Réseaux de Neurones pour le Traitement d'Antenne et la Commande Référencée Capteur

Nadine Rondel

numéro d'ordre: 419 année 1996

Thèse de Doctorat

Présentée à
l'Université de Bretagne Occidentale
par

Nadine RONDEL

Réseaux de Neurones pour le Traitement d'Antenne et la Commande Référencée Capteur

Soutenue le 15 mai 1996, devant le jury ainsi composé:

Président:	*C. Jutten*	*Professeur à l'Université de Grenoble*
Rapporteurs:	*Ph. Réfrégier*	*Professeur à l'ENSPM, Marseille*
	G. Faucon	*Professeur à l'Université de Rennes*
Examinateurs:	*G. Burel*	*Professeur Associé à l'Université de Brest*
	L.C. Calvez	*Professeur à l'Université de Brest*
	J.Y. Catros	*Responsable Etudes Amont à Thomson-TSI, Toulouse*
	P. Vilbé	*Professeur à l'Université de Brest*

Résumé

Les travaux présentés portent sur l'application de techniques neuronales à la résolution de problèmes issus des domaines couverts par le traitement d'antenne et la commande référencée capteur.

Après avoir présenté différentes structures neuronales, nous étudions le formalisme mathématique généralement utilisé pour l'estimation d'angles d'arrivée. Nous proposons une méthode neuronale précise, basée sur la mise en œuvre de contraintes au niveau des poids d'un perceptron multicouche. Lorsque, en plus de l'information géométrique relative à la forme de l'antenne, nous disposons d'informations statistiques relatives aux sources, il est possible d'adjoindre une nouvelle contrainte à ce réseau. Un tel système permet alors d'estimer les angles d'arrivée d'autant de sources qu'il y a de capteurs sur l'antenne (alors que dans une telle configuration, les méthodes classiques sont inefficaces). Plusieurs applications concrètes sont exposées: estimation de l'inclinaison des lignes d'un texte, réseaux monofréquence, séparation et localisation de voix humaines dans des conditions réelles.

Une extension à la localisation d'objets est l'asservissement d'un robot sur l'objet via le capteur: la commande référencée capteur est étudiée, et de nouvelles approches neuronales sont proposées, ainsi que des améliorations des méthodes déjà existantes.

Remerciements

Je voudrais tout d'abord remercier les membres du jury. Monsieur Jutten m'a fait l'honneur d'en assurer la présidence et je lui en suis particulièrement reconnaissante. J'adresse mes plus vifs remerciements à Messieurs Faucon et Réfrégier, qui ont évalué ce travail de manière détaillée en qualité de rapporteurs, et m'ont fait part de leurs remarques constructives. Enfin, j'exprime ma gratitude à Messieurs Calvez et Vilbé, pour leur participation au jury en tant qu'examinateurs.

Je voudrais remercier particulièrement Monsieur Burel qui a assuré l'encadrement de ce travail et m'a fait profiter de son expérience, ainsi que Monsieur Catros pour diverses discussions enrichissantes. Je leur renouvelle également mes remerciements pour avoir su, dans un contexte extrêmement difficile de restructuration industrielle, préserver les études en cours et permettre l'aboutissement de ce travail.

Je suis reconnaissante à l'ICA Demay et à l'IPA Reichart de la DGA, qui ont contribué à rendre possible la réalisation de cette thèse, et m'ont ainsi permis de concrétiser ce projet qui me tenait à cœur.

Je termine en remerciant mes collègues thèsards de Thomson, notamment Hugues Hénocq et Patrick Leprince, qui ont contribué à rendre l'ambiance de travail agréable.

Introduction

Le cerveau humain et ses vastes capacités d'apprentissage ont toujours fasciné les hommes, et en particulier les neurobiologistes. A la fin des années 40, les travaux de Hebb [1], conduisent à un modèle pour l'apprentissage. Et dès les années 60, c'est un système neuronal tout entier, le Perceptron de Rosenblatt [5], qui voit le jour. A l'époque Rosenblatt ne considérait pas vraiment le perceptron comme un modèle neuronal, mais plutôt comme une "machine à apprendre". Destiné à la reconnaissance de caratères, le perceptron est rapidement abandonné lorsque d'autres chercheurs [4], [3] montrent les lacunes de ce modèle.

Les recherches sur l'apprentissage sont donc un peu laissées de côté jusqu'en 1982, où la parution des travaux de Hopfield [2] relance l'intérêt de la communauté scientifique: en proposant un nouveau type de réseau de neurones, à savoir les réseaux entièrement connectés, Hopfield ouvre de nouveaux horizons à la fois aux neurobiologistes et aux informaticiens.

Quelques années plus tard, la parution de l'ouvrage collectif "Parallel Distributed Processing" [6] annonce la naissance d'un véritable domaine de recherche: la science des réseaux de neurones formels, ou *neuromimétique*. Depuis, nombre de conférences et congrès internationaux y sont entièrement consacrés. Cet ouvrage reprend les idées de Hopfield en y ajoutant des améliorations et diverses variantes (par exemple la Machine de Boltzmann), et surtout, il modifie le Perceptron à deux couches de Rosenblatt en lui ajoutant une ou plusieurs couches dites "cachées". Désormais appelé le Perceptron MultiCouche, ce nouveau modèle peut traiter des problèmes non linéaires, ce que ne pouvait faire le Perceptron simple de Rosenblatt. De plus, "Parallel Distributed Processing" propose un algorithme très performant de rétropropagation de l'erreur en sortie du réseau: avec quelques améliorations apportées par l'expérience, c'est presque toujours ce même algorithme qui est encore utilisé aujourd'hui pour l'apprentissage des Perceptrons MultiCouches.

Depuis la sortie du livre de Rumelhart et Mc Clelland [6], nombre de scientifiques travaillant à l'origine dans des domaines divers, souvent très éloignés de la neurobiologie, se sont intéressés à la neuromimétique. Que ce soit dans le domaine du traitement d'image, du traitement du signal, de la reconnaissance de caractères, ou même de la robotique, de nombreux problèmes jusqu'à présent difficilement traitables, ont pu être résolus grâce aux réseaux de neurones formels, et en particulier aux Perceptrons MultiCouches.

Nous nous intéressons principalement à deux domaines: le traitement d'antenne pris dans son sens le plus large, et la commande référencée capteur, qui est à la frontière antre la robotique et le traitement d'image. Par traitement d'antenne, nous sous-entendons un domaine qui intéresse la communauté du traitement du signal depuis plus de trente ans: il s'agit, à l'aide des signaux reçus sur une antenne de capteurs ponctuels, de retrouver le ou les angles d'arrivée des sources bande-étroite desquelles ces signaux émanent. En utilisant les déphasages successifs que subit l'onde entre chaque couple de capteurs successifs, on peut mettre au point des méthodes permettant de retrouver le ou les angles d'arrivée des signaux. Notre but est de mettre au point une méthode d'estimation des angles d'arrivée qui soit robuste et, si possible, optimale, contrairement aux algorithmes souvent utilisés, type Haute-Résolution.

Nous nous intéressons aussi à l'autre volet du traitement d'antenne: la séparation aveugle de signaux. Nous tenterons de fusionner les deux approches par une méthode unificatrice utilisant à la fois l'information géométrique (volet estimation d'angles d'arrivée) et l'information statistique (volet séparation aveugle de sources).

Une fois que les sources sont localisées, il serait intéressant de pouvoir mettre en œuvre un système automatique d'"accrochage" sur chaque source. Par exemple, il serait intéressant, dans le cadre d'un studio de télévision du futur, de disposer d'un système permettant de localiser le présentateur (par les ondes sonores, par exemple), puis de le suivre avec une caméra braquée sur lui. Le système permettant de suivre en permanence les mouvements du présentateur à l'aide de capteurs (son, image ou autre) est appelé système de commande référencée capteur. Nous étudierons ce problème encore ouvert et chercherons à voir si des approches neuronales peuvent être envisagées pour le résoudre, spécialement dans le cas où la vitesse du sujet n'est pas uniforme.

Le travail relaté dans cette thèse a donné lieu à cinq publications, listées à la fin du document. Après cette introduction au sujet, le chapitre 1 présente les réseaux de neurones formels: tout d'abord on explique le fonctionnement d'un neurone biologique, puis on montre comment ce neurone est modélisé de manière formelle. Ensuite plusieurs modèles neuromimétiques sont présentés: le Perceptron MultiCouche, les réseaux entièrement connectés (modèles de Hopfield, Machine de Boltzmann, réseaux stochastiques), ainsi qu'un modèle peu connu, le CMAC, désignant un modèle imitant le fonctionnement du cervelet, l'organe de commande chez l'homme. Ce réseau est plus particulièrement destiné à des applications de robotique.

Le chapitre 2 introduit le domaine du traitement d'antenne. On y présente tout d'abord le modèle des signaux. Deux méthodes dites de "Haute-Résolution", basées sur la décomposition en valeurs propres de la matrice de covariance des

observations, sont ensuite détaillées. A ces méthodes sous-optimales, nous opposons la méthode du Maximum de Vraisemblance, qui est cependant peu utilisée parce que très coûteuse en temps de calcul. Après une présentation des approches neuronales déjà proposées dans le domaine, nous développons une méthode neuronale rapide et précise, basée sur la mise en œuvre de contraintes sur les poids. De plus on démontrera qu'elle optimise le même critère que la méthode optimale du Maximum de Vraisemblance. Nous montrons ensuite qu'en contraignant en outre le réseau à minimiser un critère statistique de dépendance des sources, il est possible d'étendre notre approche au cas que ne peuvent résoudre les méthodes au second degré (Haute-Résolution, Maximum de Vraisemblance), à savoir le cas où le nombre de sources est égal au nombre de capteurs. Des courbes comparant les performances des diverses méthodes sont présentées, qui montrent l'intérêt de notre approche.

Dans le chapitre 3 sont présentées quatre applications du problème général du traitement d'antenne: tout d'abord dans le domaine de l'analyse de documents on présente une méthode complète permettant de redresser les lignes d'un texte qui a été scanné ou écrit de travers. Ensuite nous présentons une application dans le domaine des réseaux monofréquence: il s'agit de séparer des signaux numériques convolués. La troisième application concerne un problème appelé la "cocktail-party": on s'attache à séparer les voix de deux locuteurs parlant simultanément. Enfin nous appliquons le problème de l'estimation d'angles d'arrivée à des signaux de voix humaines: avec une antenne de deux micros, nous pouvons localiser une personne qui parle. Grâce à notre approche de fusion avec une technique de séparation de sources, nous pouvons aussi localiser deux personnes qui parlent, ce qui ne peut être réalisé avec les autres méthodes.

Le chapitre 4 s'intéresse à la commande référencée capteur. Après avoir présenté le problème, nous étudions les méthodes "classiques" existantes. Nous proposons des améliorations sur les estimateurs intrinsèques à ces méthodes, puis nous étendons notre étude à deux approches neuronales différentes. Des courbes comparant ces méthodes sont présentées pour des mouvements uniformes et non uniformes.

References

[1] D.O. Hebb, "The Organization of Behavior", J. Wiley and Sons, 1949

[2] J.J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities", Proc. Natl. Acad. Science, USA, Vol 79, pp. 2554-2558, April 1982

[3] M. Minsky & S. Papert, "Perceptrons, an Introduction to Computational Geometry", MIT Press, Cambridge, 1969

[4] N. J. Nilsson, "Learning Machines", Mc Graw Hill, 1965

[5] F. Rosenblatt, "Principles of Neurodynamics", Spartan, New York, 1962

[6] D.E. Rumelhart & J. L. Mc Clelland, "Parallel Distributed Processing", Bradford Book, MIT Press, 1986

Chapitre 1
Réseaux de Neurones

La capacité d'apprentissage est un des piliers majeurs de l'évolution humaine: savoir reconnaître une situation déjà appréhendée, et ajuster son action en fonction des comportements et résultats précédents, c'est se rapprocher petit à petit de l'harmonie parfaite entre l'homme et son environnement.

Cette constatation a mené les scientifiques à élaborer, dès le milieu du siècle, des systèmes doués de capacités d'apprentissage. Après des débuts cahotiques, ces systèmes, connus désormais sous l'appellation de "modèles connexionnistes", ou "réseaux de neurones formels", ont connu un engouement exponentiel à partir du milieu des années 80. Aujourd'hui on assiste à une certaine stabilisation, et si certains modèles ont été rapidement abandonnés, d'autres n'ont plus besoin de prouver leur intérêt ni leur efficacité.

Dans une première partie, nous étudierons les cellules de base de ces réseaux de neurones, depuis leur origine biologique jusqu'à leur modélisation mathématique. Cela nous permettra alors de décrire quelques modèles connexionnistes éprouvés.

Nous examinerons dans la deuxième partie l'un des modèles les plus simples: le perceptron multicouche. Nous dériverons ses règles d'apprentissage pour le cas d'un réseau "classique" à valeurs de poids et neurones réels, puis nous en tirerons une extension au cas d'un réseau à valeurs complexes.

La troisième partie sera consacrée aux réseaux entièrement connectés, à savoir le modèle de Hopfield, et son descendant la machine de Boltzmann. Nous verrons comment l'identification de l'énergie de Lyapunov du réseau à une fonction quadratique représentant le problème à optimiser, permet le calcul a priori des connexions du réseau.

Enfin, dans la dernière partie, nous étudierons un modèle encore assez peu connu: le CMAC (pour Cerebellar Model Articulation Control). S'inspirant du fonctionnement interne du cervelet, ce modèle vise à établir, par apprentissage, un système de contrôle d'actions ou de mouvements, en fonction de données relatives à des tâches ou à un environnement.

1 Le neurone formel

1.1 Historique

Les immenses possibilités d'apprentissages du cerveau humain ont toujours fasciné les chercheurs, et c'est dans cet esprit qu'à la fin des années 40, Hebb présente des travaux [13] sur l'adaptativité d'un système à un problème donné: il met au point une règle d'apprentissage basée sur des observations biologiques. Une règle de rafraîchissement des paramètres s'apparentant à une règle de gradient est par la suite mise au point par Widrow et Hoff [21] pour des problèmes simples de traitement du signal.

En 1962, se basant sur des études poussées du fonctionnement des neurones biologiques, Rosenblatt met au point un modèle mathématique simplifié du neurone [19], et bâtit un réseau d'automates cellulaires fonctionnant selon un tel schéma: c'est le Perceptron de Rosenblatt. Pour étayer sa découverte, il applique son modèle à la reconnaissance de caractères. Très rapidement, d'autres auteurs se penchent sur ce modèle [18] [17], et montrent de graves limitations à son utilisation (problèmes linéaires simples).

Les recherches dans ce domaine sont donc quasiment abandonnées jusqu'en 1982, année où Hopfield [10] décrit un nouveau type de réseau de neurones: les réseaux entièrement connectés. Ce n'est qu'en 1986, avec la sortie de "Parallel Distributed Processing" [20], qu'un certain intérêt pour les systèmes auto-adaptatifs renaît. Si les modèles présentés dans cet ouvrage ne diffèrent pas fondamentalement de ce qui existait auparavant, ils ont le mérite d'être exposés clairement, et de mettre particulièrement en valeur les possibilités de parallélisation, que les récents progrès technologiques ont rendu possible.

1.2 Le neurone biologique

Le cerveau est formé d'environ 10^{11} neurones (un peu plus à la naissance, et beaucoup moins à la mort), qui sont reliés entre eux par un système de connexions dense et relativement complexe. La figure (1) montre schématiquement un neurone biologique et quelques unes de ses connexions. Grossièrement, on peut décrire le neurone comme une cellule recevant des informations (des influx chimiques) provenant de neurones voisins, par l'intermédiaire de connexions courtes et proches de son noyau: les dendrites. La somme de ces informations est ensuite "analysée" et transformée en influx nerveux de type électrique. Cet influx électrique est alors propagé vers les neurones suivants; il transite le long de l'axone du neurone, au bout duquel des

terminaisons forment des connexions chimiques (appelées synapses) avec d'autres neurones.

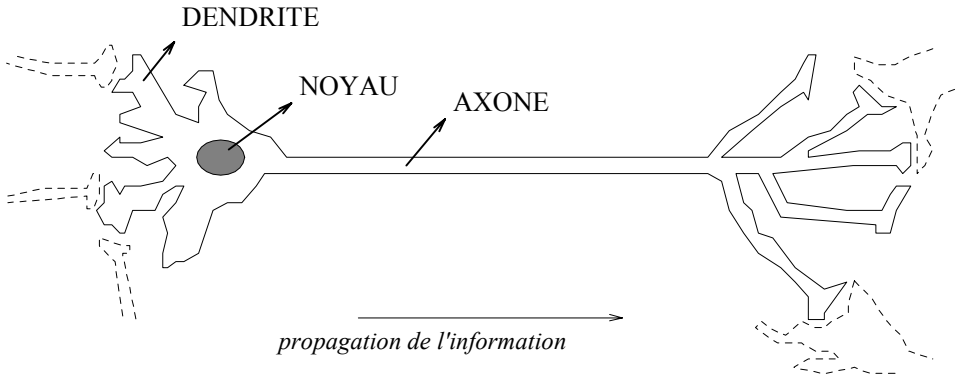

Figure 1: *Le neurone biologique*

Chaque neurone est relié en moyenne à 10^4 autres neurones, ce qui montre la densité des connexions. Suivant leur fréquence d'utilisation, certaines connexions gagnent ou perdent en force: la surface de contact entre les terminaisons de l'axone d'un neurone et les dendrites du neurone suivant, peut augmenter dans le cas d'une liaison souvent utilisée, et peut diminuer ou disparaître dans le cas contraire. Pour illustrer ce propos, il est facile de voir que l'on se souvient beaucoup mieux du visage des personnes proches, que du visage de personnes que l'on n'a rencontrées que rarement. Comme on le voit, c'est l'importance relative d'une information qui donne ou non de la force aux connexions entre les neurones.

De plus, beaucoup de ces connexions sont redondantes: malgré la perte de plusieurs milliers de neurones par jour, notre cerveau accuse peu de dégradations significatives des performances. Ceci montre la grande robustesse du système, et soulève un point intéressant: dans le cerveau, la connaissance est distribuée. Ce n'est pas un neurone unique qui contient le souvenir du visage de notre mère, c'est un ensemble de neurones. Ainsi, même si un ou plusieurs des neurones de cet ensemble est détruit, les autres neurones gardent son souvenir, et, au besoin, renforcent leurs liens pour rétablir une réponse globale équivalente.

Cette dernière remarque met en évidence cette faculté d'apprentissage perpétuel que possède le cerveau: c'est le désir de mettre en oeuvre des systèmes possédant de telles facultés, qui a poussé les savants à construire un modèle mathématique du

neurone.

1.3 Le neurone formel

Le neurone formel est construit comme un modèle mathématique simplifié du neurone biologique. La figure (2) illustre ce propos.

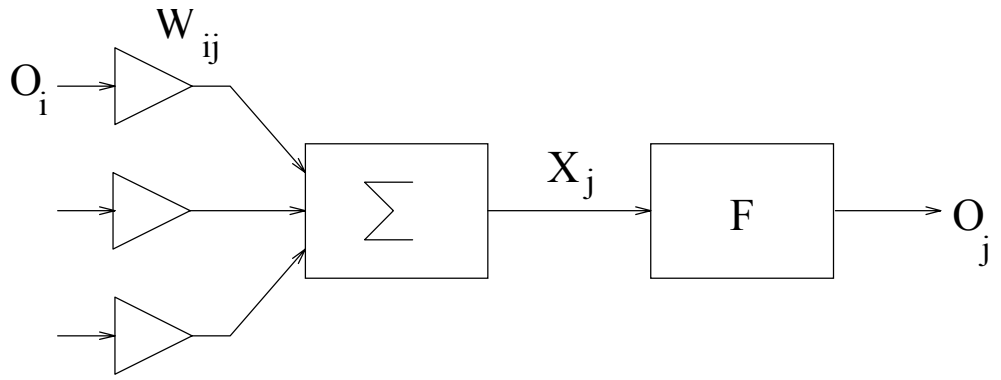

Figure 2: *Le modèle du neurone biologique*

Les informations O_i contenues dans les neurones voisins du neurone j sont pondérées par le poids W_{ij} de leur connexion, puis sont sommées pour former le potentiel interne X_j du neurone. Ce potentiel, avant d'être transmis aux neurones suivants, passe dans une fonction F plus ou moins complexe appelée *fonction de transition du neurone*. Suivant le modèle de réseau choisi, les neurones doivent répondre à certaines exigences, qui se retrouvent dans le choix de la fonction de transition. Quelques exemples vont illustrer ce propos.
Pour un Perceptron multicouche entraîné par un algorithme de rétropropagation du gradient, par exemple, il est intéressant que F présente toutefois les caractéristiques suivantes:

- F est dérivable: cette contrainte est nécessaire pour pouvoir exprimer le calcul de gradient utilisé dans l'algorithme d'apprentissage.

- F est strictement monotone: cette condition empêche l'annulation du gradient, ce qui évite tout blocage du mécanisme d'apprentissage.

- F est bornée: cela évite tout phénomène d'avalanche, et rend le réseau plus robuste vis-à-vis du bruit.

En outre, il est souvent intéressant d'utiliser une fonction de transition possédant une zone quasi-linéaire, notamment lorsque l'application nécessite une certaine précision.

Pour un Perceptron multicouche utilisant un algorithme de gradient, nous utiliserons en général pour F la tangente hyperbolique:

$$F(X_j) = th(X_j)$$

En effet, diverses études comparatives [6] sur des problèmes de classification ont montré que la fonction tangente hyperbolique donne des résultats nettement supérieurs: apprentissage plus rapide et meilleure généralisation.
Il est à noter que la dérivée de cette fonction de transition se calcule facilement:

$$F'(X_j) = 1 - O_j^2$$

A d'autres types de réseaux on associe d'autres types de neurones, et par conséquent d'autres fonctions de transition. Pour un réseau de Hopfield binaire, par exemple, les neurones doivent avoir des sorties binaires. On choisit alors la fonction signe ou une fonction de Heaviside, comme fonction de transition.
Enfin, pour un réseau de Hopfield analogique, les sorties ont leurs valeurs entre 0 et 1 en général, et la fontion utilisée est la fonction sigmoïde:

$$F(X_j) = \frac{1}{1 + e^{-X_j/\lambda}}$$

où λ est une constante à déterminer par l'opérateur en fonction de certains critères à optimiser.
Une fois qu'un modèle mathématique précis du neurone artificiel est établi, les différents Réseaux de Neurones qui seront utilisés par la suite peuvent être étudiés de manière détaillée.

2 Le Perceptron multicouche

Assez curieusement, on retrouve ici le mot "Perceptron", terme inventé par Rosenblatt pour désigner la réseau d'automates cellulaires mis en oeuvre par lui en 1962 [19], et dont on avait rapidement mis en exergue les faiblesses. En réalité, les limitations de ce premier modèle sont dues au fait que ce perceptron, parce qu'il

ne possédait que deux couches, était linéaire: il ne pouvait donc résoudre que des problèmes linéaires, c'est-à-dire des problèmes assez simples.
Ce modèle a été repris en 1986 [20], dans des travaux qui montrent que l'adjonction d'au moins une couche supplémentaire (d'où le terme de perceptron "multicouche") permet au réseau de traiter des problèmes non-linéaires.

2.1 Structure

Le Perceptron MultiCouche (ou PMC) est un réseau dont les neurones sont rangés par couches: à l'intérieur d'une couche, un neurone n'est relié à aucun voisin. Par contre, il reçoit ses entrées des neurones de la couche précédente, et transmet son état de sortie aux neurones de la couche suivante. Un exemple de PMC est présenté en figure (3).

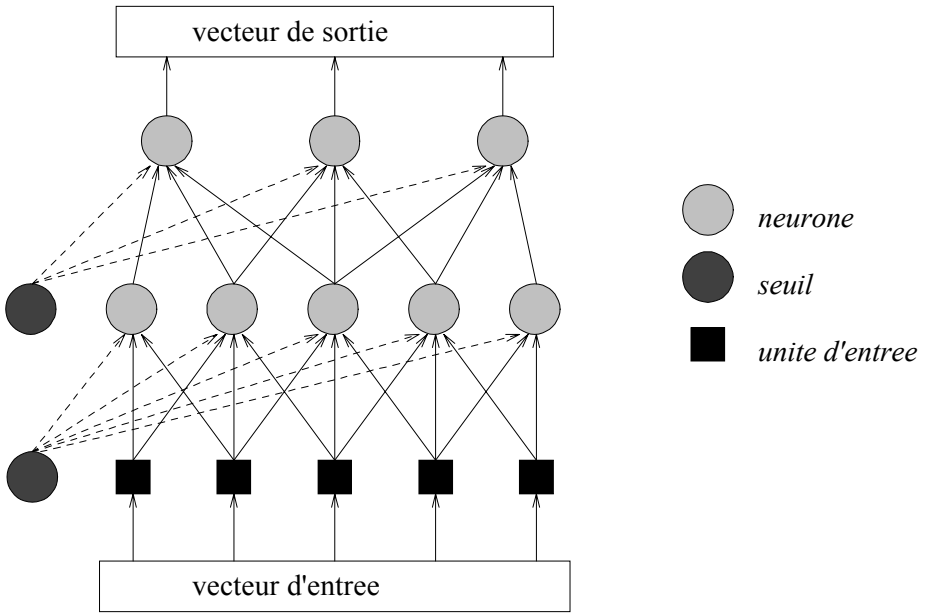

Figure 3: *Le Perceptron MultiCouche*

Mis à part les neurones de la couche d'entrée, les neurones qui forment un Perceptron multicouche sont bâtis sur le modèle formel présenté en figure (2). Les neurones de la couche d'entrée (ou première couche) ne font que recopier la valeur d'entrée qu'ils reçoivent.
Les neurones de la couche de sortie fournissent directement les valeurs du vecteur de sortie. Entre ces deux couches, il peut y avoir une ou plusieurs couches intermédiaires, appelées également couches cachées, parce qu'en général on n'a pas accès à ces

couches: vu de l'extérieur, un réseau multicouche fonctionne un peu comme un filtre non-linéaire dans lequel on entre un vecteur d'entrée, et qui fournit en sortie le vecteur issu de ce "filtrage".

2.2 Equations du réseau

Reprenons le schéma (4) d'un neurone n'appartenant pas à la couche d'entrée:

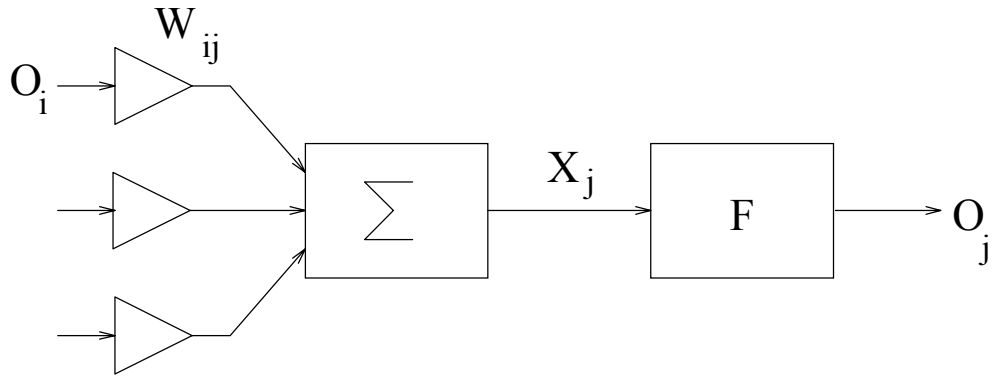

Figure 4: *Le neurone modélisé*

et notons:

- O_j = état du neurone j (sa valeur de sortie)
- F = fonction de transition du neurone
- W_{ij} = coefficient de pondération (ou "poids") affecté à la connexion entre les neurones i et j
- X_j = potentiel du neurone j (le résultat de la somme pondérée de ses entrées)

Nous pouvons donc écrire les équations générales du réseau:

$$X_j = \sum_i W_{ij} O_i \tag{1}$$

et

$$O_j = F(X_j) \tag{2}$$

Lorsqu'un vecteur d'entrée $\mathcal{E}$ est présenté au réseau, celui-ci calcule, par simple propagation, couche après couche, les états (ou valeurs) des neurones de sortie. Lorsque les poids W_{ij} ont des valeurs aléatoires, le vecteur de sortie calculé $\mathcal{O}$ est très certainement éloigné du vecteur de sortie désiré $\mathcal{S}$. Pour que le réseau effectue correctement le travail de classification que l'on souhaite le voir opérer, il convient donc de lui *apprendre* à fournir en sortie la réponse appropriée à l'entrée qu'on lui présente.

2.3 Apprentissage par rétropropagation - cas réel

L'apprentissage consiste à adapter les poids du réseau de façon que pour chaque vecteur d'entré $\mathcal{E}$ présenté, celui-ci fournisse le vecteur de sortie $\mathcal{O}$ le plus proche possible du vecteur de sortie correct $\mathcal{S}$. Pour cela, on définit une fonction d'erreur à minimiser. L'erreur quadratique en sortie pour un exemple donné est:

$$e_Q = \frac{1}{2} \sum_{j \in \text{sortie}} (O_j - S_j)^2 \tag{3}$$

La phase d'apprentissage demande que l'on dispose d'une base d'exemples: cette base d'apprentissage est composée de couples de vecteurs entrée-sortie associés, et on suppose que cette base est *représentative* du problème que l'on désire traiter. On présente alors plusieurs fois tous les exemples dans un ordre aléatoire au réseau. A chaque présentation d'un exemple, on fait varier les poids dans le sens inverse du gradient de l'erreur:

$$\Delta W_{ij} = -\alpha \frac{\partial e_Q}{\partial W_{ij}} \tag{4}$$

où α est une constante positive, ce qui assure la décroissance de l'erreur.
Notons g_{ij} le gradient local de l'erreur quadratique:

$$g_{ij} = \frac{\partial e_Q}{\partial W_{ij}} \tag{5}$$

En décomposant:

$$g_{ij} = \frac{\partial e_Q}{\partial X_j} \frac{\partial X_j}{\partial W_{ij}} \tag{6}$$

Si on pose

$$\delta_j = \frac{\partial e_Q}{\partial X_j} \tag{7}$$

alors, d'après la définition de X_j (équation (1)):

$$g_{ij} = \delta_j O_j \tag{8}$$

et δ_j est déterminé comme suit:

$\Longrightarrow$ Si j est sur la couche de sortie:

$$\begin{aligned} \delta_j &= \frac{\partial e_Q}{\partial O_j}\frac{\partial O_j}{\partial X_j} \\ &= (O_j - S_j)F'(X_j) \end{aligned}$$

d'après l'équation (3)

$\Longrightarrow$ Si j est sur une couche cachée:

$$\begin{aligned} \delta_j &= \frac{\partial e_Q}{\partial O_j}\frac{\partial O_j}{\partial X_j} \\ &= \left[\sum_{k \in succ(j)} \left(\frac{\partial e_Q}{\partial X_k}\frac{\partial X_k}{\partial O_j}\right)\right]\frac{\partial O_j}{\partial X_j} \\ &= \left[\sum_{k \in succ(j)} W_{jk}\delta_k\right] F'(X_j) \end{aligned}$$

où $succ(j)$ désigne l'ensemble des successeurs du neurone j.

En pratique, ce n'est pas l'erreur quadratique sur un exemple que l'on doit minimiser, mais l'erreur quadratique moyenne sur tous les exemples:

$$e_{QM} = E\{e_Q\} \tag{9}$$

ce qui entraîne que le gradient à considérer n'est plus le gradient local g_{ij} mais le gradient moyen:

$$\bar{g}_{ij} = E\{g_{ij}\} \tag{10}$$

que l'on estime par un simple filtrage passe-bas. Utilisons un indice t qui est incrémenté à chaque fois qu'un nouvel exemple est présenté:

$$\bar{g}_{ij}(t) = (1-\beta)g_{ij}(t) + \beta\bar{g}_{ij}(t-1) \tag{11}$$

Avec un terme d'oubli β proche de 1 (typiquement 0.98), on obtient une bonne estimation du gradient moyen.

2.4 Apprentissage par rétropropagation - cas complexe

Comme nous le verrons dans le chapitre suivant, il peut être intéressant, dans un problème mettant en jeu des données complexes, de mettre en œuvre un réseau de neurones à poids complexes, plutôt que de considérer un réseau à poids réels prenant en compte séparément les parties réelles et imaginaires des données (comme si ces données étaient en fait réelles). Dans cette optique, nous allons maintenant étendre les règles de rétropropagation au cas d'un réseau à poids complexes.

L'erreur quadratique en sortie d'un réseau complexe s'écrit de manière identique à celle d'un réseau réel:

$$e_Q = \frac{1}{2} \sum_{j \in \text{sortie}} |O_j - S_j|^2 \tag{12}$$

et l'apprentissage se déroule de la même façon, c'est-à-dire par rétropropagation, en faisant varier les poids dans le sens inverse du gradient de l'erreur.

Choisissons par convention de noter la dérivation d'un réel r par un complexe z:

$$\frac{\partial r}{\partial z} = \frac{\partial r}{\partial z^R} + j\frac{\partial r}{\partial z^I}, \tag{13}$$

où z^R et z^I sont les parties réelle et imaginaire de z, respectivement.
Avec cette convention, la règle du gradient s'exprime par:

$$\Delta W_{ij} = -\alpha \frac{\partial e_Q}{\partial W_{ij}} \tag{14}$$

Pour expliciter cette expression dans le cas complexe, nous allons d'abord démontrer une propriété nécessaire à cette dérivation.

Propriété 1: si un réel e_Q dépend d'un complexe z suivant la structure présentée sur la figure (5), avec $x_1, ..., x_n$ des complexes, h une fonction dérivable par rapport à ses parties réelle et imaginaire, et $g_1, ..., g_n$ des fonctions holomorphes, alors on a:

$$\frac{\partial e_Q}{\partial z} = \sum_k \frac{\partial e_Q}{\partial x_k} \left(\frac{\partial x_k}{\partial z} \right)^* \tag{15}$$

Démonstration: e_Q est tel que:

$$de_Q = \sum_k \left(\frac{\partial e_Q}{\partial x_k^R} dx_k^R + \frac{\partial e_Q}{\partial x_k^I} dx_k^I \right)$$

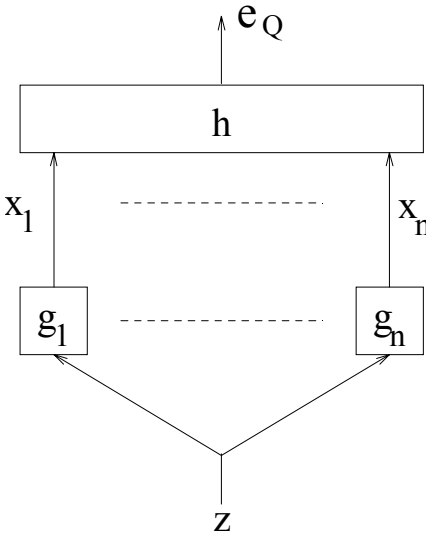

Figure 5: *Relation de dépendance entre e_Q et z*

$$\begin{aligned} &= \sum_k Re\left\{\frac{\partial e_Q}{\partial x_k} dx_k^*\right\} \\ &= Re\left\{\sum_k \frac{\partial e_Q}{\partial x_k} dx_k^*\right\} \end{aligned} \tag{16}$$

De la même façon nous pouvons écrire

$$de_Q = Re\left\{\frac{\partial e_Q}{\partial z} dz^*\right\} \tag{17}$$

Les fonctions g_k sont holomorphes, d'où

$$dx_k = \frac{\partial x_k}{\partial z} dz \tag{18}$$

ce qui, combiné à l'équation (16), conduit à:

$$de_Q = Re\left\{\sum_k \frac{\partial e_Q}{\partial x_k} \left(\frac{\partial x_k}{\partial z}\right)^* dz^*\right\} \tag{19}$$

L'identification avec l'équation (17) donne

$$Re\left\{\left[\frac{\partial e_Q}{\partial z} - \sum_k \frac{\partial e_Q}{\partial x_k} \left(\frac{\partial x_k}{\partial z}\right)^*\right] dz^*\right\} = 0 \tag{20}$$

De ce fait la propriété 1 est prouvée, puisque

$$\begin{array}{rl} Re\{ab\} = 0 \ \forall b & \Longrightarrow Re\{1.a\} = 0 \text{ et } \ Re\{j.a\} = 0 \\ & \Longrightarrow \ a = 0 \end{array} \tag{21}$$

La représentation d'un neurone complexe est schématisée en figure (6).

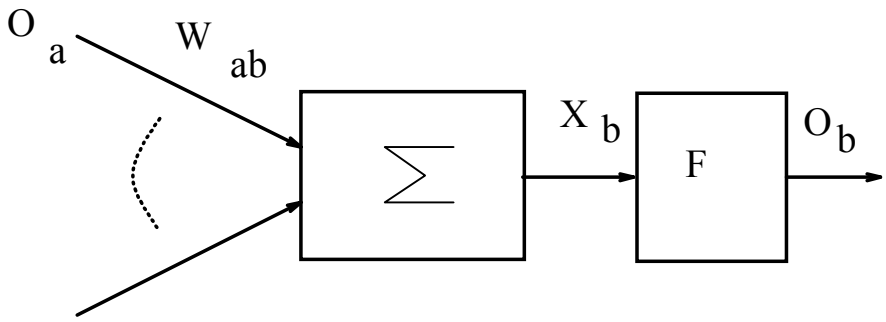

Figure 6: *Le neurone complexe*

Pour un neurone complexe tel que représenté sur la figure (6), la propriété 1 s'exprime par

$$\frac{\partial e_Q}{\partial W_{ab}} = \frac{\partial e_Q}{\partial X_b} \left(\frac{\partial X_b}{\partial W_{ab}} \right)^* \tag{22}$$

en utilisant ($z = W_{ab}, n = 1$ et $x_1 = X_b$ ici). Avec la notation précédemment utilisée dans le cas réel $\delta_b = \dfrac{\partial e_Q}{\partial X_b}$, et utilisant d'autre part la relation $X_b = \sum_a W_{ab} O_a$, on peut simplifier le gradient local:

$$g_{ab} = \frac{\partial e_Q}{\partial W_{ab}} = \delta_b O_a^* \tag{23}$$

Il nous faut maintenant expliciter δ_b. Deux cas se présentent:

- Si F est holomorphe, on a (d'après la propriété 1, avec $z = X_b, n = 1, x_1 = O_b$):

$$\delta_b = \frac{\partial e_Q}{\partial X_b} = \left(\frac{\partial e_Q}{\partial O_b} \right) \left(\frac{\partial O_b}{\partial X_b} \right)^* \tag{24}$$

- sinon il faut décomposer $\delta_b = \delta_b^R + j\delta_b^I$ et calculer ces deux termes séparément:

$$\delta_b^R = \frac{\partial e_Q}{\partial O_b^R}\frac{\partial O_b^R}{\partial X_b^R} + \frac{\partial e_Q}{\partial O_b^I}\frac{\partial O_b^I}{\partial X_b^R} \tag{25}$$

$$\delta_b^I = \frac{\partial e_Q}{\partial O_b^R}\frac{\partial O_b^R}{\partial X_b^I} + \frac{\partial e_Q}{\partial O_b^I}\frac{\partial O_b^I}{\partial X_b^I} \tag{26}$$

Pour les équations (24) à (26), la détermination de $\frac{\partial e_Q}{\partial O_b} = \frac{\partial e_Q}{\partial O_b^R} + j\frac{\partial e_Q}{\partial O_b^I}$ diffère suivant la position du neurone b.

$\Longrightarrow$ Si b est sur la couche de sortie:
Pour une erreur quadratique en sortie $e_Q = \frac{1}{2}\sum_b \mid O_b - S_b \mid^2$, on a:

$$\frac{\partial e_Q}{\partial O_b^R} = O_b^R - S_b^R$$

$$\frac{\partial e_Q}{\partial O_b^I} = O_b^I - S_b^I$$

En conséquence, on obtient:

$$\frac{\partial e_Q}{\partial O_b} = O_b - S_b$$

$\Longrightarrow$ Si b est sur une couche cachée:
$z = O_b$ et $x_k = X_k$ dans la propriété 1 conduisent à:

$$\frac{\partial e_Q}{\partial O_b} = \sum_{k \in succ(b)} \frac{\partial e_Q}{\partial X_k}\left(\frac{\partial X_k}{\partial O_b}\right)^*$$

soit, avec $X_k = \sum_b W_{bk}O_b$:

$$\frac{\partial e_Q}{\partial O_b} = \sum_{k \in succ(b)} \delta_k W_{bk}^* \tag{27}$$

où $succ(b)$ désigne l'ensemble des successeurs du neurone b.

Choix de la fonction de transition

Le choix de la fonction de transition dépend de l'application. Par exemple, pour une application de classification, on cherchera à respecter les contraintes suivantes, qui s'inspirent des contraintes imposées dans le cas réel:

- Le module de $F(z)$ est borné.
- Pour z proche de 0, on a $F(z) \simeq z$
- La restriction de F au domaine réel est une fonction respectant les contraintes imposées précédemment dans le cas réel. Plus précisément, on peut chercher à respecter $F(z) = th(z)$ pour z réel.

D'apres le théorème de Liouville, il n'existe pas de fonction F bornée qui soit holomorphe (hormis les fonctions constantes). On va d'abord présenter le cas d'une fonction holomorphe, qui ne peut pas être utilisée lorsque le problème à traiter est un problème de classification, puisque les conditions imposées ci-dessus ne sont pas respectées. Puis on présentera deux cas de fonctions F non holomorphes, mais qui respectent les conditions imposées ci-dessus.

Pour toute fonction F holomorphe telle que $O_b = F(X_b)$, on peut écrire

$$\frac{\partial O_b}{\partial X_b} = F'(X_b)$$

donc la relation (24) se simplifie:

$$\delta_b = \left(\frac{\partial e_Q}{\partial O_b}\right) (F'(X_b))^* \tag{28}$$

où

$$\begin{aligned} \frac{\partial e_Q}{\partial O_b} &= O_b - S_b \quad \text{pour la couche de sortie} \\ &= \sum_{k \in succ(b)} \delta_k W_{bk}^* \quad \text{pour une couche cachée} \end{aligned}$$

Un exemple simple de fonction holomorphe est la fonction identité: $F(z) = z$. Dans ce cas, $F'(z) = 1$, d'où une écriture très simple du terme δ_b:

$$\begin{aligned} \delta_b &= O_b - S_b \quad \text{pour la couche de sortie} \\ &= \sum_{k \in succ(b)} \delta_k W_{bk}^* \quad \text{pour une couche cachée} \end{aligned}$$

Examinons maintenant le cas de deux fonctions F qui ne sont pas holomorphes, mais qui satisfont aux conditions imposées pour des réseaux utilisés en classification:

$$F_a(z) = F_1(|z|)e^{jF_2(Arg\ z)}$$
$$F_b(z) = F_1(z^R) + jF_2(z^I)$$

où F_1 et F_2 sont des fonctions de $\mathcal{R}$ dans $\mathcal{R}$.

Il est facile de vérifier que ces fonctions ne sont pas holomorphes: en effet, les fonctions module $g(z) = |z|$ et argument $h(z) = Arg\ z$ ne sont pas holomorphes elles-mêmes, puisque l'on ne peut pas les décomposer en série de Taylor.

Dans le premier cas, on peut prendre la fonction F_a suivante:

$$F_a(z) = th|z|\ e^{j\ Arg\ z} = \frac{z}{|z|} th|z|$$

Cette fonction respecte les contraintes énoncées plus haut. Pour le calcul des dérivées partielles, établissons d'abord un résultat préliminaire (le symbole X désigne R ou I):

$$\begin{aligned}
\frac{\partial}{\partial z^X}\left(\frac{th|z|}{|z|}\right) &= \frac{1}{|z|^2}\left\{|z|\frac{\partial th|z|}{\partial z^X} - \frac{\partial |z|}{\partial z^X} th|z|\right\} \\
&= \frac{1}{|z|^2}\frac{\partial |z|}{\partial z^X}\left\{|z|(1-th^2|z|) - th|z|\right\} \\
&= \frac{1}{|z|^2}\frac{z^X}{|z|}\left\{|z|(1-th^2|z|) - th|z|\right\} \\
&= \frac{z^X}{|z|^2}\left\{1 - th^2|z| - \frac{th|z|}{|z|}\right\}
\end{aligned}$$

On a alors (les symboles X et Y désignant R ou I):

$$\begin{aligned}
\frac{\partial F_a^Y}{\partial z^X} &= \frac{\partial z^Y}{\partial z^X}\frac{th|z|}{|z|} + z^Y \frac{\partial}{\partial z^X}\left(\frac{th|z|}{|z|}\right) \\
&= \delta_{XY}\frac{th|z|}{|z|} + \frac{z^X z^Y}{|z|^2}\left\{1 - th^2|z| - \frac{th|z|}{|z|}\right\}
\end{aligned}$$

où δ_{XY} désigne le symbole de Kronecker. Finalement, on peut détailler les quatre dérivées de la fonction F_a:

$$\begin{aligned}
\frac{\partial F_a^R}{\partial z^R} &= \frac{th|z|}{|z|} + \frac{z^R z^R}{|z|^2}\left\{1 - th^2|z| - \frac{th|z|}{|z|}\right\} \\
\frac{\partial F_a^I}{\partial z^R} &= \frac{z^R z^I}{|z|^2}\left\{1 - th^2|z| - \frac{th|z|}{|z|}\right\} \\
\frac{\partial F_a^R}{\partial z^I} &= \frac{z^R z^I}{|z|^2}\left\{1 - th^2|z| - \frac{th|z|}{|z|}\right\} \\
\frac{\partial F_a^I}{\partial z^I} &= \frac{th|z|}{|z|} + \frac{z^I z^I}{|z|^2}\left\{1 - th^2|z| - \frac{th|z|}{|z|}\right\}
\end{aligned} \tag{29}$$

Dans le second cas, on peut prendre $F_b(z) = th(z^R) + j\ th(z^I)$. Cette fonction respecte les contraintes énoncées, et l'on a:

$$\begin{aligned}
\frac{\partial F_b^R}{\partial z^R} &= 1 - th^2(z^R) \\
\frac{\partial F_b^I}{\partial z^R} &= 0 \\
\frac{\partial F_b^R}{\partial z^I} &= 0 \\
\frac{\partial F_b^I}{\partial z^I} &= 1 - th^2(z^I)
\end{aligned}$$

Comme les équations (29), ces quatre équations sont à inclure dans les calculs obtenus précédemment dans le cas de fonctions non holomorphes (équations (25) et (26)). Les termes $\frac{\partial F^R}{\partial z^R}$, $\frac{\partial F^I}{\partial z^R}$, $\frac{\partial F^R}{\partial z^I}$ et $\frac{\partial F^I}{\partial z^I}$ y remplaceront respectivement les dérivées $\frac{\partial O_b^R}{\partial X_b^R}$, $\frac{\partial O_b^I}{\partial X_b^R}$, $\frac{\partial O_b^R}{\partial X_b^I}$ et $\frac{\partial O_b^I}{\partial X_b^I}$.

3 Les réseaux entièrement connectés

Si 1986 marque le véritable renouveau de l'intérêt pour les réseaux de neurones, il faut savoir que dès 1982 [10] on a vu des chercheurs se pencher sur l'optimisation, par des réseaux d'automates entièrement connectés, de problèmes complexes pour lesquels les solutions classiques nécessitaient des puissances et des temps de calcul trop grands pour être même seulement envisageables. Hopfield et Tank ont montré [10], [11], [12] que des réseaux de neurones entièrement connectés pouvaient trouver, de manière assez rapide, de bonnes solutions à ce type de problème.

Nous allons examiner le modèle développé par Hopfield, pour comprendre ce qui permet à ces réseaux sans apprentissage de résoudre de manière apparemment simple des problèmes d'optimisation supposés complexes. Puis nous nous intéresserons à des variantes de l'algorithme de base, qui présente le défaut majeur de ne garantir la convergence que vers un minimum local.

3.1 Modèle de base: le réseau de Hopfield

Un réseau de neurones formels dit “de Hopfield" est un réseau composé de neurones

entièrement connectés, c'est-à-dire que la sortie d'un neurone est reliée à l'entrée de tous les autres neurones, et qu'en entrée il reçoit les sorties de tous les autres neurones.

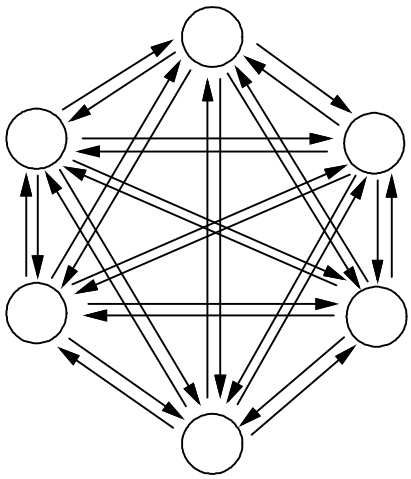

Figure 7: *Un réseau entièrement connecté dit réseau de Hopfield.*

La figure (7) montre un réseau de Hopfield: ce réseau peut être soit binaire, soit analogique, c'est-à-dire que ses neurones peuvent prendre soit des valeurs binaires (0 et 1, ou bien -1 et $+1$), soit des valeurs analogiques (toutes les valeurs possibles entre 0 et 1).
Un neurone sera dit binaire si sa fonction de transition f est une fonction de Heaviside ou une fonction signe, et il sera dit analogique si sa fonction de transition est une fonction continue et monotone croissante à valeurs dans $[0, 1]$. Nous allons étudier de manière séparée le fonctionnement interne des réseaux de Hopfield suivant qu'ils sont binaires ou analogiques.

3.1.1 Réseaux binaires

Soit un réseau de N neurones tous connectés entre eux. La connexion entre la sortie du neurone i et l'entrée du neurone j est pondérée par un poids synaptique T_{ij}. En général, la matrice des poids synaptiques est symétrique ($T_{ij} = T_{ji}$). D'autre part, pour éviter les oscillations, il est préférable que le poids (T_{ii}) reliant la sortie d'un neurone à son entrée soit nul.
Chaque neurone possède en outre une entrée extérieure I_i, qui peut s'apparenter à un biais ou un seuil. La figure (8) montre le modèle d'un neurone binaire appartenant à un réseau de Hopfield.

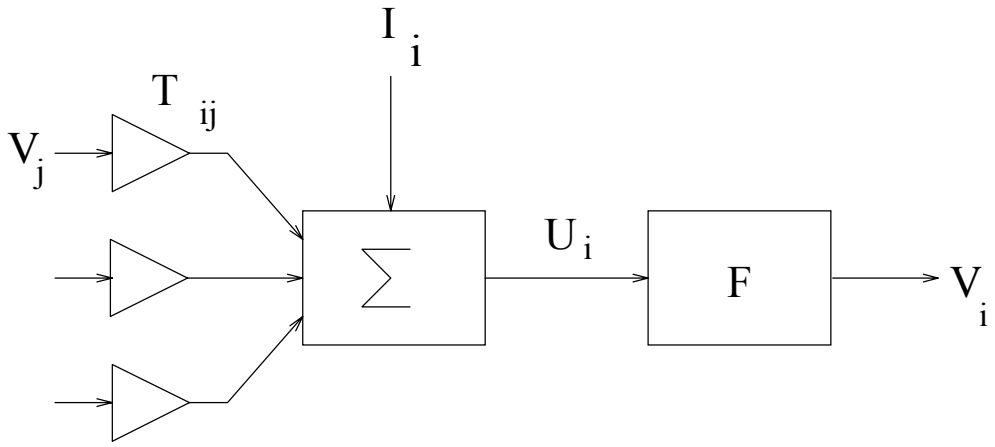

Figure 8: *Schéma d'un neurone binaire de Hopfield.*

L'état interne $U_i(t)$ et la sortie $V_i(t)$ du neurone i sont définis par:

$$U_i(t+1) = \sum_{j=1}^{N} T_{ji} V_j(t) + I_i \tag{30}$$

$$V_i(t+1) = f\left(U_i(t+1)\right) \tag{31}$$

Pour un réseau binaire la fonction de transfert f est soit la fonction signe, soit une fonction de Heaviside. Les N sorties du réseau forment donc un "mot" de N bits qui constitue, après convergence, la solution du problème.
La mise en oeuvre d'un réseau de Hopfield se fait en plusieurs étapes:

1. calcul des poids synaptiques T_{ij} et des entrées externes I_i en fonction du problème à traiter.

2. initialisation aléatoire uniforme des sorties binaires V_i $(t = 0)$.

3. $t = t + 1$. calcul de l'état interne $U_i(t)$ et de la sortie $V_i(t)$ des neurones.

4. rebouclage sur l'étape 3 jusqu'à la convergence du réseau (stabilisation de l'énergie du réseau, définie plus loin).

Les réseaux binaires eux-mêmes se séparent en deux catégories, suivant qu'ils

fonctionnent en mode séquentiel (les neurones s'activent l'un après l'autre à tour de rôle) ou en mode parallèle (tous les neurones changent d'état en même temps).

Mode séquentiel

En mode séquentiel, l'étape 3 mentionnée ci-dessus se compose de N sous-étapes: les N neurones sont rafraîchis les uns après les autres dans un ordre aléatoire. Le principe du réseau de Hopfield est de se définir une énergie E correspondant au problème à résoudre, et que l'on cherche à minimiser. L'énergie E du réseau s'appelle aussi fonction de Lyapunov du réseau [16]. Soit l'énergie suivante:

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} T_{ij}V_i(t)V_j(t) - \sum_{i=1}^{N} I_i V_i(t) \tag{32}$$

et soit k le neurone que l'on cherche à rafraîchir. Nous allons calculer la variation d'énergie totale du réseau engendrée par le changement d'état du neurone k.
L'énergie du réseau avant le changement d'état du neurone k peut s'écrire comme:

$$E = C - \frac{1}{2}\sum_{j\neq k} T_{kj}V_k(t)V_j(t) - \frac{1}{2}\sum_{i\neq k} T_{ik}V_i(t)V_k(t) - \frac{1}{2}T_{kk}V_k^2(t) - I_k V_k(t) \tag{33}$$

où C désigne un terme ne dépendant pas de k. Puisque l'on a choisi une matrice de poids symétrique, on a $T_{ij} = T_{ji}\ \ \forall(i,j)$, donc E se simplifie:

$$E = C - \sum_{j\neq k} T_{kj}V_k(t)V_j(t) - \frac{1}{2}T_{kk}V_k^2(t) - I_k V_k(t)$$

Après le changement d'état du neurone k, cette énergie devient égale à:

$$E' = C - \sum_{j\neq k} T_{kj}\left(V_k(t) + \Delta V_k(t)\right)V_j(t) - \frac{1}{2}T_{kk}\left(V_k(t) + \Delta V_k(t)\right)^2 - I_k\left(V_k(t) + \Delta V_k(t)\right)$$

ce qui donne une différence d'énergie de:

$$\Delta E = E' - E = -\sum_{j\neq k} T_{kj}V_j(t)\Delta V_k(t) - \frac{1}{2}T_{kk}\left(2V_k(t)\Delta V_k(t) + \left(\Delta V_k(t)\right)^2\right) - I_k\Delta V_k(t)$$

En simplifiant:

$$\Delta E = -\sum_{j=1}^{N} T_{kj}V_j(t)\Delta V_k(t) - I_k\Delta V_k(t) - \frac{1}{2}T_{kk}\left(\Delta V_k(t)\right)^2$$

soit, d'après la définition de l'état interne $U_i(t)$:

$$\Delta E = -U_k(t+1)\Delta V_k(t) - \frac{1}{2}T_{kk}\left(\Delta V_k(t)\right)^2 \tag{34}$$

Il est important de noter que pour une fonction de transition monotone croissante

telle que celle choisie (fonction signe ou Heaviside), le premier terme de cette relation est toujours négatif. Le second terme n'est pas de signe garanti, dès lors que T_{kk} est issu de la modélisation du problème, ce qui montre un nouvel avantage à imposer des poids T_{ii} nuls lorsque cela est possible, afin de s'assurer de la décroissance de l'énergie.
Supposons que nous ayons effectivement choisi des T_{ii} nuls. Nous pouvons maintenant écrire

$$\Delta E = -U_k(t+1)\Delta V_k(t) \tag{35}$$

Si $U_k(t+1)$ et $U_k(t)$ sont de signe opposé, alors ΔV_k est non nul et possède le même signe que $U_k(t+1)$. Le changement d'état du neurone k est donc intéressant pour le réseau puisqu'il fait décroître son énergie globale. Si au contraire $U_k(t+1)$ et $U_k(t)$ sont de même signe, alors l'état de sortie V_k du neurone k reste identique (le neurone ne change pas d'état) et l'energie reste au même niveau (elle ne croit ni ne décroit). Si lors d'une itération aucun des N neurones ne change d'état, un minimum d'énergie est atteint, et l'état constaté des neurones correspond à un état stable. Un même réseau peut posséder plusieurs états stables, correspondant à autant de minima locaux, que l'on ne peut mettre en exergue que si l'on procède à plusieurs minimisations d'énergie initialisées aléatoirement par des états de départ V_i différents. Il suffit ensuite de comparer les valeurs respectives de l'énergie sur ces minima pour en déduire l'état correspondant au minimum global du réseau.

En pratique, il se peut que le minimum global se trouve dans un bassin d'attraction très étroit, ce qui veut dire qu'il y a très peu de chances que l'état aléatoire initial soit dans ce bassin, auquel cas cette méthode a peu de chances de fournir le minimum global cherché. D'un autre côté, il se peut aussi que le problème possède un très grand nombre de minima locaux, ce qui là encore rend la découverte du minimum global peu probable.

Mode parallèle
En mode parallèle, ce sont tous les neurones qui sont rafraîchis en même temps: cette façon de procéder confère un avantage certain à ce type de réseau, car il est facilement parallélisable, étant donné que tous les neurones sont des entités identiques et effectuant les mêmes opérations au même moment. Ce réseau est donc aussi potentiellement beaucoup plus rapide que le précédent, ce qui peut s'avérer intéressant si, toujours dans le but de trouver le minimum global de l'énergie, l'on désire procéder à plusieurs initialisations différentes de l'état de départ du réseau.
La valeur de l'énergie du réseau avant le changement d'état s'écrit:

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} T_{ij}V_i(t)V_j(t) - \sum_{i=1}^{N} I_i V_i(t)$$

Après le changement d'état, cette énergie sera

$$E' = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} T_{ij}\left(V_i(t)+\Delta V_i(t)\right)\left(V_j(t)+\Delta V_j(t)\right) - \sum_{i=1}^{N} I_i\left(V_i(t)+\Delta V_i(t)\right)$$

ce qui donne une différence d'énergie de

$$\Delta E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} T_{ij}\left(V_j(t)\Delta V_i(t)+V_i(t)\Delta V_j(t)\right) - \sum_{i=1}^{N} I_i\Delta V_i(t) - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} T_{ij}\Delta V_i(t)\Delta V_j(t)$$

Et puisque $T_{ij} = T_{ji}$:

$$\Delta E = -\sum_{i=1}^{N}\sum_{j=1}^{N} T_{ij}V_j(t)\Delta V_i(t) - \sum_{i=1}^{N} I_i\Delta V_i(t) - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} T_{ij}\Delta V_i(t)\Delta V_j(t)$$

soit, d'après la définition de $U_i(t+1)$:

$$\Delta E = -\sum_{i=1}^{N} U_i(t+1)\Delta V_i(t) - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} T_{ij}\Delta V_i(t)\Delta V_j(t) \tag{36}$$

Avec un choix de fonction de transition monotone croissante, le premier terme est toujours négatif. Par contre, même avec la convention $T_{ii} = 0$, le second terme est de signe quelconque. On ne peut donc pas garantir la décroissance de l'énergie, comme on avait pu le faire pour un réseau séquentiel.
Heureusement, en modifiant la définition de l'énergie de l'équation (32), on obtient une nouvelle énergie de Lyapunov adaptée au mode parallèle:

$$F = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} T_{ij}V_i(t)V_j(t+1) - \frac{1}{2}\sum_{i=1}^{N} I_i\left(V_i(t)+V_i(t+1)\right) \tag{37}$$

D'après [7] et [8], il est possible de montrer que cette nouvelle formulation de l'énergie assure sa décroissance à chaque rafraîchissement de l'état du réseau, et que le réseau convergera alors vers un état stable (minimum local) ou un cycle de longueur 2 qui minimise F.

Comme on l'a vu dans le paragraphe précédent, et encore plus ici, les réseaux de Hopfield binaires fonctionnent de manière très simple, et lorsque le mode est parallèle, ces réseaux sont également très rapides. L'inconvénient des réseaux binaires est qu'ils sont restreints aux coins de l'hypercube de dimension N définissant l'ensemble des états de sortie possibles du réseau. De ce fait, la convergence est fortement tributaire de l'état initial, et il existe une probabilité assez importante de tomber dans un minimum local ou dans un cycle.
On peut s'affranchir de cet inconvénient en procédant à plusieurs convergences: en partant à chaque fois d'un état aléatoire différent pour l'initialisation des V_i, on

peut obtenir à chaque fois un minimum d'énergie éventuellement différent en fin de convergence. En prenant le minimum des minima d'énergie ainsi définis, on peut penser qu'on a alors plus de chances de se trouver au minimum global du problème.

3.1.2 Réseaux analogiques

C'est en partie pour tenter de résoudre les problèmes de minima locaux artificiellement engendrés par l'emploi de réseaux à valeurs binaires, qu'ont été créés les réseaux de Hopfield analogiques. Les neurones de ces réseaux peuvent prendre toutes les valeurs entre 0 et 1, ce qui évite les problèmes de blocage dans les minima locaux dus à la discrétisation des paramètres. En outre, il faut savoir qu'au départ cette notion de réseau analogique implique l'utilisation d'un vrai circuit analogique câblé avec Amplificateurs Opérationels, résistances et capacités. De ce fait, les neurones fonctionnent de manière continue: la convergence d'un tel réseau est extrêmement rapide (aussi rapide que la diffusion des courants dans un circuit), et rend son utilisation très intéressante pour des problèmes temps-réel.
Le réseau de Hopfield analogique comporte plus de paramètres que le réseau numérique. Tout d'abord, le fonction de transition $V_i = f(U_i)$ n'est plus une fonction binaire, mais une fonction continue à valeurs dans $[0, 1]$. On utilise la fonction sigmoïde, qui est monotone croissante:

$$f(x) = \frac{1}{1 + e^{-x/T}} \tag{38}$$

où T est la "température" du réseau. On remarque que pour des températures faibles, on retrouve une fonction f proche d'un fonctionnement en mode binaire: seuls les coins de l'hypercube sont atteints. Au contraire, pour des températures élevées, le réseau possède une grande liberté pour se déplacer à l'intérieur du cube, ce qui lui évite de se retrouver prisonnier des minima locaux artificiels du mode binaire. Cependant, on désire en général que la solution finale fournie par le réseau soit un vecteur binaire de N états. La technique utilisée consiste donc à travailler à une température "moyenne", ce qui permet à la fois de décrire l'intérieur du cube, et en même temps de converger vers une solution proche d'un vecteur binaire. Il est aussi possible d'ajouter une contrainte "binarisante", comme nous le verrons plus loin.

L'énergie de Lyapunov d'un réseau analogique comporte un terme supplémentaire par rapport au réseau binaire:

$$E = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} T_{ij}V_i(t)V_j(t) - \sum_{i=1}^{N} I_i V_i(t) + \sum_{i=1}^{N}\frac{1}{\tau}\int_0^{V_i(t)} f^{-1}(v)dv \tag{39}$$

où τ est la constante de temps d'un neurone analogique. Ici encore, pour une fonction de transition monotone croissante et des T_{ij} symétriques, on peut montrer [11], [12] que l'énergie est une fonction décroissante du temps, ce qui garantit la convergence

du réseau vers un état final stable.
En outre, si l'on désire s'assurer un état final binaire [11], [12], il suffit d'ajouter un terme correcteur à l'énergie:

$$\sum_{i=1}^{N} \alpha_i V_i(t) \left(V_i(t) - 1\right)$$

tel que le nouveau poids T'_{ii} vérifie $T'_{ii} = T_{ii} + \alpha_i = 0$.

Lorsque leur utilisation est possible, les réseaux de Hopfield analogiques apparaissent donc comme un type de solution possédant des qualités très intéressantes: grande rapidité, possibilité de converger une solution binaire ou non binaire selon le choix de l'utilisateur, et suppression des minima artificiels engendrés par la discrétisation des états donc garantie d'une convergence vers une "bonne" solution. D'un autre côté, il intervient désormais de nouveaux paramètres qu'il faut savoir régler judicieusement, et qui nécessitent l'intervention d'un "spécialiste", ou en tout cas d'une persone expérimentée. Enfin, même si les minima artificiels créés par la discrétisation des $V_i(t)$ pour les neurones binaires ont disparu, il reste que les minima locaux inhérents au problème d'optimisation lui-même sont toujours présents: seule la convergence vers un minimum local du problème est garantie.

3.2 La machine de Boltzmann ou recuit simulé

Issue des travaux de Kirkpatrick [14], la machine de Boltzmann allie astucieusement les caractéristiques des réseaux de Hopfield binaires, des ressemblances avec les réseaux analogiques, et les propriétés de la physique statistique (ou physique de Boltzmann). Dans ce modèle, les états des neurones sont binaires, mais la loi d'activation est probabiliste.
Si on revient aux défauts des réseaux numériques de Hopfield, on remarque que l'énergie du réseau décroit à chaque itération, ce qui peut l'entraîner dans un minimum local. Avec la machine de Boltzmann, on va au contraire accepter (avec une certaine probabilité) que l'énergie totale du réseau augmente, justement dans le but d'éviter ces minima locaux. Cependant, pour éviter que le réseau ne diverge, la probabilité d'acceptation d'une transition neuronale augmentant l'énergie sera d'autant plus faible que cette augmentation d'énergie sera grande. C'est ici que l'on retrouve la physique de Boltzmann: une transition ΔV_i sera acceptée avec une probabilité

$$p\left(\Delta V_i\right) = \frac{1}{1 + e^{-\Delta E_i / T}} \tag{40}$$

où ΔE_i est la variation d'énergie provoquée par la transition ΔV_i du neurone i. L'appellation "machine de Boltzmann" provient donc clairement du fait que

l'on retrouve ici le *facteur de Boltzmann* $e^{-\Delta E_i/T}$. On remarque au passage la correspondance avec les réseaux analogiques, puisque la forme de cette probabilité est la même que celle de la fonction de transition d'un neurone analogique (équation (38)). La valeur moyenne de sortie d'un neurone de Boltzmann est donc égale à la valeur de sortie d'un neurone de Hopfield analogique, ce qui montre que toutes ces variantes du réseau de Hopfield de base sont liées.
D'autre part, d'après l'équation (35), la variation d'énergie provoquée par la transition du neurone i dans un réseau binaire symétrique tel que $T_{ii} = 0$, est $\Delta E = -U_i \Delta V_i$. En séparant le cas $V_i(t+1) = 1$ du cas $V_i(t+1) = 0$ et en normalisant les deux expressions, il est possible d'expliciter (40) en fonction de $U_i(t+1)$. Pour un état interne calculé

$$U_i(t+1) = \sum_{j=1}^{N} T_{ji} V_j(t) + I_i$$

l'état de sortie $V_i(t+1)$ du neurone ne sera plus une fonction binaire déterministe de $U_i(t+1)$, mais vaudra 0 ou 1 avec une probabilité qui dépend directement de $U_i(t+1)$:

$$p\left(V_i(t+1) = 1\right) = \frac{1}{1 + e^{-U_i(t+1)/T}}$$

et

$$p\left(V_i(t+1) = 0\right) = 1 - p\left(V_i(t+1) = 1\right) = \frac{e^{-U_i(t+1)/T}}{1 + e^{-U_i(t+1)/T}}$$

De la même manière que précédemment, on démarre avec une température T élevée, ce qui implique des probabilités de transition proches de $\frac{1}{2}$:

$$p\left(V_i(t+1) = 1\right) = p\left(V_i(t+1) = 0\right) = \frac{1}{2}$$

Les V_i se comportent ici comme des variables aléatoires uniformes ne dépendant pratiquement pas de l'état iterne U_i. On explore ainsi tous les coins du cube. Puis la température diminue petit à petit, et c'est l'état interne qui détermine alors la sortie V_i: pour un état interne négatif, la probabilité d'avoir une sortie égale à 1 est faible, mais non nulle. On autorise donc (avec des probabilités de plus en plus faibles au fur et à mesure que la température décroit) que l'énergie croisse lors d'une transition, ce qui permet de sortir de minima locaux, mais ne permet pas de sortir du minimum global. La figure (9) illustre bien ce phénomène: le saut d'énergie nécessaire pour faire sortir le réseau du minimum local (à gauche) n'est pas trop important.

Pour une température pas trop faible, la probabilité de sortir de ce minimum local est donc non négligeable. Au contraire, le minimum global étant par définition le plus profond, la probabilité d'en sortir est beaucoup plus faible. Si la température baisse un peu, les neurones ont alors une probabilité très importante de se trouver dans le bassin d'attraction du minimum global. Si la température baisse encore, les neurones

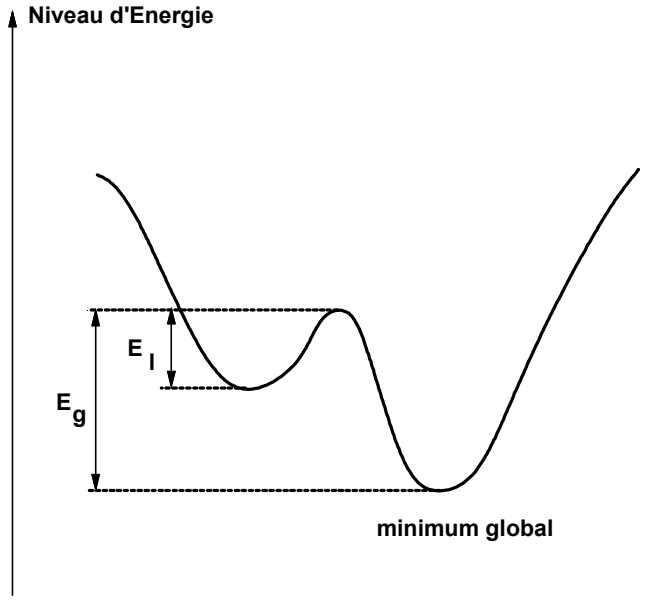

Figure 9: *Le saut d'énergie nécessaire pour s'échapper d'un minimum local est plus faible que le saut correspondant au minimum global.*

ont un comportement pratiquement déterministe, tout à fait semblable à ceux d'un réseau de Holpfield binaire, ce qui garantit en théorie la convergence finale vers le minimum de ce bassin, c'est-à-dire le *minimum global* du problème. D'après une étude théorique [1] s'intéressant au problème de la convergence de ce type de réseau, il est possible de montrer que la convergence vers le minimum global du problème à optimiser n'est garantie que si la loi de décroissance de la température est de la forme

$$T(t) = \frac{1}{\log t}$$

avec un nombre d'itérations théoriquement infini.

L'algorithme de Boltzmann se déroule de la manière suivante:

1. calcul des poids synaptiques T_{ij} et des entrées externes I_i par identification de l'énergie de Lyapunov du problème avec la forme quadratique à optimiser.

2. initialisation aléatoire uniforme des sorties binaires V_i. Choix de la valeur initiale T_0 de la température.

3. pour $i = 1$ jusqu'à Nombre_de_Neurones:

(a) calcul de l'état interne $U_i(t)$.

(b) acceptation du changement d'état ΔV_i avec la probabilité $p(\Delta V_i)$ donnée dans l'équation (40).

4. $t = t + 1$. Diminution de la température: $T(t) = T_0/\log t$

5. Si pour M essais consécutifs, aucune transition n'est acceptée, arrêt de l'algorithme. Sinon, retour à l'étape 3.

L'intérêt de la Machine de Boltzmann est donc la garantie de convergence vers le minimum global à condition que la loi de décroissance de la température soit suffisamment lente [1]. En pratique, il est clair que la loi en $1/\log t$ est trop lente, et que l'on ne peut se permettre d'opérer une minimisation sur un temps "théoriquement" infini. En général, on utilise donc des lois de décroissance plus rapides, et on limite le nombre d'itérations, ce qui ne garantit plus la convergence théorique vers le minimum global.

Le principal défaut de ce type de réseau réside dans sa grande lenteur. Ainsi, lorsque se pose un problème d'optimisation, il faut d'abord répondre à la question suivante avant de le résoudre: désire-t-on une solution rapide et simple à mettre en œuvre? Dans ce cas le réseau de Hopfield binaire semble le plus approprié. Désire-t-on une solution *très* rapide? Dans ce cas le réseau analogique est approprié (encore faut-il disposer du matériel câblé immédiatement disponible). Désire-t-on la meilleure solution? Dans ce cas la machine de Boltzmann est recommandée, mais il est nécessaire de bien connaître le comportement du système face aux réglages de température, et surtout, il ne faut pas être dans l'urgence. Comme on le voit, ce n'est pas forcément la "meilleure" solution qui sera employée, parce qu'elle demande des précautions d'usage importantes, un personnel qualifié, et beaucoup de temps.

3.3 Les réseaux analogiques stochastiques

Les réseaux stochastiques analogiques possèdent les avantages de la machine de Boltzmann (possibilité d'accepter une petite augmentation de l'énergie du réseau, ce qui améliore la probabilité de convergence vers le minimum global), sans leurs inconvénients: les neurones sont analogiques donc la convergence est rapide.
Le principe de fonctionnement de ces réseaux se base sur un processus stochastique dirigé par un bruit blanc gaussien d'intensité décroissante avec le temps: l'état interne $U_i(t + 1)$ s'écrit maintenant:

$$U_i(t+1) = \sum_{j=1}^{N} T_{ji} V_j(t) + I_i + \lambda_i(t) b_i(t)$$

La perturbation provoquée dans U_i se retrouve sur V_i par la fonction sigmoïde

$$V_i = f(U_i) = \frac{1}{1 + e^{-U_i/T}}$$

Au démarrage, l'intensité $\lambda_i(t)$ du bruit ajouté est assez forte. U_i ne dépend donc que très peu de l'état des autres neurones; sa valeur est presque aléatoire, et V_i se comporte donc comme une variable aléatoire uniforme à valeurs dans $[0, 1]$. On retrouve ici, en version analogique, le comportement de la valeur de V_i (équation (40)) pour une machine de Boltzmann avec une température élevée.
Après un certain nombre d'itérations, on laisse l'intensité $\lambda_i(t)$ décroître. U_i dépend alors à la fois de l'état interne du réseau, et d'un bruit perturbateur dont la conséquence peut être l'augmentation faible mais ponctuelle de l'énergie totale du réseau. Sans opérer de la même façon, on reconnaît ici une caractéristique du réseau de Boltzmann, qui est d'accepter (lorsque le tirage au sort en a décidé ainsi) une petite augmentation de l'énergie de Lyapunov. Il est bon de rappeler que ce sont ces petites augmentations de l'énergie qui permettent de s'échapper des minima locaux, car leurs barrières de potentiel sont par essence moins importantes que celle du minimum global.
Lorsque $\lambda_i(t)$ devient négligeable, le réseau stochastique se comporte comme un réseau de Hopfield analogique, et converge rapidement vers le minimum du bassin dans lequel il se trouve. Là encore, se profile en transparence le comportement de la machine de Boltzmann qui, pour des températures faibles, se comporte comme un réseau de Hopfield numérique.

En conclusion, on peut dire que le réseau stochastique analogique converge vers le minimum global au même titre que la machine de Boltzmann. Les sauts d'énergie que la machine de Boltzmann crée en acceptant, avec une faible probabilité, une transition ΔV_i contraire au signe de U_i, sont à comparer aux sauts d'énergie que les réseaux stochastiques produisent en ajoutant du bruit dans le calcul de U_i. Cependant, de par leur nature purement analogique, les réseaux stochastiques sont beaucoup plus rapides, et donc plus intéressants à mettre en œuvre, que la machine de Boltzmann. Enfin, la difficulté de réglage du paramètre température dans le cadre de l'algorithme de Boltzmann se retrouve toute entière dans les réseaux stochastiques analogiques, puisqu'il s'agit ici encore de trouver une loi de décroissance de $\lambda_i(t)$ qui garantisse la convergence du réseau vers le minimum global.
Finalement, on peut définir le cadre d'emploi des différents réseaux entièrement connectés de la manière suivante: pour un problème binaire simple dont on ne cherche qu'une solution "possible", on peut employer un réseau de Hopfield numérique. Pour un problème d'optimisation dont on désire trouver une "bonne" solution rapidement, on peut employer un réseau de Hopfield analogique. Enfin pour un problème où la

meilleure solution est exigée, quel que soit le temps de convergence, on peut employer un réseau stochastique analogique, tout en sachant que cette solution exige de bien savoir régler les différents paramètres mis en jeu.

4 Le modèle du cervelet d'Albus

4.1 Introduction

Au début des années 70, prenant comme point de départ une étude neurophysiologique de Grossman [9], Albus [2] s'est penché de manière approfondie sur les fonctions opérationnelles du cervelet. Le cervelet est une petite partie du cerveau plus particulièrement dédiée au contrôle des mouvements chez l'homme. D'après ces deux études, les informations arrivant au cervelet proviennent des muscles, des articulations ou de la peau, sous la forme de données relatives à la position ou a la vitesse des membres. Des informations de haut niveau, telles des commandes de mouvements désirés, peuvent également y parvenir. D'après les études théoriques [9] [2], le cervelet fonctionnerait comme un tableau d'adressage: chaque vecteur d'entrée (incluant des informations sensorielles et des commandes de haut niveau) se comporterait comme l'adresse d'une case mémoire à l'intérieur de laquelle se trouvent les signaux qu'il faut envoyer aux muscles pour que la commande désirée soit réalisée.

La théorie complète d'Albus décrivant le fonctionnement interne du cervelet biologique est décrite dans [3] et [4]. Par la suite, Albus [5] a reformulé ce modèle de manière plus mathématique, et l'a baptisé CMAC (pour Cerebellar Model Articulation Controller[1]). C'est ce modèle mathématique que nous allons décrire ici.

4.2 Structure du CMAC

4.2.1 Une première ébauche

La façon dont la structure du CMAC a germé dans l'esprit d'Albus montre que celui-ci s'est au départ beaucoup inspiré du perceptron à deux couches de Rosenblatt [19]. A l'époque les ordinateurs n'étaient pas très courants, et c'est pourquoi les

[1] Contrôleur d'Articulation Modélisant le Cervelet

fonctions internes du CMAC d'Albus (comme la plupart des "machines à apprendre" de l'époque) semblent s'inspirer d'un câblage réel de composants électroniques. Une première ébauche du CMAC est présentée en figure (10). Pour Albus, il est possible

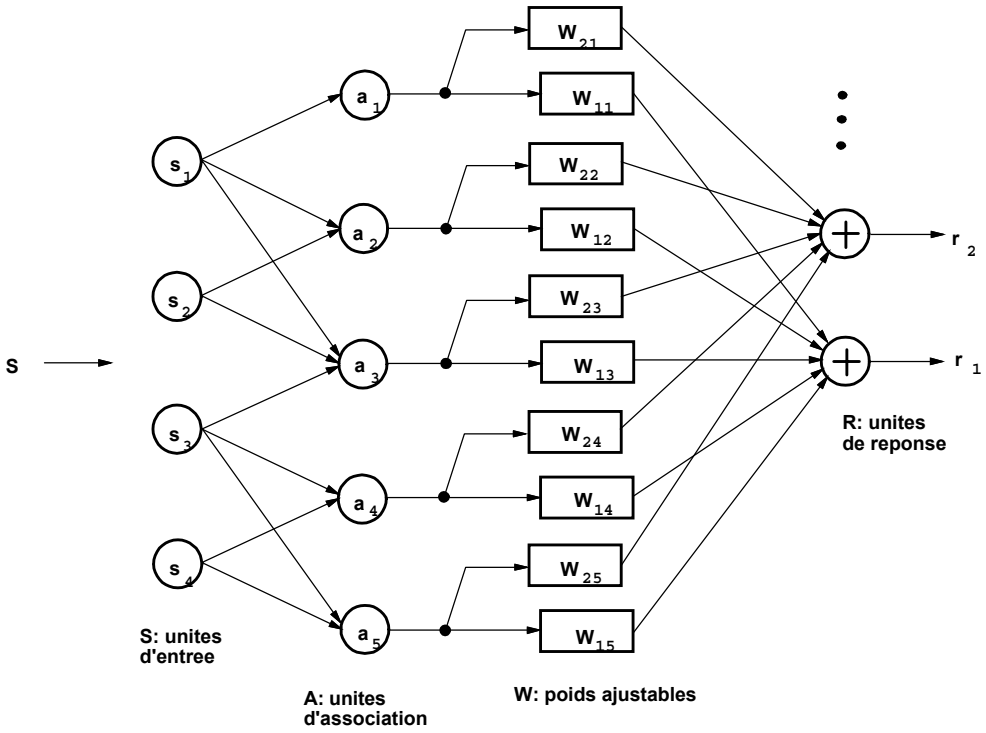

Figure 10: *Représentation d'une première ébauche du CMAC d'Albus*

de résumer ce CMAC en une juxtaposition de deux étapes différentes:

- une première étape (fixe) associe les unités d'entrée de S aux cellules d'association de A:

$$f: \quad S \quad \rightarrow \quad A$$

- une seconde étape envoie les sorties de A sur les unités de réponse R à travers des poids ajustables par apprentissage:

$$g: \quad A \quad \rightarrow \quad R$$

Les unités d'entrée s_i sont binaires ou L-aires suivant le problème. Elles ne sont pas complètement connectées aux unités d'association a_j, en ce sens qu'une unité a_j ne reçoit pas forcément des signaux de *tous* les s_i. Les sorties de A sont binaires: elles valent 0 ou 1. Pour un vecteur d'entrée S donné, beaucoup

d'éléments de A seront à zéro: l'ensemble des éléments de A qui sont à 1 est dénoté A^*.

Chaque élément r_k du vecteur sortie R correspond à la somme de ses poids W_{kl} qui sont reliés aux sorties de l'ensemble A^*. Remarquons qu'à chaque sortie r_k est associée un jeu de poids $(W_{k1}, W_{k2}, ..., W_{kN})$ différent.

Albus observe que le vecteur d'entrée S peut être considéré comme une adresse, et le vecteur réponse R peut être considéré comme le *contenu* de cette adresse. Si pour une entrée S on désire changer[2] la sortie R, il suffit donc de modifier les poids associés aux unités de A^*. Les autres poids ne sont pas modifiés. Une représentation illustrant ce caractère adresse/contenu dans le cas d'une seule unité réponse r est montrée en figure (11). On voit que pour le vecteur d'entrée considéré, l'ensemble

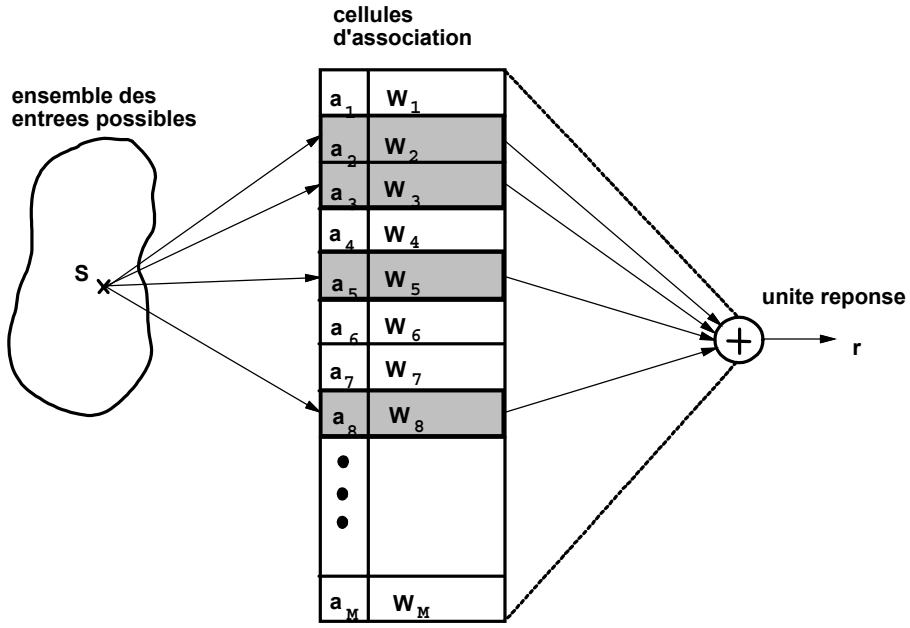

Figure 11: *Exemple de CMAC avec une seule unité de réponse. Ici,* $A^* = \{a_2, a_3, a_5, a_8\}$ *d'où* $r = W_2 + W_3 + W_5 + W_8$

A^* des unités d'association qui sont à 1 est:

$$A^* = \{a_2, a_3, a_5, a_8\}$$

La réponse r est alors la somme des poids activés par ces unités:

$$r = W_2 + W_3 + W_5 + W_8$$

[2]pour se conformer à une base d'apprentissage, par exemple

Pour modifier la sortie r associée à l'entrée S, il suffit donc de modifier ces 4 poids.

Albus met alors en avant le fait que deux vecteurs d'entrée S_1 et S_2 peuvent activer des ensembles d'unités d'association A_1^* et A_2^* dont l'intersection n'est pas nulle (voir figure (12)). Si nous souhaitons voir le CMAC associer à S_1 et S_2 des

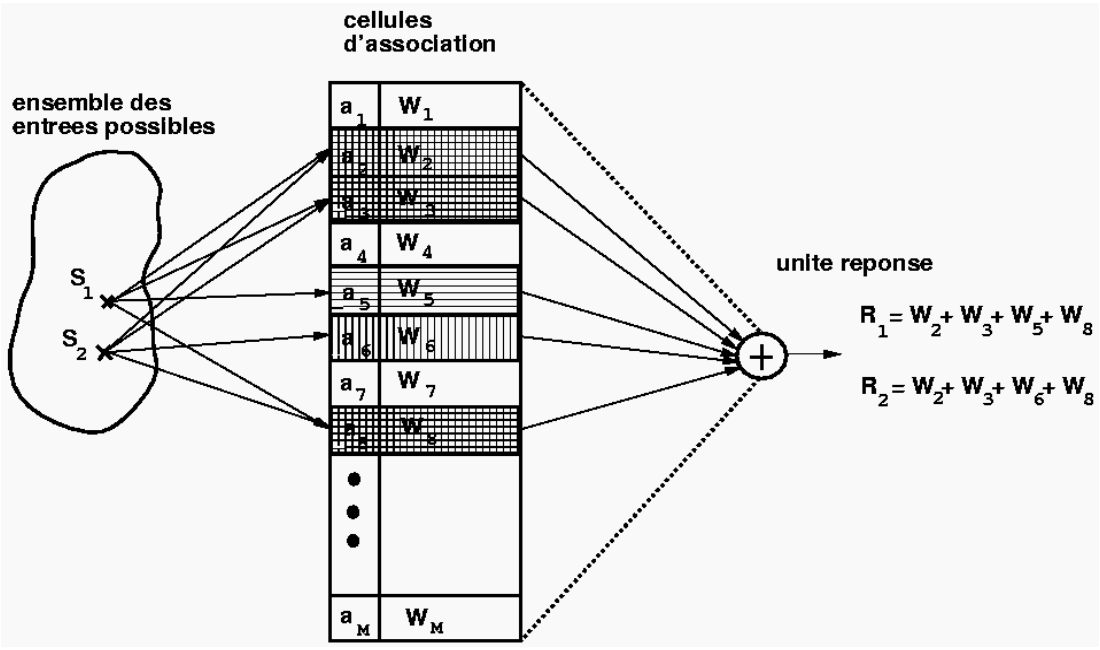

Figure 12: *Cas d'intersection non nulle entre deux ensembles activés par des entrées différentes*

réponses R_1 et R_2 proches, alors le fait que l'intersection $A_1^* \cap A_2^*$ n'est pas nulle est bénéfique: cette configuration rappelle les capacités des organismes vivants à s'adapter, à apprendre et à généraliser d'une expérience sur un autre. Ici la généralisation est favorisée. Si au contraire nous souhaitons que S_1 et S_2 soient associés à des réponses sensiblement différentes, alors cette configuration pose problème: l'apprentissage de l'exemple S_1 contrarie l'apprentissage de l'exemple S_2 et réciproquement. C'est ce que l'on appelle l'interférence sur apprentissage[3] ou inhibition rétroactive.

Cela conduit à penser qu'il faudrait construire les connexions entre S et A de manière à favoriser l'apprentissage, et à éliminer les inhibitions rétroactives. En d'autres termes, si deux vecteurs S_1 et S_2 sont séparés par une petite distance dans l'espace des entrées, alors l'intersection $A_1^* \cap A_2^*$ devrait être grande. Si au contraire S_1 et S_2 sont séparés par une grande distance dans l'espace des entrées,

[3]learning interference en anglais

alors l'intersection $A_1^* \cap A_2^*$ devrait être nulle.
On en arrive à une certaine notion de voisinage dans l'espace des entrées, qui sera développée dans la section suivante. Pour ce qui est de cette étude préliminaire, Albus propose la règle suivante:

> On construira les connexions de S vers A de manière à ce que $A_1^* \cap A_2^*$ soit nulle pour tous les vecteurs d'entrée S_1 et S_2, sauf pour les S_1 et S_2 "voisins".

Avant de s'étendre sur cette notion de voisinage, il est nécessaire de quantifier les tailles des différentes couches du CMAC. Pour s'assurer que $A_1^* \cap A_2^*$ est pratiquement tout le temps nulle, il est nécessaire que le nombre total d'éléments de A soit beaucoup plus grand que le nombre d'éléments dans A^*. Soit $|A^*|$ le nombre d'éléments dans A^*. On considère que ce nombre est le même[4] pour chaque vecteur d'entrée S.

Notons $|A|$ le nombre d'unités potentiellement actives du vecteur d'association A. Albus pense que dans le cervelet humain, le rapport entre $|A^*|$ et $|A|$ est de l'ordre de 1%. On choisit donc ici $|A|$ au moins 100 fois supérieur à $|A^*|$.
Pour éviter que deux vecteurs d'entrée différents S_1 et S_2 activent exactement les mêmes unités d'association A^*, il est nécessaire que le nombre de vecteurs d'entrée possibles soit inférieur ou égal au nombre de combinaisons de $|A^*|$ éléments qu'il est possible de choisir parmi $|A|$ éléments au total.

Soit N la longueur du vecteur d'entrée S. Pour des entrées L-aires, on a L^N vecteurs d'entrée possibles. D'autre part le nombre de combinaisons de $|A^*|$ éléments parmi $|A|$ est:

$$C_{uv}^u \qquad \text{où} \quad v = \frac{|A|}{|A^*|} \quad \text{et} \quad u = |A^*|$$

soit:

$$\begin{aligned} C_{uv}^u = \frac{(vu)!}{u!(uv-u)!} &> \frac{(uv-u)^u}{u!} \\ &> \frac{(uv-u)^u}{u^u} = (v-1)^u \end{aligned}$$

Ainsi, avec $|A|$ au moins 100 fois supérieur à $|A^*|$, la condition nécessaire pour

[4] cela garantit entre autres une certaine homogénéité dans l'ordre de grandeur des sorties r_i. En effet, chaque sortie r_i est la somme de $|A^*|$ poids W_{ij}. Si on admet que ces poids ont tous à peu près le même ordre de grandeur, alors il est préférable d'avoir, pour chaque sortie, la somme du même nombre de poids. Cela favorise également, lors de l'apprentissage, une modification équirépartie des poids.

que deux entrées S_1 et S_2 différentes correspondent à des A_1^* et A_2^* disjoints est approximée par:

$$L^N < 99^{|A^*|} \tag{41}$$

4.2.2 Structure détaillée du CMAC

Maintenant que nous avons quantifié la taille des couches du CMAC d'Albus, voyons plus en détail la structure complète de ce réseau. Dans le CMAC simplifié évoqué plus haut, nous avions mentionné des liens reliant directement l'entrée S aux éléments de A. En réalité, dans la version complète du CMAC imaginé par Albus, les connexions entre le vecteur d'entrée S et les unités de A ne se font pas de manière directe, mais résultent de la mise en cascade de deux fonctions:

$$\begin{array}{rcl} S & \rightarrow & M \\ \text{puis} \quad M & \rightarrow & A \end{array}$$

comme indiqué sur la figure (13). L'ensemble M forme la couche des "Détecteurs d'Espace d'Etats". Ces éléments sont des ET logiques. Les unités d'entrée (S) ne sont pas complètement interconnectées aux Détecteurs d'Espace d'Etats: elles sont connectées de manière éparse mais régulière, de façon à ce que chaque vecteur d'entrée excite *exactement* C Détecteurs d'Espace d'Etats.

Les sorties des Détecteurs d'Espace d'Etats sont connectées de manière aléatoire[5] à un ensemble A plus petit d'unités appelées les Détecteurs de Champs Multiples. Ces unités sont des OU logiques. A chaque Détecteur d'Espace d'Etats n'est relié qu'un seul Détecteur de Champs Multiples. Il y a ensuite bijection entre les Détecteurs de Champs Multiples et les poids ajustables. La sortie correspondante est la somme des poids associés aux Détecteurs de Champs Multiples qui sont activés. Lors de l'apprentissage, l'erreur de sortie est rétropropagée sur la couche de poids uniquement, par un algorithme de gradient. Les autres couches ne sont pas modifiées.

Remarquons ici que C, qui est le nombre de Détecteurs d'Espace d'Etats excités par chaque vecteur d'entrée S, correspond également au nombre de Détecteurs de Champs Multiples excités. En effet, un Détecteur d'Espace d'Etats ne pointe que vers un seul Détecteur de Champs Multiples. Or les Détecteurs de Champs Multiples sont des OU logiques, donc sauf collision il y a C Détecteurs de Champs Multiples

[5] que nous détaillerons par la suite

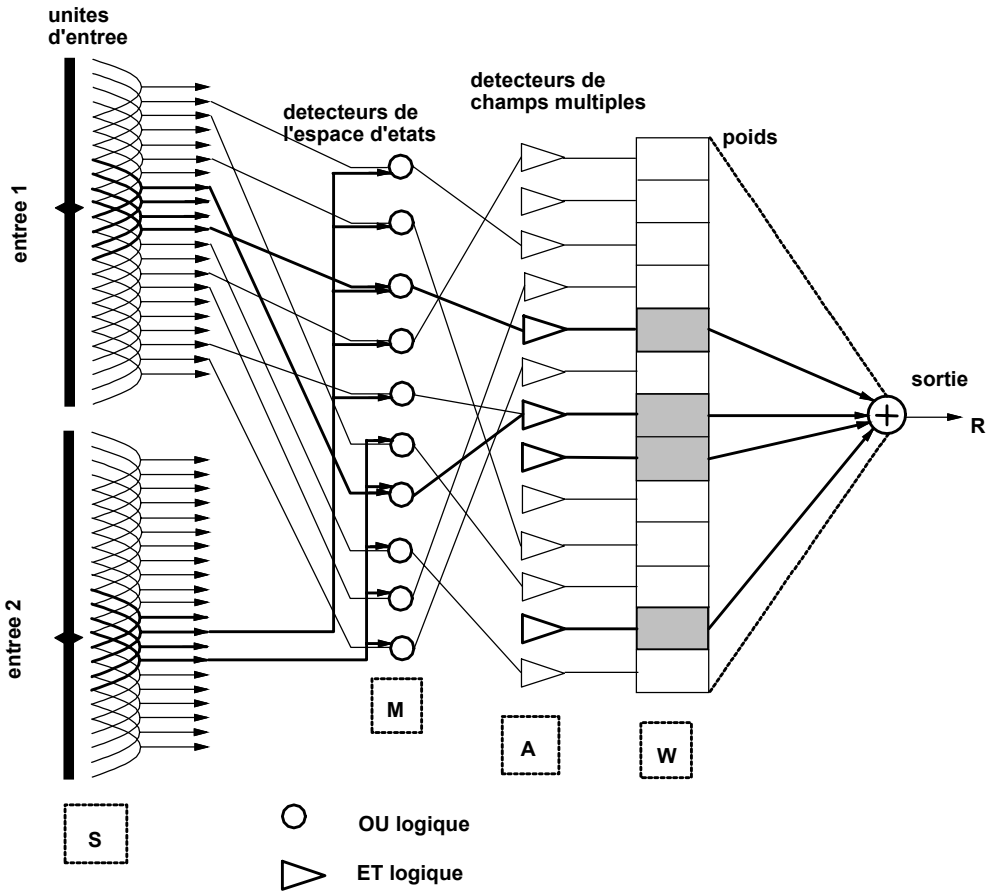

Figure 13: *Schéma logique du CMAC*

sélectionnés[6] pour chaque vecteur d'entrée S. Ce nombre C correspond aussi au nombre $|A^*|$ évoqué dans la section précédente.
Pour un vecteur d'entrée S de longueur N dont les éléments sont L-aires, C doit donc obéir à:

$$L^N \leq 99^C$$

d'après l'équation (41). Pour C, cela équivaut à

$$C \geq \frac{N \ln L}{\ln 99}$$

D'autre part, le nombre de Détecteurs de Champs Multiples (qui est aussi le nombre P de poids par sortie) est le nombre $|A|$ évoqué précédemment. Ce nombre vaut:

[6]dans le cas de 2 Détecteurs d'Espace d'Etats pointant sur le même Détecteur de Champs Multiples, on considère que ce Détecteur de Champs Multiples est sélectionné deux fois, et donc que le poids auquel il correspond est sommé deux fois dans le calcul final de la sortie R

$$P = |A| = 100\ |A^*| = 100\ C$$

Il ne reste plus maintenant qu'à examiner la façon dont les liens se font entre les couches S, M et A. Nous verrons comment ces liens sont créés dans le cas d'une seule entrée s, puis dans le cas de deux entrées s_1 et s_2. Une généralisation sera alors envisagée pour le cas multidimensionnel $(s_1, s_2, ..., s_N)$.

