SVM pour la démodulation et le codage conjoints

Gastón DE BONI ROVELLA[1,2] Meryem BENAMMAR[2] Delphine GOURMEL[3] Mehdi DJELLOUL[4]

[1]Laboratoire TéSA, 7 Bd de la Gare, 31500 Toulouse, France

[2]ISAE-SUPAERO, Université de Toulouse, 10 Av. Edouard Belin, 31400 Toulouse, France

[3]Direction Générale de l'Aviation Civile, 1 av du Dr Grynfogel, 31000, Toulouse, France

[4]École National d'Aviation Civile, 7 Av. Edouard Belin, 31400, Toulouse, France

Résumé – Les solutions basées sur l'apprentissage automatique pour la démodulation et le décodage de canal ont gagné en popularité ces dernières années en raison de leur capacité à traiter de gros volumes de données, de leur faible latence et de leur robustesse face aux erreurs de modèle. Dans ce contexte, des travaux antérieurs ont appliqué les Machines à Vecteurs de Support (SVM) au problème de décodage, produisant des résultats favorables pour les codes très courts. Dans ce travail, nous introduisons une approche nouvelle, bit-par-bit, qui réduit considérablement la complexité du décodeur basé sur les SVM, ainsi que la taille de l'ensemble de données d'entraînement. Enfin, nous analysons la complexité de notre système et montrons que les décodeurs basés sur les SVM restent des solutions hautement complexes qui ne sont pas encore applicables aux longueurs de codes plus grandes.

Abstract – Machine learning-based solutions for channel demodulation and decoding have gained popularity in recent years due to their high-throughput capabilities, low latency, and robustness to model errors. In this context, previous works have applied Support Vector Machines (SVM) to the decoding problem, producing favorable results for very short codes. In this work, we introduce a novel bitwise approach that considerably reduces the complexity of the SVM-based decoder, along with a reduction of the training dataset's size. Finally, we analyze the complexity of our system and show that SVM-based decoders remain highly complex solutions that are not yet applicable to larger codes.

1 Introduction

L'introduction des normes de technologie sans fil beyond-5G et 6G ces dernières années a conduit à un intérêt croissant pour les outils de communication permettant des vitesses "ultra-élevées" avec une latence de traitement "ultra-basse". Dans ce scénario, l'Intelligence Artificielle (IA) a rapidement gagné en notoriété en tant qu'outil puissant pour développer des décodeurs de canal rapides et efficaces, qui reposent principalement sur des données réelles et sont donc robustes aux erreurs dans le modèle mathématique.

Les premiers travaux sur les solutions d'IA pour le décodage ont été présentés il y a plus de trente ans [12, 5, 11], et leur intérêt a considérablement augmenté depuis. Les avancées récentes en informatique et en puissance de calcul ont conduit à de grandes avancées dans le domaine des réseaux neuronaux ainsi que d'autres solutions nécessitant des algorithmes d'optimisation lourds, tels que les Machines à Vecteurs de Support (SVM).

Les SVM ont été introduits dans les années 90 par Vapnik, Guyon et Boser [4, 6], et sont devenus populaires pour trois raisons principales : (i) ce sont des classificateurs à marge maximale ; (ii) l'algorithme d'optimisation à résoudre est convexe, et donc converge vers un minimum global ; et (iii) en raison de la nature du problème, il y a très peu de risque de surapprentissage aux données. Cela a suscité un grand intérêt dans la communauté du codage canal [2, 8, 10], mais leurs approches *one vs. all* et *one vs. rest* ne passent pas à l'échelle pour des codes longs, produisant au moins une fonction de décision par mot de code valide, et donc, en nombre exponentiel.

Dans ce travail, nous présentons une approche bit-à-bit qui réduit considérablement le nombre de SVM nécessaires pour le décodage, passant d'une croissance exponentielle par rapport à la longueur du message à une croissance linéaire, ce qui ne nécessite qu'une seule fonction de décision basée sur SVM par bit de message. De plus, nous montrons qu'un seul mot de code est nécessaire par classe, réduisant encore la complexité du processus d'optimisation et du système SVM résultant. Enfin, nous analysons sa complexité par rapport au décodeur de Maximum de Vraisemblance (ML), et montrons son équivalence lorsqu'il est appliqué au modèle de canal de bruit blanc additif Gaussien (AWGN).

