorithms and solving subset sum problems », *Mathematical Programming*, vol. 66, p. 181–199, September 1994.

[35] B. Hassibi et H. Vikalo, « On the expected complexity of sphere decoding », *in Proceedings of Asilomar Conference on Signals, Systems, and Computers*, vol. 2, p. 1051–1055, 2001.

[36] E. Agrell, T. Eriksson, A. Vardy et K. Zeger, « Closest point search in lattices », *IEEE Transactions On Information Theory*, vol. 48, p. 2201–2214, August 2002.

[37] M. O. Damen, H. ElGamal et G. Caire, « On maximum likelihood detection and the search for the closest lattice point », *IEEE Transactions on Information Theory*, p. 2389–2402, October 2003.

[38] E. Zimmermann, W. Rave et G. Fettweis, « On the complexity of sphere decoding », *in Proceeding International Symposium on Wireless Pers. Multimedia Communications*, September 2004.

[39] R. Johannesson et K. S. Zigangirov, « Fundamentals of convolutional coding », *IEEE series on digital and mobile communication*, p. 269–315, 1999.

[40] R. M. Fano, « A heuristic discussion of probabilistic decoding », *IEEE Transactions On Information Theory*, vol. 9, p. 64–74, April 1963.

[41] K. S. Zigangirov, « Some sequential decoding procedures », *Probl. Peredach. Inform.*, vol. 2, p. 13–25, 1966.

[42] A. D. Murugan, H. ElGamal, M. O. Damen et G. Caire, « A unified framework for tree search decoding : rediscovering the sequential decoder », *IEEE Transactions On Information Theory*, vol. 52, p. 933–953, March 2006.

[43] J. Yue, K. Kim, J. Gibson et R. Iltis, « Channel Estimation and Data Detection for MIMO-OFDM Systems », *in Proc. GLOBECOM*, p. 581–585, 2003.

[44] W. Chin, « QRD Based Tree Search Data Detection for MIMO Communication Systems », *Vehicular Technology Conference, VTC 2005-Spring. 2005 IEEE 61st*, vol. 3, p. 1624–1627, June 2005.

[45] C. Berrou et A. Glavieux, « A near-optimum error correcting coding and decoding : Turbo codes », *IEEE Transactions on Communications*, vol. 44, p. 1261–1271, October 1996.

[46] R. G. Gallager, « Low density parity check codes », *IRE Trans.Inform. Theory*, vol. 8, p. 21–28, January 1962.

[47] P. Elias, « Coding for noisy channels », *IRE International Convention Record (Part IV)*, p. 37–46, 1955.

[48] J. Hagenauer, E. Offer et L. Papke, « Iterative decoding of binary block and convolutional codes », *IEEE Transactions on Information Theory*, vol. 42, p. 429–445, March 1996.

[49] J. Boutros, N. Gresset, L. Brunel et M. Fossorier, « Soft-input soft-output lattice sphere decoder for linear channels », *IEEE Global Telecommunications Conference GLOBECOM*, 2003.

[50] B. M. Hochwald et S. ten Brink, « Achieving near-capacity on a multiple-antenna channel », *IEEE Transactions on Communication*, vol. 53, p. 389–399, March 2003.

[51] G. J. Foschini et M. J. Gans, « On limits on wireless communication in a fading environment when using multiple antennas », *Wireless Personnal communications*, vol. 6, p. 311–335, March 1998.

[52] I. E. Telatar, « Capacity on multi-antennas Gaussian Channels », *IEEE Transactions on Information Theory*, vol. 45, p. 670–681, November-December 1999.

[53] T. Kiolding, « Performance aspects of WCDMA systems with high speed downlink packet access (HSDPA) », *IEEE Vehicular Thechnology conference*, 2001.

[54] R. Love, A. Glosh, W. Xiao et R. Ratasuk, « Performance of 3GPP high speed downlink access (HSDPA) », *IEEE Vehicular Thechnology conference*, 2004.

[55] A. Gorokhov, D. A. Gore et A. J. Paulraj, « Receive Antenna Selection for MIMO Spatial Multiplexing : Theory and Algorithms », *IEEE Transactions on signal processing*, vol. 51, p. 2796–2807, November 2003.

[56] R. W. Heath, S. Sandhu et A. Paulraj, « Antenna Selection for Spatial Multiplexing Systems with Linear Receivers », *IEEE Communication Letters*, vol. 5, p. 142–144, April 2001.

[57] R. W. Heath et A. Paulraj, « Antenna Selection for Spatial Multiplexing Systems Based on Minimum Error Rate », *IEEE International Conference on Communications 2001, Helsinki, Finland*, vol. 7, p. 2276–2280, June 2001.

[58] E. Hardouin, J.-M. Chaufray et T. Clessienne, « Environment-Adaptive Receivers : A performance Prediction Approach », *ICC '06, IEEE International Conference on Communications*, vol. vol. 12, pp.5709-5714, June 2006.