Cas monodimensionnel

On expliquera ici ce qui se passe pour une seule entrée et une seule sortie. La généralisation à plusieurs sorties est évidente: voir l'Annexe A à ce sujet.

Nous avons une seule entrée: $N = 1$. Choisissons pour illustrer cet exemple de prendre $L = 5$ niveaux de quantification pour ce paramètre d'entrée s. On peut noter ici que plus L est grand, plus la discrimination entre des entrées différentes est possible. Choisissons d'autre part le nombre d'unités actives par entrée: soit $C = 4$ par exemple. On peut noter que plus C est grand, plus la généralisation des entrées proches est possible (car il y a alors un grand nombre d'éléments dans l'intersection $A_1^* \cap A_2^*$). Le tableau ci-dessous montre, en fonction du numéro de l'intervalle dans lequel se trouve l'entrée s, l'image m de cette entrée:

s	m
1	A, B, C, D
2	E, B, C, D
3	E, F, C, D
4	E, F, G, D
5	E, F, G, H

La figure (14) montre comment ce tableau est contruit: à chaque entrée s on attribue un numéro en fonction de l'intervalle de quantification dans lequel cette valeur se situe. A ce numéro correspondent C éléments de l'ensemble m. Plus deux éléments sont proches (sur la figure s et s' sont dans deux intervalles voisins), plus leurs images sont proches. Ici, l'intersection des images de s et s' contient 3 éléments: C, D et E. Notons que pour mettre en valeur la ressemblance entre deux entrées proches, les éléments de l'image conservent toujours la même position dans le tableau. Par exemple, après avoir supprimé l'élément A et ajouté l'élément E dans le tableau en ligne 2, on aurait pu penser garder l'ordre alphabétique: B, C, D, E. Au contraire on garde, pour les lettres communes à la ligne précédente, la même position dans la liste, et on place le nouveau symbole dans la place laissée vacante. Ici, cela donne: E, B, C, D qui fait suite à A, B, C, D. Nous verrons par la suite pourquoi cette règle est si importante.

En réalité, ce codage n'est pas effectué avec des symboles alphabétiques, mais

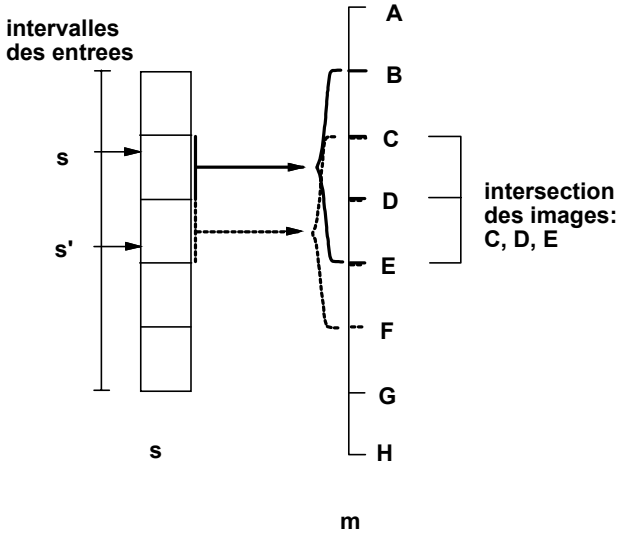

Figure 14: *Détermination de l'image d'une entrée s*

avec des entiers représentant des clés d'adresses: A a la valeur 1, B la valeur 2, etc. Quant à la couche des Détecteurs de Champs Multiples, elle n'existe pas vraiment: les clés codées sur la couche M servent à calculer des adresses informatiques, qui se trouvent être les adresses des poids W. Ce calcul est effectué par une fonction de hachage [15], qui s'exprime très simplement dans le cas monodimensionnel.
Il y a P poids reliés à la sortie, mais seuls C poids (ici $C = 4$) seront effectivement activés. Les poids sont rangés dans des cases mémoire numérotées de 0 à $P - 1$. Pour une entrée s située dans l'intervalle de quantification numéro 4, par exemple, on a $m = [E, F, G, D]$, soit, pour l'équivalent en chiffres, $m = [5, 6, 7, 4]$. Notons ce vecteur de longueur C de manière plus littérale:

$$m = [m(1), m(2), ..., m(i), ..., m(C)]$$

pour un cas général où C Détecteurs d'Espace d'Etats doivent être activés. Les C poids qui seront sélectionnés sont les poids dont l'adresse vaut $m(i)$ modulo P: soit u_i l'adresse d'un poids W_i sélectionné ($u_i \in [0, P-1]$). On a:

$$\forall i \in [1, C], \quad u_i = m(i) \text{ modulo } P$$

Nous verrons que pour deux entrées ou plus, la fonction de hachage s'exprime de manière un peu plus élaborée.

Cas bidimensionnel
Nous étudions ici un réseau à deux entrées: s_1 et s_2. Nous avons donc $N = 2$. Choisissons, pour illustrer un exemple très général, de quantifier s_1 et s_2 sur un

nombre différent d'intervalles:

- s_1 codé sur 5 niveaux
- s_2 codé sur 7 niveaux

Et prenons comme le cas précédent $C = 4$ unités actives par entrée. Rappelons que C doit être le même pour toutes les entrées, car C correspond aussi au nombre de poids qui sont sommés pour donner la réponse finale R. Ce nombre ne peut donc varier. Nous allons d'ailleurs voir plus loin que la structure du codage au niveau de la couche M ne peut autoriser des C différents pour les différents paramètres d'entrée.

Le tableau suivant représente le codage $s_1 \rightarrow m_1$ pour l'entrée s_1 uniquement:

s_1	m_1
1	A, B, C, D
2	E, B, C, D
3	E, F, C, D
4	E, F, G, D
5	E, F, G, H

On constate que ce tableau correspond exactement au tableau qui avait été montré plus haut pour le cas d'une entrée ayant les mêmes paramètres structurels ($C = 4$, $L = 5$). Le tableau correspondant au codage $s_2 \rightarrow m_2$ est représenté ci-dessous:

s_2	m_2
1	a, b, c, d
2	e, b, c, d
3	e, f, c, d
4	e, f, g, d
5	e, f, g, h
6	i, f, g, h
7	i, j, g, h

On peut dès lors construire le tableau global représentant le codage $(s_1, s_2) \rightarrow (m_1, m_2)$, et c'est ici que l'on comprend la nécessité d'avoir pour toutes les entrées le même nombre C: c'est une question de compatibilité des tableaux de codage. Le tableau suivant illustre la concaténation des deux tableaux qui viennent d'être présentés:

	m_1	A, B, C, D	E, B, C, D	E, F, C, D	E, F, G, D	E, F, G, H
m_2	$s_2 \backslash s_1$	1	2	3	4	5
a, b, c, d	1	AaBbCcDd	EaBbCcDd	EaFbCcDd	EaFbGcDd	EaFbGcHd
e, b, c, d	2	AeBbCcDd	EeBbCcDd	EeFbCcDd	EeFbGcDd	EeFbGcHd
e, f, c, d	3	AeBfCcDd	EeBfCcDd	EeFfCcDd	EeFfGcDd	EeFfGcHd
e, f, g, d	4	AeBfCgDd	EeBfCgDd	EeFfCgDd	EeFfGgDd	EeFfGgHd
e, f, g, h	5	AeBfCgDh	EeBfCgDh	EeFfCgDh	EeFfGgDh	EeFfGgHh
i, f, g, h	6	AiBfCgDh	EiBfCgDh	EiFfCgDh	EiFfGgDh	EiFfGgHh
i, j, g, h	7	AiBjCgDh	EiBjCgDh	EiFjCgDh	EiFjGgDh	EiFjGgHh

Pour mieux comprendre comment ce tableau, en apparence complexe et de règles obscures, fonctionne, étudions le cas pratique suivant (représenté en figure (15)): soit un couple (s_1, s_2) d'entrées tel que s_1 soit dans l'intervalle numéro 2, et

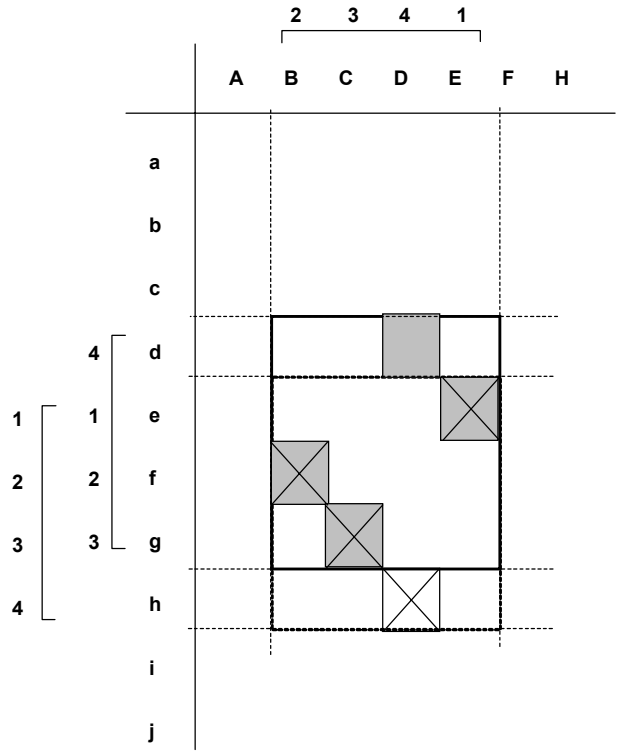

Figure 15: *Illustration pratique de la concaténation des tableaux de s_1 et s_2*

s_2 dans l'intervalle numéro 4. On sait donc, d'après les deux premiers tableaux, que s_1 est codé par (E, B, C, D) et s_2 par (e, f, g, d), dans cet ordre. Sur la figure (15), l'intersection brute entre ces deux listes est matérialisée par le grand carré cerclé de noir. Sans le codage judicieux d'Albus, cette intersection brute est matérialisée par $C \times C = 4 \times 4 = 16$ petits carrés; cela représente une place mémoire importante, surtout si on pense qu'en général il y a plus que 2 entrées! Un tel codage demande en fait C^N symboles par intersection, pour un réseau à N entrées, ce qui peut devenir

bien trop lourd à gérer. C'est pourquoi Albus a l'idée de ranger les symboles dans un ordre strict, de telle façon qu'un symbole donné occupe toujours la même place dans la liste. Cette numérotation est matérialisée sur les côtés de la figure (15). En ne faisant fusionner que les symboles qui ont le même numéro de place (numéro $\in [1, C]$), on remplace le codage en C^N symboles par un codage en C symboles: ici, seuls $C + 4$ petits carrés représentent désormais l'intersection des images de s_1 et s_2. Ces 4 carrés sont matérialisés par du grisé sur la figure.

L'économie substantielle réalisée par ce codage (remplacement de C^N symboles par C symboles seulement) n'est nullement contrebalancée par une perte des propriétés de l'intersection entre les images de 2 entrées proches. En effet, cherchons maintenant l'image de (s_1, s'_2), avec s'_2 proche de s_2:

$$\begin{array}{ccc} s_2 & \rightarrow & (e, f, g, d) \\ s'_2 & \rightarrow & (e, f, g, h) \end{array}$$

On voit que s_2 et s'_2 ont une grande partie de leurs images qui est commune. Au niveau de l'intersection brute, on a (s_1, s_2) matérialisé par le carré cerclé de noir, et (s_1, s'_2) matérialisé par le carré cerclé de pointillés. Leur intersection fait 12 petits carrés sur un maximum de 16, soit un rapport de 3/4. Au niveau de l'intersection résultant du codage numéroté d'Albus, on a: les petits carrés grisés représentent (s_1, s_2) et les petits carrés marqués d'une croix représentent (s_1, s'_2). Il y a trois carrés à la fois marqués d'une croix et grisés, pour un maximum de 4 possibles, soit ici encore, un rapport de 3/4.
Le codage numéroté d'Albus respecte donc parfaitement la notion du voisinage, tout en permettant un codage peu coûteux.
En ce qui concerne les connexions entre les sorties des Détecteurs d'Espace d'Etats et les poids, la fonction de hachage est moins dépouillée que dans le cas précédent à une seule entrée.
On vient de voir ici que pour s_1 codé sur le niveau 2 et s_2 codé sur le niveau 4, on a:

$$\left\{ \begin{array}{lcl} m_1 & = & [E, B, C, D] \\ m_2 & = & [e, f, g, d] \end{array} \right.$$

soit, en chiffres:

$$\left\{ \begin{array}{lcl} m_1 & = & [5, 2, 3, 4] \\ m_2 & = & [5, 6, 7, 4] \end{array} \right.$$

Notons ces deux vecteurs m_1 et m_2 (de longueur $C = 4$) de manière plus littérale:

$$\left\{ \begin{array}{lcl} m_1 & = & [m_1(1), m_1(2), ..., m_1(C)] \\ m_2 & = & [m_2(1), m_2(2), ..., m_2(C)] \end{array} \right.$$

On se donne un nombre H assez grand (mais inférieur à P), qui n'a pas de diviseurs communs avec P. Ce nombre H va introduire un aspect pseudo-aléatoire dans les connexions entre les Détecteurs d'Espace d'Etats et les Détecteurs de Champs Multiples (qui sont directement liés aux poids). Ainsi, pour chaque unité active i, l'adresse u_i du poids sélectionné par la conjonction des entrées s_1 et s_2 est:

$$u_i = [m_2(i) + H\ m_1(i)] \text{ modulo } P \quad \forall i \in [1, C]$$

L'utilisation d'un grand entier premier permet une bonne répartition des poids activés sur toute l'étendue des poids possibles. On voit par ailleurs que les collisions sont autorisées car il se peut que deux couples $(m_1(i), m_2(i))$ et $(m_1(j), m_2(j))$ différents sélectionnent le même poids. Dans ce cas ce poids sera sommé deux fois dans le calcul de la sortie.

Cas multidimensionnel
D'après l'exemple précédent, on peut extrapoler la façon de construire le codage d'Albus $S \rightarrow M$ dans le cas de plusieurs entrées: soit N le nombre d'entrées, soit L le nombre maximal de niveaux de quantification des entrées[7]. La valeur minimale de C est calculée par:

$$C \geq \frac{N \ln L}{\ln 99}$$

avec C entier. On peut alors construire les N tableaux représentant le codage sur C symboles de chacune des entrées. Il suffit alors de concaténer ces N tableaux monodimensionnels en un unique tableau N-dimensionnel, où, grâce au codage numéroté d'Albus, à chaque N-uplet d'entrée $(s_1, s_2, ..., s_N)$, correspond un unique C-uplet de symboles. On voit l'intérêt, pour N et L grands, du codage d'Albus sur C symboles seulement.

Une fois que les images des entrées $(s_1, s_2, ..., s_N)$ sont calculées, il ne reste plus qu'à faire pointer chacune de ces "clés" sur une case contenant un poids. L'adressage vers les poids est réalisé par une fonction de hachage [15] avec collisions. Ce code permet de n'avoir à stocker qu'un faible nombre de poids, par rapport au très grand nombre de combinaisons possibles en entrée.
Pour un vecteur d'entrée $[s_1, s_2, ..., s_N]^T$, l'image $m_j, j \in [1, N]$ de chaque entrée s_j est notée

$$\forall j \in [1, N], \quad m_j = [m_j(1), m_j(2), ..., m_j(C)]$$

[7]En général il est plus facile de prendre le même L pour toutes les entrées, mais il peut arriver, par exemple, que certaines entrées seulement soient binaires. Dans ce cas certaines entrées ont deux niveaux de quantification et d'autres en ont L. C'est ce L maximal qui servira à évaluer le C optimal.

Chaque unité (ou Détecteur d'Espace d'Etats) activée par les entrées s'écrit:

$$[m_1(i), m_2(i), ..., m_N(i)] \quad i \in [1, C]$$

et elle pointe sur le poids W_i correspondant à l'adresse u_i calculée par:

$$u_i = [m_N(i) + H.(m_{N-1}(i) + H.(... + m_1(i)))] \text{ modulo } P$$

C'est ici que l'on comprend mieux comment ce codage permet, par une méthode pseudo-aléatoire[8], de passer d'un ensemble virtuellement très grand à un ensemble beaucoup plus petit. Notons que ce codage est utilisé, par exemple, pour le stockage de matrices très grandes mais dont presque tous les éléments sont à zéro[9].

Pour C unités d'entrée actives, on a dit au départ qu'il fallait environ $P = |A| = 100C$ poids. On voit donc malgré tout, que même pour C petit, le nombre de poids est non négligeable. De plus, ce nombre $P = 100C$ de poids est valable pour une sortie. Pour r sorties, il faut r fois plus de poids, soit $100rC$ poids au total. Pour donner un exemple chiffré, disons que pour une application où on aurait $N = 30$ entrées codées chacune sur 20 niveaux de quantification, on aurait

$$C \geq \frac{30 \ln 20}{\ln 99} = 19.56$$

soit $C = 20$. On a alors

$$P = 100C = 2000$$

poids pour chaque sortie. Ainsi, pour $r = 6$ sorties, par exemple, on arrive à un total de

$$100rC = 12000$$

poids, ce qui n'est pas négligeable.

4.3 Apprentissage

Le CMAC ne possède qu'une seule couche de poids ajustables (le reste de ses connexions est fixé lors de la création de la structure), ce qui rend son algorithme

[8] mais déterministe, puisqu'à partir d'un vecteur d'entrée donné, on retrouve toujours les mêmes adresses u_i de poids

[9] pour cette application proprement dite, il est nécessaire de gérer les collisions afin d'éviter d'écraser certaines valeurs. Par contre, pour le CMAC, les collisions ne sont pas gênantes, tant que leur probabilité reste faible

d'apprentissage très simple.

Nous expliciterons ici les calculs pour une seule sortie R. Etant donné que les poids des sorties sont indépendants entre deux sorties différentes, nous avons parfaitement le droit de séparer les calculs sortie par sortie.

Nous appellerons W_i les poids reliés à la sortie étudiée. Rappelons que pour une entrée S donnée, seuls C Détecteurs de Champs Multiples sont allumés, donc il y a C poids au plus qui sont activés par ces Détecteurs de Champs Multiples. En général C poids sont effectivement activés. Dans le cas d'une collision dans la fonction de hachage, il se peut qu'un même poids soit activé par deux Détecteurs de Champs Multiples différents. Dans ce cas on prendra la convention de sommer ce poids deux fois, afin de conserver l'homogénéité des sorties, sans que cela ne gêne les calculs qui suivent.

En sortie on a:

$$R = \sum_i W_i c_i$$

où

- $c_i = 0$ ou 1 (valeur de la sortie du Détecteur de Champs Multiples correspondant au poids W_i)
- $\sum_i c_i \leq C$ si on ne somme pas deux fois les poids sélectionnés deux fois, et $\sum_i c_i = C$ sinon

La formule d'ajustement des poids suit la règle du gradient:

$$\Delta W_i = -\alpha \frac{\partial e}{\partial W_i}$$

où e est l'erreur instantanée en sortie (norme au carré du vecteur sortie obtenue moins sortie désirée):

$$e = \frac{1}{2}(R - R_d)^2$$

On a:

$$\begin{aligned} \frac{\partial e}{\partial W_i} &= \frac{1}{2} \frac{\partial (R - R_d)^2}{\partial W_i} \\ &= (R - R_d) \frac{\partial (R - R_d)}{\partial W_i} \\ &= (R - R_d) c_i \end{aligned}$$

On obtient finalement la règle de renouvellement des poids après chaque passage d'exemple:

$$\Delta W_i = -\alpha(R - R_d)c_i$$

Remarquons que c_i vaut zéro pour tous les poids qui n'ont pas été activés lors de l'introduction de cette entrée. Il est donc tout-à-fait normal de ne modifier que les poids qui ont joué un rôle dans le calcul de la sortie correspondante.

La simplicité de la règle de renouvellement des poids laisse à penser que l'apprentissage du CMAC est extrêmement rapide.

4.4 Conclusion

Le CMAC (pour Cerebellar Model Articulation Controller) d'Albus se présente, après étude, sous les traits d'un tableau d'adressage judicieusement simplifié qui permet, à toute entrée donnée, d'associer une ou plusieurs somme(s) de C poids, créant ainsi ses sorties correspondantes. Les poids sont ajustables par rétropropagation suivant la règle du gradient.

L'intérêt principal du CMAC est qu'il devrait présenter de bonnes perfomances en robotique, puisqu'il a été imaginé à partir du cervelet, l'organe de contrôle des mouvements chez l'homme. Notons que cette ressemblance va jusqu'à permettre au CMAC de conserver la notion de voisinage. Le second intérêt de ce réseau est sa grande simplicité et sa grande rapidité d'apprentissage.

Enfin, malgré l'emploi (au niveau informatique) d'une fonction de hachage avec collisions, il semble que la place mémoire occupée par les poids stockés dans le CMAC devienne relativement importante lorsque le nombre d'entrées et de sorties croît.

Annexe A: Câblage d'un CMAC à plusieurs sorties

Partons d'un CMAC à une seule sortie tel que celui présenté en Section (4.2.2.1.). Un vecteur d'entrée présenté au réseau active C Détecteurs d'Espace d'Etats, qui à leur tour activent C Détecteurs de Champs Multiples. On a dit qu'il y a bijection entre les Détecteurs de Champs Multiples et les poids de sortie. A chaque Détecteur de Champs Multiples est donc associé un unique poids. Les C poids qui sont activés sont alors sommés pour former la réponse du CMAC. Ici la réponse du CMAC est un scalaire, puisqu'il n'y a qu'une seule sortie.

Plaçons-nous maintenant dans un cas à plusieurs sorties: toute la première partie de la structure du CMAC est absolument identique à celle que nous venons de décrire. Ce n'est qu'au niveau de la couche de poids qu'un changement est visible: au lieu de faire pointer la sortie de chaque Détecteur de Champs Multiples sur un seul poids, on la fait pointer sur une *liste* de r poids (s'il y a r sorties), comme indiqué sur la figure (16).

Une fois que C Détecteurs de Champs Multiples sont sélectionnés, nous possédons

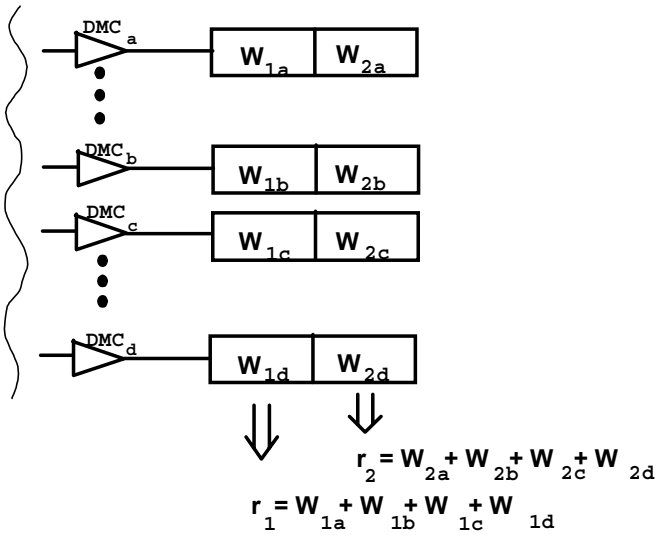

Figure 16: *Illustration de la généralisation à plusieurs sorties. Ici* $C = 4$ *et il y a* $r = 2$ *sorties*

alors C listes ordonnées de r poids, soit, après sommation, r sorties correspondant chacune à la somme de C poids. On voit donc que la généralisation à plusieurs sorties s'effectue de manière extrêmement simple.

References

[1] D.H. Ackley, G.E.Hinton & T.J. Sjenowski, "A Learning Algorithm for Boltzmann Machine", Cognitive Science, vol 9, pp 147-169, 1985

[2] J.S. Albus, "A Theory of Cerebellar Function", Mathematical Biosciences, vol 10, pp 25-61, 1971

[3] J.S. Albus, "A Robot Conditioned Reflex System Modelled after the Cerebellum", Proceedings Fall Joint Computer Conference, vol XLI, pp 1095-1104, 1972

[4] J.S. Albus, "Theoretical and Experimental Aspects of a Cerebellar Model", Ph.D. Thesis, University of Maryland, Dec 1972

[5] J.S. Albus, "A New Approach to Manipulator Control: Cerebellar Model Articulation Controller (CMAC)" Transactions of the ASME, Journal of Dynamic Systems, Measurements and Control, pp 220-227, Sept 1975

[6] G. Burel, "Réseaux de Neurones en Traitement d'Images: des Modèles Théoriques aux Applications Industrielles", Thèse de Doctorat de l'Université de Bretagne Occidentale, Brest, 1991

[7] E. Goles, F. Fogelman & D. Pellegrin, "Decreasing Energy Functions as a Tool for Studying Threshold Networks", Discrete and Applied Mathematics, vol 12, pp 261-277, 1985

[8] E. Goles & G.Y. Vichniac, "Lyapunov Function for Parallel Neural Networks", Proceedings of American Institute of Physics Conference on Neural Networks for Computing, Vol 151, pp 165-180, 1986

[9] S.P. Grossman, "The Motor System and Mechanics of Basic Sensory-Motor Integration", Textbook of Physiological Psychology, Chapitre 4, Wiley, New York, 1967

[10] J.J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities", Proc. Natl. Acad. Science, USA, Vol 79, pp. 2554-2558, April 1982

[11] J.J. Hopfield & D.W. Tank, "Neural Computation of Decisions in Optimisation Problems", Biological Cybernetics, vol 52, pp 141-152, 1985

[12] J.J. Hopfield & D.W. Tank, "Simple Neural Optimisation Networks: an A/D converter, Signal Decision Circuit and a Linear Programming Circuit", IEEE Trans. on Circuits and Systems, vol 33, pp 533-541, 1986

[13] D.O. Hebb, "The Organization of Behavior", J. Wiley and Sons, 1949

[14] S. Kirkpatrick, C.D. Gelatt, & M.P. Vecchi, "Optimisation by Simulated Annealing", Science, vol 220, pp 671-680, 1983

[15] D. Knuth, "Sorting and Searching", The Art of Computer Programming, vol 3, Addison Wesley, Menlo Park, California, USA, 1973

[16] A. M. Lyapunov, "Problème général de la stabilité du mouvement", édition originale en Russe en 1892, traduction du Russe en Français par E. Davaux, Annales de la faculté des Sciences de Toulouse, série 2, vol 9, 1907; reproduction en Français dans "Annals of Mathematics Studies", Princeton University Press, 1949

[17] M. Minsky & S. Papert, "Perceptrons, an Introduction to Computational Geometry", MIT Press, Cambridge, 1969

[18] N. J. Nilsson, "Learning Machines", Mc Graw Hill, 1965

[19] F. Rosenblatt, "Principles of Neurodynamics", Spartan, New York, 1962

[20] D.E. Rumelhart & J. L. Mc Clelland, "Parallel Distributed Processing", Bradford Book, MIT Press, 1986

[21] G. Widrow & M.E. Hoff, "Adaptive Switching Circuits", Institute of Radio Engineers, Western Electronic Show and Convention, part 4, pp 96-104, 1960

Chapitre 2
Estimation d'Angles d'Arrivée sur un Réseau de Capteurs

Depuis plus de trente ans, le problème de l'estimation d'Angles d'Arrivée (AdA) sur un réseau de capteurs intéresse la communauté scientifique spécialisée dans le traitement du signal. Pouvant s'appliquer à la radiogoniométrie, aux systèmes de localisation basés sur l'utilisation de radars ou de sonars, à l'astronomie et à la géophysique (localisation de gisements de pétrole, de nappes phréatiques , ...), l'estimation d'Angles d'Arrivée connaît aujourd'hui encore un engouement certain.

Toutes les méthodes se basent sur le fait que lorsqu'une onde plane bande étroite atteint successivement une série de capteurs alignés, sa phase augmente d'un facteur proportionnel à son angle d'arrivée sur cette ligne de capteurs. Diverses méthodes ont été mises au point, tout d'abord orientées vers une modélisation AR, MA, et ARMA, jusqu'à la découverte de Pisarenko, en 1973, qu'une décomposition harmonique de la matrice de covariance des signaux conduit à l'estimation des paramètres recherchés.

La première partie de ce Chapitre est consacrée à un historique du problème, et la seconde partie à une présentation détaillée du modèle envisagé. C'est la connaissance d'une certaine information géométrique caractérisant l'antenne qui permet d'estimer les Angles d'Arrivée.

Dans une troisième partie, des travaux basés sur des algorithmes de Haute-Résolution (MUSIC, ESPRIT) sont présentés. Ces algorithmes utilisent des décompositions en valeurs propres de la matrice de covariance des signaux. Toutefois, ces méthodes dites de Haute-Résolution sont sous-optimales: pour un bruit gaussien blanc, seule la méthode du Maximum de Vraisemblance (MV), qui maximise un critère de probabilité sur l'espace multidimensionnel des paramètres, est optimale. En pratique cette méthode est rarement employée, parce que sa mise en oeuvre est trop longue et trop coûteuse. D'autres travaux plus récents, basés sur l'utilisation de techniques neuronales, sont aussi présentés. Ils ne seront pas retenus car une étude théorique de leurs fondements montre des lacunes évidentes dans les critères qu'ils minimisent.

La quatrième partie propose une nouvelle approche neuronale utilisant un Perceptron à trois couches baptisé PRI (Perceptron Réel d'Initialisation). Nous proposons d'utiliser des données d'entrée issues de la matrice de covariance que nous "normalisons" de façon à rendre le réseau plus robuste.

Dans la cinquième partie, sans perdre de vue l'avantage fondamental de la méthode du MV, un réseau de neurones à valeurs complexes procédant par descente de gradient, est construit pour résoudre le problème. Relevant d'une structure contrainte totalement adaptée au problème d'AdA, ce réseau minimise un critère d'erreur théoriquement équivalent au critère du Maximum de Vraisemblance. De plus, les minima locaux sont évités par l'initialisation préalable des poids dans une région proche de la solution, grâce au PRI qui fournit une première estimation des paramètres.

La sixième partie s'intéresse au deuxième volet du traitement d'antennes: au lieu de chercher à estimer les Angles d'Arrivée des signaux que l'on reçoit (grâce à de l'information géométrique), on essaie maintenant de séparer ces signaux (grâce à de l'information statistique: indépendance des sources). Deux méthodes de Séparation Aveugle de Sources sont présentées, dont l'une est basée sur la minimisation d'un critère de dépendance par descente de gradient.

La septième partie présente une approche de fusion vis-à-vis des deux problèmes. Lorsque l'on dispose à la fois d'information géométrique et d'information statistique, on montre qu'il est possible, avec le Perceptron Complexe à Structure Contrainte (PCSC) introduit en Section 5, de minimiser de manière concomitante le critère du MV et un critère de dépendance des sources. Seule cette approche autorise l'estimation des angles de n sources sur une antenne de n capteurs. Cette approche fournit également une estimation des sources originales.

Enfin, la huitième partie est consacrée à une comparaison de nos résultats avec ceux des approches antérieures. On présente aussi les performances obtenues par l'approche de fusion Angles d'Arrivée/Séparation Aveugle de Sources. Une conclusion est finalement dressée en partie 9.

1 Historique

Comme souvent dans les découvertes humaines, les premières applications sont militaires, puis les besoins civils entraînent l'utilisation extensive de techniques qui entrent alors dans le domaine de la guerre économique.

Au début du siècle, c'est dans le cadre du développement des moyens de transport (en particulier aéronautiques et maritimes), que le besoin de détecter (afin de se protéger) des objets non coopératifs (exemple: des icebergs) pouvant entraver la trajectoire d'un véhicule rapide, a commencé à se faire sentir.

La radio-navigation pouvait bien sûr permettre la détermination de sa propre position par rapport à des points connus balisés à l'aide de dispositifs émetteurs. Cependant, dans le cas d'objets inconnus ou changeants (fonte des glaciers), le balisage ne pouvait plus être utilisé. Le RADAR (pour RAdio Detection And Ranging) a donc tout d'abord été développé dans le but de détecter des objets non coopératifs. En France, c'est sur le paquebot Normandie que fut installé, en 1935, le premier "radar", qui n'était alors qu'un simple dispositif anti-collision basé sur des mesures de variation de phase en présence d'obstacles. Dans le même temps et de manière indépendante, le radar fut également mis au point en Grande-Bretagne, en Allemagne et aux Etats-Unis.

Pendant la Seconde Guerre Mondiale (et dès 1939), les radars intégraient déjà la technique des impulsions, permettant ainsi non seulement de détecter un objet hostile, mais également de déterminer sa distance par soustraction des déphasages entre l'onde émise et l'onde réfléchie. Ainsi, avec trois radars placés à des points suffisamment éloignés, (cf figure (1)) on peut localiser l'objet par simple triangulation. Cependant, dès que la distance radar-cible devient trop grande par rapport aux distances séparant les radars entre eux, ce procédé devient peu précis. De plus, il ne peut pas fonctionner si l'on se trouve en présence de plusieurs cibles.

C'est donc pour déterminer plus finement la direction dans laquelle se trouve l'objet, que l'on a songé à utiliser la directivité des antennes. En effet, l'énergie électromagnétique diffusée par une antenne n'est pas répartie uniformément dans l'espace: elle est émise d'une manière bien spécifique qui définit le "diagramme de l'antenne". Un exemple de répartition angulaire de l'énergie diffusée par une antenne est illustré en figure (2). On voit que la majeure partie de l'énergie se trouve concentrée dans une direction privilégiée: c'est le lobe principal. L'angle dans lequel se concentre l'énergie est caractérisé par θ, l'ouverture en degrés à mi-puissance, aussi appelé ouverture du pinceau. Les autres secteurs de l'espace (ceux éclairés par les lobes secondaires) reçoivent beaucoup moins d'énergie; ainsi, une cible présente dans le faisceau principal reçoit (et donc réfléchit) une énergie beaucoup plus importante que les objets se trouvant hors de ce secteur, ce qui permet de localiser une cible

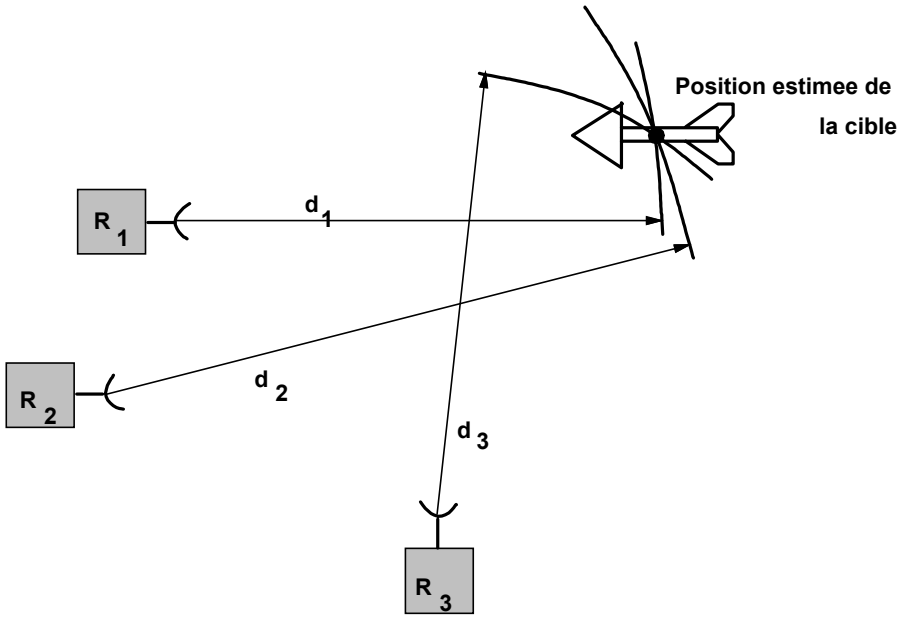

Figure 1: *Localisation d'une cible par triangulation en distance*

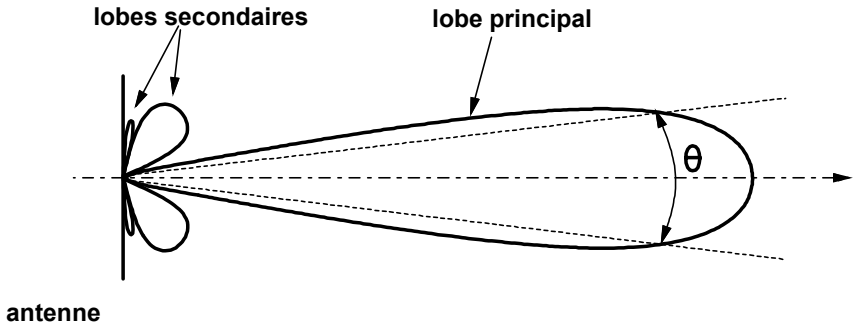

Figure 2: *Allure de la répartition angulaire de l'énergie électromagnétique diffusée par une antenne*

assez précisément, dans un pinceau de largeur θ.

Outre la précision en localisation apportée par une antenne directive par rapport à une antenne omnidirectionnelle, l'avantage de l'antenne directive est de pouvoir concentrer l'énergie dans une direction jugée intéressante: le gain en puissance est en

général très important, ce qui augmente considérablement la probabilité de détecter des objets éloignés ou peu réfléchissants.

Cependant, les limites de ce système, pourtant très performant, ne tardèrent pas à être mises en exergue: tout d'abord, il faut balayer tout l'espace avec le faisceau si l'on désire obtenir la localisation des sources dans toutes les directions. Il faut donc pour cela faire tourner l'antenne autour d'un axe de révolution. Or un système mobile étant forcément plus fragile qu'un système fixe, le mouvement de rotation oblige à construire des antennes très robustes, donc plus lourdes, et surtout plus coûteuses. Ensuite, il est clair qu'un système qui se doit de balayer tout l'espace de façon mécanique ne peut fournir un rafraîchissement rapide des images, car il lui faut décrire les 360^o d'espace avant de revenir à sa position initiale.

Les recherches se sont donc orientées vers un autre type de balayage de l'espace: le balayage électronique. Le dispositif d'émission se compose de plusieurs petites antennes disposées régulièrement en réseaux: lorsque des déphasages judicieux sont appliqués à ces émetteurs, la combinaison des ondes dans l'espace crée des lobes d'énergie comparables à ceux formés par une antenne classique, à la différence que c'est par calcul, et non par mouvement mécanique dans l'espace, que l'on oriente les lobes dans telle ou telle direction intéressante. Ainsi, les antennes (et les récepteurs) sont fixes, donc moins coûteux, et grâce aux progrès récents des calculateurs, on peut balayer l'espace entier beaucoup plus facilement et rapidement. On peut même choisir de ne balayer que des petites zones autour des régions qui nous intéressent, ce qui permet un rafraîchissement des cibles très fréquent.

Un autre défaut du radar classique, est que si l'antenne possède des lobes secondaires non négligeables par rapport au lobe principal, il peut arriver qu'un brouilleur (source de forte puissance) se trouvant dans une direction secondaire puisse gêner ou empêcher la détection d'une cible dans la direction principale, en masquant les signaux réfléchis par cette cible.

C'est précisément ici que les algorithmes estimateurs d'Angles d'Arrivée entrent en jeu: en plaçant en réception un réseau de capteurs espacés de manière judicieuse, il est possible de retrouver par le calcul la configuration des rayons des signaux qui arrivent (à la fois les véritables signaux réfléchis par les cibles, et les "faux signaux" émis par des brouilleurs). Un tel système, aussi appelé "antenne adaptative", se compose donc d'un réseau de plusieurs capteurs (la partie "antenne") recevant des ondes de plusieurs "sources", en sortie desquels se trouve tout un système algorithmique (la partie "adaptative") permettant de retrouver les Angles d'Arrivée.

Nous allons maintenant détailler le problème d'un point de vue plus mathématique.

2 Description du problème d'estimation des angles d'arrivée

2.1 Problème général

Considérons un réseau de m capteurs placés à des positions arbitraires, et possédant des caractéristiques directionnelles arbitraires. Soient n sources en bande étroite centrées autour de la même fréquence ν_0. C'est par exemple le cas lorqu'un système actif équipé d'un radar émet à une fréquence ν_0, et que l'on désire déterminer la provenance des signaux renvoyés (localiser les objets réféchissants, et, éventuellement, les brouilleurs).

Les angles d'arrivée des n sources sur le réseau de capteurs sont respectivement $\theta_1, ..., \theta_n$. Notons:

- $\rho_i(\theta_k)$ le gain du capteur i en réponse à une onde arrivant sous l'angle θ_k
- $\tau_i(\theta_k)$ le délai de propagation entre un point de référence et le $i-$ème capteur, pour une onde arrivant sous l'angle θ_k
- $b_i(t)$ l'enveloppe complexe du bruit au $i-$ème capteur
- $x_k(t)$ l'enveloppe complexe du signal émis par la $k-$ième source et reçu au point de référence
- $y_i(t)$ l'enveloppe complexe du signal reçu au $i-$ème capteur

Le signal envoyé par la $k-$ième source et reçu au $i-$ème capteur est:

$$Y_{ik}(t) = \mathrm{Re}\left\{x_k(t - \tau_i(\theta_k))\rho_i(\theta_k)e^{j2\pi\nu_0(t-\tau_i(\theta_k))}\right\} \tag{1}$$

L'hypothèse de signaux bande étroite autorise à dire que le signal varie lentement en fonction du temps, ou tout au moins que sa variation sur une distance correspondant à la dimension du réseau est négligeable: $x_k(t-\tau_i(\theta_k)) \simeq x_k(t)$. De ce fait, l'enveloppe complexe du signal reçu au $i-$ème capteur s'écrit:

$$y_i(t) = \sum_{k=1}^{n} \rho_i(\theta_k)e^{-j2\pi\nu_0\tau_i(\theta_k)}x_k(t) + b_i(t) \tag{2}$$

L'utilisation de la notation matricielle permet de séparer les parties dépendant des

angles d'arrivée, de celles qui n'en dépendent pas:

$$\begin{pmatrix} y_1 \\ . \\ . \\ . \\ . \\ . \\ y_m \end{pmatrix} = \begin{pmatrix} a_1(\theta_1) & ... & a_1(\theta_n) \\ . & & . \\ . & & . \\ . & & . \\ . & & . \\ . & & . \\ a_m(\theta_1) & ... & a_m(\theta_n) \end{pmatrix} \begin{pmatrix} x_1 \\ . \\ . \\ . \\ x_n \end{pmatrix} + \begin{pmatrix} b_1 \\ . \\ . \\ . \\ . \\ . \\ b_m \end{pmatrix} \tag{3}$$

où $a_i(\theta_k)$ est la réponse en gain du capteur i dans la direction θ_k, et tenant compte du délai de propagation par rapport au point de référence:

$$a_i(\theta_k) = \rho_i(\theta_k) e^{-j2\pi\nu_0\tau_i(\theta_k)} \tag{4}$$

Le vecteur $\boldsymbol{a}(\theta_k)$ tel que:

$$\boldsymbol{a}(\theta_k) = \begin{bmatrix} a_1(\theta_k) \\ \vdots \\ a_i(\theta_k) \\ \vdots \\ a_m(\theta_k) \end{bmatrix} \tag{5}$$

est appelé "vecteur directionnel" ("steering vector" en anglais), et la matrice

$$A(\Theta) = [\boldsymbol{a}(\theta_1), ..., \boldsymbol{a}(\theta_n)] \tag{6}$$

est appelée la "matrice directionnelle" (ou "steering matrix"). Ainsi, l'équation (3) devient, sous une forme plus compacte:

$$\boldsymbol{y}(t) = A(\Theta)\boldsymbol{x}(t) + \boldsymbol{b}(t) \tag{7}$$

où $\Theta = [\theta_1, ..., \theta_n]^T$ est le vecteur paramètre recherché.

Disposant de N observations $\boldsymbol{y}(1), ..., \boldsymbol{y}(N)$ recueillies à des instants différents, le problème de l'estimation d'Angles d'Arrivée se résume à l'estimation du vecteur Θ et (éventuellement) à l'estimation des sources $\boldsymbol{x}(1), ..., \boldsymbol{x}(N)$.

Dans la suite de l'exposé, nous supposerons, sauf lorsqu'il en est fait mention contraire, que les sources sont centrées. Par suite, les signaux reçus sur les capteurs sont également centrés[1]. De ce fait, les matrices de covariance et de corrélation sont

[1]En pratique, si cela n'était pas le cas, il suffirait de centrer les signaux capteurs.

confondues.

Note importante:
Il faut noter ici quatre hypothèses de travail indispensables à la construction d'un estimateur des angles d'arrivée.

- *Hypothèse 1*: Les n ondes arrivant sur l'antenne sont non complètement corrélées, c'est-à-dire que la matrice de corrélation des sources

$$\mathcal{S} = E\{\boldsymbol{x}\boldsymbol{x}^H\} \tag{8}$$

  est non singulière, de rang plein n. En réalité cette hypothèse n'est nécessaire que pour l'algorithme MUSIC, que nous étudierons par la suite; cela veut dire que pour comparer MUSIC à d'autres estimateurs, il faut respecter cette hypothèse.

- *Hypothèse 2*: Tout ensemble de n vecteurs directionnels construits à partir de paramètres d'arrivée θ_k tous différents, est de rang n. Sous forme contractée:

$$\text{si les } \theta_k \text{ sont tous différents, alors } \quad \text{rang}[\boldsymbol{a}(\theta_1), ..., \boldsymbol{a}(\theta_k), ..., \boldsymbol{a}(\theta_n)] = n \tag{9}$$

- *Hypothèse 3*: Le nombre de capteurs est strictement supérieur au nombre maximal d'angles à estimer:

$$m > n \tag{10}$$

  En effet, l'étude de [29] montre que cette condition est nécessaire pour tout estimateur d'Angle d'Arrivée utilisant une solution au second ordre. Nous verrons par la suite que pour des estimateurs utilisant des ordres supérieurs, cette condition n'est pas indispensable.

- *Hypothèse 4*: Sauf lorsqu'il en sera mentionné différemment, on considérera que le problème (annexe mais indispensable) de l'estimation du *nombre* d'angles d'arrivée est résolu, et nous noterons par extension n au lieu de $\hat{n}$ cette estimation. Pour plus de précisions sur des méthodes d'estimation du nombre d'angles d'arrivée (procédant par des calculs sur les valeurs propres de la matrice de covariance des signaux), voir [28] par exemple.

2.2 Problème simplifié du réseau linéaire uniforme

Les difficultés supplémentaires engendrées par l'utilisation de capteurs à réponse directionnelle non-uniforme (c'est-à-dire des capteurs dont le gain dépend lui aussi de l'angle d'arrivée), ont poussé les chercheurs à circonscrire le problème au maximum. Pour ce faire, les capteurs utilisés sont omnidirectionnels:

$$\forall k \in [1, n], \qquad \rho_i(\theta_k) = \text{cste} \tag{11}$$

et de plus ils sont tous identiques:

$$\forall i \in [1, m], \qquad \rho_i(\theta_k) = \text{cste} \tag{12}$$

Comme on le verra par la suite, bien des méthodes [19] peuvent ensuite être généralisées pour un problème avec des capteurs différents et directionnels, mais en contrepartie des dispositions spécifiques ou des informations supplémentaires sont nécessaires.

Ayant réduit les $\rho_i(\theta_k)$ à un facteur multiplicatif, on peut les supprimer de l'équation (4):

$$a_i(\theta_k) = e^{-j2\pi\nu_0\tau_i(\theta_k)} \tag{13}$$

Ainsi, pour un réseau dont les capteurs sont tous identiques et possèdent un gain directionnel uniforme, la dépendance en θ_k ne s'exprime désormais plus que sous une forme exponentielle, ce qui simplifie considérablement la représentation du modèle.

Enfin, lorsque le réseau est *linéaire*, c'est-à-dire lorsque les capteurs sont alignés, $\tau_i(\theta_k)$ se simplifie encore (voir figure (3)). Pour deux capteurs séparés par une distance d, une onde plane arrivant sur le premier capteur doit parcourir la distance supplémentaire $d \sin\theta_k$ avant d'atteindre le second capteur. Le temps supplémentaire pour parcourir cette distance est:

$$\delta\tau(\theta_k) = \frac{d \sin\theta_k}{c} \tag{14}$$

si l'onde se propage à une célérité c. Si tous les capteurs sont séparés par la même distance d, et que le premier capteur est pris pour point de référence, le délai entre ce premier capteur et le capteur i s'écrit:

$$\tau_i(\theta_k) = \frac{(i-1)d \sin\theta_k}{c} \tag{15}$$

L'équation (13) explicitant les composantes du vecteur directionnel $\boldsymbol{a}(\theta_k)$ se simplifie:

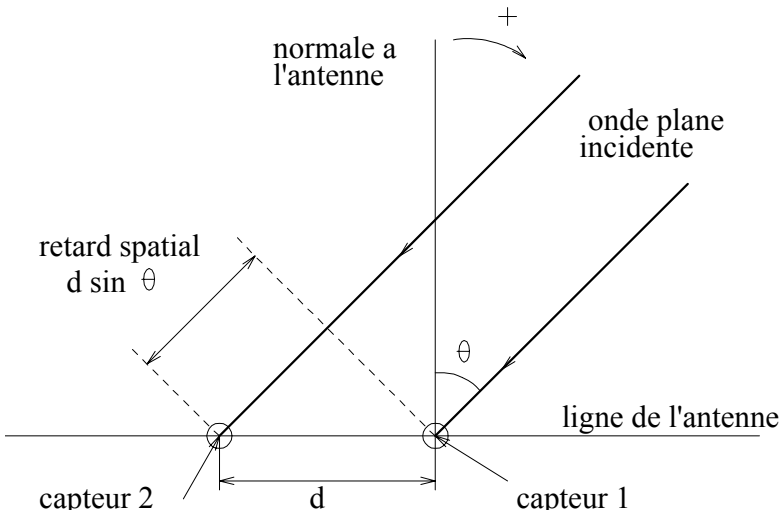

Figure 3: *Cas du réseau linéaire uniforme*

$$a_i(\theta_k) = e^{-j(i-1)\omega_k} \tag{16}$$

où

$$\begin{aligned} \omega_k &= \frac{2\pi d\nu_0}{c}\sin\theta_k \\ &= \frac{2\pi d}{\lambda}\sin\theta_k \end{aligned} \tag{17}$$

si λ est la longueur d'onde correspondant à la fréquence ν_0.

Le vecteur directionnel correspondant à la direction d'arrivée θ_k devient:

$$\boldsymbol{a}(\theta_k) = \begin{bmatrix} 1 \\ e^{-j\omega_k} \\ \vdots \\ e^{-j(i-1)\omega_k} \\ \vdots \\ e^{-j(m-1)\omega_k} \end{bmatrix} \tag{18}$$

Comme on le constate, la véritable variable est maintenant ω_k. On notera donc désormais $\boldsymbol{a}(\omega_k)$ le vecteur directionnel relatif à l'onde k, $\Omega = [\omega_1, ..., \omega_n]^T$ le vecteur paramètre recherché, et $A(\Omega)$ la matrice directionnelle:

$$A(\Omega) = [\boldsymbol{a}(\omega_1), ..., \boldsymbol{a}(\omega_n)] \tag{19}$$

Pour que la relation (17) qui lie ω_k à θ_k soit bien bijective, une remarque doit être faite: le caractère périodique de la fonction exponentielle complexe montre, d'après l'équation (16), que le paramètre ω_k est défini modulo 2π. Ainsi, pour éviter tout phénomène d'aliasing, la distance intercapteur doit vérifier:

$$\begin{aligned} \frac{2\pi d}{\lambda}|\sin\theta_k| &\leq \pi \\ \frac{2d}{\lambda} &\leq 1 \\ d &\leq \frac{\lambda}{2} \end{aligned} \tag{20}$$

Lorsque le problème traité relève effectivement d'un système actif (c'est-à-dire un problème où la fréquence porteuse ν_0 est au préalable émise par la partie active du système), le choix de la distance intercapteur ne se pose pas: à la limite du théorème de Shannon, et pour un maximum d'ouverture, il est judicieux de choisir $d = \frac{\lambda}{2}$. De ce fait l'équation (17) se simplifie considérablement:

$$\omega_k = \pi \sin\theta_k \tag{21}$$

Si au contraire on se trouve dans un cas où la fréquence était inconnue avant la mise en oeuvre de l'antenne, il faut s'assurer, dès que l'on connaît cette fréquence, qu'elle vérifie bien l'équation (20).

3 Travaux antérieurs

3.1 Algorithmes de Haute Résolution

Les premiers modèles pour poser le problème considéraient celui-ci sous l'angle "filtrage temporel", utilisant le fait que le signal reçu sur un capteur était intuitivement très proche du signal reçu sur le capteur précédent, à un décalage temporel près. Les méthodes de résolution du problème posé sous cette forme font donc appel à des solutions de type AR, MA, ou mieux, ARMA (voir [6], [5]).

Par la suite, Pisarenko a découvert que la matrice de covariance des signaux possédait des propriétés puissantes, permettant la mise en évidence d'une décomposition harmonique correspondant justement aux Angles d'Arrivée recherchés. La méthode

de Pisarenko fait entre autres appel à une décomposition en vecteurs propres, et à l'époque, on s'aperçut que ses performances étaient largement supérieures à celles des autres méthodes, d'où l'expression "algorithme de Haute Résolution".

On appelle désormais par extension "algorithme de Haute Résolution", toute méthode faisant appel à une décomposition en vecteurs propres de la matrice de covariance des signaux capteurs, telle MUSIC et ESPRIT, qui elles-mêmes surpassent aujourd'hui la méthode de Pisarenko.

3.1.1 MUSIC

Introduction
L'avènement de l'algorithme MUSIC (pour MUltiple SIgnal Classification [20], [21]) se base avant tout sur une étude théorique préalable qui suit une très grande logique.

Reprenons l'équation de base du problème:

$$\boldsymbol{y}(t) = A(\Theta)\boldsymbol{x}(t) + \boldsymbol{b}(t) \tag{22}$$

On supposera les signaux centrés. La matrice de covariance des signaux reçus sur l'antenne s'écrit:

$$\begin{aligned} R_y = E\{\boldsymbol{y}\boldsymbol{y}^H\} &= AE\{\boldsymbol{x}\boldsymbol{x}^H\}A^H + \sigma^2 I \\ &= AR_xA^H + \sigma^2 I \end{aligned} \tag{23}$$

où R_x est la matrice $n \times n$ de covariance des sources.

Soient $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_m$ les valeurs propres de la matrice R_y, et soient $\nu_1 \geq \nu_2 \geq ... \geq \nu_m$ celles de la matrice AR_xA^H. On peut alors écrire:

$$\forall i \in [1, m] \quad \lambda_i = \nu_i + \sigma^2 \tag{24}$$

Si les sources sont indépendantes les unes des autres,

$$R_x = diag(P_1, P_2, ..., P_n) \tag{25}$$

où $P_i = E\{x_i x_i^H\}$ est la puissance moyenne de la $i-$ème source. Si au contraire les sources ne sont pas indépendantes, R_x peut ne plus être diagonale. Cependant, l'hypothèse (8) que les sources sont non complètement corrélées assure que

$$rang\ R_x\ =\ dim\ R_x\ = n \tag{26}$$

De plus, d'après l'hypothèse (9), le rang de la matrice rectangulaire $(m \times n)$ A est n $(n < m)$. On en conclut que la matrice AR_xA^H possède n valeurs propres non nulles, et $m - n$ valeurs propres nulles. Ainsi:

$$\forall i > n \quad \nu_i = 0 \tag{27}$$

De ce fait, la plus petite valeur propre de R_y est donc σ^2, avec la multiplicité $m - n$. L'équation (24) se récrit alors:

$$\lambda_i = \begin{cases} \nu_i + \sigma^2 & i \in [1, n] \\ \sigma^2 & i \in [n+1, m] \end{cases} \tag{28}$$

La figure (4) montre un exemple de spectre $\lambda_i = f(i)$ pour une matrice de corrélation R_y calculée sur une antenne de $m = 10$ capteurs recevant $n = 4$ signaux.

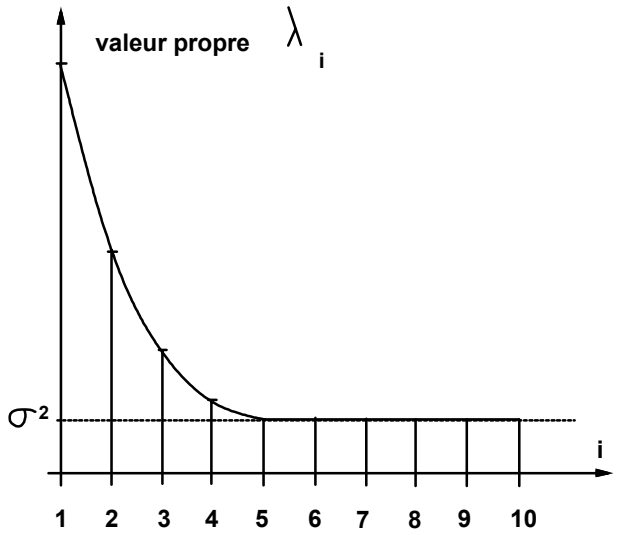

Figure 4: *Un exemple de spectre pour une matrice de corrélation (ici $m = 10$ et $n = 4$)*

Fondements théoriques de MUSIC
D'après la décomposition en valeurs singulières évoquée au paragraphe précédent, les vecteurs propres $\boldsymbol{q}_i,\ i \in [n+1, m]$ de la matrice R_y sont associés à la valeur propre σ^2. On a donc

$$R_y \boldsymbol{q}_i = \sigma^2 \boldsymbol{q}_i, \quad i \in [n+1, m] \tag{29}$$

soit

$$\left(R_y - \sigma^2 I\right) \boldsymbol{q}_i = 0, \quad i \in [n+1, m] \tag{30}$$

ce qui, d'après (23), équivaut à

$$A R_x A^H \boldsymbol{q}_i = 0, \quad i \in [n+1, m] \tag{31}$$

La matrice A est rectangulaire de rang plein égal au nombre de ses colonnes, et R_x est carrée de rang plein. L'équation précédente est donc équivalente à

$$A^H \boldsymbol{q}_i = 0, \quad i \in [n+1, m] \tag{32}$$

Soit, en détaillant A^H:

$$\boldsymbol{a}^H(\omega_k) \boldsymbol{q}_i = 0 \quad \begin{cases} i \in [n+1, m] \\ k \in [1, n] \end{cases} \tag{33}$$

où $\boldsymbol{a}^H(\omega_k)$ est la $k-$ième ligne de la matrice A^H.

Conséquences:

Les vecteurs propres ont la propriété d'être orthogonaux entre eux. En particulier, les $\boldsymbol{q}_i, i \in [n+1, m]$ sont indépendants. Ainsi le sous-espace engendré par ces vecteurs, $Vect\{\boldsymbol{q}_{n+1}, ..., \boldsymbol{q}_m\}$ est un sous-espace de dimension $m - n$. On l'appelle le "sous-espace bruit" car il est construit à partir des vecteurs propres associés aux valeurs propres de R_y égales à la variance du bruit σ^2. L'équation (33) indique que chaque vecteur directionnel $\boldsymbol{a}^H(\omega_k)$ est orthogonal au sous-espace bruit. Comme en outre les $\boldsymbol{a}^H(\omega_k)$ sont indépendants (hypothèse (9)), alors ils forment un sous-espace

de dimension n, orthogonal au sous-espace bruit (de dimension $m-n$). Ces deux sous-espaces forment donc une partition de l'espace considéré, de dimension m.

Il est à noter, puisque tous les vecteurs propres sont orthogonaux, que les n vecteurs propres $\boldsymbol{q}_1, ..., \boldsymbol{q}_n$ associés aux n plus grandes valeurs propres de R, forment un sous-espace (de dimension n) orthogonal au sous-espace propre de bruit. Ils définissent donc le même sous-espace que les vecteurs directionnels $\boldsymbol{a}(\omega_k)$ correspondant aux signaux reçus. De ce fait le sous-espace associé aux n plus grandes valeurs propres est appelé "sous-espace signal", ou plus justement "sous-espace signal+bruit", puisque ses valeurs propres comportent également la variance du bruit d'après l'équation(28).

Finalement, la décomposition en vecteurs propres de la matrice $m \times m$ de corrélation R_y suggère les deux remarques suivantes:

1. L'espace engendré par les vecteurs propres de R_y se décompose en deux sous-espaces disjoints (qui forment partition): un sous-espace appelé "signal plus bruit", qui est engendré par les vecteurs propres associés aux n plus grandes valeurs propres de R_y; et un sous-espace appelé "bruit", qui est engendré par les vecteurs propres associés aux $m-n$ plus petites valeurs propres de R. Ces deux sous-espaces sont orthogonaux entre eux.

2. Le sous-espace "signal plus bruit" est également généré par les n vecteurs directionnels $\boldsymbol{a}(\omega_k)$.

La remarque (2) suggère qu'*il est possible d'estimer les paramètres ω_k du problème, en cherchant les n valeurs de ω qui rendent le vecteur directionnel $\boldsymbol{a}(\omega)$ orthogonal au sous-espace de bruit.* C'est sur cette simple idée que se base l'algorithme de Schmidt [20], [21]: le vecteur $\boldsymbol{a}(\omega) = [1, e^{-j\omega}, ..., e^{-j(m-1)\omega}]^T$ est orthogonal au sous-espace de bruit si, et seulement si, sa projection orthogonale sur ce sous-espace est nulle.

En pratique on ne dispose pas de la matrice de corrélation exacte $R_y = E\{\boldsymbol{y}\boldsymbol{y}^H\}$, mais de son estimation sur l'échantillon limité d'observations dont on dispose. De ce fait, la décomposition en valeurs propres de R_y est entachée d'erreurs, et les vecteurs propres associés aux plus grandes valeurs propres ne génèrent plus exactement le même sous-espace que les vecteurs directionnels. Les relations d'orthogonalité (33) desquelles découle la remarque (2) ne sont donc plus vérifiées. Par suite, au lieu de chercher à annuler la projection de $\boldsymbol{a}(\omega)$ sur le sous-espace bruit, on se contentera de minimiser celle-ci.

Construction de l'algorithme

D'après les considérations précédentes, l'algorithme MUSIC se construit de la façon suivante. Soient $\boldsymbol{y}_t,\ t \in [1, N]$ les observations en sortie d'antenne à divers instants t quelconques. On construit une estimation de la matrice de corrélation des signaux

$$\hat{R}_y = \frac{1}{N} \sum_{t=1}^{N} \boldsymbol{y}_t \boldsymbol{y}_t{}^H \tag{34}$$

De la décomposition en valeurs propres $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_m$ de la matrice $\hat{R}_y$, on ne garde que les vecteurs propres associés aux plus petites valeurs propres: $(\boldsymbol{q}_i, \lambda_i),\ i \in [n+1, m]$. Ces vecteurs propres sont appelés "vecteurs propres de bruit", et la juxtaposition de leurs colonnes forme la matrice de bruit:

$$V_b = \left[\boldsymbol{q}_{n+1}, ..., \boldsymbol{q}_m\right] \tag{35}$$

Le carré de la norme de la projection du vecteur $\boldsymbol{a}(\omega)$ sur le sous-espace engendré par les $\boldsymbol{q}_i,\ i \in [n+1, m]$ s'écrit:

$$\begin{aligned} \left[\mathrm{Proj}_{V_b}\left(\boldsymbol{a}(\omega)\right)\right]^2 &= \sum_{i=n+1}^{m} \left|\boldsymbol{a}^H(\omega)\boldsymbol{q}_i\right|^2 \\ &= \boldsymbol{a}^H(\omega) V_b V_b^H \boldsymbol{a}(\omega) \end{aligned} \tag{36}$$

La fonction à minimiser pour trouver les paramètres du problème est

$$f(\omega) = \boldsymbol{a}^H(\omega) V_b V_b^H \boldsymbol{a}(\omega) \tag{37}$$

d'après l'équation (36). En pratique, au lieu de chercher les minima à zéro de cette fonction, il est plus intéressant de chercher les maxima (qui tendent par conséquent vers l'infini) de son inverse: les contrastes sont de cette façon beaucoup plus marqués (voir figure (5), et mettent en valeur ce que l'on connaît comme les "pics" de l'algorithme MUSIC:

$$\{\hat{\omega}_k; k \in [1, n]\} = \{\omega \in [-\pi, \pi];\ \omega = \mathrm{Argmax} \frac{1}{f(\omega)}\} \tag{38}$$

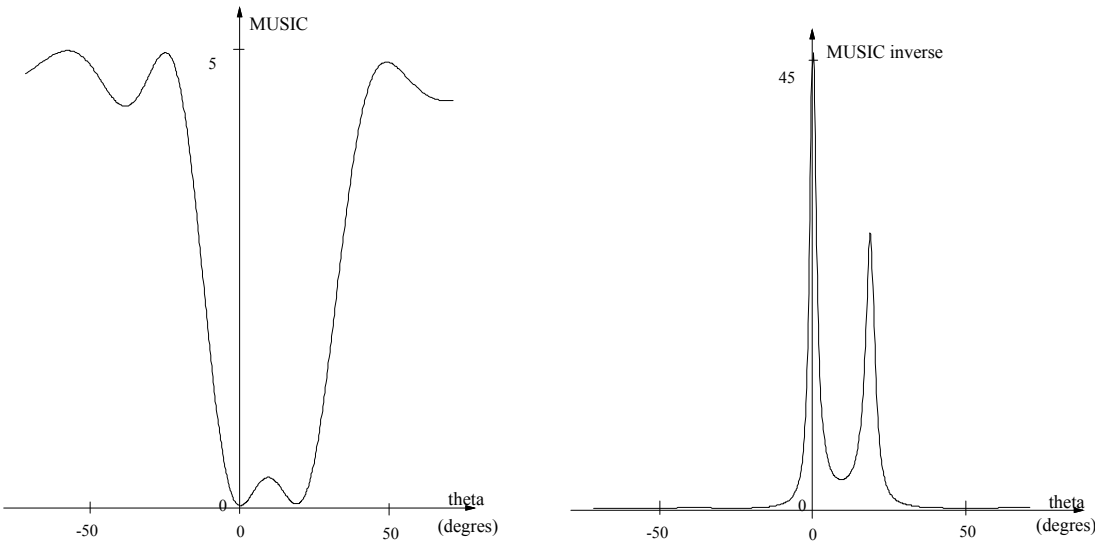

Figure 5: *Comparaison des minima de $f(\omega)$ et des maxima de $f^{-1}(\omega)$*

Résumé de l'algorithme MUSIC

1. Estimer la matrice de covariance

$$\hat{R}_y = \frac{1}{N}\sum_{t=1}^{N} \boldsymbol{y}_t \boldsymbol{y}_t{}^H \tag{39}$$

2. Décomposer $\hat{R}_y$ en valeurs propres:

$$\hat{R}_y = Q\Lambda Q^H \tag{40}$$

où $\Lambda = diag[\lambda_1, ..., \lambda_m]$ dans laquelle les λ_i sont rangés par ordre décroissant, et $Q = [\boldsymbol{q}_1, ..., \boldsymbol{q}_m]$

3. La matrice de bruit est:

$$V_b = [\boldsymbol{q}_{n+1}, ..., \boldsymbol{q}_m] \tag{41}$$

4. Localiser les n pics $[\hat{\omega}_1, ..., \hat{\omega}_n]$ de

$$S_{MUSIC}(\omega) = \frac{1}{\boldsymbol{a}^H(\omega) V_b V_b^H \boldsymbol{a}(\omega)} \tag{42}$$

où $\boldsymbol{a}(\omega) = [1, e^{-j\omega}, ..., e^{-j(m-1)\omega}]^T$ pour un réseau linéaire uniforme.

5. Obtenir les n angles d'arrivée par l'inversion de l'équation (17):

$$\hat{\theta}_k = \arcsin\left(\frac{\lambda\hat{\omega}_k}{2\pi d}\right) \quad k \in [1, n] \tag{43}$$

Avantages et inconvénients de l'algorithme MUSIC
Le grand avantage de l'algorithme MUSIC réside dans la grande logique et la grande simplicité de sa base théorique: en bref il s'agit de déterminer les directions d'arrivée en retrouvant tous les vecteurs directionnels qui appartiennent au sous-espace signal, c'est-à-dire ceux dont la projection sur l'espace bruit est minimale.
De plus, les notions mathématiques auxquelles cet algorithme fait appel sont très bien connues:

- estimation d'une matrice de corrélation sur un échantillonnage fini
- mise en œuvre d'une décomposition en valeurs propres et vecteurs propres
- calcul de la projection orthogonale d'un vecteur sur un sous-espace
- estimation des maxima d'une fonction réelle à une seule variable

Ces qualités, qui en font par exemple un algorithme idéal à présenter à des étudiants novices dans le domaine des techniques dites de "Haute Résolution", lui valent d'être certainement le mieux connu et le plus étudié de par le monde.
En outre, comme on l'a vu dans l'étude précédente, l'algorithme MUSIC se comporte de manière parfaite en théorie, c'est-à-dire lorsque la matrice de covariance $\hat{R}_y$ est exacte, ce qui revient à considérer le nombre d'échantillons N comme infini. Pour N et m grands, et si de surcroît les sources sont indépendantes, on peut montrer [24]

que l'estimateur MUSIC tend vers les bornes de Cramer-Rao.

Par contre, dès que (comme c'est le cas dans la réalité) le nombre d'observations N est fini, les performances de MUSIC se dégradent, et ce phénomène s'accentue encore lorsque les sources sont corrélées. Un comportement assez connu de la fonction $S_{MUSIC}(\omega)$ illustre assez bien certaines failles de MUSIC. Soient deux sources non corrélées dont il s'agit d'estimer les angles d'arrivée. Plaçons-nous dans un cas où peu d'échantillons sont disponibles ($N = 10$ par exemple), et où le réseau n'est composé que de $m = 5$ capteurs. La figure (6) montre le tracé de la fonction $S_{MUSIC}(\omega)$ pour divers écartements entre ces angles d'arrivée.

On le voit, pour des angles d'arrivée trop proches, les pics de MUSIC fusionnent, et

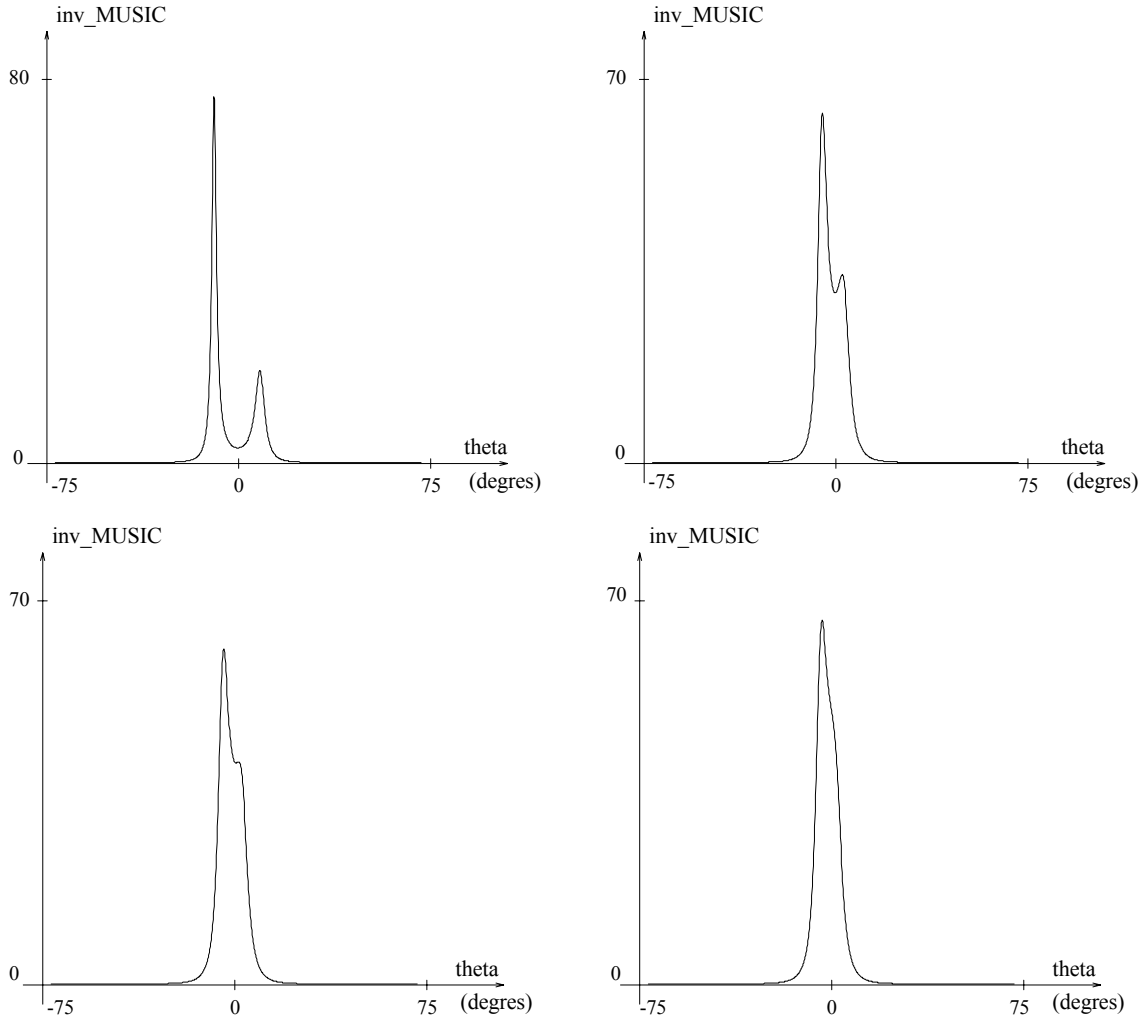

Figure 6: *Illustration du phénomène de fusion entre les pics de MUSIC lorsque l'écart entre les Angles d'Arrivée diminue (diminution de gauche à droite et de haut en bas)*

l'algorithme ne distingue alors plus qu'une seule direction d'arrivée, située en général entre les deux valeurs vraies: cela occasionne un certain biais dans l'estimation

fournie par MUSIC. De plus, cela laisse croire qu'un seul objet émet ou renvoie des signaux, ce qui peut s'avérer gênant pour certaines applications.

On peut essayer de comprendre ce qui, physiquement, fait fusionner les deux pics d'angles proches. Lorsque deux angles θ_1 et θ_2 sont proches, les retards de phase ω_1 et ω_2 qu'elles entraînent sur l'antenne sont également proches. De même, les vecteurs directionnels $\boldsymbol{a}(\omega_1)$ et $\boldsymbol{a}(\omega_2)$ sont très proches. Le sous-espace de dimension 2 qu'ils définissent devient donc très instable, puisqu'un changement minime de ω_1 ou ω_2 crée un effet de levier important sur le sous-espace (cf figure (7)).

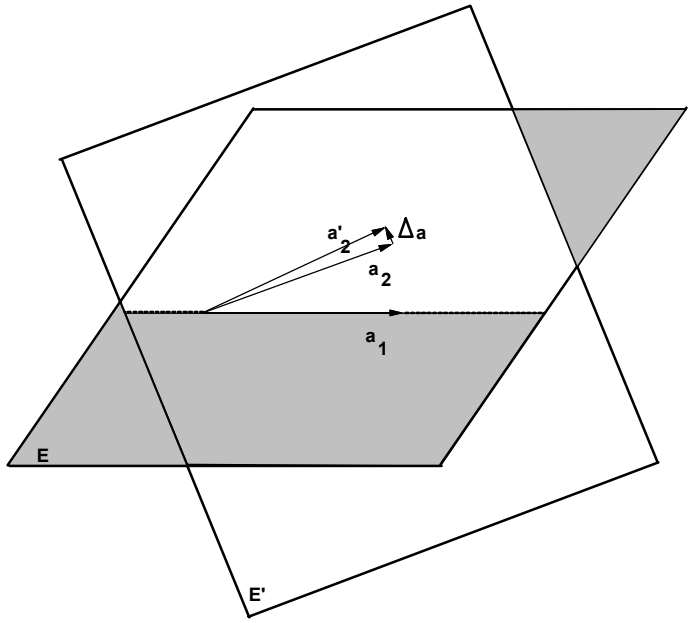

Figure 7: *Illustration de l'effet "levier" sur le sous-espace signal*

A l'inverse, comme la matrice $\hat{R}_y$ n'est qu'une estimation de la vraie matrice de covariance R_y, les données comportent des erreurs dès le départ. En outre la décomposition en valeurs propres d'une matrice complexe se fait numériquement par itérations. Or celles-ci ne sont arrêtées que lorsqu'un certain critère d'erreur passe *sous* un seuil fixé d'avance. Ce seuil est un seuil générique, et, s'il convient pour la grande majorité des décompositions en valeurs singulières, il est peut-être trop fort dans les cas d'angles très proches. D'un autre côté, plus on baisse le seuil, plus les calculs nécessaires à la décomposition en valeurs propres s'allongent. Un compromis est indispensable. Il conduit cependant à des erreurs qui s'ajoutent à celles dûes au nombre limité d'échantillons. Pour deux angles très proches, la première valeur propre risque de devenir très grande devant toutes les autres, et son vecteur propre prendre à peu près la direction moyenne $\boldsymbol{a}(\omega_1) + \boldsymbol{a}(\omega_2)$. La seconde valeur propre, elle, prendra alors la direction orthogonale à la première (on rappelle que les vecteurs propres ont la propriété d'être tous orthogonaux), et contenue dans le sous-espace

$Vect\{\boldsymbol{a}(\omega_1), \boldsymbol{a}(\omega_2)\}$, c'est-à-dire la direction $\boldsymbol{a}(\omega_1) - \boldsymbol{a}(\omega_2)$, puisque les vecteurs directionnels sont de normes égales. Cette direction, au contraire, est très instable, et étant de plus orthogonale à la direction privilégiée, sa valeur propre associée est faible. De ce fait, il est à craindre que le calcul du second vecteur propre soit assez sensible aux erreurs mentionnées précédemment. On retrouve alors l'effet "bras de levier" évoqué au départ, et qui explique le mauvais comportement de la méthode lorsque les angles sont proches.

3.1.2 ESPRIT

Introduction

Comme le soulignent Kailath et Roy [19], si MUSIC a été le premier algorithme de Haute Résolution à exploiter avec succès le modèle sous-jacent des signaux bande étroite noyés dans un bruit additif, la méthode recèle certaines limitations, comme par exemple le fait que la recherche des maxima doit se faire sur l'espace entier des paramètres, ce qui est long et coûteux.

D'autre part, MUSIC nécessite la maîtrise exacte de toutes les caractéristiques du réseau récepteur (gain directionnel de chaque récepteur, alignement des capteurs, distance égale entre les capteurs successifs), ce qui est parfois difficile à réaliser. L'intérêt de la méthode de Haute Résolution ESPRIT (pour Estimation of Signal Parameters via Rotational Invariance), est qu'elle nécessite beaucoup moins de contraintes physiques quant aux caractéristiques de l'antenne (bien qu'une antenne linéaire uniforme convienne aussi pour ESPRIT).

La seule contrainte imposée à l'antenne est qu'elle soit de géométrie plane, décomposable en m paires de capteurs (soit $2m$ capteurs en tout), comme le montre la figure (8). Les éléments d'un même doublet doivent avoir le même diagramme directionnel, et être séparés par une translation fixe $\vec{\Delta}$. Par contre, aucune contrainte n'est imposée entre doublets, si ce n'est qu'ils soient coplanaires. On peut donc décrire ce réseau de $2m$ capteurs comme la juxtaposition de deux sous-réseaux identiques de m capteurs, séparés d'un vecteur de translation $\vec{\Delta}$.

On voit que les contraintes imposées à l'antenne sont bien moindres que celles exigées par MUSIC. En particulier, les capteurs à l'intérieur d'un sous-réseau ne sont pas nécessairement identiques, et la connaissance de leur caractéristiques directionnelles n'est pas demandée. En outre, leur disposition géométrique peut rester inconnue et arbitraire, tant qu'ils restent coplanaires et que le second sous-réseau est parfaitement identique au premier, à une translation $\vec{\Delta}$ près. La calibration d'un tel réseau est donc beaucoup plus aisée et moins coûteuse.

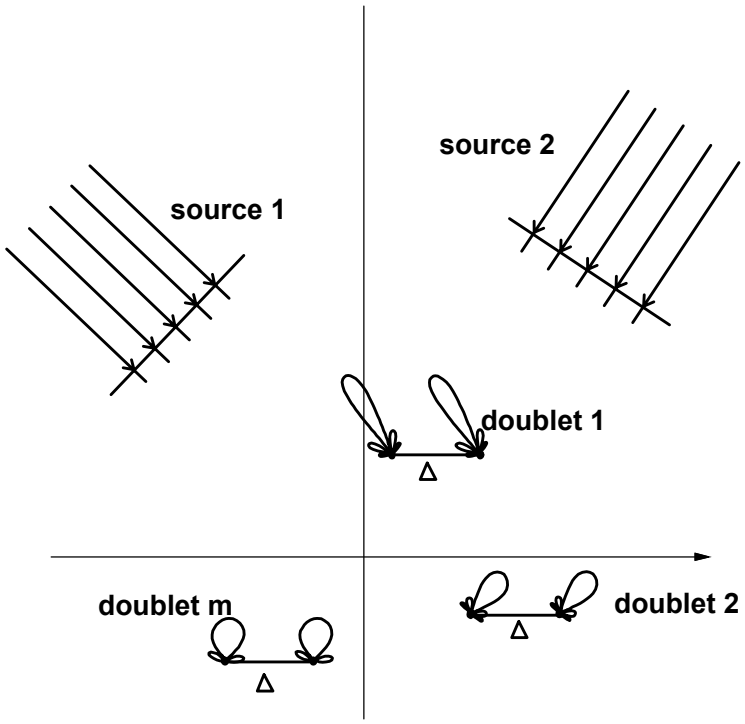

Figure 8: *Géométrie d'un réseau de doublets de capteurs utilisés pour l'algorithme ESPRIT*

Comme on le verra par la suite, le second avantage de ESPRIT sur MUSIC, est qu'aucune recherche de minima n'est nécessaire, ce qui peut réduire fortement les temps de calcul.

Equations générales

Soit un réseau de $2m$ capteurs constitué, comme en figure (8), de deux sous-réseaux identiques et coplanaires de m capteurs non forcément tous identiques, séparés par un vecteur de translation $\vec{\Delta}$. Soient n ($n \leq m$) ondes planes bande étroite de même fréquence ν_0, arrivant sur le réseau. Il est intéressant de noter que pour MUSIC on aurait $n < 2m$, ce qui rend possible la localisation d'environ deux fois plus de cibles qu'ESPRIT. Cependant, pour un réseau linéaire uniforme tel que celui utilisé par MUSIC, il suffit de faire judicieusement se chevaucher les deux sous-réseaux (cf figure (9)), pour que l'algorithme ESPRIT puisse de la même façon localiser $n < 2m$ sources.

Dans le cas général, si l'on appelle C_Y et C_Z les deux sous-réseaux séparés par le vecteur de translation $\vec{\Delta}$, les signaux reçus sur le i-ème doublet s'expriment:

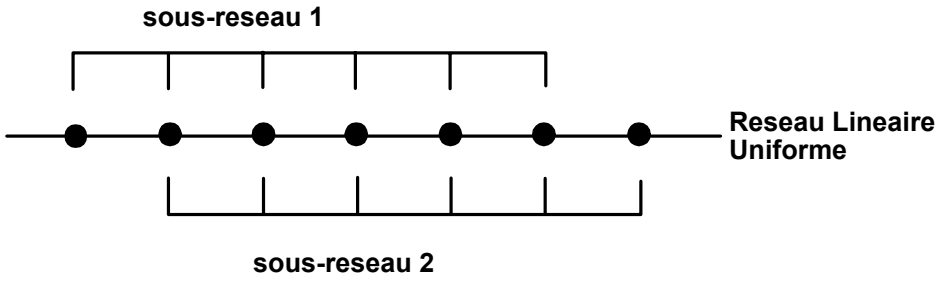

Figure 9: *Disposition de deux sous-réseaux se chavauchant, dans le cas d'un réseau linéaire uniforme pour l'algorithme ESPRIT*

$$\begin{cases} y_i(t) = \sum_{k=1}^{n} a_i(\theta_k) x_k(t) + b_{yi}(t) \\ z_i(t) = \sum_{k=1}^{n} a_i(\theta_k) e^{j2\pi\nu_0 \Delta \sin(\theta_k)/c} x_k(t) + b_{zi}(t) \end{cases} \tag{44}$$

où θ_k est l'angle d'arrivée de la k-ième source, exprimé relativement à la direction orthogonale au vecteur de déplacement $\vec{\Delta}$.

Si l'on considère tous les doublets, les équations (44) deviennent, sous forme matricielle:

$$\begin{cases} \boldsymbol{y}(t) = A\boldsymbol{x}(t) + \boldsymbol{b}_y(t) \\ \boldsymbol{z}(t) = A\Phi\boldsymbol{x}(t) + \boldsymbol{b}_z(t) \end{cases} \tag{45}$$

où

- $\boldsymbol{x}(t)$ est le $n \times 1$ vecteur des signaux arrivant sur le sous-réseau de référence C_Y.

- $\boldsymbol{b}_y(t)$ et $\boldsymbol{b}_z(t)$ sont les $m \times 1$ vecteurs de bruit (blanc et gaussien) sur les capteurs de chaque sous-réseau.

- A est la matrice $m \times n$ représentant la directivité des capteurs d'une sous-antenne (C_Y et C_Z ont des caractéristiques directionnelles égales) et les retards à l'intérieur de cette sous-antenne.

- Φ est la matrice diagonale $n \times n$ des retards de phase consécutifs au déplacement $\vec{\Delta}$ entre les deux sous-réseaux:

$$\Phi = \begin{pmatrix} e^{j\gamma_1} & 0 & ... & 0 \\ 0 & e^{j\gamma_2} & ... & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & ... & e^{j\gamma_n} \end{pmatrix} \tag{46}$$

$$\text{où} \qquad \gamma_k = \frac{2\pi\nu_0\Delta\sin(\theta_k)}{c} \tag{47}$$

La matrice Φ est unitaire et peut selon Kailath [19] être assimilée à un opérateur de rotation, d'où l'origine du mot "Rotational" dans l'acronyme ESPRIT.

Si on appelle $\boldsymbol{u}(t)$ le vecteur représentant la totalité des signaux arrivant aux $2m$ capteurs à l'instant t, alors

$$\boldsymbol{u}(t) = \begin{bmatrix} \boldsymbol{y}(t) \\ \boldsymbol{z}(t) \end{bmatrix} = \bar{A}\boldsymbol{x}(t) + \boldsymbol{n}_u(t) \tag{48}$$

$$\text{où} \qquad \bar{A} = \begin{bmatrix} A \\ A\Phi \end{bmatrix} \tag{49}$$

et $\boldsymbol{n}_u(t) = \begin{bmatrix} \boldsymbol{n}_y(t) \\ \boldsymbol{n}_z(t) \end{bmatrix}$

Toute l'astuce de l'algorithme ESPRIT provient du fait que la spécificité de la structure de la matrice $\bar{A}$ est exploitée pour estimer les éléments diagonaux de Φ *sans avoir à connaître* A.

Fondements de l'agorithme ESPRIT
L'idée qui sous-tend ESPRIT est l'utilisation d'une certaine invariance en rotation (Rotational Invariance) du sous-espace signal, qui découle elle-même de l'invariance en translation entre les doublets de capteurs du réseau.

Soit R_u la matrice de covariance $2m \times 2m$ de signaux reçus sur l'antenne:

$$R_u = \bar{A}R_x\bar{A}^* + \sigma^2 I$$

Comme précédemment pour l'algorithme MUSIC, les $2m - n$ plus petites valeurs propres de R_u valent σ^2, et les n plus grandes valeurs propres sont associées aux n vecteurs propres $\boldsymbol{e}_1, ..., \boldsymbol{e}_n$. En l'absence de bruit, ces vecteurs définissent le sous-espace signal $Vect\{E_S\} = Vect\{\bar{A}\}$, où $E_S = [\boldsymbol{e}_1, ..., \boldsymbol{e}_n]$, et $Vect\{M\}$ désigne le sous-espace engendré par les colonnes d'une matrice M.

Puisque les n vecteurs colonnes de E_S et de $\bar{A}$ définissent le même sous-espace, il

existe une unique matrice $n \times n$ inversible T, telle que:

$$E_S = \bar{A}T$$

Donc si l'on décompose E_S comme la contribution jointe des deux sous-réseaux C_Y et C_Z, on a:

$$E_S = \begin{bmatrix} E_Y \\ E_Z \end{bmatrix} = \begin{bmatrix} AT \\ A\Phi T \end{bmatrix} \tag{50}$$

d'après l'équation (49). Notons que A $(m \times n)$ est de rang n d'après l'hypothèse (9), et que T $(n \times n)$ est par définition inversible. On en déduit que les sous-espaces vectoriels engendrés par les vecteurs de E_Y, ceux de E_Z, ou ceux de A, sont égaux et de dimension n:

$$Vect\{E_Y\} = Vect\{E_Z\} = Vect\{A\}$$

Partant du fait que E_Y et E_Z (matrices $m \times n$) définissent le même sous-espace de dimension n, il vient que le rang de

$$E_{YZ} \stackrel{\text{def}}{=} [E_Y|E_Z]$$

est n également. Le noyau de E_{YZ} est donc de dimension $2n - n = n$. Soit F une matrice $2n \times n$ dont les colonnes génèrent le noyau de E_{YZ}, on a alors:

$$[E_Y|E_Z]\,F = E_Y F_Y + E_Z F_Z = 0 \tag{51}$$

avec la notation $F = \begin{bmatrix} F_Y \\ F_Z \end{bmatrix}$.

Soit, d'après (50):

$$ATF_Y + A\Phi TF_Z = 0 \tag{52}$$

Pour pouvoir résoudre ce système dont l'inconnue est Φ, il nous faut inverser F_Z. Nous allons démontrer par l'absurde que F_Z est inversible.

Supposons que F_Z, qui est de dimensions $n \times n$, ne soit pas de rang n. Il existe donc un vecteur g non nul de longueur n tel que:

$$F_Z g = 0 \tag{53}$$

Or on sait d'après l'équation (51) que

$$E_Y F_Y = -E_Z F_Z$$

d'où

$$E_Y^H E_Y F_Y = -E_Y^H E_Z F_Z$$

On rappelle ici que E_Y est de dimension $m \times n$ (où $m \geq n$) et de rang n. De ce fait, $E_Y^H E_Y$ est une matrice carrée de dimension $n \times n$, et inversible puisque de rang n. On peut donc écrire:

$$F_Y = -(E_Y^H E_Y)^{-1} E_Y^H E_Z F_Z$$

D'après (53), on en déduit:

$$\begin{aligned} \exists g \neq 0, \quad F_Y g &= -(E_Y^H E_Y)^{-1} E_Y^H E_Z F_Z g \\ &= 0 \end{aligned}$$

Pour la matrice F (de dimension $2n \times n$ et de rang n), on en déduit:

$$\exists g \neq 0, \quad Fg = \begin{bmatrix} F_Y \\ F_Z \end{bmatrix} g = \begin{bmatrix} F_Y g \\ F_Z g \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} = 0$$

ce qui contredit l'hypothèse selon laquelle F est de rang n. Nous avons donc prouvé par l'absurde que F_Z est inversible.
Si on note

$$\Psi \stackrel{\text{def}}{=} -F_Y F_Z^{-1}$$

l'équation (52) peut être réduite sous la forme:

$$AT\Psi = A\Phi T$$

soit, puisque T est inversible:

$$AT\Psi T^{-1} = A\Phi$$

La matrice A étant de rang plein, on obtient finalement:

$$\Phi = T\Psi T^{-1} \tag{54}$$

ce qui signifie, puisque Φ est diagonale, que les valeurs propres de Ψ sont égales aux éléments diagonaux de Φ. Ainsi les angles d'arrivée θ_k sont des fonctions non-linéaires (cf équation (47)) des valeurs propres de la matrice Ψ.

Il est intéressant de revenir ici sur l'origine de l'expression "Rotational Invariance" dans l'acronyme ESPRIT. On a dit plus haut que le mot "Rotational" provenait de la matrice diagonale Φ, parce qu'elle représente un opérateur rotation sur un espace à valeurs réelles. Or, dans la relation (50), on voit que

$$\begin{cases} E_Y = AT \\ E_Z = A\Phi T \end{cases}$$

Ainsi E_Y et E_Z, qui génèrent le même sous-espace (à savoir le sous-espace signal), sont par définition égaux à une multiplication par Φ près. Cela signifie que le sous-espace signal est invariant par l'opérateur de rotation Φ: c'est là l'origine de "Rotational Invariance Techniques".

Si l'on examine maintenant l'équation (54), on voit que Φ et Ψ représentent le même opérateur rotation. Ainsi, les paramètres recherchés (c'est-à-dire les angles d'arrivée) sont finalement estimés comme des fonctions non-linéaires des valeurs propres de la matrice Ψ, qui est un opérateur d'invariance en rotation entre les deux représentations E_Y et E_Z du sous-espace signal. La dénomination Estimation of Signal Parameters via Rotational Invariance Techniques est alors très clairement expliquée.

Construction de l'algorithme ESPRIT

De la même manière que dans la mise en oeuvre de l'algorithme MUSIC, on ne dispose pas ici de la vraie matrice de corrélation R_u, mais de son estimation $\hat{R}_u$ sur un échantillon fini d'observations:

$$\hat{R}_u = \frac{1}{N}\sum_{t=1}^{N} \boldsymbol{u}(t)\boldsymbol{u}^H(t)$$

Par suite, les sous-espaces engendrés par E_S et $\bar{A}$ ne sont plus exactement les mêmes. De ce fait, les sous-espaces engendrés par E_Y, E_Z et A ne sont plus identiques, ce qui entraîne une modification dans la mise en oeuvre de la suite de l'algorithme: la matrice F, qui était définie comme générant le noyau de $E_{YZ} = [E_Y|E_Z]$ (c'est-à-dire

l'ensemble des vecteurs g qui annulent $E_{YZ}g$), sera maintenant calculée comme générant l'ensemble des vecteurs g qui minimisent $\|E_{YZ}g\|$.