Le travail est organisé comme suit : la section 2 décrit le système utilisé, énonce le problème et donne quelques notions de base sur les SVM. La section 4 présente notre solution et ses avantages par rapport aux travaux précédents. Dans la section 4.5, nous analysons la complexité de notre solution par rapport à l'algorithme ML, et la section 5 conclut le travail.

Notations : Dans la suite, les lettres romaines en majuscules et en gras (X et $\boldsymbol{X}$) désignent des variables et des vecteurs aléatoires, et les lettres minuscules (x et $\boldsymbol{x}$) leurs réalisations. $\mathbb{P}(\cdot)$ représente la probabilité d'événement, $\mathbb{I}(\cdot)$ l'opérateur d'indicateur booléen et i.i.d. signifie indépendants et identiquement distribués.

2 Énoncé du problème et préliminaires

2.1 Le problème de codage de canal

Considérons une transmission codée sur un modèle de canal à bruit blanc additif Gaussien (AWGN) en figure 1. Le modèle de système peut être décrit comme suit. Le message d'entrée $\boldsymbol{u} \in \{0,1\}^k$ est composé de k bits indépendants uniformément distribués, qui sont ensuite mappés via un code correcteur

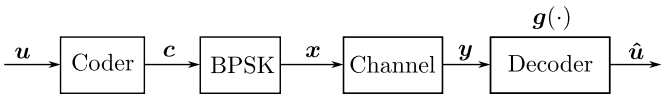

FIGURE 1 : Modèle de système général

d'erreurs linéaire en blocs de taille (n, k) en un mot de code $\boldsymbol{c} \in \{0,1\}^n$ de n bits. Le mot de code obtenu est transformé via une modulation BPSK [1] à l'entrée du canal $\boldsymbol{x} \triangleq 2\boldsymbol{c} - 1$ conduisant ainsi à $\boldsymbol{x} \in \{-1, 1\}^n$. La sortie du canal $\boldsymbol{y}$ résulte de la transmission de l'entrée du canal $\boldsymbol{x}$ à travers un canal AWGN sans mémoire $\boldsymbol{y} \triangleq \boldsymbol{x} + \boldsymbol{w}$ où $\boldsymbol{w} \overset{\text{i.i.d}}{\sim} \mathcal{N}(0, \sigma^2/2)$ est le bruit additif. Le signal reçu $\boldsymbol{y}$ est ensuite décodé en une estimation du message d'origine de k bits $\hat{\boldsymbol{u}} \in \{0,1\}^k$ à travers une règle de décodage $\boldsymbol{g}(\cdot)$ conduisant donc à $\hat{\boldsymbol{u}} \triangleq \boldsymbol{g}(\boldsymbol{y})$.

Pour un code linéaire de taille (n, k) fixé, le problème de codage de canal consiste à développer des règles de décodage minimisant la Probabilité d'Erreur de Bit (PEB) définie par

$$P_e^b \triangleq \frac{1}{k} \sum_{j=1}^{k} \mathbb{P}(\hat{U}_j \neq U_j). \tag{1}$$

2.2 La règle de décodage Bit-MAP

La règle de décodage théoriquement optimale $\boldsymbol{g}(\cdot)$ pour la PEB est la règle du bit-Maximum A Posteriori (Bit-MAP), que nous présentons dans le lemme suivant.

Lemme 1 (Règle de décodage Bit-MAP)**.**
La règle de décodage optimale pour la BEP définie dans (1) *est donnée par la concaténation de k règles de décodage bit-MAP $\{g_j^\star(\cdot)\}_{1 \leq j \leq k}$ où, pour tout $j \in \{1, ..., k\}$:*

$$\hat{u}_j^\star(\boldsymbol{y}) = g_j^\star(\boldsymbol{y}) \triangleq \underset{u \in \{0,1\}}{\textit{argmax}}\, \mathbb{P}(U_j = u | \boldsymbol{Y} = \boldsymbol{y}), \tag{2}$$

Démonstration. La preuve découle d'outils classiques de la théorie de la détection et est omise par souci de concision. □

La règle de décodage Bit-MAP peut être simplifiée sous l'hypothèse d'entrées binaires U_j i.i.d et uniformes.