[59] E. Hardouin et J.-M. Chaufray, « A criterion for adaptive Rake/Equalizer configuration of mobile receivers », *Vehicular Technology Conference, 2006. VTC 2006-Spring. IEEE 63rd*, vol. 5, p. 2434– 2438, 2006.

[60] L. Zhou et M. Shimizu, « A Novel Condition Number-Based Antenna Shuffling Scheme for D-STTD OFDM System », *Vehicular Technology Conference,VTC2009-Spring. IEEE 69rd, Baecelona, Spain*, April 2009.

[61] « Low-Complexity Channel Adaptive MIMO Detection With Just-Acceptable Error Rate », *Vehicular Technology Conference,VTC2009-Spring. IEEE 69rd, Baecelona, Spain*, April 2009.

[62] E. Telatar, « Capacity of multi-antenna gaussian channels », *Internal Technical Report, Bell Laborateries*, 1995.

[63] H. ElGamal, G. Caire et M. Damen, « Lattice coding and decoding achieve the optimal diversity-vs-multiplexing tredeoff of MIMO », *submitted on IEEE Transactions On Information Theory*, November 2003.

[64] J.-C. Belfiore, G. Rekaya et E. Viterbo, « The Golden Code : a 2x2 full rate space-time code with non-vanishing determinants », *IEEE Transactions on Information Theory*, vol. 51, p. 1596–1600, November 2004.

[65] H. Yao et G. w.Wornell, « Lattice-Reduction-Aided Detectors for MIMO Communication Systems », *in Proceedings of IEEE Global Telecommunications Conference*, November 2002.

[66] M. Seysen, « Simultaneous reduction of a lattice basis and its reciprocal basis », *Combinatorica*, vol. 13, p. 363–376, 1993.

[67] B. LaMacchia, « Basis reduction algorithms and subset sum problems »,

[68] H. Minkowsk, « Geometrie der zahlen », *Teubner-Verlag*, 1896.

[69] A. Korkine et G. Zolotareff, « Sur les formes quadratiques positives ternaires », *Mathematische Annalen*, vol. 5, p. 581Ű583, 1872.

[70] A. Korkine et G. Zolotareff, « Sur les formes quadratiques », *Mathematische Annalen*, vol. 6, p. 336–389, 1873.

[71] C. P. Schnorr, « A hierarchy of polynomial time lattice basis reduction algorithms », *Theoretical Computer Science*, vol. 53, p. 201Ű224, 1987.

[72] M. Taherzadeh, A. Mobasher et A. K. Khadani, « LLL Lattice-Basis Reduction Achieves the Maximum Diversity in MIMO Systems »,

[73] M. Seysen, « Simultaneous reduction of a lattice basis and its reciprocal basis », *Combinatorica*, vol. 13, p. 363–376, 1993.

[74] A. Hottinen et O. Tirkkonen, « Precoder designs for high rate space-time block codes », *in Proc. Conference on Information Sciences and Systems, Princeton, NJ*, March 2004.

[75] E. Biglieri, Y. Hong et E. Viterbo, « On fast-decodable space-time block codes », *IEEE Transactions On Information Theory*, vol. 55, p. 524–530, February 2009.

[76] C. Hollanti, J. Lahtonen, K. Ranto, R. Vehkalahti et E. Viterbo, « On the algebraic structure of the silver code », *IEEE Information Theory Workshop, Porto, Portugal*, May 2008.

Publications

- **R. Ouertani**, A. Saadani, G. Rekaya Ben-Othman et J-C. Belfiore, "On the Golden Code Performance for MIMO-HSDPA System", *64th IEEE Vehicular Technology Conference, VTC-Fall*, Montreal, Septembre 2006.
- **R. Ouertani**, G. Rekaya Ben-Othman et A. Salah, "The Spherical Bound Stack decoder", *4th IEEE International Conference on Wireless and Mobile Computing, Wimob*, pp 322-327, Avignon, France, Octobre 2008.
- A. Salah, **R. Ouertani**, G. Rekaya Ben-Othman et S. Guillouard, "New Soft Stack Decoder for MIMO Channel", *Asilomar Conference on Signals, Systems and Computers*, California, USA, Octobre 2008.
- **R. Ouertani**, G. Rekaya Ben-Othman et J.-C. Belfiore, "An adaptive MIMO decoder", *69th IEEE Vehicular Technology Conference, VTC-Spring*, Barcelona, Spain, Avril 2009.
- **R. Ouertani** et G. Rekaya Ben-Othman, "A new Stack decoder with limited tree-search", *3rd International Conference on Signals, Circuits and Systems, SCS09*, Djerba, Tunisie, Novembre 2009.
- G. Rekaya Ben-Othman, **R. Ouertani** et J.-C. Belfiore. Brevet : "Décodeur adaptatif MIMO", Février 2008. Nr. FR08/50690.