Ainsi, il suffit de décomposer en vecteurs propres la matrice

$$E_{YZ}^H E_{YZ} = G\Lambda G^{-1}$$

où les valeurs propres de Λ sont rangées par ordre décroissant, et de partitionner G en sous-matrices $n \times n$:

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix}$$

La matrice F, représentant le noyau de E_{YZ}, est donc associée aux plus petites valeurs propres de Λ, ce qui donne

$$F = \begin{bmatrix} G_{12} \\ G_{22} \end{bmatrix}$$

Finalement, on obtient Ψ:

$$\Psi = -G_{12}G_{22}^{-1}$$

L'estimation des éléments de la diagonale de Φ nécessite une dernière décomposition en valeurs propres (cf équation (54)), celle de Ψ. On notera $\hat{\lambda}_k$ les valeurs propres obtenues. On a alors:

$$\hat{\Phi}_k = \hat{\lambda}_k, \qquad k \in [1, n] \tag{55}$$

et la relation (47) donne finalement

$$\hat{\theta}_k = Arcsin\{\frac{c}{2\pi\nu_0\Delta} Arg(\hat{\Phi}_k)\}$$

Résumé de l'algorithme ESPRIT

1. Grouper les sorties respectives $\boldsymbol{y}(t)$ et $\boldsymbol{z}(t)$ des deux sous-réseaux C_Y et C_Z:

$$\boldsymbol{u}(t) = \begin{bmatrix} \boldsymbol{y}(t) \\ \boldsymbol{z}(t) \end{bmatrix} \tag{56}$$

2. Estimer la matrice de covariance

$$\hat{R}_u = \frac{1}{N}\sum_{t=1}^{N} \boldsymbol{u}(t)\boldsymbol{u}^H(t) \tag{57}$$

3. Décomposer $\hat{R}_u$ en valeurs propres:

$$\hat{R}_u = E\Sigma E^H \tag{58}$$

où $\Sigma = diag[\sigma_1, ..., \sigma_{2m}]$ dans laquelle les σ_i sont rangés par ordre décroissant, et $E = [\boldsymbol{e}_1, ..., \boldsymbol{e}_{2m}]$

4. Conserver les n premiers vecteurs propres, soit ceux associés aux plus grandes valeurs propres: $E_S = [\boldsymbol{e}_1, ..., \boldsymbol{e}_n]$, et décomposer E_S en deux sous-matrices $m \times n$:

$$E_S = \begin{bmatrix} E_Y \\ E_Z \end{bmatrix} \tag{59}$$

associées à chacun des sous-réseaux.

5. Former la matrice $E_{YZ} \stackrel{\text{def}}{=} [E_Y | E_Z]$ et effectuer la décomposition de $E_{YZ}^H E_{YZ}$ en valeurs propres (rangées par ordre décroissant)

$$E_{YZ}^H E_{YZ} = G\Lambda G^{-1} \tag{60}$$

6. Diviser G en sous-matrices $n \times n$:

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix} \tag{61}$$

7. Décomposer la matrice $\Psi = -G_{12}G_{22}^{-1}$ en valeurs propres $\hat{\lambda}_1, ..., \hat{\lambda}_n$

8. Les paramètres recherchés sont les angles d'arrivée θ_k estimés par:

$$\hat{\theta}_k = Arcsin\{\frac{c}{2\pi\nu_0\Delta} Arg(\hat{\lambda}_k)\} \tag{62}$$

Avantages et inconvénients de l'algorithme ESPRIT
ESPRIT, comme MUSIC, est un algorithme de Haute Résolution, donc présente comme lui des performances supérieures aux méthodes dites "classiques". Par ailleurs, des avantages supplémentaires ont été mis en exergue dans le développement précédent, tels que: très grande souplesse dans le choix de la géométrie de l'antenne, faibles contraintes sur la directivité des capteurs, faibles contraintes de calibration de l'antenne. A ces avantages liés à la partie "mécanique" du système, s'ajoute pour la partie algorithmique proprement dite, l'absence de recherche de maxima, comme cela était nécessaire pour MUSIC (recherche des "pics" de $S_{MUSIC}(\omega)$ à l'équation (42)).

Comme dans tout système, à ces avantages sont associés des inconvénients. Examinons tout d'abord les astuces mécaniques qui allègent les exigences quant aux caractéristiques de l'antenne: dans le cas d'une antenne linéaire uniforme, l'utilisation des réseaux qui se chevauchent annule tous les avantages de ESPRIT sur MUSIC. En effet, on peut facilement montrer que dans le cas de capteurs alignés et équidistants, ESPRIT nécessite que tous les capteurs aient les mêmes caractéristiques directionnelles. Si maintenant l'antenne est non linéaire, on a vu que le nombre minimum de capteurs nécessaires à la localisation de n sources est $2n$, ce qui est à peu près le double de ce que demande MUSIC. Le coût du calcul de la matrice de corrélation des signaux est donc bien plus important, de même que les décompositions en valeurs propres qui en découlent. Il faut donc être sûr de pouvoir disposer d'une puissance de calcul suffisante pour utiliser l'algorithme ESPRIT.

Ensuite, si l'algorithme MUSIC est désavantagé par la nécessité d'une recherche (souvent coûteuse) de maxima, il est bon de tempérer ce point par la remarque suivante: MUSIC utilise une seule décomposition en valeurs propres, tandis que ESPRIT en nécessite trois. Or de telles décompositions demandent des calculs d'autant plus importants que la précision exigée est grande. Il s'agit de minimiser les éléments non-diagonaux de la matrice à décomposer. Cette minimisation par rapport à un critère (plus ou moins élevé selon la précision exigée), peut parfois s'avérer tout aussi coûteuse que la recherche des pics de $S_{MUSIC}(\omega)$.

Enfin, comme MUSIC, ESPRIT est un algorithme sous-optimal, et de la même façon, ses performances se dégradent lorsque le nombre d'observations diminue, lorsque le niveau de bruit augmente, que les angles des sources se rapprochent, ou que les sources deviennent corrélées.

3.2 Une méthode optimale: le Maximum de Vraisemblance

Rappelons que dans le problème d'estimation que l'on cherche à résoudre, les données dont on dispose (les $\boldsymbol{y}(t)$) ont été modélisées par l'équation (7):

$$\boldsymbol{y}(t) = A(\Omega)\boldsymbol{x}(t) + \boldsymbol{b}(t) \tag{63}$$

C'est-à-dire que les éléments du vecteur $\boldsymbol{y}(t)$ résultent de la propagation et de la réception des sources sur l'antenne, auxquelles s'ajoute un bruit gaussien blanc décorellé complexe. Le principe de la méthode du Maximum de Vraisemblance s'appuie essentiellement sur cette hypothèse que le bruit est un vecteur gaussien de longueur m, de moyenne 0 et de covariance $\Gamma = \sigma^2 I$. Si on admet que cette hypothèse est vraie dans le problème que l'on traite, alors il suffit de trouver les valeurs des paramètres qui maximisent la probabilité pour que le bruit prenne la valeur $\boldsymbol{b} = \boldsymbol{y} - \hat{A}\hat{\boldsymbol{x}}$. En effet, la probabilité que l'estimation $(\hat{A}, \hat{\boldsymbol{x}}, \hat{\sigma})$ soit vraie est égale à la probabilité pour que le bruit $\boldsymbol{b}$ de variance $\hat{\sigma}$ prenne la valeur $\boldsymbol{y} - \hat{A}\hat{\boldsymbol{x}}$. La probabilité conditionnelle du vecteur de bruit s'écrit:

$$p(\boldsymbol{b}/\hat{A}, \hat{\boldsymbol{x}}, \hat{\sigma}) = \frac{1}{(2\pi)^{m/2}\sqrt{|\Gamma|}} \exp\left(-\frac{1}{2}(\boldsymbol{y} - \hat{A}\hat{\boldsymbol{x}})^H \Gamma^{-1} (\boldsymbol{y} - \hat{A}\hat{\boldsymbol{x}})\right) \tag{64}$$

En outre, d'après la définition de Γ, on peut écrire $\Gamma^{-1} = \frac{1}{\hat{\sigma}^2} I$ et $|\Gamma| = \hat{\sigma}^{2m}$. Par suite la vraisemblance est:

$$p(\boldsymbol{b}/\hat{A}, \hat{\boldsymbol{x}}, \hat{\sigma}) = \frac{1}{(2\pi)^{m/2}\hat{\sigma}^m} \exp\left(-\frac{1}{2\hat{\sigma}^2}||\boldsymbol{y} - \hat{A}\hat{\boldsymbol{x}}||^2\right) \tag{65}$$

Il est alors plus simple de maximiser la Log-vraisemblance:

$$\begin{aligned} \mathrm{L}_1 &= \mathrm{Log}\, p(\boldsymbol{b}/\hat{A}, \hat{\boldsymbol{x}}, \hat{\sigma}) \\ &= -\frac{m}{2}\,\mathrm{Log}(2\pi) - m\,\mathrm{Log}\,\hat{\sigma} - \frac{1}{2\hat{\sigma}^2}||\boldsymbol{y} - \hat{A}\hat{\boldsymbol{x}}||^2 \end{aligned} \tag{66}$$

La longueur du vecteur bruit m étant une donnée fixée par le problème lui même, elle est constante. On cherche donc à maximiser

$$\mathrm{L}_2 = -m\,\mathrm{Log}\,\hat{\sigma} - \frac{1}{2\hat{\sigma}^2}||\boldsymbol{y} - \hat{A}\hat{\boldsymbol{x}}||^2 \tag{67}$$

D'autre part il est clair que la Log-vraisemblance doit être maximale pour toutes les valeurs de $\hat{\sigma}$, donc l'optimisation se réduit à la minimisation de

$$\mathrm{L}_3 = ||\boldsymbol{y} - \hat{A}\hat{\boldsymbol{x}}||^2 \tag{68}$$

qui doit être appliquée à l'ensemble de N observations $\boldsymbol{y_t}$, $t \in [1, N]$. Finalement, il apparaît que le critère du Maximum de Vraisemblance consiste en la minimisation de

$$e_{MV} = \sum_{t=1}^{N} ||\boldsymbol{y_t} - \hat{A}\hat{\boldsymbol{x}}_t||^2 \tag{69}$$

Le vecteur $\hat{A}\boldsymbol{\hat{x}_t}$ appartient au sous-espace généré par $\hat{A}$, appelé $Vect\{\hat{A}\}$. De ce fait le meilleur choix pour $\boldsymbol{\hat{x}}_t$ est de prendre $\hat{A}\boldsymbol{\hat{x}}_t$ égal à la projection orthogonale de $\boldsymbol{y_t}$ sur $Vect\{\hat{A}\}$.
En conclusion, la technique du Maximum de Vraisemblance est équivalente à la minimisation de

$$e_{MV} = \sum_{t=1}^{N} ||\boldsymbol{y_t} - P_{\hat{A}}(\boldsymbol{y_t})||^2 \tag{70}$$

où $P_{\hat{A}} = \hat{A}(\hat{A}^H\hat{A})^{-1}\hat{A}^H$ est la matrice définissant la projection orthogonale sur $Vect\{\hat{A}\}$, et $\hat{A}$ est l'estimation de A.

Se poser un problème d'estimation d'angles d'arrivée, c'est se donner des $\boldsymbol{y_t}$, $t \in [1, N]$, la disposition géométrique d'une antenne, et le nombre $\hat{n}$ de sources à estimer. Partant de là, il s'agit d'estimer les angles respectifs d'arrivée de ces sources. De ce fait, résoudre le problème par la méthode du Maximum de Vraisemblance va consister à échantillonner l'espace (à $\hat{n}$ dimensions) des angles d'arrivée en formant des $\hat{n}$-uplets d'angles possibles $\hat{\Omega} = [\hat{\omega}_1, \hat{\omega}_2, ..., \hat{\omega}_{\hat{n}}]^T$. Avec chaque $\hat{n}$-uplet d'angles, on construit une matrice présumée $\hat{A}(\hat{\Omega}) = \hat{A}(\hat{\omega}_1, \hat{\omega}_2, ..., \hat{\omega}_{\hat{n}})$ selon la disposition géométrique de l'antenne. Par exemple, si l'antenne est linéaire uniforme (voir section 2.2), $\hat{A}(\hat{\Omega})$ s'exprime comme:

$$\hat{A}(\hat{\Omega}) = [\boldsymbol{a}(\hat{\omega}_1), ..., \boldsymbol{a}(\hat{\omega}_{\hat{n}})]$$

avec

$$\boldsymbol{a}(\hat{\omega}_k) = \begin{bmatrix} 1 \\ e^{-j\hat{\omega}_k} \\ \vdots \\ e^{-j(i-1)\hat{\omega}_k} \\ \vdots \\ e^{-j(m-1)\hat{\omega}_k} \end{bmatrix} \tag{71}$$

Grâce à $\hat{A}$, pour chaque $\hat{n}$-uplet d'angles on peut maintenant calculer la matrice de projection orthogonale sur le sous-espace $Vect\{\hat{A}\}$:

$$P_{\hat{A}} = \hat{A}(\hat{A}^H\hat{A})^{-1}\hat{A}^H$$

Finalement, avec cette matrice de projection et les données du problème $\boldsymbol{y_t}$, $t \in [1, N]$, on peut calculer la valeur du critère de vraisemblance explicité dans l'équation (70). En échantillonnant convenablement tout l'espace des angles d'arrivée, on obtient une hypersurface dont il s'agit de déterminer le minimum. Lorsque le problème ne comporte qu'un voire 2 angles, il est encore possible de réaliser ces calculs, tout

en sachant que plus on désire de précision dans l'estimation, plus il nous faudra échantillonner finement; mais quand on aborde un problème à 3 angles d'arrivée ou plus, il est bien trop coûteux et trop long d'envisager l'utilisation de cette méthode. Pour un problème à deux angles, par exemple, il faut échantillonner chacun des deux angles θ_1 et θ_2 entre -90^o et $+90^o$. Si on envisage une précision de 0.2^o, cela fait 900 échantillons chacun, soit 810000 couples d'angles en tout, pour lesquels il faut calculer la matrice $\hat{A}$, la matrice de projection orthogonale sur $Vect\{\hat{A}\}$ (avec une inversion de matrice), et finalement le critère de l'équation (70) comportant un calcul à effectuer sur toutes les observations.

Ainsi, même si la méthode du Maximum de Vraisemblance est optimale pour un bruit blanc gaussien, elle est peu employée parce que trop gourmande en temps de calcul. S'inspirant du critère du Maximum de Vraisemblance, Rastogi *et. al.* [17] ont essayé d'utiliser un réseau entièrement connecté pour résoudre le problème de l'estimation d'angles d'arrivée.

3.3 Une méthode neuromimétique: le réseau de Hopfield

3.3.1 Position du problème

Comme on l'a vu au Chapitre 1, l'utilisation d'une approche neuronale entièrement connectée suppose l'identification d'un certain critère quadratique représentant le problème à optimiser, avec l'énergie de Lyapunov d'un réseau de Hopfield. Dans le cas de l'estimation d'angles d'arrivée, l'idée de base proposée par Rastogi *et. al.* [17] est d'associer à chaque direction d'arrivée un neurone qui, s'il est activé (état de sortie égal à 1), indique la présence d'une source dans cette direction. Le nombre total P de neurones dans le réseau est donc directement proportionnel à la précision désirée sur les paramètres à estimer.
Plaçons nous d'emblée dans le cadre le plus souvent utilisé, à savoir une antenne linéaire uniforme; soit $\boldsymbol{s}_i$, $i \in [1, P]$ un vecteur générique représentant l'impact du signal i arrivant sur cette antenne de m capteurs sous l'angle ω_i, avec une phase ϕ_i et une puissance c_i^2. Pour une antenne linéaire uniforme, le vecteur $\boldsymbol{s}_i$ s'écrit:

$$\begin{aligned} \boldsymbol{s}_i &= c_i e^{j\phi_i} \boldsymbol{a}(\omega_i) \\ &= c_i e^{j\phi_i} [1, e^{-j\omega_i}, e^{-j2\omega_i}, ..., e^{-j(m-1)\omega_i}]^T \end{aligned} \tag{72}$$

où $\boldsymbol{a}(\omega_i)$ est le vecteur directionnel correspondant à une antenne linéaire uniforme de m capteurs. Par la suite, le vecteur $\boldsymbol{s}_i$ sera utilisé sans avoir à en expliciter le

contenu. Par conséquent, si une autre géométrie d'antenne devait être employée, il suffirait de remplacer $\boldsymbol{a}(\omega_i)$ par le vecteur directionnel correspondant à cette nouvelle géométrie dans l'équation (72), sans avoir à modifier une seule des équations qui suivent.

Soit $\boldsymbol{v} = [v_1, ..., v_P]^T$ le vecteur binaire représentant les états de sortie du réseau de Hopfield. La valeur $v_i = 1$ indique que de la direction ω_i provient un signal de phase ϕ_i et de puissance c_i^2. La valeur $v_i = 0$ indique qu'il n'y a aucune présence possédant ces caractéristiques. En s'inspirant de l'équation (68), on peut écrire la forme de l'erreur Q_R à minimiser:

$$Q_R = ||\boldsymbol{y} - [\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_P]\boldsymbol{v}||^2 \tag{73}$$

où le "R" de Q_R fait référence à Rastogi. En développant cette expression, on obtient

$$Q_R = \boldsymbol{y}^H\boldsymbol{y} + \boldsymbol{v}^T S^H S\boldsymbol{v} - \boldsymbol{y}^H S\boldsymbol{v} - \boldsymbol{v}^T S^H\boldsymbol{y} \tag{74}$$

où $S = [\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_P]$. On rappelle que dans le réseau de Hopfield les connexions synaptiques sont fixées au départ, et que c'est l'état final des neurones qui détermine la solution du problème. Le but recherché étant de minimiser Q_R par rapport à $\boldsymbol{v}$, on peut ôter le premier terme $\boldsymbol{y}^H\boldsymbol{y}$ de Q_R, car il est indépendant de $\boldsymbol{v}$. Le réseau cherche donc à minimiser

$$Q_R = \boldsymbol{v}^T S^H S\boldsymbol{v} - 2\Re\left[\boldsymbol{y}^H S\boldsymbol{v}\right] \tag{75}$$

D'autre part, d'après la Section (3.1.1) du Chapitre 1, l'énergie de Lyapunov du réseau est

$$E = -\frac{1}{2}\sum_{i=1}^{P}\sum_{j=1}^{P} T_{ij}V_i(t)V_j(t) - \sum_{i=1}^{P} I_iV_i(t) \tag{76}$$

Comme on a vu que les réseaux de Hopfield binaires avaient une moins grande probabilité de converger vers le minimum global que les réseaux analogiques, nous utiliserons ici un réseau analogique. De ce fait, puisque nous désirons quand même un vecteur d'état final binaire, nous allons ajouter à Q_R une contrainte pour faire tendre les v_i vers des valeurs binaires. De plus, pour empêcher des oscillations dues à des "auto-connexions", nous imposons la contrainte $T_{ii} = 0$. Ainsi, pour que Q_R corresponde au plus près à une fonction de Lyapunov, le terme $-\sum_{i=1}^{P}\left(\boldsymbol{s}_i^H\boldsymbol{s}_i\right)v_i(v_i - 1)$ est ajouté à Q_R:

$$Q_R = \boldsymbol{v}^T S^H S\boldsymbol{v} - 2\Re\left[\boldsymbol{y}^H S\boldsymbol{v}\right] - \sum_{i=1}^{P}\left(\boldsymbol{s}_i^H\boldsymbol{s}_i\right)v_i(v_i - 1) \tag{77}$$

ce qui permet d'annuler le terme correspondant à T_{ii} dans Q_R.
D'autre part, on peut remarquer que

$$\boldsymbol{v}^T S^H S \boldsymbol{v} = \sum_{i=1}^{P} \sum_{\substack{j=1 \\ j \neq i}}^{P} \left(\boldsymbol{s}_i^H \boldsymbol{s}_j \right) v_i v_j + \sum_{i=1}^{P} \left(\boldsymbol{s}_i^H \boldsymbol{s}_i \right) v_i^2 \tag{78}$$

or on sait que

$$\begin{aligned} \left(\boldsymbol{s}_i^H \boldsymbol{s}_j \right) v_i v_j + \left(\boldsymbol{s}_j^H \boldsymbol{s}_i \right) v_j v_i &= 2\Re \left[\left(\boldsymbol{s}_i^H \boldsymbol{s}_j \right) v_i v_j \right] \\ &= \Re \left[\left(\boldsymbol{s}_i^H \boldsymbol{s}_j \right) v_i v_j \right] + \Re \left[\left(\boldsymbol{s}_i^H \boldsymbol{s}_j \right) v_j v_i \right] \end{aligned} \tag{79}$$

d'où on déduit que

$$\boldsymbol{v}^T S^H S \boldsymbol{v} = \sum_{i=1}^{P} \sum_{\substack{j=1 \\ j \neq i}}^{P} \Re \left[\left(\boldsymbol{s}_i^H \boldsymbol{s}_j \right) v_i v_j \right] + \sum_{i=1}^{P} \left(\boldsymbol{s}_i^H \boldsymbol{s}_i \right) v_i^2 \tag{80}$$

d'où l'écriture finale de Q_R:

$$Q_R = \Re \left[\sum_{i=1}^{P} \sum_{\substack{j=1 \\ j \neq i}}^{P} \left(\boldsymbol{s}_i^H \boldsymbol{s}_j \right) v_i v_j - \sum_{i=1}^{P} \left(2 \boldsymbol{y}^H \boldsymbol{s}_i - \boldsymbol{s}_i^H \boldsymbol{s}_i \right) v_i \right] \tag{81}$$

L'identification de Q_R avec les termes de la fonction de Lyapunov (équation (76)) donne:

$$\left\{ \begin{array}{lcll} T_{ij} & = & -2\, \Re \left[\boldsymbol{s}_i^H \boldsymbol{s}_j \right] & \text{si } i \neq j \\ T_{ii} & = & 0 & \text{pour } i \in [1, P] \\ I_i & = & \Re \left[2 \boldsymbol{y}^H \boldsymbol{s}_i - \boldsymbol{s}_i^H \boldsymbol{s}_i \right] & \text{pour } i \in [1, P] \end{array} \right. \tag{82}$$

3.3.2 Réduction de complexité

La section précédente a montré comment on peut, en identifiant l'énergie de Lyapunov du réseau à une fonction quadratique s'inspirant du critère du Maximum de Vraisemblance, calculer les paramètres internes du réseau: poids synaptiques T_{ij} et entrées extérieures I_i. Pour estimer les inconnues ω_i, ϕ_i et c_i, il faut donc maintenant construire un réseau entièrement connecté possédant des paramètres internes conformes à ceux calculés dans l'équation (82). On initialise ensuite l'état des neurones à des valeurs aléatoires uniformes. Il suffit alors de rafraîchir les valeurs des neurones à chaque itération suivant l'un ou l'autre des algorithmes exposés au Chapitre 1, jusqu'à la convergence.
Il est bon de noter que le temps de convergence du réseau dépend fortement du nombre de ses neurones. Or le nombre total des neurones vaut $P = p^3$, si on décide d'échantillonner les paramètres inconnus ω_i, ϕ_i et c_i selon p valeurs chacun. Le

problème est que si l'on désire obtenir une bonne estimation des angles d'arrivée, il nous faut échantillonner finement, donc il faut choisir un p grand, ce qui implique un temps de calcul extrêmement long. Une première solution à ce dilemne est proposée par Rastogi *et. al.* [17] et reprise dans [8], [9], [12] et [13].

Approche de Rastogi

Pour Rastogi, la diminution du nombre de neurones passe par l'élimination des paramètres inconnus que l'on ne désire pas estimer. En effet, l'approche connexionniste proposée jusqu'ici permet d'estimer non seulement les angles d'arrivée ω_i, mais aussi leur phase ϕ_i et leur puissance c_i^2. Or ces deux derniers paramètres, bien qu'inconnus, ne sont généralement pas inclus dans ce qu'on appelle le problème d'estimation des angles d'arrivée.
Rastogi propose donc de modifier les équations (82) afin de faire disparaître les inconnues que l'on ne désire pas estimer. Or, d'après la définition de l'erreur Q_R (équation 73), les paramètres indésirables interviennent au niveau de $\boldsymbol{s}_i = c_i e^{j\phi_i}\boldsymbol{a}(\omega_i)$. L'erreur Q_R fait intervenir la différence entre le vecteur reçu sur l'antenne $\boldsymbol{y}$ et la contribution $\boldsymbol{s}_i$ du signal i. Comme il s'agit de minimiser la norme de cette différence, Rastogi propose de remplacer astucieusement le vecteur $\boldsymbol{s}_i$ par la projection du vecteur $\boldsymbol{y}$ selon la direction $\boldsymbol{s}_i$. Puisque $\boldsymbol{a}(\omega_i)$ et $\boldsymbol{s}_i$ définissent la même direction, projeter $\boldsymbol{y}$ sur $\boldsymbol{s}_i$ est équivalent à projeter $\boldsymbol{y}$ sur le vecteur directionnel $\boldsymbol{a}(\omega_i)$. Rastogi parvient ainsi à éliminer les paramètres indésirables ϕ_i et c_i. L'équation (73) devient alors:

$$Q_R^s = ||\boldsymbol{y} - [\boldsymbol{p}_1, \boldsymbol{p}_2, ..., \boldsymbol{p}_P]\boldsymbol{v}||^2 \tag{83}$$

où Q_R^s fait référence au critère *simplifié* de Rastogi, et $\boldsymbol{p}_i$ est la projection du vecteur $\boldsymbol{y}$ sur le vecteur directionnel $\boldsymbol{a}(\omega_i)$:

$$\boldsymbol{p}_i = P_{\boldsymbol{a}(\omega_i)}\boldsymbol{y} = \boldsymbol{a}(\omega_i)\left(\boldsymbol{a}^H(\omega_i)\boldsymbol{a}(\omega_i)\right)^{-1}\boldsymbol{a}^H(\omega_i)\boldsymbol{y} \tag{84}$$

D'où une nouvelle écriture de Q_R^s:

$$Q_R^s = ||\boldsymbol{y} - \left[P_{\boldsymbol{a}(\omega_1)}\boldsymbol{y}, P_{\boldsymbol{a}(\omega_2)}\boldsymbol{y}, ..., P_{\boldsymbol{a}(\omega_P)}\boldsymbol{y}\right]\boldsymbol{v}||^2 \tag{85}$$

La minimisation de Q_R^s se fera de la même manière qu'avant: une valeur de v_i proche de 1 signifie maintenant la présence d'une onde provenant de la direction d'arrivée ω_i, sans distinction de puissance ni de phase au premier capteur.

Si l'on compare les équations (73) et (85), on voit que $\boldsymbol{s}_i$ dans (73) est remplacé par $P_{\boldsymbol{a}(\omega_i)}\boldsymbol{y}$ dans (85). Les paramètres internes du réseau seront donc modifiés de la même façon à partir de l'équation (82):

$$\begin{cases} T_{ij} &= -2\Re\left[\boldsymbol{y}^H P^H_{\boldsymbol{a}(\omega_i)} P_{\boldsymbol{a}(\omega_j)}\boldsymbol{y}\right] & \text{si } i \neq j \\ T_{ii} &= 0 & \text{pour } i \in [1,P] \\ I_i &= \Re\left[2\boldsymbol{y}^H P_{\boldsymbol{a}(\omega_i)}\boldsymbol{y} - \boldsymbol{y}^H P^H_{\boldsymbol{a}(\omega_i)} P_{\boldsymbol{a}(\omega_i)}\boldsymbol{y}\right] & \text{pour } i \in [1,P] \end{cases} \tag{86}$$

Or, d'après la définition de $P_{\boldsymbol{a}(\omega_i)}$, on a

$$P^H_{\boldsymbol{a}(\omega_i)} = \left[\boldsymbol{a}(\omega_i)\left(\boldsymbol{a}^H(\omega_i)\boldsymbol{a}(\omega_i)\right)^{-1}\boldsymbol{a}^H(\omega_i)\right]^H = \boldsymbol{a}(\omega_i)\left(\boldsymbol{a}^H(\omega_i)\boldsymbol{a}(\omega_i)\right)^{-1}\boldsymbol{a}^H(\omega_i) = P_{\boldsymbol{a}(\omega_i)}$$

ce qui revient à dire que dans le nouveau calcul de I_i, la projection sur $\boldsymbol{a}(\omega_i)$ intervient deux fois successivement. La projection étant un morphisme idempotent, ce calcul se simplifie, ce qui donne au total:

$$\left\{\begin{array}{lcll} T_{ij} & = & -2\Re\left[\boldsymbol{y}^H P_{\boldsymbol{a}(\omega_i)} P_{\boldsymbol{a}(\omega_j)}\boldsymbol{y}\right] & \text{si } i \neq j \\ T_{ii} & = & 0 & \text{pour } i \in [1,P] \\ I_i & = & \Re\left[\boldsymbol{y}^H P_{\boldsymbol{a}(\omega_i)}\boldsymbol{y}\right] & \text{pour } i \in [1,P] \end{array}\right. \quad (87)$$

Le travail de recherche demandé au réseau est maintenant beaucoup moins lourd, puisque l'échantillonnage sur les neurones ne décrit plus que les angles d'arrivée. Ainsi, pour des angles d'arrivée entre -90^o et $+90^o$, un échantillonnage au demi degré près demande un réseau entièrement connecté de 361 neurones, ce qui est déjà non négligeable. Si on recherche une précision plus importante, il faut encore accroître le nombre de neurones, ce qui non seulement fait reculer la vitesse de convergence, mais en outre compromet les chances du réseau de trouver effectivement le minimum global (il est clair que plus on a de neurones, plus on crée des minima locaux à la fonction de Lyapunov du réseau). De plus, Rastogi ne prend en compte qu'une seule observation des signaux sur l'antenne, alors que MUSIC, ESPRIT et le Maximum de Vraisemblance en utilisent plusieurs, et que Stoica a démontré [24] que l'erreur sur l'estimation décroît quand le nombre d'observations croît.
C'est en analysant ces considérations, et en détaillant les calculs que Rastogi met en œuvre pour obtenir les paramètres internes du réseau, que Martin *et. al.* [16] ont pu montrer pourquoi la méthode pourtant si élégante proposée par Rastogi ne pouvait fonctionner telle que. Martin *et. al.* ont de fait proposé une alternative permettant de reprendre l'approche correctement.

Correction de Martin et Lobert
Bien que Rastogi lui-même, à la fin de son article, mentionne le fait que sa méthode ne fonctionne pas toujours très bien, notamment dans le cas d'ondes provenant de directions proches, Jha *et. al.* [8] reprennent l'approche sans déceler d'erreur, et tentent d'améliorer son fonctionnement en essayant diverses variantes au réseau de Hopfield analogique de base: annulation de gain, machine de Boltzmann, réseau stochastique, descente réitérée. Pourtant aucune comparaison n'est faite avec des méthodes connues (MUSIC, ESPRIT, ...). En outre, le modèle des signaux est simplifié à l'extrême, puisque les approches Rastogi et Jha ne prennent en compte qu'une seule observation ($N = 1$).

C'est sans doute en tentant de mettre en pratique la méthode de Rastogi que Martin *et. al.* [16] ont constaté son mauvais fonctionnement. En analysant alors

plus en détail les calculs utilisés par Rastogi, ils ont mis en évidence ce que ni Jha *et. al.* ni Rastogi lui-même n'avaient remarqué.

Martin *et. al.* partent de l'équation (73) définissant l'erreur Q_R généralisée, et qui mène à un premier système (82) détaillant la valeur des paramètres internes du réseau. L'équation (73):

$$Q_R = ||\boldsymbol{y}_t - [\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_P]\boldsymbol{v}||^2 \tag{88}$$

peut se réécrire

$$Q_R = ||\boldsymbol{y}_t - [\boldsymbol{a}(\omega_1), \boldsymbol{a}(\omega_2), ..., \boldsymbol{a}(\omega_P)] \begin{bmatrix} x_1 v_1 \\ x_2 v_2 \\ \vdots \\ x_P v_P \end{bmatrix} ||^2 \tag{89}$$

où $x_i = c_i e^{j\phi_i}$. Seuls les i tels que $v_i = 1$ seront présents réellement, et cette équation est à rapporter à l'équation (68)

$$\mathrm{L}_3 = ||\boldsymbol{y_t} - \hat{A}\hat{\boldsymbol{x}}_t||^2 \tag{90}$$

puisque $\hat{A} = [\boldsymbol{a}(\omega_1), \boldsymbol{a}(\omega_2), ..., \boldsymbol{a}(\omega_n)]$ où n est le nombre de sources réellement présentes. Jusqu'ici, l'approche de Rastogi peut donc être considérée comme une approche équivalente à celle du Maximum de Vraisemblance.
A partir de là, les simplifications envisagées divergent. On simplifie l'équation (68) du Maximum de Vraisemblance en disant que le vecteur $\hat{A}\hat{\boldsymbol{x}}_t$ appartient au sous-espace généré par $\hat{A}$, appelé $Vect\{\hat{A}\}$. De ce fait le meilleur choix pour $\hat{\boldsymbol{x}}_t$ est de prendre $\hat{A}\hat{\boldsymbol{x}}_t$ égal à la projection de $\boldsymbol{y}_t$ sur $Vect\{\hat{A}\}$. Le critère du Maximum de Vraisemblance devient alors

$$E_{MV} = ||\boldsymbol{y}_t - P_{\hat{A}}(\boldsymbol{y}_t)||^2 \tag{91}$$

D'un autre côté, on peut réécrire (89) comme

$$Q_R = ||\boldsymbol{y}_t - \sum_{i=1}^{P} \boldsymbol{a}(\omega_i) x_i v_i||^2 \tag{92}$$

L'erreur de Rastogi est alors de calquer son raisonnement sur celui qui permet de simplifier la formule du Maximum de Vraisemblance, sans vérifier s'il en a le droit. En effet, il remplace chaque $\boldsymbol{a}(\omega_i)x_i$ de l'équation (92) par la projection orthogonale de $\boldsymbol{y}_t$ sur $\boldsymbol{a}(\omega_i)$:

$$Q_R^s = ||\boldsymbol{y}_t - \sum_{i=1}^{P} P_{\boldsymbol{a}(\omega_i)} \boldsymbol{y}_t \; v_i||^2 \tag{93}$$

d'où l'équation (85) dérivée au paragraphe précédent. Le fait est que le critère du Maximum de Vraisemblance (91) n'est pas équivalent au critère simplifié Q_R^s de Rastogi. En effet, il n'y a égalité entre les deux critères que dans les deux cas suivants:

- il n'y a qu'une seule source, auquel cas $P_{A(\omega)} = P_{\boldsymbol{a}(\omega)}$

- les vecteurs directionnels $\boldsymbol{a}(\omega_i)$ sont orthogonaux. Dans ce cas seulement, la projection de la somme est égale à la somme des projections:

$$P_{A(\Omega)} = \sum_{i=1}^{n} P_{\boldsymbol{a}(\omega_i)} \tag{94}$$

Rastogi lui-même remarque que pour un angle seul ou des angles bien séparés, son critère simplifié peut être utilisé. Par contre, il note que pour des angles proches, "il faudrait prendre en compte les interactions entre les vecteurs directionnels".

Partant du fait que le critère de Rastogi, bien que non parfaitement équivalent au critère du Maximum de Vraisemblance, est d'un usage plus aisé que ce dernier, et peut malgré tout fournir une estimation assez proche de la réalité, Martin *et. al.* proposent d'utiliser la méthode de Rastogi comme point de départ à une méthode plus précise, mais qui nécessite un point d'initialisation.

Amélioration de la méthode

Martin *et. al.*, s'appuyant sur leur analyse du critère de Rastogi, proposent la démarche suivante: puisque le critère de Rastogi n'est pas optimal, on ne peut l'utiliser que comme point de départ à un autre algorithme plus précis. D'autre part Martin *et. al.* incluent la possibilité de prendre en compte plusieurs ($N > 1$) observations. Enfin, puisque la vitesse de convergence dépend du nombre de neurones, ce premier réseau comportera un "faible" nombre de neurones, car on ne lui demande pas de grande précision. D'où une approche en deux étapes:

1. balayage de l'espace avec un échantillonnage large, pour obtenir une estimation grossière de la localisation des sources.

2. focalisation autour de chaque estimation grossière obtenue à l'étape précédente: pour cette estimation fine, un critère plus précis que celui de Rastogi doit être utilisé.

En ce qui concerne la première étape, Martin *et. al.* introduisent la possibilité de prendre en compte plusieurs observations, ce qui n'était pas le cas dans les équations

(87) de Rastogi. Si on dispose de N observations, les équations (87) sont maintenant pondérées par une moyenne sur $t, t \in [1, N]$:

$$\begin{cases} T_{ij} &= -\dfrac{2}{N}\displaystyle\sum_{t=1}^{N} \Re\left[{\boldsymbol{y}_t}^H P_{\boldsymbol{a}(\omega_i)} P_{\boldsymbol{a}(\omega_j)} \boldsymbol{y}_t\right] & \text{si } i \neq j \\ T_{ii} &= 0 & \text{pour } i \in [1, P] \\ I_i &= \dfrac{1}{N}\displaystyle\sum_{t=1}^{N} \Re\left[{\boldsymbol{y}_t}^H P_{\boldsymbol{a}(\omega_i)} \boldsymbol{y}_t\right] & \text{pour } i \in [1, P] \end{cases} \tag{95}$$

La première étape de la méthode de Martin étant maintenant connue, on va s'attacher ici à expliciter la seconde. A ce stade, nous sommes donc en possession des données sur l'antenne $\boldsymbol{y}_t$ et d'une estimation grossière des angles d'arrivée $\tilde{\Omega} = [\tilde{\omega}_1, ..., \tilde{\omega}_n]$. Reprenons le critère du Maximum de Vraisemblance (68):

$$\begin{aligned} ||\boldsymbol{y} - A(\Omega)\boldsymbol{x}||^2 &= ||\boldsymbol{y} - [\boldsymbol{a}(\omega_1), ..., \boldsymbol{a}(\omega_n)] \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} ||^2 \\ &= ||\boldsymbol{y} - \sum_{\substack{i=1 \\ i \neq k}}^{n} \boldsymbol{a}(\omega_i) x_i - \boldsymbol{a}(\omega_k) x_k||^2 \\ &= ||\tilde{\boldsymbol{y}}^k - \boldsymbol{a}(\omega_k) x_k||^2 \end{aligned} \tag{96}$$

où $\tilde{\boldsymbol{y}}^k = \boldsymbol{y} - A(\Omega)\tilde{\boldsymbol{x}}^k$, et $\tilde{\boldsymbol{x}}^k$ est le vecteur $\boldsymbol{x}$ avec la k-ième composante nulle. De cette façon, le vecteur $\tilde{\boldsymbol{y}}^k$ ne contient plus que les apports de la k-ième source. Le critère de minimisation (96) se comporte donc comme un critère monodimensionnel, ce qui permet d'écrire

$$Q_M = ||\tilde{\boldsymbol{y}}^k - \boldsymbol{a}(\omega_k) x_k||^2 = ||\tilde{\boldsymbol{y}}^k - P_{\boldsymbol{a}(\omega_k)} \tilde{\boldsymbol{y}}^k||^2 \tag{97}$$

où le "M" indique le critère de Martin. Ce critère, qui pourrait être comparé à un critère de Rastogi monodimensionnel, est théoriquement optimal, car il a été obtenu directement à partir du critère du Maximum de Vraisemblance.

Martin *et. al.* ne détaillent pas le calcul nécessaire pour construire le vecteur $\tilde{\boldsymbol{y}}^k$, mais il est clair que l'on doit utiliser l'estimation grossière des angles fournie par la méthode de Rastogi. Explicitons ce calcul:

$$\tilde{\boldsymbol{y}}^k = \boldsymbol{y} - A(\Omega)\tilde{\boldsymbol{x}}^k \tag{98}$$

Nous connaissons parfaitement $\boldsymbol{y}$, car cela fait partie des données propres du problème. Avec l'estimation $\tilde{\Omega}$ des angles, on peut calculer une estimation grossière $A(\tilde{\Omega})$ de $A(\Omega)$. Il est toutefois plus difficile d'estimer $\boldsymbol{x}$ (pour en tirer $\tilde{\boldsymbol{x}}^k$). En effet, ce vecteur n'est pas directement fourni par la méthode simplifiée de Rastogi, donc

on doit l'obtenir par déduction. D'après le modèle des signaux:

$$\boldsymbol{y} = A(\Omega)\boldsymbol{x} + \boldsymbol{b} \tag{99}$$

Comme on ne connaît pas le bruit, on approxime $\boldsymbol{y} \simeq A(\Omega)\boldsymbol{x}$, ce qui nous permet d'écrire:

$$\boldsymbol{x} \simeq A^{+}(\Omega)\boldsymbol{y} \tag{100}$$

où la pseudo-inverse W^{+} d'une matrice W quelconque s'écrit $W^{+} = (W^{H}W)^{-1}W^{H}$. On obtient finalement la valeur recherchée

$$\tilde{\boldsymbol{y}}^{k} \simeq \boldsymbol{y} - A(\tilde{\Omega}) \left[\widetilde{A^{+}(\tilde{\Omega})\boldsymbol{y}} \right]^{k} \tag{101}$$

Tous les paramètres de l'équation (97) étant explicités, on peut écrire le critère total prenant en compte toutes les observations:

$$Q_M = \frac{1}{N} \sum_{t=1}^{N} ||{\tilde{\boldsymbol{y}}_t}^{k} - P_{\boldsymbol{a}(\omega_k)} {\tilde{\boldsymbol{y}}_t}^{k}||^2 \tag{102}$$

Il suffit ensuite d'identifier cette dernière équation à l'énergie de Lyapunov du réseau pour trouver les valeurs des paramètres internes:

$$\begin{cases} T_{ij} &= -\dfrac{2}{N} \displaystyle\sum_{i=1}^{N} \Re \left[\left({\tilde{\boldsymbol{y}}_t}^{k} \right)^{H} P_{\boldsymbol{a}(\omega_i)} P_{\boldsymbol{a}(\omega_j)} {\tilde{\boldsymbol{y}}_t}^{k} \right] & \text{si } i \neq j \\ T_{ii} &= 0 & \text{pour } i \in [1, P] \\ I_i &= \dfrac{1}{N} \displaystyle\sum_{i=1}^{N} \Re \left[\left({\tilde{\boldsymbol{y}}_t}^{k} \right)^{H} P_{\boldsymbol{a}(\omega_i)} {\tilde{\boldsymbol{y}}_t}^{k} \right] & \text{pour } i \in [1, P] \end{cases} \tag{103}$$

Résumé de l'algorithme de Martin:

1. Construire un réseau de Hopfield analogique possédant peu de neurones (grand pas d'échantillonnage)

2. Utiliser le critère simplifié de Rastogi Q_R^s (équation (85)) pour calculer les paramètres internes T_{ij} et I_i (équation (95))

3. Faire converger le réseau rapidement par un algorithme approprié (*cf* Chapitre 1) $\Rightarrow$ première estimation: $\tilde{\Omega} = [\tilde{\omega}_1, ..., \tilde{\omega}_n]$

4. Construire n réseaux de Hopfield analogiques basés autour de chaque $\tilde{\omega}_k$ estimé précédemment

5. Utiliser le critère Q_M du Maximum de Vraisemblance "monodimensionnalisé" par Martin pour calculer les paramètres internes T_{ij} et I_i (équation (103))

6. De manière indépendante, faire converger ces réseaux. Ils fournissent chacun une estimation $\hat{\omega}_k$ de l'angle auquel ils sont respectivement dédiés.

Discussion:

Le critère théorique Q_M développé par Martin à l'équation (96) est totalement équivalent au critère du Maximum de Vraisemblance; seule la manière de séparer les variables est différente. Toutefois, la dérivation poursuivie à l'équation (97) n'est équivalente au MV que si la valeur de $\tilde{\boldsymbol{y}}^k$ est exacte. Or les équations (100) et (101) montrent que cette valeur est *estimée* en utilisant des approximations de deux ordres: d'une part on néglige le bruit, et d'autre part on approxime la matrice $A(\Omega)$ par une estimation $A(\tilde{\Omega})$ découlant de l'utilisation grossière du critère de Rastogi sur un réseau peu échantillonné. Ainsi, puisque la valeur de $\tilde{\boldsymbol{y}}^k$ n'est pas exacte, on ne peut pas dire que le critère de minimisation Q_M présenté à l'équation (97) est équivalent au critère du Maximum de Vraisemblance.

Toutefois, même s'il n'est pas optimal au sens du MV, ce raffinement de la méthode de Rastogi apporte quand même plus de précision dans l'estimation des angles d'arrivée, comme le montrent les résultats présentés par Martin *et. al.* De plus, il est clair que cette méthode en deux étapes est plus rapide que la méthode originale de Rastogi, puisque les réseaux employés sont beaucoup plus petits: pour un domaine d'étude situé dans l'intervalle $[0, 60^o]$ avec un pas d'échantillonnage de 0.3^o, la méthode brute de Rastogi nécessite un réseau de 201 neurones. Avec la méthode en deux étapes de Martin *et. al.*, on peut se contenter d'un pas de 3^o pour le premier réseau, soit un réseau de 21 neurones (un tel réseau converge très rapidement). Pour les réseaux de l'étape de raffinement, avec un pas d'échantillonnage fin de 0.3^o, il suffit de 11 neurones pour garder la couverture de $\pm 1.5^o$ autour de l'angle estimé dans l'étape 1. Avec un ordre de grandeur aussi important entre les deux approches, on pourrait même songer à opérer un troisième raffinement, en partant des estimations fournies par l'étape 2 (beaucoup plus précises que celles de l'étape 1). Il suffirait alors de recalculer les poids synaptiques associés à ces nouvelles estimations, puis de relancer l'étape 2 dans des conditions qui, cette fois, seraient presque optimales par rapport au Maximum de Vraisemblance.

3.4 Une autre approche neuronale: utilisation d'un PMC

3.4.1 Introduction

Même si l'approche de Martin *et. al.* donne des estimations plus précises que l'approche originale de Rastogi tout en étant plus rapide, il ne faut pas oublier que toute méthode utilisant une convergence selon la minimisation d'un critère est forcément plus lente (parce que le temps de convergence est rarement garanti) qu'une méthode directe.

Il semble alors assez naturel de penser à utiliser un réseau unidirectionnel tel que le Perceptron MultiCouche (PMC), parce que si on peut construire un PMC de taille raisonnable sachant répondre au problème posé, son temps de réponse sera constant et surtout très court. Comme on *connaît* un modèle mathématique (et analytique) du problème, il est possible de constituer une base d'exemples représentative. De ce fait, le Perceptron MultiCouche, qui procède par apprentissage supervisé, semble a priori utilisable pour le problème d'estimation d'angles d'arrivée. Tant que le nombre de paramètres et de données d'entrée reste raisonnable, on peut donc facilement construire un réseau dédié à l'estimation d'Angles d'Arrivée, que l'on entraînera sur une base judicieusement choisie, et représentative de la dimensionnalité du problème.

3.4.2 Approche par secteurs

S'inspirant de l'étude pionnière de Rastogi *et. al.*, qui traite le problème des Angles d'Arrivée par une approche neuronale entièrement connectée, des auteurs ont cherché à élargir l'approche à des réseaux organisés par couches.

Dans [30] et [7], une antenne linéaire uniforme de capteurs est associée à un perceptron à trois couches. Le problème est restreint à l'estimation de l'Angle d'Arrivée d'un seul signal. La région d'arrivée $[-90^o, 90^o]$ est divisée en d secteurs. Wells *et. al.* proposent d'entraîner un perceptron (voir figure (10)) à retrouver le secteur angulaire contenant l'angle d'arrivée du signal: la couche de sortie est composée de d neurones correspondant chacun à un secteur angulaire de largeur $\frac{180^o}{d}$. Pour un exemple donné, un seul des neurones de sortie est à 1: c'est celui qui est associé au secteur d'arrivée. Les autres neurones de sortie sont à 0. La couche cachée comprend c neurones, avec

$$2^{c-1} < d \leq 2^c \tag{104}$$

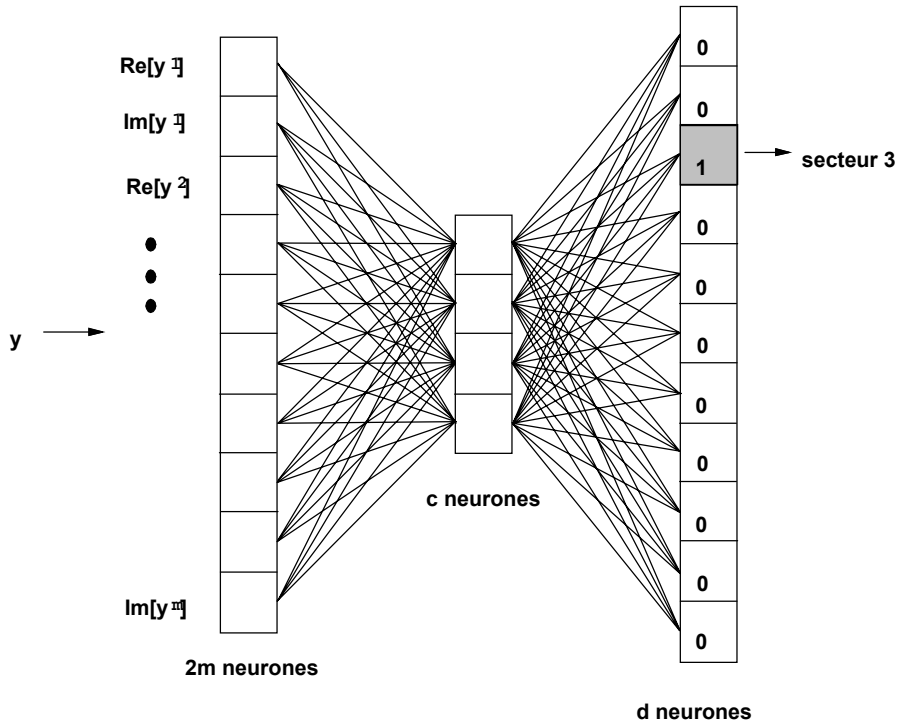

Figure 10: *Le perceptron à trois couches utilisé par Wells*

c'est-à-dire que d peut être codé en binaire sur c bits au minimum (c'est pour Wells *et. al.* une façon de s'assurer qu'il n'y a pas de perte d'informaiton au niveau de la couche cachée). La couche d'entrée reçoit le signal brut $\boldsymbol{y}$ recueilli en pied d'antenne. Cependant, comme ce perceptron est composé de données réelles uniquement, les m éléments complexes du vecteur $\boldsymbol{y}$ sont considérés ici comme $2m$ éléments réels formant le vecteur d'entrée du PMC. Il est à noter que le réseau n'utilise qu'un seul vecteur $\boldsymbol{y}$ pour estimer l'angle d'arrivée, contrairement aux approches précédentes (MUSIC, ESPRIT, ...) qui utilisent N observations différentes $\boldsymbol{y}_t,\ t \in [1, N]$. Nous verrons par la suite ce qui en résulte.

Wells *et. al.* entraînent leur réseau sur une base de données représentant le problème qu'ils cherchent à traiter: une seule source arrivant par un angle décrivant $[-90^o, 90^o]$ sur une antenne linéaire uniforme de 4 capteurs. Il y a 10 neurones sur la couche de sortie, pour 10 secteurs angulaires de largeur 18^o chacun. La couche d'entrée se compose de $2m = 8$ neurones, et la couche cachée de c neurones, où c est la partie entière supérieure de $\dfrac{\log 10}{\log 2}$, soit $c = 4$. Leur base d'apprentissage ne contient pas de bruit, pas plus que leur base de test. Pourtant, alors que les secteurs qui divisent l'intervalle $[-90^o, 90^o]$ font 18^o de large, Wells *et. al.* reconnaissent que près de 7% de leurs signaux de test sont mal classifiés.

Ils proposent alors d'améliorer leur méthode en forçant le réseau à effectuer certaines opérations de codage sur la couche cachée. En contrepartie, les poids du réseau ne sont plus obtenus par rétropropagation à partir de l'erreur de sortie (*cf* Chapitre 1 Section (2.3)), mais sont calculés de manière directe suivant une méthode des moindres carrés. La structure du réseau utilisé est la même, mais les neurones de la couche cachée sont désormais contraints de représenter le code Gray du secteur dans lequel se trouve l'angle d'arrivée. Les poids entre la couche cachée et la couche de sortie sont donc fixes, et la fonction de transfert des neurones de la couche cachée est une fonction signe (voir figure (11)): $v = sg(u)$.

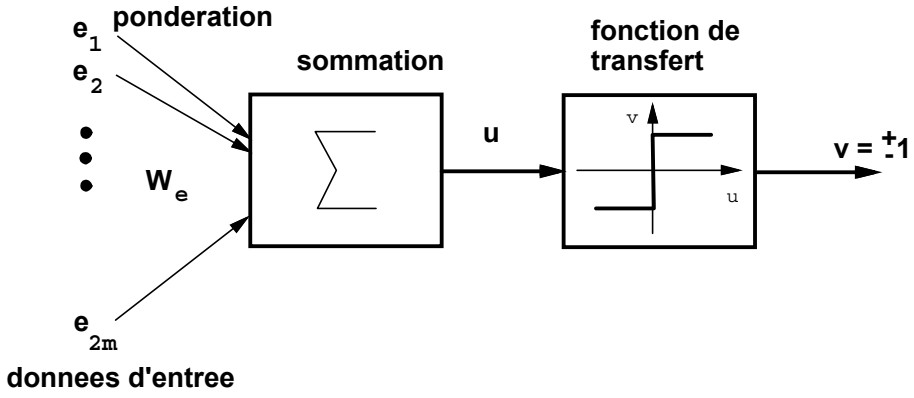

Figure 11: *Représentation d'un neurone de la couche cachée*

Pour l'exemple pratique précédent, ($m = 4$ et $d = 10$), la matrice $c \times d$ des poids de sortie est donc W_s telle que:

$$W_s = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & -1 & -1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \end{bmatrix} \tag{105}$$

Notons en passant que du fait que le codage entre la couche cachée et la couche de sortie est un codage prédéterminé (codage Gray), on peut considérer que le réseau se comporte de la même façon qu'un réseau deux couches.

D'après Wells *et. al.*, les mauvaises classifications survenues lors de l'utilisation de leur premier réseau, sont causées par un positionnement flou des frontières entre les différents secteurs. Au lieu de chercher à sur-entraîner leur réseau avec une base d'apprentissage plus grande, ou un nombre d'itérations plus important, ils proposent de *calculer* la matrice W_e des poids d'entrée. Pour Wells, W_e est la solution de

Wiener-Hopf [27] aux moindres carrés de l'équation

$$M = EW_e \tag{106}$$

où M et E représentent les matrices de la couche cachée et de la couche d'entrée, respectivement. E a été construite à partir des $2m$ composantes réelles et imaginaires du vecteur $\boldsymbol{y}$ en pied d'antenne, pour l'angle situé au centre de chacun des 10 secteurs à discriminer. De la même façon, la matrice M est la représentation du code Gray pour chacun des 10 secteurs. La solution aux moindres carrés de l'équation (106) s'écrit:

$$W_e = E^+ M$$

où $E^+ = (E^T E)^{-1} E^T$ est la pseudo-inverse de E. Remarquons que pour $m = 4$, on a $2m = 8 \leq 10$, donc la matrice $E^T E$ est bien inversible (pour E de dimension $10 \times 2m$). Cependant, Wells *et. al.* soulignent que si on cherche à augmenter le nombre de secteurs à discriminer (pour un nombre de capteurs m constant), la matrice E devient mal conditionnée, et son inversion produit des résultats qui se dégradent. Ils proposent alors de n'utiliser que des matrices E carrées, c'est-à-dire que m capteurs permettent de séparer l'espace en $2m$ secteurs seulement.

Afin d'augmenter le nombre des secteurs divisant l'espace, Wells *et.al.* proposent une troisième méthode: c'est une méthode graphique utilisant la "polarité" des fonctions trigonométriques, et qui fournit, comme précédemment, un code Gray qu'il s'agit ensuite de décoder pour retrouver le numéro de secteur.
Considérons la figure (12) représentant une antenne linéaire de capteurs identiques, mais non équirépartis. Le vecteur directionnel associé à cette antenne est

$$\boldsymbol{a}(\omega) = \begin{bmatrix} 1 \\ e^{-j\omega/2} \\ e^{-j\omega} \\ e^{-j2\omega} \\ e^{-j4\omega} \end{bmatrix}$$

avec comme d'habitude $\omega = \pi \sin\theta$, pour un angle d'arrivée θ situé entre -90^o et 90^o. Le signal reçu sur l'antenne se modélise par

$$\boldsymbol{y} = x\,\boldsymbol{a}(\omega) + \boldsymbol{b}$$

Wells se place dans une hypothèse sans bruit, donc on peut simplifier cette équation en divisant $\boldsymbol{y}$ par sa première composante. On obtient alors:

$$\boldsymbol{y}^d = \boldsymbol{a}(\omega)$$

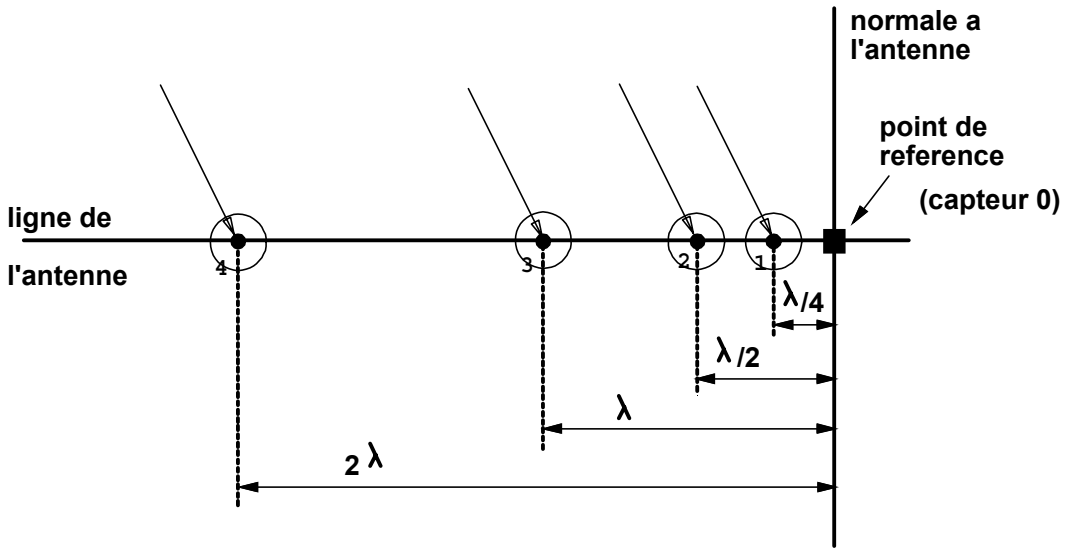

Figure 12: *Configuration non équirépartie de l'antenne pour la méthode de Wells graphique*

ce qui permet de se débarasser de la puissance de la source. De cette façon, on peut se séparer de la première composante de $\boldsymbol{y}^d$ (qui vaut désormais 1), c'est-à-dire que l'on ne considère plus le capteur situé au point de référence (d'où la numérotation 0 pour le capteur référence).
Il faut noter que cette simplification n'est valable que dans un environnement sans bruit, et surtout, en présence d'une seule source. En effet, si $\boldsymbol{y}$ est le résultat de l'arrivée sur l'antenne de deux sources dans un environnement non bruité, on a

$$\boldsymbol{y} = x_1 \boldsymbol{a}(\omega_1) + x_2 \boldsymbol{a}(\omega_2)$$

donc la simplification de $\boldsymbol{y}$ par division par sa première composante donne

$$\boldsymbol{y}^d = \frac{1}{x_1 + x_2} \begin{bmatrix} x_1 + x_2 \\ x_1 e^{-j\omega_1/2} + x_2 e^{-j\omega_2/2} \\ \vdots \\ x_1 e^{-j4\omega_1} + x_2 e^{-j4\omega_2} \end{bmatrix}$$

ce qui en vérité complique plus les choses qu'il ne les simplifie, contrairement au cas une source, comme nous allons le voir maintenant.

Examinons le graphe (figure (13)) de l'opposé de la partie imaginaire du signal (non bruité) reçu sur le capteur 1, et le graphe de la partie réelle du signal reçu sur le capteur 2. On peut voir que la combinaison de ces deux graphes permet de définir quatre secteurs distincts, suivant le signe de ces fonctions. Le tableau ci-dessous montre comment ces quatre secteurs se répartissent.

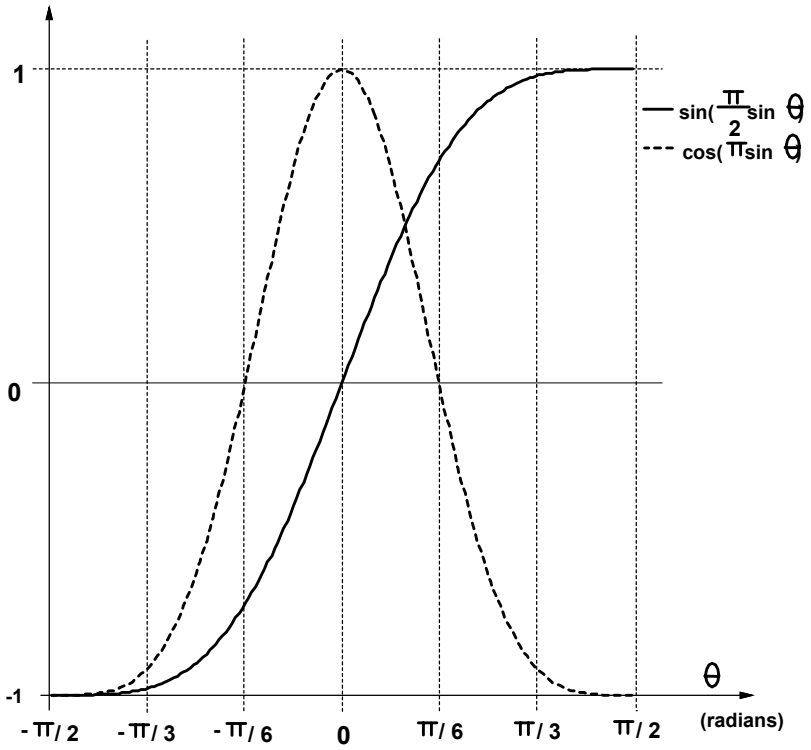

Figure 13: *Variations de* $\sin(\frac{\pi}{2}\sin\theta)$ *et* $cos(\pi\sin\theta)$ *en fonction de* θ *entre* -90^o *et* 90^o

θ	-90^o	-30^o	0^o	30^o	90^o
secteur	1	2	3	4	
$\sin(\frac{\pi}{2}\sin\theta)$	$-$	$-$	$+$	$+$	
$\cos(\pi\sin\theta)$	$-$	$+$	$+$	$-$	

Si on trace, de la même façon, le comportement de l'opposé des parties réelles des capteurs 3 et 4, soit $-\cos(2\pi\sin\theta)$ et $-\cos(4\pi\sin\theta)$, on obtient au total une différenciation sur 16 secteurs, comme indiqué dans le tableau suivant (on note comme d'habitude $\omega = \pi\sin\theta$).

secteur	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$\sin(\frac{\omega}{2})$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$-$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$
$\cos(\omega)$	$-$	$-$	$-$	$-$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$+$	$-$	$-$	$-$	$-$
$-\cos(2\omega)$	$-$	$-$	$+$	$+$	$+$	$+$	$-$	$-$	$-$	$-$	$+$	$+$	$+$	$+$	$-$	$-$
$-\cos(4\omega)$	$-$	$+$	$+$	$-$	$-$	$+$	$+$	$-$	$-$	$+$	$+$	$-$	$-$	$+$	$+$	$-$

On retrouve le code Gray précédemment dérivé dans l'équation (105) représentant les poids de la matrice W_s. Le réseau est donc maintenant représenté comme un réseau

à deux couches: les neurones de la couche d'entrée reçoivent les signaux de l'antenne et opèrent une fonction de transfert égale à la fonction signe. La sortie des neurones d'entrée fournit donc directement le code Gray du numéro du secteur angulaire associé (figure (14)). Il est bon de noter que si, avec cette nouvelle méthode, on peut

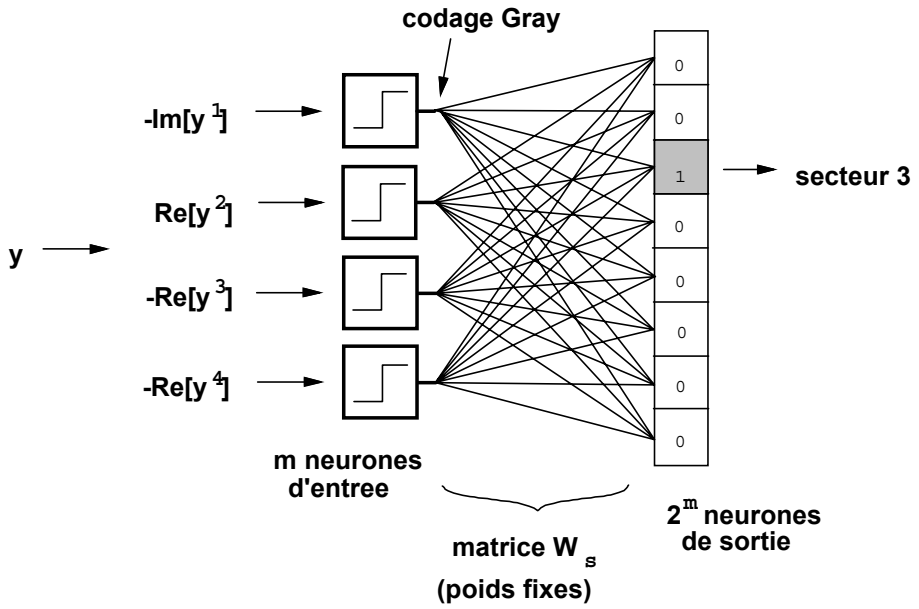

Figure 14: *Le perceptron à deux couches correspondant à la troisième méthode de Wells*

distinguer plus de secteurs, la largeur angulaire de ces secteurs est variable: de 7.2^o pour les secteurs proches de 0^o, à 29^o pour les secteurs proches de $\pm 90^o$.

Pour 4 (+1) capteurs, on peut discriminer 16 secteurs avec cette méthode. Et, de manière générale, pour m capteurs, on peut distinguer 2^{m-1} secteurs.

3.4.3 Conclusion

Avantages et inconvénients des méthodes proposées:

Il est clair que le principal avantage des méthodes de Wells réside dans leur simplicité d'utilisation (pas ou peu d'entraînement pour les réseaux, neurones simplifiés à l'extrême). L'inconvénient principal, qui en découle directement, est qu'elles ne peuvent résoudre que des problèmes simples: un seul signal, pas ou peu de bruit. En outre, leur résolution est assez médiocre, puisqu'elles se contentent de classer les angles par secteurs. Remarquons au passage que Wells utilise les perceptrons comme des classifieurs, alors que l'on se trouve dans un contexte où la

séparation entre deux classes n'a aucune marge d'erreur: les angles juste inférieurs à 0^o sont dans une classe différente des angles juste supérieurs à 0^o. Cette très grande discontinuité sur un problème aussi complexe est difficile à modéliser par un perceptron. Enfin, ces méthodes semblent difficilement extrapolables à plus d'un angle, en particulier à cause du fait qu'elles nécessitent un codage dont la complexité croîtrait de manière exponentielle avec le nombre de signaux. La troisième méthode est d'ailleurs impossible à étendre à plus d'un seul signal, comme on l'a fait remarquer plus haut. Cependant, des remarques suggérées par l'étude de ce travail, on peut tirer des observations intéressantes.

Discussion:

Pour un perceptron multicouche, l'approche par secteurs semble déconseillée car dès que le nombre d'angles croît, ou dès que l'on veut une analyse fine, il faut augmenter considérablement à la fois le nombre de neurones en sortie, et le nombre d'exemples sur la base d'apprentissage, rendant ainsi l'approche de plus en plus longue et complexe.

Malgré leur apparence, et malgré le nom de "Perceptrons Multicouches" que donne Wells à ses estimateurs, il est plus juste de les considérer comme des perceptrons à deux couches, en raison du fait que le codage Gray $\rightarrow$ numéro de secteur n'apporte ni n'enlève aucune information: puisque ce codage est totalement bijectif, on peut le supprimer (ou le mettre dans une étape de post-traitement classique). Les formidables possibilités des Perceptrons à trois couches n'ont donc finalement pas été explorées.

D'autre part, il faut considérer le bruit comme un élément à part entière du problème, car s'il est omis dans la modélisation des solutions (comme cela est fait chez Wells), le comportement de la méthode en phase de test se dégrade considérablement en présence de bruit.

En ce qui concerne le bruit, on peut faire une autre remarque: Wells *et. al.* n'utilisent qu'une seule observation $\boldsymbol{y}$ pour estimer l'angle d'arrivée. Lorsqu'il n'y a pas de bruit, cette unique mesure suffit parfaitement à déterminer l'angle d'arrivée puisqu'on a alors:

$$\boldsymbol{y} = x\,\boldsymbol{a}(\omega) = x \begin{bmatrix} 1 \\ e^{-j\omega} \\ \vdots \\ e^{-j(m-1)\omega} \end{bmatrix}$$

pour une antenne linéaire uniforme. Si on divise le vecteur $\boldsymbol{y}$ par sa première composante, on obtient un vecteur directionnel pur:

$$\boldsymbol{y}^d = \begin{bmatrix} 1 \\ e^{-j\omega} \\ \vdots \\ e^{-j(m-1)\omega} \end{bmatrix}$$

Il est alors facile d'obtenir l'angle d'arrivée ω. Par contre, dès que nous sommes en présence de bruit, cette simplification n'est plus possible: $\boldsymbol{y} = x\boldsymbol{a}(\omega) + \boldsymbol{b}$ donnerait:

$$\boldsymbol{y}^d = \frac{1}{x + b_1} \begin{bmatrix} x + b_1 \\ xe^{-j\omega} + b_2 \\ \vdots \\ xe^{-j(m-1)\omega} + b_m \end{bmatrix}$$

ce qui montre comment le bruit fausse alors complètement le déroulement des méthodes de Wells.
Au contraire, les méthodes ESPRIT et MUSIC, le Maximum de Vraisemblance, ainsi que la méthode de Martin *et. al.*, nécessitent l'utilisation de *plusieurs* observations $\boldsymbol{y}_t,\ t \in [1, N]$. En effet, pour les méthodes de Haute Résolution, l'estimation par le calcul de la matrice de covariance

$$\hat{R} = \frac{1}{N} \sum_{t=1}^{N} \boldsymbol{y}_t \boldsymbol{y}_t^H$$

montre que plus N est grand, meilleure sera l'estimation $\hat{R}$ de la vraie matrice de covariance, et meilleure sera l'estimation des angles d'arrivée. En effet, il apparaît clairement que plus le nombre d'observations N est important, plus la matrice estimée $\hat{R}$ est proche de la vraie valeur R de la matrice de covariance des signaux. Cette remarque vaut également pour la méthode du Maximum de Vraisemblance (*cf* équation (70)), et la méthode entièrement connectée (*cf* équation (95)), qui utilisent des moyennages sur $\boldsymbol{y}_t$ pour diminuer l'influence du bruit. Il est d'ailleurs intéressant de mentionner ici un résultat obtenu par Stoica et Nehorai dans [24]. Pour n ondes arrivant sur une antenne linéaire de m capteurs dont on dispose de N observations, la borne de Cramer-Rao asymptotique (pour m et N grands) pour la variance de l'erreur sur ω est:

$$CRB = \frac{6\sigma^2}{m^3 N} \begin{bmatrix} \frac{1}{P_1} & & 0 \\ & \ddots & \\ 0 & & \frac{1}{P_n} \end{bmatrix}$$

où P_i est la puissance (variance) du signal i et σ^2 la puissance (variance) du bruit. Comme on le voit, il est intéressant de pouvoir disposer de nombreuses observations $\boldsymbol{y}_t,\ t \in [1, N]$ plutôt que de n'en utiliser qu'une seule, comme chez Wells, d'autant plus que le bruit est forcément présent dans tout scénario réaliste.

De ces remarques, on peut essayer de tirer des idées pour construire un perceptron à trois couches permettant d'estimer les angles d'arrivée de plusieurs signaux bruités arrivant sur une antenne.

4 Approche par le PMC: vers une nouvelle idée

4.1 Introduction

Les remarques observées jusqu'ici vont être reprises une par une pour en tirer les grandes lignes qui nous permettrons de construire un perceptron dédié au problème d'Angles d'Arrivée[2].

- L'idée de procéder par secteurs ne semble pas porteuse de grands espoirs du point de vue efficacité, parce que l'on se limite forcément à un seuil de résolution qui risque, si on cherche à le diminuer, de coûter cher en temps de calcul. On pourrait bien sûr penser à interpoler ensuite entre deux secteurs, mais il semble plus simple de chercher à estimer l'angle θ de manière directe. La première tendance serait donc d'apprendre à un perceptron multicouche (PMC) à fournir une estimation des θ_k sur sa couche de sortie. Cette solution possède en outre l'avantage d'offrir des sorties qui sont des fonctions continues en tout point, au contraire de l'approche par secteurs, où un angle qui passait (par exemple) de 0^{o-} à 0^{o+} changeait de secteur.

- Cette première idée peut être poussée plus loin en notant que pour alléger la tâche confiée au PMC, il est préférable d'inclure dans des étapes de pré- ou de post-traitement, toutes les opérations "déterministes" ou calculatoires que l'on sait devoir être effectuées *de toute façon* par l'ensemble du système d'estimation. Ainsi, pour un réseau composé de deux doublets séparés par une distance Δ, tel que celui utilisé pour l'algorithme ESPRIT, on sait d'après l'équation (47) que la connaissance de θ_k est équivalente à celle de $\gamma_k = \frac{2\pi\nu_0\Delta\sin(\theta_k)}{c}$. De même, pour un réseau linéaire uniforme, on sait d'après l'équation (21) que la connaissance de θ_k est équivalente à celle de $\omega_k = \pi\sin\theta_k$. Or on a vu que si θ_k est bien la véritable inconnue, c'est d'abord ω_k (ou γ_k selon le cas), qui est obtenue par l'estimateur avant d'en déduire θ_k (*cf* MUSIC équation (42), ESPRIT équations (46) et (55), Maximum de Vraisemblance équations (19), (63) et (69), Rastogi équation (93), et Martin équation (97)). En conséquence, on peut concevoir un perceptron qui estimera en sortie les ω_k au lieu des θ_k. De cette façon, on allège

[2]La méthode présentée dans la suite de ce Chapitre a été en partie publiée dans [4] et [18]

le calcul demandé au PMC (en effet, on "linéarise" une partie du traitement neuronal en ôtant l'inversion de sinus), tout en garantissant l'équivalence entre ω_k et θ_k (d'après l'équation (21)). Notons que pour d'autres types d'antennes correspond une équation semblable à l'équation (21), et permettant de la même façon d'enlever une partie de la non-linéarité dans le calcul demandé au PMC. Par exemple, pour une antenne formée de doublets de capteurs, comme celle utilisée dans l'algorithme ESPRIT, c'est l'équation (47) qui remplace l'équation (21).

- On a vu que la méthode Wells n'était pas très robuste au bruit, en particulier parce qu'elle ne prenait en compte qu'une seule observation $\boldsymbol{y}$ en pied d'antenne. Au contraire, les autres méthodes, qui sont moins sensibles au bruit, utilisent un nombre N d'observations le plus grand possible, ceci afin de réduire la variance de l'erreur d'estimation. Toutefois, ces observations ne sont pas utilisées directement, mais sont "moyennées" avec ou sans pondération pour former une matrice où l'influence du bruit est théoriquement moindre, du fait des hypothèses de non-corrélation du bruit avec lui-même, et de non-corrélation du bruit avec les signaux. On forme donc la matrice de "moyennage" V:

$$V = \frac{1}{N} \sum_{t=1}^{N} \boldsymbol{y}_t U \boldsymbol{y}_t{}^H$$

où U est une matrice de pondération éventuelle.

La matrice U vaut l'identité pour les algorithmes MUSIC et ESPRIT de base (il existe par ailleurs des versions pondérées de ces algorithmes, comme par exemple [22] pour MUSIC). Lorsque $U = Id$, V est alors tout simplement l'estimation de la matrice de covariance des signaux). Pour l'approche Rastogi, U fait appel à des matrices de projection sur des vecteurs directionnels (voir les équations (95)). La manière la plus simple d'employer les $\boldsymbol{y}_t$, $t \in [1, N]$ pour l'estimation des angles d'arrivée semble donc être celle consistant à calculer l'estimée de la matrice de covariance. On peut souligner que la connaissance de la matrice de covariance est en principe suffisante pour estimer les angles d'arrivée, puisque c'est la seule donnée qu'utilisent MUSIC et ESPRIT. Toutes ces considérations nous poussent à choisir l'estimation de la matrice de covariance

$$\hat{R} = \frac{1}{N} \sum_{t=1}^{N} \boldsymbol{y}_t \boldsymbol{y}_t{}^H$$

comme donnée d'entrée du PMC.

- Le problème de l'estimation des angles d'arrivée n'est pas un problème simple, surtout lorsque l'on doit estimer l'angle de plusieurs sources, et que

l'environnement est très bruité. Il est donc clair que le problème est loin d'être linéaire. Par conséquent, le perceptron que nous allons construire doit se composer d'une couche d'entrée, d'une couche de sortie, et d'*au moins* une couche cachée. En d'autres termes, ce perceptron doit comporter au moins trois couches (rappelons que malgré des apparences trompeuses, les perceptrons de Wells comportent au plus 2 couches).

- En ce qui concerne la base d'apprentissage du PMC, on peut tenir la même remarque concernant l'approche de Wells vis-à-vis du bruit: Wells n'inclut aucun bruit dans ses bases d'apprentissage, et constate que le comportement de ses méthodes en présence de bruit (phases de test) se dégrade de manière non négligeable. Comme d'autre part les scénarios réalistes comportent forcément du bruit, il semble judicieux d'inclure du bruit dans les données d'apprentissage.

Utilisant ces remarques, nous pouvons maintenant détailler la structure du perceptron qui nous permettra d'estimer les angles d'arrivée.

4.2 Structure du Réseau

Pour un jeu d'observations $\boldsymbol{y}_t,\ t \in [1, N]$, nous allons d'abord, comme dans une approche de Haute Résolution telle que MUSIC, calculer une estimation de la matrice de covariance:

$$\hat{R} = \frac{1}{N} \sum_{t=1}^{N} \boldsymbol{y}_t \boldsymbol{y}_t{}^H$$

Or $\boldsymbol{y}_t$ est un vecteur complexe de longueur le nombre de capteurs; la matrice $\hat{R}$ est donc carrée, hermitienne de dimension $m \times m$. Elle peut se décomposer comme:

$$\hat{R} = A + jB$$

où A et B sont des matrices réelles symétriques, et la diagonale de B est formée de zéros. De ce fait, A comporte en tout

$$\frac{m^2 - m}{2} + m = \frac{m(m+1)}{2}$$

valeurs réelles indépendantes, et B en contient

$$\frac{m^2 - m}{2} = \frac{m(m-1)}{2}$$

Au total, $\hat{R}$ contient

$$\frac{m(m+1)}{2} + \frac{m(m-1)}{2} = m^2$$

valeurs réelles indépendantes. Pour une antenne de m capteurs, la couche d'entrée du perceptron comportera donc m^2 unités. On a choisi par ailleurs de chercher à obtenir les ω_k en sortie du PMC. Par conséquent la couche de sortie comportera autant d'unités qu'il y aura d'angles d'arrivée, c'est-à-dire n. Enfin, le perceptron doit comporter *au moins* trois couches. Limitons-nous à trois couches au départ, quitte à passer à quatre couches si les résultats obtenus semblent médiocres. La structure du perceptron ainsi défini est représentée figure (15).

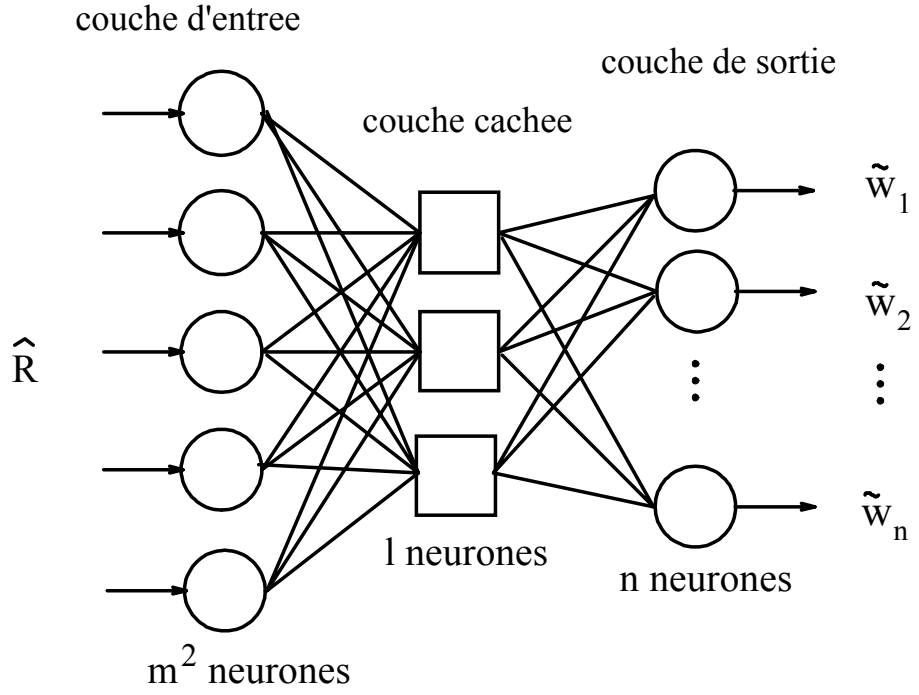

Figure 15: *Structure du perceptron à trois couches chargé d'estimer les* ω_k

Quelques remarques importantes peuvent être faites au sujet du nombre de neurones de ce réseau:

- Pour une antenne donnée, le nombre m de capteurs est constant. Le nombre de neurones de la couche d'entrée est donc fixe pour une antenne donnée.

- Le nombre de neurones de la couche cachée ne peut pas être déterminé par calcul: en général, le nombre de neurones de la couche cachée est choisi de manière à réduire le nombre (souvent important) des paramètres d'entrée, afin de ne conserver que des données essentielles au problème. En effet, garder un nombre de neurones important sur la couche cachée peut conduire à un phénomène de sur-apprentissage, c'est-à-dire que le perceptron "apprend" le bruit contenu dans la base d'apprentissage: ce phénomène peut survenir lorsque le rapport entre le nombre d'exemples de la base d'apprentissage et le nombre de neurones de la couche cachée, est trop faible. D'un autre côté, il ne faut pas réduire la dimensionnalité de l'espace généré par la couche cachée au point de ne plus pouvoir *apprendre* la base d'apprentissage. Pour le problème auquel on s'intéresse, le nombre l de neurones de la couche cachée doit être au moins égal au nombre d'angles d'arrivée, puisque ces données sont indépendantes. On a donc

  $$m^2 > l > n$$

  et puisque d'après l'hypothèse 3 (équation (10)) on a $m > n$, il semble intéressant de choisir l de l'ordre de m.

- Le nombre de neurones de la couche de sortie est n. Or, pour une antenne de m capteurs donnée, le nombre d'angles d'arrivée physiquement décelables par l'antenne peut varier de 1 à $m-1$ (*cf* équation (10) également). On doit donc construire $m-1$ perceptrons d'après le modèle de la figure (15), chacun étant plus précisément dédié à l'estimation de n angles d'arrivée, pour $n \in [1, m-1]$. En effet, comme on a supposé au départ (*cf* hypothèse 4) que l'on connaissait le nombre d'angles d'arrivée n, ou au moins son estimation $\hat{n}$, il paraît logique de séparer le problème de l'estimation du nombre d'angles d'arrivée, du problème de l'estimation des angles d'arrivée eux-mêmes. Rappelons que les approches MUSIC, ESPRIT et MV, qui sont en cela les algorithmes de référence, supposent eux-aussi que $\hat{n}$ est connu (jusqu'ici, seule l'aproche de Rastogi s'affranchit de cette condition). C'est pourquoi, au lieu d'utiliser un seul réseau comportant $m-1$ sorties et utilisant un codage spécial (qui sans doute compliquerait encore le problème) pour signaler les $m-1-\hat{n}$ sorties qui n'estiment pas d'angle d'arrivée (voir figure (16)), on préfère réduire la complexité du problème en séparant les sous-problèmes indépendants.

Ainsi, on a réduit la complexité de l'approche en proposant la construction de $m-1$ perceptrons à trois couches comportant m^2 entrées, environ m neurones sur la couche cachée, et respectivement n neurones, $n \in [1, m-1]$ sur la couche de sortie. Ces $m-1$ réseaux seront entraînés séparément, sur $m-1$ bases d'apprentissage distinctes, puisque chaque base d'apprentissage est restreinte à la représentation de n sources arrivant sur l'antenne.

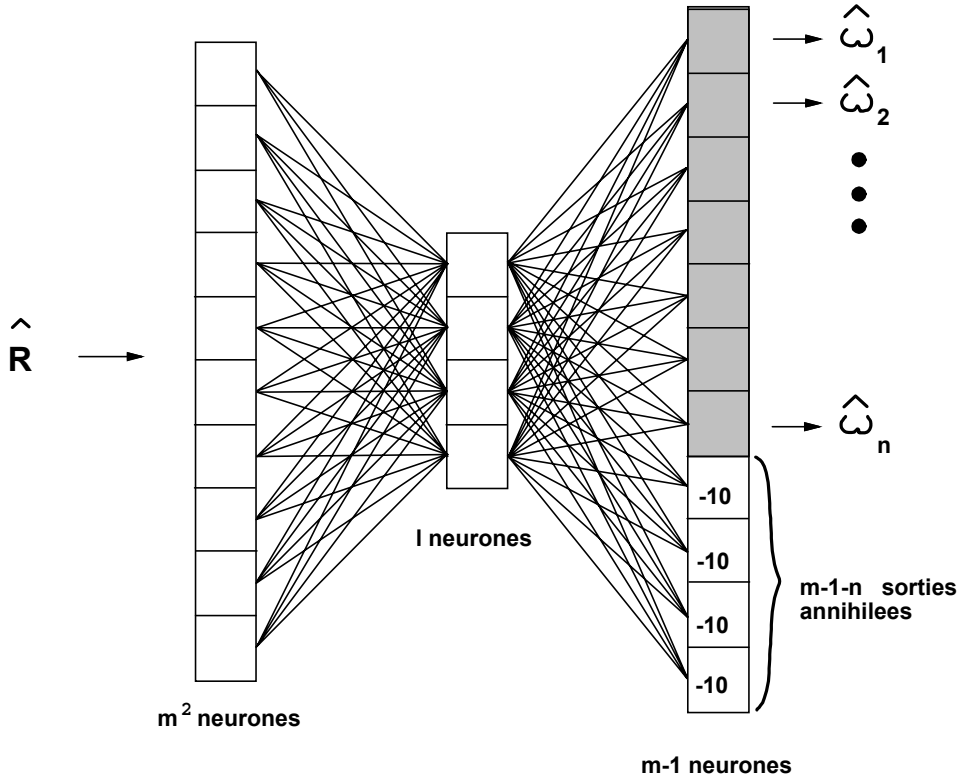

Figure 16: *Perceptron à $m-1$ sorties dont n seulement sont utilisées: le PMC doit estimer à la fois n et les ω_k, $k \in [1, n]$*

4.3 Données d'entrée

Les données d'entrée du réseau sont les m^2 valeurs réelles indépendantes représentant l'estimation de la matrice de corrélation $\hat{R}$. Rappelons que $\hat{R}$ a été calculée en utilisant les N observations dont on dispose:

$$\hat{R} = \frac{1}{N} \sum_{t=1}^{N} \boldsymbol{y}_t \boldsymbol{y}_t{}^H$$

où $\boldsymbol{y}_t$ est modélisé comme suit:

$$\boldsymbol{y}_t = A(\Omega)\boldsymbol{x}_t + \boldsymbol{b}_t \tag{107}$$

Nous cherchons à estimer le vecteur paramètre $\Omega = [\omega_1, \omega_2, ..., \omega_n]$ qui n'intervient qu'au travers de la matrice $A(\Omega)$. Pour une géométrie d'antenne donnée, $A(\Omega)$ ne dépend que de Ω. Par contre, pour un même vecteur paramètre Ω, les vecteurs $\boldsymbol{x}_t$ et $\boldsymbol{b}_t$ peuvent prendre n'importe quelle valeurs.

On cherche à estimer Ω à partir de $\hat{R}$, et ce, quelles que soient les valeurs de

$\boldsymbol{x}_t$ et $\boldsymbol{b}_t$. Il nous faut donc rendre $\hat{R}$ robuste par rapport aux variations de $\boldsymbol{x}_t$ et $\boldsymbol{b}_t$ lorsque Ω reste constant. De (107) on déduit:

$$\begin{aligned}\hat{R} &= \frac{1}{N}A(\Omega)\sum_{t=1}^{N}\boldsymbol{x}_t\boldsymbol{x}_t{}^H A^H(\Omega) + \hat{\sigma}^2 I \\ &= A(\Omega)\hat{\mathcal{S}}A^H(\Omega) + \hat{\sigma}^2 I\end{aligned}$$

où $\hat{\mathcal{S}} = \frac{1}{N}\sum_{t=1}^{N}\boldsymbol{x}_t\boldsymbol{x}_t{}^H$ est la matrice de covariance des sources arrivant sur l'antenne.

Pour éviter de compliquer inutilement les calculs qui suivent, on va supposer pour le moment que le bruit est nul et que les sources sont totalement décorrélées entre elles. Pour un nombre d'échantillons N infini, on a alors une matrice R parfaite, telle que

$$R = A(\Omega)\mathcal{S}A^H(\Omega)$$

où

$$\mathcal{S} = \begin{pmatrix} P_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & P_n \end{pmatrix} \tag{108}$$

soit, après développement de $A(\Omega)$:

$$R = \sum_{k=1}^{n}\begin{pmatrix} P_k & P_k e^{j\omega_k} & P_k e^{j2\omega_k} & \dots & P_k e^{j(m-1)\omega_k} \\ P_k e^{-j\omega_k} & P_k & P_k e^{j\omega_k} & \dots & P_k e^{j(m-2)\omega_k} \\ P_k e^{-j2\omega_k} & P_k e^{-j\omega_k} & P_k & \dots & P_k e^{j(m-3)\omega_k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_k e^{-j(m-1)\omega_k} & P_k e^{-j(m-2)\omega_k} & P_k e^{-j(m-3)\omega_k} & \dots & P_k \end{pmatrix}$$

On peut remarquer que l'influence de chaque ω_k est pondérée par la "puissance" P_k de la source qui lui est associée. Si la cible i se rapproche, on imagine aisément qu'elle va réfléchir une plus grande quantité d'énergie: P_i va donc croître. La variation de P_i va faire varier les coefficients de la matrice R, alors qu'en réalité les angles d'arrivée ne changent pas forcément. On comprend donc bien qu'il est nécessaire de "normaliser" la matrice R afin de s'affranchir des fluctuations éventuelles de la puissance des sources. Nous choisirons de normaliser la matrice R par la somme des n puissances des sources.

En pratique, la matrice $\hat{R}$ est estimée pour des sources pas forcément décorrélées, et il se trouve du bruit dans le calcul de $\hat{R}$. La normalisation se fera donc en divisant la matrice $\hat{R}$ par la moyenne de ses éléments (réels) diagonaux:

$$\hat{R}^{nor} = \frac{1}{r_{moy}}\hat{R}$$

où

$$r_{moy} = \frac{1}{m} \sum_{i=1}^{m} \left(\hat{R}\right)_{ii}$$

Finalement, les entrées du perceptron à trois couches sont définies comme les m^2 valeurs réelles indépendantes formant la matrice $\hat{R}^{nor}$, telle que

$$\hat{R}^{nor} = \frac{m}{\sum_{i=1}^{m} \left(\hat{R}\right)_{ii}} \hat{R} \tag{109}$$

4.4 Apprentissage

La création d'une base d'apprentissage appropriée passe d'abord par l'étude du type de signaux que l'on aura à traiter. En principe, les signaux seront du type $\boldsymbol{y}_t,\ t \in [1, N]$, où

$$\boldsymbol{y}_t = A(\Omega)\boldsymbol{x}_t + \boldsymbol{b}_t$$

avec

- $\boldsymbol{b}_t$ un vecteur $m \times 1$ de bruit, composé de variables complexes gaussiennes de moyenne 0 et de variance σ^2; ce vecteur est non corrélé avec lui-même ni avec les sources.

- $\boldsymbol{x}_t$ le vecteur $n \times 1$ des sources. Par exemple, dans le domaine radar, à cause de phénomènes de scintillement des sources [26], ce vecteur n'est pas constant. On peut le modéliser par un vecteur de variables complexes gaussiennes de moyenne 0 et de variance sa puissance $P_i,\ i \in [1, n]$. Notons que d'autres modèles existent pour les sources (voir [26] pour plus de détails), comme par exemple des constantes, des distributions uniformes, des distributions de Rice, etc. Cependant, comme la méthode employée ne reçoit que de l'information au second ordre (la matrice de covariance des signaux arrivant sur l'antenne), toute information d'ordre supérieur sur les densités de probabilité des sources (exemple: moments d'ordre supérieur à 2) est perdue pour le système, donc tout se passe comme si les sources étaient gaussiennes, de variance leur puissance P_i. Notons que les sources peuvent être (partiellement ou totalement) corrélées entre elles.

- $A(\Omega)$ la matrice directionnelle caractérisant les angles d'arrivée: $A(\Omega) = [\boldsymbol{a}(\omega_1), ...\boldsymbol{a}(\omega_n)]$. Pour une antenne linéaire uniforme, $\boldsymbol{a}(\omega_k)$ s'écrit $[1, e^{-j\omega_k}, ..., e^{-j(m-1)\omega_k}]^T$ où $\omega_k = \pi \sin\theta_k$, et θ_k peut varier de -90^o à 90^o.

Pour une antenne donnée (géométrie donnée) de m capteurs, nous avons donc à construire $m-1$ perceptrons à trois couches (m^2 entrées, l neurones cachés) paramétrés par leur nombre de neurones de sortie n, $n \in [1, m-1]$. A chacun de ces perceptrons on associe une base d'apprentissage composée de couples entrée/sortie désirée. Ces couples d'exemples comportent, pour la partie "entrée", m^2 valeurs réelles créées à partir de la matrice de covariance normalisée (*cf* équation (109)), et pour la partie "sortie désirée", n paramètres ω_k, $k \in [1, n]$, qui sont ceux que l'on a utilisés pour construire la matrice $\hat{R}^{nor}$ qui leur est associée. Les paramètres de la base d'apprentissage sont donc:

- σ^2, la variance de chaque variable complexe appartenant au vecteur bruit $\boldsymbol{b}_t$.