Lemme 2 (Simplification de Bit-MAP)**.**
Sous l'hypothèse d'entrées binaires U_j i.i.d et uniformes, la règle de décodage Bit-MAP peut être simplifiée comme suit :

$$g_j^\star(\boldsymbol{y}) = \mathbb{I}\Big\{ \sum_{\substack{\boldsymbol{u} \\ u_j = 1}} P_{\boldsymbol{Y}|\boldsymbol{U}}(\boldsymbol{y}|\boldsymbol{u}) > \sum_{\substack{\boldsymbol{u} \\ u_j = 0}} P_{\boldsymbol{Y}|\boldsymbol{U}}(\boldsymbol{y}|\boldsymbol{u}) \Big\}. \tag{3}$$

où $P_{\boldsymbol{Y}|\boldsymbol{U}}(\boldsymbol{y}|\boldsymbol{u})$ est une densité de probabilité conditionnelle.

Démonstration. La preuve est reportée à l'annexe. □

Anisi que formulées en (3), les sommes impliquent une marginalisation sur des séquences binaires de longueur $k - 1$, rendant ainsi la complexité du décodeur bit-MAP exponentielle en k. Par conséquent, il ne peut être implémenté pour les codes génériques pour lesquels la marginalisation ne peut pas être effectuée sur des graphes ou des treillis.

Dans ce travail, nous étudierons la construction de décodeurs basés sur des SVM qui peuvent fournir des algorithmes pratiques pour le problème de codage de canal à l'avance, tout en demeurant optimaux du point de vue de la PEB.

[1] Les résultats peuvent être étendus de manière triviale à d'autres modulations telles que la modulation de phase PSK, ou la modulation en quadrature d'amplitude QAM. Pour cela, il suffit de concaténer les parties réelles et imaginaires du signal reçu avant de les injecter dans le décodeur.

3 SVM pour la classification

3.1 Données linéairement séparables

Considérons un jeu de données étiquetées $\{\boldsymbol{y}_i, l_i\}_{1 \leq i \leq N}$ consistant en N vecteurs $\boldsymbol{y}_i \in \mathbb{R}^n$ et leurs étiquettes $l_i \in \{-1, +1\}$. Les vecteurs $\boldsymbol{y}_i$ peuvent être classifiés en deux classes C_0 (resp. C_1) correspondant à une étiquette $l_i = -1$ (resp. $l_i = +1$). Ces données sont dites linéairement séparables s'il existe $\boldsymbol{w} \in \mathbb{R}^n$ et $b \in \mathbb{R}$ et un hyperplan $P \in \mathbb{R}^n$

$$P : \{\boldsymbol{y} \in \mathbb{R}^n \mid f(\boldsymbol{y}) = 0\} \tag{4}$$

où $f(\boldsymbol{y}) = \boldsymbol{w}^T\boldsymbol{y} + b$, tel que $f(\boldsymbol{y}) < 0$ pour tout $\boldsymbol{y} \in C_0$ et $f(\boldsymbol{y}) > 0$ pour tout $\boldsymbol{y} \in C_1$. Le principe des SVM est de construire un hyperplan P qui vérifie *la propriété de marge maximale*, c.à.d, qui soit à une distance égale et maximale des points les plus proches de chaque classe (dits *vecteurs de support*). Cet hyperplan est donné en combinant la solution $\boldsymbol{w}$ du problème d'optimisation suivant,

$$\underset{\boldsymbol{w}}{\text{argmax}} \frac{1}{||\boldsymbol{w}||} \tag{5}$$

$$\text{t.q. } \min_{i=1,..,N} |\boldsymbol{w}^T\boldsymbol{y}_i + b| = 1. \tag{6}$$

avec (6) pour trouver b. Finalement, $\boldsymbol{y} \in C_1$ si $f(\boldsymbol{y}) > 0$, et $\boldsymbol{y} \in C_0$ sinon. Voir [1] pour la preuve complète.

3.2 Données linéairement non-séparables

Sous leur forme standard, les SVM sont des classificateurs binaires pour les données linéairement séparables. Cependant, les systèmes de décodage de canal doivent recourir à des techniques plus avancées qui permettent une classification multi-classes de données non linéairement séparables. Pour la classification non-linéaire, des *méthodes de noyau* peuvent être utilisées, où les points de données sont projetés dans un espace de grande dimension via une fonction non-linéaire $\Phi(\cdot)$ dans lequel ils deviennent linéairement séparables. Étant donnée cette fonction de projection Φ, sa fonction de noyau associée est donnée par $K(\boldsymbol{y}, \boldsymbol{y}') = \langle \Phi(\boldsymbol{y}), \Phi(\boldsymbol{y}') \rangle$, où $\langle , \rangle$ désigne le produit interne dans l'espace défini par $\{\Phi(\boldsymbol{y}) : \boldsymbol{y} \in \mathbb{R}^n\}$. En considérant la fonction Φ et un ensemble de données $\{\boldsymbol{y}_1, ..., \boldsymbol{y}_N\}$ avec leurs étiquettes correspondantes $\{l_1, ..., l_N\}$, le problème d'optimisation à résoudre est le suivant :

$$\begin{aligned} &\underset{\alpha_i, i=1,..,N}{\text{argmax}} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j l_i l_j K(\boldsymbol{y}_i, \boldsymbol{y}_j) \\ &\text{t.q. } 0 \leqslant \alpha_i \leqslant C \text{ and } \textstyle\sum_{i=1}^{N} l_i \alpha_i = 0, \end{aligned} \tag{7}$$

où $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_N)$ est la variable d'optimisation et C est un hyper-paramètre de relaxation à choisir. Les lecteurs intéressés par la déduction complète sont renvoyés à [1]. La fonction de décision est alors donnée par :

$$f(\boldsymbol{y}) = \sum_{i=1}^{N} l_i \alpha_i K(\boldsymbol{y}_i, \boldsymbol{y}) + b, \tag{8}$$

où $\boldsymbol{y}$ appartient à la classe 1 si $f(\boldsymbol{y}) > 0$ et à la classe 0 sinon, et les vecteurs de support sont déterminés par les indices i pour lesquels $\alpha_i > 0$. Dans ce travail, en tant que noyau, nous utiliserons la fonction de base radiale (RBF) bien connue :

$$K_{RBF}(\boldsymbol{y}, \boldsymbol{y}') = e^{-\gamma ||\boldsymbol{y} - \boldsymbol{y}'||^2}. \tag{9}$$

4 SVM pour le décodage

4.1 Travaux précédents et SVM multi-classes

Indépendamment de la séparabilité linéaire ou non du jeu de données, les SVM sont par nature des classificateurs binaires. Afin de décoder un code linéaire en blocs (n, k), il est nécessaire de recourir à une classification multi-classes avec une classe par mot de code valide, c'est-à-dire 2^k classes. Les solutions précédentes [2, 8, 10] pour la démodulation et le décodage conjoints basés sur les SVM ont employé les approches dites *one vs. all* et *one vs. one* [7].

La méthode *one vs. all* est basée sur la production de 2^k SVM binaires, chacun isolant une classe contre toutes les autres. Tous les SVM sont ensuite appliqués au signal reçu $\boldsymbol{y}$, et la classe sélectionnée sera celle qui (i) appartient à la classe correspondante et (ii) présente la plus grande valeur $f_j(\boldsymbol{y})$ pour $j = \{1, .., 2^k\}$, c'est-à-dire la plus grande distance à l'hyperplan séparateur. Si aucun classificateur ne donne de résultat positif, la classe la plus proche est sélectionnée –celle dont la valeur négative est la plus proche de 0. Pour une lecture approfondie de cette méthode et d'autres, y compris l'approche *one vs. rest*, se reporter à [7].

4.2 Solution proposée : SVM bit-à-bit

Les approches précédentes ont pour principal inconvénient une complexité exponentielle croissante, avec au moins 2^k SVM pour un code de taille (n, k). Pour y remédier, nous proposons une nouvelle approche dite bit-à-bit qui utilise seulement k classificateurs pour un code de taille (n, k). Cette méthode transforme également le problème multi-classes en une série de classifications binaires, mais au lieu de générer un SVM par mot de code valide, nous produirons un SVM par bit de message. À cet effet, nous diviserons l'ensemble de données en deux sous-ensembles distincts : (i) l'ensemble $\mathcal{S}_j^1$ des vecteurs $\boldsymbol{y} \in \{\boldsymbol{y}_1, ..., \boldsymbol{y}_N\}$ pour lesquels leur message correspondant $\boldsymbol{u}$ est tel que sa j-ème composante est égale à 1, (ii) idem, avec $u_j = 0$, représenté par $\mathcal{S}_j^0$. Par conséquent, chaque classificateur $f_j(\boldsymbol{y})\ \forall j = \{1, ..., k\}$ décidera si le j-ème bit du message estimé $\hat{\boldsymbol{u}}$ est un 0 ou un 1. Comme expliqué précédemment, cela réduit le nombre de SVM nécessaires à k, et permet une implémentation en parallèle afin de réduire la latence.