- $\mathcal{S}$, la matrice de covariance des sources, dont la diagonale est formée des P_i, $i \in [1, n]$, la variance de chaque variable complexe appartenant au vecteur source $\boldsymbol{x}_t$. Le rapport Signal sur Bruit (SNR pour "Signal to Noise Ratio" en Anglais) vaut:

  $$\forall\, i \in [1, n], \quad SNR_i = 10 \log \frac{P_i}{\sigma^2}$$

  puisque par définition le SNR est le rapport (en échelle logarithmique) de la puissance du signal sur la puissance du bruit. Notons que $\mathcal{S}$ n'est diagonale que si les sources sont non corrélées.

- N, le nombre d'observations par expérience.

- un schéma de variation des angles θ_i, $i \in [1, n]$, de telle façon que la base d'apprentissage soit représentative des exemples que nous pourrions avoir à traiter par la suite.

Il est clair que plus la base d'apprentissage sera diverse et tiendra compte de tous les cas de corrélation et de niveaux de puissance possibles, meilleure devrait être la généralisation. D'un autre côté, plus on mettra d'exemples dans cette base, plus elle sera lourde à manier, et surtout, plus l'apprentissage sera long. N'oublions pas que lorsque l'on se trouve en présence de *plusieurs* ($n > 1$) angles d'arrivée, le nombre d'exemples de la base varie par puissance de n, ceci afin de prendre en compte tous les cas angulaires possibles.

Aussi, dans un souci d'allègement de procédure, seuls les cas comportant des sources non corrélées seront inclus dans la base d'apprentissage. De ce fait, on peut d'ores et déjà simplifier la matrice de covariance des sources puisqu'elle s'exprime alors sous forme diagonale, telle que dans l'équation (108).

Les vecteurs sources $\boldsymbol{x}_t$ utilisés pour construire la base d'apprentissage sont donc composés de n variables aléatoires indépendantes, gaussiennes de moyenne 0 et de variance P_i. Dans le même souci de simplification (ne pas multiplier inconsidérément le nombre d'exemples de la base), le même niveau de puissance sera choisi pour chaque source:

$$\forall\ (i,j) \in [1,n]^2, \quad SNR_i = SNR_j$$

que l'on notera désormais SNR. La simplification peut se poursuivre en identifiant $\mathcal{S}$ à la matrice identité (car on fait une normalisation de $\hat{R}$) et en écrivant

$$SNR = 10 \log \frac{1}{\sigma^2}$$

De ce fait, une fois que l'on a choisi un SNR convenable pour la base, la variance du bruit est calculée par

$$\sigma^2 = \frac{1}{10^{SNR/10}}$$

Quant au choix de la valeur du SNR elle-même, s'il faut effectivement que la base d'apprentissage tienne compte du bruit, il ne faut pas non plus noyer le signal par le bruit. L'adjonction du bruit permet au perceptron de ne pas "sur-apprendre" la base d'apprentissage, et de se ménager ainsi un certain "potentiel de généralisation". D'un autre côté, il ne faut pas prendre un SNR trop faible sur la base d'apprentissage, afin que le PMC puisse quand même induire des liens de causalité entre ses entrées et ses sorties. On propose donc de prendre un SNR de $25dB$, sachant bien sûr que rien n'empêchera par la suite de tester le perceptron concerné avec toutes autres valeurs de SNR.

Le détail de la génération des vecteurs $\boldsymbol{x}_t$ et $\boldsymbol{b}_t$ est donné comme suit: à chaque instant t, $t \in [1,N]$, le vecteur $\boldsymbol{x}_t$ (respectivement $\boldsymbol{b}_t$) est généré comme une suite indépendante de n (respectivement m) valeurs complexes dont la partie réelle et la partie imaginaire sont gaussiennes et indépendantes, de moyenne 0 et de variance $\frac{1}{2}$ (respectivement $\frac{\sigma^2}{2}$). En effet, pour que la variance d'une variable aléatoire complexe $u + jv$ soit σ^2 (par exemple), et sachant que u et v ont la même distribution, la variance de u et de v vaut $\frac{\sigma^2}{2}$ puisque

$$var_{u+jv} = var_u + var_v = 2var_u = 2var_v$$

Pour chaque expérience (chaque couple entrée/sortie désirée), la matrice $\hat{R}^{nor}$ est estimée à partir de N observations. Pour la base d'apprentissage, le nombre

d'observations par expérience ne doit pas être trop élevé (pour la même raison que le SNR ne doit pas être trop grand: il faut éviter tout "sur-apprentissge" de cette base), ni trop faible, pour que le PMC puisse reconnaître une certaine "régularité" de la matrice $\hat{R}^{nor}$, afin d'en extraire les sorties désirées (les ω_k). On choisira $N = 40$ pour les bases d'apprentissage.

Enfin il ne reste plus qu'à définir le schéma de distribution des $n-$uplets d'angles. Tout d'abord, afin qu'il n'y ait pas répétition, on ne génèrera que des exemples où l'inégalité $\theta_1 < \theta_2 < ... < \theta_n$ est respectée. Remarquons que cette inégalité entraîne $\omega_1 < \omega_2 < ... < \omega_n$ en sortie. Cela nous permet d'économiser sur le nombre d'exemples de la base d'apprentissage, et aussi cela nous permet d'alléger la tâche du perceptron, puisqu'ainsi le tri des angles par ordre de grandeur est déjà effectué. Nous limiterons enfin l'éventail décrit par les angles θ à l'intervalle $[-60^o, 60^o]$, parce qu'avec les antennes linéaires uniformes, la variance de l'erreur d'un estimateur croît lorsque θ croît. En effet, rappelons que pour une antenne linéaire uniforme, les estimateurs étudiés jusqu'ici estiment d'abord $\omega = \pi \sin\theta$ avant d'en déduire θ. Or l'erreur quadratique moyenne expérimentalement obtenue sur ω (erreur commise par l'estimateur) est:

$$\begin{aligned} e^2_{(\hat{\omega}-\omega)} &= E\{(\hat{\omega}-\omega)^2\} \\ &= E\{\Big(\pi\sin\hat{\theta} - \pi\sin\theta\Big)^2\} \\ &= \pi^2 E\{\Big(\sin\hat{\theta} - \sin\theta\Big)^2\} \end{aligned}$$

L'égalité trigonométrique

$$\sin a - \sin b = 2\cos\left(\frac{a+b}{2}\right)\sin\left(\frac{a-b}{2}\right)$$

conduit à

$$e^2_{(\hat{\omega}-\omega)} = \pi^2 E\{\left(2\cos\left(\frac{\hat{\theta}+\theta}{2}\right)\sin\left(\frac{\hat{\theta}-\theta}{2}\right)\right)^2$$

et si l'on suppose que l'estimateur n'est pas trop médiocre, on peut s'attendre à avoir $\hat{\theta}$ proche de θ. On peut donc faire les approximations

$$\begin{aligned} \cos\left(\frac{\hat{\theta}+\theta}{2}\right) &\simeq \cos\theta \\ \text{et} \qquad \sin\left(\frac{\hat{\theta}-\theta}{2}\right) &\simeq \frac{\hat{\theta}-\theta}{2} \quad \text{(en radians)} \end{aligned}$$

On obtient alors

$$e^2_{(\hat{\omega}-\omega)} \simeq \pi^2 \cos^2\theta E\{(\hat{\theta}-\theta)^2\}$$

Soit finalement

$$e^2_{(\hat{\theta}-\theta)} \simeq \frac{1}{\pi^2 \cos^2\theta} e^2_{(\hat{\omega}-\omega)} \tag{110}$$

Cette dernière égalité montre que pour une erreur sur $\hat{\omega}$ constante, l'erreur sur $\hat{\theta}$ croît avec θ. En particulier, lorsque θ tend vers $\pm 90^o$, le $\cos^2\theta$ tend vers 0 et l'erreur sur $\hat{\theta}$ tend vers l'infini. Il est donc préférable, si l'on désire par exemple décrire les 360^o de l'espace, d'utiliser trois antennes linéaires disposées en triangle et dédiée chacune à l'étude de l'espace $[-60^o, 60^o]$ qui lui fait face.

Il est possible de mettre en exergue un dernier paramètre de la base d'apprentissage: c'est le nombre d'expériences L par n-uplet d'angles. En effet, puisque l'on se trouve en présence de bruit sur les capteurs et que les sources ont des puissances instantanées variables, deux matrices $\hat{R}^{nor}$ construites pour un même n-uplet d'angles et un même nombre d'observations N peuvent s'avérer être assez différentes du fait du caractère aléatoire de certaines variables. Aussi, lorsque le nombre d'exemples total de la base n'est pas trop grand, on peut envisager d'employer $L = 2$ à 5 expériences par n-uplet d'angles. Notons ici qu'il est important de ne pas confondre N, le nombre d'observations (échantillons) par expérience, avec L, le nombre d'expériences par n-uplet d'angles. Lorsque N croît, la matrice de covariance estimée $\hat{R}$ se rapproche de plus en plus de la vraie matrice de covariance R. Lorsque L croît, c'est la richesse de la base d'apprentissage qui croît, puisque $L > 1$ procure *plusieurs* matrices $\hat{R}$ différentes pour un même n-uplet d'angles, c'est-à-dire, pour le réseau, plusieurs vecteurs d'entrée différents pour un même vecteur de sortie.

4.5 Conclusion

Dans cette partie, nous avons imaginé et construit un estimateur d'angles d'arrivée basé sur l'emploi de perceptrons à trois couches. Les données d'entrée de ces perceptrons sont calculées à partir des m^2 éléments réels indépendants de la matrice de covariance "normalisée" des signaux reçus sur l'antenne. La couche cachée contient l neurones, avec l de l'ordre de m (le nombre de capteurs). Le PMC est entraîné à fournir $\omega_k = \pi \sin\theta_k$ ($k \in [1, n]$) sur le k-ième neurone de sa couche de sortie (qui en compte n au total), avec les ω_k rangés par ordre croissant.

En pratique les perceptrons apprennent bien à retrouver les angles d'arrivée ω_k , et les résultats obtenus sur la base d'apprentissage sont bons. Cependant, on a

vu qu'il aurait été trop lourd et trop long de faire varier tous les paramètres (SNR, N, corrélation entre sources, ...) sur la base d'apprentissage. De ce fait, même si en phase de test les estimations fournies par ces perceptrons sont assez bonnes et bien plus rapides[3] elles restent souvent moins précises que celles fournies par les méthodes plus "classiques" de Haute-Résolution.

5 Raffinement de la méthode

5.1 Introduction

On a vu précédemment qu'un perceptron à trois couches pouvait, après entraînement, trouver une estimation des angles d'arrivée sur un réseau de capteurs, à partir des valeurs contenues dans la matrice de covariance des signaux reçus sur ces capteurs. Malgré la grande rapidité de l'estimateur en phase de test (en effet, le nombre de multiplications demandé est très faible, à savoir environ m^3 pour une antenne de m capteurs), ses performances ne dépassent pas celles des estimateurs (pourtant sous-optimaux) MUSIC et ESPRIT.

D'un autre côté, on sait que le meilleur estimateur des angles d'arrivée est le Maximum de Vraisemblance. Cependant, son emploi est rendu très lourd par la phase obligatoire de test de *toutes* les valeurs possibles de *tous* les paramètres, ceci afin d'en trouver l'extremum global: en théorie le temps de calcul du Maximum de Vraisemblance est infini. Nous proposons [4], [18] ci-dessous une approche neuronale basée sur la mise en œuvre de contraintes sur les poids d'un réseau de neurones complexes. Ce réseau, nous le verrons, optimise un critère équivalent à celui du Maximum de Vraisemblance, tout en étant beaucoup plus rapide.

5.2 Réseau complexe à structure contrainte

5.2.1 Rappels sur le problème

Avant de commencer à décrire plus précisément la structure choisie pour le réseau, rappelons que nous travaillons ici avec des signaux modélisés par l'équation

[3] Pour comparaison, l'Annexe A indique le nombre de multiplications complexes nécessaires pour la mise en œuvre de MUSIC, d'ESPRIT, du MV et du PRI

$$\boldsymbol{y}_t = A(\Omega)\boldsymbol{x}_t + \boldsymbol{b}_t, \qquad t \in [1, N]$$

où $A(\Omega) = [\boldsymbol{a}(\omega_1), ..., \boldsymbol{a}(\omega_n)]$ est la matrice directionnelle associée au réseau et au vecteur paramètre recherché $\Omega = [\omega_1, ..., \omega_n]^T$. Ainsi, puisque l'antenne possède m capteurs, les informations disponibles (à savoir les $\boldsymbol{y}_t$, $t \in [1, N]$) appartiennent à un espace de dimension m; toutefois, si on néglige le bruit $\boldsymbol{b}_t$, on peut observer que les $\boldsymbol{y}_t$ ne forment qu'un sous-espace de dimension n au plus, puisque d'après l'équation précédente les $\boldsymbol{y}_t$ sont créés à partir des n sources formant le vecteur $\boldsymbol{x}_t$. Si les sources ne sont pas complètement corrélées, c'est-à-dire si la matrice

$$R_x = E\{\boldsymbol{x}\boldsymbol{x}^H\}$$

est régulière, alors le sous-espace contenant les vecteurs $\boldsymbol{y}_t$ est de dimension n.

C'est à partir de cette remarque que nous allons construire un nouveau perceptron chargé d'estimer les Angles d'Arrivée des sources. Nous présenterons dans un premier temps toutes ses caractéristiques internes, et c'est ensuite seulement que nous montrerons dans quelle mesure ce perceptron peut estimer les paramètres recherchés.

5.2.2 Structure interne du réseau

D'après les remarques précédentes, on sait que les vecteurs $\boldsymbol{y}_t$ (de longueur m) sont contenus dans un sous-espace de dimension n. De ce fait, la dimensionnalité du problème nous incite à construire un réseau à trois couches tel que celui présenté figure (17). La première couche comporte m neurones et reçoit le vecteur $\boldsymbol{y}_t$ en entrée. Le réseau reçoit donc des valeurs complexes, aussi est-il formé de neurones et de poids tous complexes. La couche de sortie comporte aussi m neurones, et la couche cachée comporte n neurones. Puisque n est la dimensionnalité du problème, nous savons que même après réduction du nombre de neurones (passage de m à n neurones entre la couche d'entrée et la couche cachée), il est possible de retrouver en sortie une estimation du vecteur $\boldsymbol{y}_t$, car celui-ci appartient à un sous-espace de dimension n.

D'autre part, nous savons que le lien entre $\boldsymbol{x}_t$ et $\boldsymbol{y}_t$ est modélisé (en théorie) par:

$$\boldsymbol{y}_t = A(\Omega)\boldsymbol{x}_t$$

si on néglige le bruit additif $\boldsymbol{b}_t$. Pour une antenne linéaire uniforme, par exemple, la matrice $A(\Omega)$ s'écrit:

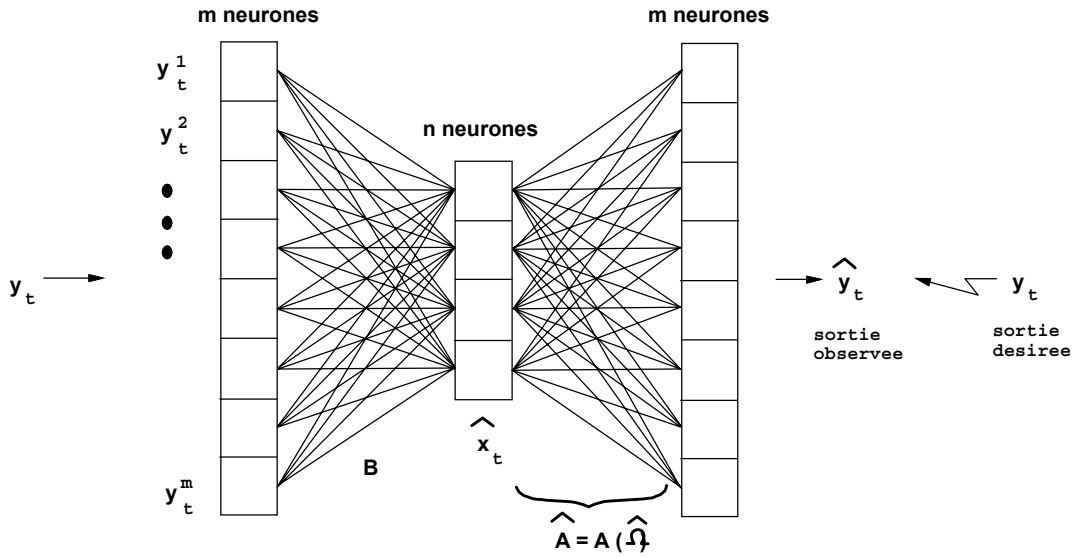

Figure 17: *Le perceptron complexe à structure contrainte*

$$A(\Omega) = \begin{pmatrix} 1 & 1 & ... & 1 \\ e^{-j\omega_1} & e^{-j\omega_2} & ... & e^{-j\omega_n} \\ \vdots & \vdots & \ddots & \vdots \\ e^{-j(m-1)\omega_1} & e^{-j(m-1)\omega_2} & ... & e^{-j(m-1)\omega_n} \end{pmatrix}$$

Or dans ce réseau, le lien entre la couche cachée et la couche de sortie se situe au niveau de la matrice des poids reliant ces deux couches. Nous allons donc *contraindre* les poids de la matrice de sortie $\hat{A}$ à suivre la forme d'une matrice directionnelle[4]:

$$\hat{A} = A(\hat{\Omega}) \tag{111}$$

où $\hat{\Omega} = [\hat{\omega}_1, ..., \hat{\omega}_n]^T$. Compte tenu de cette contrainte, en forçant la sortie à approcher au mieux le vecteur $\boldsymbol{y}_t$, on oblige le réseau à fournir une estimation de $\boldsymbol{x}_t$ sur sa couche cachée et une estimation des angles d'arrivée dans sa seconde couche de poids. En effet, on remarque que nous avons construit ce réseau de telle sorte qu'une partie de sa structure (entre la couche intermédiaire et la couche de sortie) corresponde exactement au modèle des signaux[5].

D'autre part nous choisissons d'utiliser des neurones ayant une fonction de transition égale à l'identité (c'est une fonction holomorphe), tels que celui présenté

[4]Notons ici que nous pouvons contraindre la matrice $\hat{A}$ de sortie à prendre la forme d'une matrice directionnelle, parce que nous *connaissons* la géométrie de l'antenne. Nous connaissons donc la forme analytique de la matrice directionnelle $A(\Omega)$, ce qui nous permet de contraindre $\hat{A}$ à la forme voulue.

[5]Ce modèle est pris en compte de manière implicite dans la structure du réseau

sur la figure (18). Par suite nous pouvons écrire le vecteur de sortie comme

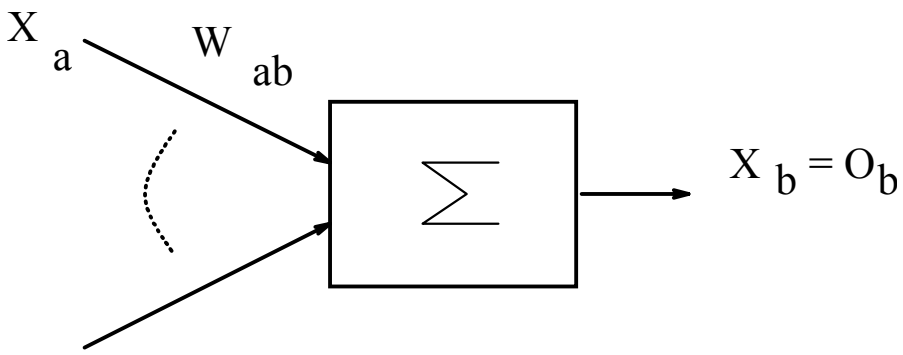

Figure 18: *Neurone complexe avec fonction de transition holomorphe égale à l'identité*

$$\forall t,\ t \in [1, N], \quad \hat{\boldsymbol{y}}_t = A(\hat{\Omega})\hat{\boldsymbol{x}}_t$$

où $\hat{\boldsymbol{x}}_t$ est le vecteur des neurones sur la couche cachée. Enfin, puisque nous forçons le réseau à retrouver en sortie le vecteur $\boldsymbol{y}_t$ reçu sur l'antenne, le critère d'erreur (l'erreur quadratique en sortie) du réseau est

$$e_{QM} = \frac{1}{N}\sum_{t=1}^{N}\frac{1}{2}||\boldsymbol{y}_t - \hat{\boldsymbol{y}}_t||^2 = \frac{1}{N}\sum_{t=1}^{N}\frac{1}{2}||\boldsymbol{y}_t - A(\hat{\Omega})\hat{\boldsymbol{x}}_t||^2 \tag{112}$$

où N est le nombre d'observations $\boldsymbol{y}_t$ dont on dispose. Il suffit alors de rétropropager cette erreur sur les matrices de poids du réseau par la règle du gradient (*cf* Chapitre 1 Section (2.4)), pour faire baisser l'erreur à chaque itération.

Remarque:

Contrairement à un perceptron multicouche "classique" en phase d'apprentissage, les poids de la couche de sortie ne sont pas libres: ils sont liés entre eux par la contrainte précédente $\hat{A} = A(\hat{\Omega})$. Seuls les $\hat{\omega}_k$ sont des paramètres "libres". A l'inverse, la matrice B des poids reliant la couche d'entrée à la couche cachée n'est pas contrainte.

Nous appellerons désormais Perceptron Complexe à Structure Contrainte le réseau proposé dans cette partie, et utiliserons parfois l'abréviation PCSC.

5.2.3 Le PCSC optimise un critère de vraisemblance

On a vu à la section 3.3 que l'estimateur du Maximum de Vraisemblance consistait

(équation (69)) en la minimisation de

$$e_{MV} = \frac{1}{N}\sum_{t=1}^{N} ||\boldsymbol{y_t} - \hat{A}\hat{\boldsymbol{x}}_t||^2$$

où $\hat{A}$ est une estimation de la matrice directionnelle, et $\hat{\boldsymbol{x}}_t$ une estimation des sources. En faisant remarquer que le vecteur $\hat{A}\hat{\boldsymbol{x}}_t$ appartient à $Vect\{\hat{A}\}$, le sous-espace généré par $\hat{A}$, on déduit que le meilleur choix pour $\hat{\boldsymbol{x}}_t$ est de prendre $\hat{A}\hat{\boldsymbol{x}}_t$ égal à la projection de $\boldsymbol{y_t}$ sur $Vect\{\hat{A}\}$. Cela permet de conclure que le critère du Maximum de Vraisemblance est équivalent à la minimisation de

$$e_{MV} = \frac{1}{N}\sum_{t=1}^{N} ||\boldsymbol{y_t} - P_{\hat{A}}\boldsymbol{y}_t||^2$$

En ce qui concerne le Perceptron Complexe à Structure Contrainte, on a construit le réseau de telle manière que la couche de sortie soit contrainte à fournir une estimation du vecteur $\boldsymbol{y_t}$ en sortie. L'erreur quadratique moyenne en sortie est donc

$$e_{QM} = \frac{1}{N}\sum_{t=1}^{N} \frac{1}{2}||\boldsymbol{y_t} - \hat{\boldsymbol{y}}_t||^2$$

D'autre part la structure de la matrice $\hat{A}$ des poids reliant la couche cachée à la couche de sortie a été *contrainte* à respecter la forme d'une matrice directionnelle. On a donc $\hat{A} = A(\hat{\Omega})$, où $\hat{\Omega}$ est le vecteur paramètre recherché. Enfin, la couche cachée comporte n neurones formant un vecteur appelé $\hat{\boldsymbol{x}}_t$, et si on choisit des neurones de sortie ayant une fonction de transition égale à l'identité, on a

$$\forall t,\ t \in [1, N], \quad \hat{\boldsymbol{y}}_t = A(\hat{\Omega})\hat{\boldsymbol{x}}_t$$

Nous devons donc minimiser la norme du vecteur différence entre $\boldsymbol{y}_t$ et $\hat{\boldsymbol{y}}_t$, sachant que $\hat{\boldsymbol{y}}_t$ appartient à $Vect\{A(\hat{\Omega})\}$. De la même façon que précédemment, le meilleur choix pour $\hat{\boldsymbol{y}}_t$ est alors la projection sur $A(\Omega)$ de $\boldsymbol{y}_t$. Nous en déduisons que le réseau minimise en sortie l'erreur quadratique moyenne

$$e_{QM} = \frac{1}{N}\sum_{t=1}^{N} \frac{1}{2}||\boldsymbol{y_t} - P_{A(\hat{\Omega})}\boldsymbol{y}_t||^2$$

ce qui démontre que le Maximum de Vraisemblance et le perceptron complexe à structure contrainte cherchent à trouver le minimum du même critère. De ce fait, ils conduisent en théorie à la même solution.

Le Perceptron Complexe à Structure Contrainte possède cependant un avantage certain sur le Maximum de Vraisemblance. En effet, procédant par descente de gradient (c'est l'apprentissage d'un perceptron), la minimisation qu'il utilise est

considérablement plus rapide (*cf* Annexe A) que celle du Maximum de Vraisemblance, qui doit tester toutes les hypothèses l'une après l'autre.

Il est à noter qu'après convergence du réseau (dans un minimum de l'hypersurface e_{QM}), il ne reste plus qu'à extraire les paramètres recherchés (c'est-à-dire les $\hat{\omega}_k$ formant le vecteur $\hat{\Omega}$) de la matrice des poids de sortie $\hat{A} = A(\hat{\Omega})$. Ainsi, contrairement à ce qui est l'habitude avec les perceptrons multicouches, la solution de notre problème ne se trouve pas sur la couche de sortie du réseau, mais dans les poids du réseau.

D'autre part, puisqu'au final $\hat{A} = A(\hat{\Omega})$ est une estimation de la vraie matrice directionnelle, la structure contrainte du réseau (*cf* schéma (17)) combinée à la minimisation de l'erreur en sortie (équation (112)) imposent que les composantes du vecteur $\hat{\boldsymbol{x}}_t$ de la couche cachée forment une estimation du vecteur source originel $\boldsymbol{x}_t$. Cette observation montre que la méthode basée sur le Perceptron Complexe à Structure Contrainte fournit plus d'informations de manière directe que les méthodes de Haute Résolution.

Remarque:

Comme on vient de le voir, le meilleur choix pour $\hat{\boldsymbol{y}}_t$ est la projection orthogonale de $\boldsymbol{y}_t$ sur $Vect\{A(\hat{\Omega})\}$, et ce choix correspond également à la méthode du Maximum de Vraisemblance. On ne gagnerait donc pas en performance en utilisant des neurones non-linéaires (c'est-à-dire des neurones utilisant une fonction de transition autre que l'identité), puisque toute projection orthogonale peut être réalisée par une matrice. Cela explique pourquoi nous utilisons des neurones à sortie linéaire dans le perceptron complexe à structure contrainte.

5.2.4 Propriétés du PCSC

La structure proposée pour le PCSC est génératrice de propriétés intéressantes et originales, à la fois par rapport à l'utilisation classique d'un réseau de neurones, et par rapport aux estimateurs usuellement utilisés en estimation d'AdA (MUSIC et ESPRIT). En effet, notons tout d'abord que le PCSC procède par minimisation d'un critère d'erreur en sortie. Cependant, contrairement à l'utilisation classique d'un perceptron, les informations qui nous intéressent ne se trouvent pas sur la couche de sortie, mais dans les poids. En outre, le réseau ne réalise pas d'apprentissage préalable, comme cela est l'habitude. Au contraire, l'algorithme d'apprentissage est réalisé, à chaque expérience, sur le jeu de données (les $\boldsymbol{y}_t$) dont on dispose à ce moment-là. A travers l'apprentissage, c'est plutôt la minimisation du critère de sortie que nous recherchons; en effet, pour minimiser l'erreur

$$e_{QM} = \frac{1}{N}\sum_{t=1}^{N}\frac{1}{2}||\boldsymbol{y_t} - \hat{\boldsymbol{y}}_{\boldsymbol{t}}||^2$$

en sortie, il nous faudra faire approcher $\hat{\boldsymbol{y}}_t = A(\hat{\Omega})\hat{\boldsymbol{x}}_t$ au plus près possible de $\boldsymbol{y}_t$. Grâce à la rétropropagation du gradient de l'erreur, de la sortie vers l'entrée, on va pouvoir modifier les poids du réseau (et donc modifier les $\hat{\omega}_k$) dans le sens de la *diminution* de l'erreur en sortie. De ce fait, plus on diminuera l'erreur en sortie, plus les $\hat{\omega}_k$ se rapprocheront des ω_k, et plus les $\hat{\boldsymbol{x}}_t$ se rapprocheront des $\boldsymbol{x}_t$. On pourrait dire d'une autre façon que le réseau *apprend* à "expliquer au mieux" le jeu d'observations $\boldsymbol{y}_t,\ t \in [1, N]$.

Au départ on dispose d'un réseau "générique"; après apprentissage, le réseau est associé au jeu d'observations $\boldsymbol{y}_t,\ t \in [1, N]$, en ce sens que ses poids contiennent désormais l'estimation $\hat{\Omega} = [\hat{\omega}_1, ..., \hat{\omega}_n]^T$ des angles d'arrivée, et que pour chaque $\boldsymbol{y}_t,\ t \in [1, N]$ que l'on envoie en entrée du réseau, on peut recueillir sur la couche cachée l'estimation $\hat{\boldsymbol{x}}_t$ du vecteur $\boldsymbol{x}_t$ initialement associé à cette observation $\boldsymbol{y}_t$.

Cet estimateur diffère par conséquent des estimateurs de Haute-Résolution MUSIC et ESPRIT, puisqu'il estime non seulement les angles d'arrivée, mais il estime également les sources $\boldsymbol{x}_t$ de manière directe.

En résumé, les particularités du PCSC sont les suivantes:

- le réseau n'utilise aucun apprentissage préalable: l'apprentissage est réalisé uniquement sur le jeu d'observations $\boldsymbol{y}_t$ dont on dispose.
- l'algorithme d'apprentissage est utilisé uniquement dans le but de minimiser un critère de sortie: l'information utile ne se trouve donc pas sur la couche de sortie. L'information qui nous intéresse (les angles d'arrivée) se trouve toute entière contenue dans les poids du réseau.
- les poids reliant la couche cachée à la couche de sortie sont *contraints* par des informations sur la géométrie de l'antenne. Leur modification lors de la rétropropagation du gradient est donc elle-aussi contrainte à certaines règles spécifiques.
- il est possible d'estimer directement les sources $\boldsymbol{x}_t$, car elles sont estimées en permanence sur la couche cachée (vecteur $\hat{\boldsymbol{x}}_t$).

5.2.5 Initialisation du réseau

Les algorithmes d'optimisation par descente de gradient possèdent un talon d'Achille qui peut se révéler fatal: si les paramètres à estimer ne sont pas initialisés dans le

voisinage de la solution, deux risques sont à craindre. Tout d'abord, en démarrant loin du minimum, on peut passer par des zones de faibles gradient. La pente est alors très faible et la descente par gradient ("steepest descent", ou "plus grande pente" en anglais) vers le minimum peut s'avérer très lente. Ensuite, si le problème possède des minima locaux, rien ne garantit que le gradient n'entraînera pas l'algorithme dans l'un de ces minima. Une fois un minimum local atteint, l'algorithme procédant par gradient a convergé et n'en sortira pas. La solution trouvée peut donc être soit très longue à atteindre, soit mauvaise, soit les deux.

C'est pour remédier à ce type de problème que l'on va préférer initialiser les paramètres du PCSC dans le voisinage de la solution globale. De cette façon, puisque le réseau part d'une région proche du minimum global, il convergera rapidement et sûrement vers ce minimum.

L'initialisation des paramètres est fournie par le perceptron à trois couches développé dans la partie 4. Comme on l'avait mentionné alors, ce perceptron est entraîné une fois pour toutes sur une base d'apprentissage. Une fois entraîné, ce perceptron peut fournir *très rapidement* (environ m^3 multiplications) une bonne solution $\tilde{\Omega} = [\tilde{\omega}_1, ..., \tilde{\omega}_n]^T$ au problème. Cette estimation, que l'on peut qualifier de "grossière" en comparaison du "raffinement" que va procurer le Perceptron à Structure Contrainte, va servir à *initialiser* les paramètres internes de ce PCSC: la matrice des poids de sortie est initialisée par

$$\hat{A}_{init} = A(\tilde{\Omega}) = [\boldsymbol{a}(\tilde{\omega}_1), \boldsymbol{a}(\tilde{\omega}_2), ..., \boldsymbol{a}(\tilde{\omega}_n)] \tag{113}$$

D'autre part, puisque la fonction de transition des neurones complexes de ce réseau à structure contrainte est l'identité, on peut écrire à tout instant

$$\forall\, t \in [1, N], \quad \hat{\boldsymbol{y}}_t = \hat{A}\hat{\boldsymbol{x}}_t$$

Cette égalité montre que $\hat{\boldsymbol{y}}_t$ appartient au sous-espace engendré par $\hat{A}$. De ce fait, le meilleur choix pour que le réseau minimise l'erreur entre $\hat{\boldsymbol{y}}_t$ et $\boldsymbol{y}_t$ en sortie, est de prendre $\hat{\boldsymbol{y}}_t$ égal à la projection orthogonale de $\boldsymbol{y}_t$ sur $\hat{A}$:

$$\hat{\boldsymbol{y}}_t = P_{\hat{A}}(\boldsymbol{y}_t) = \hat{A}\left(\hat{A}^H\hat{A}\right)^{-1}\hat{A}\boldsymbol{y}_t$$

Comme d'autre part la structure du réseau impose

$$\hat{\boldsymbol{y}}_t = \hat{A}B\boldsymbol{y}_t$$

on voit que le meilleur choix d'initialisation de la matrice des poids d'entrée B est:

$$B_{init} = \left[A^H(\tilde{\Omega})A(\tilde{\Omega})\right]^{-1}A^H(\tilde{\Omega}) \tag{114}$$

où $\tilde{\Omega}$ désigne l'estimation "grossière" fournie par le premier perceptron.

5.3 Equations de propagation du PCSC

Lorsque se présente un problème d'estimation d'angles d'arrivée de n sources sur une antenne de m capteurs, nous allons d'abord chercher à obtenir une estimation "grossière" grâce au premier perceptron. Pour cela, nous construisons la matrice de covariance normalisée des signaux, et nous envoyons ces valeurs sur la couche d'entrée du PMC.

Nous disposons alors d'une première estimation $\tilde{\Omega} = [\tilde{\omega}_1, ...\tilde{\omega}_n]^T$ en sortie. Nous pouvons alors initialiser les matrices de poids $\hat{A}_{init}$ et B_{init} du second perceptron suivant les équations (113) et (114). Pour ce perceptron à structure contrainte, les données reçues en entrée sont les m composantes complexes d'un vecteur signal $\boldsymbol{y}_t$ reçu sur l'antenne. A ce vecteur d'entrée $\boldsymbol{y}_t$ est associé un vecteur de sortie désiré $\boldsymbol{y}_t$ (le même vecteur, en fait). La base d'entraînement sur laquelle l'erreur en sortie va être minimisée, est le jeu $(\boldsymbol{y}_t)_{t\in[1,N]}$ des observations dont on dispose. Comme on le verra dans la partie expérimentations, il suffit de très peu d'itérations de l'algorithme de rétropropagation de l'erreur pour faire converger le réseau. En règle générale, 15 itérations suffisent.

Au Chapitre 1 (Section (2.4)), on a vu que la rétropropagation d'un réseau complexe est un peu plus délicate à exprimer que pour un réseau réel. Puisque la fonction de transition choisie est holomorphe, les équations de propagation de ce réseau peuvent être décrites comme suit. La variation du poids reliant le neurone a au neurone b est proportionnelle au gradient

$$\Delta W_{ab} = -\alpha \frac{\partial e_{QM}}{\partial W_{ab}}$$

où l'erreur quadratique moyenne e_{QM} est obtenue en théorie par l'espérance mathématique sur l'erreur quadratique instantanée e_Q (pour des raisons de clarté, l'indice t est omis dans l'expression e_Q). En pratique, on *estime* e_{QM} par

$$e_{QM} = \frac{1}{N} \sum_t e_Q$$

si N est le nombre d'observations. Comme la dérivation est une opération linéaire, on obtient par conséquent

$$\frac{\partial e_{QM}}{\partial W_{ab}} = \frac{1}{N} \sum_t \frac{\partial e_Q}{\partial W_{ab}}$$

et d'après le Chapitre 1 équation (23), on a (pour une fonction de transition égale à

l'identité):

$$\frac{\partial e_Q}{\partial W_{ab}} = \delta_b X_a^* \tag{115}$$

Pour une fonction de transition F holomorphe, la relation (24) du Chapitre 1 indique que δ_b s'écrit:

$$\delta_b = \left(\frac{\partial e_Q}{\partial O_b}\right) (F'(X_b))^*$$

où

$$\begin{aligned} \frac{\partial e_Q}{\partial O_b} &= O_b - S_b \quad \text{pour la couche de sortie} \\ &= \sum_{k \in succ(b)} \delta_k W_{bk}^* \quad \text{pour les autres couches} \end{aligned}$$

On a dit d'autre part que l'on choisissait la fonction identité pour F, ce qui nous donne finalement:

$$\begin{aligned} \delta_b &= O_b - S_b \quad \text{pour la couche de sortie} \\ &= \sum_{k \in succ(b)} \delta_k W_{bk}^* \quad \text{pour les autres couches} \end{aligned} \tag{116}$$

Cependant, nous ne pouvons pas appliquer ces règles aussi simplement. En effet, le réseau que nous utilisons est un réseau spécifique possédant une *structure contrainte*. Il est vrai que la matrice B des poids reliant la couche d'entrée à la couche cachée n'est pas contrainte, et nous pourrons utiliser ces règles pour le renouvellement des poids de la matrice B: d'après les équations (115) et (116), le gradient instantané en entrée d'un réseau à trois couches est

$$\frac{\partial e_{QM}}{\partial W_{ab}^B} = \sum_{k \in succ(b)} \delta_k W_{bk}^* X_a^* \tag{117}$$

Notons au passage que pour un poids W_{ab}^B reliant un neurone d'entrée à un neurone caché b, nous avons X_a égal à la a-ième composante du vecteur d'entrée $\boldsymbol{y}_t$, soit $X_a = y_{ta}$.

Dans la matrice d'entrée B , tous les poids sont libres. Ce sont donc bien les poids qui sont les paramètres (libres) à rafraîchir à chaque itération. Par contre, dans la matrice $\hat{A}$, les poids *ne sont pas* libres. Les seuls paramètres libres de la matrice

$$\hat{A} = A(\hat{\Omega}) = [\boldsymbol{a}(\hat{\omega}_1), \boldsymbol{a}(\hat{\omega}_2), ..., \boldsymbol{a}(\hat{\omega}_n)]$$

sont les $\hat{\omega}_k$. Des règles de rétropropagation du gradient spécialement adaptées à la structure contrainte de la matrice $A(\hat{\Omega})$ doivent donc être dérivées. Pour commencer, le gradient ne s'écrit plus

$$\frac{\partial e_Q}{\partial W_{ab}}$$

pour les poids de la matrice $\hat{A}$. En effet, ces poids sont contraints à suivre la forme d'une matrice directionnelle $A(\hat{\Omega})$ où les seuls paramètres pouvant varier sont les $\hat{\omega}_a$. De ce fait, on calculera désormais $\Delta\hat{\omega}_a$ pour tous les paramètres du vecteur $\hat{\Omega}$, puis nous réajusterons le nouveau $\hat{\Omega}$ en conséquence:

$$\hat{\Omega}' = \hat{\Omega} + \Delta\hat{\Omega}$$

Il suffira alors de calculer la nouvelle matrice de poids $\hat{A}' = A(\hat{\Omega}')$ pour véritablement faire varier les poids de la matrice $\hat{A}$. Pour les paramètres libres $\hat{\omega}_a$, le gradient s'écrit:

$$\frac{\partial e_Q}{\partial \hat{\omega}_a}$$

et selon les règles du calcul différentiel nous avons

$$\frac{\partial e_Q}{\partial \hat{\omega}_a} = \sum_b \left(\frac{\partial e_Q}{\partial W_{ab}^R} \frac{\partial W_{ab}^R}{\partial \hat{\omega}_a} + \frac{\partial e_Q}{\partial W_{ab}^I} \frac{\partial W_{ab}^I}{\partial \hat{\omega}_a} \right) \tag{118}$$

D'autre part, la convention (13) du Chapitre 1 indique que:

$$\frac{\partial e_Q}{\partial W_{ab}} = \frac{\partial e_Q}{\partial W_{ab}^R} + j \frac{\partial e_Q}{\partial W_{ab}^I}$$

Par suite nous avons

$$\begin{aligned}
\sum_b \left(\frac{\partial e_Q}{\partial W_{ab}} \right) \left(\frac{\partial W_{ab}}{\partial \hat{\omega}_a} \right)^* &= \sum_b \left(\frac{\partial e_Q}{\partial W_{ab}^R} + j \frac{\partial e_Q}{\partial W_{ab}^I} \right) \left(\frac{\partial W_{ab}^R}{\partial \hat{\omega}_a} - j \frac{\partial W_{ab}^I}{\partial \hat{\omega}_a} \right) \\
&= \sum_b \left[\frac{\partial e_Q}{\partial W_{ab}^R} \frac{\partial W_{ab}^R}{\partial \hat{\omega}_a} + \frac{\partial e_Q}{\partial W_{ab}^I} \frac{\partial W_{ab}^I}{\partial \hat{\omega}_a} \right] \\
&\quad + j \sum_b \left[-\frac{\partial e_Q}{\partial W_{ab}^R} \frac{\partial W_{ab}^I}{\partial \hat{\omega}_a} + \frac{\partial e_Q}{\partial W_{ab}^I} \frac{\partial W_{ab}^R}{\partial \hat{\omega}_a} \right]
\end{aligned}$$

où l'on reconnaît, dans la partie réelle, le terme de droite de l'équation (118). Nous pouvons donc écrire en définitive:

$$\frac{\partial e_Q}{\partial \hat{\omega}_a} = \mathrm{Re}\left(\sum_b \left(\frac{\partial e_Q}{\partial W_{ab}} \right) \left(\frac{\partial W_{ab}}{\partial \hat{\omega}_a} \right)^* \right) \tag{119}$$

où $\frac{\partial e_Q}{\partial W_{ab}}$ est classiquement donné par l'équation (23) du Chapitre 1:

$$\frac{\partial e_Q}{\partial W_{ab}} = \delta_b X_a^*$$

et où X_a^* désigne en fait la valeur du neurone a sur la couche cachée: c'est la a-ième composante du vecteur $\hat{\boldsymbol{x}}_t$, soit $\hat{x}_{ta}$, d'après nos notations précédentes (figure (17)). Par ailleurs, puisque les poids auxquels nous nous intéressons sont reliés à la couche de sortie, on peut écrire

$$\delta_b = \frac{\partial e_Q}{\partial X_b} = \hat{y}_{tb} - y_{tb} \tag{120}$$

où y_{tb} (respectivement $\hat{y}_{tb}$) représente la b-ième composante (complexe) du vecteur $\boldsymbol{y}_t$ (respectivement $\hat{\boldsymbol{y}}_t$). Ainsi, pour le b-ième neurone de sortie, on a

$$\frac{\partial e_Q}{\partial W_{ab}} = (\hat{y}_{tb} - y_{tb})\, X_a^* \tag{121}$$

ce qui fournit la partie gauche de la partie réelle (équation (119)) que nous cherchons à expliciter. Le terme de droite de cette partie réelle, à savoir $\frac{\partial W_{ab}}{\partial \hat{\omega}_a}$, dépend de la forme de W_{ab} en fonction de $\hat{\omega}_a$. Pour une antenne de m capteurs de forme quelconque, le vecteur directionnel $\boldsymbol{a}(\omega)$ s'exprime par:

$$\boldsymbol{a}(\omega) = [e^{f_1(\omega)}, e^{f_2(\omega)}, ..., e^{f_m(\omega)}]^T,$$

ce qui implique $W_{ab} = e^{f_b(\hat{\omega}_a)}$, où f_k représente une fonction complexe de la variable réelle ω. Dans ce cas, on peut écrire

$$\frac{\partial W_{ab}}{\partial \hat{\omega}_a} = f_b'(\hat{\omega}_a) W_{ab}$$

ce qui montre que cette méthode peut s'adapter à tout type d'antenne, quelle que soit sa forme, du moment que ses caractéristiques géométriques sont connues, c'est-à-dire du moment qu'elles peuvent s'exprimer sous la forme $f_k(\omega_l)$.

Nous allons ici nous contenter d'utiliser le type d'antenne le plus courant (cela permettra par la suite d'effectuer facilement des comparaisons avec d'autres méthodes), c'est-à-dire l'antenne linéaire uniforme. Pour ce type d'antenne, on a

$$W_{ab} = e^{-j(b-1)\hat{\omega}_a}$$

d'où

$$\frac{\partial W_{ab}}{\partial \hat{\omega}_a} = -j(b-1) W_{ab} \tag{122}$$

Finalement, les équations (119), (121) et (122) se combinent pour donner place à

l'expression finale du gradient en sortie:

$$\frac{\partial e_Q}{\partial \hat{\omega}_a} = \mathrm{Re}\left(\sum_b \left(\hat{y}_{tb} - y_{tb}\right) X_a^* j(b-1) W_{ab}^*\right)$$

Il est aussi possible de simplifier l'équation (117) décrivant le renouvellement des poids de la matrice B. Rappelons que l'étude du renouvellement de la connexion entre le neurone a de la couche d'entrée et le neurone b de la couche cachée, fait appel à tous les neurones successeurs du neurone b. Pour un perceptron à trois couches, nous savons que tout neurone k succédant à un neurone b de la couche cachée, se trouve sur la couche de sortie, donc

$$\delta_k = \frac{\partial e_Q}{\partial X_k} = \hat{y}_{tk} - y_{tk}$$

d'après (120). De la même façon, W_{bk}, en tant que poids de la matrice $\hat{A}$, s'exprime par

$$W_{bk} = e^{-j(k-1)\hat{\omega}_b}$$

pour une antenne linéaire uniforme. Enfin, on rappelle que la couche d'entrée reçoit le vecteur $\boldsymbol{y}_t$, donc sa a-ième composante X_a vaut y_{ta}. Finalement, l'équation (117) se réécrit plus simplement:

$$\Delta W_{ab}^B = -\frac{\alpha}{N} \sum_{t=1}^{N} \left[\sum_{k \in succ(b)} \left(\hat{y}_{tk} - y_{tk}\right) e^{j(k-1)\hat{\omega}_b} y_{ta}^* \right]$$

Nous possédons désormais tous les outils pour mettre en œuvre l'algorithme du gradient dans le cas du réseau complexe à structure contrainte.

5.4 Résumé de l'algorithme de rétropropagation utilisé

Après construction du réseau complexe à trois couches (m neurones d'entrée, n neurones cachés, m neurones de sortie), nous avons initialisé la matrice contrainte $\hat{A}$ des poids de sortie par $\hat{A}_{init} = A(\tilde{\Omega})$, et la matrice B des poids d'entrée par $B_{init} = \left[A^H(\tilde{\Omega})A(\tilde{\Omega})\right]^{-1} A^H(\tilde{\Omega})$, où $\tilde{\Omega}$ désigne l'estimation "grossière" obtenue en sortie du premier perceptron. $\hat{A}$ étant par nature *contrainte* à avoir la forme d'une matrice directionnelle $A(\hat{\Omega})$, les $\hat{\omega}_k$ formant le vecteur $\hat{\Omega}$ sont les seuls paramètres libres. La rétropropagation de l'erreur quadratique moyenne en sortie est donc elle aussi contrainte à suivre certaines règles spécifiques à ce réseau. Nous allons décrire ces règles dans le cas d'une antenne linéaire uniforme.

- pour chaque itération
  - pour chaque vecteur observation $\boldsymbol{y}_t$
    - passage du vecteur $\boldsymbol{y}_t$ dans le réseau. Récupération du vecteur $\hat{\boldsymbol{x}}_t$ sur la couche cachée et $\hat{\boldsymbol{y}}_t$ en sortie.
    - calcul du gradient instantané en sortie

$$\frac{\partial e_Q}{\partial \hat{\omega}_a}(t) = \mathrm{Re}\left(\sum_b \left(\hat{y}_{tb} - y_{tb}\right) \hat{x}_{ta}^* j(b-1) e^{j(b-1)\hat{\omega}_a}\right) \tag{123}$$

    - calcul du gradient instantané en entrée

$$\frac{\partial e_Q}{\partial W_{ab}^B}(t) = \sum_{k \in sortie} \left(\hat{y}_{tk} - y_{tk}\right) e^{j(k-1)\hat{\omega}_b} y_{ta}^* \tag{124}$$

  - calcul du gradient moyen en sortie

$$\frac{\partial e_{QM}}{\partial \hat{\omega}_a} = \frac{1}{N} \sum_{t=1}^{N} \frac{\partial e_Q}{\partial \hat{\omega}_a}(t) \tag{125}$$

    et en entrée

$$\frac{\partial e_{QM}}{\partial W_{ab}^B} = \frac{1}{N} \sum_{t=1}^{N} \frac{\partial e_Q}{\partial W_{ab}^B}(t) \tag{126}$$

  - renouvellement des paramètres $\hat{\omega}_a$ formant le vecteur $\hat{\Omega} = [\hat{\omega}_1, ..., \hat{\omega}_n]^T$

$$\Delta \hat{\omega}_a = -\alpha \frac{\partial e_{QM}}{\partial \hat{\omega}_a}$$

  - renouvellement des poids de la matrice $\hat{A}$ grâce au nouvel $\hat{\Omega}$:

$$\hat{A}_{nouv} = A(\hat{\Omega}_{nouv})$$

  - renouvellement des poids de la matrice B:

$$\Delta W_{ab}^B = -\alpha \frac{\partial e_{QM}}{\partial W_{ab}^B}$$

- fin des itérations lorsque le réseau a convergé (en pratique une quinzaine d'itérations suffit). Extraction du vecteur paramètre $\hat{\Omega} = [\hat{\omega}_1, ..., \hat{\omega}_n]^T$ à partir de la matrice des poids de sortie $\hat{A} = A(\hat{\Omega})$. Si on le désire, extraction des estimations des sources $\hat{\boldsymbol{x}}_t$ à partir des valeurs des neurones de la couche cachée.

5.5 Discussion

La méthode d'estimation qui vient d'être présentée possède des atouts évidents: tout d'abord, elle minimise le même critère théorique que la méthode du Maximum de Vraisemblance. Or, même si elle reste peu employée du fait de sa grande gourmandise en temps de calcul, la méthode du Maximum de Vraisemblance reste la meilleure jusqu'à maintenant (voir les études théoriques poussées de Stoica et Nehorai [24], [25]). Le réseau complexe à structure contrainte est cependant beaucoup plus rapide que le Maximum de Vraisemblance, parce que sa minimisation procède par descente de gradient.

Ces remarques en font donc une méthode performante en théorie. Cependant, comme tous les algorithmes utilisant une descente de gradient, le risque pour le PCSC est soit de se retrouver piégé dans un minimum local (conduisant alors à une solution différente de celle du MV), soit de se retrouver, après initialisation, très loin du minimum global, voire dans une partie de l'espace où la pente de la surface e_{QM} est très faible (conduisant alors à une solution éloignée du minimum, parce que le nombre d'itérations attribué n'aura pas été suffisant pour l'amener au terme de sa descente).

C'est pourquoi les paramètres internes du PCSC sont initialisés dans un voisinage proche de la solution globale, grâce à une première estimation fournie par le perceptron réel à trois couches étudié dans la partie 4. Cette initialisation à proximité de la solution permet d'éviter que le PCSC ne tombe dans un minimum local. L'autre intérêt de l'initialisation dans un voisinage du minimum global, est de diminuer de façon importante le nombre d'itérations nécessaires à la convergence finale du réseau. En effet, on a pu constater qu'en général une quinzaine d'itérations étaient suffisantes pour stabiliser le réseau.

Notons enfin que la méthode utilisant le PCSC fournit *directement*, outre l'estimation $\hat{\Omega}$ des angles d'arrivée contenue dans les poids de la matrice de sortie $\hat{A}$, une estimation $\hat{\boldsymbol{x}}_t$ des vecteurs source originels qu'il suffit d'extraire de la couche cachée. Contrairement aux méthodes de Haute Résolution telles que MUSIC et ESPRIT, l'algorithme basé sur l'utilisation du PCSC estime donc les $\hat{\boldsymbol{x}}_t$ directement sans calcul supplémentaire, ce qui peut s'avérer utile pour certaines applications, comme nous allons le constater dans les sections suivantes.

6 Séparation Aveugle de Signaux

6.1 Introduction au problème

La séparation de signaux est une partie importante du domaine couvert par l'appellation générique de "traitement du signal". En effet, les "signaux" fournis par des capteurs sont rarement les sources pures que l'on cherche à analyser, mais plus souvent des mélanges de plusieurs sources indépendantes. La séparation de sources consiste à extraire les signaux originaux de ces mélanges. Si ce problème se conçoit aisément en traitement d'antenne pour les signaux radar ou sonar, il ne faut pas oublier les domaines plus "terre-à-terre" dans lesquels se pose clairement le problème de la séparation de sources: séparation des signaux véhiculés par les fibres nerveuses à partir des muscles vers le cerveau, séparation de signaux de paroles entre plusieurs locuteurs s'exprimant simultanément (cocktail party), ou tout simplement séparation entre les signaux de parole et le "bruit" ambiant (bruit de moteur dans un cockpit, amélioration du "débruitage" pour les téléphones cellulaires utilisés dans une voiture ou au milieu d'une foule, etc).

Suivant la nature des informations dont on dispose, la séparation de sources se fera de manière plus ou moins lourde. Il est clair que moins on dispose d'informations, plus il nous faudra déployer d'efforts pour séparer les sources. Nous nous plaçons dans le cas de séparation de sources dite "séparation aveugle", c'est-à-dire un cas où:

- le nombre de capteurs dont on dispose est égal au nombre de sources (sous-entendu on connaît le nombre de sources).
- les sources sont indépendantes entre elles.
- nous connaissons une forme paramétrique du mélange (par exemple, nous savons que le mélange est linéaire).

Bien que l'hypothèse (forte) d'indépendance des sources soit utilisée, une séparation de sources telle que celle décrite ci-dessus est définie comme étant "aveugle" parce qu'aucune hypothèse sur les *sources* elles-mêmes n'est faite. Le même algorithme doit donc pouvoir fonctionner pour des sources binaires, sinusoïdales, aléatoires, etc.

Nous noterons $\boldsymbol{x}_t$ le vecteur représentant les sources originales (indépendantes), $\boldsymbol{y}_t$ le vecteur des observations (mélange en sortie des capteurs), $\boldsymbol{s}_t$ le vecteur obtenu en sortie du dispositif de séparation, et n le nombre de sources. Le temps t est supposé discret (pas forcément issu d'un échantillonnage uniforme, sauf lorsqu'une mention spéciale en fera part).

Signalons que le vecteur $\boldsymbol{s}_t$ obtenu est supposé (si l'algorithme fonctionne bien) représenter n signaux indépendants. Cependant, ce vecteur n'est *pas* une estimation directe de $\boldsymbol{x}_t$. En effet, il sera montré dans la section suivante que $\boldsymbol{s}_t$ représente

les sources originales modulo une permutation (d'ordre), et une distorsion éventuelles.

6.2 Une première approche

Les premiers auteurs à s'être intéressés, et à *résoudre* le problème de Séparation Aveugle de Sources[6] (SAS), sont Jutten et Hérault [14]. Ces auteurs consacrent tout d'abord plus particulièrement leur étude à la séparation de mélanges linéaires instantanés. Dans ce cas, la relation entre les signaux capteurs et les sources originales indépendantes s'écrit:

$$\boldsymbol{y}_t = A\boldsymbol{x}_t$$

où A est une matrice carrée $n \times n$ inconnue. L'opération de séparation réside dans l'identification d'une matrice $(n \times n)$ F telle que:

$$\boldsymbol{s}_t = F\boldsymbol{y}_t$$

où les composantes de $\boldsymbol{s}_t$ sont statistiquement indépendantes. Il est alors facile de montrer pourquoi $\boldsymbol{s}_t$ est égal à $\boldsymbol{x}_t$ modulo une permutation et une dilatation. En effet, rappelons qu'un algorithme de séparation aveugle ne dispose d'aucune hypothèse a priori sur les sources, si ce n'est l'hypothèse de leur indépendance statistique. Ainsi, puisque les composantes de $\boldsymbol{x}_t$ sont indépendantes, un vecteur $\boldsymbol{s}_t = PD\boldsymbol{x}_t$ formé à partir de n'importe quelle dilatation D (matrice diagonale) et n'importe quelle permutation P, a aussi des composantes indépendantes. Il est donc impossible de lever cette indétermination dès lors que l'on ne possède aucune autre information sur les sources originales.

L'approche proposée par Jutten et Hérault [14], [15], [11] propose d'identifier F, la matrice inverse du mélange, sous la forme $F = (C + I)^{-1}$, où I est la matrice identité de dimension n, et C une matrice dont la diagonale est nulle. Par cette astuce, la matrice solution trouvée est contrainte à être de rang n, ce qui permet d'éviter les cas triviaux. Le schéma de fonctionnement de cet estimateur est présenté en figure (19). Pour un problème à deux sources et deux capteurs, par exemple, la matrice C à estimer s'écrit:

$$C = \begin{pmatrix} 0 & c_{12} \\ c_{21} & 0 \end{pmatrix}$$

où les modifications incrémentales des coefficients c_{12} et c_{21} dépendent indirectement de l'estimation $\boldsymbol{s}_t = [s_{t1}, s_{t2}]^T$ en sortie, par l'intermédiaire de fonctions f et g:

[6] D'autres méthodes ou variantes ont été proposées depuis. Notre objectif n'est pas de les passer toutes en revue ici: nous nous limiterons à l'exposé de la méthode de Jutten et Hérault, qui est l'une de celles qui présentent le meilleur compromis performances/complexité

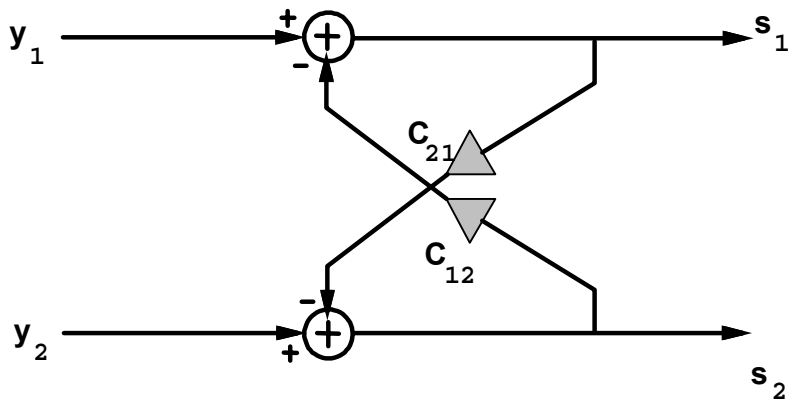

Figure 19: *Schéma de principe de l'estimateur Jutten-Hérault de séparation aveugle de sources (ici, cas pour deux sources)*

$$\begin{aligned} \Delta c_{12} &= \mu f(s_{t1}) g(\tilde{s_{t2}}) \\ \Delta c_{21} &= \mu f(s_{t2}) g(\tilde{s_{t1}}) \end{aligned}$$

où μ est un gain d'adaptation, $\tilde{s_{ti}}$ désigne s_{ti} centré pour $i \in \{1, 2\}$. Dans [14], Jutten et Hérault montrent que les fonctions f et g doivent respecter certaines contraintes pour que l'algorithme converge. En particulier, f et g sont non-linéaires, impaires en croissantes. Les auteurs suggèrent d'utiliser par exemple

$$f(a) = a^3$$
$$\text{et} \qquad g(a) = \text{Arctan}(10a)$$

Jutten et Hérault ont montré que lorsque ces conditions sont remplies, la matrice C qui correspond à la bonne solution est un état stable de l'algorithme. D'autres articles [10], [23] ont cependant montré qu'il peut exister d'autres états stables de l'algorithme qui ne sont pas solution du problème, et que selon l'état initial des paramètres c_{12} et c_{21}, et selon le comportement des sources, l'estimateur proposé pouvait ne pas converger vers la bonne solution.

Il apparaît que l'algorithme proposé par Jutten et Hérault ne se base pas sur la minimisation d'une fonction de coût, mais se rapproche davantage de la recherche simultanée des zéros de plusieurs fonctions, ce qui rend son étude théorique assez difficile [10]. Cet algorithme présente l'avantage d'être simple, rapide et bien adapté à une implantation parallèle. De plus, la convergence vers une mauvaise matrice C se rencontre rarement en pratique.

Notre but final étant de pouvoir fusionner l'approche "estimation d'angles d'arrivée" avec l'approche "séparation de sources", il est naturel d'envisager un algorithme de séparation basé sur un critère d'erreur (une fonction de coût) à minimiser, qui pourra se marier plus facilement au critère que minimise le Perceptron à Structure Contrainte évoqué dans la section précédente.

6.3 Approche par minimisation d'une fonction de coût

6.3.1 Position du problème

La méthode que nous allons étudier ici [2], [3] est une méthode qui peut réaliser la séparation de mélanges linéaires aussi bien que non-linéaires. Par souci de généralité nous ne pouvons donc plus utiliser les notations matricielles de la section précédente. Nous noterons désormais

$$\boldsymbol{y}_t = v(\boldsymbol{x}_t)$$

la transformation instantanée opérant le mélange, avec v une forme paramétrique pas forcément linéaire, mais dont nous *connaissons* la structure. Si on appelle w l'inverse de cette forme paramétrique, il s'agit pour nous d'estimer les paramètres de w de telle sorte que les composantes de

$$\boldsymbol{s}_t = w(\boldsymbol{y}_t)$$

soient indépendantes.
Un exemple de forme paramétrique non-linéaire est donné ci-dessous:

$$v = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{bmatrix} sgn(.)[.]^2 \\ sgn(.)[.]^2 \end{bmatrix} \begin{pmatrix} e & f \\ g & h \end{pmatrix}$$

où les valeurs $a, b, ..., h$ sont inconnues. Cette transformation consiste donc à multiplier le vecteur source par une matrice, à élever les composantes du résultat au carré (en conservant le signe), et à effectuer à nouveau une multiplication par une matrice. On sait qu'une forme paramétrique inverse de ce mélange est:

$$w = \begin{pmatrix} a' & b' \\ c' & d' \end{pmatrix} \begin{bmatrix} sgn(.)\sqrt{|\ .\ |} \\ sgn(.)\sqrt{|\ .\ |} \end{bmatrix} \begin{pmatrix} e' & f' \\ g' & h' \end{pmatrix}$$

c'est-à-dire que la méthode de séparation de sources que nous allons présenter doit ici être capable de trouver un octuplet $(a', b', ..., h')$ tel que la forme associée fournisse des vecteurs $\boldsymbol{s}_t = w(\boldsymbol{y}_t)$ dont les composantes soient statistiquement indépendantes.

La méthode de séparation de sources de Burel [2], [3] se base sur une procédure en deux étapes:

- définition d'une mesure de dépendance statistique de l'estimation $\boldsymbol{s}_t$
- mise au point d'un algorithme de calcul des paramètres internes de w qui minimisent cette mesure.

Nous verrons que cette méthode possède l'avantage de pouvoir transposer le problème de séparation de sources à un problème de minimisation de coût en sortie d'un réseau de neurones multicouche pour toute forme paramétrique (éventuellement non-linéaire) v. En effet, puisque la *forme* paramétrique de v est connue (hypothèse de départ), on peut en déduire la forme paramétrique de son inverse w, et répertorier ainsi les paramètres inconnus de w.
Observons que cette hypothèse n'est pas plus forte que l'hypothèse utilisée dans l'approche précédente, puisque l'on y supposait que v était linéaire. Par ailleurs cette hypothèse a trait à la *forme* du mélange et non aux sources elles-mêmes, ce qui n'ôte aucunement le caractère "aveugle" de la séparation.

L'approche du problème de la séparation de sources par l'utilisation d'un réseau de neurones semble assez naturelle au départ[7]. En effet, on imagine bien que les neurones d'entrée recevront les vecteurs $\boldsymbol{y}_t$ du mélange, et que les neurones de sortie fourniront un vecteur $\boldsymbol{s}_t$, avec lequel sera calculée la mesure de dépendance évoquée plus haut. Aux paramètres internes inconnus de w seront associés les poids (ajustés lors de la minimisation par gradient du critère de dépendance en sortie). Enfin, les fonctions (linéaires ou non) de la forme w (ici la fonction $sgn(.)\sqrt{|\ .\ |}$ en ce qui concerne l'exemple donné précédemment) se retrouveront dans les fonctions de transition des neurones. Un schéma général de cette approche est proposé en figure (20).

Avant de rentrer dans les détails de la structure de ce réseau de neurones, étudions d'abord la mesure de dépendance de signaux introduite par Burel [2].

6.3.2 Mesure de dépendance

Nous utiliserons les notations suivantes:

$$\boldsymbol{s}_t = [s_1(t), s_2(t), ..., s_n(t)]^T$$

est le résultat de la méthode de séparation du mélange $\boldsymbol{y}_t$. Soit S_i la variable

[7] c'est d'ailleurs également sous cette forme que Jutten et Hérault introduisent leur approche

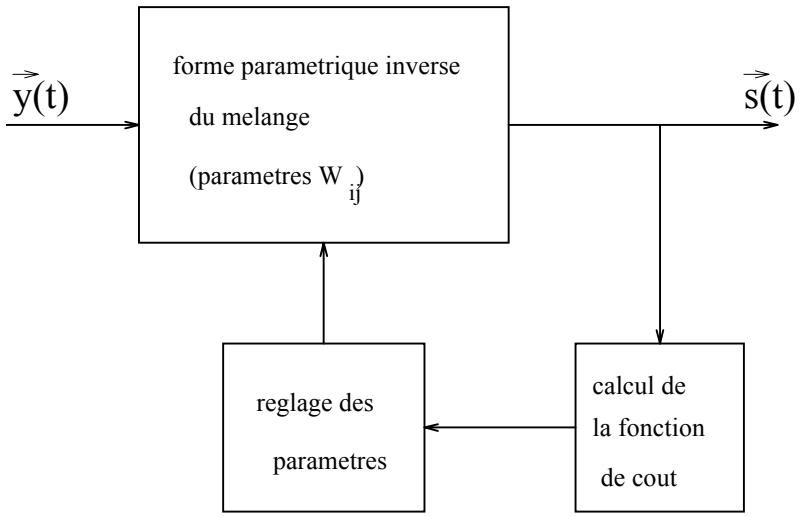

Figure 20: *Principe de la méthode de séparation de sources de Burel*

aléatoire correspondant au signal $s_i(t)$. On note:

$$\vec{\Theta}_i = [\overbrace{0,...,0}^{i-1},1,0,...,0]^T_n$$

$$R_{\alpha_1...\alpha_n} = E\left\{S_1^{\alpha_1}...S_n^{\alpha_n}\right\}$$

$$\begin{aligned} M_{\alpha_1...\alpha_n} &= E\left\{S_1^{\alpha_1}...S_n^{\alpha_n}\right\} - E\left\{S_1^{\alpha_1}\right\}...E\left\{S_n^{\alpha_n}\right\} \\ &= R_{\alpha_1...\alpha_n} - R_{\alpha_1\vec{\Theta}_1}...R_{\alpha_n\vec{\Theta}_n} \end{aligned}$$

Notre but est d'obtenir une estimation (à une permutation et une dilatation près) de $\boldsymbol{x}_t$, c'est-à-dire de rendre les composantes de $\boldsymbol{s}_t$ indépendantes. Une condition nécessaire et suffisante pour l'indépendance est:

$$\forall s_1,...,s_n \quad p_{S_1...S_n}(s_1,...,s_n) = p_{S_1}(s_1)...p_{S_n}(s_n)$$

La *mesure de dépendance* de Burel se base sur cette condition d'indépendance, et est définie par:

$$m_d = \int_{-\infty}^{+\infty}...\int_{-\infty}^{+\infty}\left([p_{S_1...S_n}(s_1,...,s_n) - p_{S_1}(s_1)...p_{S_n}(s_n)] * \prod_{i=1}^{n}\frac{1}{\sqrt{2\pi}\sigma_i}e^{-\frac{S_i^2}{2\sigma_i^2}}\right)^2 ds_1...ds_n \tag{127}$$

où le symbole "$*$" désigne le produit de convolution. Un filtrage gaussien sur le terme à intégrer a été introduit dans le but de rendre la mesure mieux adaptée au traitement de signaux réels. En effet, ce filtrage permet d'éviter des variations brutales de la mesure de dépendance, et d'atténuer les effets causés par des erreurs d'arrondi ou de mesure tels que ceux fréquemment rencontrés en pratique. Le filtrage gaussien évite en outre la divergence de l'intégrale lorsque les sorties sont discrètes.

La mesure de dépendance de n signaux définie à l'équation (127) n'est pas facilement utilisable sous cette forme. Burel [2] a montré qu'il est possible de réécrire cette mesure sous une forme plus adaptée au traitement du signal, et mettant en œuvre des moments:

$$\boxed{m_d = \sum_{\alpha_1=0}^{\infty} \dots \sum_{\alpha_n=0}^{\infty} \sum_{\beta_1=0}^{\infty} \dots \sum_{\beta_n=0}^{\infty} G_{\alpha_1 \dots \alpha_n, \beta_1 \dots \beta_n} M_{\alpha_1 \dots \alpha_n} M_{\beta_1 \dots \beta_n}} \tag{128}$$

où

$$G_{\alpha_1 \dots \alpha_n, \beta_1 \dots \beta_n} = \prod_{i=1}^{n} g(\alpha_i, \beta_i, \sigma_i)$$

et:

$$\begin{aligned} g(\alpha, \beta, \sigma) &= \frac{1}{\alpha! \beta!} (-1)^{\frac{\alpha-\beta}{2}} J_{\alpha+\beta}(\sigma) && \text{si } \alpha+\beta \text{ est pair} \\ &= 0 && \text{sinon} \end{aligned}$$

avec $J_{2k}(\sigma)$ la valeur suivante (voir le calcul détaillé dans l'Annexe B):

$$J_{2k}(\sigma) = \int_{-\infty}^{+\infty} x^{2k} e^{-\sigma^2 x^2} dx = \frac{(2k)!}{4^k k!} \frac{\sqrt{\pi}}{\sigma^{2k+1}}$$

L'expression définie à l'équation (127) étant une intégrale de carré, elle est toujours positive. De ce fait, l'expression (128), qui représente la même mesure, est une forme quadratique définie positive en fonction des moments $M_{\alpha_1 \dots \alpha_n}$. D'après la théorie des formes quadratiques (à ce sujet voir [1] par exemple), cela signifie que cette forme est nulle si, et seulement si, tous les moments $M_{\alpha_1 \dots \alpha_n}$ sont nuls. Il y a donc bien équivalence entre l'annulation de la "mesure de dépendance" m_d et l'annulation des moments $M_{\alpha_1 \dots \alpha_n}$ à tous les ordres, c'est-à-dire l'indépendance des composantes du vecteur $\boldsymbol{s}_t$.

On peut montrer (*cf* Annexe C) que lorsque l'un au moins des indices α_i tend vers l'infini, le coefficient $G_{\alpha_1 \dots \alpha_n, \beta_1 \dots \beta_n}$ tend vers zéro. Cette observation permet en pratique de limiter la sommation à un ordre fini. Si on limite la sommation à l'ordre K, on obtient:

$$\boxed{m_d^K = \sum_{\alpha_1+\ldots+\alpha_n \leq K} \sum_{\beta_1+\ldots+\beta_n \leq K} G_{\alpha_1\ldots\alpha_n,\beta_1\ldots\beta_n} M_{\alpha_1\ldots\alpha_n} M_{\beta_1\ldots\beta_n}} \quad (129)$$

qui est également une forme quadratique définie positive (*cf* [2]). De ce fait, annuler m_d^K équivaut à annuler tous les moments jusqu'à l'ordre K: l'indépendance des sources est alors assurée jusqu'à l'ordre K.

6.3.3 Minimisation d'une fonction de coût

La méthode de séparation de sources utilisant la mesure de dépendance est transposée sur un réseau de neurones à deux couches ou plus, tel que présenté en figure (21). Les seules couches obligatoirement présentes sont les couches d'entrée et de sortie, qui comportent chacune n neurones. Pour séparer un mélange linéaire, aucune couche

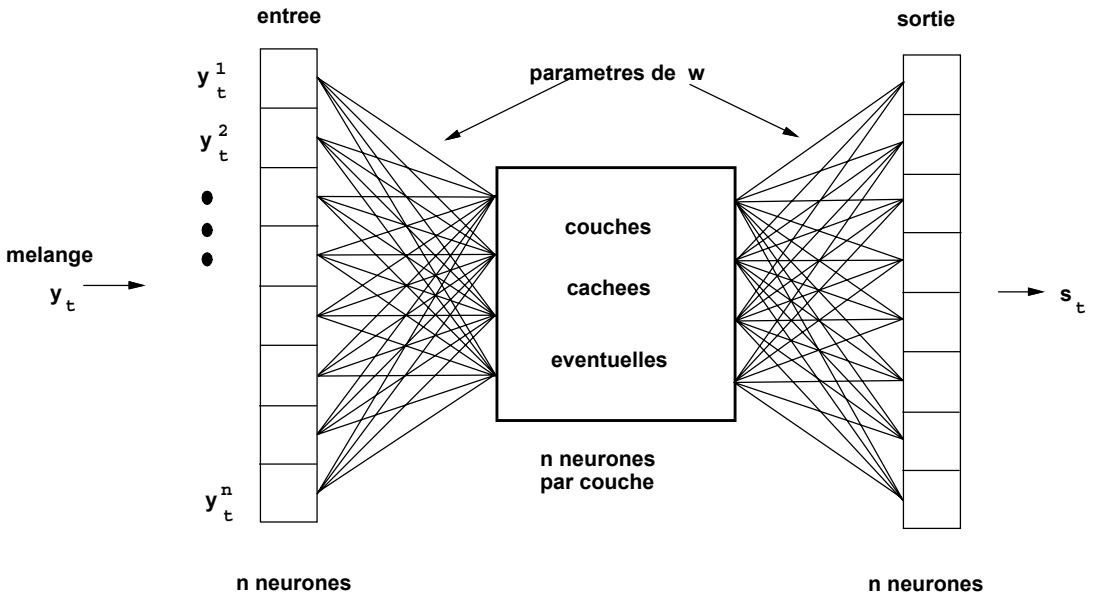

Figure 21: *Réseau de neurones à deux couches ou plus, minimisant le critère de dépendance en sortie*

cachée n'est requise. Pour un mélange non-linéaire, une ou plusieurs couches cachées seront nécessaires à la séparation, selon l'expression plus ou moins complexe de la forme paramétrique inverse w.

Contrairement à l'utilisation "classique" d'un perceptron multicouche, l'apprentissage n'est pas supervisé: les sorties $\boldsymbol{s}_t$ ne sont pas comparées à des sorties désirées $\boldsymbol{s}_t^d$. En

effet, les $\boldsymbol{s}_t$ sont utilisées pour calculer un coût de dépendance (c'est-à-dire un critère à minimiser) en sortie. La minimisation de ce critère est réalisée par rétropropagation de gradient. En principe, après un certain nombre d'itérations, on peut annuler - ou tout au moins rendre très petit - le coût en sortie. Les vecteurs de sortie $\boldsymbol{s}_t$ devraient alors être indépendants, et les poids du réseau représenter les paramètres internes de la forme w.

Burel propose d'utiliser la fonction de coût suivante:

$$\begin{aligned} C_{md} &= \frac{1}{2}\left\{\sum_{i=1}^{n} \mid E\{S_i\} \mid^2 + \sum_{i=1}^{n} \mid E\{{S_i}^2\} - 1 \mid^2 \right. \\ &\quad \left. + \sum_{\alpha_1+\ldots+\alpha_n \leq K} \; \sum_{\beta_1+\ldots+\beta_n \leq K} G_{\alpha_1\ldots\alpha_n,\beta_1\ldots\beta_n} M_{\alpha_1\ldots\alpha_n} M_{\beta_1\ldots\beta_n} \right\} \end{aligned} \tag{130}$$

Cette fonction de coût est la demi-somme de trois termes positifs:

- la mesure de dépendance m_d^K (équation (129)), qui, lorsqu'elle est nulle, assure l'indépendance des sources jusqu'à l'ordre K.
- le terme $\sum_{i=1}^{n} \mid E\{{S_i}^2\} - 1 \mid^2$, qui correspond à la condition

  $$\forall i \in \{1, ..., n\}, \quad E\{{S_i}^2\} = 1$$

  c'est-à-dire que la puissance de chaque source est "forcée" à 1; cette astuce permet d'éviter la solution triviale (sources identiquement nulles).
- le terme $\sum_{i=1}^{n} \mid E\{S_i\} \mid^2$, qui correspond à la condition

  $$\forall i \in \{1, ..., n\}, \quad E\{S_i\} = 0$$

  qui est une sorte de normalisation des sorties obtenues (la condition précédente peut également se voir comme une normalisation).

D'autres conditions de normalisation pourraient être utilisées pour certaines applications dont on connaîtrait, par exemple, des statistiques sur les sources. Dans le cas présent ces conditions ne jouent pas sur la forme des sorties obtenues, mais seulement sur leur amplitude ou leur moyenne, ce qui n'est aucunement dommageable puisque de toute façon, la solution $\boldsymbol{s}_t$ est égale à $\boldsymbol{x}_t$ modulo une permutation et une distorsion.

Le détail de la dérivation du coût en sortie pour le calcul de gradient est présenté dans [2]. Suivant la configuration dans laquelle on se trouve, deux méthodes de dérivation différentes peuvent être employées: lorsque l'on dispose de tous les mélanges $\boldsymbol{s}_t,\ t \in [1, N]$ *avant* de commencer la séparation de sources, l'apprentissage

du réseau se fait en mode dit "global", c'est-à-dire que tous les exemples sont passés l'un après l'autre dans le réseau, et c'est après ce moment-là seulement que la fonction de coût est calculée. Au contraire, si l'on reçoit les $\boldsymbol{s}_t$ en "temps réel", alors on ne peut plus procéder ainsi. Dans ce cas, la fonction de coût est *estimée* grâce à un filtrage passe bas. C'est l'apprentissage dit "continu".

Dérivée du coût en mode global

En mode global, les moments peuvent être estimés par un moyennage sur les N échantillons disponibles:

$$\hat{R}_{\alpha_1 \ldots \alpha_n} = \frac{1}{N} \sum_{t=1}^{N} S_1^{\alpha_1}(t) \ldots S_n^{\alpha_n}(t)$$

et nous avons:

$$\hat{M}_{\alpha_1 \ldots \alpha_n} = \hat{R}_{\alpha_1 \ldots \alpha_n} - \hat{R}_{\alpha_1 \vec{\Theta}_1} \ldots \hat{R}_{\alpha_n \vec{\Theta}_n}$$

Il est donc possible d'écrire une estimation du coût:

$$\begin{aligned} \hat{C}_{md} &= \frac{1}{2} \left\{ \sum_{i=1}^{n} \mid \hat{R}_{\vec{\Theta}_i} \mid^2 + \sum_{i=1}^{n} \mid \hat{R}_{2\vec{\Theta}_i} - 1 \mid^2 \right. \\ &\quad \left. + \sum_{\alpha_1 + \ldots + \alpha_n \leq K} \; \sum_{\beta_1 + \ldots + \beta_n \leq K} G_{\alpha_1 \ldots \alpha_n, \beta_1 \ldots \beta_n} \hat{M}_{\alpha_1 \ldots \alpha_n} \hat{M}_{\beta_1 \ldots \beta_n} \right\} \end{aligned}$$

Pour le calcul de rétropropagation du gradient (*cf* Chapitre 1 Sections (2.3) et (2.4)), il est nécessaire de connaître la dérivée du coût par rapport aux poids du réseau. Pour un réseau à valeurs réelles, on a:

$$\frac{\partial \hat{C}_{md}}{\partial W_{ab}} = \sum_{t=1}^{N} \sum_{k=1}^{n} \frac{\partial \hat{C}_{md}}{\partial S_k(t)} \frac{\partial S_k(t)}{\partial W_{ab}}$$

où $\dfrac{\partial S_k(t)}{\partial W_{ab}}$ se calcule aisément en suivant les règles génériques établies au Chapitre 1, et qui ne dépendent que de la place de W_{ab} dans le réseau, et des fonctions de transition des neurones. Par ailleurs Burel démontre [2] que l'on a:

$$\begin{aligned} \frac{\partial \hat{C}_{md}}{\partial S_k(t)} &= \frac{1}{N} \left\{ \hat{R}_{\vec{\Theta}_k} + 2 \left(\hat{R}_{2\vec{\Theta}_k} - 1 \right) S_k(t) \right\} \\ &+ \frac{1}{N} \sum_{\substack{\alpha_1 + \ldots + \alpha_n \leq K \\ \alpha_k > 0}} \alpha_k S_k^{\alpha_k - 1}(t) \left(\prod_{\substack{l=1 \\ l \neq k}}^{n} S_l^{\alpha_l}(t) - \prod_{\substack{l=1 \\ l \neq k}}^{n} \hat{R}_{\alpha_l \vec{\Theta}_l} \right) \left(\sum_{\beta_1 + \ldots + \beta_n \leq K} G_{\alpha_1 \ldots \alpha_n, \beta_1 \ldots \beta_n} \hat{M}_{\beta_1 \ldots \beta_n} \right) \end{aligned}$$

Dérivée du coût en mode continu

En mode continu, les moments sont estimés à l'aide de filtres passe-bas:

$$\hat{R}_{\alpha_1...\alpha_n}(t) = (1-\beta)S_1^{\alpha_1}(t)...S_n^{\alpha_n}(t) + \beta\hat{R}_{\alpha_1...\alpha_n}(t-1)$$

Après un calcul très semblable à celui développé pour l'approche en mode global, Burel montre que pour le mode continu, la dérivée du coût s'exprime par:

$$\frac{\partial\hat{C}_{md}(t)}{\partial S_k(t)} = (1-\beta)\left\{\hat{R}_{\vec{\Theta}_k} + 2\left(\hat{R}_{2\vec{\Theta}_k} - 1\right)S_k(t)\right\}$$
$$+(1-\beta)\sum_{\substack{\alpha_1+...+\alpha_n\leq K \\ \alpha_k>0}} \alpha_k S_k^{\alpha_k-1}(t)\left(\prod_{\substack{l=1 \\ l\neq k}}^{n} S_l^{\alpha_l}(t) - \prod_{\substack{l=1 \\ l\neq k}}^{n}\hat{R}_{\alpha_l\vec{\Theta}_l}\right)\left(\sum_{\beta_1+...+\beta_n\leq K} G_{\alpha_1...\alpha_n,\beta_1...\beta_n}\hat{M}_{\beta_1...\beta_n}\right)$$

Une fois que la dérivée partielle du coût par rapport aux sorties est connue, les règles de rétropropagation de ce perceptron sont totalement similaires à celles présentées au Chapitre 1. Notons en outre que toutes ces règles, qui ont été présentées dans le cas réel, sont extrapolables au cas complexe, et l'on utilisera alors les règles de dérivation du gradient spécifiques au cas complexe (voir Chapitre 1 Section (2.4)).

6.4 Conclusion

La mesure de dépendance de Burel définie à l'équation (127) est une notion intéressante, parce qu'elle peut être reformulée comme une forme quadratique définie positive fonction des moments des signaux $\boldsymbol{s}_t$. De ce fait, annuler la mesure de dépendance sur les signaux $\boldsymbol{s}_t$ équivaut à rendre les composantes de $\boldsymbol{s}_t$ indépendantes. Par suite, le problème de séparation aveugle de sources peut être retranscrit comme la minimisation, par rétropropagation du gradient, d'un critère de sortie basé sur la mesure de dépendance des signaux obtenus sur la dernière couche d'un perceptron. On retrouve alors un schéma algorithmique très semblable à celui utilisé dans la mise en œuvre du Perceptron Complexe à Structure Contrainte, avec lequel il s'agissait également de minimiser un critère d'erreur en sortie par rétropropagation du gradient. Rappelons que la méthode du PCSC permettait non seulement d'estimer les angles d'arrivée $\omega_1, ..., \omega_n$ des sources, mais aussi d'estimer les sources $\boldsymbol{x}_t$ elles-mêmes. Cette remarque montre que les deux approches sont liées à la fois par le fond et par la forme, ce qui nous conduit à envisager l'étude de la fusion de ces deux approches.

Avant de clore cette section, il est intéressant de faire le point sur ce que

peuvent, et ce que ne peuvent pas, réaliser les méthodes de Séparation Aveugle de Sources. Comme leur nom l'indique, ces méthodes permettent de *séparer* des sources (indépendantes). Par contre, aucune méthode de SAS ne permet d'éliminer le bruit: si du bruit est présent dans le mélange, il se retrouve sur les estimations des sources. Eventuellement, si le bruit est trop fort, il peut perturber l'estimation et même faire échoir la recherche de l'indépendance sur les $\boldsymbol{s}_t$. Il suffit pour s'en convaincre de considérer à nouveau les équations générales du mélange et d'un séparateur:

$$\left\{ \begin{array}{rcl} \boldsymbol{y} & = & A\boldsymbol{x} + \boldsymbol{b} \\ \boldsymbol{s} & = & B\boldsymbol{y} \end{array} \right.$$

On peut alors exprimer les sorties en fonction des sources originales comme:

$$\boldsymbol{s} = BA\boldsymbol{x} + B\boldsymbol{b}$$

et puisque le séparateur idéal est tel que la matrice BA vaut l'identité (modulo une permutation et une dilatation que l'on omet ici), on peut écrire $B = A^{-1}$, ce qui donne au final

$$\boldsymbol{s} = \boldsymbol{x} + A^{-1}\boldsymbol{b}$$

Cette dernière équation montre bien que l'estimation $\boldsymbol{s}$ obtenue est de plus en plus éloignée de la source originale $\boldsymbol{x}$ au fur et à mesure que le bruit croît, même dans le cas extrême où le séparateur serait idéal.
Par contre, si le bruit peut être considéré comme l'une des n sources indépendantes qui ont servi à former le mélange, alors un algorithme de Séparation Aveugle de Sources est capable de rendre les n composantes du vecteur $\boldsymbol{s}_t$ (dont la composante "bruit") indépendantes, c'est-à-dire qu'il est capable (en principe) de retrouver les n sources originales.

7 Séparation Aveugle et Estimation d'Angles d'Arrivée

7.1 Introduction

Le problème de l'estimation d'Angles d'Arrivée (AdA) et le problème de Séparation Aveugle de Sources (SAS) ne semblent pas des problèmes très liés à première vue. Dans le premier, on cherche la direction des signaux que l'on reçoit, alors que dans le second on cherche plutôt à estimer la forme des signaux reçus. Aussi, pour tenter de faire comprendre au mieux la démarche qui nous a menés à fusionner les deux approches, nous présentons tout d'abord un rappel concis des hypothèses et

des conditions régissant chaque problème, en mettant le plus possible en exergue les parallèles qui peuvent être faits. Nous montrons ensuite dans quelle mesure les solutions apportées à ces deux problèmes présentent elles aussi des similitudes importantes.

7.1.1 Rappels sur les deux problèmes

De manière succinte, on peut introduire la comparaison entre les conditions régissant les deux problèmes comme suit:

- Estimation d'Angles d'Arrivée
  - signaux: n signaux reçus sur m capteurs, $m > n$
  - but: estimer les n angles d'arrivée des sources, et éventuellement estimer les n sources elles mêmes.
  - information utilisée: géométrie de l'antenne
  - aucune connaissance sur les sources
  - Hypothèses:
    - bruit décorrélé, indépendant des sources
    - $m > n$
- Séparation Aveugle de Sources
  - signaux: n signaux reçus sur m capteurs, avec $m \geq n$.
  - but: retrouver les n sources originelles
  - information utilisée:
    - indépendance des sources originelles
    - forme du mélange
  - aucune connaissance sur les sources (à part leur indépendance)
  - Hypothèses: niveau de bruit assez faible par rapport au niveau de puissance des sources. Sinon, des problèmes de convergence de l'algorithme peuvent survenir

7.1.2 Solutions neuromimétiques proposées

Pour chaque problème, nous allons présenter ici une solution qui, par sa forme et les techniques qu'elle utilise, permet de faire un lien direct entre l'approche AdA et l'approche SAS.

- Estimation d'Angles d'Arrivée
  - étape 1: estimation grossière des ω_k fournie par un perceptron réel à trois couches utilisant les informations contenues dans la matrice de covariance. Ces premières estimations $\tilde{\omega}_k$ sont utilisées pour initialiser les poids du second réseau.
  - étape 2: minimisation par descente de gradient de l'erreur quadratique en sortie d'un perceptron complexe à structure contrainte. La contrainte porte sur des informations géométriques relatives à l'antenne.
  - étape 3: après convergence, obtention d'une estimation des angles d'arrivée θ_k, et des sources originales $\boldsymbol{x}_t$
- Séparation Aveugle de Sources
  - étape 1: examen de la structure du mélange à inverser (linéaire ou non, forme des non-linéarités, etc) pour évaluation de la forme du mélange inverse et dénombrement des paramètres nécessaires à la construction du séparateur neuromimétique de Burel
  - étape 2: minimisation par descente de gradient d'un coût de dépendance en sortie d'un perceptron qui réalise le mélange inverse
  - étape 3: après convergence, les sorties $\boldsymbol{s}_t$ sont une estimation (modulo une dilatation et une permutation) des sources originales $\boldsymbol{x}_t$

Cette présentation montre que pour chaque problème, il est possible de mettre en œuvre une solution neuromimétique basée sur la minimisation d'un coût en sortie. Pour chaque problème, le coût à minimiser est basé de manière directe sur les informations dont on dispose; lorsque l'on dispose d'informations statistiques (sources indépendantes), le coût à minimiser est construit à partir d'une mesure de dépendance statistique. Lorsque l'on dispose d'informations géométriques (exemple: antenne linéaire uniforme), le coût à minimiser est une erreur quadratique moyenne et l'information géométrique est exploitée via une contrainte sur la matrice des poids de sortie du réseau.
Par conséquent, si les deux types d'informations - statistique et géométrique - sont disponibles, il serait intéressant d'étudier comment ces deux approches pourraient coopérer.

7.2 Fusion des approches

7.2.1 Structure du réseau réalisant la fusion

Plaçons nous dans le cas du problème d'estimation d'angles d'arrivée sur une antenne de capteurs, puis ajoutons à nos hypothèses habituelles les hypothèses supplémentaires nécessaires à la Séparation Aveugle de Sources. D'après la géométrie de l'antenne (représentée par la matrice directionnelle $A(\Omega)$), les vecteur reçus aux capteurs sont de la forme

$$\boldsymbol{y}_t = A(\Omega)\boldsymbol{x}_t + \boldsymbol{b}_t$$

où le vecteur $\boldsymbol{x}_t$ représente les sources et $\boldsymbol{b}_t$ représente le bruit. Les hypothèses habituelles du problème d'AdA n'exigent pas que les sources soient indépendantes, mais demandent que le nombre de capteurs soit strictement supérieur au nombre de sources: $m > n$. Le Perceptron Complexe à Structure Contrainte introduit pour résoudre le problème est montré figure (22). Si on souhaite maintenant résoudre

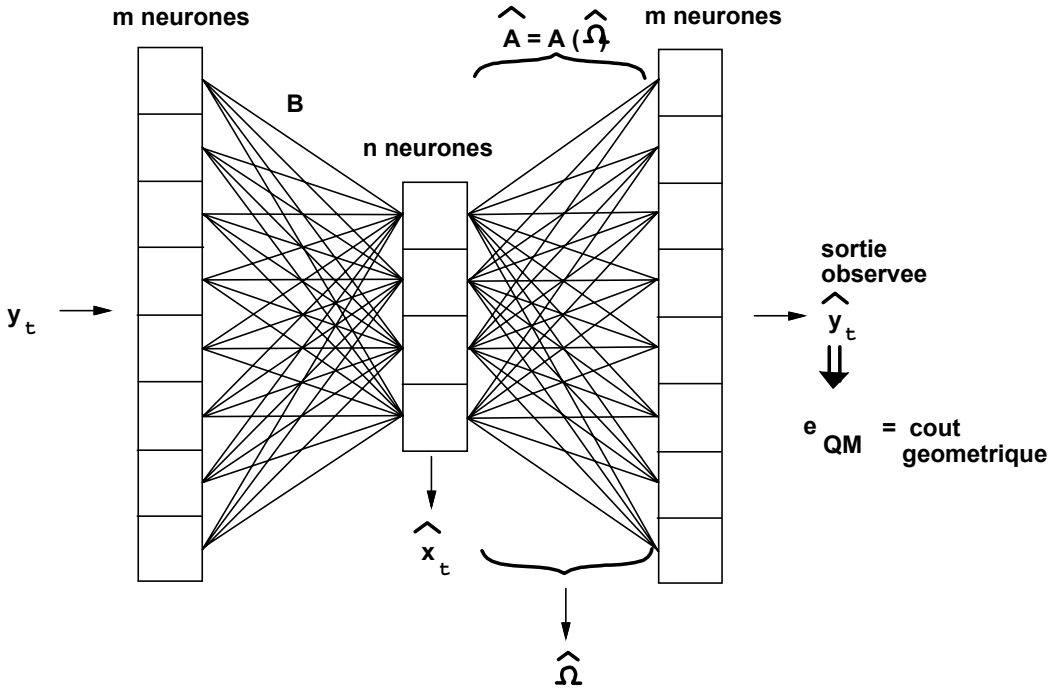

Figure 22: *Le perceptron complexe à structure contrainte minimisant un coût géométrique en sortie pour le problème d'AdA*

ce problème en le considérant à travers l'approche Séparation Aveugle de Sources, il est nécessaire d'introduire des hypothèses supplémentaires. Nous supposerons

pour commencer que le nombre de capteurs peut être égal au nombre de sources: $m \geq n$. Dans ces conditions, toutes les approches "classiques" d'estimation des AdA (MUSIC, ESPRIT, MV, etc) échouent dès que $m = n$, car les informations au deuxième ordre qu'elles utilisent ne sont plus suffisantes pour estimer les ω_k dès lors que $m = n$ (*cf* [29]). Toutefois, l'approche SAS introduit également l'hypothèse que les sources originales $\boldsymbol{x}_t$ sont indépendantes, ce qui permet d'envisager l'utilisation de statistiques d'ordre supérieur, et donc de pouvoir éventuellement résoudre le problème d'AdA grâce aux équations supplémentaires issues de l'approche statistique.