4.3 Optimisation à mot unique par classe

Introduisons un dernier élément dans notre système qui nous permettra de réduire davantage la complexité du processus d'optimisation. Rappelons que la caractéristique principale du classificateur binaire SVM est la propriété de marge maximale, qui sépare les points les plus proches entre les classes –appelés *vecteurs de support*– par un hyperplan de décision équidistant.

Pour un schéma de décodage sur un canal AWGN, cela équivaut à un classificateur à marge maximale entre *seulement* les mots de code originaux sans bruit. Par conséquent, il suffit d'entraîner les SVM sur des données non-bruitées *à mot de code unique*, contrairement à d'autres solutions basées sur l'IA qui nécessitent plusieurs points par classe. En prenant cela en considération, pour un code de taille (n, k), l'ensemble de données sera composé des 2^k mots de code valides $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_{2^k}\}$ avec n éléments chacun, et k classificateurs binaires seront produits en suivant l'approche bit-à-bit de la section 4.2. Pour un vecteur non classifié $\boldsymbol{y}$, chaque classificateur f_j déterminera la valeur du j-ème bit du message estimé $\hat{\boldsymbol{u}}$.

TABLE 1 : Nombre de classificateurs par méthode

	Bit-à-bit	One vs rest	One vs one
# de classificateurs	k	2^k	$2^k(k-1)$
# de termes dans (8)	2^k	$N \geq 2^k$	$\approx \frac{N}{2^{k-1}} \geq 2$

4.4 Simulations numériques

Nous avons implémenté notre solution pour un code polaire $(16, 8)$ [3] avec un modèle de canal AWGN, et calculé le taux d'erreur binaire (TEB) pour estimer le PEB par simulation de Monte Carlo avec un critère d'arrêt de 500 erreurs de trame. Les résultats sont présentés dans la Figure 2, où nous pouvons voir que notre solution présente les mêmes performances que le décodeur Bit-MAP. Dans la section suivante, nous prouverons que ce n'est pas une coïncidence et que le décodeur basé sur SVM est en réalité équivalent à la règle de décision Bit-MAP.

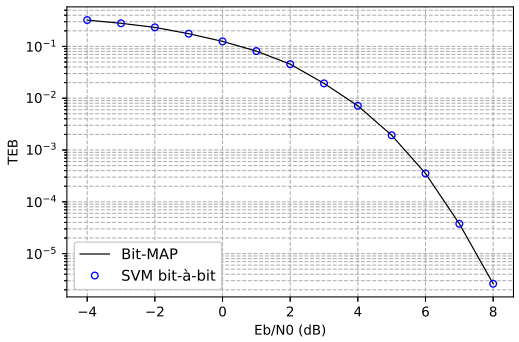

FIGURE 2 : Estimation du PEB pour le décodeur SVM pour une code polaire $(16, 8)$ sur canal AWGN, avec $\gamma = 0.25$.

4.5 Étude de complexité

Le Tableau 1 résume la complexité de décodage de notre méthode et des méthodes les plus courantes dans la littérature. Comme nous pouvons l'observer, l'approche bit-à-bit est la première à permettre un nombre linéairement croissant de SVM, ce qui permet un passage à l'échelle contrairement aux méthodes à croissance exponentielle. Il en va de même pour l'ensemble de données : pour que le SVM apprenne une règle de décision entre deux classes, il doit "voir" au moins un élément de chaque classe. Avec notre optimisation d'un seul mot de code par classe, nous avons réduit la taille de l'ensemble de données à son minimum, $N = 2^k$.

Cependant, la complexité ne dépend pas seulement du nombre de fonctions à évaluer, mais aussi du nombre d'opérations requises pour effectuer chacune de ces classifications. Même avec notre méthode à complexité réduite, la taille de l'ensemble de données est de 2^k, avec un élément par mot de code valide. Cela implique une croissance exponentielle, car la taille de l'ensemble de données détermine le nombre de termes dans (8). Pour cette raison, il peut être intéressant de comparer notre méthode avec le décodeur bit-MAP optimal.