Le "mélange" des sources $\boldsymbol{x}_t$ est un mélange linéaire:

$$\boldsymbol{y}_t = A(\Omega)\boldsymbol{x}_t$$

auquel est ajouté un bruit $\boldsymbol{b}_t$ indépendant. Répétons que le mélange $\boldsymbol{y}_t$ est linéaire en $\boldsymbol{x}_t$ (puisqu'il est réalisé par la *matrice* $A(\Omega)$), bien que la matrice $A(\Omega)$ ne soit pas elle-même linéaire vis-à-vis des ω_k formant le vecteur Ω. La Séparation Aveugle de Sources peut donc être réalisée par la minimisation du coût de dépendance en sortie du perceptron représenté figure (23), où la matrice B représente le mélange linéaire

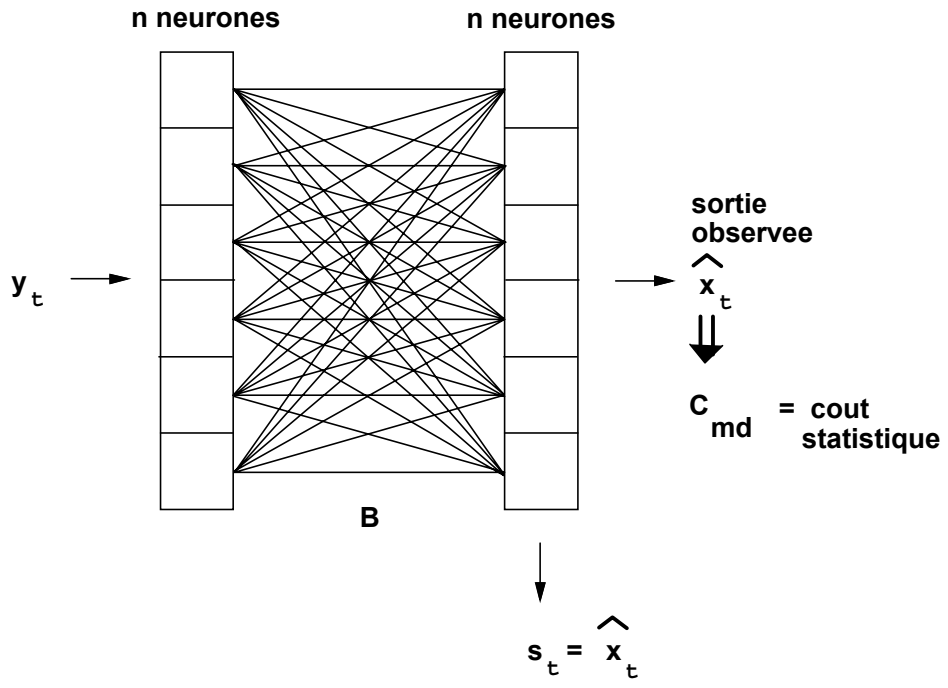

Figure 23: *Le perceptron complexe linéaire à deux couches minimisant un coût de dépendance statistique en sortie pour le problème de SAS*

inverse.

D'après ces considérations, on peut envisager la fusion des deux approches

comme suit: un perceptron complexe à structure contrainte est construit comme indiqué sur la figure (24). Le nombre de neurones est de m sur les couches d'entrée

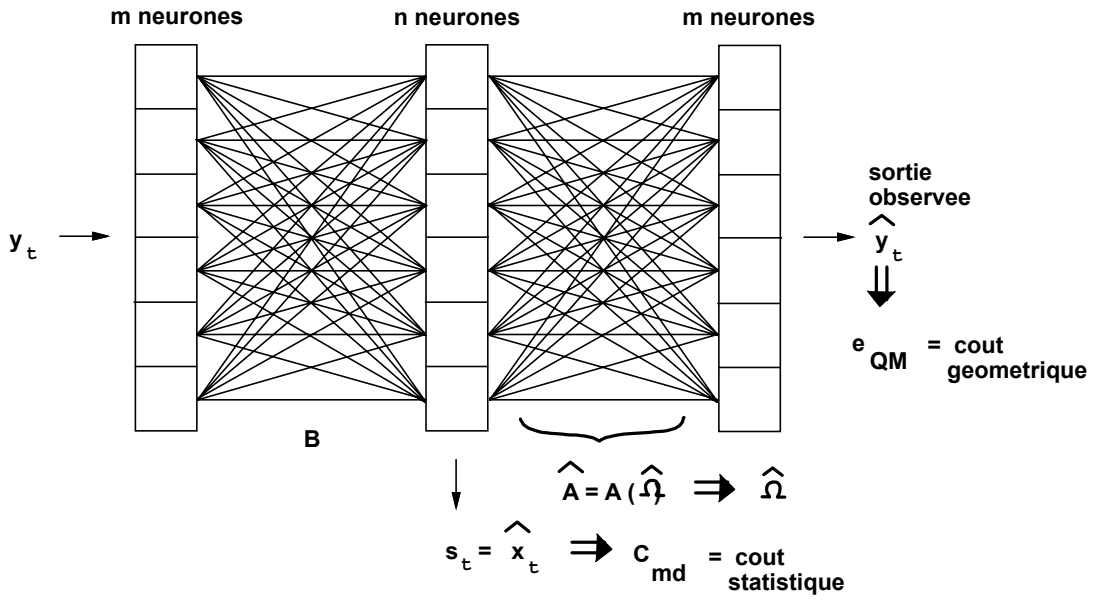

Figure 24: *Le perceptron complexe linéaire à trois couches minimisant à la fois un coût de dépendance statistique et un coût de dépendance géométrique pour le problème de SAS combiné au problème d'AdA*

et de sortie, et de n sur la couche cachée. Dans le cas où $m = n$, les trois couches comportent chacune n neurones. La matrice des poids de sortie $\hat{A}$ est contrainte à suivre la forme $A(\hat{\Omega})$; elle est initialisée grâce à l'estimation "grossière" $\tilde{\Omega}$ fournie par le perceptron réel introduit en Section 4 : $\hat{A}_{init} = A(\tilde{\Omega})$. Dans le cas général (vu précédemment) on initialise la matrice B des poids d'entrée comme la pseudoinverse de la matrice $\hat{A}_{init}$, soit $B_{init} = \hat{A}^{+}_{init}$. Dans le cas extrême où $m = n$, la matrice $\hat{A}$ est carrée $(n \times n)$, et la matrice B est tout simplement initialisée comme l'inverse de la matrice $\hat{A}_{init}$, soit $B_{init} = \hat{A}^{-1}_{init}$. Notons que l'inversibilité de la matrice directionnelle $A(\Omega)$ est assurée par le fait que le problème que nous cherchons à résoudre est *supposé* résoluble! En effet, si A n'était pas inversible, elle serait de rang inférieur à n (ce qui est le signe d'une antenne mal conçue). Par suite les vecteurs $\boldsymbol{y}_t$ formeraient un sous-espace de dimension inférieure à n. Il serait donc matériellement impossible, à partir des $\boldsymbol{y}_t$, de retrouver les n sources indépendantes formant le vecteur $\boldsymbol{x}_t$.

7.2.2 Critère à minimiser

Partons de l'estimation grossière $\tilde{\Omega}$ fournie par le perceptron réel. L'initialisation des matrices de poids du réseau de fusion (figure (24)) donne $\hat{A}_{init} = A(\tilde{\Omega})$ et

$B_{init} = \hat{A}^{+}_{init}$. Si $m = n$, les algorithmes de Haute-Résolution échouent, d'après nos remarques précédentes. Que se passe-t-il pour le PCSC? Pour $m = n$ on a $B_{init} = \hat{A}^{-1}_{init}$, ce qui implique:

$$\begin{aligned} \hat{\boldsymbol{y}}_t &= \hat{A}_{init} B_{init} \boldsymbol{y}_t \\ &= \hat{A}_{init} \hat{A}^{-1}_{init} \boldsymbol{y}_t \\ &= \boldsymbol{y}_t \end{aligned}$$

L'erreur quadratique moyenne en sortie

$$e_{QM}(\hat{\boldsymbol{y}}) = \frac{1}{N} \sum_{t=1}^{N} \frac{1}{2} ||\boldsymbol{y_t} - \hat{\boldsymbol{y}}_t||^2$$

est donc identiquement nulle lorsque le réseau est dans son état initial, ce qui confirme le fait que pour n capteurs et n sources, aucune des méthodes au second ordre (comme ici le PCSC et le MV) ne peut estimer les angles d'arrivée avec les éléments dont elle dispose. Plus précisément, en ce qui concerne le PCSC, on constate d'après l'équation précédente que le réseau ne peut pas faire varier ses poids, à moins qu'on ne lui fournisse un autre apport d'informations. C'est pourquoi nous choisissons d'utiliser une fonction de coût qui est basée sur les deux approches à la fois: une partie du coût ($e_{QM}(\hat{\boldsymbol{y}})$) contraint le réseau à respecter la géométrie de l'antenne, et l'autre partie du coût ($C_{md}(\hat{\boldsymbol{x}})$) pousse le réseau à rendre les composantes du vecteur $\hat{\boldsymbol{x}}_t$ indépendantes. Le coût total

$$e_{fus}(\hat{\boldsymbol{y}}, \hat{\boldsymbol{x}}) = e_{QM}(\hat{\boldsymbol{y}}) + \gamma C_{md}(\hat{\boldsymbol{x}})$$

fait donc appel au coût de dépendance introduit à la section 6 (d'après la mesure de dépendance de Burel [2], [3]) via un coefficient de pondération γ positif. Ce coût de dépendance $C_{md}(\hat{\boldsymbol{x}})$ est calculé à partir des valeurs des neurones obtenues sur la couche cachée du réseau (les vecteurs $\hat{\boldsymbol{x}}_t$). Observons que sur une couche cachée:

$$\hat{\boldsymbol{x}}_t = B\boldsymbol{y}_t = BA\boldsymbol{x}_t + B\boldsymbol{b}_t$$

d'après le modèle du mélange. De ce fait, si le rapport Signal à Bruit n'est pas trop faible, on peut écrire:

$$\hat{\boldsymbol{x}}_t \simeq BA\boldsymbol{x}_t$$

où les $\boldsymbol{x}_t$ sont indépendants par hypothèse. On en conclut que le minimum de $C_{md}(\hat{\boldsymbol{x}})$ est atteint lorsque BA est égal au produit d'une matrice de dilatation par une matrice de permutation (en effet, seule une telle égalité - à savoir $\hat{\boldsymbol{x}}_t = PD\boldsymbol{x}_t$ - assure que les $\hat{\boldsymbol{x}}_t$ soient statistiquement indépendants).

Remarque:
Dans la section 7, le coût de dépendance utilisé pour la séparation de sources est la somme de trois termes (équation (130)):

$$C_{md} = \frac{1}{2}\left\{\sum_{i=1}^{n} \mid E\{S_i\} \mid^2 + \sum_{i=1}^{n} \mid E\{{S_i}^2\} - 1 \mid^2 + m_d(\hat{\boldsymbol{x}})\right\}$$

où les deux premiers termes visent à normaliser le vecteur de sortie $\boldsymbol{s}_t = \hat{\boldsymbol{x}}_t$ en moyenne et en variance. En effet, pour éviter que la solution trouvée ne soit identiquement nulle (solution triviale), il est nécessaire de forcer la variance des composantes de $\hat{\boldsymbol{x}}_t$ à être non nulle. Comme d'autre part il n'est pas possible de retrouver les composantes originales de $\boldsymbol{x}_t$, mais seulement une estimation $\hat{\boldsymbol{x}}_t = PD\boldsymbol{x}_t$ modulo une permutation P et une dilatation D, il apparaît judicieux, si on ne dispose d'aucune référence quant au niveau de puissance des sources, de normaliser la solution en moyenne ($E\{S_i\} = 0$) et en variance ($E\{{S_i}^2\} = 1$).

Il en va tout autrement lorsque la Séparation Aveugle de Sources est fusionnée avec un estimateur d'AdA. En effet, lorsqu'il y a fusion des deux approches, la minimisation du critère géométrique

$$e_{QM}(\hat{\boldsymbol{y}}) = \frac{1}{N}\sum_{t=1}^{N}\frac{1}{2}||\boldsymbol{y_t} - \hat{\boldsymbol{y}}_t||^2$$

implique que $\hat{\boldsymbol{y}}_t$ doit être aussi proche que possible de $\boldsymbol{y}_t$. Si l'une des sources estimées dans $\hat{\boldsymbol{x}}_t$ était identiquement nulle, alors d'après la figure (24), on aurait

$$\hat{\boldsymbol{y}}_t = A(\hat{\Omega})\hat{\boldsymbol{x}}_t$$

qui serait confiné à un sous-espace vectoriel de dimension $n-1$, alors que $\boldsymbol{y}_t$ génère par hypothèse un sous-espace vectoriel de dimension n. De ce fait, $\hat{\boldsymbol{y}}_t$ ne pourrait être aussi proche que l'on souhaite de $\boldsymbol{y}_t$, ce qui ferait augmenter le coût géométrique $e_{QM}(\hat{\boldsymbol{y}})$, et par suite ferait augmenter le coût total $e_{fus}(\hat{\boldsymbol{y}}, \hat{\boldsymbol{x}})$. On voit donc que la minimisation du coût géométrique $e_{QM}(\hat{\boldsymbol{y}})$ inclut déjà implicitement une contrainte sur la variance et la moyenne de $\hat{\boldsymbol{y}}_t$, et par conséquent, une contrainte sur la variance et la moyenne de $\hat{\boldsymbol{x}}_t$. En conséquence, les contraintes en normalisation

$$E\{{S_i}^2\} = 1 \qquad \text{et} \qquad E\{S_i\} = 0$$

conseillées par Burel dans la SAS pure, ne seront pas utilisées dans notre approche de fusion.
Finalement, le coût global à minimiser par le réseau fusion est:

$$e_{fus}(\hat{\boldsymbol{y}}, \hat{\boldsymbol{x}}) = e_{QM}(\hat{\boldsymbol{y}}) + \gamma m_d(\hat{\boldsymbol{x}})$$

où $m_d(\hat{\boldsymbol{x}})$ est la mesure de dépendance de Burel appliquée au vecteur $\hat{\boldsymbol{x}}_t$ de la couche cachée (voir réseau figure (24)).

7.2.3 Règles de rétropropagation

On a vu à la section 5 que les règles de rétropropagation dans un perceptron complexe à structure contrainte s'effectuaient en deux étapes: tout d'abord une rétropropagation contrainte de la couche de sortie vers la couche cachée puis une rétropropagation "classique" de la couche cachée vers la couche de sortie. A ces règles viennent désormais s'ajouter les règles propres à la minimisation du coût de dépendance. Par suite, les règles de dérivation du réseau opérant la fusion se présentent comme suit:

- rétropropagation contrainte de la couche de sortie vers la couche cachée par calcul de $\frac{\partial e_{QM}(\hat{\boldsymbol{y}})}{\partial \hat{\omega}_a}$ selon les règles détaillées aux équations (123) et (125). D'où un renouvellement des $\hat{\omega}_a$ par

$$\Delta\hat{\omega}_a = -\alpha\frac{\partial e_{QM}}{\partial\hat{\omega}_a}$$

- rétropropagation pondérée de la couche cachée vers la couche d'entrée:

$$\Delta W_{ab}^B = -\alpha\frac{\partial e_{fus}(\hat{\boldsymbol{y}},\hat{\boldsymbol{x}})}{\partial W_{ab}^B} = -\alpha\frac{\partial e_{QM}(\hat{\boldsymbol{y}})}{\partial W_{ab}^B} - \alpha\gamma\frac{\partial m_d(\hat{\boldsymbol{x}})}{\partial W_{ab}^B} \tag{131}$$

  où W_{ab}^B est un poids de la matrice B (matrice reliant la couche d'entrée à la couche cachée).

Le premier terme de l'équation (131) a été calculé précédemment (*cf* équations (124) et (126)) et représente le renouvellement des poids de la matrice d'entrée B dans un Perceptron Complexe à Structure Contrainte. Le second terme est la pondération par γ du renouvellement des poids de la matrice inverse du mélange. Ce terme a été détaillé par Burel [2] pour un mélange de sources réelles:

$$\frac{\partial m_d(S)}{\partial W_{ab}} = \sum_{t=1}^{N}\sum_{k=1}^{n}\frac{\partial m_d(S)}{\partial S_k(t)}\frac{\partial S_k(t)}{\partial W_{ab}}$$

Pour nous, les sources $S(t)$ sont représentées par le vecteur *complexe* $\hat{\boldsymbol{x}}_t$, d'où une nouvelle expression pour le renouvellement des poids:

$$\frac{\partial m_d(\hat{\boldsymbol{x}})}{\partial W_{ab}} = \sum_{t=1}^{N}\sum_{k=1}^{n}\frac{\partial m_d(\hat{\boldsymbol{x}})}{\partial \hat{x}_{tk}}\left(\frac{\partial \hat{x}_{tk}}{\partial W_{ab}}\right)^* \tag{132}$$

d'après la propriété 1 (Chapitre 1 équation (15)). L'équation précédente est une équation générique pouvant s'adapter à tout type de perceptron (deux couches ou plus). Pour le cas qui nous intéresse, il n'y a qu'une seule matrice de poids (à savoir B) entre la couche d'entrée $\boldsymbol{y}_t$ et la couche où l'on souhaite recueillir une estimation

des sources $\hat{\boldsymbol{x}}_t$. De ce fait, les seuls poids W_{ab} concernés par cette dérivation sont les poids W_{ab}^B de la matrice B. Par ailleurs, le poids W_{ab}^B pondère par définition la connexion entre le neurone a de la couche d'entrée et le neurone b de la couche cachée. On en déduit que le seul indice k pour lequel $\hat{x}_{tk}$ soit fonction de W_{ab}^B est $k = b$. Par suite, l'équation (132) se simplifie encore:

$$\frac{\partial m_d(\hat{\boldsymbol{x}})}{\partial W_{ab}^B} = \sum_{t=1}^{N} \frac{\partial m_d(\hat{\boldsymbol{x}})}{\partial \hat{x}_{tb}} \left(\frac{\partial \hat{x}_{tb}}{\partial W_{ab}^B} \right)^*$$

et comme la fonction de transition des neurones est l'identité, on a $\hat{x}_{tb} = \sum_a W_{ab}^B y_{ta}$, d'où

$$\frac{\partial \hat{x}_{tb}}{\partial W_{ab}^B} = y_{ta}$$

Il ne reste plus qu'à détailler la dérivation de $m_d(\hat{\boldsymbol{x}})$ par rapport aux neurones $\hat{x}_{tb}$ de la couche cachée. Dans [2], Burel indique la règle de dérivation en mode global de la mesure de dépendance $\hat{m}_d(S)$ par rapport à la variable réelle indépendante $S_k(t)$:

$$\frac{\partial m_d(S)}{\partial S_k(t)} = \frac{1}{N} \sum_{\substack{\alpha_1+\ldots+\alpha_n \leq K \\ \alpha_k > 0}} \alpha_k S_k^{\alpha_k - 1}(t) \left(\prod_{\substack{l=1 \\ l \neq j}}^{n} S_l^{\alpha_l}(t) - \prod_{\substack{l=1 \\ l \neq j}}^{n} \hat{R}_{\alpha_l \vec{\Theta}_l} \right) \left(\sum_{\beta_1+\ldots+\beta_n \leq K} G_{\alpha_1 \ldots \alpha_n, \beta_1 \ldots \beta_n} \hat{M}_{\beta_1 \ldots \beta_n} \right) \tag{133}$$

Or ici, nous utilisons des variables complexes $\hat{x}_{tb}$ qui sont indépendantes entre elles, mais dont les parties réelles et imaginaires sont aussi indépendantes entre elles. Ce que nous cherchons à obtenir est donc l'indépendance des $2n$ variables réelles et imaginaires de $\hat{\boldsymbol{x}}_t$. En posant

$$S = \left[\hat{x}_1^R, \hat{x}_1^I, \hat{x}_2^R, ..., \hat{x}_n^I \right]^T \tag{134}$$

où S est un vecteur réel de longueur $2n$, nous pouvons utiliser l'équation (133) telle que. Maintenant que nous savons comment calculer les dérivées de m_d par rapport aux parties réelles et imaginaires des composantes de $\hat{\boldsymbol{x}}_t$, il suffit de se remémorer la convention de la dérivation par rapport à un complexe (*cf* Chapitre 1 Section (2.4)) pour écrire

$$\frac{\partial m_d(\hat{\boldsymbol{x}})}{\partial \hat{x}_{tb}} = \frac{\partial m_d(\hat{\boldsymbol{x}})}{\partial \hat{x}_{tb}^R} + j \frac{\partial m_d(\hat{\boldsymbol{x}})}{\partial \hat{x}_{tb}^I}$$

La dérivation de la mesure de dépendance par rapport à un poids de la matrice B est donc complètement explicitée:

$$\frac{\partial m_d(\hat{\boldsymbol{x}})}{\partial W_{ab}^B} = \sum_{t=1}^{N} \left(\frac{\partial m_d(S)}{\partial S_{2b-1}(t)} + j \frac{\partial m_d(S)}{\partial S_{2b}(t)} \right) y_{ta}^*$$

où pour $k \in [1, 2n]$, $\dfrac{\partial m_d(S)}{\partial S_k(t)}$ est donné par l'équation (133).

Remarque: Comme on a ici un mélange linéaire, il suffit, pour que les variables soient indépendantes, qu'elles le soient deux à deux. Une variante possible consiste donc à remplacer la mesure de dépendance globale par une somme de mesures de dépendance sur les couples de composantes de (134). Dans le cas particulier où certaines sources n'auraient pas des parties réelles et imaginaires indépendantes, les couples correspondants ne doivent pas être pris en compte.

7.3 Initialisation du réseau de fusion

7.3.1 Rappels sur l'initialisation du PCSC

On a vu que le Perceptron Complexe à Structure Contrainte (PCSC) minimise un critère équivalent à celui du Maximum de Vraisemblance, mais en un temps bien moindre puisque cette structure est basée sur la rétropropagation par descente de gradient. Le Maximum de Vraisemblance est basé sur l'utilisation d'information au second ordre (voir Annexe D à ce sujet), et par voie de conséquence il en est de même pour le Perceptron Complexe à Structure Contrainte.

On a vu par ailleurs que pour compenser le défaut des systèmes basés sur une minimisation par descente de gradient - à savoir la tendance à se retrouver piégé dans un minimum local - il était utile d'initialiser les paramètres internes du PCSC à l'aide d'un premier perceptron, que l'on pourrait appeler PRI (pour Perceptron Réel d'Initialisation).

Puisque le PCSC n'utilise que de l'information au second ordre, il apparaît a posteriori très sensé d'envoyer au PRI de l'information au second ordre également, à savoir une version normalisée de la matrice de covariance:

$$\hat{R}^{nor} = \frac{m}{\sum_{i=1}^{m} \left(\hat{R}\right)_{ii}} \hat{R}$$

où $\hat{R}$ est l'estimation de la matrice de covariance relative aux signaux $\boldsymbol{y}_t$ reçus aux capteurs:

$$\hat{R} = \frac{1}{N} \sum_{t=1}^{N} \boldsymbol{y}_t \boldsymbol{y}_t^H$$

Toutefois, dès lors que le PCSC est associé à une approche par séparation aveugle,

c'est-à-dire une approche où les seules informations au second ordre ne suffisent plus, il apparaît que les données fournies au Perceptron Réel d'Initialisation doivent elles aussi comporter des informations d'ordre supérieur.

7.3.2 Structure de l'information reçue par le PRI

Information à l'ordre 2

La résolution du problème d'AdA simple basée sur le PRI est fondée sur une étude théorique qui montre que les statistiques au second ordre sur les signaux reçus aux capteurs fournissent suffisamment d'équations par rapport au nombre d'inconnues. Plaçons-nous en effet dans un cas de problème d'Angle d'Arrivée où, pour simplifier, on dira que les sources sont indépendantes et que le bruit est nul. On a alors, en allégeant les notations usuelles:

$$\boldsymbol{y} = A\boldsymbol{x}$$

d'où

$$R_y = AR_xA^H \tag{135}$$

où R_x est la matrice diagonale de puissances des sources:

$$R_x = \text{diag}\,[P_1, P_2, ..., P_n]$$

avec $\forall k \in [1, n],\ P_k = E\{|x_k|^2\}$, et A est la matrice directionnelle:

$$A = \begin{bmatrix} 1 & 1 & ... & 1 \\ e^{-j\omega_1} & e^{-j\omega_2} & ... & e^{-j\omega_n} \\ \vdots & \vdots & \ddots & \vdots \\ e^{-j(m-1)\omega_1} & e^{-j(m-1)\omega_2} & ... & e^{-j(m-1)\omega_n} \end{bmatrix}$$

Après développement du calcul dans (135), on obtient

$$E\{\begin{pmatrix} |y_1|^2 & y_1y_2^* & ... & y_1y_m^* \\ y_1^*y_2 & |y_2|^2 & ... & y_2y_m^* \\ \vdots & \vdots & \ddots & \vdots \\ y_1^*y_m & y_2^*y_m & ... & |y_m|^2 \end{pmatrix}\} \quad =$$

$$\sum_{k=1}^{n} \begin{pmatrix} P_k & P_ke^{j\omega_k} & P_ke^{j2\omega_k} & ... & P_ke^{j(m-1)\omega_k} \\ P_ke^{-j\omega_k} & P_k & P_ke^{j\omega_k} & ... & P_ke^{j(m-2)\omega_k} \\ P_ke^{-j2\omega_k} & P_ke^{-j\omega_k} & P_k & ... & P_ke^{j(m-3)\omega_k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ P_ke^{-j(m-1)\omega_k} & P_ke^{-j(m-2)\omega_k} & P_ke^{-j(m-3)\omega_k} & ... & P_k \end{pmatrix}$$

Il est intéressant de remarquer que cette dernière matrice est non seulement de symétrie hermitienne, mais de plus chacune de ses sous-diagonales est composée d'éléments identiques entre eux. Ainsi, sur tous les éléments de la première sous-diagonale du haut, on retrouve le terme

$$\sum_{k=1}^{n} P_k e^{j\omega_k}$$

tandis que la première sous-diagonale du bas est formée du terme $\sum_{k=1}^{n} P_k e^{-j\omega_k}$ répété $m-1$ fois. Il en va de même pour toutes les autres sous-diagonales. Finalement, on s'aperçoit que la connaissance de la première colonne de la matrice suffit pour connaître la matrice toute entière. Cette colonne est composée de $m-1$ complexes et 1 réel, soit finalement $2(m-1)+1 = 2m-1$ réels. La connaissance de la matrice de covariance (terme de droite) nous assure donc $2m-1$ équations au plus, pour $2n$ inconnues (les P_k et les ω_k). La condition nécessaire (et suffisante d'après l'étude de Wax [29]) pour que le système soit résoluble est donc

$$2m-1 \geq 2n$$

soit

$$m \geq n + \frac{1}{2}$$

où m et n sont des entiers; finalement, cette condition revient à

$$m > n$$

Cela prouve que quand $m > n$, ce qui est la cas dans un problème d'AdA classique, les informations fournies par la matrice de covariance R_y sont théoriquement suffisantes à la résolution du problème. Par contre lorsque $m = n$ (comme cela peut être le cas dans l'approche de fusion entre AdA et Séparation Aveugle de Sources), la connaissance de la matrice de covariance ne suffit plus.

Information à l'ordre 4

La matrice de covariance est formée des statistiques d'ordre 2:

$$R_{ij} = E\{y_i y_j^*\} \qquad (i,j) \in [1,m]^2$$

calculées sur les signaux capteurs $\boldsymbol{y}_t$. Lorsque $m = n$, ces statistiques ne suffisent plus et l'on doit utiliser des statistiques d'ordre supérieur, à commencer par les tenseurs[8] d'ordre 4:

[8] On emploie ici le terme "tenseur" par commodité. Si l'on souhaitait utiliser effectivement des propriétés tensorielles (ce qui n'est pas nécessaire pour notre propos), il faudrait vérifier le caractère tensoriel de T_{ijkl}.

$$T_{ijkl} = E\{y_i y_j^* y_k y_l^*\} \qquad (i,j,k,l) \in [1,m]^4$$

Remarque: Pour que cette information soit non triviale, il faut que les signaux $\boldsymbol{y}_t$ soient non gaussiens (toute statistique supérieure à l'ordre 2 d'une variable gaussienne se déduit des statistiques d'ordre inférieur ou égal à 2). Il faut par conséquent que les sources $\boldsymbol{x}_t$ elles-mêmes soient non gaussiennes.

Utilisation pratique des informations contenues dans le tenseur T d'ordre 4

Rappelons-nous que sur les $2m^2$ (ou ici $2n^2$ si $m = n$) valeurs réelles a priori contenues dans la matrice de covariance $\hat{R}_{nor}$, seules m^2 valeurs avaient été retenues. En effet, cette matrice étant hermitienne par définition, sa diagonale est réelle et tous ses éléments imaginaires sont antisymétriques. On en avait déduit que seules m^2 valeurs réelles a priori indépendantes pouvaient être utilisées.

De la même façon pour le tenseur d'ordre 4, il apparaît que certaines valeurs se répètent clairement une ou plusieurs fois. En effet, observons que:

$$\begin{aligned} T_{ijkl} &= T_{klij} \\ T_{ijji} &= T_{iijj} \\ T_{ijkl} &= T_{jilk}^* \end{aligned}$$

Ainsi, parmi les $2m^4$ réels a priori contenus dans T, il va être possible d'en retenir beaucoup moins. Les trois types d'égalité mises en exergue ci-dessus vont être utilisées pour diminuer le nombre minimum d'éléments de T représentant de l'information non redondante.

- $T_{ijkl} = T_{klij}$:
  on enlève la moitié des T concernés par cette égalité.
  - T non concernés: les T tels que $i = k$ et $j = l$, soit m^2 cas.
  - T concernés: le reste, soit $m^4 - m^2$ cas.

  Si on enlève la moitié des T concernés, il reste donc finalement

$$m^4 - \frac{1}{2}(m^4 - m^2) = \frac{m^4 + m^2}{2}$$

- $\underline{T_{ijji} = T_{iijj}}$:
  on garde les T_{ijji} et on enlève les T_{iijj}. Ne sont donc concernés que les T_{iijj} avec $i \neq j$. Il y a au total $m(m-1)$ T dans ce cas, dont la moitié a déjà été éliminée à l'itération précédente, puisqu'ils ne faisaient pas partie de l'exception. Il faut donc enlever $\frac{1}{2}m(m-1)$ variables T. Il en reste alors

$$\frac{m^4+m^2}{2} - \frac{m(m-1)}{2} = \frac{m^4+m}{2}$$

- $\underline{T_{ijkl} = T^*_{jilk}}$:
  on enlève la moitié des T concernés par cette égalité.

  - T non concernés:
    - les T_{iiii}, qui sont au nombre de m
    - les T_{ijji} où $j \neq i$, qui sont au même nombre que les T_{iijj} avec $i \neq j$ avant qu'ils ne soient éliminés à l'itération précédente, à savoir $\frac{1}{2}m(m-1)$. En effet, ces T_{ijji} où $j \neq i$ ne sont pas concernés, puisque d'après la première itération, on a déjà pris en compte l'égalité (1), qui correspond aussi à l'égalité (3) dans ce cas précis: $T_{ijji} = T_{jiij}$.

      $\hookrightarrow$ total des T non concernés:

$$m + \frac{m(m-1)}{2} = \frac{m^2+m}{2}$$

  - T concernés: le reste, soit

$$\frac{m^4+m}{2} - \frac{m^2+m}{2} = \frac{m^4-m^2}{2}$$

    Si on enlève la moitié des T concernés, il reste donc finalement

$$\frac{m^4+m}{2} - \frac{m^4-m^2}{4} = \frac{m^4+m^2+2m}{4}$$

Ce nombre final correspond au nombre des T_{ijkl} qui sont a priori indépendants. Les T_{ijkl} purement réels sont les T_{iiii}, qui sont au nombre de m, et les T_{ijji} où $j \neq i$ restants, soit $\frac{1}{2}m(m-1)$. Il y a donc $\frac{m^2+m}{2}$ T_{ijkl} réels, et les autres sont pleinement complexes, soit

$$\frac{m^4+m^2+2m}{4} - \frac{m^2+m}{2} = \frac{m^4-m^2}{4}$$

complexes. Au grand total, nous obtenons finalement

$$\frac{m^2+m}{2}+2\frac{m^4-m^2}{4}=\frac{m^4+m}{2}$$

réels a priori indépendants à partir du tenseur T_{ijkl} d'ordre 4.

Quand dans cette approche à l'ordre 4 nous ne considérons que $\frac{m^4+m}{2}$ parmi les $2m^4$ réels contenus dans le tenseur $\hat{T}$, il faut faire le parallèle avec l'approche au second ordre où on n'avait considéré que m^2 parmi les $2m^2$ réels contenus dans la matrice $\hat{R}^{nor}$.

Normalisation du tenseur d'ordre 4

On se rappelle que la matrice de covariance $\hat{R}$ estimée d'après les signaux reçus aux capteurs avait été normalisée dans le but de la rendre plus robuste par rapport aux changements de puissance éventuels des sources. Cette normalisation a été obtenue en divisant tous les éléments de $\hat{R}$ par la moyenne des réels formant sa diagonale. Les éléments de la diagonale de R peuvent se décrire littéralement sous la forme

$$E\{|y_i|^2\}$$

et sont réels.

Nous nous inspirerons de ces remarques pour la normalisation du tenseur d'ordre 4. Observons tout d'abord qu'il ne semble pas très judicieux de fournir en entrée du Perceptron Réel d'Initialisation des données qui ne sont ni du même type, ni du même ordre. En effet, les données issues de R sont du type $E\{y_i y_j^*\}$ et celles issues de T sont du type $E\{y_i y_j^* y_k y_l^*\}$. On voit que les rapports entre les T_{ijkl} et les R_{ij} sont de l'ordre du carré. Pour remédier à ce défaut, nous allons tout d'abord modifier les données issues des informations d'ordre 4 en prenant la racine carrée de leur module:

$$T_{ijkl}=\rho e^{j\theta} \quad \longmapsto \quad T^r_{ijkl}=\sqrt{\rho}e^{j\theta}=\frac{T_{ijkl}}{|T_{ijkl}|^{1/2}}$$

De cette façon, les modules des rapports entre les données au second ordre et celles à l'ordre 4 devraient être proches de 1 en moyenne. D'autre part, les informations contenues dans les phases des T_{ijkl} ont été conservées. Il reste maintenant à normaliser les T^r_{ijkl} par rapport à des variations de puissance des sources. Puisque les R_{ij} et les T^r_{ijkl} sont maintenant de même ordre, on choisit de normaliser les T^r_{ijkl} en les divisant par la même valeur que celle utilisée pour normaliser $\hat{R}$, à savoir la moyenne des éléments de la diagonale de $\hat{R}$:

$$\hat{T}^{nor}_{ijkl} = \frac{n}{\sum_{i=1}^{n}\left(\hat{R}\right)_{ii}}\hat{T}^{r}_{ijkl} = \frac{n}{\sum_{i=1}^{n}\left(\hat{R}\right)_{ii}} \frac{\hat{T}_{ijkl}}{|\hat{T}_{ijkl}|^{1/2}}$$

Ces éléments réduits et normalisés du tenseur T nous fournissent $\frac{m^4+m}{2}$ réels[9] qui sont à associer aux m^2 réels de la matrice $\hat{R}^{nor}$. Le Perceptron Réel d'Initialisation adapté à l'approche fusion comprend donc

$$\frac{m^4+m}{2} + m^2 = \frac{m^4+2m^2+m}{2}$$

unités sur sa couche d'entrée. Pour un problème à deux sources et deux capteurs, cela donne 13 neurones d'entrée. Sa couche de sortie contient comme auparavant m neurones ($m = n = 2$ pour l'exemple considéré). La couche cachée sera composée de l neurones, avec l compris entre n et $\frac{m^4+2m^2+m}{2}$. Nous proposons de choisir l

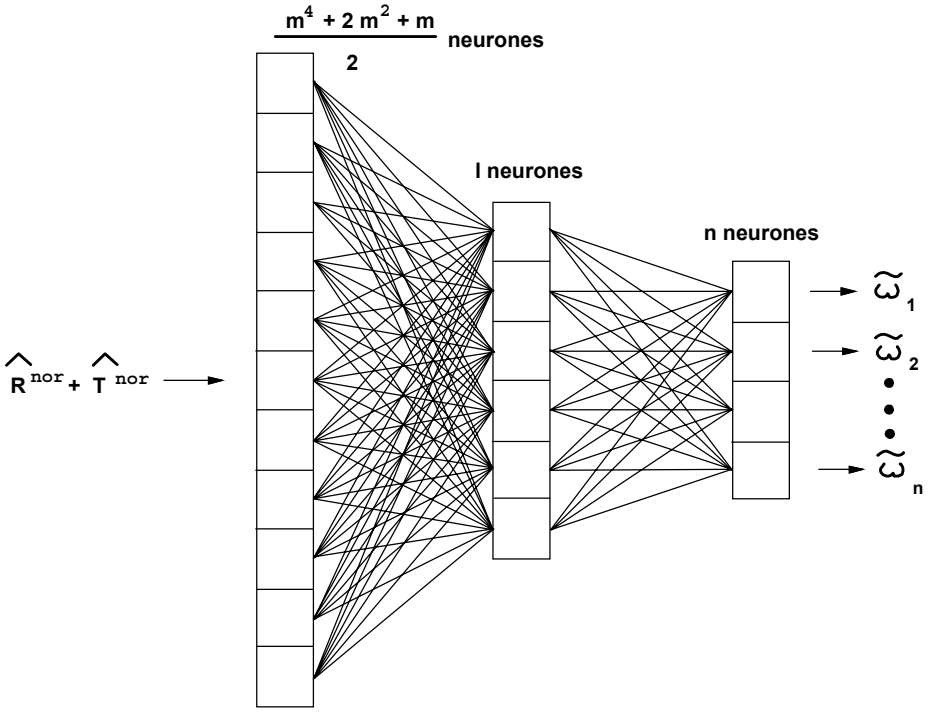

Figure 25: *Structure du PRI pour l'approche fusion AdA et SAS*

de l'ordre de n^2. Par exemple ici $l = 5$ semble convenir. La structure du Perceptron Réel d'Initialisation est illustrée en figure (25).

[9]Pour m grand, on pourra se limiter à un sous-ensemble des moments d'ordre 4

7.4 Conclusion

Dans cette partie consacrée à une approche novatrice permettant d'estimer les Angles d'Arrivée de signaux inconnus, nous avons montré que même dans un cas habituellement considéré comme "impossible"[29], à savoir lorsque le nombre de sources est égal au nombre de capteurs sur l'antenne, l'approche géométrique du Perceptron Complexe à Structure Contrainte peut être judicieusement jumelée à l'approche statistique d'un algorithme de Séparation Aveugle de Sources.

En effet, les algorithmes "classiques" d'estimation d'Angles d'Arrivée sont généralement basés sur l'utilisation d'informations d'ordre 2 (la matrice de covariance des signaux capteurs), informations qui deviennent insuffisantes lorsque le nombre de signaux devient égal au nombre de capteurs. Nous avons montré que la fusion des approches PCSC et Mesure de Dépendance, toutes deux basées sur la minimisation d'un critère d'erreur par rétropropagation du gradient, pouvait permettre l'estimation des n Angles d'Arrivée sur une antenne de n capteurs ou plus. En complétant les informations "classiques" à l'ordre 2 de l'approche PCSC (contraintes correspondant à la modélisation analytique de la géométrie de l'antenne) par des statistiques d'ordre supérieur (approche Séparation Aveugle de Sources de Burel) portant sur l'indépendance supposée des signaux originels, nous proposons une nouvelle approche qui permet, outre l'estimation des Angles d'Arrivée, l'estimation des signaux originels émanant des sources considérées.

De la même façon que pour l'estimation d'Angles d'Arrivée simple, les poids du PCSC de fusion sont initialisés par une estimation "grossière" des angles fournie par un perceptron entraîné a priori. Puisque nous travaillons maintenant dans un contexte de statistiques d'ordre supérieur, les données d'entrée du Perceptron Réel d'Initialisation comportent désormais à la fois des données provenant de la matrice de covariance normalisée $\hat{R}^{nor}$ (ordre 2), et des données provenant des tenseurs T_{ijkl} (ordre 4). Nous avons montré que pour des sources indépendantes non gaussiennes, ces données sont suffisantes pour une première estimation.

8 Résultats expérimentaux

Pour illustrer les différents algorithmes évoqués jusqu'ici, et montrer des exemples pratiques de résolution des problèmes que nous cherchons à résoudre, des résultats obtenus sur des signaux simulés sont présentés ici. Divers cas de figure relatant les différents problèmes d'Angles d'Arrivée sont étudiés. Notons au préalable que nous utiliserons dans tous les cas de figure une antenne linéaire uniforme: les capteurs sont équidistants, identiques et omnidirectionnels. De plus ils sont alignés. De cette façon, nous pourrons comparer sans peine les différentes méthodes d'estimation.

8.1 Cas d'un seul signal

Nous nous plaçons ici dans le cas le plus simple: celui de l'estimation d'un seul angle d'arrivée sur un réseau de 2 capteurs ou plus. Pour cette expérience, nous utiliserons une antenne de 5 capteurs, mais le cas plus difficile de l'estimation d'un seul angle d'arrivée par une antenne de deux capteurs seulement sera examiné au Chapitre suivant sur des données réelles.

Pour traiter ce problème de l'estimation de l'Angle d'Arrivée d'un signal sur une antenne de 5 capteurs, nous montrerons les performances comparées des algorithmes suivants: algorithmes de Haute-Résolution MUSIC et ESPRIT, méthode du Maximum de Vraisemblance, et méthode neuronale du Perceptron Complexe à Structure Contrainte initialisé par le PRI (Perceptron Réel d'Initialisation).

Le PRI utilisé est formé, comme indiqué dans la Section 4, de trois couches consécutives. Il y a $m^2 = 25$ neurones sur la couche d'entrée, $n = 1$ neurone sur la couche de sortie (estimation "grossière" $\tilde{\omega}$ du paramètre ω), et la couche cachée comporte l neurones. Une description plus détaillée de ce réseau est présentée à l'Annexe E. Pour le problème à une source et 5 capteurs, le PCSC se compose comme suit: $m = 5$ neurones sur la couche d'entrée, ainsi que sur la couche de sortie, et $n = 1$ neurone sur la couche cachée.

Il n'est pas besoin de décrire à nouveau la procédure des algorithmes de Haute-Résolution MUSIC et ESPRIT. On peut se référer à la Section 3 de ce Chapitre. Quant à la méthode du Maximum de Vraisemblance, nous avons échantillonné l'espace de l'angle d'arrivée de façon suffisamment fine pour que l'effet de l'échantillonnage sur les erreurs d'estimation soit négligeable. En l'occurence, nous savons que sur la base de test, le paramètre θ peut varier de -50^o à $+50^o$. Le paramètre ω varie donc de -2.41 à $+2.41$. Pour un pas d'échantillonage de 0.015 radians, nous devons donc générer 312 paramètres ω régulièrement espacés, afin de décrire tout l'espace de travail. Pour chaque ω, le critère du Maximum de Vraisemblance e_{MV} (voir équation(70)) est calculé, puis comparé aux valeurs obtenues avec les autres valeurs de ω. La valeur de ω associée au minimum du critère e_{MV} est sélectionnée comme la valeur optimale du MV.

Décrivons à présent la base de test que nous allons utiliser. La base de test est construite sur le même principe que la base d'entraînement du PRI, toutefois elle est contituée de données beaucoup plus bruitées. La base de test est composée de signaux simulés $\boldsymbol{y}_t$ tels que:

$$\boldsymbol{y}_t = \boldsymbol{a}(\omega)x_t + \boldsymbol{b}_t$$

où

- $\omega = \pi \sin\theta$ avec θ variant de -50^o à $+50^o$
- le rapport Signal à Bruit vaut $SNR = 5$dB, soit beaucoup moins que le SNR de la base d'apprentissage du PRI
- pour chaque expérience, le nombre d'observations $\boldsymbol{y}_t$ est $N = 10$, ce qui est également très faible pour pouvoir estimer l'Angle d'Arrivée

Nous nous plaçons donc dans un cas bruité et peu aisé à estimer, puisque d'après l'étude de Stoica [24], l'erreur sur l'estimation sur ω croît lorsque N décroît et lorsque le SNR décroît. Remarquons que nous préférons montrer un cas d'estimation assez difficile. En effet, les cas d'estimation faciles (N grand, m grand et SNR grand) ne sont pas intéressants à étudier, puisque pour de telles expériences, Stoica [24] montre que même des estimateurs sous-optimaux tels que MUSIC atteignent les bornes de Cramer-Rao. Dans de tels cas nous ne pourrions pas différencier les méthodes, ce qui ferait perdre tout son intérêt à cette étude comparative.

Nous montrons une courbe représentant la moyenne sur les erreurs. Cette moyenne est calculée sur $L = 30$ occurences du même angle, avec des tirages aléatoires différents. Ainsi, chaque point de la courbe est le reflet de 30 expériences simulées. Enfin, l'erreur sur ω s'exprime comme suit:

$$err_\omega = \sqrt{E\{(\omega - \hat{\omega})^2\}}$$

où l'espérance mathématique est estimée par

$$E\{(\omega - \hat{\omega})^2\} = \frac{1}{L}\sum_{l=1}^{L} |\omega - \hat{\omega}_l|^2$$

et l désigne l'indice de l'occurence considérée ($l \in [1, L]$). La courbe montrant les erreurs err_ω en fonction de l'angle θ est représentée figure (26). Rappelons que nous avons démontré à la Section 4, équation (110), que l'erreur sur θ et l'erreur sur ω sont liées par l'égalité approximative (pour $\hat{\theta}$ proche de θ)

$$E\{(\hat{\theta} - \theta)^2\} \simeq \frac{1}{\pi^2 \cos^2\theta} E\{(\hat{\omega} - \omega)^2\} \tag{136}$$

De ce fait, puisque tous les algorithmes d'estimation des Angles d'Arrivée estiment d'abord ω avant d'en déduire θ, il est préférable de montrer une courbe représentant l'erreur sur ω. En effet, l'équation (136) montre que pour une erreur sur ω constante, l'erreur sur θ croît lorsque θ croît. La courbe de l'erreur sur θ serait donc artificiellement relevée sur les bords (pour θ grand), ce qui serait représentatif d'un phénomène totalement indépendant des performances des estimateurs considérés. C'est pourquoi la courbe que nous présentons illustre l'erreur sur ω et non l'erreur sur θ. D'un autre côté, la courbe que nous montrons en figure (26) est fonction de θ, et non de ω, parce que le but ultime que nous recherchons est l'estimation de l'Angle

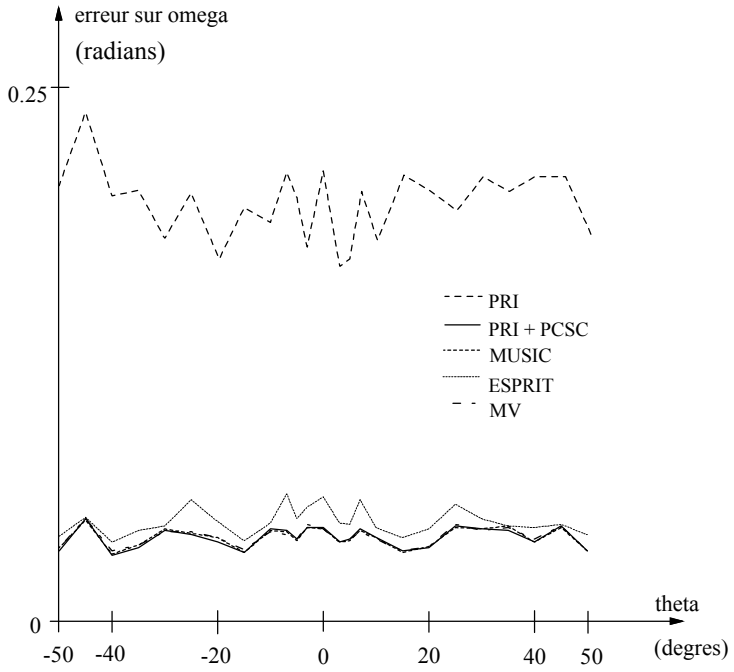

Figure 26: *Comparaison des courbes d'erreurs pour le problème d'AdA à 5 capteurs et une source*

d'Arrivée θ, et non l'estimation du paramètre intermédiaire ω. C'est pourquoi l'axe des abscisses montre l'angle θ variant de -50^o à $+50^o$.

Discussion:
La courbe des erreurs sur ω qui est présentée en figure (26) a été effectuée dans des conditions assez "dures": le SNR est faible, le nombre N d'observations par expérience est très faible, et le nombre de capteurs est également assez faible. On peut ainsi mieux observer le comportement des différents algorithmes. On voit, tout d'abord, que tous les algorithmes offrent une erreur pratiquement semblable en chaque point. Le cas de la courbe représentant les erreurs du PRI (Perceptron Réel d'Initialisation) est à mettre à part, puisque les estimations qu'il fournit ne sont pas utilisées directement: ses estimations servent uniquement de point de départ au Perceptron Complexe à Structure Contrainte. Ainsi initialisé dans le voisinage de la solution, ce dernier peut dès lors rapidement converger vers l'estimation finale.

Le fait que les courbes d'erreurs représentant les quatre estimateurs soient pratiquement identiques montre que pour le cas de l'estimation d'un seul angle, même en présence de fort bruit, le problème est suffisamment simple pour que son analyse par quatre méthodes différentes conduise aux même résultat. En d'autres termes, on pourrait dire que les conditions de l'expérience ne sont pas suffisamment

complexes pour que les algorithmes les plus performants puissent exprimer leur supériorité. Notons toutefois que l'algorithme ESPRIT se comporte légèrement moins bien que les autres. Peut-être est-ce dû au fait que celui-ci nécessite trois décompositions en valeurs propres (ce qui entraîne forcément des erreurs puisque ces décompositions sont obtenues par des algorithmes itératifs[10]). Ainsi, lorsque les erreurs des trois décompositions en valeurs propres s'ajoutent, l'erreur finale de l'algorithme ESPRIT peut devenir plus importante que l'erreur des autres algorithmes, alors que si on avait procédé[11] à des décompositions en valeurs propres parfaites, l'erreur d'ESPRIT aurait été semblable à celle des autres algorithmes. Cela dit, l'erreur commise par ESPRIT n'est que très légèrement supérieure à celle des autres algorithmes, et on ne peut la considérer comme décisive.

Dans le cas de l'estimation de plusieurs angles, nous allons voir que les différences entre les algorithmes sont plus affirmées, ce qui augmente l'intérêt de la comparaison.

8.2 Cas de deux signaux indépendants

8.2.1 Nombre de capteurs suffisant ($m > n$)

Nous utilisons pour cette Section la même antenne que précédemment ($m = 5$ capteurs), mais il y a désormais deux sources. De ce fait le PRI utilisé est différent de celui de la Section précédente: il possède maintenant $n = 2$ neurones sur sa couche de sortie. La base d'apprentissage de ce PRI est détaillée en Annexe F. Dans la première expérience que nous décrivons, les deux sources ont le même rapport signal-à-bruit: $SNR_1 = SNR_2 = 10$dB. L'un des angles est fixé à zéro degré, et l'autre varie de -50^o à $+50^o$. Nous disposons de $N = 15$ observations par expérience.

Le PCSC utilisé pour cette Section comporte $m = 4$ neurones sur ses couches d'entrée et de sortie, et $n = 2$ neurones sur sa couche cachée. Les expériences sont réalisées avec l'approche neuronale PRI + PCSC, mais aussi avec les méthodes MUSIC, ESPRIT et MV. Pour la méthode du MV, on effectue un échantillonnage en 2D sur tous les couples d'angles possibles. Pour les expériences, on a dit que les paramètres θ_i peuvent varier de -50^o à $+50^o$, soit ω_i variant de -2.41 à $+2.41$ radians. Pour un pas d'échantillonnage de 0.015 radians, cela fait 312 échantillons par angle, soit un total de $312^2 = 97344$ couples d'angles. Pour qu'un couple (α, β) ne soit pas retesté une deuxième fois sous la forme (β, α), on peut réduire la base des échantillons en utilisant la contrainte $\omega_1^{ech} < \omega_1^{ech}$ sur chaque couple. Le nombre

[10] donc non exacts

[11] dans l'idéal

total de couples est alors réduit à $\frac{312 \times 313}{2} = 48828$, ce qui divise environ par deux le nombre total de couples à tester. On voit que ce nombre est tout de même important, ce qui conforte l'idée que la méthode du Maximum de Vraisemblance est longue et coûteuse.

Dans une première expérience, les signaux sont gaussiens centrés de puissance 1, le bruit est gaussien blanc, et on a:

- $\omega_i = \pi \sin \theta_i$ avec $\theta_1 = 0^o$ et θ_2 variant de -50^o à $+50^o$.
- $SNR_1 = SNR_2 = 10\text{dB}$,
- le nombre d'observations est $N = 15$ pour chaque expérience.

Nous montrons une courbe représentant la moyenne sur les erreurs commises sur les *deux* angles:

$$\begin{aligned} err_\omega &= \frac{1}{2}\left(err_{\omega_1} + err_{\omega_2}\right) \\ &= \frac{1}{2}\left(\sqrt{E\{(\hat{\omega}_1 - \omega_1)^2\}} + \sqrt{E\{(\hat{\omega}_2 - \omega_2)^2\}}\right) \end{aligned} \tag{137}$$

où l'espérance est estimée par:

$$E\{(\hat{\omega}_i - \omega_i)^2\} = \frac{1}{L}\sum_{l=1}^{L} |\omega_i - \hat{\omega}_i|^2$$

Cette estimation est calculée sur $L = 30$ occurences du même couple d'angles, avec des tirages aléatoires différents: chaque point de la courbe est le reflet de 30 expériences simulées. La courbe montrant err_ω en fonction de θ_2 est représentée en figure (27). Les points caractéristiques de cette courbe sont les suivants:

- pour un écart entre θ_1 et θ_2 grand, toutes les méthodes (MUSIC, ESPRIT, PRI + PCSC, MV) réagissent bien et ne présentent qu'une faible erreur: leurs performances sont d'ailleurs comparables.
- au fur et à mesure que θ_2 se rapproche de θ_1, l'erreur sur l'estimation croît plus vite pour l'estimateur MUSIC, qui "décroche" dès que θ_2 et θ_1 sont séparés de moins de 12 degrés (pour un écart inférieur, les deux "pics" de la fonction MUSIC se confondent et l'algorithme ne distingue plus qu'*un seul* Angle d'Arrivée).
- lorsque θ_2 se rapproche un peu plus de θ_1, c'est ESPRIT qui à son tour "décroche": pour un écart $|\theta_2 - \theta_1|$ inférieur à 5^o, ESPRIT fournit des résultats totalement erronés.

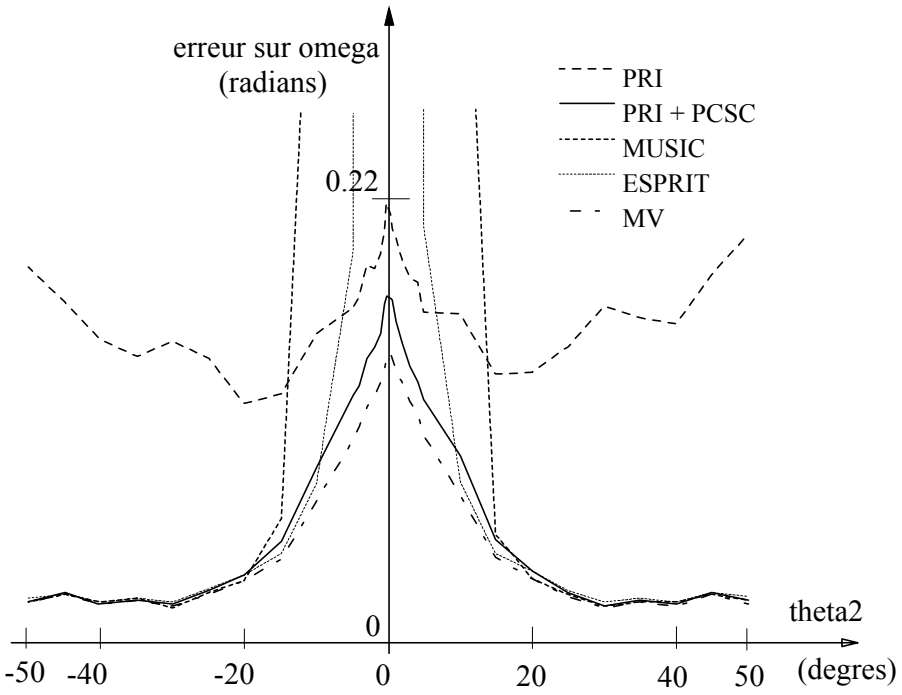

Figure 27: *Comparaison des courbes d'erreurs pour le problème d'AdA à 5 capteurs et deux sources: ici les SNR des deux sources sont égaux.*

- on constate que ni notre méthode neuronale, ni la méthode du MV ne "décrochent". Par contre, l'erreur croît de plus en plus au fur et à mesure que θ_2 se rapproche de θ_1. On voit que les performances du PCSC s'éloignent peu de celles du MV, ce qui semble normal puisque ces deux méthodes minimisent le même critère. En augmentant le nombre d'itérations du PCSC, on fait se rapprocher les deux courbes de plus en plus.

Une seconde expérience a été conduite avec les mêmes valeurs de paramètres que la première, sauf au niveau des puissances des sources et des rapports signal-à-bruit: la source 1 a toujours une puissance de 1, mais la source 2 a à présent une puissance de 4. Le bruit a également été augmenté de 5dB, et on a donc à présent $SNR_1 = 5$dB et $SNR_2 = 11$dB. Les conditions expérimentales sont donc nettement moins bonnes. En outre, cette expérience permet de voir si le PRI, entraîné sur une base où les puissances moyennes des deux sources sont égales, peut quand même fournir des résultats suffisamment proches de la solution pour donner une bonne initialisation aux poids du PCSC lorsque les puissances des sources diffèrent fortement. La courbe relatant cette expérience est présentée en figure (28). La première remarque que l'on peut faire au sujet de cette courbe, est que si effectivement l'erreur brute commise par le PRI semble plus importante en moyenne, le PCSC est toujours initialisé dans un voisinage suffisamment proche de la solution pour que ses performances n'en

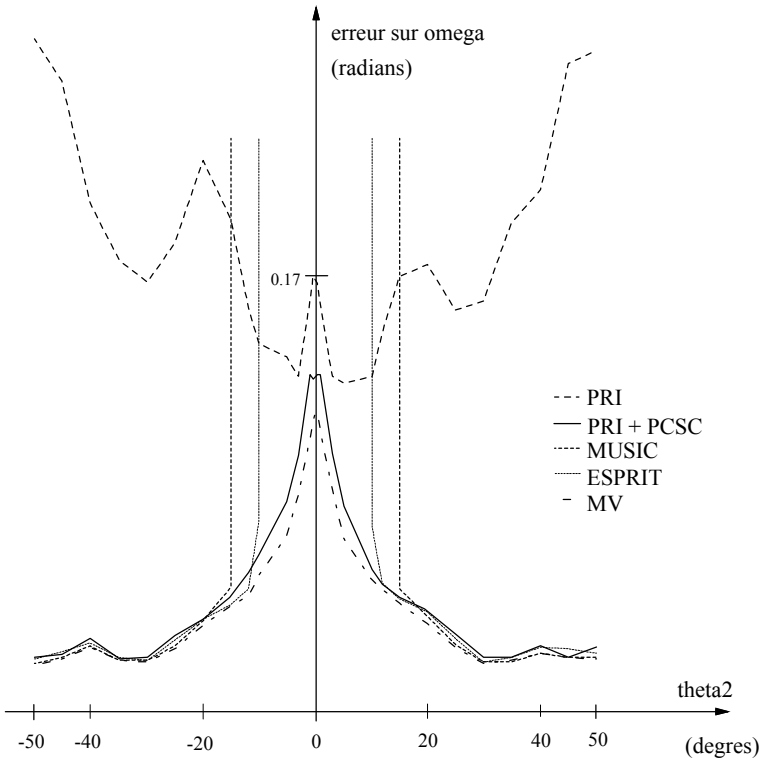

Figure 28: *Comparaison des courbes d'erreurs pour le problème d'AdA à 5 capteurs et deux sources: ici le SNR de la source 2 est double de celui de la source 1.*

soient pas affectées.

Comme dans l'expérience précédente, on constate que:

- pour des angles θ_1 et θ_2 éloignés, les quatre méthodes sont pratiquement équivalentes.
- pour θ_2 se rapprochant de θ_1, les méthodes Haute-Résolution "décrochent" l'une après l'autre: MUSIC décroche pour $|\theta_2 - \theta_1|$ inférieur à 15^o, et ESPRIT décroche pour $|\theta_2 - \theta_1|$ inférieur à 10 degrés.
- la courbe PRI + PCSC est très proche de la courbe du MV, bien que cette méthode soit nettement moins lourde que le MV.

8.2.2 Nombre de capteurs insuffisant ($m = n$)

Dans cette nouvelle expérience, on dispose d'une antenne de $m = 2$ capteurs pour estimer les Angles d'Arrivée de $n = 2$ sources. De ce fait, les approches au second ordre type MUSIC, ESPRIT ou MV sont inefficaces.

Seule une approche aux ordres supérieurs peut espérer fonctionner sur ce type

de problème. Nous testons ici le PCSC de fusion (*cf* Section 7) initialisé par un PRI utilisant des données aux ordres 2 et 4.
La structure du PRI utilisé est la suivante: il y a $m^2 = 4$ entrées pour représenter la matrice de covariance normalisée $\hat{R}^{nor}$, et $\frac{m^4+m}{2} = 9$ entrées pour représenter les tenseurs T_{ijkl} normalisés (voir Section (7.3)), soit 13 neurones sur la couche d'entrée. Il y a $l = 5$ neurones sur la couche cachée et $n = 2$ neurones sur la couche de sortie. Les sources utilisées pour la base d'entraînement du PRI sont des signaux obtenus par tirage aléatoire uniforme entre -1 et 1 des parties réelles et imaginaires (ces signaux ne sont donc pas gaussiens). Le bruit sur la base est gaussien blanc, de variance telle que le rapport signal-à-bruit moyen vaut $SNR = 30$dB. Les Angles d'Arrivée varient de -60^o à $+60^o$ par pas de 1.5^o, avec $\theta_1 < \theta_2$ sur la base: il y a comme en Section précédente 3240 couples d'angles, et $L = 2$ occurences de chaque couple sont générées pour des tirages aléatoires différents, soit 6480 exemples au total. Le nombre d'observations par expérience est plus grand que pour les conjonctures où l'on avait $m > n$: ici $N = 300$ observations par expérience sur la base d'apprentissage.

Le PCSC utilisé cherche à minimiser à la fois un critère géométrique (contrainte sur la forme de l'antenne) et un critère statistique (indépendance des sources). Il est composé de $m = 2$ neurones sur la couche d'entrée, $n = 2$ neurones sur la couche cachée et $m = 2$ neurones sur la couche de sortie. C'est donc un petit réseau (6 neurones et 8 connexions en tout).

Les expériences sont menées sur des signaux simulés: ce sont des signaux de type aléatoire obtenus par tirage uniforme entre -1 et 1. Du bruit blanc gaussien est ajouté, de telle sorte que le rapport signal-à-bruit vaut $SNR = 20$dB. L'angle θ_1 est fixé à zéro tandis que θ_2 varie de -50^o à $+50^o$. On trace l'erreur err_ω définie à l'équation (137) en fonction de θ_2 sur la figure (29). On constate que pour un écart entre θ_2 et θ_1 supérieur à trois degrés, notre estimateur neuronal fonctionne bien, et les erreurs d'estimation sur les Angles d'Arrivée sont assez faibles. Dans le même temps, rappelons que les autres méthodes sont totalement inefficaces. Par contre, si l'écart entre θ_2 et θ_1 devient inférieur à trois degrés, l'erreur sur l'estimation croît.
Comme on l'a dit, les conditions expérimentales sont assez difficiles, étant donné que l'on n'utilise que deux capteurs pour estimer l'Angle d'Arrivée de deux sources. De plus, nous ne disposons que de $N = 200$ observations par expérience, ce qui est assez faible pour séparer correctement des signaux indépendants. Cela montre que même dans des conditions difficiles, la méthode reste fiable et robuste.

8.3 Cas de deux signaux corrélés

Nous présentons ici des expériences effectuées sur une antenne de $m = 5$ capteurs: on reçoit deux signaux qui peuvent être partiellement corrélés. En fait, on reprend exactement la même expérience que celle présentée en figure (27) où les sources

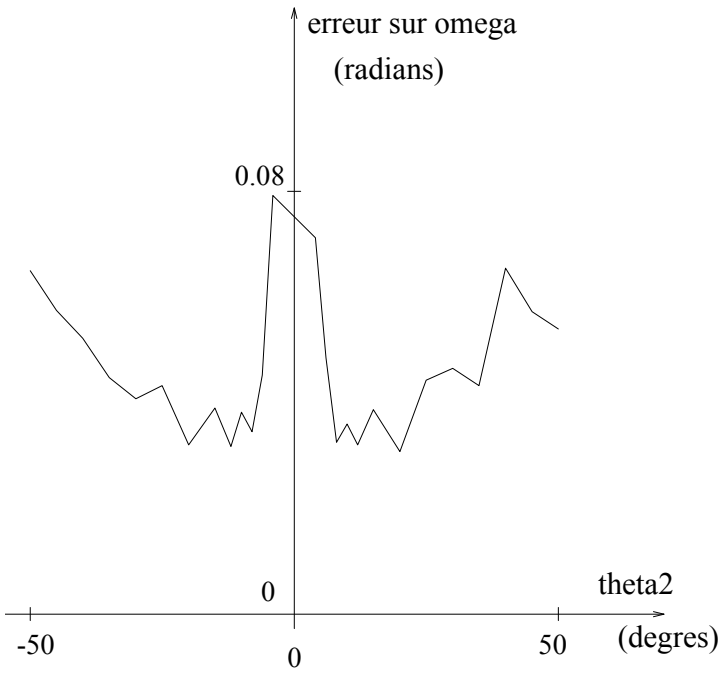

Figure 29: *Courbe d'erreurs pour le problème d'AdA à deux capteurs et deux sources: ici seule notre approche de fusion est efficace.*

étaient indépendantes, et on augmente petit à petit la corrélation entre les sources. La matrice de corrélation des sources est:

$$P_x = E\{\boldsymbol{x}\boldsymbol{x}^H\} = \begin{pmatrix} E\{|x_1^2|\} & E\{x_1 x_2^*\} \\ E\{x_1^* x_2\} & E\{|x_2^2|\} \end{pmatrix}$$

Pour l'expérience de la figure (27), les sources sont indépendantes donc cette matrice est diagonale:

$$P_x = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

On dit aussi que le facteur de corrélation ρ est nul.
Nous présentons ici une deuxième expérience (figure (30)) où les signaux sont fortement corrélés ($\rho = 0.9$):

$$P_x = \begin{pmatrix} 1 & 0.9 \\ 0.9 & 1 \end{pmatrix}$$

et une expérience (figure (31)) où les signaux sont très fortement corrélés:

$$P_x = \begin{pmatrix} 1 & 0.99 \\ 0.99 & 1 \end{pmatrix}$$

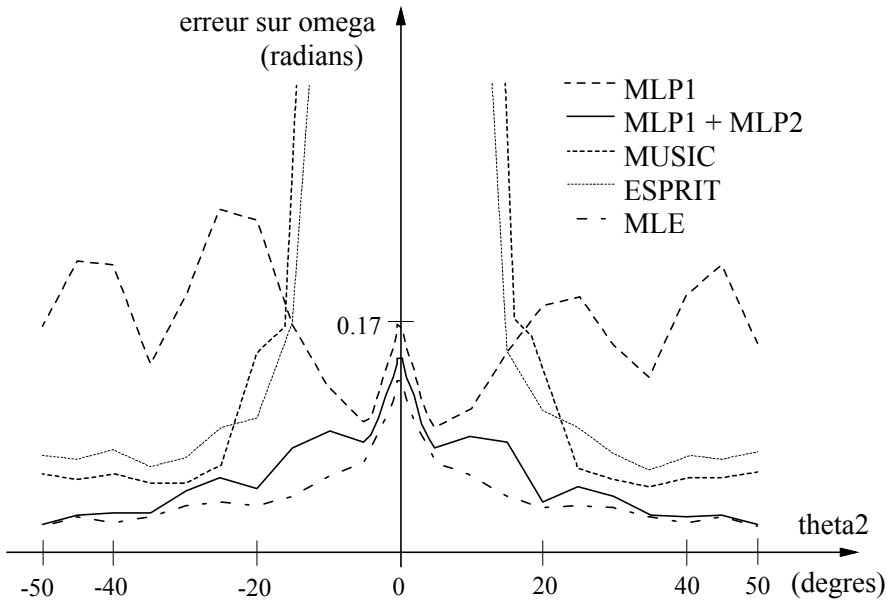

Figure 30: *Courbes d'erreurs pour le problème d'AdA à 5 capteurs et deux sources: ici le facteur de corrélation est* $\rho = 0.9$.

On constate déjà que pour un facteur de corrélation $\rho = 0.9$, les méthodes de Haute-Résolution MUSIC et ESPRIT rencontrent des difficultés: d'une part leurs erreurs sont visiblement plus importantes que la méthode MV pour des angles éloignés, mais en outre elles "décrochent" plus vite que lorsque les sources n'étaient pas corrélées: MUSIC décroche pour $|\theta_2 - \theta_1|$ inférieur à 16^o au lieu de 12^o, et ESPRIT décroche à 15^o au lieu de 5^o.

Pour un facteur de corrélation de $\rho = 0.99$, ni ESPRIT ni MUSIC ne fonctionnent. On explique en Annexe G la raison pour laquelle MUSIC ne peut pas fonctionner lorsque les sources sont entièrement corrélées ($\rho = 1$). Cela peut expliquer pourquoi à $\rho = 0.99$ on n'obtient pas de résultats non plus. L'algorithme ESPRIT est plus difficile à manipuler, mais les raisons qui font que MUSIC est inefficace sont sans doute les mêmes pour ESPRIT.

On pouvait d'ailleurs prévoir que MUSIC ne fonctionnerait pas pour des ρ très proches de 1, car dans [24], Stoica indique que la variance de l'erreur sur ω vaut

$$var_{MU}(\hat{\omega}_i) = \frac{6\sigma^2}{Nm^3}\left(P_x^{-1}\right)_{ii}$$

pour MUSIC. Ainsi, si

$$P_x = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$$

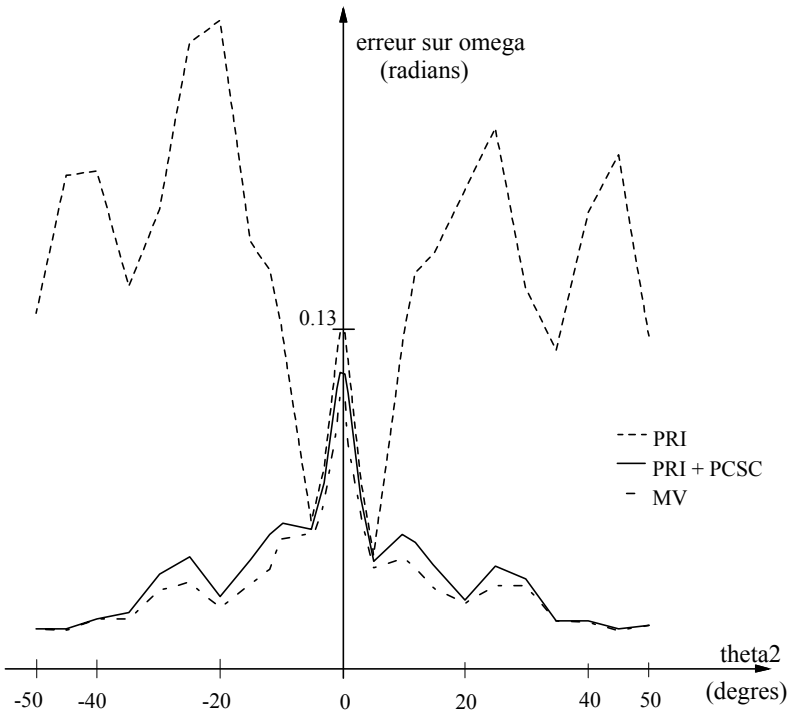

Figure 31: *Courbes d'erreurs pour le problème d'AdA à 5 capteurs et deux sources: ici le facteur de corrélation est* $\rho = 0.99$.

avec ρ proche de 1, la matrice P_x est presque singulière, donc $(P_x^{-1})_{ii}$ tend vers l'infini, ce qui explique la faillite de MUSIC.
Par ailleurs, Stoica montre que la variance du Maximum de Vraisemblance s'exprime comme

$$var_{MV}(\hat{\omega}_i) = \left(1 + \frac{1}{mSNR}\right) \frac{6\sigma^2}{Nm^3} \frac{1}{P_{ii}}$$

On voit donc que pour le MV, le fait que P_{ii} soit proche de 1 (ou même égal à 1) altère beaucoup moins les performances de la méthode. C'est pourquoi ici, pour $\rho = 0.99$, on constate que les performances du Maximum de Vraisemblance et de notre méthode neuronale basée sur le même critère, sont toujours relativement bonnes.

9 Conclusion

Dans de chapitre, nous nous sommes intéressés au problème général de l'estimation d'Angles d'Arrivée. Deux cas majeurs sont envisagés: lorsque le nombre de capteurs est suffisant par rapport au nombre de sources, et lorsque le nombre de capteurs est insuffisant.

Lorsque le nombre de capteurs est "suffisant", c'est-à-dire lorsqu'il est strictement supérieur au nombre de sources, des approches "classiques" type Haute-Résolution, comme MUSIC et ESPRIT, sont envisageables. Elles sont cependant sous-optimales. Des approches neuronales ont également été envisagées par le passé, mais à partir de bases théoriques insuffisantes pour pouvoir concurrencer les algorithmes de Haute-Résolution. La meilleure méthode reste celle du Maximum de Vraisemblance, même si elle est peu employée du fait de son coût prohibitif en temps de calcul. Nous proposons une approche neuronale très robuste, qui optimise le même critère que le Maximum de Vraisemblance, mais dans un temps bien moindre puisque la recherche de la solution se fait par descente de gradient. Les minima locaux sont évités grâce à une initialisation originale des paramètres par un premier perceptron. Des expériences comparant les performances des diverses solutions citées montrent que notre approche fournit des résultats très comparables à ceux du Maximum de Vraisemblance, tout en étant beaucoup plus rapide. De plus, lorsque les angles à estimer sont proches, ou que les sources sont très corrélées, notre approche reste viable alors que des méthodes comme MUSIC et ESPRIT deviennent inefficaces.

Lorsque le nombre de capteurs est égal au nombre de sources, les approches au second ordre telles que MUSIC, ESPRIT ou le Maximum de Vraisemblance ne fonctionnent plus. En étendant notre approche à la minimisation d'un critère mixte jouant à la fois sur l'information géométrique (forme de l'antenne) et sur l'information statistique (indépendance des sources), il devient possible d'estimer les Angles d'Arrivée de n sources sur une antenne de n capteurs. Cette approche neuronale dite de "fusion" entre le problème d'Angles d'Arrivée et le problème de Séparation Aveugle de Sources, a été expérimenté avec succès sur des signaux à distribution uniforme. Même pour des angles très proches, cette solution montre de très bonnes performances.

Annexe A: Nombre d'opérations requis pour chaque méthode d'estimation d'Angles d'Arrivée

Nous avons calculé le nombre de multiplications complexes requis par les différentes méthodes d'estimation d'Angles d'Arrivée citées dans ce document.

On notera:

- N le nombre d'observations
- m le nombre de capteurs
- n le nombre de sources
- I le nombre moyen d'itérations de la méthode de diagonalisation de Jacobi[12]
- I' le nombre d'itérations du Perceptron Complexe à Structure Contrainte
- q le nombre de niveaux de quantification sur $\omega = \pi \sin\theta$. Ce nombre est utilisé dans l'algorithme MUSIC lors de la recherche des pics de la fonction, et dans la méthode du Maximum de Vraisemblance pour le test de tous les $n-$uplets d'angles possibles.

On obtient les résultats suivants:

MUSIC	$Nm^2 + 6Im^2(m-1) + q(m-n)m$	19 875
ESPRIT	$12I\{2(m-1)^2(2m-3) + n^2(5n-3)\}$ $+4(m-1)\{N(m-1)+n^2\} + 2n^3$	61 520
PRI + PCSC	$Nm^2 + \frac{1}{4}m^3 + 7I'Nnm$	16 156
MV	$q^n N\{2n^2m + nm^2 + n^3 + 6In^2(n-1)\}$	2.2×10^9

Pour l'exemple numérique indiqué dans le tableau, nous avons pris $N = 15$, $m = 5$, $n = 2$, $I = 20$, $I' = 15$ et $q = 500$ (ce qui correspond approximativement à un échantillonnage par pas de 0.2^o pour θ variant de -50^o à $+50^o$).

Pour information, le PRI seul demande $Nm^2 + \frac{1}{4}m^3$ multiplications complexes, soit numériquement 406 multiplications complexes pour l'exemple évoqué ci-dessus.

[12] La diagonalisation d'une matrice complexe hermitienne $p \times p$ nécessite $6Ip^2(p-1)$ multiplications complexes. Lorsqu'une méthode nécessite une inversion de matrice, cette inversion est également réalisée en passant par une diagonalisation et une inversion des valeurs propres.

Annexe B: Moments d'une gaussienne

Pour estimer la valeur de $J_{2k}(\sigma)$, nous nous intéresserons tout d'abord à la valeur de

$$I_{2k}(\sigma) = \frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{+\infty} x^{2k} e^{-\frac{x^2}{2\sigma^2}} dx$$

que l'on peut réécrire comme

$$I_{2k}(\sigma) = \frac{-\sigma^2}{\sqrt{2\pi}\sigma} \int_{-\infty}^{+\infty} x^{2k-1} \left(-\frac{x}{\sigma^2}\right) e^{-\frac{x^2}{2\sigma^2}} dx$$

En intégrant par parties, on obtient :

$$I_{2n}(\sigma) = \frac{-\sigma^2}{\sqrt{2\pi}\sigma} \left\{ \left[x^{2k-1} e^{-\frac{x^2}{2\sigma^2}} \right]_{-\infty}^{+\infty} - \int_{-\infty}^{+\infty} (2k-1) x^{2k-2} e^{-\frac{x^2}{2\sigma^2}} dx \right\}$$

Soit :

$$I_{2k} = \sigma^2 (2k-1) I_{2k-2}$$

Comme $I_0(\sigma) = 1$, on en déduit :

$$\begin{aligned} I_{2k} &= \sigma^{2k}(2k-1)(2k-3)...1 \\ &= \sigma^{2k} \frac{(2k)!}{2^k k!} \end{aligned}$$

On vérifiera aisément que $I_{2k+1}(\sigma)$ est nul, car on intègre alors une fonction impaire.

Pour le problème qui nous intéresse, nous avons à calculer :

$$\begin{aligned} J_{2k}(\sigma) &= \int_{-\infty}^{+\infty} x^{2k} e^{-\sigma^2 x^2} dx \\ &= \frac{\sqrt{\pi}}{\sigma} I_{2k}\left(\frac{1}{\sqrt{2}\sigma}\right) \end{aligned}$$

Et on a donc :

$$\boxed{J_{2k}(\sigma) = \frac{(2k)!}{4^k k!} \frac{\sqrt{\pi}}{\sigma^{2k+1}}}$$

Annexe C: Comportement à l'infini des coefficients de la mesure de dépendance

Nous cherchons ici à montrer que le coefficient générique $G_{\alpha_1 \ldots \alpha_n, \beta_1 \ldots \beta_n}$ intervenant dans l'expression de la mesure de dépendance, tend vers zéro lorsque l'un au moins des indices α_i tend vers l'infini.
D'après [2], on a

$$G_{\alpha_1 \ldots \alpha_n, \beta_1 \ldots \beta_n} = \prod_{i=1}^{n} g(\alpha_i, \beta_i, \sigma_i)$$

avec

$$\begin{aligned} g(\alpha, \beta, \sigma) &= \frac{1}{\alpha! \beta!} (-1)^{\frac{\alpha-\beta}{2}} J_{\alpha+\beta}(\sigma) && \text{si } \alpha+\beta \text{ est pair} \\ &= 0 && \text{sinon} \end{aligned}$$

et $J_{2k}(\sigma)$ représente le moment d'ordre $2k$ d'une gaussienne:

$$J_{2k}(\sigma) = \frac{(2k)!}{4^k k!} \frac{\sqrt{\pi}}{\sigma^{2k+1}}$$

On a alors

$$J_{\alpha+\beta} = \frac{(\alpha+\beta)!}{2^{\alpha+\beta} \left(\frac{\alpha+\beta}{2}\right)!} \frac{\sqrt{\pi}}{\sigma^{\alpha+\beta+1}}$$

d'où l'expression de g développée pour $\alpha+\beta$ pair:

$$g(\alpha, \beta, \sigma) = \frac{(-1)^{\frac{\alpha-\beta}{2}} \sqrt{\pi}\ (\alpha+\beta)!}{\alpha! \beta!\ 2^{\alpha+\beta} \left(\frac{\alpha+\beta}{2}\right)!\ \sigma^{\alpha+\beta+1}}$$

On rappelle que l'on a l'approximation suivante pour la factorielle :

$$\alpha! \simeq \sqrt{2\pi\alpha}\ \alpha^{\alpha} e^{-\alpha} \left(1 + \frac{1}{12\alpha}\right)$$

Il est intéressant de noter que cette approximation est assez fine, même pour les faibles valeurs de α. En effet, pour $\alpha = 1$ par exemple, cette approximation n'engendre qu'une erreur relative de 10^{-3}. En remplaçant les factorielles par leurs approximations dans l'expression de g, on obtient après simplifications

$$g(\alpha, \beta, \sigma) \simeq \frac{(-1)^{\frac{\alpha-\beta}{2}} (\alpha+\beta)^{\frac{\alpha+\beta}{2}}}{\alpha^{\alpha+\frac{1}{2}} \beta^{\beta+\frac{1}{2}} \sigma^{\alpha+\beta+1} \sqrt{2\pi}} \left(\frac{e}{2}\right)^{\frac{\alpha+\beta}{2}} \frac{1 + \frac{1}{12(\alpha+\beta)}}{\left(1+\frac{1}{12\alpha}\right)\left(1+\frac{1}{12\beta}\right)\left(1+\frac{1}{6(\alpha+\beta)}\right)}$$

Lorsque α ou β tend vers l'infini, cette fraction se comporte en valeur absolue comme

$$\frac{(\alpha+\beta)^{\frac{\alpha+\beta}{2}}}{\alpha^{\alpha+\frac{1}{2}}\beta^{\beta+\frac{1}{2}}}$$

Ainsi, pour β fixé et α tendant vers l'infini, cette dernière expression se comporte comme $\frac{\alpha^{\frac{\alpha}{2}}}{\alpha^{\alpha}}$, c'est à dire comme $\alpha^{-\frac{\alpha}{2}}$. Puisque α et β ont un rôle identique dans l'expression, on en conclut que $g(\alpha,\beta,\sigma)$ tend vers l'infini dès que α ou β tend vers l'infini. Enfin, comme le coefficient $G_{\alpha_1\ldots\alpha_n,\beta_1\ldots\beta_n}$ utilisé dans la mesure de dépendance est un produit en i de $g(\alpha_i,\beta_i,\sigma_i)$, il apparaît finalement qu'il suffit que l'un au moins des α_i ou β_i devienne important pour que $G_{\alpha_1\ldots\alpha_n,\beta_1\ldots\beta_n}$ tende vers zéro.

Annexe D: MUSIC, ESPRIT, MV et PCSC utilisent de l'information au second ordre seulement

MUSIC, ESPRIT

Pour les algorithmes de Haute-Résolution MUSIC et ESPRIT, il est évident que la seule information utilisée est au second ordre, puisque ces deux méthodes se basent uniquement sur des décompositions en valeurs propres de la matrice de covariance des signaux capteurs

$$\hat{R}_y = \frac{1}{N}\sum_{t=1} N \boldsymbol{y}_t \boldsymbol{y}_t^H$$

MV, PCSC

Le Maximum de Vraisemblance et le Perceptron Complexe à Structure Contrainte minimisent le même critère, donc il suffit de prouver l'assertion pour un seul des deux algorithmes. Le critère du Maximum de Vraisemblance s'exprime par

$$e_{MV} = \frac{1}{N}\sum_{t=1}^{N} ||\boldsymbol{y_t} - P_{\hat{A}}(\boldsymbol{y_t})||^2$$

d'après l'équation (70), où $P_{\hat{A}} = \hat{A}(\hat{A}^H\hat{A})^{-1}\hat{A}^H$ est la matrice définissant la projection orthogonale sur $Vect\{\hat{A}\}$, et $\hat{A}$ est l'estimation de A. En développant, on obtient:

$$\begin{aligned} e_{MV} &= \frac{1}{N}\sum_{t=1}^{N} (\boldsymbol{y_t} - P_{\hat{A}}\boldsymbol{y_t})^H (\boldsymbol{y_t} - P_{\hat{A}}\boldsymbol{y_t}) \\ &= \frac{1}{N}\sum_{t=1}^{N} \left(\boldsymbol{y_t}^H - \boldsymbol{y_t}^H P_{\hat{A}}^H\right) (\boldsymbol{y_t} - P_{\hat{A}}\boldsymbol{y_t}) \\ &= \frac{1}{N}\sum_{t=1}^{N} \left(\boldsymbol{y_t}^H\boldsymbol{y_t} - \boldsymbol{y_t}^H P_{\hat{A}}\boldsymbol{y_t} - \boldsymbol{y_t}^H P_{\hat{A}}^H\boldsymbol{y_t} + \boldsymbol{y_t}^H P_{\hat{A}}^H P_{\hat{A}}\boldsymbol{y_t}\right) \end{aligned}$$

Or on a déjà vu dans la partie consacrée à la méthode de Rastogi que l'une des propriétés de la projection orthogonale est d'être égale à sa transposée hermitienne:

$$P_{\hat{A}}^H = \left[\hat{A}(\hat{A}^H\hat{A})^{-1}\hat{A}^H\right]^H = \hat{A}(\hat{A}^H\hat{A})^{-1}\hat{A}^H = P_{\hat{A}}$$

et d'autre part la projection est un morphisme idempotent, donc nous avons aussi

$$P_{\hat{A}}^H P_{\hat{A}} = P_{\hat{A}} P_{\hat{A}} = P_{\hat{A}}$$

ce qui nous permet de simplifier l'écriture du critère de Vraisemblance:

$$\begin{aligned} e_{MV} &= \frac{1}{N}\sum_{t=1}^{N}\left(\boldsymbol{y_t}^H\boldsymbol{y_t} - \boldsymbol{y_t}^H P_{\hat{A}}\boldsymbol{y_t}\right) \\ &= -\frac{1}{N}\sum_{t=1}^{N}\mathrm{tr}\left\{P_{\hat{A}}\boldsymbol{y_t}\boldsymbol{y_t}^H - \boldsymbol{y_t}\boldsymbol{y_t}^H\right\} \\ &= -\mathrm{tr}\left\{(P_{\hat{A}} - I)\left(\frac{1}{N}\sum_{t=1}^{N}\boldsymbol{y_t}\boldsymbol{y_t}^H\right)\right\} \\ &= -\mathrm{tr}\left\{(P_{\hat{A}} - I)\hat{R}_y\right\} \end{aligned}$$

où le signe "tr" désigne la trace d'une matrice carrée. Cette décomposition prouve que le Maximum de Vraisemblance et le Perceptron Complexe à Structure Contrainte n'utilisent eux-aussi que de l'information au second ordre, c'est à dire l'information contenue dans la matrice de covariance.

Annexe E: Structure détaillée du PRI pour le cas $n = 1$ source et $m = 5$ capteurs

La taille du Perceptron Réel d'Initialisation pour le cas $n = 1$ source et $m = 5$ capteurs est la suivante: trois couches, dont $m^2 = 25$ neurones sur la couche d'entrée, l neurones sur la couche cachée, et $n = 1$ neurone sur la couche de sortie.

Dans la Section 4, nous avions conseillé de choisir l de l'ordre de m. Nous choisissons $l = 5$ neurones sur la couche cachée. Le Perceptron Réel d'Initialisation (PRI) associé à ce problème est représenté en figure (32). Ce PRI est entraîné *avant*

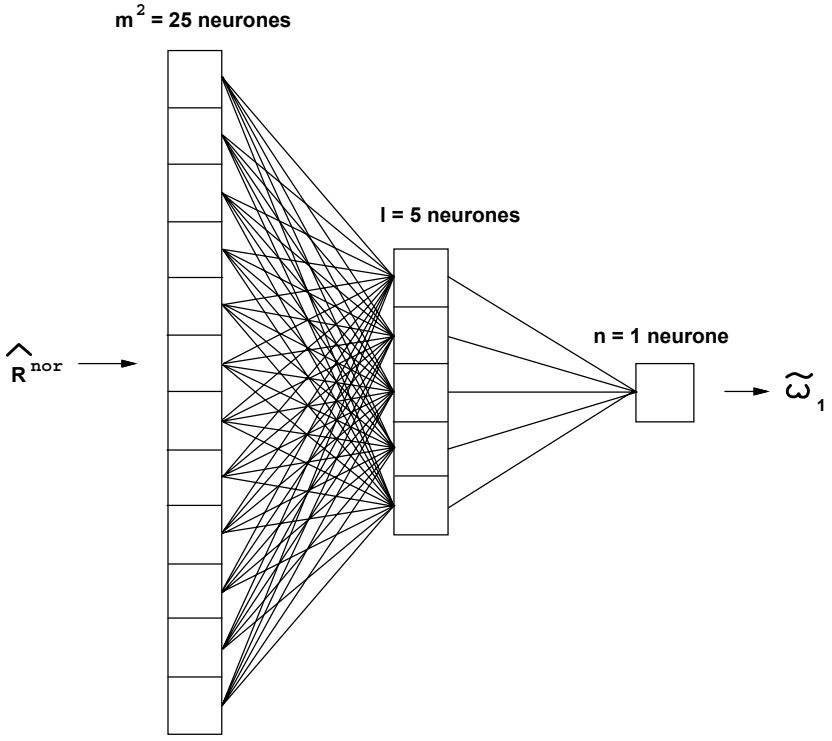

Figure 32: *Le Perceptron Réel d'Initialisation à 25 entrées, 5 neurones cachés et une sortie pour le problème d'AdA à 5 capteurs et une source*

la phase de test sur une base formée de signaux représentatifs du problème. La base d'entraînement est formée de signaux capteurs $\boldsymbol{y}_t$ tels que:

$$\boldsymbol{y}_t = \boldsymbol{a}(\omega)x_t + \boldsymbol{b}_t$$

où

- x_t est une variable aléatoire complexe: x_t est gaussienne de moyenne zéro et de variance 1

- ω est le paramètre recherché pour pouvoir déterminer l'Angle d'Arrivée θ. En effet, dès que l'on a estimé ω, il suffit d'inverser l'équation

$$\omega = \frac{2\pi d}{\lambda} \sin\theta$$

(avec $d \leq \lambda/2$) pour connaître θ

- $\boldsymbol{a}(\omega)$ est le vecteur directionnel associé au paramètre ω. Pour une antenne linéaire uniforme de 5 capteurs, on a:

$$\boldsymbol{a}(\omega) = [1, e^{-j\omega}, e^{-2j\omega}, e^{-3j\omega}, e^{-4j\omega}]^T \tag{138}$$

- $\boldsymbol{b}_t$ est un vecteur complexe 5×1 de bruit gaussien blanc indépendant des sources: chacune de ses composantes a la même variance σ^2

- la variance du bruit σ^2 est calculée grâce au Signal Rapport à Bruit (SNR en anglais) que l'on se donne:

$$SNR(\text{dB}) = 10 \log \frac{1}{\sigma^2}$$

Pour la base d'entraînement nous choisissons un rapport signal à bruit assez bon, de manière à ce que le Perceptron apprenne facilement. Le SNR n'est pas non plus trop fort de façon à ce que le PRI puisse extrapoler ses résultats s'il était confronté à une base de test assez bruitée: le SNR de la base d'entraînement est choisi égal à 25dB.

Dans la base d'apprentissage, l'angle d'arrivée θ varie de -60^o à $+60^o$ par pas de 1^o, et comme nous prenons $d = \lambda/2$, nous avons

$$\omega = \pi \sin\theta$$

d'où ω qui varie de -2.72 à $+2.72$ sur la base d'entraînement. Pour chaque ω de la base d'entraînement, nous générons aléatoirement N occurences du signal $\boldsymbol{y}_t$. Pour le cas à 5 capteurs, nous choisissons de prendre $N = 40$ observations par expérience, sur la base d'apprentissage. C'est avec ces N observations (bruitées) que nous calculons les composantes de la matrice de covariance normalisée $\hat{R}^{nor}$ définie à l'équation (109), et qui seront les entrées du Perceptron. La sortie désirée du Perceptron est la variable réelle ω ayant servi à générer ces signaux. Le bruit $\boldsymbol{b}_t$ étant non nul et le nombre d'observations étant fini, à un seul paramètre ω peuvent correspondre plusieurs matrices de covariance normalisée $\hat{R}^{nor}$. Dans un souci de généralisation, la base d'apprentissage comporte donc plusieurs occurences du même angle d'arrivée, mais associé à des groupes de signaux aléatoires $\boldsymbol{y}_t$, $t \in [1, N]$ différents, et donc à des matrices $\hat{R}^{nor}$ différentes. Ici, pour ne pas trop alourdir la base d'apprentissage, nous prendrons $L = 10$ occurences différentes par angle d'arrivée. Au total nous avons donc $10 \times (2 \times 60 + 1) = 1210$ exemples sur la base d'apprentissage.

Après apprentissage sur la base que nous venons de décrire, les poids du PRI sont gelés, et ne seront plus modifiés.

Annexe F: Base d'apprentissage du PRI pour le cas $m = 5$ capteurs et $n = 2$ sources

Pour le PRI destiné à l'estimation des Angles d'Arrivée de deux sources sur une antenne de $m = 5$ capteurs, la taille du réseau est $m^2 = 25$ neurones sur la couche d'entrée, $l = 5$ neurones sur la couche cachée, et $n = 2$ neurones sur la couche de sortie.
La base d'apprentissage de ce PRI est composée de signaux simulés correspondant à la réception de deux signaux indépendants de $SNR = 25$dB, pour des Angles d'Arrivée variant de -60^o à $+60^o$. Les angles sont échantillonnés par pas de 1.5^o, et la dimension de la base est réduite en veillant à toujours respecter l'inégalité $\theta_1 < \theta_2$ (d'où en sortie $\tilde{\omega}_1 < \tilde{\omega}_2$) sur la base d'exemples. Cette façon de procéder permet en outre d'alléger la tâche du perceptron puisque le tri des angles par ordre de grandeur ($\tilde{\omega}_1 < \tilde{\omega}_2$ en sortie) est déjà effectué pour lui. On a donc θ_1 qui varie de -60^o à $+58.5^o$ par pas de 1.5^o, et θ_2 qui varie de $\theta_1 + 1.5^o$ à $+60^o$ par pas de 1.5^o. Cela fait un total de 3240 couples d'angles. Pour rendre le réseau plus robuste au bruit, nous utilisons $L = 2$ expériences pour chaque couple possible, soit un total de 6480 exemples dans la base. On dispose, pour calculer chaque entrée $\hat{R}^{nor}$, de $N = 40$ observations, ce qui permet au réseau d'apprendre relativement facilement.