Théorème 1 (Equivalence Bit-MAP et SVM bit-à-bit)**.** *Considérons le cadre de la section 2.1, le décodeur SVM bit à bit de la section 4.2 et le problème d'optimisation dans l'équation* (7) *avec une fonction de noyau RBF. Alors, pour* $\gamma = 1/(2\sigma^2)$*, la solution* $\boldsymbol{\alpha} = (1, 1, ..., 1)$ *et* $b = 0$ *est optimale et coïncide avec l'algorithme de décodage Bit-MAP.*

Preuve. Commençons par remplacer la solution proposée $\{\boldsymbol{\alpha} = (1, 1, ..., 1),\ b = 0\}$, la fonction de noyau RBF et l'ensemble de données réduit $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_{2^k}\}$ dans la fonction de décision (8) :

$$f_j(\boldsymbol{y}) = \sum_{i=1}^{2^k} l_i e^{-\gamma||\boldsymbol{x}_i - \boldsymbol{y}||^2}. \tag{10}$$

Nous pouvons maintenant réarranger la somme en séparant les termes avec $l_i = +1$ de ceux avec $l_i = -1$, correspondant respectivement à $\boldsymbol{x}_i \in \mathcal{S}_j^1$ et $\boldsymbol{x}_i \in \mathcal{S}_j^0$:

$$f_j(\boldsymbol{y}) = \sum_{\boldsymbol{x}_i \in \mathcal{S}_j^1} e^{-\gamma||\boldsymbol{x}i - \boldsymbol{y}||^2} - \sum_{\boldsymbol{x}_i \in \mathcal{S}_j^0} e^{-\gamma||\boldsymbol{x}_i - \boldsymbol{y}||^2}. \tag{11}$$

À partir de (11), il est facile de déduire l'expression de la règle de décision pour le j-ème bit :

$$\hat{u}_j(\boldsymbol{y}) = \mathbb{I}\left\{ \sum_{\boldsymbol{x}_i \in \mathcal{S}_j^1} e^{-\gamma||\boldsymbol{x}_i - \boldsymbol{y}||^2} > \sum_{\boldsymbol{x}_i \in \mathcal{S}_j^0} e^{-\gamma||\boldsymbol{x}_i - \boldsymbol{y}||^2} \right\} \tag{12}$$

En sélectionnant $\gamma = 1/(2\sigma^2)$, l'équation (12) correspond à la règle Bit-MAP dans l'équation (3) pour un canal AWGN avec une puissance de bruit $\sigma^2/2$ en remarquant que $P_{\boldsymbol{Y}|\boldsymbol{U}}(\boldsymbol{y}|\boldsymbol{u}) = P_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{y}|\boldsymbol{x}(\boldsymbol{u})) = e^{-\gamma||\boldsymbol{x}(\boldsymbol{u}) - \boldsymbol{y}||^2}$. □

5 Conclusion

Dans cette étude, nous avons présenté une méthode pour réduire la complexité des décodeurs basés sur les SVM. Nous sommes passés d'une approche *one vs. rest* qui repose sur 2^k étapes de décodage à une approche *bit-à-bit* qui ne nécessite que k étapes de décodage. De plus, nous avons présenté une méthode pour réduire la taille de l'ensemble d'entraînement à un seul élément par mot de code valide, ce qui vise également à réduire la complexité et à rendre les décodeurs hautement évolutifs.

Cependant, nous avons montré que même cette approche simplifiée reste exponentiellement complexe en raison de la nature du processus d'optimisation des SVM, qui nécessite au moins un élément par classe. De plus, dans le modèle de canal AWGN, nous avons prouvé que notre décodeur est équivalent à un algorithme bit-MAP, ce qui le rend inadapté aux longueurs de code importantes.

Annexe : preuve du lemme 2

Soit $\boldsymbol{y}$ une séquence reçue en entrée du décodeur bit-MAP $g^\star(\cdot)$. Supposons en outre que $\boldsymbol{y}$ observé a une fonction de densité de probabilité marginale non nulle $P_{\boldsymbol{Y}}(\boldsymbol{y}) \neq 0$. Alors, pour $j \in 1, ..., k$, nous avons :

$$\begin{aligned}
g_j^\star(\boldsymbol{y}) &\triangleq \underset{u \in \{0,1\}}{\operatorname{argmax}}\ \mathbb{P}(U_j = u | \boldsymbol{Y} = \boldsymbol{y}) \\
&\overset{(a)}{=} \mathbb{I}\Big\{ \mathbb{P}(U_j = 1 | \boldsymbol{Y} = \boldsymbol{y}) > \mathbb{P}(U_j = 0 | \boldsymbol{Y} = \boldsymbol{y}) \Big\} \\
&\overset{(b)}{=} \mathbb{I}\Big\{ \sum_{\substack{\boldsymbol{u} \\ u_j = 1}} \mathbb{P}(\boldsymbol{U} = \boldsymbol{u} | \boldsymbol{Y} = \boldsymbol{y}) > \sum_{\substack{\boldsymbol{u} \\ u_j = 0}} \mathbb{P}(\boldsymbol{U} = \boldsymbol{u} | \boldsymbol{Y} = \boldsymbol{y}) \Big\} \\
&\overset{(c)}{=} \mathbb{I}\Big\{ \sum_{\substack{\boldsymbol{u} \\ u_j = 1}} \mathbb{P}(\boldsymbol{U} = \boldsymbol{u}) P_{\boldsymbol{Y}|\boldsymbol{U}}(\boldsymbol{y}|\boldsymbol{u}) > \sum_{\substack{\boldsymbol{u} \\ u_j = 0}} \mathbb{P}(\boldsymbol{U} = \boldsymbol{u}) P_{\boldsymbol{Y}|\boldsymbol{U}}(\boldsymbol{y}|\boldsymbol{u}) \Big\} \\
&\overset{(d)}{=} \mathbb{I}\Big\{ \sum_{\substack{\boldsymbol{u} \\ u_j = 1}} P_{\boldsymbol{Y}|\boldsymbol{U}}(\boldsymbol{y}|\boldsymbol{u}) > \sum_{\substack{\boldsymbol{u} \\ u_j = 0}} P_{\boldsymbol{Y}|\boldsymbol{U}}(\boldsymbol{y}|\boldsymbol{u}) \Big\},
\end{aligned}$$

où (a) découle de la formulation booléenne d'un problème de classification binaire, (b) est dû à la propriété de marginalisation de probabilité, (c) découle de l'identité de Bayes ainsi que de l'hypothèse que $\mathbb{P}(\boldsymbol{Y} = \boldsymbol{y}) \neq 0$, tandis que (d) provient de l'hypothèse de bits d'entrée uniformément distribués et i.i.d., c'est-à-dire $\mathbb{P}(\boldsymbol{U} = \boldsymbol{u}) = 2^{-k} > 0$ pour tout $\boldsymbol{u} \in \{0, 1\}^k$. □

Références

[1] Y. S. ABU-MOSTAFA, M. MAGDON-ISMAIL et H.T. LIN : *Learning from data*, volume 4. AMLBook New York, 2012.

[2] S. AKIN, M. PENNER et J. PEISSIG : Joint Channel Estimation and Data Decoding using SVM-based Receivers, 2020.

[3] E. ARIKAN : Channel Polarization : A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels. *IEEE Transactions on Information Theory*, 55(7):3051–3073, jul 2009.

[4] B. E. BOSER, I. M. GUYON et V. N. VAPNIK : A Training Algorithm for Optimal Margin Classifiers. *In Proceedings of the fifth annual workshop on Computational learning theory*. ACM, jul 1992.

[5] J. BRUCK et M. BLAUM : Neural Networks, Error-Correcting Codes, and Polynomials over the Binary n-Cube. *IEEE Transactions on Information Theory*, 35(5):976–987, 1989.

[6] C. CORTES et V. VAPNIK : Support-Vector Vetworks. *Machine Learning*, 20(3):273–297, sep 1995.

[7] C.W. HSU et C. LIN : A Comparison of Methods for Multiclass Support Vector Machines. *IEEE Transactions on Neural Networks*, 13(2):415–425, mar 2002.

[8] J.W.H. KAO et S. M. BERBER : Error Control Coding Based on Support Vector Machine. 2008.

[9] T. O'SHEA et J. HOYDIS : An Introduction to Deep Learning for the Physical Layer. *IEEE Transactions on Cognitive Communications and Networking*, 3(4):563–575, dec 2017.

[10] V. SUDHARSAN et B. YAMUNA : Support Vector Machine Based Decoding Algorithm for BCH Codes. *Journal of telecommunications and information technology*, 2016, 2016.

[11] J. YUAN, V.K. BHARGAVA et Q. WANG : An Error Correcting Neural Network. *In Conference Proceeding IEEE Pacific Rim Conference on Communications, Computers and Signal Processing*, 1989.

[12] G. ZENG, D. HUSH et N. AHMED : An Application of Neural Net in Decoding Error-Correcting Codes. *In IEEE International Symposium on Circuits and Systems*, 1989.