Annexe G: MUSIC ne peut résoudre le problème d'Angles d'Arrivée pour deux sources totalement corrélées

Soient deux sources x_1 et x_2 totalement corrélées. On sait alors que le facteur de corrélation ρ entre ces deux sources vaut 1:

$$\rho = \frac{|E\{x_1 x_2^*\}|}{\sqrt{E\{|x_1|^2\}E\{|x_2|^2\}}} = 1$$

d'où

$$|E\{x_1 x_2^*\}|^2 = E\{|x_1|^2\}E\{|x_2|^2\} \tag{139}$$

On supposera que les sources ne sont pas triviales, c'est-à-dire que ni l'une ni l'autre ne sont identiquement nulles:

$$\exists t, x_1(t) \neq 0 \quad \text{et} \quad \exists t', x_2(t') \neq 0$$

Les espérances mathématiques de l'équation (139) sont estimées par une moyenne sur toutes les observations:

$$|\frac{1}{N}\sum_{t=1}^{N} x_1(t)x_2^*(t)|^2 = \left(\frac{1}{N}\sum_{t=1}^{N}|x_1(t)|^2\right)\ \left(\frac{1}{N}\sum_{t=1}^{N}|x_2(t)|^2\right)$$

soit, en simplifiant:

$$|\sum_{t} x_1(t)x_2^*(t)|^2 = \left(\sum_{t}|x_1(t)|^2\right)\ \left(\sum_{t}|x_2(t)|^2\right)$$

Si on pose $a_n = x_1(n), b_n = x_2(n)$, et si n varie sur le même intervalle que t, alors on voit tout de suite que l'équation précédente recèle une certaine analogie avec une égalité entre produits scalaires de vecteurs complexes:

$$|\sum_{n} a_n b_n^*|^2 = \left(\sum_{n}|a_n|^2\right)\ \left(\sum_{n}|b_n|^2\right)$$

Si on appelle $\vec{a}$ le vecteur $[a_1, ..., a_n, ..., a_N]^T$ et $\vec{b}$ le vecteur $[b_1, ..., b_n, ..., b_N]^T$, alors cette dernière équation équivaut à:

$$| < \vec{a}, \vec{b} > |^2 = \|\vec{a}\|^2.\|\vec{b}\|^2 \tag{140}$$

Séparons le vecteur $\vec{b}$ en deux vecteurs orthogonaux l'un à l'autre:

$$\vec{b} = \alpha\vec{a} + \vec{b}_\perp$$

tel que le vecteur $\vec{b}_\perp$ soit orthogonal à $\vec{a}$. On a alors:

$$< \vec{a}, \vec{b} >=< \vec{a}, \alpha\vec{a} + \vec{b}_\perp >= \alpha\|\vec{a}\|^2$$

et

$$\|\vec{b}\|^2 =< \vec{b}, \vec{b} >=< \alpha\vec{a} + \vec{b}_\perp, \alpha\vec{a} + \vec{b}_\perp >= |\alpha|^2.\|\vec{a}\|^2 + \|\vec{b}_\perp\|^2$$

d'où, en remplaçant dans (140):

$$|\alpha|^2.\|\vec{a}\|^4 = \|\vec{a}\|^2 \left(|\alpha|^2.\|\vec{a}\|^2 + \|\vec{b}_\perp\|^2\right) = |\alpha|^2.\|\vec{a}\|^4 + \|\vec{a}\|^2.\|\vec{b}_\perp\|^2$$

soit, en éliminant le terme $|\alpha|^2.\|\vec{a}\|^4$ qui apparaît des deux côtés:

$$\|\vec{a}\|^2.\|\vec{b}_\perp\|^2 = 0$$

Or on a dit au départ qu'aucune des deux sources n'est identiquement nulle. On a donc $\|\vec{a}\| \neq 0$. Par suite on en déduit:

$$\|\vec{b}_\perp\| = 0$$

d'où

$$\vec{b} = \alpha\vec{a} + \vec{b}_\perp = \alpha\vec{a}$$

c'est-à-dire:

$$\forall n, \;\; a_n = \alpha b_n$$

ou, pour en revenir aux notations de départ:

$$\forall t, \;\; x_1(t) = \alpha x_2^*(t)$$

Le vecteur instantané des sources $\boldsymbol{x}(t) = [x_1(t), x_2(t)]^T$ est donc de dimension 1. Ainsi, en omettant le bruit, on peut dire que chaque observation $\boldsymbol{y}(t)$ reçue sur l'antenne est un vecteur de longueur m appartenant à un sous-espace de dimension 1. Or MUSIC cherche à mettre en exergue un sous-espace propre SIGNAL de dimension $n = 2$, ce qui n'est évidemment pas possible ici. Cela explique pourquoi MUSIC est inefficace dans le cas de deux sources entièrement corrélées.

Au contraire, la méthode du Maximum de Vraisemblance[13] cherche à minimiser le critère

$$e_{MV} = \mathrm{tr}\{(P_{\hat{A}} - I)\hat{R}_y\}$$

d'après l'Annexe D. On voit bien que le fait que $\hat{R}_y$ ne soit pas de rang 2 mais de rang 1, ne modifie pas la minimisation du critère, puisque de toute façon, cette minimisation se fait sur des paramètres contenus dans $P_{\hat{A}}$, et non dans $\hat{R}_y$.

[13] ainsi que notre méthode neuronale qui minimise le même critère que le MV

References

[1] A. Angot, "Compléments de mathématiques", 6^e édition, MASSON 1982

[2] G. Burel, "Réseaux de Neurones en Traitement d'Images: des Modèles Théoriques aux Applications Industrielles", Thèse de Doctorat de l'Université de Bretagne Occidentale, Brest, 1991

[3] G. Burel, "Blind Separation of Sources: a Non-Linear Algorithm", Neural Networks, vol 5, No 6, pp 937-947, 1992

[4] G. Burel & N. Rondel, "Neural Networks for Array Processing: from DOA Estimation to Blind Separation of Sources", IEEE Conference on Systems, Man and Cyberbetics, Invited Paper, Le Touquet, France, 17-20 Oct 1993

[5] J.P. Burg, "Maximum Entropy Spectral Analysis", Proceedings of the 37th International SEG Meeting, Oklahoma City, 1967

[6] J. Capon, "High-Resolution Frequency-Wavenumber Spectrum Analysis", Proceedings of the IEEE, Vol 57, pp 1408-1418, 1969

[7] P.C.J. Hill & P.D. Wells, "Neural Network Beamformer Design for Mobile Radio Applications", SBT/IEEE International Telecommunications Symposium, Rio de Janeiro, Brazil, pp 220-224, August 22-25, 1994

[8] S. Jha, R. Chapman & T.S. Durrani, "Bearing Estimation using Neural Networks", Proc IEEE ICASSP, New-York, USA, 1988

[9] S. Jha, R. Chapman & T.S. Durrani, "Investigation into Neural Networks for Bearing Estimation", Signal Processing IV: *Theories and Applications*, J.L. Lacoume, A. Chehikean, N. Martyin & J. Malbos Eds, Elsevier Science Publishers, 1988

[10] P. Comon, C. Jutten & J. Hérault, "Blind Separation of Sources - Part 2: Problem Statement", Signal processing, vol 24, No 1, July 1991

[11] J. Hérault & C. Jutten, "Réseaux Neuronaux et Traitement du Signal", collection *Traitement du Signal*, Hermès, Paris, 1994

[12] S. Jha & T.S. Durrani, "Bearing Estimation using Neural Optimisation Methods", Proc IEEE ICASSP, Albuquerque, N.M., USA, 1990

[13] S. Jha & T.S. Durrani, "Direction of Arrival Estimation using Artificial Neural Networks", IEEE Trans. on Systems, Man and Cybernetics, vol 21, No 5, pp 1192-1200, 1991

[14] C. Jutten & J. Hérault, "Une Solution Neuromimétique au Problème de Séparation de Sources", Traitement du Signal, vol 5, No 6, 1988

[15] C. Jutten & J. Hérault, "Blind Separation of Sources - Part 1: an Adaptive Algorithm based on a Neuromimetic Architecture", Signal processing, vol 24, No 1, July 1991

[16] P. Martin & B. Lobert, "Optimisation par réseaux de neurones: application au traitement d'antennes", Revue Technique Thomson-CSF, vol 23, No 1, mars 1991

[17] R. Rastogi, P.K. Gupta & R. Kumaresan, "Array Signal Processing with Interconnected Neuron-Like Elements", Proc. ICASSP, pp 2328-2331, 1987

[18] N. Rondel & G. Burel, "Cooperation of Multi-Layer Perceptrons for Angles of Arrival Estimation", IEE 4th International Conference on Artificial Neural Networks, Cambridge, GB, 26-28 Juin, 1995

[19] R. Roy & T. Kailath, "ESPRIT-Estimation of Signal Parameters via Rotational Invariance Techniques", IEEE Trans. on Acoustics, Speech and Signal Processing, vol 37, No7, July 1989

[20] R.O. Schmidt, "Multiple Emitter Location and Signal Parameter Estimation", Proceedings of the RADC Spectrum Estimation Workshop, Rome Air Development Center, pp 234-257, Oct 3-5, 1979, republié dans IEEE Transactions on Antennas and Propagation, Vol. 24, No 3, pp 276-280, 1986

[21] R.O. Schmidt, "A Signal Subspace Approach to Multiple Emitter Location and Spectral Estimation", Ph.D. dissertation, Stanford University, 1981

[22] K. Sharman & T.S. Durrani, "A Comparative Study of Modern Eigenstructure Methods for Bearing Estimation - A New High Performance Approach", Proc. 25th IEEE Conference on Decision and Control, Athens, Greece, pp 1737-1742, Dec 1986

[23] E. Sorouchyari, "Blind Separation of Sources - Part 3: Stability Analysis", Signal processing, vol 24, No 1, July 1991

[24] P. Stoica & A. Nehorai, "MUSIC, Maximum Likelihood, and Cramer-Rao Bound", IEEE Trans. on Acoustics, Speech and Signal Processing, vol 37, No 5, pp 720-741, May 1989

[25] P. Stoica & A. Nehorai, "MUSIC, Maximum Likelihood, and Cramer-Rao Bound: further Results and Comparisons", IEEE Trans. on Acoustics, Speech and Signal Processing, vol 38, No 12, pp 2140-2150, Dec 1990

[26] P. Swerling, "Probability of Detection for Fluctuating Targets", IRE Transactions, vol 6, pp 269-308, Apr 1960

[27] C.W. Therrien, "Discrete Random Signals and Statistical Signal Processing", Prentice Hall, pp 519-522, 1992

[28] M. Wax & T.Kailath, "Detection of Signals by Information Theoretic Criteria", IEEE Trans. on Acoustics, Speech and Signal Processing, vol 33, pp 387-392, Apr 1985

[29] M. Wax & I.Ziskind, "On Unique Localization of Multiple Sources by Passive Sensor Arrays", IEEE Trans. on Acoustics, Speech and Signal Processing, vol 37, No 7, pp 996-1000, July 1989

[30] P.D. Wells & P.C.J. Hill, "Adaptive Antenna Beamforming Arrays using Artificial Neural Networks", Proc. of the Workshop on Neural Networks Applications and Tools, Liverpool, UK, Sept 13-14, 1993, Computer Society Press, editors P.J.G. Lisboa and M.J. Taylor, pp 1-5, 1994

Chapitre 3
Applications du Problème de l'Estimation d'Angles d'Arrivée

Ce Chapitre est dédié à quatre applications pratiques des méthodes théoriques proposées au Chapitre précédent. Dans ces quatre applications seront utilisées des méthodes d'estimation d'Angles d'Arrivée, une extension convolutive de la méthode de Séparation de Sources de Burel, et notre méthode de fusion entre ces deux approches.

La première application concerne l'Analyse de Documents. Avec l'intérêt croissant des administrations pour des systèmes de lecture automatique de caractères, on a vu naître un véritable domaine de recherche consacré à ces systèmes: c'est l'Analyse de Documents. Un des problèmes majeurs rencontrés par ces chercheurs est de savoir "redresser" des lignes de texte qui ne sont pas orthogonales au bord de la feuille. Nous proposons une reformulation du problème de l'estimation de l'Angle d'Inclinaison des Lignes d'un Texte dans le cadre plus familier du traitement d'antenne, et nous montrons comment il peut être rapproché du problème de l'estimation d'Angles d'Arrivée. Des résultats expérimentatux sont présentés.

La deuxième partie est consacrée à un problème très actuel pour les réseaux de télévision: c'est celui posé par le passage à un seul canal de transmission par chaîne, au lieu de 9 canaux actuellement. Nous montrons que le passage sans précaution à 1 seul canal rend la réception virtuellement impossible. Nous montrons ensuite comment, par une reformulation du modèle dans une optique "Séparation Aveugle de Sources", on peut séparer les échos et envisager la bonne réception des signaux TV.

Dans la troisième partie, il s'agit de traiter de signaux de parole: deux personnes parlent simultanément et on cherche à séparer les deux voix en question. Souvent appelé la "cocktail party", nous verrons que ce problème n'est pas simple et qu'il faut envisager une extension convolutive au modèle des signaux et au modèle du séparateur aveugle, pour résoudre le problème. Des expériences portant sur des données réelles sont relatées en fin de Section.

La quatrième et dernière partie s'intéresse également aux signaux de parole. Toutefois, il s'agit ici, à l'aide de deux micros formant une antenne, de retrouver l'Angle d'Arrivée d'une ou deux voix humaines. On teste ainsi les méthodes simples d'AdA (dans le cas d'une seule voix), et notre méthode fusionnant l'approche AdA avec la Séparation Aveugle (cela est nécessaire dans le cas de 2 micros et 2 voix). Des résultats relatant les expériences menées sont ensuite présentés et discutés.

1 Inclinaison des lignes d'un texte

1.1 Introduction

Nous nous plaçons ici dans le cadre d'un nouveau domaine de recherche appelé communément "Analyse de Documents" (acronyme AD). L'Analyse de Documents a pour but, à partir d'un document obtenu par scanning, d'analyser son contenu à la manière dont un être humain le ferait: séparation des parties écrites et des parties picturales, puis analyse du contenu de chacune des parties. Peu de chercheurs officiant dans le domaine de l'Analyse de Documents se sont penchés sur l'analyse des images, excepté ceux s'intéressant à l'analyse des cartes (Map Reading en anglais). Au contraire, l'analyse du contenu des parties écrites mobilise un nombre croissant de chercheurs, parce que ses applications sont multiples et représentent un marché potentiel formidable: lecture et analyse automatique des annuaires téléphoniques pour retranscription sous forme de code informatique, lecture automatique des divers formulaires administratifs (feuilles de Sécurité Sociale, chèques, déclarations d'impôts, adresses sur les lettres transitant par La Poste, etc), lecture et analyse automatiques des réponses aux formulaires destinés aux instituts de sondage, etc. On peut imaginer d'autres applications à l'infini, comme par exemple la lecture automatique (à haute voix) de tout document écrit aux aveugles et mal-voyants: modes d'emploi de médicaments ou d'ustensiles divers, livres et revues, courrier courant (lettres manuscrites et formulaires administratifs), dates de péremption des produits alimentaires, etc.

L'analyse du contenu des parties écrites passe par la Reconnaissance Optique des Caractères (ROC). Le texte à analyser est séparé en lignes, puis en mots (par diverses techniques morphologiques ou d'imagerie comme l'analyse en parties connexes). Enfin, des algorithmes tentent d'analyser ces mots lettre par lettre, ou dans leur globalité[1]. L'un des problèmes qui se posent alors aux scientifiques qui s'attachent à élaborer des algorithmes de "lecture automatique" est que les textes scannés sont rarement droits. En d'autres termes, la page (scannée à la main) a été placée "de travers" sur l'écran du scanner, si bien qu'au final les lignes de texte ne sont pas perpendiculaires au bord de la feuille. Ce phénomène d'"inclinaison des lignes" est encore plus frappant lorsqu'il s'agit d'analyser un texte manuscrit. En effet, il est clair que peu de personnes sont capables d'écrire des lignes parfaitement perpendiculaires au bord de la feuille!

[1] la technique utilisée dépend souvent du dictionnaire utilisé pour l'application. Par exemple s'il s'agit de "lire" des réponses manuscrites du type *oui/non*, il est préférable d'analyser le mot dans sa globalité. Au contraire, s'il s'agit d'un texte dactylographié pouvant faire appel à des mots très variés, il est certainement préférable de "lire" les mots lettre à lettre

C'est à ce problème de l'inclinaison des lignes d'un texte (typographié ou manuscrit) que nous nous intéressons plus précisément. Hinds *et. al.* [8] proposent d'estimer l'Angle d'Inclinaison des Lignes d'un Texte (AILT) par l'utilisation de techniques basées sur la transformée de Hough. Cependant, comme le soulignent fort judicieusement Aghajan et Kailath [1], cette méthode est assez peu robuste au bruit, et surtout, les techniques employées sont très coûteuses à la fois en temps de calcul et en espace mémoire.

Dans [1], une approche différente est proposée, basée sur la reformulation du problème dans le cadre du traitement d'antenne. Cette approche montre d'excellents résultats sur des textes dactylographiés, mais exige une étape de pré-traitement (afin de réduire le bruit, ainsi que pour renforcer la linéarité des lignes de texte) qui peut s'avérer difficile à mettre en œuvre sur un document manuscrit.

Dans cette section nous montrerons comment, par une reformulation globale du problème d'AILT, il est possible de proposer une solution simple et efficace utilisant les perceptrons PRI et PCSC [5], [12] imaginés au Chapitre 2. Cette nouvelle solution [13], parce qu'elle se base sur une analyse en profondeur des parallèles entre le problème d'Angles d'Arrivée et le problème d'Angle d'Inclinaison des Lignes d'un Texte, évite le passage par une étape de pré-traitement. En outre, elle peut être utilisée aussi bien pour des textes manuscrits que pour des textes dactylographiés. Enfin, elle permet d'estimer sans surcroît de calculs le nombre de lignes de texte, ainsi que leurs offsets.

1.2 Position du problème

Maintenant que nous avons exposé l'origine du problème, attachons-nous à en exprimer les modalités de manière littérale. Nous disposons d'un document (sous forme d'image numérique) contenant un texte (dactylographié ou manuscrit) dont les lignes sont susceptibles de n'être pas orthogonales au bord de la feuille (voir figure (1) pour un exemple). Notre but est d'estimer aussi précisément que possible l'Angle d'Inclinaison des Lignes[2] de manière à pouvoir par la suite "redresser" le texte, et améliorer ainsi les performances potentielles des algorithmes des Reconnaissance Optique de Caractères.

Considérons la figure (2) schématisant la représentation de l'image scannée ($L \times K$ pixels) d'un texte. Notre problème revient à estimer l'angle θ d'inclinaison des lignes, et, si cela est souhaité, le nombre de lignes ainsi que leur offset respectif x_i^0.

[2]Skew Angle en anglais

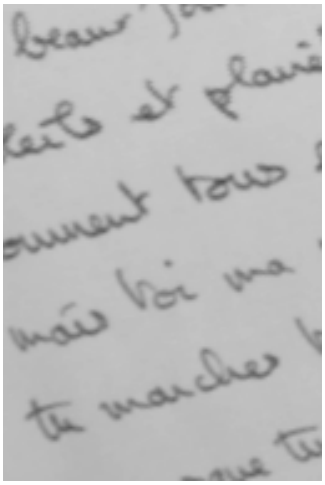

Figure 1: *Exemple d'image de texte dont les lignes d'écriture ne sont pas orthogonales au bord de la feuille*

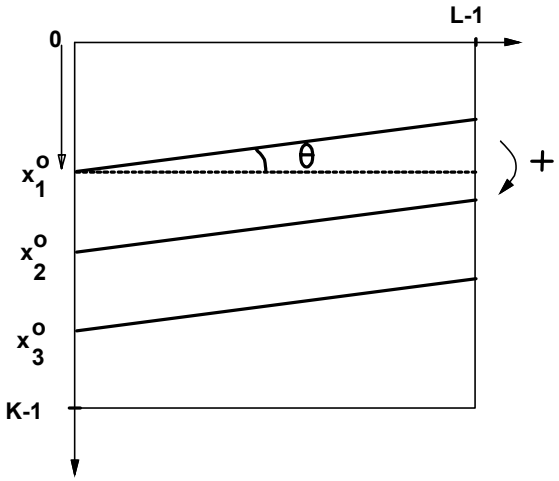

Figure 2: *Le problème de l'estimation de l'Angle d'Inclinaison des Lignes d'un Texte (AILT)*

1.3 Travaux Antérieurs

Si l'on considère que l'approche utilisant le transformée de Hough est d'une mise en œuvre délicate, parce que nécessitant de grandes capacités en mémoire et en temps de calcul (sans parler de la faible robustesse au bruit), on s'intéressera plus pariculièrement à l'approche d'Aghajan *et. al.* [1], [2]. Cette approche montre qu'un lien existe entre la modélisation du problème d'AdA et la modélisation du problème de l'estimation de l'Angle d'Inclinaison des Lignes d'un Texte (AILT). Pour bien montrer ce parallèle entre les deux approches, revenons tout d'abord aux bases du problème d'AdA.

1.3.1 Formulation du problème d'AdA

Nous nous plaçons ici dans le cadre de l'estimation d'un seul Angle d'Arrivée sur une antenne de L capteurs (figure (3)). Considérons un signal bande étroite de longueur

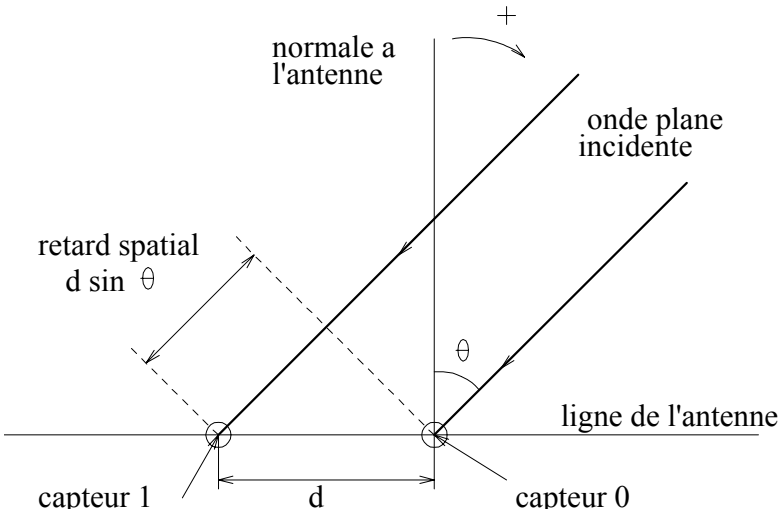

Figure 3: *Cas du réseau linéaire uniforme*

d'onde λ et de fréquence ν arrivant sur une antenne linéaire uniforme sous l'angle θ. Le retard spatial pris par l'onde entre deux capteurs successifs est $d \sin\theta$, si d est la distance commune entre deux capteurs successifs. Le retard de phase sur l'enveloppe complexe est donc

$$\omega = \frac{2\pi d}{\lambda} \sin\theta \tag{1}$$

Si l'on prend la référence s au premier capteur, l'enveloppe complexe du signal reçu sur le capteur l s'écrit:

$$y_l = e^{-j\omega l}.s + n_l \tag{2}$$

où n_l représente le bruit. Le vecteur signal reçu sur une antenne linéaire uniforme de L capteurs au temps t est donc

$$\boldsymbol{z_t} = [y_0, ..., y_l, ..., y_{L-1}]^T = \boldsymbol{a}(\omega)s_t + \boldsymbol{n_t}$$

où $\boldsymbol{a}(\omega)$ est le vecteur directionnel relatif à la direction ω:

$$\boldsymbol{a}(\omega) = [1, e^{-j\omega}, ..., e^{-j(L-1)\omega}]^T$$

et $\boldsymbol{n_t}$ est un vecteur de bruit de longueur L.

Si nous disposons de N observations, nous pouvons construire la matrice des observations

$$M = [\boldsymbol{z_1}, ..., \boldsymbol{z_t}, ..., \boldsymbol{z_N}]$$

Aghajan [1] [2] emploie alors une méthode Haute-Résolution, à savoir TLS-ESPRIT [14] se basant sur la décomposition en valeurs propres de la matrice de covariance estimée

$$\hat{R} = \frac{1}{N} M M^H$$

pour estimer ω. Aghajan en déduit alors θ par l'équation (1).

1.3.2 Formulation selon Aghajan du problème d'AILT

Maintenant que nous avons exprimé de manière succinte les différentes étapes de la formulation (et de la résolution) du problème d'AdA, il est plus facile de montrer comment Aghajan *et. al.* tracent les parallèles entre cette formulation et la modélisation du problème d'AILT.

La méthode proposée par Aghajan exige une étape de pré-traitement. Aghajan effectue tout d'abord un lissage horizontal[3] sur l'image, pour rendre connexes les composants d'une ligne, suivi d'un seuillage afin de binariser les pixels. Enfin, il effectue une détection (positive) de contours[4] qui transforme l'image en une suite de fines lignes parallèles à peu près rectilignes.

Une fois cette étape de pré-traitement effectuée, l'image obtenue n'est plus une image de texte en niveaux de gris, mais une image binaire (noir/blanc) représentant les fines lignes parallèles que l'on vient d'évoquer. Aussi l'image obtenue est-elle très semblable à la représentation schématique de la figure (2). Notons que la plupart des pixels de cette image sont à zéro (c'est le fond). Seuls les pixels matérialisant les fines lignes parallèles sont à 1.

Supposons que dans la colonne l de l'image, se trouvent k pixels non nuls situés aux ordonnées[5] $q_1, ..., q_k$ respectivement; supposons que nous ayons placé, en

[3] horizontal blurr en anglais

[4] positive edge detection en anglais

[5] dans une image l'abscisse correspond au numéro de colonne (en partant de la gauche) et l'ordonnée correspond au numéro de ligne (en partant du haut)

haut de la colonne l, un capteur qui reçoit tous les pixels contenus dans cette colonne, avec, pour chaque pixel, un retard proportionnel à la distance entre le capteur et ce pixel. Les pixels nuls (la majorité) n'envoient aucun signal. Par contre, les pixels non nuls envoient chacun le signal $e^{j\mu q_i}$, où μ est un paramètre. Le signal total reçu au capteur situé en haut de la colonne l est

$$y_l = \sum_{i=1}^{k} e^{j\mu q_i}$$

Si on note x_i^0 l'offset de la i-ème ligne de texte dans la colonne 0, l'offset de cette i-ème ligne dans la colonne l s'écrit:

$$q_i \simeq x_i^0 + l \tan\theta$$

Par suite nous avons

$$y_l = e^{j\mu l \tan\theta}.s + n_l \tag{3}$$

où n_l représente le bruit et $s = \sum_{i=1}^{k} e^{j\mu x_i^0}$. A ce stade, le lien entre l'AdA et l'AILT est facile à faire: il est clair que toute méthode permettant d'estimer ω dans l'équation (2) est susceptible d'estimer $\mu \tan\theta$ dans l'équation (3).

C'est en se basant sur cette similarité très forte qu'Aghajan utilise une méthode d'estimation d'AdA (à savoir ici TLS-ESPRIT [14]) pour résoudre le problème d'AILT.

Cependant, son approche n'est pas complètement satisfaisante en ce sens qu'elle ne fonctionne que sur des textes dactylographiés ayant subi une étape de pré-traitement quelque peu coûteuse en temps de calcul et nécessitant le réglage de différents paramètres (seuil de détection, paramètres de lissage, etc). Nous proposons ici une nouvelle approche [13] pouvant traiter tous les types de textes (dactylographiés ou non), et qui n'utilise aucun pré-traitement. Les données que nous utilisons sont donc l'image brute, c'est-à-dire des pixels en niveaux de gris, et il n'y a aucun seuil ni paramètre à régler.

1.4 Méthode proposée

1.4.1 Un nouveau modèle de signal

L'approche que nous proposons utilise tous les ressorts du formalisme habituel en

traitement d'antenne. Considérons une antenne linéaire de L capteurs située en haut de l'image brute, de telle manière que le capteur l soit situé exactement en vis-à-vis de la colonne l, pour tout $l \in [0, L-1]$. Le "signal" arrivant sur ce capteur est le contenu de sa colonne associée. Tout se passe comme si l'image traversait le réseau de capteurs, chaque capteur recevant le contenu de la colonne qui lui correspond.

Dans la suite, nous noterons $Y_l(.)$ le signal dans la colonne l, tel que $Y_l(k)$ soit la valeur, en niveau de gris, du pixel (k, l). Comme les méthodes d'AdA sont des méthodes bande-étroite, les données devraient être les enveloppes complexes des signaux reçus à chaque capteur, à une fréquence ν donnée. Avec $Y_l(.)$ représentant le signal dans la colonne l, l'enveloppe complexe y_l du signal $Y_l(.)$ à la fréquence ν s'écrit:

$$y_l = \frac{1}{\sqrt{K}} \sum_{k=0}^{K-1} Y_l(k) e^{-j2\pi\nu k} \tag{4}$$

où K, on le rappelle, est le nombre de lignes de pixels dans l'image. Remarquons que, contrairement à l'approche d'Aghajan, nous utilisons les signaux directement en niveaux de gris[6], ce qui évite toutes les étapes de pré-traitement utilisées dans son approche, et en particulier l'étape de binarisation[7].

1.4.2 Choix de la fréquence de travail

Les y_l étant connus, le problème est maintenant d'estimer l'Angle d'Inclinaison θ, ainsi que (si cela est demandé) le nombre de lignes de texte et leurs offsets respectifs. En traitement d'antenne classique, la fréquence de travail ν est fixée par des contraintes physiques (par exemple, c'est la fréquence du radar utilisé pour illuminer les objets à localiser). Au contraire ici, dans le problème de l'estimation de l'Angle d'Inclinaison des Lignes d'un Texte, aucune contrainte n'est fixée sur ν. Aghajan *et. al.* considérent le paramètre $\mu = 2\pi\nu$ à la place de ν, et conseillent de prendre $\mu = 1$, mais ne basent ce choix sur aucune considération théorique précise.

D'un autre côté, si on se base sur des considérations d'ordre purement théorique de traitement du signal, il apparaît assez judicieux de choisir la fréquence ν qui contient le *maximum* de l'énergie des signaux. En effet, les fréquences ν contenant peu d'énergie peuvent être considérées comme du bruit, et contiennent donc en principe peu d'information.

Nous allons effectuer une transposition dans l'espace des fréquences (par une

[6] en général, $Y_l(k)$ sera un entier entre 0 et 255

[7] pour la méthode d'Aghajan, on aurait $Y_l(k) = 0$ ou 1

Transformée de Fourier Discrète[8]) afin de déterminer la fréquence ν qui contient le maximum d'énergie.

Soit $\hat{y}_l(n)$ la TFD de la colonne l:

$$\hat{y}_l(n) = \frac{1}{\sqrt{K}} \sum_{k=0}^{K-1} Y_l(k) e^{\frac{-j2\pi nk}{K}} \tag{5}$$

Les symétries[9] de la TFD permettent de ne considérer que les indices n tels que $1 \leq n \leq \frac{K}{2}$. La fréquence optimale est $\nu = \frac{n_0}{K}$ telle que n_0 est la valeur de n qui maximise

$$e(n) = \sum_{l=0}^{L-1} |\hat{y}_l(n)|^2$$

où $e(n)$ est l'énergie totale (somme sur toutes les colonnes) à la fréquence $\frac{n}{K}$. Remarquons que puisque d'après la définition de la Transformée de Fourier Discrète, n_0 représente le nombre de cycles dans une colonne, nous en déduisons que n_0 représente aussi une estimation du nombre de lignes.

En effet, dans une image de lignes de texte parallèles, il est clair que le "signal pur", ce sont les lignes, et le bruit, c'est le fond, ainsi que les points dépassant de la ligne (les accents, les queues des lettres longues comme les l, les q ou les g). Par suite, dès lors que le Transformée de Fourier indique que le maximum de puissance du signal correspond à un nombre de cycles n_0, il paraît clair que ce nombre de cycles par colonnes est lié au signal lui-même. En examinant une page de texte à une distance telle que l'on ne puisse plus lire les mots, il apparaît clairement que l'aspect cyclique est tout entier contenu dans la régularité de l'espacement entre les lignes de texte; par suite, on en déduit que le nombre de cycles n_0 est égal au nombre de lignes du texte.

1.4.3 Estimation des offsets des lignes

Nous venons de montrer qu'il est possible d'estimer le nombre de lignes d'écriture par

[8] nous utiliserons par la suite un certain nombre de propriétés de la TFD que nous ne rappellerons pas systématiquement. Pour un énoncé des propriétés de la TFD, on pourra se référer à un ouvrage de traitement du signal, tel que [11] par exemple

[9] d'une part on peut éliminer les indices négatifs car il y a une symétrie par rapport à zéro. En effet, le signal étant réel, on a $\hat{y}_l(K-n) = \hat{y}_l^*(n)$. Par ailleurs la composante continue $n = 0$ ne présente pas d'intérêt.

$$n_0 = \max_{(n)} \left\{ \sum_{l=0}^{L-1} |\hat{y}_l(n)|^2,\ n \in [1, \frac{K}{2}] \right\}$$

De plus, ce nombre de lignes estimé est lié à la fréquence de travail ν par

$$\nu = \frac{n_0}{K}$$

Comment estimer l'offset de chacune de ces lignes d'écriture, c'est-à-dire comment estimer le numéro du pixel où démarre chacune des lignes dans la colonne $l = 0$? En choisissant de travailler en bande étroite, à la fréquence $\nu = \frac{n_0}{K}$, nous approximons le signal original large bande $Y_l(.)$ par une sinusoïde de fréquence $\frac{n_0}{K}$.

La Transformée de Fourier inverse de l'équation (5) à la fréquence $\frac{n_0}{K}$ donne, dans la colonne $l = 0$:

$$Y_0(k) \simeq \frac{2|\hat{y}_0(n_0)|}{\sqrt{K}} cos \left\{ 2\pi\nu \left(k + \frac{Arg(\hat{y}_0(n_0))}{2\pi\nu} \right) \right\} \tag{6}$$

En réalité, bien sûr, il faudrait ajouter toutes les autres fréquences qui, bien que contenant moins de puissance que $\frac{n_0}{K}$, contribuent au signal $Y_0(k)$. Le signal observé dans la colonne l est donc ici approximé par la sinusoïde de l'équation (6) représentée en figure (4). Pour un texte représenté en niveaux de gris, le codage du fond (blanc) correspond à des niveaux élevés, et le codage des lignes d'écritures (noires) correspond à des niveaux bas. Par conséquent, l'offset de la première ligne d'écriture correspond au premier minimum de la sinusoïde représentée en figure (4). L'offset de la première ligne[10] s'écrit donc

$$\hat{x}_0 = min_{(h)} \left\{ x = \frac{-Arg(\hat{y}_0(n_0)) + \pi + 2h\pi}{2\pi\nu}; x > 0 \right\}$$

Pour les $n_0 - 1$ lignes d'écriture suivantes, il faut décaler cet offset de $\frac{2\pi}{2\pi\nu}$ pixels, soit $\frac{1}{\nu}$ pixels à chaque fois, ce qui permet d'écrire l'offset de la ligne p comme:

$$\hat{x}_p^o = \hat{x}_0 + \frac{p}{\nu} \quad p \in [1, n_0 - 1]$$

[10]Pour certaines images difficiles (lignes fortement tronquées par exemple), cette méthode simple peut être mise en défaut, et demanderait à être améliorée. Nous nous sommes ici intéressés en priorité à la détermination de l'orientation, l'estimation des offsets étant considérée comme un simple problème annexe, qui sort de notre domaine d'intérêt

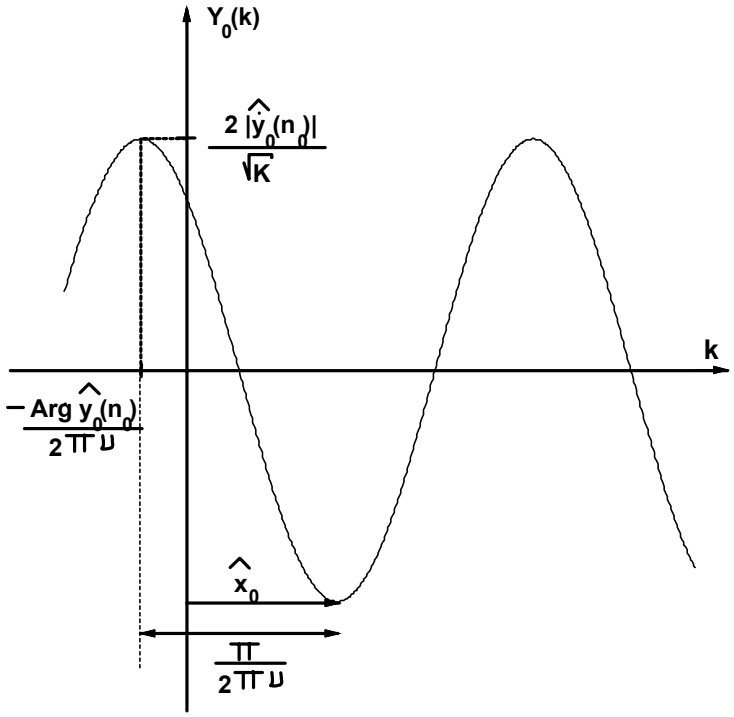

Figure 4: *Approximation par une sinusoïde du signal dans la colonne* 0

1.4.4 Estimation de l'Angle d'Inclinaison

Pour estimer l'Angle d'Inclinaison de Lignes du Textes, nous ne disposons que d'une seule observation du signal reçu sur les L capteurs situés en haut de la page:

$$[y_0, ..., y_l, ..., y_{L-1}]^T$$

où y_l est obtenu par l'équation (4) avec $\nu = \dfrac{n_0}{K}$.

Or, dans un problème typique d'estimation d'Angles d'Arrivée (voir Chapitre 2), on dispose de N observations différentes reçues sur une antenne de m capteurs, avec $m > n$, si n est le nombre de signaux. Le nombre de capteurs, m, n'est pas forcément grand, mais on essaie toujours d'avoir un nombre d'observations N le plus grand possible. Le fait de disposer de plusieurs observations permet par exemple d'estimer au mieux la matrice de covariance des signaux.

Lorsque, dans un problème d'estimation d'AdA, on se retrouve devant une seule observation sur une antenne très longue, on emploie une technique communément appelée "Spatial Smoothing" ou "Lissage Spatial", qui permet de transformer une seule observation sur une très longue antenne en plusieurs observations "virtuelles" sur une antenne "virtuelle" de taille moyenne.

Pour ce faire, on se donne une distance inter-capteur virtuelle d, et un nombre virtuel de capteurs m. Cela nous permet de construire une matrice d'observations virtuelle M telle que:

$$M = [\boldsymbol{z}_0, ..., \boldsymbol{z}_{d-1}] = \begin{bmatrix} y_0 & y_1 & \cdots & y_{d-1} \\ y_d & y_{d+1} & \cdots & y_{2d-1} \\ \vdots & \vdots & \ddots & \vdots \\ y_{(m-1)d} & y_{(m-1)d+1} & \cdots & y_{md-1} \end{bmatrix}$$

Le nombre d'observations virtuel est alors égal à d (nombre de vecteurs z_i). Pour $d \geq m$, la matrice de corrélation $m \times m$ s'écrivant

$$\hat{R} = \frac{1}{d} M M^H$$

est considérée comme une bonne estimation de la vraie matrice de corrélation.

Sous réserve que $md \leq L$, on a ainsi transformé un problème à L capteurs (L grand) et une seule observation, en un problème à m capteurs (m faible) et d observations (voir figure (5)). Nous devons maintenant définir la valeur optimale de

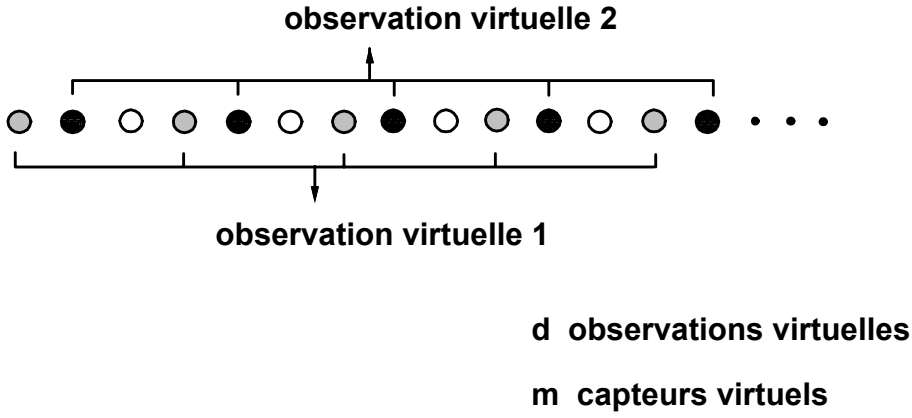

Figure 5: *Transformation par "Lissage Spatial"*

la distance inter-capteurs d. Dans un texte incliné de l'angle θ par rapport aux bords de la feuille, on a

$$Y_l(k) \simeq Y_0(k + l \tan\theta)$$

ce qui donne, au niveau de l'enveloppe complexe du signal:

$$y_l \simeq y_0 e^{j2\pi\nu l \tan\theta}$$

Si on change la distance inter-capteur en la faisant passer de 1 à d (distance virtuelle), alors le nouvel indiçage des capteurs peut s'exprimer comme:

$$\tilde{y}_{l,i} = y_{ld+i}$$

ce qui permet d'écrire, pour l'enveloppe complexe:

$$\tilde{y}_{l,i} \simeq \tilde{y}_{0,i} e^{j2\pi\nu l d \tan\theta}$$

équation qui est à comparer à son équivalent AdA:

$$y_l \simeq y_0 e^{-j\omega l}$$

Les deux approches sont alors absolument identiques, et après avoir estimé le paramètre intermédiaire ω par une méthode AdA, il est possible de remonter à l'Angle d'Inclinaison θ par:

$$\omega = -2\pi\nu d \tan\theta$$

Si on suppose que les Angles d'Inclinaison sont inférieurs à 63^o en valeur absolue, alors on a:

$$-63^o \leq \theta \leq 63^o$$

car il est très improbable qu'un Angle d'Inclinaison dépasse ces limites. Cela revient à:

$$-2 \leq \tan\theta \leq 2$$

soit

$$-4\pi\nu d \leq \omega \leq 4\pi\nu d$$

Pour éviter toute ambiguité, ω doit par ailleurs rester dans l'intervalle $[-\pi, \pi]$, donc on en déduit que

$$d = \frac{1}{4\nu}$$

est un bon choix pour la distance inter-capteur.

La longueur d'onde associée à la fréquence de travail ν est le nombre λ tel que $2\pi\nu\lambda = 2\pi$ (un cycle complet). On a donc

$$\lambda = \frac{1}{\nu}$$

Ainsi l'équation précédente peut-elle aussi s'écrire:

$$d = \frac{\lambda}{4}$$

ce qui veut dire que la valeur optimale de la distance inter-capteur est de l'ordre du quart de la longueur d'onde choisie.

Nous possédons maintenant toutes les données nécessaires à la résolution du problème d'AILT par notre méthode d'estimation d'AdA:

- matrice $m \times m$ de covariance:

  $$\hat{R} = \frac{1}{d} M M^H$$

  pour l'estimation "grossière" du PRI

- matrice $m \times d$ des observations:

  $$M = [\boldsymbol{z}_0, ..., \boldsymbol{z}_{d-1}]$$

  où

  $$\boldsymbol{z}_l = [y_l, y_{d+l}, ..., y_{(m-1)d+l}]^T$$

  pour l'estimation fine du PCSC.

D'après les dimensions de la matrice de covariance, les indications du Chapitre 2 Section 4 nous poussent à utiliser un Perceptron Réel d'Initialisation à trois couches: m^2 neurones sur la couche d'entrée, l proche de m neurones sur la couche cachée, et 1 neurone en sortie.

En ce qui concerne le PCSC, la Section 5 du Chapitre 2 indique qu'il faut construire un réseau complexe à trois couches: m neurones sur la couche d'entrée, de même que sur la couche de sortie, et 1 neurone sur la couche cachée (voir figure (6)). Les poids du PCSC seront initialisés grâce à l'estimation $\tilde{\omega}$ fournie par le PRI:

$$\hat{\boldsymbol{a}}_{init} = \boldsymbol{a}(\tilde{\omega})$$

et

$$B_{init} = \left[\hat{\boldsymbol{a}}_{init}^H \hat{\boldsymbol{a}}_{init}\right]^{-1} \hat{\boldsymbol{a}}_{init}^H = \frac{1}{m} \boldsymbol{a}^H(\tilde{\omega})$$

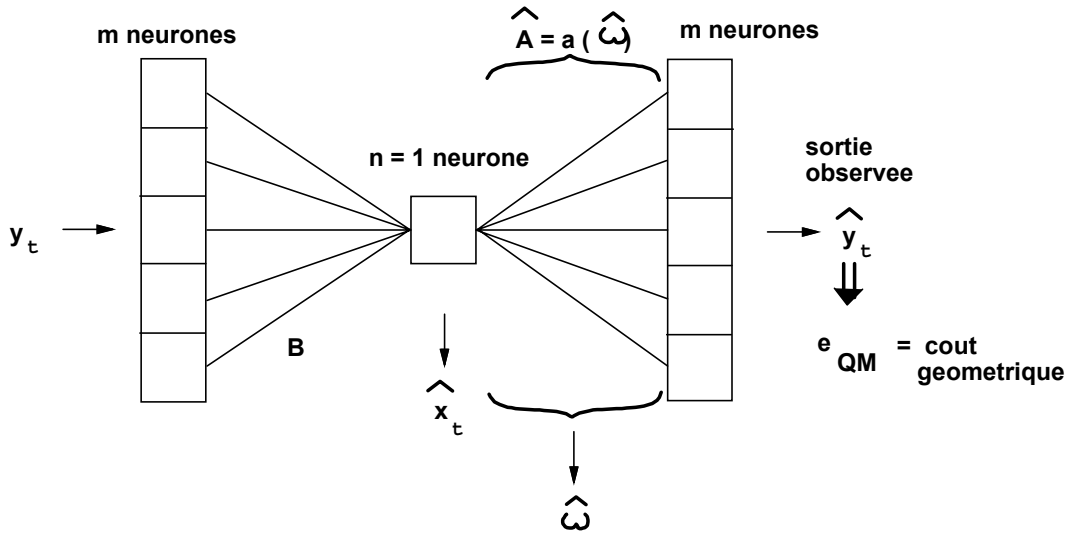

Figure 6: *Le Perceptron Complexe à Structure Contrainte pouvant servir à l'estimation de l'Angle d'Inclinaison des Lignes d'un Texte*

1.4.5 Résumé de la méthode proposée

La démarche à effectuer pour reformuler le problème de l'estimation d'AILT dans le cadre plus familier de l'estimation d'AdA est présentée ici étape par étape. En outre, nous proposons, pour les problèmes d'Analyse de Documents qui le nécessiteraient, une méthode d'estimation du nombre de lignes du texte, et des différents offsets de ces lignes.

L'algorithme d'estimation d'AILT que nous proposons est applicable sur tout document ou fragment de document, que le texte soit dactylographié ou manuscrit, et s'utilise directement sur un codage en niveaux de gris.

- Prendre l'image sur K lignes et L colonnes d'un texte en niveaux de gris et calculer la Transformée de Fourier Discrète $\hat{y}_l(n)$ des colonnes:

$$\hat{y}_l(n) = \frac{1}{\sqrt{K}} \sum_{k=0}^{K-1} Y_l(k) e^{\frac{-j2\pi nk}{K}}$$

  où $Y_l(k)$ est le niveau de gris du pixel (k, l).

- Le nombre n_0 de lignes du texte est estimé par la valeur de n qui maximise

$$\sum_{l=0}^{L-1} |\hat{y}_l(n)|^2$$

- La fréquence de travail optimale est $\nu = \frac{n_0}{K}$ (longueur d'onde $\lambda = \frac{1}{\nu} = \frac{K}{n_0}$).
- L'offset de la première ligne de texte est

$$\hat{x}_0 = min_{(h)} \left\{ x = \frac{-Arg(\hat{y}_0(n_0)) + \pi + 2h\pi}{2\pi\nu}; x > 0 \right\}$$

- L'offset des $n_0 - 1$ lignes suivantes est

$$\hat{x}_p^o = \hat{x}_0 + \frac{p}{\nu} \qquad p \in [1, n_0 - 1]$$

- Les signaux capteurs sont les y_l:

$$y_l = \frac{1}{\sqrt{K}} \sum_{k=0}^{K-1} Y_l(k) e^{\frac{-j2\pi n_0 k}{K}}$$

  pour $l \in [0, L-1]$
- La distance inter-capteur optimale est

$$d = \frac{\lambda}{4}$$

  et permet de construire la matrice $m \times d$ des observations:

$$M = \begin{bmatrix} y_0 & y_1 & \cdots & y_{d-1} \\ y_d & y_{d+1} & \cdots & y_{2d-1} \\ \vdots & \vdots & \ddots & \vdots \\ y_{(m-1)d} & y_{(m-1)d+1} & \cdots & y_{md-1} \end{bmatrix}$$

  ainsi que la matrice de covariance

$$\hat{R} = \frac{1}{d} M M^H$$

  avec $md \leq L$.
- Avec la matrice covariance normalisée $\hat{R}^{nor}$ issue de $\hat{R}$ (voir Chapitre 2 équation (109)), on peut obtenir une première estimation du paramètre ω: soit $\tilde{\omega}$ l'estimation fournie par le PRI.
- Initialisation des poids de sortie du PCSC grâce à $\tilde{\omega}$:

$$\begin{cases} \hat{\boldsymbol{a}}_{init} & = & \boldsymbol{a}(\tilde{\omega}) \\ B_{init} & = & \frac{1}{m} \boldsymbol{a}^H(\tilde{\omega}) \end{cases}$$

  et lancement de 15 itérations (passage des d observations virtuelles et rétropropagation de l'erreur en sortie).

- Extraction de l'estimation "affinée" $\hat{\omega}$ des poids du PCSC, et calcul de $\hat{\theta}$ en inversant l'équation

$$\omega = -2\pi\nu d \tan\theta$$

1.5 Expérimentations

Nous avons mené des expérimentations sur une imagette issue d'un texte manuscrit représentant un poème. La grandeur de l'image utilisée est $L = 128$ et $K = 192$. Il est évident que les résultats seraient meilleurs sur une image entière, mais nous allons voir que même sur une petite image, la méthode proposée estime correctement les Angles d'Inclinaison des Lignes du Texte. Notons que de toute façon, l'Angle d'Inclinaison effectif peut difficilement être défini (mesuré) avec une meilleur précision que quelques degrés, étant donné que le texte est écrit à la main et sans contraintes.

L'expérience est menée pour des lignes inclinées de 0^o, de 15^o, et de 30^o (figure 7)). L'algorithme utilisé est très précisément celui qui vient juste d'être résumé: on

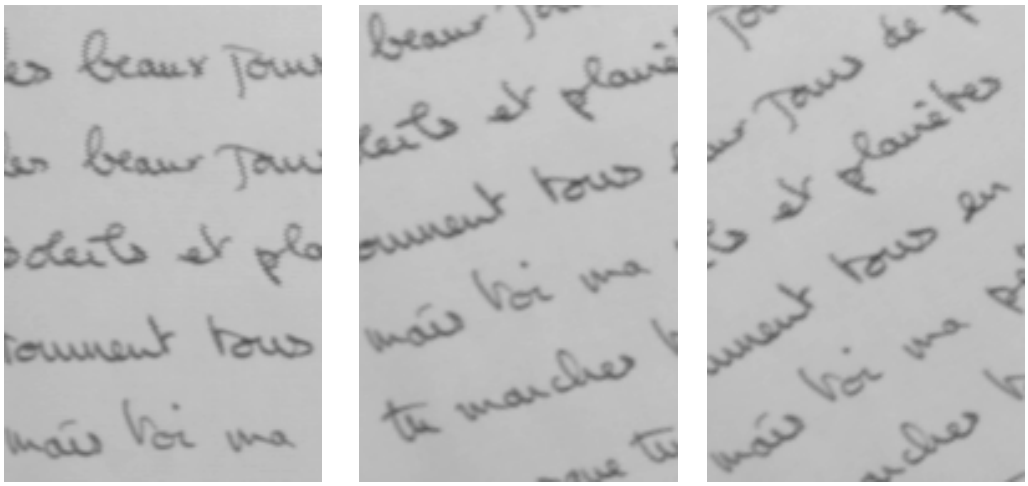

Figure 7: *Images du texte incliné à* $0^o, 15^o$ *et* 30^o

estime d'abord le nombre de lignes n_0, par une recherche de la fréquence qui porte le maximum d'énergie. La figure (8) montre pour exemple l'énergie en fonction de μ pour l'image à 30^o. On voit que le choix optimal, ici, serait de prendre $\mu = 0.131$[11], alors que le choix apparamment arbitraire préconisé par Aghajan ($\mu = 1$) correspond

[11] cela correspond à une fréquence optimale de $\nu = 0.021$, et à un nombre de lignes $n_0 = 4$

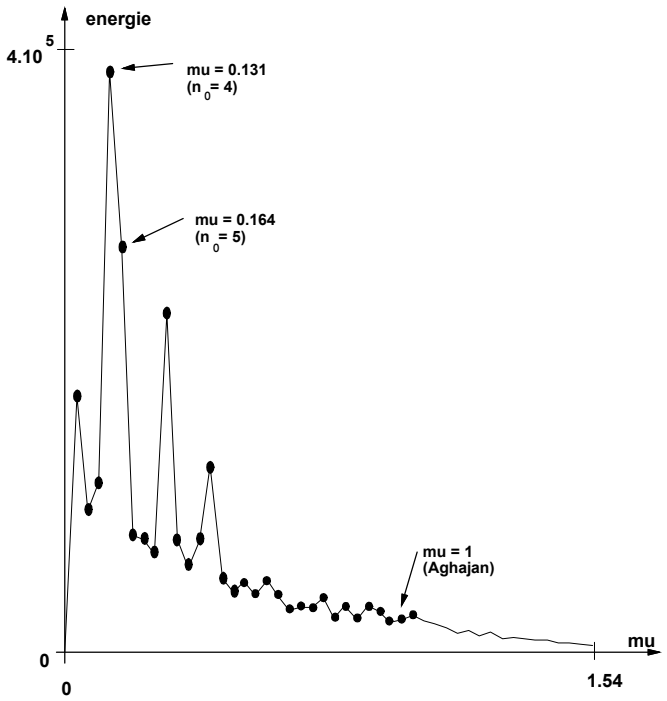

Figure 8: *Energie en fonction de μ dans l'image à* 30^o

à une quantité très faible d'énergie, et donc d'information, au sens propre du terme.

L'énergie étant maximisée, on en déduit la fréquence de travail ν, les offsets $\hat{x}_p^o$ des lignes dans la colonne 0, et la distance inter-capteur optimale d. On peut alors reformuler l'expression des signaux dans un cadre plus familier: celui du problème d'AdA. Les résultats obtenus sont présentés dans le tableau ci-dessous.

image	n_0	ν	d	θ (PRI)	θ (PCSC)	θ (Aghajan)
0^o	5	0.026	10	0.9^o	0.4^o	-0.3^o
15^o	5	0.026	10	8.7^o	15.0^o	-0.4^o
30^o	4	0.021	12	35.6^o	32.9^o	1.1^o

Le Perceptron Réel d'Initialisation utilisé pour cette application est un perceptron à trois couches de $100, 10$ et 1 neurones respectivement. La base d'entraînement de ce perceptron est composée de données simulées: on a simulé une antenne linéaire uniforme de $m = 10$ capteurs équidistants de $\lambda/2$ recevant un signal gaussien de moyenne zéro et de variance 1. Le bruit est à $SNR = 25$dB, et on dispose de $N = 40$ observations pour la construction des vecteurs d'entrée $\hat{R}^{nor}$. Les paramètres ω varient de -2.72 à $+2.72$ radians. En fait cette base est la même que celle évoquée au Chapitre 2 Section (8.1), avec $m = 10$ capteurs au lieu de 5 alors.

Si on compare nos résultats avec ceux obtenus sur les mêmes documents en utilisant la méthode d'Aghajan, on peut remarquer que cette méthode ne donne

pas satisfaction: toutes les estimations obtenues avec l'algorithme d'Aghajan sont proches 0^o, quel que soit l'Angles réel d'Inclinaison des Lignes.

En fait il est assez facile d'expliquer ce comportement: nous avons appliqué l'algorithme d'Aghajan au document brut (niveaux de gris), sans y effectuer la phase de pré-traitement, coûteuse en temps de calcul, que notre méthode ne nécessite pas.

Notons d'autre part que, comme cela est indiqué dans la méthode d'Aghajan, le paramètre μ utilisé est $\mu = 1$. Or, pour autant que nous ayons pu le constater[12], $\mu = 1$ correspond à $\nu = \frac{1}{2\pi} = 0.16$, c'est-à-dire une fréquence beaucoup plus grande (environ 8 fois plus grande) que la fréquence idéale que nous avons estimée $\nu = \frac{n_0}{K}$ (qui vaut environ 0.02). On peut même dire que cette fréquence élevée correspond davantage à du bruit qu'au vrai signal. Or, si le signal intéressant (le texte) est effectivement incliné (de 15^o ou 30^o), le bruit, lui, n'a pas d'Angle d'Inclinaison particulier. Il apparaît donc somme toutes assez normal que la méthode d'Aghajan trouve des Angles d'Inclinaison proches de zéro[13].

En ce qui concerne les résultats fournis par notre méthode, on peut constater que les Angles d'Inclinaison des Lignes du Texte sont très correctement estimés, d'autant plus que l'écriture manuscrite n'est pas aussi régulière qu'un texte dactylographié, ce qui signifie que l'Angle d'Inclinaison peut éventuellement varier sur une même ligne, et d'une ligne à l'autre. De plus, l'espacement entre les lignes n'est pas constant.

Pour illustrer la capacité de l'algorithme à estimer non seulement l'Angle d'Inclinaison, mais également le nombre de lignes et leur offset respectif dans la colonne 0, nous avons matérialisé l'estimation de ces paramètres sur la figure (9), dans le cas de lignes inclinées de 15^o. On constate que la méthode est très performante, même dans un cas difficile tel ce texte manuscrit non pré-traité.

1.6 Extension de l'approche proposée

1.6.1 Introduction

On a dit que pour affiner une estimation (d'AdA ou d'AILT selon le cas), il était

[12] voir figure (8) par exemple

[13] Notre avis sur la raison qui a poussé Aghajan à conseiller de choisir $\mu = 1$ est sans doute le fait que sur le texte dactylographié qu'il a utilisé pour ses expérimentations, *il se trouve*, par hasard, que le nombre de lignes du texte corresponde *justement* à $\mu = 1$. Si Aghajan avait essayé son algorithme sur un texte différent, il aurait sans doute constaté que $\mu = 1$ ne convient pas

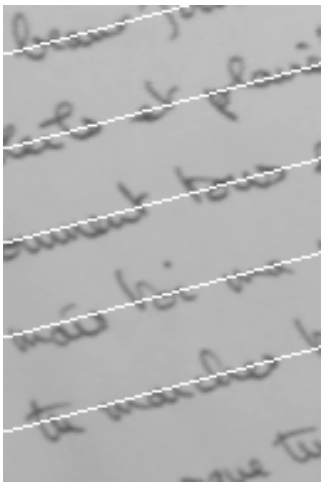

Figure 9: *Image à 15^o superposée avec la représentation de son Angle d'Inclinaison estimé et des offsets estimés de ses lignes.*

intéressant de pouvoir disposer du plus grand nombre possible d'observations. Or ici, pour l'estimation d'AILT avec la technique du "Lissage Spatial", on a vu que le nombre d'observations était limité par le nombre de colonnes L de l'image utilisée. En effet, pour un nombre virtuel m de capteurs, on obtient un nombre d'observations d tel que:

$$md \leq L$$

Par exemple, pour les expériences que nous venons de mener, on a choisi une distance inter-capteur $d = 12$, de qui donne un nombre de capteurs $m = 10$ puisque $L = 128$ colonnes de pixels. Le nombre d'observations est donc égal à 12, ce qui est assez limité si on demande une grande précision dans l'estimation. Il serait donc intéressant, le cas échéant, de pouvoir construire de nouveaux signaux capteurs (observations) à partir de la même image, afin d'affiner les résultats.

Il serait effectivement possible, d'une façon assez directe, d'obtenir de nouvelles observations y_l. En effet, il suffirait pour cela de calculer l'enveloppe complexe y_l à une fréquence ν' différente de la fréquence optimale calculée $\nu = \frac{n_0}{K}$. Nous disposerions alors de $d = 12$ nouvelles observations, nous permettant ainsi d'effectuer une nouvelle estimation de l'Angle d'Inclinaison. Cependant, une telle façon de faire n'est pas très intéressante, parce qu'en fin de compte nous obtenons deux estimations θ et θ' différentes, obtenues séparément sur deux jeux d'observations y_l et y'_l. De fait, les observations y_l et y'_l, associées respectivement aux fréquences ν et ν', ne sont pas compatibles. En effet, avec les observations y_l associées à la fréquence ν, les signaux sont modélisés par

$$\boldsymbol{y} = \begin{bmatrix} 1 \\ e^{-j\omega} \\ \vdots \\ e^{-j(m-1)\omega} \end{bmatrix} x$$

et un algorithme d'AdA estime ω, d'où l'estimation de $\hat{\theta}$ par l'inversion de

$$\omega = -2\pi\nu d \tan\theta \tag{7}$$

D'un autre côté, avec les observations y'_l associées à la fréquence ν', les signaux sont modélisés par

$$\boldsymbol{y'} = \begin{bmatrix} 1 \\ e^{-j\omega'} \\ \vdots \\ e^{-j(m-1)\omega'} \end{bmatrix} x' \tag{8}$$

et avec un estimateur d'AdA on obtient ω' d'où θ par l'inversion de

$$\omega' = -2\pi\nu' d \tan\theta \tag{9}$$

De fait ω et ω' sont incompatibles, car leur relation à θ est différente, ce qui rend les jeux d'observations y_l et y'_l incompatibles.

1.6.2 Normalisation en fréquence

Pour rendre les signaux y_l issus du filtrage à la fréquence ν compatibles avec les signaux y'_l issus du filtrage à la fréquence ν', il faudrait pouvoir en quelque sorte "normaliser" ces y'_l de façon à les associer à ω au lieu de ω'.

Or on voit bien, d'après les relations (7) et (9), que ω et ω' sont liées par

$$\omega = \omega' \frac{\nu}{\nu'}$$

Cette remarque pousse à essayer de multiplier l'argument de tous les éléments du vecteur $\boldsymbol{y'}$ dans l'équation (8) par $\dfrac{\nu}{\nu'}$, afin de faire apparaître ω. Soit y'_l un élément de $\boldsymbol{y'}$. On a:

$$y'_l = e^{-jl\omega'} x' = e^{-jl\omega'} \rho' e^{j\phi'}$$

en posant $x' = \rho' e^{j\phi'}$. Si on multiplie l'argument de y'_l par $\frac{\nu}{\nu'}$, on arrive à

$$\begin{aligned} \hat{y}'_l &= \rho' e^{j(-l\omega'+\phi')\frac{\nu}{\nu'}} \\ &= \rho' e^{j(-l\omega+\phi'\frac{\nu}{\nu'})} \\ &= e^{-jl\omega}\hat{x}' \end{aligned}$$

où $\hat{x}' = \rho' e^{j\phi'\frac{\nu}{\nu'}}$ ne dépend pas de l. On obtient donc finalement un nouveau vecteur observation $\hat{\boldsymbol{y}}'$:

$$\hat{\boldsymbol{y}}' = \begin{bmatrix} 1 \\ e^{-j\omega} \\ \vdots \\ e^{-j(m-1)\omega} \end{bmatrix} \hat{x}'$$

qui est compatible avec le vecteur $\boldsymbol{y}$ associé à la fréquence ν.

Grâce à cette astuce (multiplication de l'argument de tous les éléments de $\boldsymbol{y}'$ par le rapport des fréquences $\frac{\nu}{\nu'}$), on peut obtenir 2 jeux d'observations qui sont compatibles, et donc associables, pour réaliser une estimation précise sur ω. On obtient ensuite θ par l'équation (7).

Remarque importante:

Pour la démonstration, nous avons effectué la multiplication par un réel de l'argument d'un complexe, sans vérifier si nous en avions le droit. En réalité, une telle opération est dangereuse et peut mener à des résultats totalement erronés, si elle est conduite sans précaution.

En effet, lorsque nous écrivons

$$y'_l = e^{-jl\omega'}x' = \rho' e^{j(-l\omega'+\phi')}$$

nous avons, au niveau de l'argument:

$$Arg(y'_l) = -l\omega' + \phi' + 2k'_l\pi \quad \in]-\pi, \pi] \tag{10}$$

Le "$+2k'_l\pi$" est très important et ne doit pas être oublié, car lorsque nous faisons appel à une fonction Argument (comme ici puisque nous avons besoin de multiplier l'argument de y'_l par $\frac{\nu}{\nu'}$), nous obtenons obligatoirement un réel situé entre $-\pi$ et π. Or ϕ', l'argument de x', est un réel quelconque appartenant à $]-\pi, \pi]$. Il est donc tout à fait vraisemblable que $l\omega' + \phi'$ puisse dépasser les limites $]-\pi, \pi]$ imposées *de facto* sur $Arg(y'_l)$. De plus, comme cela est clairement indiqué sur la figure (10),

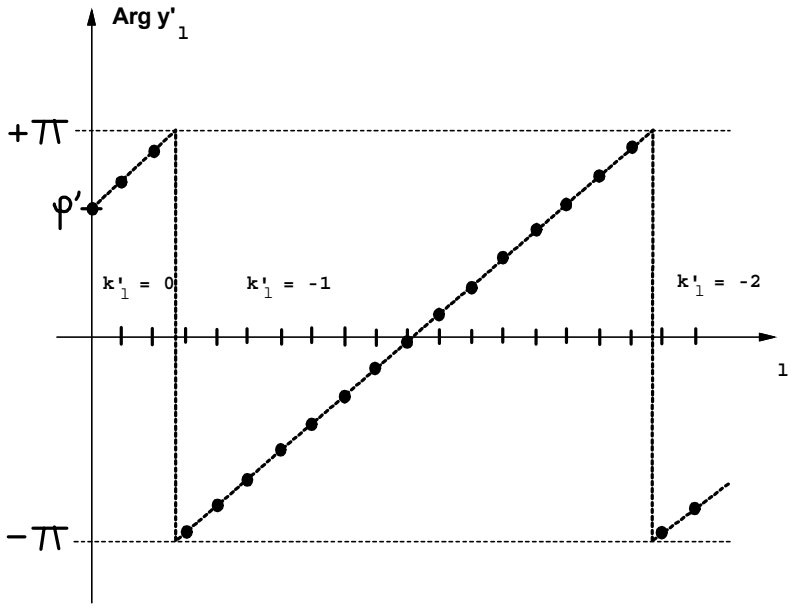

Figure 10: *Illustration de la non-continuité de l'argument de y'_l en fonction de l*

lorsque le numéro de capteur l croît, il peut arriver que la phase $-l\omega' + \phi'$ fasse plusieurs tours de cadran. Par conséquent le "$+2k'_l\pi$" ajouté à la fin de l'équation (10) peut effectivement *varier* en fonction de l.

Si on multiplie sans précaution l'argument de y'_l par $\dfrac{\nu}{\nu'}$, que se passe-t-il? Sans rétablissement de continuité de l'argument, on a:

$$Arg(\hat{y}'_l) = \frac{\nu}{\nu'} Arg(y'_l) = -l\omega + \frac{\nu}{\nu'}\phi' + 2k'_l\frac{\nu}{\nu'}\pi$$

Or le dernier terme, de manière évidente, dépend de l. Il y aura donc interférence avec le premier terme "$l\omega$", ce qui va entraîner une grande confusion au niveau des ω, puisque le modèle des signaux devient alors:

$$\hat{\boldsymbol{y}}' = \begin{bmatrix} 1 \\ e^{-j\left(\omega - 2k'_1\frac{\nu}{\nu'}\pi\right)} \\ \vdots \\ e^{-j(m-1)\left(\omega - 2k'_{m-1}\frac{\nu}{\nu'}\pi\right)} \end{bmatrix} \hat{x}'$$

avec des k'_l pas forcément tous égaux entre eux. On voit bien que ce modèle ne correspond pas au modèle habituel de l'AdA, et qu'il risque par conséquent de produire des résultats dénués de toute signification.

Cela prouve qu'il est nécessaire, avant de multiplier l'argument de y'_l par $\dfrac{\nu}{\nu'}$,

de rétablir la continuité de la phase. Pour cela, il suffit de créer un petit algorithme qui va tester les sauts de phase éventuels entre deux capteurs successifs (comme par exemple sur la figure (10) où on remarque un saut de phase entre y_2 et y_3), afin de rétablir cette continuité de phase entre les deux capteurs.

Après rétablissement de la continuité de la phase, on a, pour chaque capteur l:

$$\tilde{Arg}(y'_l) = -l\omega' + \phi'$$

où $\tilde{Arg}(.)$ n'appartient pas forcément à $]-\pi, \pi]$. On peut alors multiplier sans crainte cet argument par le rapport des fréquences $\frac{\nu}{\nu'}$. On a finalement:

$$\tilde{Arg}(\hat{y}'_l) = \frac{\nu}{\nu'}\tilde{Arg}(y'_l) = -l\omega + \frac{\nu}{\nu'}\phi'$$

On obtient alors un signal $\hat{\boldsymbol{y}}'$ normalisé à la fréquence ν et tout-à-fait compatible avec le signal $\boldsymbol{y}$ directement issu du filtrage à la fréquence ν.

Choix de la nouvelle fréquence ν':

On se rappelle, en début de section, avoir indiqué le choix de la fréquence $\nu = \frac{n_0}{K}$ comme la fréquence optimale associée à l'entier n_0 maximisant:

$$\sum_{l=0}^{L-1} |\hat{y}_l(n)|^2$$

pour $n \in [1, \frac{K}{2}]$ et n_0 correspondant à une estimation du nombre de lignes de texte dans l'image. Ce choix est motivé par le fait qu'il y a plus d'information dans une fréquence où il y a beaucoup d'énergie que dans une fréquence où il y a peu d'énergie. Cependant, il arrive que plusieurs fréquences, bien que moins importantes, contiennent également une énergie non négligeable[14]. Dans ce cas, on peut choisir de sélectionner ces fréquences qui contiennent aussi de l'information. Pour chaque $\nu' \neq \nu$ sélectionnée, on répètera l'opération consistant à multiplier[15] l'argument de l'enveloppe complexe des y'_l par $\frac{\nu}{\nu'}$. On obtient alors plusieurs jeux d'observations "normalisées en fréquence", et compatibles avec le jeu optimal associé originellement à la fréquence ν. On espère ainsi, par l'augmentation du nombre d'observations, améliorer la précision de l'estimation.

[14] c'est le cas, par exemple, si l'imagette contient des lignes tronquées (comme c'est le cas ici pour $\theta = 30^o$ et $\theta = 15^o$), car le nombre de lignes du texte peut alors osciller entre $n_0 - 1$ et $n_0 + 1$. En conséquence, la fréquence optimale ν peut osciller entre $\frac{n_0 - 1}{K}$ et $\frac{n_0 + 1}{K}$

[15] après rétablissement de la continuité de la phase

1.6.3 Expérimentations

Pour valider l'approche proposée, une expérimentation effectuée sur l'image à 30^o (figure (11)) est montrée ici. Rappelons que dans un premier temps, nous avions

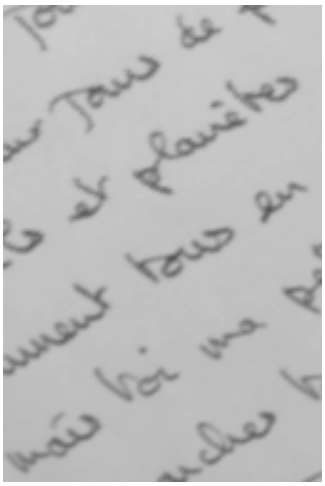

Figure 11: *Image à* 30^o*: le nombre de lignes oscille entre* 4 *et* 5

sélectionné la fréquence "optimale" $\nu = 0.021$ (soit $\mu = 0.131$) correspondant à un nombre de lignes estimé à $n_0 = 4$ lignes. En réalité, si on examine attentivement la figure (11), on voit que certaines lignes sont tronquées, incomplètes, et qu'il est possible (même pour un humain) d'hésiter entre 4 ou 5 lignes. On constate dans le même temps que pour $n'_0 = 5$, l'énergie du signal correspondant à la fréquence $\nu' = \frac{n'_0}{K}$ est effectivement l'énergie la plus élevée après celle correspondant à $n_0 = 4$ (*cf* figure (8)). De plus, cette énergie est loin d'être négligeable, ce qui porte à croire que le signal à cette fréquence contient de l'information pertinente.

Notre démarche est alors la suivante: nous allons tout d'abord faire une estimation de ω' avec les enveloppes complexes des signaux y'_l à la fréquence $\nu' = \frac{n'_0}{K} = 0.026$. On estimera alors θ par l'inverse de:

$$\omega' = -2\pi\nu' d \tan\theta$$

Nous allons ensuite "transposer" de la fréquence ν' à la fréquence ν les d vecteurs observation formés par les y'_l, par l'opération suivante:

$$\hat{y}'_l = |y'_l| e^{j\frac{\nu}{\nu'}\tilde{Arg}(y'_l)}$$

en ayant pris soin d'assurer la continuité de l'argument. On obtient alors d vecteurs d'observation à la fréquence ν. Ils nous permettent d'effectuer une estimation du

paramètre ω, d'où θ par l'inversion de:

$$\omega = -2\pi\nu d \tan\theta$$

Enfin, puisque les y_l et les $\hat{y}'_l$ sont tous des vecteurs observation associés à la fréquence ν, on peut les adjoindre: ils forment alors un jeu de $2d$ vecteurs observation, permettant ainsi une observation plus fine de ω. L'Angle d'Inclinaison θ est alors estimé par la même équation que ci-dessus.

Les résultats obtenus pour ces trois bases de données[16] sont comparés avec les résultats présentés précédemment pour la base formée de l'enveloppe à la fréquence ν. Le tableau suivant résume l'expérience:

estimations	$\nu = 0.021$	$\nu' = 0.026$	ν' transposée	$\nu + \nu'$ transposée
nbre de cycles n_0	4	5	$5 \rightarrow 4$	4 et $5 \rightarrow 4$
nbre d'observations d	12	12	12	24
estimation angle θ	32.9^o	27.8^o	28.0^o	31.4^o
écart en degrés	$+2.9^o$	-2.2^o	-2.0^o	$+1.4^o$

On constate que l'écart à l'Angle d'Inclinaison supposé vrai (30^o) diminue lorsque l'on passe de 12 à 24 observations, ce qui va dans le sens de la théorie développée par Stoica [15], qui dit que la variance de l'erreur sur l'estimation est proportionnelle à l'inverse du nombre d'observations.

Ces résultats montrent en fait qu'il est toujours intéressant, lorsque l'on constate que plusieurs fréquences ont des puissances non négligeables, de calculer les enveloppes complexes de l'image à ces différentes fréquences et de les transposer à la fréquence principale $\nu = \dfrac{n_0}{K}$ afin d'obtenir un maximum de vecteurs observation. On peut alors penser que l'estimation obtenue n'en sera que plus précise.

1.7 Conclusion

Dans le domaine en plein développement de l'Analyse de Documents, l'un des problèmes majeurs provient du fait que les lignes des textes à analyser ne sont pas toujours orthogonales au bord de la feuille. Il apparaît donc essentiel de développer des algorithmes permettant d'estimer l'Angle d'Inclinaison des Lignes du Texte, afin de pouvoir "redresser" ces lignes, et rendre ainsi leur lecture par un système de

[16] enveloppe complexe à la fréquence ν', enveloppe complexe à la fréquence ν' "transposée" à la fréquence ν, et association de l'enveloppe complexe à la fréquence ν avec l'enveloppe à la fréquence ν' transposée à la fréquence ν

Reconnaissance Optique de Caractères la plus aisée possible.

Après avoir montré les parallèles existant entre ce problème et l'estimation d'Angles d'Arrivée, nous avons proposé une méthode complète permettant d'estimer l'Angle d'Inclinaison des Lignes d'un Texte, le nombre de lignes, ainsi que leurs offsets respectifs.

Basée sur une reformulation du problème de l'AILT dans le cadre plus familier de l'AdA, cette méthode a montré, sur des exemples difficiles (textes manuscrits), de très bonnes performances alliées à une simplicité extrême d'utilisation. Aucun pré-traitement n'est requis, et les images en niveaux de gris sont traitées directement par l'algorithme. De plus, il n'y a aucun paramètre à régler.

Une extension basée sur une décomposition de l'image sur plusieurs bandes de fréquence , suivie par une transposition de ces signaux à une fréquence de référence, permet de multiplier le nombre des données, procurant par là-même une plus grande précision dans l'estimation.

2 Réseaux Monofréquence

2.1 Introduction

L'un des problèmes majeurs qui se pose actuellement dans le domaine des communications numériques, est celui de la saturation des fréquences hertziennes. En télévision, par exemple, ce sont les ondes UHF qui sont utilisées. Dans cette gamme de fréquence, la largeur de la bande passante (environ 8 MHz) des signaux TV divise l'intervalle UHF en 45 canaux. Si on n'utilisait qu'un seul émetteur, il serait donc possible de placer 45 chaînes de télévision différentes mobilisant chacune un seul canal. Cependant, vu la grandeur du territoire français, il est nécessaire d'utiliser des réémetteurs-amplificateurs afin de pouvoir retransmettre les programmes avec la même qualité de réception sur tout l'hexagone. Enfin, pour éviter que certains foyers ne reçoivent des échos parasites provoqués par des réémetteurs-amplificateurs trop proches les uns des autres, un système assez astucieux de répartition des canaux a été imaginé.

Actuellement, le système en place octroie 9 canaux à chaque chaîne, ce qui, au vu des 45 canaux qui composent la bande UHF, ne permet la diffusion que de 5 programmes télévisés différents. La nécessité d'avoir 9 canaux par chaîne s'explique comme suit: le territoire français est quadrillé par des émetteurs (ou plutôt des

réémetteurs) qui reçoivent le signal (par exemple les programmes FR3) sur un canal donné (par exemple le canal $c_i,\ i \in [1,9]$) puis le réémettent après l'avoir amplifié (voir figure (12)). Pour un foyer se trouvant tel qu'indiqué sur la figure,

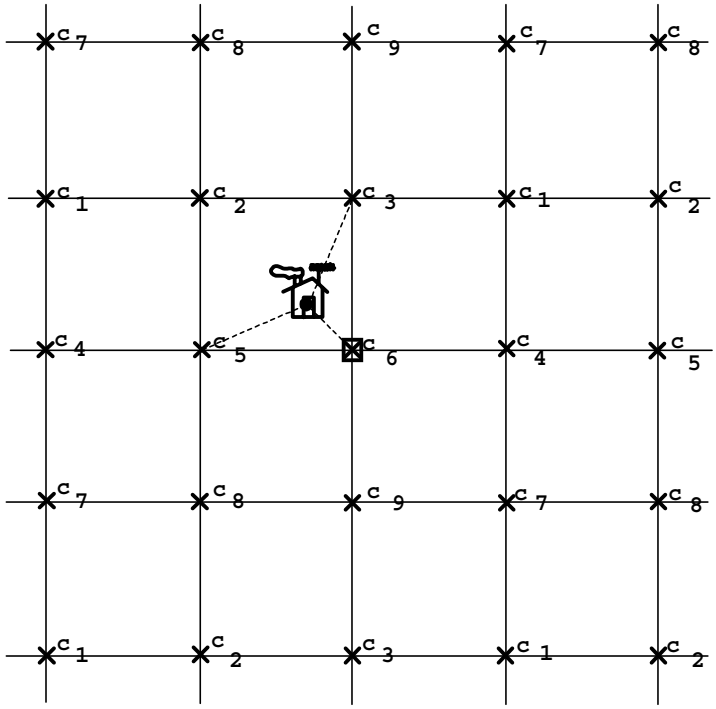

Figure 12: *Répartition géographique des* 9 *canaux pour une chaîne donnée*

l'émetteur le plus proche utilise le canal c_6, et c'est donc en principe dans cette bande de fréquence que l'énergie est la plus forte. Il suffit alors par un réglage, soit manuel soit automatique, de régler la fréquence de réception de FR3 de manière à obtenir la qualité de réception maximale. Ici, d'après la position de la maison, c'est l'émetteur utilisant le canal c_6 qui est le plus proche, et c'est probablement lui qui fournit la meilleure qualité d'image. On imagine facilement qu'un appareil de réglage automatique contient en mémoire les fréquences des 9 canaux d'émission de FR3, et qu'il choisit, après comparaison, le canal qui lui envoie le plus de puissance.

Ce système, de par sa définition, permet une très bonne qualité d'image; d'un autre côté, il nécessite l'emploi de 9 canaux par chaîne, ce qui peut sembler relativement gourmand, surtout si l'on tient compte du fait que chaque canal diffuse le même programme. On peut dire en résumé que ces 9 canaux ne sont pas "complémentaires", mais "supplémentaires".

On pourrait penser qu'un moyen facile d'augmenter le nombre de chaînes télévisuelles serait tout simplement de réduire le nombre de canaux par chaîne, et passer d'un quadrillage 3×3 à un quadrillage 2×2. Cependant, bien que très séduisante a priori,

cette idée n'est pas envisageable en pratique. En effet, considérons un quadrillage 2×2 tel que celui représenté sur la figure (13). Il y a quatre canaux en tout, et

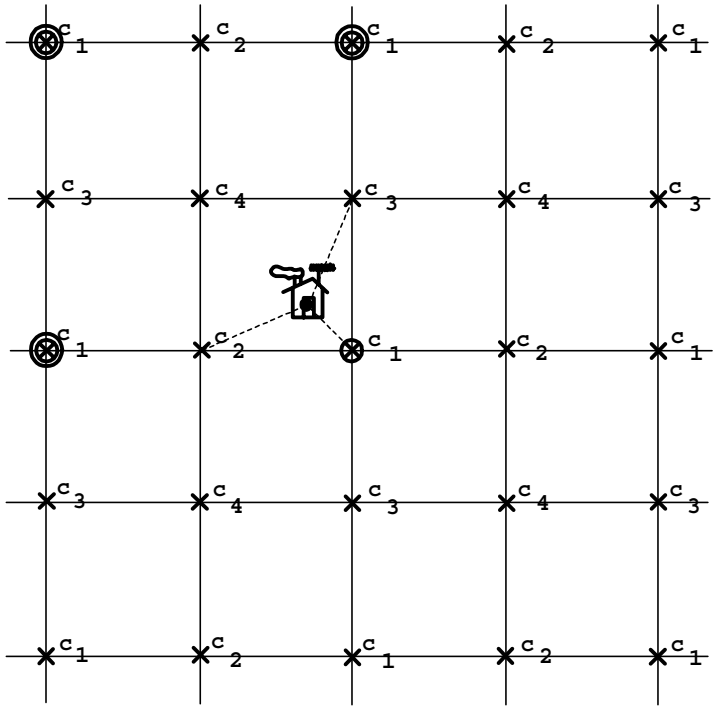

Figure 13: *Répartition géographique pour un système à* 4 *canaux*

pour une maison située à l'endroit indiqué sur le schéma, l'émetteur le plus proche utilise le canal c_1. Le réglage en fréquence devrait donc conduire au choix du canal c_1, puisque c'est ce canal qui fournit le plus de puissance. Cependant, les quatre émetteurs les plus proches utilisant le canal c_1 (ces émetteurs sont matérialisés par deux cercles concentriques sur la figure (13)), se trouvent désormais suffisamment près de la maison considérée pour que leurs signaux ne soient plus négligeables (dans le cas à 9 canaux, les émetteurs employant le même canal sont suffisamment éloignés pour ne pas interférer entre eux). De ce fait, si le récepteur est réglé sur le canal c_1, le signal reçu est la somme des différentes ondes se propageant à la fréquence c_1:

$$S_1(t) = \alpha_0 x_1(t) + \alpha_1 x_1(t - \tau_1) + \alpha_2 x_1(t - \tau_2) + \alpha_3 x_1(t - \tau_3)$$

où $x_1(t)$ est le signal correspondant au programme de FR3 sur la fréquence c_1, α_0 est le coefficient correspondant à l'émetteur le plus proche de la maison, et

$$\forall\, i \in [1, 3],\ 0 < |\alpha_i| < |\alpha_0|$$

Un retard τ_i correspond au temps de propagation entre l'émetteur principal et un émetteur secondaire, auquel est ajouté le temps de traitement et d'amplification,

ainsi que le temps de réémission vers l'antenne du téléviseur.

Comme ces signaux retardés sont de puissance non négligeable, on obtient finalement une image très brouillée par des images parasites qui sont des *échos* de l'image de base. De la même manière, le son est parasité par quatre échos qui peuvent être plus ou moins rapprochés du signal de base. Ces signaux parasites rendent la réception très médiocre, voire inintelligible. Un filtrage en fréquence ne suffit donc plus dès lors que moins de 9 canaux par chaîne sont utilisés.

Il est donc maintenant clair que si l'on cherche à diminuer le nombre de canaux par chaîne, il faudra trouver un traitement complémentaire du filtrage en fréquence. En particulier, l'étude précédente sur la propagation à 4 canaux montre que le problème qui se pose ressemble fortement à un problème de séparation de signaux.

2.2 Séparation de signaux monofréquence

2.2.1 Position du problème

On se placera à présent dans le contexte d'une transmission numérique[17]. Si la différence de trajet entre les chemins provenant des différents émetteurs est suffisante, on peut considérer qu'à la réception, le signal obtenu est un mélange instantané de signaux indépendants les uns des autres. Cette hypothèse paraît d'autant plus plausible que le codage de transmission est en général supposé rendre les symboles indépendants. En effet, plus un signal se rapproche d'un bruit blanc, et plus son spectre est plat, d'où une meilleure occupation de la bande de fréquence. D'un autre côté, un bruit blanc est indépendant de lui-même, dès lors qu'il y a un retard non nul entre les deux occurences considérées.

Pour exposer (puis résoudre) le problème, nous nous intéresserons plus précisément au cas de *deux* émetteurs utilisant la même fréquence porteuse, tout en sachant que par la suite il est toujours possible d'étendre le cas à trois émetteurs ou plus.

Le problème se pose comme suit: un émetteur E_1 émet vers le récepteur R_1 (l'antenne de réception du foyer considéré). Un autre émetteur E_2 se trouve dans le voisinage et émet sur la même fréquence. L'antenne R_1 est dirigée vers E_1, mais son diagramme étant une cardioïde, on récupère quand même sur R_1 une partie du signal émis par E_2. La figure (14) représente schématiquement le problème. Si on

[17]La modulation numérique est appelée à remplacer la modulation analogique dans un proche avenir, notamment pour la transmission des signaux TV.

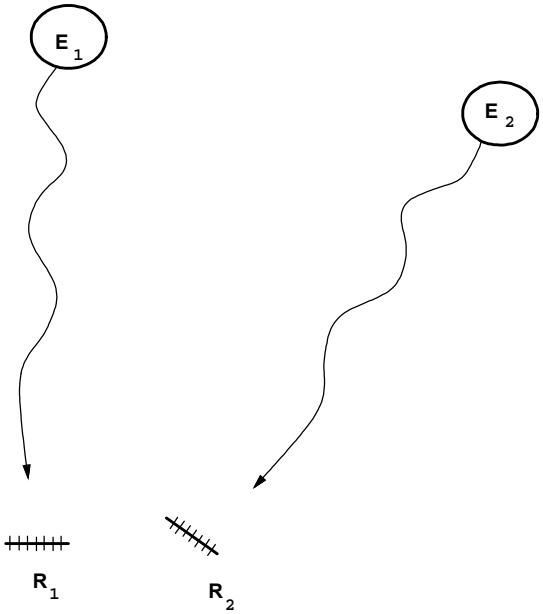

Figure 14: *Représentation symbolique du problème*

ne fait rien, c'est-à-dire si le seul traitement utilisé pour séparer toutes les ondes qui arrivent sur l'antenne R_1, est un filtrage passe bande, alors on ne récupère sur R_1 qu'un mélange (probablement initelligible) des deux signaux, ce qui rend la réception des images très médiocre.

Pour résoudre ce problème, on envisage de placer une seconde antenne R_2 à proximité de R_1, cette antenne étant dirigée vers l'émetteur parasite E_2, et de réaliser une séparation aveugle de mélanges de signaux selon une approche voisine de celle qui est proposée dans [4], mais adaptée ici à des mélanges de signaux complexes, comme ceux évoqués au chapitre précédent.

Remarque: Pour mettre en valeur l'approche Séparation Aveugle de Sources, nous n'exploiterons ici que les hypothèses d'indépendance statistique des signaux traités. Il serait par la suite possible, si le système proposé devait être utilisé de manière extensive, d'exploiter le caractère spécifique des signaux émis. En effet, l'exploitation des connaissances a priori dont on dispose devrait permettre de particulariser davantage la méthode pour cette application. On peut par exemple savoir que l'on a des paquets de 512 points et que tous les 16 paquets on émet un paquet test.

2.2.2 Modélisation du problème

On notera x_1 et x_2 les signaux complexes émis par E_1 et E_2, et y_1 et y_2 les signaux

complexes reçus par R_1 et R_2. Le mélange peut être modélisé par une transformation linéaire:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

En pratique, les ordres de grandeur sont les suivants:

- distance (E_1, E_2): 30 km
- distance (R_1, R_2): quelques mètres
- longueur d'onde: 50 cm (on est à environ 400 MHz)
- durée d'un échantillon: $\frac{1}{8}\mu s$ $(= 30m = 60\lambda)$
- type de modulation: MAQ 64
  (niveaux $-7, -5, -3, -1, 1, 3, 5, 7$ sur les parties réelles et imaginaire). Le codage est fait de telle sorte que les 64 états soient équiprobables (en conséquence, la partie réelle est indépendante de la partie imaginaire).

Nous désignerons par z_R et z_I les parties respectivement réelle et imaginaire d'un nombre complexe z. Le bruit thermique est quasi-nul (6 ou 7 dB en dessous de la diaphonie objectif). Il est donc négligé dans le modèle mathématique. Enfin, l'atténuation due à la directivité des antennes est de l'ordre de un cinquième.

2.2.3 Génération des signaux

Pour réaliser des tests, nous avons généré des signaux de la manière suivante:

1. $t = 0$
2. tirage de 4 variables aléatoires x_{1R}, x_{1I}, x_{2R} et x_{2I} selon une loi uniforme dans l'ensemble $\{-7, -5, -3, -1, 1, 3, 5, 7\}$
3. Calcul du mélange:

   $$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 1 & \frac{1}{5}e^{-j\frac{\pi}{4}} \\ \frac{1}{5}e^{j\frac{\pi}{6}} & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

   Cette matrice de mélange a été choisie pour respecter l'ordre de grandeur de l'atténuation, c'est-à-dire 1/5
4. $t = t + 1$
5. si $t < N$ retourner en (1).

La variable N représente le nombre d'échantillons dont on dispose. Il est clair que lorsque ce nombre croît, l'indépendance statistique des sources originelles est de mieux en mieux réalisée. Par suite, les estimations obtenues par l'algorithme de Séparation Aveugle sont d'autant plus proches des sources originales.

2.2.4 Algorithme de séparation

Puisque d'après nos hypothèses le mélange est linéaire, il est possible de réaliser la séparation par un séparateur linéaire. La séparation est réalisée par une matrice complexe

$$S = \begin{pmatrix} e & f \\ g & h \end{pmatrix}$$

dont les coefficients sont adaptés de manière à minimiser une mesure de dépendance construite sur les sorties du séparateur. La figure (15) représente le dispositif. La

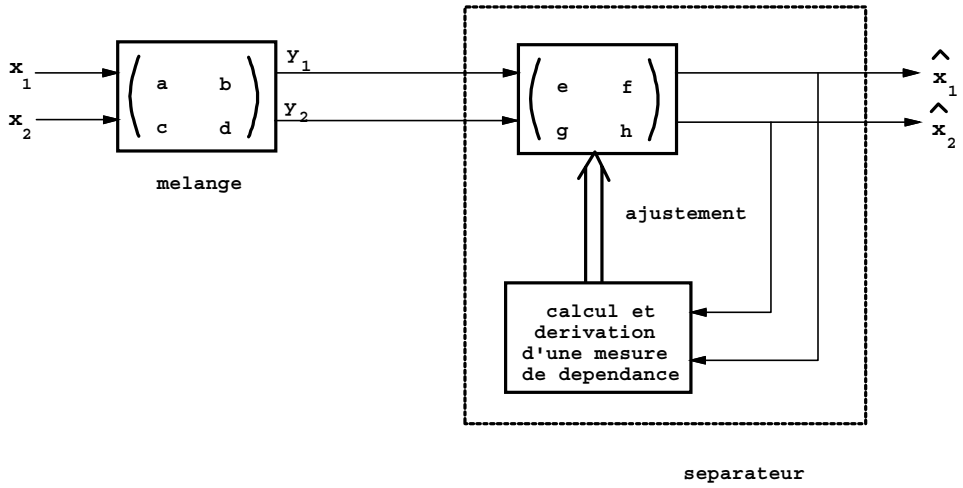

Figure 15: *Schéma de principe du séparateur utilisé*

mesure de dépendance utilisée est la mesure de Burel [4] adaptée à un mélange de signaux complexes. Nous sommes en présence d'un mélange de deux signaux complexes indépendants entre eux, et dont les parties réelles et imaginaires sont elles-mêmes indépendantes entre elles. Ainsi, ce problème peut se considérer comme un mélange de quatre signaux réels indépendants. La mesure de dépendance de Burel a été détaillée au Chapitre précédent, Section 6. Cette mesure de dépendance est un réel positif qui s'annule lorsque l'on a:

$$p(\hat{x}_{1R}, \hat{x}_{1I}, \hat{x}_{2R}, \hat{x}_{2I}) = p(\hat{x}_{1R})p(\hat{x}_{1I})p(\hat{x}_{2R})p(\hat{x}_{2I})$$

c'est-à-dire lorsque l'indépendance statistique est obtenue (p représente une densité de probabilité).

Remarque: Si la mesure de dépendance était utilisée telle quelle, la solution obtenue en sortie ne serait pas une estimation des signaux originaux: la solution obtenue serait un signal identiquement nul. Il faut donc, comme dans l'algorithme original de Burel, ajouter à la mesure de dépendance m_d un critère garantissant la non-trivialité des sorties:

$$C_d = m_d + \frac{1}{2}\left\{\sum_{i=1}^{4} \mid E\{S_i\} \mid^2 + \sum_{i=1}^{4} \mid E\{{S_i}^2\} - \sigma^2 \mid^2\right\}$$

où σ^2 est la variance supposée des sources originales et S_i représente les variables réelles x_{1R}, x_{1I}, x_{2R} et x_{2I} respectivement, pour $i \in [1,4]$. Rappelons que les sources considérées ici sont les parties réelles et imaginaires de x_1 et x_2. Ces variables sont des variables aléatoires uniformes appartenant à l'ensemble $\{-7,-5,-3,-1,1,3,5,7\}$. La variance commune de ces variables est donc

$$\sigma^2 = \frac{1}{4}\left(1^2 + 3^2 + 5^2 + 7^2\right) = 21$$

De cette manière, on s'assure que la solution trouvée ne sera pas triviale, puisque la minimisation de la contrainte ajoutée $\sum_{i=1}^{4} \mid E\{{S_i}^2\} - \sigma^2 \mid^2$ équivaut à

$$\forall i \in [1,4], \qquad E\{S_i^2\} = \sigma^2$$

Ceci implique entre autres que l'estimation en sortie sera de variance non nulle, ôtant ainsi le risque d'obtenir une estimation identiquement nulle. De plus, la variance choisie est la variance supposée des sources, ce qui permet d'obtenir, après convergence de l'algorithme, les sources originales modulo une permutation et éventuellement une inversion de signe. De même, la première contrainte équivaut de manière évidente à forcer les sorties à être de moyenne nulle (puisque les sources originales elles-mêmes sont de moyenne nulle). Bien sûr, dans le cas où nous disposerions de plus d'informations, comme par exemple la connaissance des symboles contenus dans le paquet test, l'ambiguité relative à l'inversion de signe disparaîtrait.

En ayant ajouté deux contraintes supplémentaires à la mesure de dépendance, nous avons à la fois ôté des ambiguités et rapproché fortement la forme des estimations finales de celle des sources originales, mais nous avons surtout assuré que notre estimation ne sera pas une solution triviale.

Une fois ce critère de dépendance clairement défini, il suffit de lancer l'algorithme

de décroissance par rétropropagation de l'erreur, sur le réseau à deux couches (figure (15)) Ce réseau possède quatre poids complexes e, f, g, h, qui seront réajustés constamment lors de l'apprentissage. Rappelons qu'ici, contrairement à un réseau classique, l'apprentissage ne se fait pas *avant* les expérimentations, mais *pendant*. Les N échantillons sont passés dans le réseau, et le critère de dépendance correspondant aux sorties associées est calculé, puis son gradient est rétropropagé sur les poids de manière à les modifier dans le sens d'une diminution de ce critère de sortie. L'opération entière est alors réitérée jusqu'à convergence de l'algorithme.

2.2.5 Résultats expérimentaux

La qualité de la séparation obtenue va dépendre du nombre d'échantillons pris en compte pour estimer la mesure de dépendance et faire converger le séparateur. Dans le cas limite où l'on utiliserait une infinité d'échantillons, on obtiendrait[18] $\hat{x}_1 = x_1$ et $\hat{x}_2 = x_2$, car l'estimation de la mesure de dépendance serait alors parfaite.

Le tableau suivant représente, en fonction du nombre d'échantillons pris en compte:

- dép_sources: la valeur de la mesure de dépendance, estimée sur les sources elles-mêmes
- dép_initiale: la mesure de dépendance estimée sur les sorties du séparateur lorsque celui-ci est dans son état initial
- dép_finale: la même mesure, lorsque le séparateur a convergé
- SNR($\hat{x}_{1R}$): le rapport signal-à-bruit sur $\hat{x}_{1R}$:

$$10 \log \frac{var(x_{1R})}{var(\hat{x}_{1R} - x_{1R})}$$

- SNR($\hat{x}_{1I}$): le rapport signal-à-bruit sur $\hat{x}_{1I}$
- $\max|\hat{x}_{1R} - x_{1R}|$: l'écart maximal obtenu sur l'estimation de la partie réelle
- $\max|\hat{x}_{1I} - x_{1I}|$: l'écart maximal obtenu sur l'estimation de la partie imaginaire
- % err: le pourcentage d'erreur après seuillage. Le seuillage consiste à remplacer $\hat{x}_1$ par la plus proche des 64 valeurs permises. Si $\hat{x}_1^{seuille}$ est différent de x_1, on compte une erreur.

[18] après correction de la permutation et de l'inversion de signe éventuelles

Les 5 dernières mesures sont calculées sur des échantillons nouveaux, n'ayant pas servi à l'estimation des coefficients du séparateur. Expérimentalement, on a toujours obtenu, à quelques centièmes de pourcent près, les mêmes valeurs du rapport signal-à-bruit et de l'erreur maximale, pour la partie réelle et pour la partie imaginaire. Aussi n'indiquons-nous ces valeurs que pour la partie réelle afin de ne pas surcharger le tableau.

nb_ech	100	1000	10.000
dep_sources	0.280	0.0378	0.00488
dep_initiale	5.80	4.00	4.02
dep_finale	0.204	0.0342	0.00268
$\mathrm{SNR}(\hat{x}_{1R})$	43.7	65.1	90.6
$\max \lvert\hat{x}_{1R} - x_{1R}\rvert$	1.38	0.434	0.130
% err	7.91	0	0

Les figures (16) à (17) montrent la constellation de points $\hat{x}_1$ obtenue en sortie

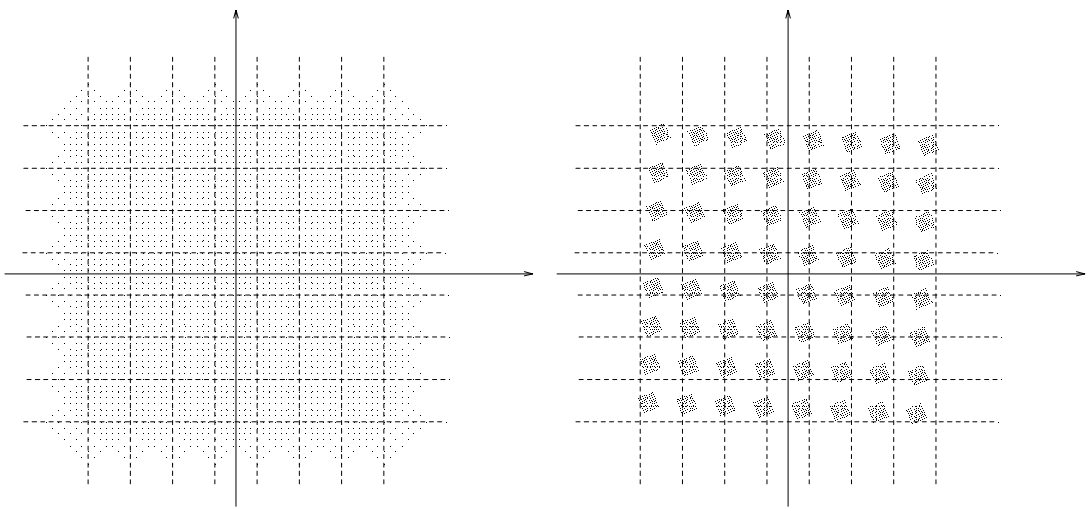

Figure 16: *Constellation de points obtenue (a) lorsque le séparateur n'est pas utilisé, (b) en sortie du séparateur après convergence sur 100 échantillons*

du séparateur selon le nombre d'échantillons pris en compte. Les intersections des droites en pointillés correspondent chacune à la position correcte d'un symbole de la modulation MAQ64. La figure (16a) correspond à la constellation obtenue si l'on ne met pas en œuvre un séparateur. Comme on le voit, il est illusoire de vouloir décoder les signaux obtenus sans utiliser un algorithme de séparation spécifique: le filtrage en fréquence actuellement utilisé (en relation avec le sytème à 9 canaux) ne suffit plus.

La durée d'une simulation sur Sun (Sparc 1) est assez importante. En secondes, elle est de l'ordre de 2 fois le nombre N d'échantillons utilisé. Toutefois, dans un contexte d'utilisation extensive, une reprogrammation spécialement adaptée à

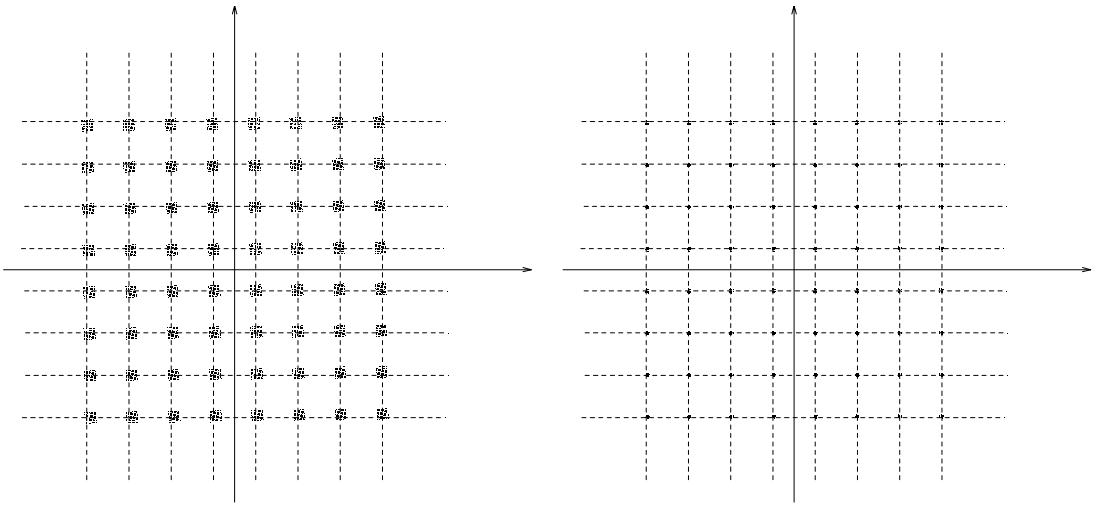

Figure 17: *Constellation de points obtenue en sortie du séparateur (a) après convergence sur 1000 échantillons, (b) après convergence sur 10000 échantillons*

cette application ainsi que l'utilisation d'une machine plus performante devraient permettre une gain de plusieurs ordres de grandeur.

2.2.6 Discussion

Le tableau confirme les prévisions de la théorie: plus on prend d'échantillons en compte, meilleure sera la séparation. Si l'on effectue la séparation sur 100 échantillons seulement, le maximum de l'écart entre $\hat{x}_{1R}$ et x_{1R} peut être supérieur à 1. Or l'écart minimal admissible entre deux symboles est de 2 (puisque dans la modulation MAQ64, les symboles sont les entiers impairs situés entre -7 et 7). De ce fait, pour être sûr de ne commettre aucune erreur d'estimation des symboles, il faudrait un écart maximal strictement inférieur à la moitié de l'intervalle séparant deux symboles successifs, c'est-à-dire une écart maximal de 1. Ainsi, pour seulement $N = 100$ échantillons, on remarque que l'écart maximal est 1.38, ce qui signifie que des erreurs sont commises. En effet, le tableau indique un taux d'erreur de 7.91% pour $N = 100$ échantillons.

Dès que l'on passe à 1000 échantillons, l'écart maximal entre la valeur estimée et la valeur émise tombe à 0.434, c'est-à-dire moins de la moitié de l'écart maximal autorisé. Il n'y a déjà plus de risque d'erreur d'estimation. Avec 10000 échantillons, on obtient un rapport signal-à-bruit supérieur à 90dB; l'écart maximal entre la valeur estimée et la valeur émise se réduit à 0.13, et le pourcentage d'erreur est là-aussi nul. On remarque d'ailleurs que sur la figure, les points sont presque parfaitement situés aux intersections des droites qui matérialisent la constellation théorique.

Les valeurs de la mesure de dépendance permettent de comprendre pourquoi le nombre d'échantillons est le paramètre qui limite la précision. En effet, plus on prend d'échantillons, plus on s'approche de l'hypothèse à la base de la méthode, c'est-à-dire une indépendance totale entre les sources. Il est d'ailleurs amusant de constater que la méthode de séparation sépare si bien les sources de manière à les rendre indépendantes, que la mesure de dépendance sur les estimations est toujours plus faible que la mesure sur les sources originales. En fait, ceci n'a rien d'étonnant: le nombre d'échantillons étant fini, la mesure de dépendance sur les sources originales reste assez élevée, et le séparateur arrive à trouver une meilleure solution que celle correspondant aux sources originales.

De la même façon, les figures (16) à (17) permettent également de constater l'amélioration de la séparation en fonction du nombre d'échantillons.

2.3 Séparation de signaux monofréquence convolués

2.3.1 Une nouvelle formulation du problème

Les hypothèses utilisées jusqu'ici supposaient que le mélange des deux signaux reçus sur l'antenne R_1 (et sur l'antenne R_2) était un mélange linéaire. En réalité, cette hypothèse est quelque peu simplificatrice: en présence de multitrajets dûs surtout à l'environnement proche (jusqu'à quelques km) des émetteurs, aussi bien qu'à l'environnement proche des récepteurs, tout se passe comme si les signaux émis étaient convolués puis mélangés (voir figure (18)).

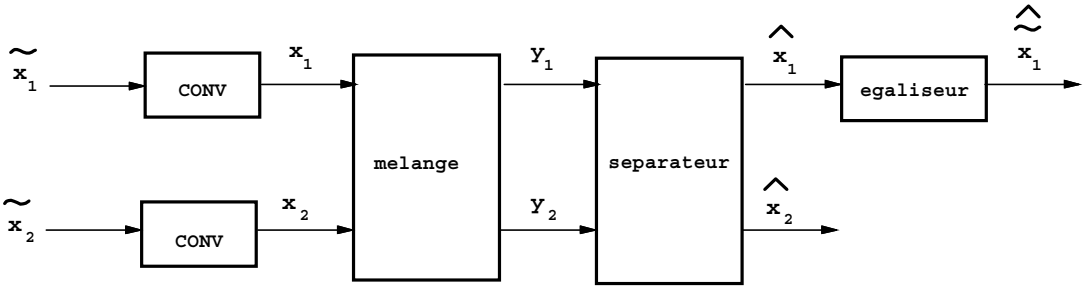

Figure 18: *Synoptique du problème en présence de multitrajets*

Si on appelle maintenant $\tilde{x}_i$ les signaux originaux, et x_i les signaux convolués, on a, d'après nos nouvelles hypothèses:

$$\begin{cases} x_1(t) &= \tilde{x}_1(t) + h_{11}\tilde{x}_1(t-1) + ... + h_{1n}\tilde{x}_1(t-n) \\ x_2(t) &= \tilde{x}_2(t) + h_{21}\tilde{x}_2(t-1) + ... + h_{2n}\tilde{x}_2(t-n) \end{cases}$$

si la convolution est d'ordre n. Le mélange des nouveaux "signaux" $x_1(t)$ et $x_2(t)$ se fait alors comme auparavant, c'est-à-dire de façon linéaire:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

Pour retrouver les signaux $\tilde{x}_i$ originaux, il faut procéder dans l'ordre inverse, comme indiqué sur la figure (18). Le rôle du séparateur est d'estimer les sorties des convolueurs. La déconvolution est le rôle de l'égaliseur.

Remarque: comme les environnements proches des récepteurs (éloignés de quelques mètres l'un de l'autre) sont les mêmes, les convolutions peuvent être semblables.

2.3.2 Une nouvelle formulation de la solution

Dans le cas d'un mélange linéaire simple, la solution utilisée se base sur l'indépendance totale des parties rélles et des parties imaginaires des sources. Dans le cas qui nous concerne ici, la séparation ne s'applique plus aux sources originelles $\tilde{x}_i$, mais à une convolution x_i de ces sources. Cette convolution implique que les parties imaginaires des signaux x_i ne sont plus indépendantes de leurs parties réelles:

$$\begin{aligned} p(x_{1R}, x_{1I}) &\neq p(x_{1R})p(x_{1I}) \\ p(x_{2R}, x_{2I}) &\neq p(x_{2R})p(x_{2I}) \end{aligned}$$

En conséquence il n'est plus possible de minimiser une mesure de dépendance sur le quadruplet $(\hat{x}_{1R}, \hat{x}_{1I}, \hat{x}_{2R}, \hat{x}_{2I})$, comme auparavant. Pour résoudre ce problème, nous proposons de minimiser une somme de mesures de dépendance sur les couples:

$$\begin{gathered} (\hat{x}_{1R}, \hat{x}_{2R}) \\ (\hat{x}_{1R}, \hat{x}_{2I}) \\ (\hat{x}_{1I}, \hat{x}_{2R}) \\ (\hat{x}_{1I}, \hat{x}_{2I}) \end{gathered}$$

Dans le cas d'un mélange linéaire simple, nous avions ajouté à la mesure de dépendance des contraintes sur la moyenne et la variance des parties réelles et imaginaires des estimations. En effet, nous avons vu que les parties réelles et les

parties imaginaires de signaux MAQ64 ont une moyenne nulle et une variance $\sigma^2 = 21$. Notons ici que nous ne pouvons plus imposer de contraintes sur la moyenne ou la variance des parties rélles et imaginaires des estimations, puisque nous ne connaissons pas les coefficients du convolueur. Nous ne pouvons donc pas savoir comment se transforment les moyennes et les variances des signaux obtenus en sortie de convolueur. Dans ces conditions, x_1 ne pourra plus être estimé qu'*à un facteur multiplicatif près*. Cependant, ce facteur multiplicatif peut toujours être reporté symboliquement dans le convolueur, et annihilé par l'égaliseur.

2.3.3 Expérimentations et discussion

Pour les expérimentations, nous avons simulé l'effet des multitrajets par des filtres d'ordre 2:

$$\begin{cases} x_1(t) &= \tilde{x}_1(t) + h_{11}\tilde{x}_1(t-1) + h_{12}\tilde{x}_1(t-2) \\ x_2(t) &= \tilde{x}_2(t) + h_{21}\tilde{x}_2(t-1) + h_{22}\tilde{x}_2(t-2) \end{cases}$$

$$\begin{array}{rll} \text{avec} & h_{11} = 0.3e^{j\frac{5\pi}{3}} & h_{11} = 0.07e^{j\frac{\pi}{6}} \\ & h_{11} = 0.2e^{j\frac{7\pi}{6}} & h_{11} = 0.15e^{-j\frac{\pi}{4}} \end{array}$$

L'étude a été réalisée en faisant converger le séparateur sur 10.000 échantillons. Le tableau suivant donne les résultats obtenus en sortie du séparateur:

Résultats en sortie du séparateur (signaux convolués):

nb_ech	10.000
dep_sources	0.00128
dep_initiale	0.833
dep_finale	0.00127
SNR($\hat{x}_{1R}$)	58.1
SNR($\hat{x}_{1I}$)	57.5
max $\lvert\hat{x}_{1R} - x_{1R}\rvert$	0.679
max $\lvert\hat{x}_{1I} - x_{1I}\rvert$	0.713

Il est toutefois difficile de juger de la qualité de la séparation au vu de ces mesures. C'est pourquoi nous donnons également des mesures sur $\hat{x}_1$ déconvolué (que nous noterons $\hat{\tilde{x}}_1$). Pour réaliser la déconvolution, nous avons appliqué le filtre inverse du convolueur:

$$\begin{array}{rl} \text{convolution:} & 1 + h_{11}z^{-1} + h_{12}z^{-2} \\ \text{déconvolution:} & \dfrac{1}{1 + h_{11}z^{-1} + h_{12}z^{-2}} \end{array}$$

donc $\hat{\tilde{x}}_1(t) = \hat{x}_1(t) - h_{11}\hat{\tilde{x}}_1(t-1) - h_{12}\hat{\tilde{x}}_1(t-2)$. C'est le filtre vers lequel doit converger un égaliseur, à un facteur multiplicatif près.

Résultats en sortie de l'égaliseur (signaux déconvolués):

nb_ech	10.000
SNR($\hat{\tilde{x}}_{1R}$)	57.8
SNR($\hat{\tilde{x}}_{1I}$)	57.6
max $\lvert\hat{\tilde{x}}_{1R} - \tilde{x}_{1R}\rvert$	0.597
max $\lvert\hat{\tilde{x}}_{1I} - \tilde{x}_{1I}\rvert$	0.565
% err	0.0

On obtient donc, après convergence du séparateur sur 10.000 échantillons, un rapport signal-à-bruit supérieur à 57 dB et un écart maximal entre le signal original et son estimation de 0.6. On ne dépasse donc jamais l'écart limite de 1: il n'y a aucune erreur d'estimation. La figure (19) représente la constellation obtenue.

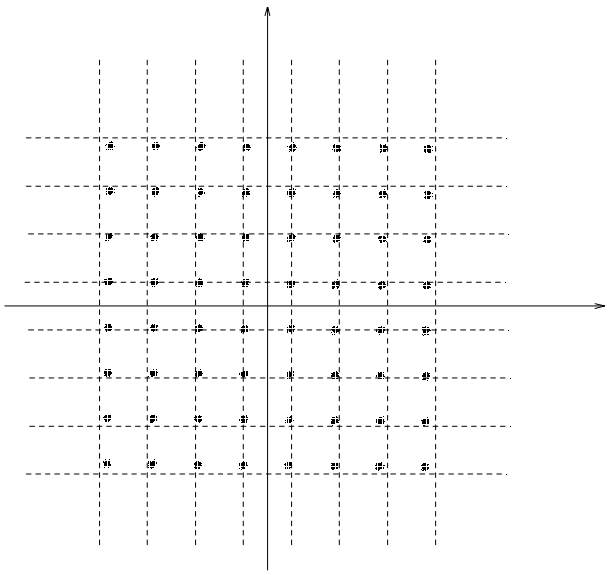

Figure 19: *Constellation de points obtenue après séparation et déconvolution. Le séparateur a convergé sur 10000 échantillons*

Remarque:

L'examen de la figure (19) montre qu'il existe une dilatation globale entre la grille et la constellation obtenue. Ceci est dû à l'incertitude sur le facteur multiplicatif, que nous avions mentionnée précédemment. Ce facteur multiplicatif aurait normalement été inversé (c'est-à-dire annihilé) pour un véritable égaliseur, alors que nous n'avons employé ici que l'inverse du convolueur. Avec la compensation

de ce facteur multiplicatif, on obtiendrait un rapport signal-à-bruit supérieur, car la dispersion des groupes de points autour de leurs barycentres respectifs est assez faible.

2.4 Conclusion

Dans cette partie concernant une application de l'extension de l'algorithme de séparation aveugle de Burel à des signaux complexes, nous avons examiné divers modèles de mélanges de signaux monofréquence, et envisagé une solution de séparation pour chacun d'eux; alors qu'aujourd'hui les chaînes de télévision mobilisent chacunes 9 canaux pour pouvoir être reçues dans des conditions convenables, nous avons imaginé comment pourrait être envisagé un système n'utilisant qu'un seul canal par chaîne.

Dans le premier modèle envisagé, le signal reçu est un mélange linéaire de signaux monofréquence. Pour résoudre ce problème (c'est-à-dire pour séparer les signaux et ne garder qu'une seule occurence du programme TV que l'on désire recevoir), il suffit de placer, dans le voisinage de son antenne TV, autant d'antennes qu'il existe de signaux parasites. En principe, le nombre de signaux parasite est limité, puisque plus les émetteurs sont éloignés, plus leurs signaux deviennent faibles à la réception. On peut penser qu'au maximum trois signaux parasites pourraient (éventuellement) troubler la réception. Les mélanges de signaux reçus par ces antennes sont alors séparés en mettant en œuvre un réseau de neurones à deux couches minimisant le critère de dépendance de Burel. Les expériences montrent que la solution apportée à ce problème convient parfaitement. Cependant, le modèle choisi est quelque peu simpliste, puisque l'on s'aperçoit qu'en réalité les échos des signaux émis provoquent des convolutions à la réception.

On envisage alors un modèle plus évolué, où des convolutions sur les signaux originaux peuvent survenir avant leur mélange. La résolution de ce nouveau problème se base sur l'utilisation de l'algorithme (modifié) de minimisation d'une erreur de dépendance sur les sorties d'un séparateur. En outre, les signaux obtenus sont alors déconvolués en les faisant passer dans un filtre correspondant à l'inverse du convolueur. A nouveau, les expérimentations montrent que la solution envisagée convient bien pour résoudre le problème posé.

En réalité, il est plus plausible de penser que les convolutions ne se font pas *avant* le mélange, mais plutôt de manière concomitante. Ce modèle est alors beaucoup plus difficile à évaluer, et le problème qu'il engendre est beaucoup plus difficile à résoudre, puisque le système basé sur le séparateur doit également mettre en œuvre des déconvolutions. Ce point précis fait d'ailleurs l'objet d'une autre thèse [6] qui a débuté récemment au laboratoire.

3 Séparation de voix humaines

3.1 Introduction

Le premier algorithme de séparation aveugle [9] [10] se base sur l'évidence que les cerveaux de grenouilles ne peuvent raisonnablement pas connaître, de manière innée, tous les coefficients permettant de séparer les informations en vitesse des informations en position provenant des muscles de leurs cuisses. C'est en partant de cette idée qu'a été mise au point la première méthode *adaptative* [9] permettant de retrouver les coefficients de mélanges instantanés. Par la suite, les algorithmes de séparation aveugle proposés par différents auteurs ont surtout été utilisés pour séparer des mélanges artificiels de signaux artificiels (cas d'école servant surtout à démontrer l'efficacité des algorithmes).

Une nouvelle extension des algorithmes de Séparation Aveugle a alors été imaginée: c'est le problème de la "cocktail party". Ce problème, parce qu'il fait appel à des signaux réels (donc forcément moins "parfaits" que des signaux obtenus par simulation), est par définition beaucoup plus difficile à résoudre que les problèmes "artificiels". En effet, dans un problème où les données ont été générées par simulation, on *connaît*, de manière évidente, le *modèle* de ces signaux. Au contraire, dans un problème issu du monde réel, le modèle des signaux est empirique: le modèle est une hypothèse que l'on tente de vérifier en essayant de résoudre le problème par une méthode se basant justement sur cette hypothèse.

Nous allons tout d'abord présenter le dispositif expérimental utilisé dans ce problème de la "cocktail party". Nous examinerons ensuite diverses méthodes de résolution de ce problème, basées chacune sur des hypothèses différentes.

3.2 Conditions expérimentales

Pour effectuer ces expériences, nous ne disposions pas de chambre anéchoïque, aussi ont-elles été réalisées dans des conditions de confort auditif très standard, à savoir dans une salle de conférence (d'environ 12m×12m) presque vide à part quelques chaises et un bureau. Nous verrons plus tard que cette configuration fut certainement à l'origine de nombreux échos. Le dispositif expérimental est présenté en figure (20). Deux personnes (deux hommes) sont placées à deux mètres des micros, à des

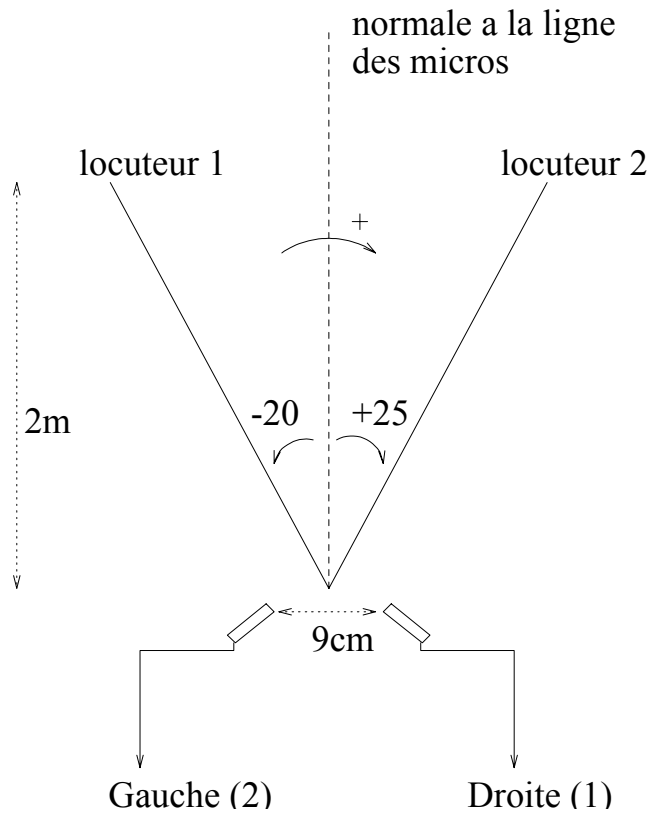

Figure 20: *Dispositif expérimental*

angles respectifs de -20^o et $+25^o$ par rapport à la perpendiculaire à la ligne des micros. Chacune de ces deux personnes parle en même temps (c'est le principe de la "cocktail party"), énonçant un texte différent l'une de l'autre. Les micros sont des micros ordinaires, de qualité moyenne, et d'après les caractéristiques du fabricant, leur diagramme de réception devrait ressembler à une cardioïde. Les micros sont écartés de 17cm en leur centre, et orientés l'un vers l'autre à 45^o par rapport à la normale de leur ligne. De ce fait, l'angle entre les deux micros est un angle droit. Dans cette configuration, la distance entre les extrémités des deux micros est de 9cm.

Les fichiers de son sont échantillonnés à une fréquence de 10kHz. Remarquons aussi que le dispositif d'enregistrement n'est pas très perfectionné, puisqu'en réalité, le signal émanant du micro de droite est échantillonné en alternance avec le signal émanant du micro de gauche. Par conséquent, puisque la période associée à la fréquence 10kHz est $\frac{1}{10^4} = 0.1$ms, le laps de temps séparant l'échantillonnage de la voie de droite de celui de la voie de gauche vaut une moité de période, c'est-à-dire 0.05ms. Les signaux recueillis par les capteurs sont approximativement centrés.

Une fois les conditions expérimentales établies, nous pouvons essayer de réaliser la séparation aveugle de signaux de parole.

3.3 Séparation simple de signaux de parole

Avec deux micros, nous disposons de deux mélanges: ces mélanges sont composés de signaux de parole (les deux personnes qui parlent) et, éventuellement de bruit. Tous les signaux sont formés de valeurs réelles (des pressions, en réalité). Sur la base d'un mélange linéaire simple sans échos, on peut écrire:

$$\begin{cases} e_1(t) &= a_{11}x_1(t) + a_{12}x_2(t) \\ e_2(t) &= a_{21}x_1(t) + a_{22}x_2(t) \end{cases}$$

De manière plus concise, cela équivaut à dire que le mélange instantané résulte de la matrice

$$M = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}$$

Sous ces hypothèses, la séparation peut être réalisée par un réseau de neurones à deux couches de 2 neurones chacune (voir figure (21)). Après convergence par

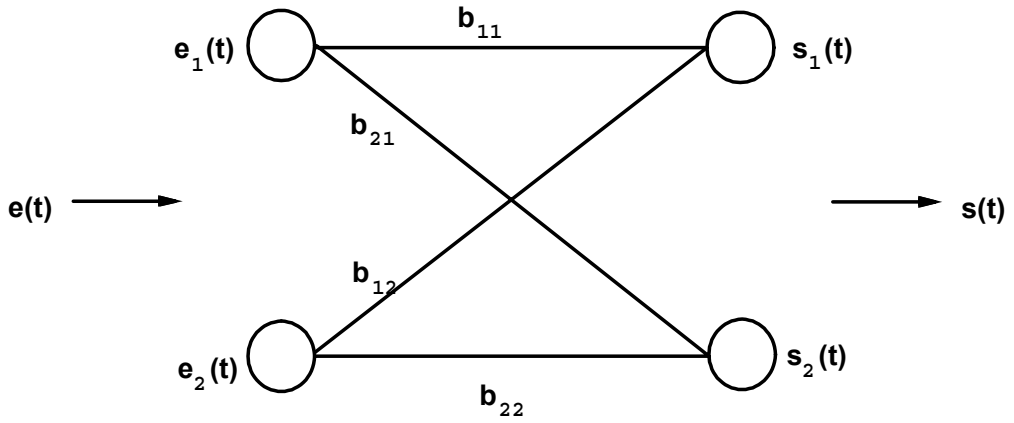

Figure 21: *Réseau permettant la séparation d'un mélange linéaire instantané de deux signaux*

minimisation du critère de Burel [4], les sorties $s_1(t)$ et $s_2(t)$ sont indépendantes, et les poids du réseau final correspondent au mélange inverse:

$$S = \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}$$

de telle sorte que SM soit égale à la matrice identité modulo une permutation et une dilatation.

Nous avons appliqué cette méthode au problème de la séparation de deux signaux de parole créés selon les conditions expérimentales mentionnées au paragraphe précédent. Après quelques itérations, le réseau converge, c'est-à-dire qu'il a rendu les sorties aussi indépendantes l'une de l'autre que possible. Cependant, lorsque l'on écoute (séparément) les signaux s_1 et s_2 (obtenus grâce aux coefficients trouvés par le réseau), on se rend compte que la séparation n'est pas bien réalisée: l'une de deux voix est légèrement atténuée, mais elle est toujours parfaitement audible.

Cet échec nous conduit à fournir deux types d'explications:

- soit l'algorithme de convergence utilise un critère inadapté au problème de séparation d'un mélange linéaire instantané de deux sources.
- soit le modèle représentant le mélange (linéaire instantané) est trop simpliste pour représenter correctement la réalité des choses.

La première explication est à rejeter pour plusieurs raisons:

- ce critère a été employé avec succès pour résoudre de nombreux problèmes mettant en œuvre des mélanges linéaires instantanés. Notons toutefois que les signaux traités dans ces problèmes étaient des signaux artificiels.
- dans le cas d'un mélange linéaire instantané, le critère minimisé correspond *exactement* à la réponse à apporter: il aspire, par sa minimisation, à l'indépendance des sorties (annulation de la mesure de dépendance $m_d(S)$); d'autre part il assure que les sorties ne soient pas identiquement nulles (contrainte sur la variance des sorties).
- si on examine en détail le nombre des équations implicitement contenues dans la minimisation du critère de Burel, on voit que ce nombre est largement suffisant pour pouvoir estimer les quatre coefficients du séparateur. En effet, pour un développement de la mesure de dépendance à l'ordre $K = 4$ (statistiques d'ordre 4 et moins), par exemple, on arrive à 8 équations[19] (voir Annexe A à ce sujet).

Après avoir montré de quelle manière on peut raisonnablement rejeter la première explication, voyons pourquoi la seconde explication (modèle trop simpliste) semble la bonne.

Tout d'abord, on a dit au départ que la salle d'enregistrement était presque vide, ce qui favorise les échos (rebondissement des ondes sur le sol, les murs et le

[19] 10 équations si les signaux ne sont pas centrés, mais seulement 8 équations si les signaux sont déjà centrés, comme ici

plafond avant d'arriver sur les micros). En présence d'échos, le mélange ne peut donc plus être considéré comme un mélange instantané. De ce fait, si le modèle jusqu'ici envisagé ne correspondait pas à la réalité, il n'apparaît pas surprenant que le séparateur basé sur ce modèle échoue dans sa tâche.

Cependant, nous allons le constater dans le paragraphe suivant, ce ne sont pas tant les échos qui perturbent le modèle du mélange, que des approximations sur certaines valeurs.

3.4 Séparation de signaux de parole convolués

3.4.1 Analyse du mélange reçu

Lorsque, au début de cette section consacrée à la séparation de voix, nous avons détaillé les conditions expérimentales de cette application, nous avons mentionné le fait que l'échantillonnage des signaux au niveau des deux micros ne se faisait pas de manière concomitante, mais de manière alternée, si bien que le laps de temps séparant la saisie de $e_1(t)$ de la saisie de $e_2(t)$ vaut une demi-période, soit 0.005ms: c'est un demi-échantillon de retard.

D'autre part, on a fait remarquer que les micros, bien qu'assez éloignés des locuteurs, étaient distants l'un de l'autre de 9cm. Pour un même locuteur, il existe donc une certaine différence de marche entre le moment où sa voix arrive sur le premier micro, et le moment où elle arrive sur le second micro. En assimilant le mélange à un mélange linéaire instantané, nous avons négligé cette différence de marche. Cette approximation est-elle valable?

Pour répondre à cette question, effectuons un petit calcul simple: pour une vitesse du son égale à 340m/s, la distance de 9cm séparant les deux micros est couverte en $0.09/340 = 0.265$ms. Pour une fréquence d'échantillonnage de 10kHz, ceci représente un peu moins de 3 échantillons.

En réalité, les locuteurs ne se trouvent pas dans le prolongement de la ligne des micros, mais obliquement à cette ligne. D'après la figure (20), on voit que le locuteur 2 se trouve à un angle de $+25^o$ par rapport à la normale à la ligne des micros, et le locuteur 1 à un angle de -20^o. Prenons la référence sur le capteur 1. Au niveau du capteur 2, la voix du locuteur 2 arrive avec un retard de $0.265 \sin 25^o = 0.11$ms, et la voix du locuteur 1 avec une avance de $0.265 \sin 20^o = 0.09$ms, soit au total un écart de 0.2ms entre les deux locuteurs, c'est-à-dire 2 échantillons.

Ce retard de 2 échantillons a été négligé dans l'hypothèse envisagée jusqu'ici du mélange instantané. Peut-on réellement considérer que le signal de parole varie peu sur une durée de 0.2ms? Considérons une composante à 1kHz du signal de parole. Entre $t = 0$ et $t = 0.2$ms, une sinusoïde à 1kHz passe de 0 à $\sin(2\pi(0.2)) = 0.39$, ce qui est loin d'être négligeable par rapport au maximum de la sinusoïde (à 1.0). L'hypothèse était donc fausse.

De ce fait, l'hypothèse du mélange instantané ne tient plus, ce qui explique a posteriori la faillite du séparateur basé sur cette hypothèse d'instantanéité. Il est maintenant nécessaire d'affiner le modèle du mélange pour tenir compte des retards ou des avances de certains signaux sur d'autres.

3.4.2 Une nouvelle modélisation du problème

D'après la figure (20), on peut considérer que le capteur (micro) 1 est plutôt orienté vers le locuteur 1, et le capteur 2 est plutôt orienté vers le locuteur 2. De ce fait, on peut choisir par convention de modéliser le mélange reçu sur le capteur 1 comme une somme pondérée du signal référence (les paroles du locuteur 1) au temps t, et du signal correspondant au locuteur 2 à un temps $t + t_2$, où $t_2 \neq 0$.

De la même façon, on modélise le mélange reçu sur le capteur 2 comme la somme pondérée du signal référence (ici les paroles du locuteur 2) au temps t, et du signal correspondant au locuteur 1 à un temps $t + t_1$, où $t_1 \neq 0$. Sous forme mathématique, cela revient à:

$$\begin{cases} e_1(t) &= x_1(t) + a_{12}x_2(t + t_2) \\ e_2(t) &= a_{21}x_1(t + t_1) + x_2(t) \end{cases} \tag{11}$$

D'après la configuration géométrique de la figure (20), on voit que le signal émis par le locuteur 2 atteint le capteur 1 *avant* d'atteindre le capteur 2. On a donc, pour cette configuration géométrique donnée, $t_1 > 0$ et $t_2 > 0$.

Le système d'équations (11) peut se réécrire comme

$$\begin{cases} e_1(t) - a_{12}e_2(t + t_2) &= x_1(t) - a_{21}a_{12}x_1(t + t_1 + t_2) \\ -a_{21}e_1(t + t_1) + e_2(t) &= -a_{21}a_{12}x_2(t + t_1 + t_2) + x_2(t) \end{cases} \tag{12}$$

De cette façon, le terme du haut ne dépend plus que de la source x_1, et le terme du bas ne dépend plus que de la source x_2. L'Annexe B décrit des expériences sensorielles relatives à la perception des échos ou des retards par une oreille humaine. Il y est montré qu'une oreille humaine ne peut distinguer un signal de son écho que

lorsque le laps de temps séparant les deux occurences est supérieur à 70ms. Dans le cas précis qui nous intéresse, nous avons vu que t_1 et t_2 étaient de l'ordre de 0.1ms, c'est-à-dire très loin de la limite après laquelle une oreille humaine peut distinguer des échos. De ce fait, entendre le signal $x_1(t)$ pur ou entendre $x_1(t) - a_{21}a_{12}x_1(t+t_1+t_2)$ est tout à fait équivalent pour une oreille humaine normale.

Cette observation montre que le système d'équations (12) correspond au séparateur du mélange:

$$\left\{ \begin{array}{rcl} s_1(t) & = & e_1(t) + b_{12}e_2(t+t_2) \\ s_2(t) & = & b_{21}e_1(t+t_1) + e_2(t) \end{array} \right.$$

où $b_{12} = -a_{12}$ et $b_{21} = -a_{21}$. Sous forme matricielle, le mélange s'écrit:

$$\left[\begin{array}{c} e_1(t) \\ e_2(t) \end{array} \right] = \left(\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array} \right) \left[\begin{array}{c} x_1(t) \\ x_2(t) \end{array} \right] + \left(\begin{array}{cc} 0 & 0 \\ a_{21} & 0 \end{array} \right) \left[\begin{array}{c} x_1(t+t_1) \\ x_2(t+t_1) \end{array} \right] + \left(\begin{array}{cc} 0 & a_{12} \\ 0 & 0 \end{array} \right) \left[\begin{array}{c} x_1(t+t_2) \\ x_2(t+t_2) \end{array} \right]$$

et pour l'inverse:

$$\left[\begin{array}{c} s_1(t) \\ s_2(t) \end{array} \right] = \left(\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array} \right) \left[\begin{array}{c} e_1(t) \\ e_2(t) \end{array} \right] + \left(\begin{array}{cc} 0 & 0 \\ b_{21} & 0 \end{array} \right) \left[\begin{array}{c} e_1(t+t_1) \\ e_2(t+t_1) \end{array} \right] + \left(\begin{array}{cc} 0 & b_{12} \\ 0 & 0 \end{array} \right) \left[\begin{array}{c} e_1(t+t_2) \\ e_2(t+t_2) \end{array} \right]$$

Il faut toutefois nuancer cette modélisation par une remarque importante: en pratique, t_1 et t_2 sont inconnus. De plus, ils ne sont pas nécessairement entiers: en effet, les différences de marche entre les deux capteurs dépendant de leur éloignement, et de l'angle des deux locuteurs par rapport à la normale à la ligne des micros. Il serait donc tout à fait fortuit que t_1 et t_2 soient entiers tous les deux! De plus il ne faut pas oublier la demi-période de retard entre l'échantillonnage de la voie de droite et celui de la voie de gauche. Il est donc nécessaire d'interpoler. Par exemple, en notant T_2 l'entier immédiatement inférieur à t_2, et en réalisant une interpolation linéaire de $e_2(t+t_2)$, on a:

$$e_2(t+t_2) = (1-\alpha)e_2(t+T_2) + \alpha e_2(t+T_2+1)$$

d'où

$$s_1(t) = e_1(t) + b_{12}(1-\alpha)e_2(t+T_2) + b_{12}\alpha e_2(t+T_2+1)$$

Enfin, il est possible que d'autres configurations conduisent à des t_1 ou des t_2 négatifs. Toutes ces considérations poussent à adopter les modèles généraux suivants:

$$\left[\begin{array}{c} e_1(t) \\ e_2(t) \end{array} \right] = \left(\begin{array}{cc} a_{11}^{(0)} & 0 \\ 0 & a_{22}^{(0)} \end{array} \right) \left[\begin{array}{c} x_1(t) \\ x_2(t) \end{array} \right] + \sum_{\tau=-T}^{T} \left(\begin{array}{cc} 0 & a_{12}^{(\tau)} \\ a_{21}^{(\tau)} & 0 \end{array} \right) \left[\begin{array}{c} x_1(t+\tau) \\ x_2(t+\tau) \end{array} \right]$$

$$\begin{bmatrix} s_1(t) \\ s_2(t) \end{bmatrix} = \begin{pmatrix} b_{11}^{(0)} & 0 \\ 0 & b_{22}^{(0)} \end{pmatrix} \begin{bmatrix} e_1(t) \\ e_2(t) \end{bmatrix} + \sum_{\tau=-\beta T}^{\beta T} \begin{pmatrix} 0 & b_{12}^{(\tau)} \\ b_{21}^{(\tau)} & 0 \end{pmatrix} \begin{bmatrix} e_1(t+\tau) \\ e_2(t+\tau) \end{bmatrix}$$

Si cela s'avérait utile pour améliorer les performances, on s'est réservé la possibilité de prendre une extension temporelle supérieure pour le séparateur ($\beta > 1$).

Pour étendre encore le modèle au cas de n capteurs et n sources, on peut écrire matriciellement:

$$\boxed{\begin{array}{l} \vec{e}(t) = A_0\vec{x}(t) + \sum_{\tau=-T}^{T} A(\tau)\vec{x}(t+\tau) \\ \vec{s}(t) = B_0\vec{e}(t) + \sum_{\tau=-\beta T}^{\beta T} B(\tau)\vec{e}(t+\tau) \end{array}}$$

où

- A_0 et B_0 sont des matrices diagonales
- les $A(\tau)$ et les $B(\tau)$ sont des matrices à diagonales nulles.
- les vecteurs $\vec{e}(t)$, $\vec{s}(t)$ et $\vec{x}(t)$ représentent respectivement les n signaux reçus aux capteurs, les n sorties fournies par le séparateur, et les n sources indépendantes originales.

La séparation correspond à l'association de deux équations matricielles du type

$$\left\{ \begin{array}{rcl} \vec{e}(t) & = & \sum_u A(u)\vec{x}(t+u) \\ \vec{s}(t) & = & \sum_v B(v)\vec{e}(t+v) \end{array} \right.$$

où $A(u=0) = A_0 + A(\tau=0)$ et de même pour B. On a donc

$$\begin{array}{rcl} \vec{s}(t) & = & \sum_v B(v) \left\{ \sum_u A(u)\vec{x}(t+u+v) \right\} \\ & = & \sum_w \{B(v)A(w-v)\}\, \vec{x}(t+w) \\ & & \qquad \text{en posant} \quad w = u+v \\ & = & \sum_w F(w)\vec{x}(t+w) \end{array}$$

où

$$F(w) = \sum_v B(v)A(w-v)$$

Pour le cas à 2 capteurs et 2 sources, la séparation sera réalisée si toutes les matrices

$F(w)$ sont diagonales, ou bien si elles sont toutes anti-diagonales. Dans le cas à n capteurs et n sources, la séparation sera réalisée si *toutes* les matrices $F(w)$ sont égales à l'identité modulo une dilatation et modulo *la même* permutation:

$$\text{séparation} \iff \exists P_n, \forall w, \exists D(w), \quad F(w) = P_n D(w)$$

où P_n est une matrice de permutation de rang n, et les matrices $D(w)$ sont des matrices (diagonales) de dilatation:

$$D(w) = \begin{pmatrix} \lambda_1(w) & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & \lambda_n(w) \end{pmatrix}$$

3.5 Minimisation d'une fonction de coût

Maintenant que nous avons construit un modèle de mélange plus réaliste, et que nous avons mis au point un séparateur adapté à ce nouveau type de mélange, il s'agit de bâtir une fonction de coût à minimiser en sortie du séparateur.

Lorsque nous nous placions dans l'hypothèse d'un simple mélange linéaire instantané, le séparateur proposé était lui-même très simple, puisque se composant d'une matrice $n \times n$. Pour le cas plus particulier qui nous intéresse, c'est-à-dire la cocktail party à 2 locuteurs, le séparateur ne comportait que $2 \times 2 = 4$ coefficients à estimer. Aussi le critère de minimisation en sortie n'avait-il pas besoin d'être très élaboré. En effet, nous avions déjà mentionné que pour $K = 4$, la minimisation du critère de Burel[20] conduisait à la mise en œuvre implicite de 8 équations. Ainsi, 8 équations pour 4 inconnues étaient largement suffisantes.

En règle générale, un mélange linéaire instantané de n sources nécessite un séparateur équivalent à une matrice $n \times n$, soit n^2 coefficients. D'un autre côté, un mélange convolutif de n sources nécessite un séparateur B tel que:

$$\vec{s}(t) = B_0 \vec{e}(t) + \sum_{\tau=-\beta T}^{\beta T} B(\tau) \vec{e}(t+\tau)$$

où B_0 est diagonale et les matrices $B(\tau)$ ont leur diagonale nulle. Ainsi, pour un séparateur convolutif qui va jusqu'à l'ordre $T' = \beta T$, le nombre de coefficients nécessaires à la construction du séparateur B est:

- n coefficients pour B_0

[20] ce critère pour $K = 4$ équivaut à la Mesure de Dépendance de Burel à l'ordre 4 complétée par les contraintes sur la moyenne et la variance des sorties

- $n^2 - n$ coefficients pour chacune des $2T' + 1$ matrices $B(\tau)$

Finalement, le nombre total de coefficients d'un séparateur convolutif allant jusqu'à l'ordre T' est:

$$n + (2T' + 1)(n^2 - n) = (2T' + 1)n^2 - 2T'n$$

Ainsi, en passant d'un séparateur linéaire instantané à un séparateur linéaire convolutif à l'ordre T', nous sommes passés de n^2 à environ $(2T' + 1)n^2$ coefficients. Pour pouvoir faire face à cette augmentation importante du nombre de coefficients à estimer, il va falloir mettre au point une méthode permettant d'augmenter le nombre des équations implicites contenues dans la fonction de coût à minimiser.

Première observation sur les signaux de parole:

Notons tout d'abord que le signal de parole a la propriété de ne pas être stationnaire. Ainsi, les statistiques calculées sur un intervalle de temps t_1 peuvent

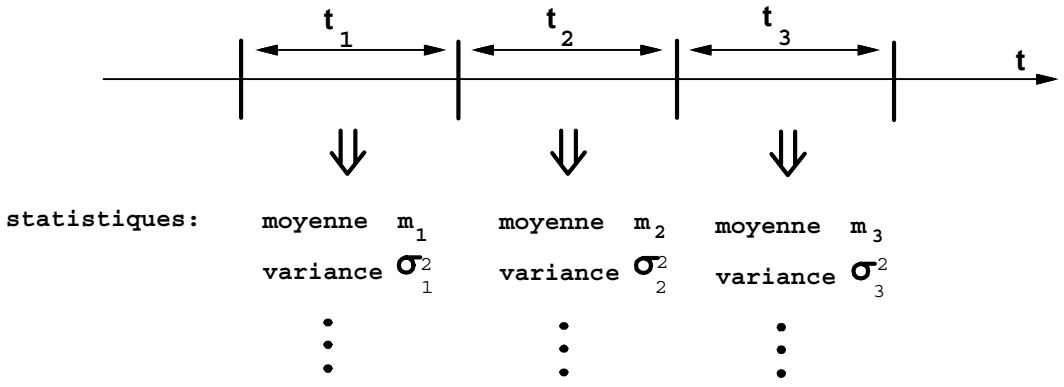

Figure 22: *Illustration de la non-stationnarité des signaux de parole*

être très différentes de celles calculées sur l'intervalle t_2 voisin, ou sur l'intervalle t_3 suivant (voir figure (22) pour illustration). On pourrait donc imaginer, au lieu de minimiser la mesure de dépendance $m_d(s)$ calculée sur un long intervalle de temps, minimiser la somme de *plusieurs* mesures de dépendance $m_d(s, g)$; ces mesures $m_d(s, g)$ seraient calculées chacune sur un intervalle de temps (un *groupe*) g différent. De cette façon, en remplaçant la mesure de dépendance habituelle $m_d(s)$ par

$$\sum_g m_d(s, g)$$

on multiplierait par N_g le nombre total d'équations implicites (avec N_g le nombre de groupes). En effet, puisque les statistiques calculées sur des intervalles de temps

différents sont différentes, les moments croisés des signaux sur des intervalles de temps différents sont également différents. Ainsi, deux mesures $m_d(s, g)$ calculées sur deux groupes différents produisent des équations (des contraintes, en fait) différentes, ce qui implique le doublement du nombre d'équations.

D'un autre côté, s'il est vrai que l'on gagne des équations en utilisant des mesures de dépendance calculées sur des groupes différents, il s'avère que l'on perd les équations correspondant aux contraintes sur la variance des sorties. En effet, lorsque l'on minimise le critère de dépendance de Burel sur un seul groupe, on utilise les contraintes suivantes:

- contrainte sur $m_d(s)$:

  $$p(s_1, s_2, ..., s_n) = p(s_1)p(s_2)...p(s_n)$$

  soit $L(K, n)$ équations

- contraintes sur les moyennes:

  $$\forall i, \quad E\{S_i\} = 0$$

  soit n équations. En réalité, ces contraintes sur la moyenne sont superflues, car compte tenu de la façon dont les signaux audio sont enregistrés, les signaux e_i ont une moyenne presque nulle. Par transition, les s_i ont aussi une moyenne nulle. Ces n équations correspondant aux contraintes sur les moyennes ne sont donc pas comptabilisées.

- contraintes sur les variances:

  $$\forall i, \quad E\{S_i^2\} = \sigma_i^2$$

  soit n équations.

Et on peut, par exemple, fixer les σ_i^2 à 1 si l'on désire obtenir des sorties unitaires. Les variances σ_i^2 assignées aux sorties s_i ne sont pas forcément égales aux variances des sources originales x_i. Dans le cas d'un séparateur de signaux monofréquences TV (modulation MAQ64) tel que celui étudié dans la section précédente, les statistiques de signaux étaient stationnaires et on avait calculé que le σ_i moyen était égal à 21.

Ici, dans le cas de signaux de parole, les sources x_i ont des variances inconnues donc il ne semble pas aberrant souhaiter forcer les variances des s_i à 1. Ces contraintes sur les variances fournissent n équations.

Par contre, puisque l'on travaille sur des signaux de parole, c'est-à-dire rappelons-le des signaux non-stationnaires, les statistiques des sources originales x_i *varient* en

fonction du groupe temporel étudié. En conséquence, les variances σ_i^2 des sorties s_i *dépendent* également du groupe considéré: $\sigma_i^2 = \sigma_i^2(g)$. De ce fait, s'il est vrai que l'on pouvait fixer arbitrairement les variances des s_i dans le cas d'un seul groupe, il n'en est plus de même dès que l'on prend en compte plusieurs groupes. En effet, même si les statistiques intergroupes sont différentes, elles portent sur les *mêmes* variables. Il y a donc des liens (latents, mais inconnus) entre les variances des s_i calculées sur des groupes différents. Il n'est donc plus possible d'utiliser les contraintes sur les variances des sorties, tout au plus pourrait-on garder les contraintes portant sur les variances sur *un seul* groupe parmi les N_g groupes. Comme il est absolument nécessaire d'empêcher le séparateur de converger vers la solution parasite consistant à fournir des sorties identiquement nulles, il faut, soit maintenir une contrainte de variance sur un groupe, soit fixer la matrice B_0 à l'identité.

Finalement, on voit que lorsque l'on passe de 1 groupe à N_g groupes, le nombre d'équations implicitement contenues dans le critère de Burel passe de

$$L(K, n) + n$$

à

$$N_g \times L(K, n) \ \ (+n)$$

Seconde observation sur les signaux de parole:

Lorsque deux locuteurs prononcent un discours différent, et même, cas extrême, lorsque deux locuteurs prononcent le même discours en décalant leurs paroles de quelques mots l'un de l'autre, on peut faire la constatation suivante: le signal $x_1(t)$ émis par le premier locuteur est indépendant du signal $x_2(t)$ émis par le second locuteur, et il est aussi indépendant de $x_2(t-1), x_2(t-2), ..., x_2(t+1), x_2(t+2), ...$ En réalité, tant que les deux discours ne seront pas les mêmes au même instant, tant que les voix ne seront pas les mêmes, il y aura indépendance entre x_1 et x_2, ainsi qu'entre tous les décalages possibles de ces signaux.

Cette remarque nous pousse à utiliser, en sus de la mesure de dépendance $m_d(s)$ basée sur l'indépendance entre $x_1(t)$ et $x_2(t)$ pour tous les t, des mesures de dépendance $m_d(s, \tau)$ basées sur l'indépendance entre $x_1(t)$ et $x_2(t-\tau)$. Pour chaque nouveau τ, on ajoute ainsi $L(K, n)$ équations à notre système.

Si on combine les mesures de dépendance calculées sur des groupes temporels différents avec les mesures de dépendance calculées sur des décalages temporels variables, on obtient une nouvelle mesure de dépendance:

$$m_d(s) = \sum_g \sum_\tau m_d(s, g, \tau)$$

qui multiplie le nombre d'équations par $N_g N_\tau$, où N_g est le nombre de groupes et N_τ le nombre de décalages.

Cette nouvelle mesure de dépendance fait ainsi appel à un nombre accru d'équations, et même pour un ordre de développement statistique K peu élevé, il semble désormais relativement aisé de disposer d'un nombre de contraintes suffisant en regard du nombre de coefficients du séparateur.

3.6 Etude de la fonction de coût pour le cas $n = 2$

Dans le cas $n = 2$, la mesure de dépendance $m_d(s, g, \tau)$ s'écrit

$$m_d(s, g, \tau) = \sum_{\alpha_1+\alpha_2 \leq K} \sum_{\beta_1+\beta_2 \leq K} G_{\alpha_1\alpha_2\beta_1\beta_2} M_{\alpha_1\alpha_2}(s, g, \tau) M_{\beta_1\beta_2}(s, g, \tau)$$

où les G sont calculés à l'aide des moments d'une gaussienne, exactement de la même façon que dans la mesure de dépendance "classique", alors que les M dépendent du groupe g et du décalage τ choisis:

$$M_{\alpha_1\alpha_2}(s, g, \tau) = R_{\alpha_1\alpha_2}(s, g, \tau) - R_{\alpha_1\vec{\theta}_1}(s, g) - R_{\alpha_2\vec{\theta}_2}(s, g)$$

où

$$\vec{\theta}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \text{et} \quad \vec{\theta}_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

et

$$\begin{cases} R_{\alpha_i\vec{\theta}_i}(s, g) & = & E_g\{S_i^{\alpha_i}\} \\ R_{\alpha_1\alpha_2}(s, g, \tau) & = & E_g\{S_1^{\alpha_1}(t) S_2^{\alpha_2}(t-\tau)\} \end{cases}$$

et E_g représente l'espérance sur le groupe g.

Classiquement, un développement de la mesure de dépendance à l'ordre K sur 2 variables apporte $\dfrac{K(K-1)}{2}$ équations, plus les n contraintes sur les moyennes et les n contraintes sur les variances. On a dit que les contraintes sur les moyennes n'étaient pas comptabilisées ici parce que superflues, ce qui fait un total de

$$\frac{K(K-1)}{2} + n$$

équations. Dans l'hypothèse de traitement de signaux de parole, on peut utiliser des mesures de dépendance portant sur plusieurs groupes et plusieurs décalages. Le nombre total d'équations est alors porté à

$$\frac{K(K-1)}{2} N_g N_\tau \quad (+n)$$

ce qui augmente considérablement la force des contraintes. Il est même possible, si on utilise suffisamment de groupes et de décalages temporels différents, de diminuer l'ordre K du développement de la mesure de dépendance.

3.6.1 Développement à l'ordre de dépendance $K = 2$

Fonction de coût

Pour $K = 2$, la mesure de dépendance se simplifie à l'extrême, et devient équivalente à un moment croisé d'ordre 2 (les moyenne sont nulles):

$$m_d(K=2) \quad \equiv \quad (E\{S_1 S_2\})^2$$

Le coût total en sortie est basé sur la minimisation de la mesure de dépendance sur plusieurs groupes et plusieurs décalages:

$$C = \frac{1}{2} \sum_g \sum_\tau (R_{s_1 s_2}(g,\tau))^2$$

où

$$\begin{aligned} R_{s_1 s_2}(g,\tau) &= E_g\{S_1(t) S_2(t-\tau)\} \\ &= \frac{1}{t_g} \sum_{t \in g} s_1(t) s_2(t-\tau) \end{aligned}$$

avec g l'indice du groupe et t_g la taille de ce groupe.

Séparateur utilisé

Rappelons le modèle du mélange:

$$\vec{e}(t) = \sum_v A(v) \vec{x}(t+v)$$

puisque le mélange est convolutif; le modèle du séparateur est

$$\vec{s}(t) = \sum_u B(u) \vec{e}(t+u) \tag{13}$$

Les coefficients du séparateur sont les $b_{ij}(u)$, où $(i,j) \in \{1,2\}^2$, et u varie sur l'intervalle de convolution des deux signaux au niveau des micros. Pour l'expérience considérée en tête de cette partie, on a vu que l'intervalle de convolution était de l'ordre de 2 échantillons (en plus ou en moins par rapport à la référence). On aura donc ici $u \in [-2,2]$, voire même $u \in [-3,3]$ si les coefficients obtenus par ce séparateur d'ordre convolutif 2 ne fournissaient pas des sorties suffisamment propres du point de vue auditif. On rappelle que les enregistrements ayant été faits directement sur le mélange, on ne peut juger du succès ou de la faillite d'un séparateur, qu'en utilisant l'oreille humaine pour tester le résultat obtenu. Le séparateur utilisé est présenté en figure (23).

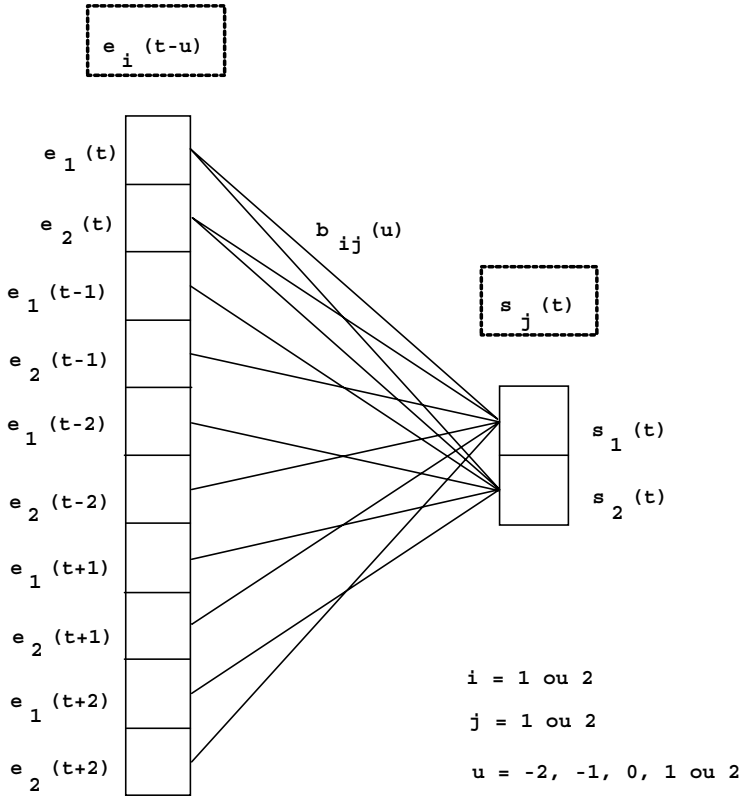

Figure 23: *Schéma du séparateur linéaire convolutif d'ordre 2 (ici, pour 2 signaux)*

Gradient de la fonction de coût

La convergence du séparateur s'opère par ajustement des poids $b_{ij}(u)$ dans le sens inverse du gradient de la fonction de coût en sortie. On a vu qu'à l'ordre $K = 2$, le coût s'écrit

$$C = \frac{1}{2}\sum_{g}\sum_{\tau}(R_{s_1 s_2}(g,\tau))^2$$

avec

$$\begin{aligned} R_{s_1 s_2}(g,\tau) &= E_g\{S_1(t)S_2(t-\tau)\} \\ &= \frac{1}{t_g}\sum_{t\in g} s_1(t)s_2(t-\tau) \end{aligned}$$

Le gradient du coût se calcule comme

$$\begin{aligned} \frac{\partial C}{\partial b_{ij}(u)} &= \sum_{g}\sum_{\tau} R_{s_1 s_2}(g,\tau)\frac{\partial R_{s_1 s_2}(g,\tau)}{\partial b_{ij}(u)} \\ &= \sum_{g}\sum_{\tau} R_{s_1 s_2}(g,\tau)\frac{1}{tg}\sum_{t\in g}\frac{\partial(s_1(t)s_2(t-\tau))}{\partial b_{ij}(u)} \\ &= \sum_{g}\sum_{\tau} R_{s_1 s_2}(g,\tau)\frac{1}{tg}\sum_{t\in g}\left[\frac{\partial s_1(t)}{\partial b_{ij}(u)}s_2(t-\tau) + \frac{\partial s_2(t-\tau)}{\partial b_{ij}(u)}s_1(t)\right] \end{aligned} \quad (14)$$

or:

$$s_i(t) = \sum_{v}\sum_{j} b_{ij}(v)e_j(t+v)$$

d'après l'équation (13), d'où:

$$\frac{\partial s_k(t)}{\partial b_{ij}(u)} = \delta_{ik}e_j(t+u)$$

En remplaçant dans (14), on obtient:

$$\begin{aligned} \frac{\partial C}{\partial b_{ij}(u)} &= \sum_{g}\sum_{\tau} R_{s_1 s_2}(g,\tau)\frac{1}{tg}\sum_{t\in g}[\delta_{i1}e_j(t+u)s_2(t-\tau) + \delta_{i2}e_j(t-\tau+u)s_1(t)] \\ &= \sum_{g}\sum_{\tau} R_{s_1 s_2}(g,\tau)\left[\delta_{i1}R_{e_j s_2}(g,u,-\tau) + \delta_{i2}R_{e_j s_1}(g,u-\tau,0)\right] \end{aligned} \quad (15)$$

en posant:

$$R_{e_j s_i}(g,a,b) = \frac{1}{tg}\sum_{t\in g} e_j(t+a)s_i(t+b)$$

3.7 Expérimentations

Deux types d'expériences ont été réalisées: le premier type est basé sur un mélange artificiel de signaux artificiels, le second sur un mélange réel de signaux réels.

3.7.1 Première expérience

Nous disposons de 18 groupes de 512 points, soit 9216 échantillons représentant deux signaux artificiels gaussiens x_1 et x_2. Les deux signaux ont une moyenne nulle, et la variance de x_2 est égale à 1. Le signal x_1 n'est pas stationnaire et sa variance suit une loi uniforme de limites 0.1 et 1.9:

$$\begin{cases} x_1(t) = \mathcal{N}(0, \sigma_1(g)) & \sigma_1^2(g) = \mathcal{U}([0.1, 1.9]) \\ x_2(t) = \mathcal{N}(0, 1) \end{cases}$$

Ces signaux sont mélangés artificiellement par:

$$\begin{bmatrix} e_1(t) \\ e_2(t) \end{bmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} + \begin{pmatrix} 0 & 0.6 \\ 0.3 & 0 \end{pmatrix} \begin{bmatrix} x_1(t+1) \\ x_2(t+1) \end{bmatrix}$$

c'est-à-dire un mélange convolutif à l'ordre 1. Sous forme compacte, ce mélange peut s'écrire

$$\begin{matrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0.6 & 0.3 & 0 \end{matrix}$$

avec la première ligne qui se réfère au temps $t-1$, la seconde au temps t, et la troisième au temps $t+1$. Un inverse de ce mélange est:

$$\begin{matrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & -0.6 & -0.3 & 0 \end{matrix}$$

On utilise le séparateur convolutif évoqué plus haut: la mesure de dépendance à l'ordre $K = 2$ est calculée sur $N_g = 18$ groupes de 512 points. De plus, $N_\tau = 3$ décalages sont employés dans ces calculs: $\tau = -1, \tau = 0$ et $\tau = +1$. En supposant le mélange convolutif à l'ordre 1 et en initialisant les coefficients du séparateur par l'identité:

$$\begin{matrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{matrix}$$

on obtient, au bout de 150 itérations, le mélange inverse suivant:

$$\begin{array}{cccc} 0 & 0.004 & -0.009 & 0 \\ 1 & 0.032 & -0.026 & 1 \\ 0 & -0.576 & -0.301 & 0 \end{array}$$

Puisqu'il s'agit d'un mélange artificiel, il est possible de vérifier, par le calcul, que les sorties $s_1(t)$ et $s_2(t)$ obtenues sont bien solutions. Il suffit pour cela de combiner les matrices représentant le séparateur et celles représentant le mélange. Une bonne séparation sera obtenue s'il reste peu (ou pas) de termes croisés.

Pour le mélange combiné au séparateur idéal, on obtiendrait:

$$\begin{array}{cccc} 0 & 0 & 0 & 0 & (t-2) \\ 0 & 0 & 0 & 0 & (t-1) \\ 1 & 0 & 0 & 1 & (t) \\ 0 & 0 & 0 & 0 & (t+1) \\ -0.18 & 0 & 0 & -0.18 & (t+2) \end{array}$$

Ce séparateur "idéal" sépare bien les signaux, mais il n'opère pas de déconvolution; ici, il a ajouté un écho artificiel aux signaux originaux:

$$\left\{ \begin{array}{rcl} s_1(t) & = & x_1(t) - 0.18\, x_1(t+2) \\ s_2(t) & = & x_2(t) - 0.18\, x_2(t+2) \end{array} \right.$$

Si nous désirions obtenir les signaux originaux purs, il faudrait, après ce séparateur, utiliser un déconvolueur. Cependant, en ce qui concerne les signaux de parole, l'oreille humaine ne peut distinguer des échos aussi rapprochés (2 échantillons = 0.2ms), ce qui autorise à garder cette solution en l'état. Remarquons également que la solution se compose du signal original $x_i(t)$ auquel on ajoute la pondération d'un écho $x_i(t+2)$ qui se situe dans le *futur*, ce qui ne serait pas possible avec un écho naturel.

Le mélange combiné au séparateur obtenu après 150 itérations donne:

$$\begin{array}{cccc} 0 & 0 & 0 & 0 & (t-2) \\ 0 & 0.004 & -0.009 & 0 & (t-1) \\ 1.001 & 0.032 & -0.026 & 0.994 & (t) \\ 0.009 & 0.023 & -0.001 & -0.015 & (t+1) \\ -0.173 & 0 & 0 & -0.181 & (t+2) \end{array}$$

On remarque là-aussi qu'il y a très peu de termes croisés (colonnes du milieu), et qu'ils sont de valeur absolue très faible. Cela montre que la séparation a été correctement effectuée, et que l'astuce consistant à multiplier le nombre de contraintes par des décalages en τ des moments croisés, et par des mesures de

dépendance calculées sur des groupes à statistiques différentes, permet de séparer tous les types de signaux, même des signaux gaussiens, pourvu que l'un au moins des signaux soit non-stationnaire.

3.7.2 Seconde expérience

La seconde expérience se base sur des signaux réels: les paroles des deux locuteurs évoqués au début du chapitre. Le mélange est également un mélange réel, et nous ne disposons pas des coefficients de ce mélange[21]. Comme nous l'avions mentionné au début du chapitre, le mélange se fait au niveau des micros: en raison de la position relative des locuteurs par rapport aux micros, en raison de l'écartement et de la disposition géométrique des micros, le mélange est convolutif. D'après les calculs effectués précédemment, l'ordre de la convolution du mélange semble se situer aux alentours de 2. Nous supposerons donc dans un premier temps que le mélange est un mélange convolutif d'ordre 2, et nous utiliserons pour la séparation le séparateur convolutif d'ordre 2 présenté en figure (23).

Nous disposons de 20 secondes d'enregistrement (voie de gauche et voie de droite), soit 200.000 échantillons. Nous n'utiliserons qu'environ 5 secondes de cet enregistrement, soit un peu plus de 50.000 points. Nous groupons ces échantillons par 512, soit 98 groupes au total. Nous calculons sur chaque groupe le coût de dépendance à l'ordre $K = 2$, construit sur les termes

$$R_{s_1 s_2}(\tau, g) = E_g\{S_1(t)S_2(t-\tau)\}$$

pour des τ allant de -2 à 2, soit $N_\tau = 5$ décalages en tout. Finalement, nous disposons de $N_g \times N_\tau = 5 \times 98 = 490$ contraintes implicites contenues dans la fonction de coût en sortie.

Pour l'initialisation des poids $b_{ij}(u)$ du séparateur, on prendra l'identité comme dans le cas précédent. Les valeurs initialisant les connexions neuronales sont donc:

$$\begin{array}{cccc} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{array}$$

C'est de cet état d'initialisation pour les poids $b_{ij}(u)$ que nous partons. Après 300 itérations utilisant le gradient développé à l'équation (15), nous obtenons les

[21] ce qui signifie que nous ne pourrons pas nous assurer de manière formelle que le séparateur a bien fonctionné. On pourra par contre le vérifier "à l'oreille"

coefficients suivants:

$$\begin{array}{cccc} 0 & -0.003 & 0.130 & 0 \\ 0 & -0.082 & -0.161 & 0 \\ 1 & 0.004 & -0.004 & 1 \\ 0 & -0.239 & -0.100 & 0 \\ 0 & -0.349 & -0.519 & 0 \end{array}$$

Comme le mélange est réel, nous ne disposons ni des coefficients exacts du mélange, ni, bien sûr, des coefficients exacts de l'inverse du mélange. Par contre, puisqu'il s'agit d'un mélange de signaux de parole, nous pouvons tester l'efficacité de la séparation grâce à l'oreille humaine. Nous avons donc écouté la sortie $s_1(t)$, puis la sortie $s_2(t)$, obtenues par séparation du mélange après ces 300 itérations.

On constate que la séparation, cette fois, est effective; alors que sur les hypothèses d'un mélange non convolutif nous avions obtenu des résultats médiocres (aucune séparation vraiment constatée), l'écoute successive des signaux $s_1(t)$ et $s_2(t)$ nous semble très bonne: on entend distinctement un seul locuteur, et si l'on tend l'oreille il est possible de distinguer un certain "bruit de fond", mais l'autre locuteur ne peut pas vraiment être discerné, même par bribes.

Ce séparateur qui ne remonte que jusqu'à une convolution d'ordre 2 convient donc parfaitement pour le problème auquel nous nous intéressions. De plus, la méthode mise en œuvre fonctionne avec le développement d'un coût de dépendance à l'ordre $K = 2$, ce qui est simple et peu coûteux en temps de calcul.

Enfin, la méthode a l'avantage de rester complètement adaptative, c'est-à-dire que même une fois que l'algorithme a convergé, il est toujours possible, de temps à autre, de relancer quelques itérations de descente du gradient en sortie, pour compenser les éventuelles modifications des coefficients du mélange, que ce soit à cause de l'échauffement des micros, d'un léger changement de position d'un des locuteurs, etc. De plus, on peut, si le besoin s'en faisait sentir, ajouter un ordre supplémentaire au modèle de la convolution. Par exemple, si les micros venaient à être déplacés, ou écartés, le mélange pourrait devenir un mélange convolutif d'ordre 3 ou plus; il suffirait alors d'ajouter au séparateur les entrées et les poids supplémentaires correspondant à cette modification de l'ordre de la convolution. Puis, partant des anciennes valeurs des coefficients estimés précédemment (et des nouveaux coefficients à zéro), il suffirait de relancer quelques itérations pour faire à nouveau converger le séparateur.

3.8 Conclusion

Nous avons effectué, dans des conditions d'environnement assez médiocres (salle non

anéchoïque, micros non professionnels, échantillonnage non simultané des micros), l'enregistrement de deux locuteurs prononçant chacun un discours différent. Deux micros ont été placés relativement loin des deux locuteurs pour cet enregistrement. D'après quelques calculs primaires, il semble que la position des locuteurs relativement aux micros, ainsi que l'écartement des 2 micros, rendent le mélange convolutif avec un retard d'environ 2 échantillons.

Nous avons tenté de séparer ce mélange sur la base d'une hypothèse de non convolution: il est alors impossible de séparer les deux locuteurs.

Nous avons ensuite supposé que le mélange était convolutif d'ordre 2. Cette hypothèse entraînant un surcroît de coefficients pour le séparateur, nous avons dû mettre au point une nouvelle version de la Mesure de Dépendance de Burel évoquée au Chapitre 2 Sections 6 et 7. Cette nouvelle mesure est basée d'une part sur le fait que les signaux de parole sont non stationnaires, et d'autre part sur le fait qu'un signal de parole est indépendant de toutes les versions décalées du signal de parole d'un autre locuteur. Ces deux assertions nous ont permis de construire une fonction de coût en sortie basée sur suffisamment de contraintes pour pouvoir résoudre le problème posé. Il est même possible, selon le nombre de nouvelles contraintes ainsi apportées, de réduire l'ordre K du développement de la mesure de dépendance.

Des expériences sur des signaux artificiels et sur des signaux réels ont été conduites avec succès, montrant l'efficacité de la méthode, même pour une mesure de dépendance d'ordre $K = 2$ seulement.
Ici on ne peut pas faire de mesure objective de la qualité de la séparation obtenue. Pour calculer une telle mesure, il faudrait modifier le dispositif expérimental de manière à disposer également des signaux originaux non mélangés (par exemple en utilisant deux micros supplémentaires, chacun étant placé à proximité d'un locuteur). Dans un tel cas, on pourrait s'inspirer des mesures de qualité décrites dans [7].

4 Angle d'arrivée de voix humaines

4.1 Introduction

Nous venons d'examiner une application pratique dans des conditions réelles, d'un des algorithmes étudiés au Chapitre 2, qui est la séparation aveugle de sources indépendantes. Nous nous étions placés dans le cadre de signaux sonores, à savoir des voix humaines enregistrées sur des microphones.

Ce même cadre expérimental peut servir pour l'application d'un autre problème étudié au Chapitre 2: l'estimation d'Angles d'Arrivée. En effet, nous disposons d'ondes planes (les signaux de paroles de locuteurs suffisamment éloignés des capteurs); ces ondes sont originellement en large bande[22], mais il est toujours possible, par filtrage, de les ramener en bande étroite, conformément aux hypothèses de base du problème étudié au Chapitre 2. Enfin, nous disposons d'une antene réseau composée de 2 capteurs: les 2 micros qui échantillonnent les signaux à la fréquence 10kHz.

4.2 Dispositif expérimental

Le dispositif expérimental est identique à celui de la Section précédente, à ceci près que les micros sont disposés parallèlement l'un à l'autre, et non inclinés de 45^o l'un vers l'autre comme précédemment. En effet, dans le problème de l'estimation d'Angles d'Arrivée, nous avions principalement basé notre étude sur une antenne linéaire uniforme. De plus, notre étude théorique supposait que l'antenne soit correctement calibrée, c'est-à-dire que l'on en connaisse un bon modèle analytique. Ici, notre antenne n'est pas calibrée: nous ne connaissons pas les gains des micros, ni leur directionnalité. Cependant, les micros ayant (d'après le fabricant) les mêmes caractéristiques, il semble préférable de les placer de façon parallèle, de manière à rendre leur réponse directionnelle la plus identique possible. La figure (24) montre la disposition géométrique des micros et des personnes pour cette expérience.

Si les angles θ_1 et θ_2 ne sont pas trop grands, on peut espérer que pour un micro donné, le gain sur un signal venant de θ_1 et celui sur un signal venant de θ_2 seront proches (d'après le fabricant la réponse directionnelle forme une cardioïde).

La distance séparant les 2 micros est $d = 17$cm, et pour s'assurer de l'hypothèse de planéité des ondes, la distance entre les locuteurs et l'antenne des micros est maintenant de 4m. Des enregistrements sont effectués pour différents angles, avec un ou deux locuteurs.

4.3 Transformation en signal bande étroite

Lorsque l'on filtre un signal large bande pour le transformer en signal bande étroite, il est clair que l'on va perdre de la puissance. Il s'agit donc pour nous d'effectuer ce filtrage en essayant de perdre le moins de puissance possible. Pour cela, on va analyser le spectre d'un des signaux de parole que nous avons enregistré séparément, afin de voir où se situe le maximum de puissance.

[22] le spectre de la voix couvre plusieurs octaves

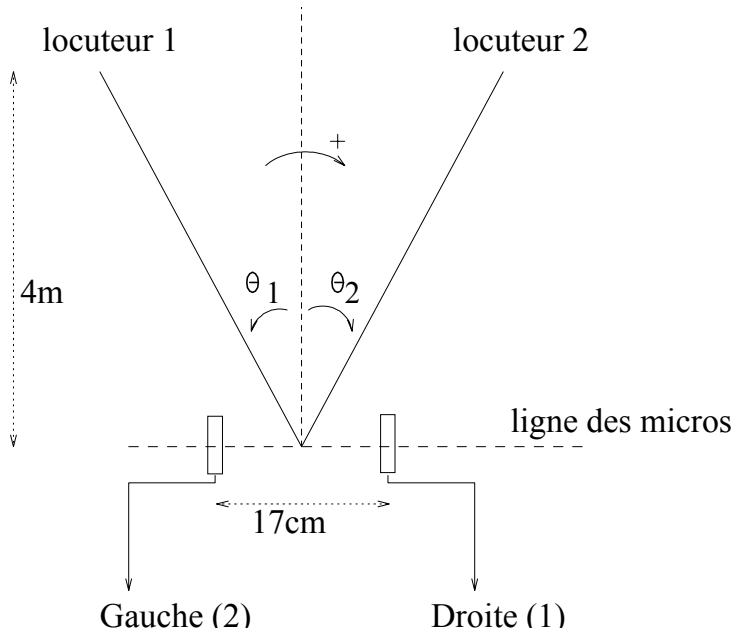

Figure 24: *Dispositif expérimental pour l'estimation d'Angles d'Arrivée de voix humaines*

L'étude qui suit a été conduite sur un enregistrement comportant la voix d'une seule personne, sur un seul micro. On a réalisé des FFT sur 100 tranches successives de 512 échantillons (sans recouvrement). En raison de la symétrie de la Transformée de Fourier d'un signal réel, seules les 256 premières valeurs de la FFT sont représentées. La luminance est proportionnelle au module, et la fréquence 0 (composante continue) a été supprimée pour augmenter la dynamique. La figure (25) est une analyse temps-fréquence du signal. Cette figure montre une forte concentration de l'énergie en dessous de 500Hz. Ceci est confirmé par la figure (26) qui représente $F(\nu) = \sqrt{\sum_{tranches} |\hat{f}_{tranche}(\nu)|^2}$. En particulier, on note que le maximum de puissance se trouve dans des fréquences de l'ordre de 400Hz.

On peut aussi remarquer la forte atténuation du spectre au delà de 4kHz. En réalité, ce phénomène n'est pas une caractéristique du signal de parole: cela est dû au filtrage analogique du dispositif d'acquisition (sa fréquence de coupure est fixée légèrement en dessous de la fréquence de Shannon, qui vaut 5kHz).

Au vu de ces figures, on pourrait penser qu'un signal de parole filtré par un passe-bas dont la fréquence de coupure est 500Hz reste parfaitement compréhensible. Pour le vérifier, nous avons effectué (par programmation informatique) un filtrage passe-bas à la fréquence de coupure 590Hz (par annulation des hautes fréquences sur

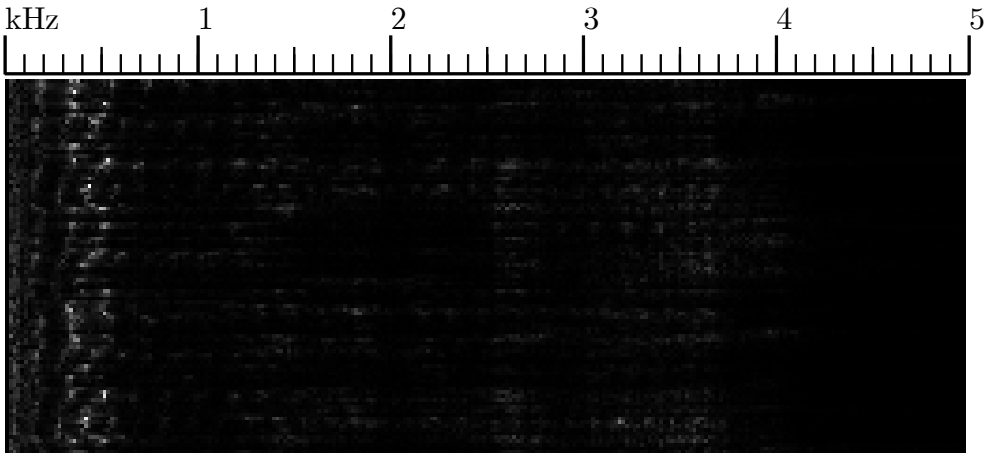

Figure 25: *Analyse temps-fréquence du signal de parole*

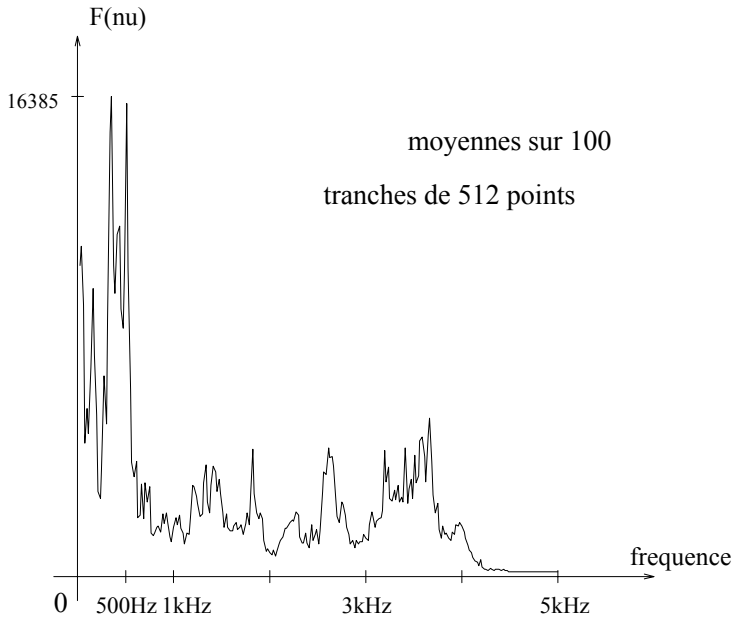

Figure 26: *Spectre du signal de parole*

la FFT, puis FFT inverse). La parole reste compréhensible, mais peu agréable... Les hautes fréquences, bien que portant peu d'énergie, sont donc importantes pour la qualité de la perception auditive.

Choix de la fréquence pour la bande étroite

D'après l'étude qui vient d'être faite, on voit que le maximum de la puissance se trouve aux alentours de $f = 400$Hz. La transformée de Fourier discrète de $f(n)$ est donnée par:

$$\hat{f}(k) = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} f(n) e^{-j2\pi \frac{nk}{N}}$$

où N est le nombre de points utilisés pour calculer la transformée (ici $N = 512$ points). La correspondance entre l'indice entier k et la fréquence réelle ν est donnée par:

$$\nu = \frac{k}{N} F_e$$

où F_e est la fréquence d'échantillonnage (ici $F_e = 10$kHz). Pour que ν approche la fréquence souhaitée qui est d'environ 400 Hz, on va donc prendre $k = 20$. De cette façon, $\nu = 390$kHz. L'enveloppe complexe y_1 du signal sur le micro 1 à la fréquence $\nu = 390$Hz ($k = 20$) est alors donnée par

$$y_1(t) = \hat{f}(k) \qquad \text{(calculé dans la tranche } t\text{)}$$

pour chaque tranche t successive de 512 points. Il suffit d'opérer la même transformation sur les signaux émanant du micro 2 pour obtenir finalement une suite de vecteurs

$$\boldsymbol{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}$$

correspondant aux signaux bande étroite reçus sur chacun des 2 micros. D'après la théorie du problème des Angles d'Arrivée, on a

$$\forall t, \quad \boldsymbol{y}(t) = A(\Omega)\boldsymbol{x}(t) + \boldsymbol{b}(t) \tag{16}$$

où $A(\Omega)$ est la matrice directionnelle contenant les Angles d'Arrivée $\Omega = [\omega_1, \omega_2]^T$. Il suffit alors de mettre en œuvre l'un ou l'autre des algorithmes d'estimation d'Angles d'Arrivée étudiés au Chapitre 2, pour pouvoir estimer ω_1 et ω_2.

4.4 Estimation de l'Angle d'Arrivée d'un seul signal

4.4.1 Etude théorique

Dans le cas d'un seul signal, l'équation (16) se simplifie en

$$\boldsymbol{y}(t) = \boldsymbol{a}(\omega)x(t) + \boldsymbol{b}(t)$$

où

- $x(t)$ est un scalaire dépendant du temps

- $\boldsymbol{y}(t)$ et $\boldsymbol{b}(t)$ sont des vecteurs de longueur 2
- $\boldsymbol{a}(\omega)$ est le vecteur directionnel

Pour une antenne dont les capteurs sont identiques et ont un gain omnidirectionnel, $\boldsymbol{a}(\omega)$ en dépend que de ω:

$$\boldsymbol{a}(\omega) = \begin{bmatrix} 1 \\ e^{-j\omega} \end{bmatrix}$$

où

$$\omega = \frac{2\pi d}{\lambda} \sin\theta \tag{17}$$

avec

- $d =$ distance entre les deux micros
- $\lambda =$ longueur d'onde du signal
- $\theta =$ angle d'arrivée

Nous avons ici $d = 17cm$, λ la longueur d'onde du signal filtré vaut $\lambda = \frac{v}{\nu}$ où $v = 340$ m/s est la vitesse du son dans l'air, et ν est la fréquence bande étroite choisie. Rappelons que pour obtenir un maximum de puissance, nous avions filtré le signal de parole original à $\nu = \frac{k}{N} F_e$ avec $k = 20$, $F_e = 10$kHz la fréquence d'échantillonnage, et $N = 512$ points pour la TFD.
Ainsi, puisque nous connaissons parfaitement d et λ, l'équation (17) montre que la connaissance de ω est équivalente à la connaissance de θ. Nous nous intéressons donc essentiellement à l'estimation de ω, puisque l'estimation de l'Angle d'Arrivée en découlera alors naturellement.

Nous disposons de N_s signaux $\boldsymbol{y}(t)$ (notés désormais $\boldsymbol{y}_t$) avec lesquels nous pouvons calculer une estimation de la matrice de covariance:

$$\hat{R}_y = \frac{1}{N_s} \sum_{t=1}^{N_s} \boldsymbol{y}_t \boldsymbol{y}_t^H \tag{18}$$

Grâce à cette matrice complexe 2×2, nous pouvons estimer le paramètre ω par l'algorithme MUSIC (*cf* Chapitre 2 Section (3.1.1)) et par l'algorithme ESPRIT (*cf* Chapitre 2 Section (3.1.2)). Nous pouvons aussi en déduire la matrice de covariance normalisée

$$R^{nor} = \frac{1}{r_{moy}} \hat{R}_y \tag{19}$$

où r_{moy} est la moyenne des éléments de la diagonale de $\hat{R}_y$ (équation (109) du Chapitre 2). Si nous disposons d'un Perceptron Réel d'Initialisation dédié à l'estimation de $n = 1$ angle sur un antenne de $m = 2$ capteurs, nous pouvons en tirer une estimation $\tilde{\omega}$.
Nous avons construit un PRI selon les indications du Chapitre 2 Section (4.4). Il comporte $m^2 = 4$ entrées, $l = 2$ neurones cachés et $n = 1$ neurone de sortie. Il a été entraîné auparavant sur une base créée par simulation. Cette base est formée d'exemples entrée/sortie R^{nor}/ω, pour des paramètres ω variant de -2.72 à $+2.72$ radians. Sur la base d'apprentissage, le signal $x(t)$ est simulé par un processus gaussien de moyenne nulle et de variance 1. Le bruit est gaussien blanc, et le SNR de cette base vaut 20dB.

La base d'apprentissage du PRI utilisé dans cette application d'estimation de l'Angle d'Arrivée de voix humaines est composée de données très différentes de celles correspondant ici aux signaux de son. On verra que cette base fournit quand même une initialisation $\tilde{\omega}$ suffisante pour le PCSC.
Le PCSC utilisé pour cette aplication est très petit: c'est un perceptron à trois couches, dont $m = 2$ neurones sur la couches d'entrée, $n = 1$ neurone sur la couche caché, et $m = 2$ neurones sur la couche de sortie.

La méthode du Maximum de Vraisemblance et les trois méthodes évoquées (MUSIC, ESPRIT, et notre méthode neuromimétique associant le PRI au PCSC) seront employées ici pour l'estimation de l'Angle d'Arrivée de la voix humaine.

4.5 Résultats

Les trois méthodes mentionnées fournissent l'estimation du paramètres ω. Il reste alors à calculer l'Angle d'Arrivée θ correspondant à ω. On a vu que:

$$\omega = \frac{2\pi df}{v} \sin\theta \tag{20}$$

avec

- $d = 0.17\text{m}$
- $v = 340\text{m/s}$
- $f = \frac{20.10^4}{512}\text{Hz}$

En outre, on a mentionné précédemment que l'appareillage n'était pas de grande qualité, puisqu'en réalité l'échantillonnage ne se faisait pas simultanément sur les deux micros, mais de manière alternée. D'après la disposition des micros, le capteur 2 (gauche) est échantillonné après le capteur 1 (droit). Donc lorsque le capteur 2 est

échantillonné à son tour, l'onde a déjà parcouru une certaine distance. Il faut donc retrancher une demi-période d'échantillonnage au délai typiquement utilisé pour les algorithmes usuels:

$$\tau = \tau_0 - \tau_{ech} = \frac{d}{v}\sin\theta - \frac{1}{2F_e}$$

d'où une réécriture de (20):

$$\omega = \frac{2\pi df}{v}\sin\theta - \frac{2\pi f}{2F_e} = \frac{2\pi df}{v}\sin\theta - \frac{20\pi}{512}$$

soit finalement:

$$\sin\theta = \frac{v}{2fd}\left(\frac{\omega}{\pi} + \frac{20}{512}\right)$$

Les résultats obtenus par les différentes méthodes sont consignés dans le tableau suivant:

	$\hat{\omega}$ (radians)	$\hat{\theta}$ (degrés)
MUSIC	-0.441	-15.1^o
ESPRIT	-0.441	-15.1^o
PRI	-0.204	-3.8^o
PRI + PCSC	-0.430	-14.5^o
MV	-0.431	-14.6^o

L'angle θ où était placé le locuteur était en fait -20^o. Comme on le voit, les différentes méthodes donnent des estimations très proches les unes des autres. Par contre, il y a un certain écart entre ces estimations $\hat{\theta}$ et la valeur supposée de $\theta = -20^o$. On peut tenter d'expliquer cet écart par les remarques suivantes:

- l'écart entre les deux micros a été mesuré (au double décimètre) à leur extrémité. Cependant, le capteur lui-même était peut-être situé plus à l'intérieur du micro, éventuellement décentré. Cette erreur, ajoutée à l'erreur de mesure, peut entraîner un léger écart.

- l'angle de -20^o a été mesuré au sol par des calculs sur les longueurs des côtés d'un triangle rectangle (figure (27)). Il est bien évident que de telles mesures entraînent des erreurs. De plus, une personne ne peut être placée en un point: elle occupe une certaine tranche d'espace qui peut induire des écarts sur θ de plusieurs degrés. En outre, la personne peut éventuellement bouger légèrement, ce qui entraîne d'autres incertitudes sur les mesures.

- les micros n'étaient pas à la même hauteur que la bouche du locuteur: il aurait donc fallu, en toute rigueur, tenir compte de cette différence d'altitude dans nos calculs, ce qui les aurait rendus très lourds.

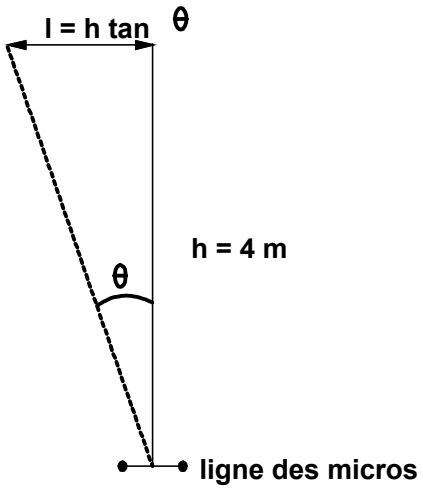

Figure 27: *Technique de mesure de l'angle au sol*

- la vitesse du son a été prise à 340m/s, mais la vitesse réelle était probablement un peu différente.
- les micros ne sont pas parfaitement omnidirectionnels. De plus, une étude pratique (voir Annexe C) a montré qu'il existait, pour une direction donnée, un rapport de gain différent entre les deux micros. De ce fait, le vecteur directionnel $\boldsymbol{a}(\omega)$ devrait être modélisé par

$$\boldsymbol{a}(\omega) = \begin{bmatrix} 1 \\ \alpha e^{-j\omega} \end{bmatrix}$$

avec α un réel positif ($\alpha \neq 1$ ici). Cette remarque montre que la calibration de l'antenne est primordiale dans le problème des Angles d'Arrivée.

4.6 Estimation de l'Angle d'Arrivée de 2 signaux

4.6.1 Etude théorique

Pour deux signaux, l'équation (16) sécrit

$$\boldsymbol{y}(t) = A(\omega_1, \omega_2)\boldsymbol{x}(t) + \boldsymbol{b}(t)$$

où $\boldsymbol{y}(t), \boldsymbol{b}(t)$ et $\boldsymbol{x}(t)$ sont des vecteurs de longueur 2 et $A(\omega_1, \omega_2)$ est une matrice directionnelle carrée 2×2. Pour une antenne dont les capteurs sont identiques et ont un gain omnidirectionnel, $A(\omega_1, \omega_2)$ ne dépend que des paramètres directionnels

ω_1 et ω_2:

$$A(\omega_1, \omega_2) = \begin{bmatrix} 1 & 1 \\ e^{-j\omega_1} & e^{-j\omega_2} \end{bmatrix}$$

où

$$\omega_i = \frac{2\pi d}{\lambda} \sin \theta_i$$

avec les mêmes données d, λ, v, f, F_e et N que précédemment. La variable θ_i est l'angle d'arrivée du locuteur $i \in \{1, 2\}$.
D'après ces renseignements, on peut donc encore affirmer que la connaissance de ω_i équivaut à celle de θ_i. C'est pourquoi nous considérons désormais les ω_i comme les paramètres à estimer.
Dans le cas de figure qui nous intéresse ici, le nombre de capteurs m est égal au nombre de sources. D'après [16], les méthodes "classiques" au second ordre, type Haute-Résolution ou Maximum de Vraisemblance ne peuvent être employées. La seule approche possible est donc l'utilisation de la méthode proposée au Chapitre 2 Section 7, c'est-à-dire l'approche fusion AdA avec SAS: on utilise un Perceptron Complexe à Structure Contrainte minimisant une erreur mixte composée, d'une part, de l'erreur quadratique de sortie, et d'autre part, de la mesure de dépendance de Burel [3], [4].

Détaillons à présent la taille et la base d'entraînement du PRI utilisé pour initialiser ce PCSC de fusion. Nous disposons de $m = 2$ capteurs: la matrice de covariance normalisée $\hat{R}^{nor}$ (voir équations (19) et (18)) peut donc fournir $m^2 = 4$ valeurs réelles. De plus, les moments croisés d'ordre 4 (les $T_{ijkl} = E\{y_i y_j^* y_k y_l^*\}$ du Chapitre 2 Section (7.3.2.2)) fournissent $\frac{m^4 + m}{2} = 9$ autres réels. En tout, nous avons donc $4 + 9 = 13$ entrées pour le PRI. Par ailleurs il y a $n = 2$ neurones sur la couche de sortie, et nous prenons $l = 5$ neurones sur la couche cachée.
La base d'entraînement de ce perceptron est composée de données simulées, puisque nous ne disposions d'aucune base de données comportant des signaux de parole réels reçus sous toutes les orientations (θ_1, θ_2) possibles. Les signaux simulés sont construits comme suit: les 13 valeurs réelles issues de la matrice de covariance normalisée $\hat{R}^{nor}$ et des tenseurs statistiques T_{ijkl} d'ordre 4 sont calculés à partir de signaux capteurs $\boldsymbol{y}(t)$ se comportant comme le modèle typique supposé:

$$\boldsymbol{y}(t) = \begin{bmatrix} 1 & 1 \\ e^{-j\omega_1} & e^{-j\omega_2} \end{bmatrix} \boldsymbol{x}(t) + \boldsymbol{b}(t)$$

où les composantes de $\boldsymbol{x}(t)$ sont des complexes aléatoires de distribution uniforme[23] entre -1 et 1, $\boldsymbol{b}(t)$ est un bruit gaussien blanc de variance telle que le rapport

[23]il faut que ces signaux soient non gaussiens de manière à ce que leurs statistiques d'ordre supérieur contiennent de l'information pertinente

signal-à-bruit vaut $SNR = 30$dB, et les paramètres ω_1 et ω_2 varient entre -2.72 et $+2.72$ radians, avec $\omega_1 < \omega_2$. En réalité ce PRI est le même que celui utilisé dans les expériences sur signaux simulés du Chapitre 2 Section (8.2.2).

4.6.2 Résultats

Comme dans le cas précédent à une seule voix, on n'obtient pas directement l'estimation des Angles d'Arrivée: on obtient d'abord l'estimation des paramètres intermédiaires ω_1 et ω_2. On en déduit les θ_i ($i \in \{1, 2\}$) par

$$\omega_i = \frac{2\pi df}{v} \sin\theta_i - \frac{20\pi}{512}$$

avec les mêmes valeurs pour d, f et v que précédemment. On obtient finalement les résultats suivants:

	$\hat{\omega}_1$ (radians)	$\hat{\omega}_2$ (radians)	$\hat{\theta}_1$ (degrés)	$\hat{\theta}_2$ (degrés)
PRI	-0.232	-0.207	-5.2^o	15.7^o
PRI + PCSC	-0.653	-0.217	-25.6^o	16.2^o

pour des Angles d'Arrivées "supposés" de -30^o et $+25^o$. On constate un certain écart entre les estimations $\hat{\theta}_1$ et $\hat{\theta}_2$, et les angles vrais θ_1 et θ_2. Toutefois, de la même manière que nous avions tenté d'expliquer cet écart dans le cas d'une seule voix, il est nécessaire de prendre en compte les incertitudes liées au dispositif expérimental lui-même:

- erreur possible sur la mesure de l'écartement des micros et sur la vitesse du son
- parallélisme approximatif des micros
- micros non omnidirectionnels
- micros situés plus bas que la bouche des locuteurs
- locuteurs de taille différente
- angles d'arrivée "vrais" mesurés de manière approximative au sol (erreur de plusieurs degrés possible).

Comme on le voit, la calibration de l'antenne compte tout autant que les performances de l'algorithme utilisé, et si une application exige une grande précision sur les Angles d'Arrivée de signaux de parole, il apparaît nécessaire d'utiliser du matériel performant, et surtout, de procéder au calibrage du réseau de micros avant toute utilisation.
Bien que l'erreur d'estimation soit non négligeable (de l'ordre de 4^o et 9^o), il convient

de rappeler que dans le cas présent (autant de sources que de capteurs), toutes les méthodes classiques (MUSIC, ESPRIT, MV) échouent. Seule la méthode proposée reste capable de fournir une estimation.

4.7 Extension

Le problème que nous venons de traiter peut se matérialiser sous de nombreuses formes dans la pratique: par exemple, dans le cadre du studio de télévision du futur (recherches auxquelles Thomson participe), où tous les appareils seront mûs automatiquement. Pour remplacer un cadreur de caméra, dont le travail consiste à diriger sa caméra vers la personne qui parle, et à la suivre si besoin est, on pourrait utiliser notre système de localisation de voix humaines, et s'en servir pour diriger automatiquement la caméra. Lorsqu'une autre personne prendrait la parole, le système localiserait cette nouvelle source, et grâce à un système de commande adéquat, enverrait l'ordre de déplacement de la caméra. Le quatrième Chapitre de cette thèse est consacré à la commande référencée capteur, dont un cas particulier est la commande (de caméra) référencée capteur sonore, que nous venons de mentionner ici.

Annexe A: Nombre d'équations implicites contenues dans le critère de dépendance à l'ordre $K = 4$ pour deux signaux

Le critère correspond à la juxtaposition de la mesure de dépendance avec des contraintes sur les moyennes et les variances des signaux.

Pour $n = 2$ signaux, les contraintes sur les moyennes

$$\forall i \in [1, n], \quad E\{S_i\} = 0$$

apportent n équations, soit ici 2 équations (sauf si les signaux sont déjà centrés, auquel cas ces deux équations ne sont pas utilisées).

Les contraintes sur les variances

$$\forall i \in [1, n], \quad E\{S_i^2\} = \sigma_i^2$$

apportent aussi $n = 2$ équations.

La mesure de dépendance de Burel

$$m_d = \sum_{\alpha_1=0}^{\infty} ... \sum_{\alpha_n=0}^{\infty} \sum_{\beta_1=0}^{\infty} ... \sum_{\beta_n=0}^{\infty} G_{\alpha_1...\alpha_n,\beta_1...\beta_n} M_{\alpha_1...\alpha_n} M_{\beta_1...\beta_n}$$

est une forme quadratique définie positive, donc elle ne s'annule que lorsque tous les moments $M_{\alpha_1...\alpha_n}$ sont nuls. Lorsque nous sommes en présence de deux signaux seulement, ces moments s'expriment d'une manière très simple:

$$M_{\alpha_1\alpha_2} = R_{\alpha_1\alpha_2} - R_{\alpha_1 0} R_{0\alpha_2}$$

avec la condition $\alpha_1 + \alpha_2 \leq K$ où

$$R_{\alpha_1\alpha_2} = E\left\{S_1^{\alpha_1} S_2^{\alpha_2}\right\}$$

Ici K correspond à une indépendance d'ordre 4. Nous allons déterminer le nombre d'équations concernées:

- <u>moments d'ordre 2:</u> $\alpha_1 + \alpha_2 = 2$ nous fournit une équation:

  $$M_{11} = 0 \quad \text{soit} \quad R_{11} - R_{10} R_{01} = 0$$

  D'après la définition de R, cela équivaut à:

  $$E\{S_1 S_2\} - E\{S_1\} E\{S_2\} = 0$$

- <u>moments d'ordre 3:</u> $\alpha_1 + \alpha_2 = 3$ nous fournit 2 équations:

$$\left\{ \begin{array}{rcl} M_{12} & = & 0 \\ M_{21} & = & 0 \end{array} \right. \quad \text{soit} \quad \begin{array}{rcl} R_{12} - R_{10}R_{02} & = & 0 \\ R_{21} - R_{20}R_{01} & = & 0 \end{array}$$

ce qui équivaut finalement à:

$$\left\{ \begin{array}{rcl} E\{S_1 S_2^2\} - E\{S_1\}E\{S_2^2\} & = & 0 \\ E\{S_1^2 S_2\} - E\{S_1^2\}E\{S_2\} & = & 0 \end{array} \right.$$

- <u>moments d'ordre 4:</u> $\alpha_1 + \alpha_2 = 4$ nous fournit 3 équations:

$$\left\{ \begin{array}{rcl} M_{13} & = & 0 \\ M_{22} & = & 0 \\ M_{31} & = & 0 \end{array} \right. \quad \text{soit} \quad \begin{array}{rcl} R_{13} - R_{10}R_{03} & = & 0 \\ R_{22} - R_{20}R_{02} & = & 0 \\ R_{31} - R_{30}R_{01} & = & 0 \end{array}$$

ce qui équivaut finalement à:

$$\left\{ \begin{array}{rcl} E\{S_1 S_2^3\} - E\{S_1\}E\{S_2^3\} & = & 0 \\ E\{S_1^2 S_2^2\} - E\{S_1^2\}E\{S_2^2\} & = & 0 \\ E\{S_1^3 S_2\} - E\{S_1^3\}E\{S_2\} & = & 0 \end{array} \right.$$

Au total nous avons donc, pour une mesure de dépendance à l'ordre $K = 4$ et pour $n = 2$ signaux, 6 équations provenant de la mesure de dépendance de Burel, et 4 équations provenant des contraintes sur les moyennes et les variances de signaux, soit en tout 10 équations. Dans le cas où les signaux sont déjà centrés, nous n'avons que 2 équations provenant des contraintes, soit $6 + 2 = 8$ équations en tout.

Annexe B: Perception des échos par une oreille humaine

Afin de mieux appréhender la façon dont une oreille humaine perçoit les échos, nous avons effectué quelques tests en relation avec le enregistrements que nous avions réalisés.
La salle dans laquelle les enregistrements de voix humaines ont été faits est une salle de conférence presque vide, où ne se trouvait aucun meuble à part quelques chaises empilées dans un coin, et un bureau. Si l'on considére que les échos les plus importants proviennent des réflexions sur le sol et le plafond, cela correspond à une différence de marche de quelques mètres. Pour fixer les ordres de grandeur, une distance de 3.4m correspond à un délai de 10ms, soit 100 échantillons.

Certains de nos enregistrements comportent les voix de deux personnes: ces enregistrements ont pour but de valider les algorithmes de Séparation Aveugle de Sources. D'autres enregistrements ne comportent qu'une seule voix, et sont destinés à tester des algorithmes d'estimation d'Angle d'Arrivée. Nous avons écouté attentivement les enregistrements ne comportant qu'une seule voix: nous ne distinguons pas d'écho.

Pour voir à partir de quel délai l'oreille est capable de distinguer un écho, nous avons mis au point un programme permettant, à partir d'un enregistrement[24] ne comportant qu'une seule voix, de restituer un signal égal au signal original auquel est ajouté un écho artificiel, sans atténuation. Le retard de cet écho est un paramètre modifiable par l'opérateur. Après quelques tests, nous avons constaté qu'un délai de 70ms (soit 700 échantillons puisque nous sommes à 10.000kHz) est nécessaire pour que l'on arrive à distinguer la présence d'un écho. Ce délai correspond à une différence de marche de 24m. Vues les dimensions de la pièce où ont été réalisés les enregistrements, il faudrait de *multiples* trajets pour arriver à une telle différence de marche. En outre, après autant de réflexions, il y a de grandes chances que ce signal soit très fortement atténué, voire négligeable. Cela explique pourquoi nous n'avons pas distingué d'échos dans les enregistrements originaux: au dessous de 70ms, les échos sont imperceptibles par une oreille humaine, et au dessus de 70ms, les échos sont trop atténués pour être audibles.
Le fait que seuls les échos dont le retard dépasse 70ms sont perceptibles par l'oreille humaine laisse à penser que cette durée de 70ms correspond peut-être plus ou moins à la taille de la fenêtre temporelle sur laquelle l'oreille réalise une analyse spectrale du son. C'est la raison pour laquelle, dans les expériences d'estimation d'Angles d'Arrivée (où on utilise l'enveloppe complexe des signaux), on a réalisé les FFT sur

[24] voie de droite ou voie de gauche

des tranches de 512 points[25] (soit ici 51.2ms).

[25]il fallait prendre la puissance de 2 la plus proche

Annexe C: Etude pratique du rapport de gain entre les micros

En théorie, le signal perçu par le second micro devrait être le même que celui perçu par le premier micro, décalé de τ et éventuellement atténué d'un facteur a. En notant $g(n)$ ce signal, on a:

$$g(n) = af(n - \tau)$$

Au niveau de la TFD (en arrondissant τ à l'entier le plus proche), ceci se traduit par

$$\hat{g}(k) = a\hat{f}(k)e^{-j2\pi\frac{k}{N}\tau}$$

On a donc, en notant $y_1 = \hat{f}(k)$ et $y_2 = \hat{g}(k)$:

$$|y_2| \,/\, |y_1| = a$$

Afin de vérifier s'il existait réellement une différence entr les deux micros utilisés dans notre dispositif, nous avons fait l'expérience suivante: dans un cas où un seul locuteur parle, nous avons calculé les FFT à 390 Hz (20^e point de la FFT, soit une fréquence de 20/512*10kHz) sur les 100 premières tranches des signaux reçus sur le micro 1 et sur le micro 2 séparément. La figure suivante montre le rapport des modules obtenus sur chaque tranche de 512 points. On note qu'en moyenne, la courbe se situe en dessous

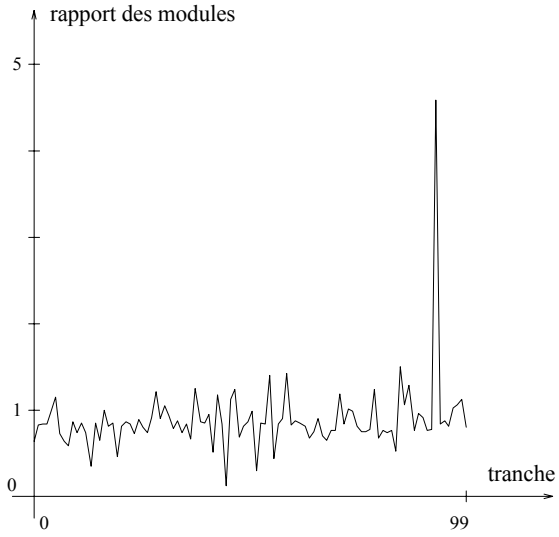

Figure 28: *Rapport des modules sur 100 tranches successives.*

du rapport escompté de 1. En fait, la moyenne du rapport des gains des micros sur

les 100 tranches étudiées, est de 0.89. En toute rigueur, il faudrait tenir compte de ce coefficient dans la modélisation mathématique de l'antenne.

References

[1] H.K. Aghajan, B.H. Khalaj & T. Kailath, "Estimation of Skew Angle in Text Image Analysis by Sensor Array Processing Techniques", SPIE vol 1906, Character Recognition Technologies, pp 49-60, 1993

[2] H.K. Aghajan & T. Kailath, "Sensor Array Processing Techniques for Super Resolution Multi-Line-Fitting and Straight Edge Detection", IEEE Transactions on Image Processing, No 4, vol II, pp 454-465, Oct 1993

[3] G. Burel, "Réseaux de Neurones en Traitement d'Images: des Modèles Théoriques aux Applications Industrielles", Thèse de Doctorat de l'Université de Bretagne Occidentale, Brest, 1991

[4] G. Burel, "Blind Separation of Sources: a Non-Linear Algorithm", Neural Networks, vol 5, No 6, pp 937-947, 1992

[5] G. Burel & N. Rondel, "Neural Networks for Array Processing: from DOA Estimation to Blind Separation of Sources", IEEE Conference on Systems, Man and Cyberbetics, Invited Paper, Le Touquet, France, 17-20 Oct 1993

[6] V. Démoulin, thèse traitant de la séparation d'émetteurs, en cours à Thomson Broadcast Systems, Centre de Rennes, sous la responsabilité de G. Burel et M. Pécot

[7] G. Faucon, R. Le Bouquin & A.A. Azirani, "Mesures Objectives de la Réduction de Bruit", Colloque GRETSI, Juan-Les-Pins, 13-16 Sept 1993

[8] S.T. Hinds, J.L. Fisher & D.P. d'Amato, "A Document Skew Detection Method using Run-Length Encoding and the Hough Transform", International Conference on Pattern Recognition, Atlantic City, New Jersey, USA, Vol 1, pp 464-468, 1990

[9] C. Jutten & J. Hérault, "Une Solution Neuromimétique au Problème de Séparation de Sources", Traitement du Signal, vol 5, No 6, 1988

[10] J. Hérault & C. Jutten, "Réseaux Neuronaux et Traitement du Signal", collection *Traitement du Signal*, Hermès, Paris, 1994

[11] P. Réfrégier, "Théorie du Signal", collection *Enseignement de la physique*, Masson, Paris, 1993

[12] N. Rondel & G. Burel, "Cooperation of Multi-Layer Perceptrons for Angles of Arrival Estimation", IEE 4th International Conference on Artificial Neural Networks, Cambridge, GB, 26-28 Juin, 1995

[13] N. Rondel & G. Burel, "Cooperation of Multi-Layer Perceptrons for the Estimation of Skew Angle in Text Document Images", IEEE 3rd International Conference on Document Analysis and Recognition, pp 1141-1144, Montréal, Canada, 14-16 August, 1995

[14] R. Roy & T. Kailath, "ESPRIT:Estimation of Signal Parameters via Rotational Invariance Techniques", IEEE Transactions on ASSP, Vol 7, No 37, pp 984-995, Juil 1989

[15] P. Stoica & A. Nehorai, "MUSIC, Maximum Likelihood, and Cramer-Rao Bound", IEEE Trans. on Acoustics, Speech and Signal Processing, vol 37, No 5, pp 720-741, May 1989

[16] M. Wax & I.Ziskind, "On Unique Localization of Multiple Sources by Passive Sensor Arrays", IEEE Trans. on Acoustics, Speech and Signal Processing, vol 37, No 7, pp 996-1000, July 1989

Chapitre 4
Commande Référencée Capteur

Un être vivant qui commence à savoir coordonner son système visuel va vouloir agir sur les choses qu'il voit (tourner la tête, fixer un objet, se rapprocher pour mieux le discerner, etc, ...). Aussi semble-t'il naturel, une fois que l'analyse des informations acquises de manière "passive" est affectuée, de relier un système de commande aux capteurs qui ont fourni les informations brutes.

La commande référencée capteur résulte de l'association d'un (ou plusieurs) capteur(s) à une entité motorisée qui (éventuellement) se meut en réponse à des informations provenant de ce(s) capteur(s). Dans le problème qui nous intéresse ici, un robot 6 axes muni, par exemple, d'une caméra CCD à l'extrémité de son bras, est associé à un système de contrôle en boucle fermée. Un objet en mouvement autonome par rapport au système, se déplace dans l'espace 3D (mouvements de translation et de rotation). Le but est de réaliser un asservissement de la caméra sur l'objet, de manière à ce que l'image fournie par la caméra soit celle, définie arbitrairement à l'avance, d'une position de référence de l'objet dans l'image.

Dans une première partie nous traiterons plus particulièrement des caractéristiques du modèle de caméra type sténopé, et de la mécanique du solide, domaines nécessaires à l'introduction du problème.

La seconde partie de ce chapitre s'intéresse aux travaux effectués antérieurement sur ce problème. Dans une troisième partie, des améliorations visant à renforcer la stabilité du système sont proposées.

La quatrième partie introduit l'utilisation de techniques neuronales dans la résolution du problème, et une nouvelle méthode, basée sur la mise en oeuvre d'un perceptron multicouche, est proposée.

Enfin la dernière partie détaille des applications possibles à ce problème général.

1 Présentation du problème

1.1 Introduction

Le problème auquel nous nous intéressons ici est un problème que nous avons voulu très général, afin de pouvoir répondre ensuite à un grand nombre de problématiques appliquées/d'applications.

On dispose d'un robot 6 axes GT6A schématisé en figure (1). A l'extrémité de son bras est montée une caméra CCD dont on connaît grossièrement la position par rapport au poignet du robot.

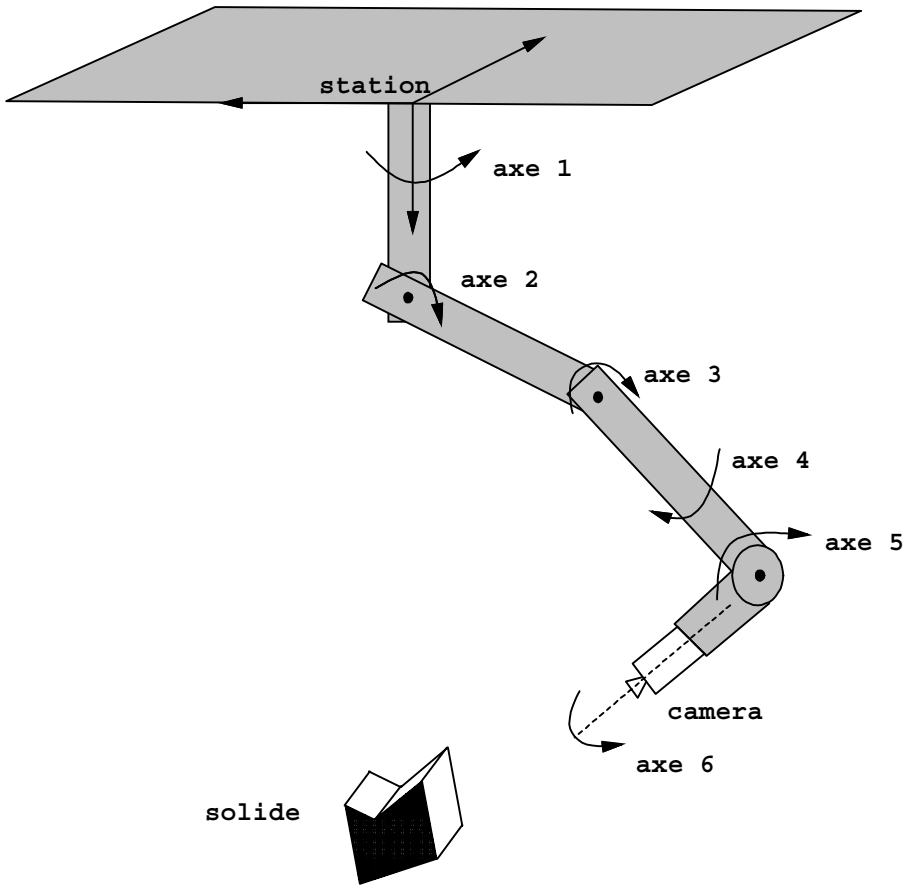

Figure 1: *Vue du système dans son ensemble.*

Un solide isolé en mouvement dans l'espace, se déplace dans l'environnement du

robot. Une position arbitraire de cet objet dans l'image caméra est appelée "position de référence de l'objet dans l'image", et constitue l'objectif de l'asservissement: on désire qu'à chaque instant, le système de contrôle agisse sur le robot pour que l'image sur l'écran de la caméra soit, à tout instant, aussi proche que possible de l'image de référence. Ainsi, c'est la position relative de la caméra par rapport à l'objet qui est contrôlée en permanence: on souhaite réaliser une liaison rigide entre la caméra et l'objet.

L'échelle des temps est divisée de manière régulière par une période d'échantillonnage suffisamment courte pour pouvoir utiliser des hypothèses de travail simplificatrices, comme on le verra par la suite. Les seules informations dont on dispose seront fournies par la caméra (capteur CCD), mais ne seront utilisables qu'une itération après leur saisie, afin de prendre en compte les délais dus au traitement d'image. De ce fait, la période d'échantillonnage est fortement conditionnée par les performances du traitement d'image considéré.

Les formules de changement de repère, les équations du mouvement d'un solide dans l'espace, ainsi que les formules de projection d'une caméra de modèle sténopé, sont nécessaires pour établir les équations qui seront employées lors du calcul de la commande à appliquer au robot. C'est pourquoi nous procédons auparavant à quelques rappels.

1.2 Modèle sténopé

Nous allons tout d'abord rappeler le modèle de la caméra de type sténopé, afin de définir les paramètres dont nous aurons besoin par la suite pour écrire les équations du système. Ce sont les équations du mouvement d'un solide dans l'espace qui vont permettre d'expliciter ces paramètres.

Le modèle de la caméra de type sténopé peut se définir comme une convention dans la projection des points de l'espace dans le plan image, suivant les indications de la figure (2).
Les équations principales régissant le modèle sténopé sont les suivantes:

$$\begin{cases} u &= \frac{f}{S_X}\frac{X_C}{Z_C} + u_0 = x + u_0 \\ v &= \frac{f}{S_Y}\frac{Y_C}{Z_C} + v_0 = y + v_0 \end{cases}$$

où

- (u, v) sont les coordonnées du point p dans le plan image (en pixels),
- (u_0, v_0) sont les coordonnées du centre optique dans le plan image,
- (X_C, Y_C, Z_C) sont les coordonnées du point P_C de l'objet dans le repère caméra,

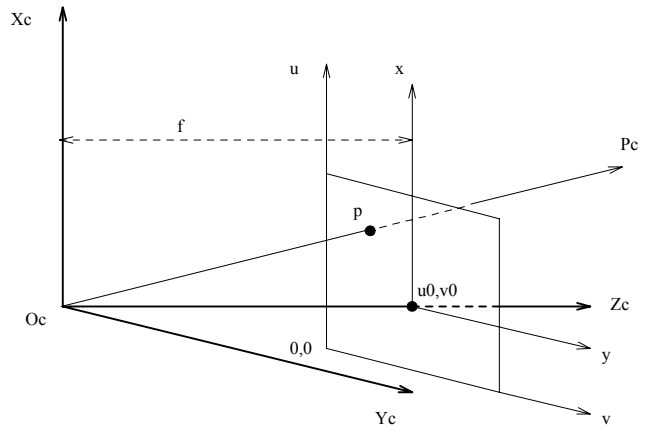

Figure 2: *Le modèle sténopé.*

- les paramètres S_X et S_Y représentent la taille du pixel, et f la distance focale.

La dérivation des paramètres images u et v donne:

$$\left\{ \begin{array}{rcl} \frac{du}{dt} & = & \frac{f}{S_X}\left(\frac{1}{Z_C}\frac{dX_C}{dt} - \frac{X_C}{Z_C^2}\frac{dZ_C}{dt}\right) \\ \frac{dv}{dt} & = & \frac{f}{S_Y}\left(\frac{1}{Z_C}\frac{dY_C}{dt} - \frac{Y_C}{Z_C^2}\frac{dZ_C}{dt}\right) \end{array} \right.$$

soit, matriciellement:

$$\left[\begin{array}{c} \frac{du}{dt} \\ \frac{dv}{dt} \end{array} \right] = \left[\begin{array}{ccc} \frac{f}{S_X Z_C} & 0 & -\frac{fX_C}{S_X Z_C^2} \\ 0 & \frac{f}{S_Y Z_C} & -\frac{fY_C}{S_Y Z_C^2} \end{array} \right] \left[\begin{array}{c} \frac{dX_C}{dt} \\ \frac{dY_C}{dt} \\ \frac{dZ_C}{dt} \end{array} \right]$$

En posant $I = [x\ y]^T$, on obtient:

$$\frac{dI}{dt} = \left[\begin{array}{ccc} \frac{f}{S_X Z_C} & 0 & -\frac{x}{Z_C} \\ 0 & \frac{f}{S_Y Z_C} & -\frac{y}{Z_C} \end{array} \right] \frac{dP_C}{dt} \tag{1}$$

On voit donc que pour écrire les équations propres au système, il faut pouvoir exprimer $\vec{V}_P^C = \frac{d\vec{P}_C}{dt}$, la vitesse du point P_C de l'objet dans le repère caméra. C'est ici qu'intervient la mécanique du solide.

1.3 Rappels de la mécanique du solide

Le mouvement instantané d'un solide est caractérisé par le torseur cinématique $C^{\mathcal{E}}(S)$ tel que:

$$C^{\mathcal{E}}(S) \equiv \left\{ \begin{array}{l} \vec{\omega}(S/\mathcal{E}) \\ \vec{V}_P^{\mathcal{E}} \end{array} \right.$$

où $\mathcal{E}$ est le référentiel, S le solide, et P un point quelconque de S. $\vec{\omega}(S/\mathcal{E})$ est la rotation instantanée de S par rapport à $\mathcal{E}$, et $\vec{V}_P^{\mathcal{E}}$ est la vitesse en translation du point P dans $\mathcal{E}$.

Soit un repère fixe R_0, et un repère (éventuellement) mobile R_1. La relation de composition des mouvements s'écrit:

$$\vec{V}_P^{R_0} = \vec{V}_P^{R_1} + \vec{V}_{P\in R_1}^{R_0}$$

où le dernier terme représente la vitesse dans R_0 du point fixe de R_1 qui coïncide avec P à ce moment.
Si on applique cette relation au référentiel fixe R du robot et au référentiel C de la caméra, on obtient:

$$\vec{V}_P^{R} = \vec{V}_P^{C} + \vec{V}_{P\in C}^{R}$$

d'où la valeur cherchée au paragraphe précédent:

$$\vec{V}_P^{C} = \vec{V}_P^{R} - \vec{V}_{P\in C}^{R}$$

Pour mieux expliciter les deux termes mis en évidence, écrivons la relation fondamentale de la cinématique du solide:

$$\forall (P,Q) \in S^2, \quad \vec{V}_Q^{\mathcal{E}} = \vec{V}_P^{\mathcal{E}} + \vec{\omega}(S/\mathcal{E}) \wedge \overrightarrow{PQ}$$

et appliquons-la à $\vec{V}_{P\in C}^{R}$:

$$\vec{V}_{P\in C}^{R} = \vec{V}_{O_C}^{R} + \vec{\omega}(C/R) \wedge \overrightarrow{O_C P}$$

où O_C est l'origine du référentiel C. De même, pour le point P appartenant au solide, on écrit:

$$\vec{V}_P^{R} = \vec{V}_{O_O}^{R} + \vec{\omega}(O/R) \wedge \overrightarrow{O_O P}$$

où O_O est l'origine du référentiel lié à l'objet. On remarque que si O_O se trouve au centre de gravité de l'objet, et que de plus l'objet est isolé dans l'espace, alors $\vec{V}_{O_O}^{R} = \vec{V}_G^{R} = cte$, et $\vec{\omega}(O/R) = cte$.

Finalement, on peut écrire la vitesse d'un point de l'objet dans le repère caméra comme:

$$\vec{V}_P^{C} = \vec{V}_{O_O}^{R} + \vec{\omega}(O/R) \wedge \overrightarrow{O_O P} - \vec{V}_{O_C}^{R} - \vec{\omega}(C/R) \wedge \overrightarrow{O_C P} \qquad (2)$$

On observe que les deux premiers termes peuvent s'interpréter comme l'influence du mouvement absolu de l'objet dans le référentiel caméra, et les deux derniers termes comme l'influence du mouvement de la caméra.

1.4 Equation fondamentale du problème

C'est la fusion des deux approches vues jusqu'ici, à savoir l'approche sténopé et l'approche mécanique du solide, qui va nous permettre d'établir les équations du problème.

Les conventions du modèle sténopé ont conduit à l'équation (1), où l'on voit que la vitesse d'un point P_C de l'objet dans l'image ne dépend que de sa position dans l'image (x et y), de la profondeur de ce point dans le référentiel caméra (Z_C), et de la vitesse $\vec{V}_P^C$ de ce point P_C dans le référentiel caméra:

$$\frac{dI}{dt} = \begin{bmatrix} \frac{f}{S_X Z_C} & 0 & -\frac{x}{Z_C} \\ 0 & \frac{f}{S_Y Z_C} & -\frac{y}{Z_C} \end{bmatrix} \vec{V}_P^C$$

Or les relations de la cinématique du solide montrent (d'après l'équation (2)) que $\vec{V}_P^C$ se décompose en la somme de deux parties bien distinctes, à savoir une partie due au mouvement absolu de l'objet dans le référentiel fixe, et une partie due au mouvement de la caméra. On peut donc écrire:

$$\frac{dI}{dt} = \frac{dI_C}{dt} + \frac{dI_O}{dt} \tag{3}$$

où $\frac{dI_C}{dt}$ est la partie du mouvement de l'objet dans l'image, due uniquement au mouvement de la caméra, et $\frac{dI_O}{dt}$ est la partie du mouvement de l'objet dans l'image, due uniquement au mouvement propre de l'objet.

Détaillons la valeur de $\frac{dI_C}{dt}$; le mouvement de la caméra dans le référentiel absolu R induit un mouvement des points dans l'image tel que:

$$\frac{dI_C}{dt} = \begin{bmatrix} \frac{f}{S_X Z_C} & 0 & -\frac{x}{Z_C} \\ 0 & \frac{f}{S_Y Z_C} & -\frac{y}{Z_C} \end{bmatrix} \left(-\vec{V}_{O_C}^R - \vec{\omega}(C/R) \wedge \overrightarrow{O_C P} \right) \tag{4}$$

Or par simple calcul, on démontre que:

$$\forall (\vec{a}, \vec{b}), \quad \vec{a} \wedge \vec{b} = \tilde{a}.\vec{b}$$

où

$$\tilde{a} = \begin{bmatrix} 0 & -a_Z & a_Y \\ a_Z & 0 & -a_X \\ -a_Z & a_X & 0 \end{bmatrix}$$

On peut donc écrire de même

$$\begin{aligned} \vec{\omega}(C/R) \wedge \overrightarrow{O_C P} &= -\overrightarrow{O_C P} \wedge \vec{\omega}(C/R) \\ &= -\widetilde{O_C P}.\vec{\omega}(C/R) \end{aligned}$$

L'équation (4) peut alors être réécrite matriciellement sous la forme:

$$\frac{dI_C}{dt} = \begin{bmatrix} \frac{f}{S_X Z_C} & 0 & -\frac{x}{Z_C} \\ 0 & \frac{f}{S_Y Z_C} & -\frac{y}{Z_C} \end{bmatrix} \times \begin{bmatrix} -1 & 0 & 0 & 0 & -Z_C & Y_C \\ 0 & -1 & 0 & Z_C & 0 & -X_C \\ 0 & 0 & -1 & -Y_C & X_C & 0 \end{bmatrix} \begin{bmatrix} \vec{V}_{O_C}^R \\ \vec{\omega}(C/R) \end{bmatrix} \quad (5)$$

ce qui donne, après calcul et simplification en utilisant les définitions de x et y:

$$\frac{dI_C}{dt} = B \begin{bmatrix} \vec{V}_{O_C}^R \\ \vec{\omega}(C/R) \end{bmatrix}$$

où B est la matrice 2×6 suivante:

$$B = \begin{bmatrix} -\frac{f}{S_X Z_C} & 0 & \frac{x}{Z_C} & \frac{S_Y}{f}xy & -\left(\frac{f}{S_X} + \frac{S_X}{f}x^2\right) & \frac{S_Y}{S_X}y \\ 0 & -\frac{f}{S_Y Z_C} & \frac{y}{Z_C} & \frac{f}{S_Y} + \frac{S_Y}{f}y^2 & -\frac{S_X}{f}xy & -\frac{S_X}{S_Y}x \end{bmatrix} \quad (6)$$

Appelons $\vec{u}_C$ le vecteur 6×1 tel que:

$$\vec{u}_C = \begin{bmatrix} \vec{V}_{O_C}^R \\ \vec{\omega}(C/R) \end{bmatrix}$$

et reprenons l'équation (3). Elle s'écrit maintenant:

$$\frac{dI}{dt} = B\vec{u}_C + \frac{dI_O}{dt}$$

Par suite, on peut écrire, aux temps t et $t-1$:

$$\begin{aligned} I^\bullet(t) &= B(t)\vec{u}_C(t) + I_0^\bullet(t) \\ I^\bullet(t-1) &= B(t-1)\vec{u}_C(t-1) + I_0^\bullet(t-1) \end{aligned}$$

en notant $I^\bullet = \frac{dI}{dt}$.
Si l'on suppose que l'intervalle de temps qui sépare les prises de vue est suffisamment petit, on peut faire l'hypothèse que les mouvements des points dans l'image dus à l'objet sont uniformes, c'est-à-dire que $I_0^\bullet(t)$ a peu varié entre $t-1$ et t. Par suite, on peut soustraire et simplifier les deux équations précédentes. On obtient:

$$I^\bullet(t) - I^\bullet(t-1) = B(t)\vec{u}_C(t) - B(t-1)\vec{u}_C(t-1) \quad (7)$$

Finalement, en écrivant[1] que $I^{\bullet}(t) = \frac{I(t+1)-I(t)}{T}$, et $\mathrm{B} = T.B$, on en tire l'équation fondamentale régissant le problème:

$$\mathrm{B}(t)\vec{u}_C(t) = I(t+1) - 2I(t) + I(t-1) + \mathrm{B}(t-1)\vec{u}_C(t-1) \tag{8}$$

La figure (3) montre les évènements dans l'ordre chronologique où ils se déroulent. A partir de l'équation (8), il faut souligner deux points qui vont permettre d'écrire

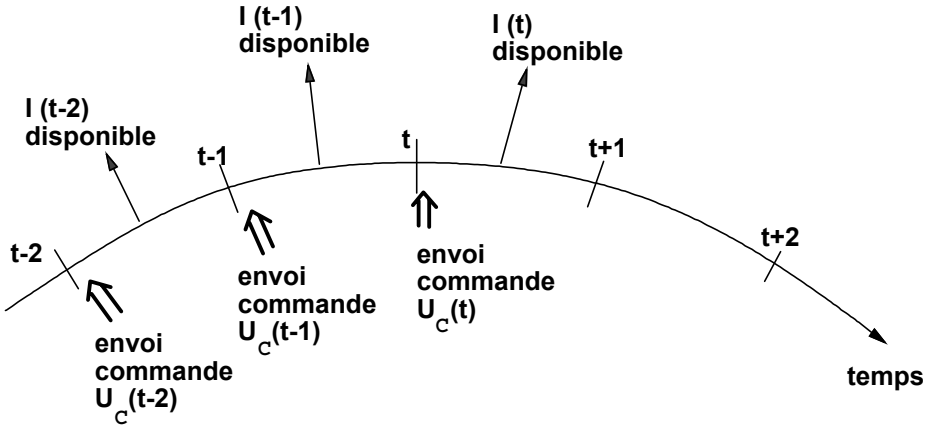

Figure 3: *Chronologie des évènements*

l'équation d'estimation de la commande. Tout d'abord, pour déterminer de façon unique la commande $\vec{u}_C(t)$ à appliquer à l'instant t, il faut pouvoir inverser la matrice $\mathrm{B}(t)$. Pour un point A de l'objet, le vecteur colonne caractérisant sa position dans l'image $I_A(t) = [x_A(t), y_A(t)]^T$ est de longueur 2, et la matrice associée $\mathrm{B}_A(t)$ est de dimension 2×6. En conséquence, il faut au minimum 3 points non colinéaires pour obtenir une matrice de rang plein. Ainsi, on appellera dorénavant $\mathrm{B}(t)$ la matrice $2N \times 6$ associée aux N points caractéristiques de l'objet (dont trois au moins sont non colinéaires), telle que:

$$\mathrm{B}(t) = \begin{bmatrix} \mathrm{B}_1(t) \\ \mathrm{B}_2(t) \\ \vdots \\ \mathrm{B}_N(t) \end{bmatrix} \tag{9}$$

De la même manière la matrice colonne $I(t)$ est maintenant de dimension $2N \times 1$:

[1] T est la période d'échantillonnage. Par commodité, on a écrit $t-1$ et $t+1$ au lieu de $t-T$ et $t+T$ pour éviter d'alourdir inutilement les équations

$$I(t) = \begin{bmatrix} x_1(t) \\ y_1(t) \\ x_2(t) \\ y_2(t) \\ \vdots \\ x_N(t) \\ y_N(t) \end{bmatrix} \tag{10}$$

La matrice $\mathrm{B}(t)$, construite à partir d'au moins 3 points non colinéaires, est maintenant de rang plein, et nous noterons $\mathrm{B}^+(t)$ sa pseudo-inverse définie par:

$$\mathrm{B}^+(t) = \left[\mathrm{B}^T(t)\mathrm{B}(t)\right]^{-1} \mathrm{B}^T(t) \tag{11}$$

D'autre part, il faut rappeler que la commande idéale appliquée à l'instant t est supposée amener la caméra au temps $t+1$ dans sa position de référence par rapport à l'objet. On a donc idéalement $I(t+1) = I_{ref}$, ce qui nous permet d'écrire:

$$\vec{u}_C(t) = \mathrm{B}^+(t)\left[I_{ref} - 2I(t) + I(t-1)\right] + \mathrm{B}^+(t)\mathrm{B}(t-1)\vec{u}_C(t-1) \tag{12}$$

2 Travaux antérieurs

Une fois posée l'équation fondamentale, diverses approches sont possibles pour la résolution complète de ce problème de contrôle.

En effet, si certains paramètres de cette équation sont directement appréhendables (coordonnées des points dans l'image caméra $I = [x, y]^T$, commande envoyée à l'itération précédente $\vec{u}_C(t-1)$), d'autres soulèvent plus de difficultés. C'est le cas de la matrice $\mathrm{B}(t)$, qui dépend des profondeurs Z_C des points P_C de l'objet dans le repère R_C de la caméra.

Lorsque deux caméras sont utilisées (vision stéréoscopique), ou dans le cas d'un couplage avec un capteur de distance (télémètre laser par exemple), la valeur de $Z_C(t)$ est directement fournie par une technique de fusion de capteurs appropriée. Lorsque l'on ne dispose, comme c'est le cas ici, que d'une seule caméra, d'autres méthodes doivent être employées pour estimer la matrice $\mathrm{B}(t)$.

2.1 Matrice B constante

Une première approche, proposée par Chaumette, Espiau et Rives (voir [2] et [3]), utilise le fait que lorsque l'on se trouve dans une position proche de l'équilibre ($I(t) \simeq I_{ref}$), des conditions suffisantes à la convergence de l'algorithme sont respectées, et l'on peut alors utiliser l'approximation $\mathrm{B}(t) = \mathrm{cte} = \mathrm{B}_{ref}$.

L'algorithme proposé par Chaumette *et. al.* se construit comme suit: soit s^* la position de référence des paramètres image (notre I_{ref}), et $s(r(t))$ la position courante (notre $I(t)$), on définit la tâche $e(t)$ par:

$$e(t) = C\,(s(r(t)) - s^*) \tag{13}$$

où $r(t)$ se réfère à la position et attitude de la caméra dans un repère fixe, et C est une matrice, dite de combinaison, qui permet notamment de prendre en compte dans la commande, un nombre d'informations visuelles supérieur au nombre de degrés de liberté du problème.

Le but de l'asservissement étant de rendre rigide la liaison objet-caméra, il s'agit de minimiser $\|e(t)\|$. Ainsi la commande proposée est:

$$\vec{u}_C(t) = -\lambda e(t) \tag{14}$$

où λ est un coefficient de gain positif.

On a:

$$e^\bullet = \frac{\partial e}{\partial r}\frac{\partial r}{\partial t} = \frac{\partial e}{\partial s}\frac{\partial s}{\partial r}\vec{u}_C(t) = -\lambda C L^T e \tag{15}$$

où L^T, appelée par Chaumette la matrice d'interaction de l'objet, est en fait notre matrice $\mathrm{B}(t)$, dans laquelle, par définition, Chaumette utilise les valeurs simplificatrices $f = S_X = S_Y = 1$ (voir équation (6)).

L'équation différentielle régissant le problème:

$$e^\bullet = -\lambda C L^T e \tag{16}$$

implique que la stabilité de l'algorithme est assurée dès que:

$$C L^T > 0 \tag{17}$$

Ainsi, le choix de $C = L^{T+}$ convient. C'est ici que Chaumette utilise l'approximation $L^T(s(t)) \simeq L^T(s^*(t))$, soit, pour nous, $\mathrm{B}(t) = \mathrm{B}_{ref}$, en arguant du fait que, lorsque le système est accroché, on a $I(t) \simeq I_{ref}$ (ou $s(t) \simeq s^*$).

Finalement, pour un objet fixe, la commande s'écrit, en reprenant nos notations:

$$\vec{u}_C(t) = -\lambda \mathrm{B}^{+}_{ref} \left(I(t) - I_{ref}\right) \tag{18}$$

On retrouve la première partie de l'équation (12):

$$\vec{u}_C(t) = \mathrm{B}^{+}(t) \left[I_{ref} - I(t)\right] - \mathrm{B}^{+}(t) \left[I(t) - I(t-1)\right] + \mathrm{B}^{+}(t)\mathrm{B}(t-1)\vec{u}_C(t-1) \tag{19}$$

avec $\lambda = 1$.

Pour le cas où l'objet est mobile, l'équation (15) devient:

$$e^{\bullet} = \frac{\partial e}{\partial r}\vec{u}_C(t) + \frac{\partial e}{\partial t} \tag{20}$$

en supposant sa vitesse constante.

Il apparaît donc un terme supplémentaire $\frac{\partial e}{\partial t}$, d'où un nouveau choix pour la commande:

$$\vec{u}_C(t) = -\lambda e - \left(\widehat{\frac{\partial e}{\partial r}}\right)^{+} \widehat{\frac{\partial e}{\partial t}} \tag{21}$$

Comme auparavant, $\dfrac{\partial e}{\partial r}$ est estimé par la valeur de CL^T lorsque la caméra est dans la position de référence. Dans ce cas, CL^T est équivalente à l'identité, d'après le choix de $C = L^{T+}$; $\dfrac{\partial e}{\partial t}$ est estimé à l'aide d'un intégrateur (par analogie avec un type de commande usuel en automatique, selon Chaumette):

$$\left(\widehat{\frac{\partial e}{\partial t}}\right)^{\bullet} = -\mu e \tag{22}$$

où μ est un gain positif.

De là la nouvelle écriture de la commande:

$$\vec{u}_C(t) = -\lambda \mathrm{B}^{+}_{ref} \left(I(t) - I_{ref}\right) - \mu \mathrm{B}^{+}_{ref} \sum_{j=0}^{t-1} \left[I(j) - I_{ref}\right] \tag{23}$$

Comme on le voit, l'appréhension d'un objet mobile ne se fait pas de la même façon dans l'approche de Chaumette[2] que dans l'équation (12), cependant il est intéressant de noter que, confronté au problème de l'estimation de $Z_C(t)$, la solution proposée

[2]Cette approche n'est qu'en partie basée sur une modélisation de la cinématique du problème; elle fait appel à des lois de commande dont la forme est donnée a priori. Elle nécessite également le réglage de paramètres (λ, μ)

par Chaumette *et. al.* comsiste à estimer plus globalement $\mathrm{B}(Z_C(t))$, en approximant cette matrice par la valeur qu'elle prend lorsque la caméra est dans la position de référence:

$$\forall t, \mathrm{B}(Z_C(t)) = \mathrm{B}_{ref} \tag{24}$$

2.2 Matrice B non constante

Pour estimer la commande $\vec{u}_C(t)$ à envoyer au robot au temps t, c'est également l'équation (12) qui est utilisée ici:

$$\vec{u}_C(t) = \mathrm{B}^+(t)\left[I_{ref} - 2I(t) + I(t-1)\right] + \mathrm{B}^+(t)\mathrm{B}(t-1)\vec{u}_C(t-1)$$

Dans un premier temps, on a décrit une approche considérant la matrice $\mathrm{B}(t)$ constante dans le temps, arguant du fait que lorsque le système est bien accroché, les coordonnées 3D de l'objet relativement à la caméra varient peu.

Une autre approche, proposée par Feddema [4], considère la matrice $\mathrm{B}(t)$ non constante. Feddema propose une méthode permettant d'estimer en permanence la matrice $\mathrm{B}(t) \simeq \theta(t)\mathrm{B}(0)$, où $\mathrm{B}(0)$ est la connaissance initiale de B, et $\theta(0)$ une matrice de dimension $2N \times 2N$, si N est le nombre de points caractéristiques de l'objet. La matrice $\theta(t)$ est remise à jour en permanence. L'inconvénient de cette méthode est la taille de la matrice $\theta(t)$ à estimer, car elle implique un algorithme gourmand en temps de calcul.

A partir des travaux de Feddema, Papanikolopoulos [7] propose un algorithme plus rapide qui consiste à estimer uniquement la profondeur de chacun des points. Le nombre de paramètres à estimer est alors réduit à N (au lieu de $4N^2$), d'où une diminution du temps de calcul.

Dans l'équation (12) rappelée ci-dessus, les vecteurs connus sont $I_{ref}, I(t-1)$, et $\vec{u}_C(t-1)$. Le vecteur à estimer est la commande $\vec{u}_C(t)$ à envoyer au prochain top d'horloge. Il reste donc à estimer $\mathrm{B}(t), I(t)$ et $\mathrm{B}(t-1)$.

Papanikolopoulos [7], [8] propose tout d'abord de supposer que $\mathrm{B}(t) \simeq \mathrm{B}(t-1)$, c'est-à-dire que la matrice $\mathrm{B}(t)$ est dans un premier temps approximée comme étant égale à $\mathrm{B}(t-1)$. Par contre, $\mathrm{B}(t-1)$ *n'est pas* supposée égale à $\mathrm{B}(t-2)$: elle est estimée par un algorithme de prédiction de type Kalman que nous détaillerons plus loin. Ainsi, Papanikolopoulos utilise, à chaque itération, l'approximation

$$\mathrm{B}(t) \simeq \mathrm{B}(t-1)$$

sans supposer que la matrice $\mathrm{B}(t)$ est constante.
Pour estimer $I(t)$, Papanikolopoulos réécrit tout simplement l'équation (8) à l'ordre inférieur ($t = t - 1$):

$$\mathrm{B}(t-1)\vec{u}_C(t-1) = I(t) - 2I(t-1) + I(t-2) + \mathrm{B}(t-2)\vec{u}_C(t-2) \qquad (25)$$

c'est-à-dire qu'il estime les coordonnées images $I(t)$ comme la valeur qu'elles prendraient si l'équation fondamentale (12) était parfaitement exacte[3].

De l'équation (25), on tire une estimation de $I(t)$:

$$I(t) = 2I(t-1) - I(t-2) + \mathrm{B}(t-1)\vec{u}_C(t-1) - \mathrm{B}(t-2)\vec{u}_C(t-2)$$

qui remplace l'inconnue $I(t)$ dans (12):

$$\vec{u}_C(t) = \mathrm{B}^+(t)\left[I_{ref} - 3I(t-1) + 2I(t-2)\right] + 2\mathrm{B}^+(t)\mathrm{B}(t-1)\vec{u}_C(t-1) - \mathrm{B}^+(t)\mathrm{B}(t-2)\vec{u}_C(t-2) \qquad (26)$$

Dans cette nouvelle équation, les seules inconnues sont les matrices $\mathrm{B}(t), \mathrm{B}(t-1)$ et $\mathrm{B}(t-2)$. Avec l'approximation $\mathrm{B}(t) \simeq \mathrm{B}(t-1)$, cette équation se simplifie en:

$$\vec{u}_C(t) = \mathrm{B}^+(t-1)\left[I_{ref} - 3I(t-1) + 2I(t-2)\right] + 2\vec{u}_C(t-1) - \mathrm{B}^+(t-1)\mathrm{B}(t-2)\vec{u}_C(t-2) \qquad (27)$$

Il reste alors à estimer $\mathrm{B}(t-1)$ et $\mathrm{B}(t-2)$. Ces deux matrices ont la même structure donc on ne s'intéressera ici qu'à $\mathrm{B}(t-1)$, sachant qu'ensuite on pourra facilement en déduire une règle pour $\mathrm{B}(t-2)$. On a:

$$\mathrm{B}(t-1) = \begin{bmatrix} \mathrm{B}_1(t-1) \\ \mathrm{B}_2(t-1) \\ \vdots \\ \mathrm{B}_N(t-1) \end{bmatrix}$$

avec, pour chaque point caractéristique i:

$$\mathrm{B}_i(.) = T \begin{bmatrix} -\frac{f}{S_X Z_C^i(.)} & 0 & \frac{x_i(.)}{Z_C^i(.)} & \frac{S_Y}{f}x_i(.)y_i(.) & -\left(\frac{f}{S_X} + \frac{S_X}{f}x_i(.)^2\right) & \frac{S_Y}{S_X}y_i(.) \\ 0 & -\frac{f}{S_Y Z_C^i(.)} & \frac{y_i(.)}{Z_C^i(.)} & \frac{f}{S_Y} + \frac{S_Y}{f}y_i(.)^2 & -\frac{S_X}{f}x_i(.)y_i(.) & -\frac{S_X}{S_Y}x_i(.) \end{bmatrix}$$

d'après l'équation (6).

On voit qu'il est possible de décomposer chaque $\mathrm{B}_i(t-1)$ en deux sous-matrices

[3] en réalité l'équation (12) provient de l'équation (8) arrangée dans un ordre différent

2×3 dont l'une ne dépend que des coordonnées $I_i(t-1)$ du point i dans l'image, et l'autre dépend à la fois de $I_i(t-1)$ et de la *profondeur* $Z_C^i(t-1)$ de ce point dans le référentiel caméra:

$$\mathrm{B}_i(t-1) = \left[\mathrm{B}_V^i(t-1)\mathrm{B}_\omega^i(t-1)\right]$$

où

$$\mathrm{B}_\omega^i(t-1) = T \begin{bmatrix} \frac{S_Y}{f}x_i(t-1)y_i(t-1) & -\left(\frac{f}{S_X} + \frac{S_X}{f}x_i(t-1)^2\right) & \frac{S_Y}{S_X}y_i(t-1) \\ \frac{f}{S_Y} + \frac{S_Y}{f}y_i(t-1)^2 & -\frac{S_X}{f}x_i(t-1)y_i(t-1) & -\frac{S_X}{S_Y}x_i(t-1) \end{bmatrix}$$

ne dépend que de $I_i(t-1)$, et

$$\mathrm{B}_V^i(t-1) = T\varepsilon_C^i(t-1) \begin{bmatrix} -1 & 0 & \frac{S_X}{f}x_i(t-1) \\ 0 & -\frac{S_X}{S_Y} & \frac{S_Y}{f}y_i(t-1) \end{bmatrix} = \varepsilon_C^i(t-1)G_i(t-1)$$

dépend à la fois de $I_i(t-1)$ et de la profondeur $Z_C^i(t-1)$ par l'intermédiaire de

$$\varepsilon_C^i(t-1) = \frac{f}{S_X Z_C^i(t-1)}$$

Nous sommes juste avant le temps t, et nous *connaissons* $I_i(t-1)$. De ce fait, $\mathrm{B}_\omega^i(t-1)$ et $G_i(t-1)$ sont parfaitement connues. Par contre, il reste à estimer, pour chacun des N points caractéristiques de l'objet, le paramètre $\varepsilon_C^i(t-1)$.

C'est ici que Papanikolopoulos propose d'utiliser un estimateur type filtrage de Kalman. Les notations suivantes sont volontairement simplifiées afin de rendre le message plus clair. Notons:

- ε_t le paramètres dépendant de la profondeur, que l'on désire estimer
- $\vec{m}_t$ la mesure (vecteur 2×1)
- b_t le bruit d'état (scalaire) et $\vec{\gamma}_t$ le bruit de mesure (vecteur 2×1)
- P_t la variance (scalaire) de l'erreur de l'estimation sur ε_t
- K_t le gain (matrice 1×2) du filtre de Kalman (pondération de l'innovation)
- R_t la matrice (2×2) de covariance de l'innovation

L'équation d'état s'écrit:

$$\varepsilon_t = \varepsilon_{t-1} + b_t$$

c'est-à-dire que l'on suppose que ε_t varie peu en fonction du temps. L'équation de mesure s'écrit symboliquement:

$$\vec{m}_t = \vec{c}_t \varepsilon_t + \vec{\gamma}_t$$

et il s'agit maintenant d'exprimer $\vec{m}_t$ et $\vec{c}_t$ en fonction de paramètres mesurables. Pour cela nous partons de l'équation (8) écrite à une ordre où toutes les données mesurables sont connues (ordre $t = t - 2$):

$$\mathrm{B}(t-2)\vec{u}_C(t-2) = I(t-1) - 2I(t-2) + I(t-3) + \mathrm{B}(t-3)\vec{u}_C(t-3)$$

Notons en effet que pour l'ordre $t = t - 1$, par exemple, on verrait apparaître $I(t)$, qui est forcément inconnue avant l'instant t.

Pour chaque point caractéristique i, nous allons séparer les matrices $\mathrm{B}_i(.)$ de cette équation en $\mathrm{B}^i_V(.)$ et $\mathrm{B}^i_\omega(.)$, et nous utiliserons la décomposation de la commande $\vec{u}_C$ en

$$\vec{u}_C(.) = \begin{bmatrix} \vec{V}(.) \\ \vec{\omega}(.) \end{bmatrix}$$

On obtient alors:

$$\begin{aligned} &\mathrm{B}^i_V(t-2)\vec{V}(t-2) + \mathrm{B}^i_\omega(t-2)\vec{\omega}(t-2) = \\ &\quad I_i(t-1) - 2I_i(t-2) + I_i(t-3) + \mathrm{B}^i_V(t-3)\vec{V}(t-3) + \mathrm{B}^i_\omega(t-3)\vec{\omega}(t-3) \end{aligned}$$

Avec la notation $\mathrm{B}^i_V(.) = \varepsilon^i_C(.)G_i(.)$, on peut mettre en exergue le paramètre recherché:

$$\begin{aligned} &\varepsilon^i_C(t-2)G_i(t-2)\vec{V}(t-2) - \varepsilon^i_C(t-3)G_i(t-3)\vec{V}(t-3) = \\ &\quad I_i(t-1) - 2I_i(t-2) + I_i(t-2) + I_i(t-3) + \mathrm{B}^i_\omega(t-3)\vec{\omega}(t-3) - \mathrm{B}^i_\omega(t-2)\vec{\omega}(t-2) \end{aligned}$$

Si on suppose que $\varepsilon^i_C(t-2) \simeq \varepsilon^i_C(t-3)$, c'est-à-dire si la profondeur du point dans le référentiel caméra varie peu entre deux images, alors on peut écrire:

$$\varepsilon^i_C(t-2)\left[G_i(t-2)\vec{V}(t-2) - G_i(t-3)\vec{V}(t-3)\right] = \vec{m}_{t-2} - \vec{\gamma}_{t-2}$$

avec

$$\vec{m}_{t-2} = I_i(t-1) - 2I_i(t-2) + I_i(t-2) + I_i(t-3) + \mathrm{B}^i_\omega(t-3)\vec{\omega}(t-3) - \mathrm{B}^i_\omega(t-2)\vec{\omega}(t-2)$$

en ne rappelant pas (par souci de lisibilité) la dépendance en i de $\vec{m}_{t-2}$. Le vecteur $\vec{\gamma}_{t-2}$ représente un bruit (l'erreur que l'on commet par exemple en assimilant $\varepsilon_C^i(t-2)$ à $\varepsilon_C^i(t-3)$.

En posant de la même façon

$$\vec{c}_{t-2} = G_i(t-2)\vec{V}(t-2) - G_i(t-3)\vec{V}(t-3)$$

on obtient l'équation de mesure recherchée:

$$\vec{m}_{t-2} = \vec{c}_{t-2}\varepsilon_{t-2} + \vec{\gamma}_{t-2} \tag{28}$$

où on a noté plus simplement ε_{t-2} à la place de $\varepsilon_C^i(t-2)$.

Nous avons donc maintenant à la fois l'équation d'état et l'équation de mesure. Rappelons en termes généraux les deux phases principales du filtrage de Kalman[4]:

- Phase de prédiction (indice $-$):
  1. ${}^-\varepsilon_t = {}^+\varepsilon_{t-1}$
  2. ${}^-P_t = {}^+P_{t-1} + var(b_t)$
- Phase d'estimation (indice $+$):
  la mesure est maintenant disponible
  3. $R_t = c_t\, {}^-P_t c_t^T + cov(\gamma_t)$
  4. $K_t = {}^-P_t c_t^T R_t^{-1}$
  5. ${}^+\varepsilon_t = {}^-\varepsilon_t + K_t\,[m_t - c_t\, {}^-\varepsilon_t]$

On voit bien que les indices temporels ne correspondent pas à ceux que l'on a énoncés lors de l'établissement de l'équation de mesure (voir équations (28) et précédentes). Nous allons donc replacer les équations de Kalman dans leur contexte temporel exact.

Plaçons-nous juste avant le temps $t-1$ (voir figure (4)).

- temps avant $t-1$:
  Nous ne disposons pas de l'image $I(t-1)$, mais nous disposons de $I(t-2)$. D'après l'étude ci-dessus (équations (28) et précédentes), nous savons que nous disposons alors de la mesure m_{t-3} et de c_{t-3}. Nous sommes donc supposés disposer déjà de ${}^-\varepsilon_{t-3}$ et ${}^-P_{t-3}$ pour chaque point caractéristique $i \in [1, N]$.

[4] on a ôté les flèches des vecteurs pour simplifier la notation déjà surchargée

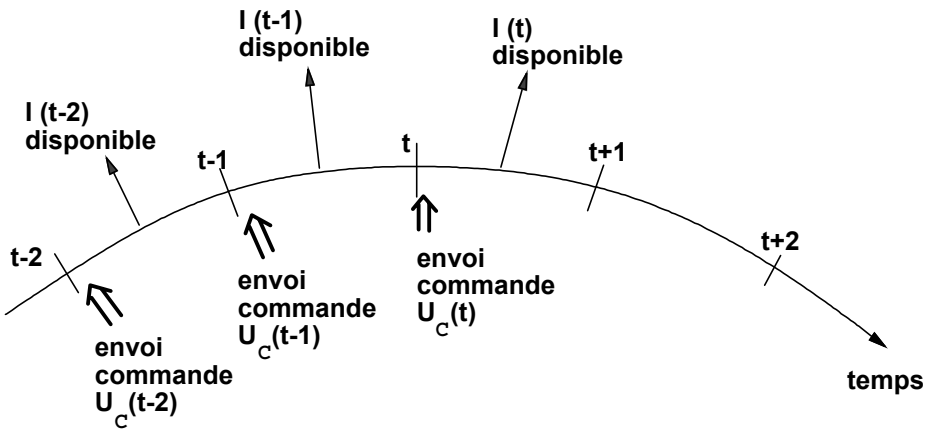

Figure 4: *Chronologie des évènements*

Les équations de la phase d'estimation du filtrage de Kalman nous fournissent R_{t-3}, K_{t-3}, ${}^{+}\varepsilon_{t-3}$ et ${}^{+}P_{t-3}$ pour chaque i. Nous pouvons donc calculer une estimation de la matrice $\mathrm{B}_V(t-3)$, d'où une estimation de $\mathrm{B}(t-3)$.

De plus, nous pouvons prédire[5] ${}^{-}\varepsilon_{t-2}$ pour chaque i:

$$ {}^{-}\varepsilon_{t-2} = {}^{+}\varepsilon_{t-3} $$

et aussi prédire ${}^{-}P_{t-2}$ d'après l'équation (2) de Kalman. Nous en déduisons une prédiction de $\mathrm{B}_V(t-2)$, d'où $\hat{\mathrm{B}}(t-2)$[6]. Ainsi, avec l'approximation $\mathrm{B}(t-1) \simeq \mathrm{B}(t-2)$, on peut utiliser l'équation (27) à l'ordre $t = t-1$ pour estimer la commande à envoyer au temps $t-1$:

$$\vec{u}_C(t-1) = \hat{\mathrm{B}}^{+}(t-2)\,[I_{ref} - 3I(t-2) + 2I(t-3)] + 2\vec{u}_C(t-2) - \mathrm{B}^{+}(t-2)\mathrm{B}(t-3)\vec{u}_C(t-3)$$

Au temps $t-1$ nous envoyons la commande $\vec{u}_C(t-1)$ qui vient d'être calculée, et une photo de l'objet au temps $t-1$ est prise.

- temps après $t-1$:
  Après traitement de cette image, nous disposons des paramètres images $I(t-1)$. Cela signifie que la mesure m_{t-2} est désormais disponible pour chaque point caractéristique $i \in [1, N]$. On peut dès lors, grâce à la phase d'estimation du filtrage de Kalman, estimer R_{t-2}, K_{t-2}, et surtout ${}^{+}\varepsilon_{t-2}$. On en déduit une estimation de la matrice $\mathrm{B}_V(t-2)$ et par suite de la matrice $\mathrm{B}(t-2)$.

[5] équation (1) du filtrage de Kalman

[6] le chapeau sur le $\hat{\mathrm{B}}(t-2)$ indique qu'il ne s'agit que d'une prédiction

On peut de surcroît prédire les $^{-}\varepsilon_{t-1}$ des points caractéristiques (première équation du filtre de Kalman). On en déduit une prédiction de $\mathrm{B}_V(t-1)$, d'où $\hat{\mathrm{B}}(t-1)$. L'équation (27) fournit alors une estimation de la commande à envoyer à l'instant t:

$$\vec{u}_C(t) = \hat{\mathrm{B}}^{+}(t-1)\left[I_{ref} - 3I(t-1) + 2I(t-2)\right] + 2\vec{u}_C(t-1) - \mathrm{B}^{+}(t-1)\mathrm{B}(t-2)\vec{u}_C(t-2)$$

et au temps t cette commande est envoyée au robot.

Nous avons ainsi explicité comment Papanikolopoulos opère l'estimation des paramètres $\varepsilon_C^i(t)$ inconnus de la matrice $\mathrm{B}(t)$, à chaque itération. S'il est vrai que cette méthode est beaucoup moins lourde à mettre en œuvre que la méthode de Feddema (qui nécessitait l'estimation de $4N^2$ paramètres au lieu de N ici), il faut aussi remarquer que cette façon de procéder possède un défaut majeur: l'initialisation des paramètres.

Ne mentionnons pas l'initialisation des $\varepsilon_C^i(t)$, qui, après tout, se retrouve dans toutes les approches. Les $\varepsilon_C^i(t)$ sont initialisés par

$$\varepsilon_C^i(0) = \varepsilon_{Cref}^i = \frac{f}{S_X Z_{Cref}^i} \quad i \in [1, N]$$

où Z_{Cref}^i est la profondeur du i-ème point dans la position de référence.
Les paramètres qui peuvent poser problème sont les paramètres du filtre de Kalman: $var(b_t), cov(\vec{\gamma}_t)$ et la valeur initiale de P_t, à savoir P_0. En effet, lorsque l'on se donne les équations d'état et de mesure:

$$\begin{cases} \varepsilon_t &= \varepsilon_{t-1} + b_t \\ \vec{m}_t &= \vec{c}_t\varepsilon_t + \vec{\gamma}_t \end{cases}$$

il faut en même temps se donner une valeur de la variance supposée de b_t et de la covariance supposée de $\vec{\gamma}_t$. Si on attribue des valeurs erronées à ces paramètres, cela risque de compromettre fortement l'estimation de ε_t obtenue par le filtrage de Kalman. De même, il faut trouver une bonne valeur d'initialisation P_0 à la variable P_t.

Il est difficile de savoir comment initialiser ces paramètres[7], or la réussite ou l'échec du filtrage de Kalman en dépendent fortement. Il s'ensuit que l'utilisation pratique de cet algorithme semble incertaine. C'est pourquoi nous avons cherché à estimer $\varepsilon_C^i(t)$ par d'autres moyens.

[7]Papanikolopoulos ne donne aucune indication théorique à ce sujet

3 Vers de nouvelles approches

3.1 Une première proposition pour B non constante

Nous nous plaçons ici dans une optique très semblable à celle de Papanikolopoulos, puisque nous partons comme lui de l'équation (12) dans laquelle nous incluons l'estimation de $I(t)$. Dans cette équation (cf équation(26)), nous utilisons comme Papanikolopoulos l'approximation $\mathrm{B}(t) \simeq \mathrm{B}(t-1)$, d'où finalement

$$\vec{u}_C(t) = \mathrm{B}^+(t-1)\left[I_{ref} - 3I(t-1) + 2I(t-2)\right] + 2\vec{u}_C(t-1) - \mathrm{B}^+(t-1)\mathrm{B}(t-2)\vec{u}_C(t-2)$$

C'est l'équation (27) de Papanikolopoulos.

Comme lui, nous observons que toutes les matrices présentes sont connues juste avant le temps t, sauf $\mathrm{B}(t-1)$ et $\mathrm{B}(t-2)$ parce que l'on ne connaît pas les $\varepsilon_C^i(t-1)$ et les $\varepsilon_C^i(t-2)$.

C'est ici que les démarches diffèrent. Tandis que Papanikolopoulos utilise un filtre de Kalman, nous préférons employer une autre méthode afin d'éviter l'initialisation quelque peu délicate de $var(b_t), cov(\vec{\gamma}_t)$ et P_0.

Supposons que nous sommes juste avant le temps t et que nous connaissons les $\varepsilon_C^i(t-3)$. Pour estimer les paramètres $\varepsilon_C^i(t-2)$, nous partons de l'équation (8) écrite à l'ordre $t = t-2$:

$$\mathrm{B}(t-2)\vec{u}_C(t-2) = I(t-1) - 2I(t-2) + I(t-3) + \mathrm{B}(t-3)\vec{u}_C(t-3)$$

Réécrivons cette équation pour chacun des points caractéristiques i en séparant les matrices $\mathrm{B}_i(t-2)$ en deux matrices $\mathrm{B}_V^i(t-2)$ et $\mathrm{B}_\omega^i(t-2)$; de même séparons les vecteurs commande $\vec{u}_C(t-2)$ en $\vec{V}(t-2)$ et $\vec{\omega}(t-2)$. On obtient alors:

$$\mathrm{B}_V^i(t-2)\vec{V}(t-2) = I_i(t-1) - 2I_i(t-2) + I_i(t-3) + \mathrm{B}_i(t-3)\vec{u}_C(t-3) - \mathrm{B}_\omega^i(t-2)\vec{\omega}(t-2) \quad (29)$$

Nous avons supposé que nous connaissons les $\varepsilon_C^i(t-3)$. Nous connaissons donc parfaitement la matrice $\mathrm{B}_i(t-3)$. En écrivant:

$$\mathrm{B}_V^i(t-2) = \varepsilon_C^i(t-2)G_i(t-2)$$

où

$$G_i(t-2) = T \begin{bmatrix} -1 & 0 & \frac{S_X}{f} x_i(t-2) \\ 0 & -\frac{S_X}{S_Y} & \frac{S_Y}{f} y_i(t-2) \end{bmatrix}$$

ne dépend pas de $\varepsilon_C^i(t-2)$, nous en déduisons au final une estimation de $\varepsilon_C^i(t-2)$:

$$\begin{aligned} \varepsilon_C^i(t-2) = \left[G_i(t-2)\vec{V}(t-2)\right]^+ \{ & I_i(t-1) - 2I_i(t-2) + I_i(t-3) \\ & + \mathrm{B}_i(t-3)\vec{u}_C(t-3) - \mathrm{B}_\omega^i(t-2)\vec{\omega}(t-2)\Big\} \end{aligned} \tag{30}$$

Nota: Le vecteur $H_i(t-2) = G_i(t-2)\vec{V}(t-2)$ est un vecteur 2×1 qui est en général inversible (pseudo-inversible, plutôt) lorsque $\vec{V}(t-2) \neq 0$. On admet ici la règle suivante:

- si $H_i(t-2)$ est non inversible, on garde $\varepsilon_C^i(t-2) = \varepsilon_C^i(t-3)$
- sinon, on estime $\varepsilon_C^i(t-2)$ par l'équation (30)

En conclusion, notre démarche se déroule ainsi:

- initialisation de $\varepsilon_C^i(0)$ par

$$\varepsilon_C^i(0) = \frac{f}{S_X Z_{Cref}^i}$$

- ...
- juste avant le temps $t-1$, on connaît les $\varepsilon_C^i(t-4)$, donc on connaît les matrices $\mathrm{B}_i(t-4)$. On connaît aussi les $I_i(t-2)$ donc on en déduit $\varepsilon_C^i(t-3)$ par

$$\begin{aligned} \varepsilon_C^i(t-3) = H_i(t-3)^+ \{ & I_i(t-2) - 2I_i(t-3) + I_i(t-4) \\ & + \mathrm{B}_i(t-4)\vec{u}_C(t-4) - \mathrm{B}_\omega^i(t-3)\vec{\omega}(t-3)\Big\} \end{aligned} \tag{31}$$

d'où les matrices $\mathrm{B}_V^i(t-3)$. Nous connaissons alors la matrice $\mathrm{B}(t-3)$ de façon complète, ce qui nous permet d'estimer la matrice $\hat{\mathrm{B}}(t-2)$ en utilisant l'approximation $\varepsilon_C^i(t-2) \simeq \varepsilon_C^i(t-3)$. On calcule alors la commande à envoyer au temps $t-1$:

$$\vec{u}_C(t-1) = \hat{\mathrm{B}}^+(t-2)\left[I_{ref} - 3I(t-2) + 2I(t-3)\right] + 2\vec{u}_C(t-2) - \mathrm{B}^+(t-2)\mathrm{B}(t-3)\vec{u}_C(t-3)$$

- juste avant le temps t, nous connaissons les $\varepsilon_C^i(t-3)$ qui ont été estimés à l'itération précédente (équation (31)), donc on connaît les matrices $\mathrm{B}_i(t-3)$. Grâce aux $I_i(t-1)$ que nous connaissons maintenant, nous pouvons estimer les $\varepsilon_C^i(t-2)$:

$$\begin{aligned}\varepsilon_C^i(t-2) = H_i(t-2)^+ \{ & I_i(t-1) - 2I_i(t-2) + I_i(t-3) \\ & + \mathrm{B}_i(t-3)\vec{u}_C(t-3) - \mathrm{B}_\omega^i(t-2)\vec{\omega}(t-2) \Big\}\end{aligned}$$

  On en déduit les matrices $\mathrm{B}_V^i(t-2)$, d'où $\mathrm{B}(t-2)$. En approximant $\varepsilon_C^i(t-1) \simeq \varepsilon_C^i(t-2)$, on peut également en déduire une prédiction $\hat{\mathrm{B}}(t-1)$ de la matrice $\mathrm{B}(t-1)$. Finalement, la commande à envoyer au robot au temps t s'écrit:

$$\vec{u}_C(t) = \hat{\mathrm{B}}^+(t-1)\left[I_{ref} - 3I(t-1) + 2I(t-2)\right] + 2\vec{u}_C(t-1) - \mathrm{B}^+(t-1)\mathrm{B}(t-2)\vec{u}_C(t-2)$$

- juste avant le temps $t+1$, etc...

Cette méthode, qui n'estime que N paramètres à chaque itération, semble plus intéressante que la méthode proposée par Papanikolopoulos, parce qu'elle évite d'utiliser des paramètres difficiles à initialiser.

Cependant, quand nous avons mis notre approche en pratique, nous nous sommes aperçus que le système pouvait parfois diverger. Nous nous sommes alors penchés sur les implications théoriques de cette méthode. Nous avons réussi à prouver, en dimension 1, que cet estimateur était instable (cf Annexe A).

Cette solution, malgré son aspect facile d'emploi, est donc à proscrire.

3.2 Nouvelles propositions pour B non constante

3.2.1 Un nouvel estimateur pour $\varepsilon_C^i(t)$

A la suite de l'approche que nous venons de présenter, nous avons imaginé une nouvelle approche pour l'estimation du paramètre $\varepsilon_C^i(t)$.

Au départ, cette nouvelle approche ressemble beaucoup à la précédente et à l'approche de Papanikolopoulos: nous partons de l'équation (27):

$$\vec{u}_C(t) = \mathrm{B}^+(t-1)\left[I_{ref} - 3I(t-1) + 2I(t-2)\right] + 2\vec{u}_C(t-1) - \mathrm{B}^+(t-1)\mathrm{B}(t-2)\vec{u}_C(t-2)$$

qui suppose $\mathrm{B}(t) \simeq \mathrm{B}(t-1)$, et nous cherchons à estimer les paramètres inconnus $\varepsilon_C^i(t-1)$ et $\varepsilon_C^i(t-2)$.

Reprenons l'équation (29) qui est une réécriture de l'équation (8) à l'ordre $t = t-2$. Nous sommes juste avant le temps t:

$$\mathrm{B}_V^i(t-2)\vec{V}(t-2) = I_i(t-1) - 2I_i(t-2) + I_i(t-3) + \mathrm{B}_i(t-3)\vec{u}_C(t-3) - \mathrm{B}_\omega^i(t-2)\vec{\omega}(t-2)$$

et développons $\mathrm{B}_i(t-3)$:

$$\mathrm{B}_i(t-3) = \left[\mathrm{B}_V^i(t-3)\ \mathrm{B}_\omega^i(t-3)\right]$$

avec $\mathrm{B}_V^i(t-3)\vec{V}(t-3) = \varepsilon_C^i(t-3)H_i(t-3)$ où H_i ne dépend pas de ε_C^i. En notant de même $\mathrm{B}_V^i(t-2)\vec{V}(t-2) = \varepsilon_C^i(t-2)H_i(t-2)$, on obtient

$$\begin{aligned}\varepsilon_C^i(t-2)H_i(t-2) - \varepsilon_C^i(t-3)H_i(t-3) = I_i(t-1) - 2I_i(t-2) + I_i(t-3) \\ + \mathrm{B}_\omega^i(t-3)\vec{\omega}(t-3) - \mathrm{B}_\omega^i(t-2)\vec{\omega}(t-2)\end{aligned}$$

Dans cette équation, le terme de droite ne contient pas de paramètre ε_C^i. Nous faisons alors l'hypothèse que $\varepsilon_C^i(t-2) \simeq \varepsilon_C^i(t-3)$ et la relation se simplifie en:

$$\begin{aligned}\varepsilon_C^i(t-2) = [H_i(t-2) - H_i(t-3)]^+ \{I_i(t-1) - 2I_i(t-2) + I_i(t-3) \\ + \mathrm{B}_\omega^i(t-3)\vec{\omega}(t-3) - \mathrm{B}_\omega^i(t-2)\vec{\omega}(t-2)\Big\}\end{aligned}$$

Lors de nos expériences, nous avons pu constater que cet estimateur était stable.

3.2.2 Filtrage avec confiance

Pour améliorer encore les performances de ce nouvel estimateur de ε_C^i, on peut lui adjoindre un filtrage avec confiance.

On remarque que l'estimation de $\varepsilon_C(t)$ est calculée[8] comme un produit de $[H(t) - H(t-1)]^+$ par un autre terme.

Or, si la matrice $H(t) - H(t-1)$ est presque nulle, alors la validité de sa

[8]On reprend ici le terme générique (plus naturel) $\varepsilon_C(t)$ à la place de $\varepsilon_C^i(t-2)$ pour cette démonstration. Les notations correspondant au paragraphe précédent seront reprises à la fin de ce paragraphe

pseudo-inverse peut être mise en doute. Ainsi, on peut dire que la confiance qu'on a dans l'estimateur grandit avec la norme de $H(t) - H(t-1)$.

Posons

$$H(t) - H(t-1) = A(t) = \begin{pmatrix} a \\ b \end{pmatrix}$$

Si on appelle "mesure absolue" la norme de $A(t)$, on peut écrire:

$$m_{abs} = \|A(t)\| = \sqrt{a^2 + b^2}$$

On peut alors nommer "confiance" le terme suivant:

$$conf = \frac{m_{abs}}{1 + m_{abs}}$$

Remarquons que la confiance est un réel inclus dans l'intervalle $[0, 1]$, ce qui convient au filtrage:

$$\boxed{\varepsilon_f(t) = \beta\ conf\ \varepsilon_C(t) + (1 - \beta\ conf)\ \varepsilon_f(t-1)}$$

Le paramètre β est un gain: $0 < \beta \leq 1$. Il permet d'adoucir ou de renforcer le filtrage selon les souhaits de l'opérateur. Nous avons choisi un valeur médiane $\beta = 0.5$.

Remarque: Avec les notations du paragraphe précédent, le filtrage s'effectue sur chaque $\varepsilon^i_C(t-2)$ par:

$$\varepsilon^i_f(t-2) = \beta\ conf\ \varepsilon^i_C(t-2) + (1 - \beta\ conf)\ \varepsilon^i_f(t-3)$$

3.2.3 Résumé de l'algorithme proposé

Les paramètres à estimer sont initialisés par:

$$\varepsilon^i_f(0) = \varepsilon^i_C(0) = \frac{f}{S_X Z^i_{Cref}}$$

Juste avant le temps t, on connaît les paramètres image $I(t-1)$. On dispose des $\varepsilon^i_C(t-3)$ et $\varepsilon^i_f(t-3)$. Les opérations à effectuer pour obtenir une estimation de la commande à appliquer au temps t sont listées ci-dessous:

- estimation des $\varepsilon_C^i(t-2)$:

$$\varepsilon_C^i(t-2) = [H_i(t-2) - H_i(t-3)]^+ \{I_i(t-1) - 2I_i(t-2) + I_i(t-3) + \mathrm{B}_\omega^i(t-3)\vec{\omega}(t-3) - \mathrm{B}_\omega^i(t-2)\vec{\omega}(t-2)\Big\}$$

  sauf si $H_i(t-2) = H_i(t-3)$, auquel cas $\varepsilon_C^i(t-2)$ est estimé par $\varepsilon_C^i(t-2) = \varepsilon_C^i(t-3)$.

- estimation des $\varepsilon_f^i(t-2)$:

$$\varepsilon_f^i(t-2) = \beta\ conf\ \varepsilon_C^i(t-2) + (1 - \beta\ conf)\ \varepsilon_f^i(t-3)$$

  où $conf = \frac{m_{abs}}{1+m_{abs}}$ et $m_{abs} = \|A(t)\| = \sqrt{a^2+b^2}$.

- calcul de $\mathrm{B}_V^i(t-2)$:

$$\mathrm{B}_V^i(t-2) = \varepsilon_f^i(t-2) H_i(t-2)$$

  d'où

$$\mathrm{B}_i(t-2) = \left[\mathrm{B}_V^i(t-2)\ \mathrm{B}_\omega^i(t-2)\right]$$

  et la matrice $\mathrm{B}(t-2)$ formée des $\mathrm{B}_i(t-2)$.

- prédiction de $\varepsilon_C^i(t-1)$:

$$\hat{\varepsilon}_C^i(t-1) = \varepsilon_f^i(t-2)$$

  d'où les $\hat{\mathrm{B}}_V^i(t-1)$. On en déduit $\hat{\mathrm{B}}(t-1)$.

- calcul de la commande à appliquer au temps t:

$$\vec{u}_C(t) = \hat{\mathrm{B}}^+(t-1)\,[I_{ref} - 3I(t-1) + 2I(t-2)] + 2\vec{u}_C(t-1) - \hat{\mathrm{B}}^+(t-1)\mathrm{B}(t-2)\vec{u}_C(t-2)$$

On dispose maintenant des $\varepsilon_C^i(t-2)$ et des $\varepsilon_f^i(t-2)$: La boucle est alors bouclée et une nouvelle itération peut commencer.

3.3 Proposition d'amélioration sur l'équation générale

3.3.1 Premières idées

L'idée qui nous motive ici pour améliorer la stabilité de l'estimateur de $\vec{u}_C(t)$ est la suivante: il est peut-être préférable, parfois, d'accepter une petite erreur en pixels au niveau de l'image caméra, que de laisser la caméra faire un grand mouvement (de rotation ou de translation) pour annuler cette erreur. En effet, comme les algorithmes de traitement d'image sont généralement coûteux en temps de calcul, on préfère souvent les simplifier un peu, dans le but de gagner du temps. En contrepartie, il peut exister de petites erreurs en pixels, au niveau de la détection des points caractéristiques dans l'image.

C'est pourquoi, lorsque l'on ne détecte que de petites erreurs entre la position courante et la position de référence des points dans l'image, on préfère accepter ces erreurs que d'engager un grand mouvement des la caméra dans le but de les annuler, d'autant plus que ces erreurs sont peut-être dûes à une moins grande précision du traitement d'image.

On présente ici l'exemple d'une translation suivant l'axe des x qui pourrait (éventuellement) compenser l'erreur en pixels au niveau de l'image. La figure (5) illustre ce propos. Nous allons comparer le coût d'une translation d de la caméra au

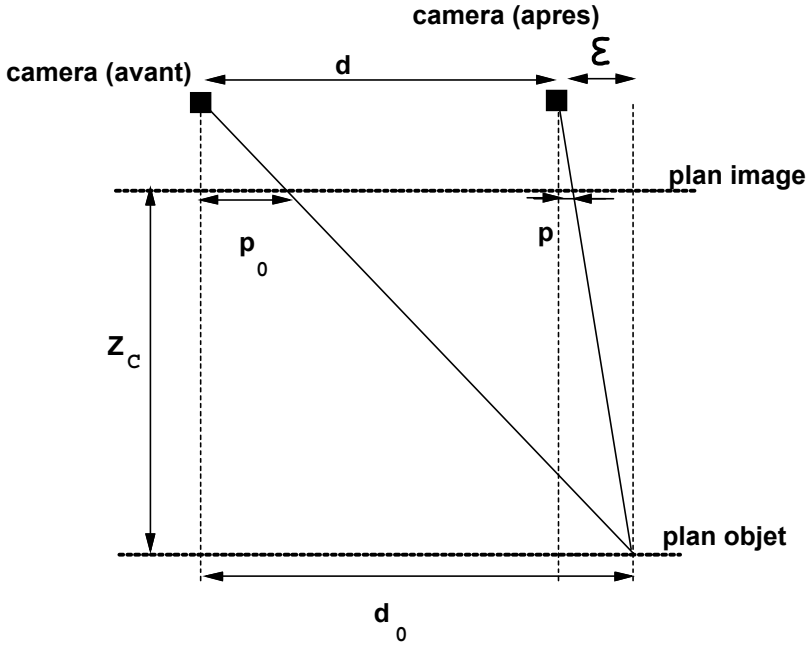

Figure 5: *Compensation des erreurs sur l'image par un mouvement de la caméra*

coût d'une erreur p en pixels au niveau de l'image. On a:

- d_0 = mouvement de l'objet (en m)

- $d =$ mouvement de la caméra (en m)
- $\varepsilon = d_0 - d$
- $p =$ écart résiduel en pixels après le déplacement de la caméra

En choisissant un coût quadratique, le coût de la translation de la caméra selon l'axe des x est $E_{trans} = d^2$, et le coût de l'erreur en pixels est $E_{pix} = p^2$. Comme nous souhaitons favoriser les petites erreurs en pixels au détriment des grands mouvements de caméra, nous allons pondérer le coût du mouvement par λ, donc l'énergie totale dépensée est:

$$\begin{aligned} E &= E_{pix} + \lambda E_{trans} \\ &= p^2 + \lambda d^2 \\ &= \left(\frac{f}{S_X Z_C}\right)^2 \varepsilon^2 + \lambda (d_0 - \varepsilon)^2 \end{aligned}$$

L'énergie minimale est obtenue pour la valeur de ε qui annule la dérivée:

$$\frac{\partial E}{\partial \varepsilon} = 2\left(\frac{f}{S_X Z_C}\right)^2 \varepsilon - 2\lambda(d_0 - \varepsilon) = 0$$

d'où finalement

$$\lambda = \left(\frac{f}{S_X Z_C}\right)^2 \frac{\varepsilon}{d_0 - \varepsilon} \simeq \left(\frac{f}{S_X Z_C}\right)^2 \frac{\varepsilon}{d_0}$$

Or on sait que

$$\varepsilon = \frac{S_X Z_C}{f} p$$

d'où

$$\boxed{\lambda = \left(\frac{f}{S_X Z_C}\right) \frac{p}{d_0}}$$

Pour notre application, on avait, à titre indicatif, $f = 16$ mm, $S_X = 13.3$ microns, et $Z_C = 0.5$ m. Si on accepte une erreur de $p = 2$ pixels sur un déplacement de $d_0 = 20$ cm, alors on obtient numériquement $\lambda = 24000$.

Le même type de calcul est effectué pour des translations en z, et des rotations autour de l'axe des y et de l'axe des z. Pour les translations en y et les rotations autour de l'axe des x, il suffit de remplacer S_X par S_Y. Le tableau suivant donne les valeurs relatives des coefficients de pondération $\lambda_X, \lambda_Z, \lambda_{\theta_Y}$, et λ_{θ_Z}.

λ_X	λ_Z	λ_{θ_Y}	λ_{θ_Z}
$\frac{f}{S_X Z_C}\frac{p}{d_0}$	$\lambda_X \sin\phi$	$Z_C^2 \lambda_X$	$\lambda_{\theta_Y} \sin\phi$

où ϕ correspond à l'angle maximal que forment un point caractéristique et le centre de l'objet, du point de vue de la caméra. Si r est le rayon de l'objet en ce point, on a

$$\tan\phi = \frac{r}{Z_C}$$

et comme cet angle est en général petit, on utilisera l'approximation $\sin\phi \simeq \tan\phi$ pour connaître $\sin\phi$.

3.3.2 Nouvelle formulation pour la commande

Après l'étude précédente, on peut formuler plus précisément l'expression littérale du coût total engagé à chaque mouvement de la caméra:

$$E = E_{pix} + E_{mouv}$$

où E_{pix} est l'erreur (en pixels) commise au niveau des points caractéristiques sur l'image:

$$E_{pix} = \sum_{i=1}^{N} \left(\Delta x_i^2 + \Delta y_i^2\right) \tag{32}$$

et E_{mouv} représente le coût du mouvement effectué par la caméra:

$$E_{mouv} = N\lambda_X \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{S_X}{S_Y} & 0 \\ 0 & 0 & \sin\phi \end{pmatrix} \begin{pmatrix} d_X^2 \\ d_Y^2 \\ d_Z^2 \end{pmatrix} + N\lambda_{\theta_X} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \frac{S_X}{S_Y} & 0 \\ 0 & 0 & \sin\phi \end{pmatrix} \begin{pmatrix} \theta_X^2 \\ \theta_Y^2 \\ \theta_Z^2 \end{pmatrix}$$

où $d_j,\ j \in \{X, Y, Z\}$ est l'amplitude du mouvement en translation suivant l'axe j, et θ_j l'amplitude de l'angle de rotation autour de l'axe j. Le vecteur commande $\vec{u}_C(t)$ peut s'exprimer en fonction de ces paramètres comme:

$$\vec{u}_C(t) = \frac{1}{T} \begin{bmatrix} d_X(t) \\ d_Y(t) \\ d_Z(t) \\ \theta_X(t) \\ \theta_Y(t) \\ \theta_Z(t) \end{bmatrix}$$

où T est la période d'échantillonnage des mouvements. On peut donc écrire de manière plus compacte le coût des mouvements caméra au temps t:

$$E_{mouv}(t) = \vec{u}_C^T(t) W \vec{u}_C(t)$$

avec W la matrice 3×3 de pondération suivante:

$$W = NT^2\lambda_X \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{S_X}{S_Y} & 0 & 0 & 0 & 0 \\ 0 & 0 & \sin\phi & 0 & 0 & 0 \\ 0 & 0 & 0 & Z_{Cref}^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{S_X}{S_Y} Z_{Cref}^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & Z_{Cref}^2 \sin\phi \end{pmatrix}$$

De même, on peut exprimer l'erreur sur l'image en fonction du vecteur $I(t+1)$ qui découle de l'envoi de la commande $\vec{u}_C(t)$. D'après l'équation (32), E_{pix} est la somme sur tous les points caractéristiques, de l'erreur commise entre la position courante et la position de référence du point considéré:

$$E_{pix} = \left[I_{ref} - I(t+1)\right]^T \left[I_{ref} - I(t+1)\right]$$

On obtient donc la valeur du coût total engagé pour l'itération t:

$$E(t) = \left[I_{ref} - I(t+1)\right]^T \left[I_{ref} - I(t+1)\right] + \vec{u}_C^T(t) W \vec{u}_C(t) \tag{33}$$

Notre but est de minimiser l'énergie dépensée à l'itération t. Cette minimisation se fait par rapport à la commande $\vec{u}_C(t)$ envoyée au temps t: on cherche la commande "idéale". De ce fait, nous cherchons le $\vec{u}_C(t)$ qui annule

$$\frac{\partial E(t)}{\partial \vec{u}_C(t)}$$

Il nous faut pour cela exprimer le $I(t+1)$ apparaissant dans l'expression (33) en fonction de $\vec{u}_C(t)$. La relation (8) nous donne:

$$I(t+1) = \mathrm{B}(t)\vec{u}_C(t) + 2I(t) - I(t-1) - \mathrm{B}(t-1)\vec{u}_C(t-1) \tag{34}$$

on a donc:

$$\begin{aligned} \frac{\partial E(t)}{\partial \vec{u}_C(t)} &= 2\frac{\partial I(t+1)}{\partial \vec{u}_C(t)} \left[I(t+1) - I_{ref}\right] + 2W\vec{u}_C(t) \\ &= 2\mathrm{B}^T(t) \left[I(t+1) - I_{ref}\right] + 2W\vec{u}_C(t) \end{aligned}$$

Au minimum, on a

$$\frac{\partial E(t)}{\partial \vec{u}_C(t)} = 0$$

soit

$$\mathrm{B}^T(t)\left[I(t+1) - I_{ref}\right] + W\vec{u}_C(t) = 0$$

d'où, avec la valeur de $I(t+1)$ exprimée dans l'équation (34):

$$\vec{u}_C(t) = \left[\mathrm{B}^T(t)\mathrm{B}(t) + W\right]^{-1}\mathrm{B}^T(t)\left\{I_{ref} - 2I(t) + I(t-1) + \mathrm{B}(t-1)\vec{u}_C(t-1)\right\} \tag{35}$$

C'est la nouvelle expression de la commande. Elle est pondérée par une matrice de poids W qui décourage les grands mouvements et autorise de petites erreurs en pixels au niveau de l'image.

Remarque:
Si on fait $W = 0$ (matrice identiquement nulle: pas de pondération du coût des mouvements, c'est-à-dire qu'aucune erreur en pixels n'est autorisée sur l'image), alors

$$\left[\mathrm{B}^T(t)\mathrm{B}(t) + W\right]^{-1}\mathrm{B}^T(t) = \left[\mathrm{B}^T(t)\mathrm{B}(t)\right]^{-1}\mathrm{B}^T(t) = \mathrm{B}^+(t)$$

Ainsi pour $W = 0$, l'équation (35) correspond *exactement* à l'équation de commande (12) que nous avions développée au début du chapitre. On retrouve donc une certaine cohérence entre les deux expressions.

4 Approches neuronales du problème

Après avoir cherché à améliorer l'approche "classique" du problème, il nous a semblé intéressant d'étudier si des approches neuronales pouvaient fournir d'autres types de solutions possibles. Nous allons d'abord recenser les paramètres essentiels du problème, afin de définir quels seraient les couples entrée/sortie les meilleurs. Ensuite nous proposerons différentes soltions neuronales en fonction de ces couples entrée/sortie.

4.1 Paramètres essentiels du problème

La recherche des paramètres essentiels du problème ne peut se faire sans une étude de l'approche "classique" (par le calcul). On a vu dans l'étude précédente que l'équation (27) propose d'estimer la commande à appliquer au temps t par:

$$\vec{u}_C(t) = \mathrm{B}^+(t-1)\,[I_{ref} - 3I(t-1) + 2I(t-2)] + 2\vec{u}_C(t-1) - \mathrm{B}^+(t-1)\mathrm{B}(t-2)\vec{u}_C(t-2)$$

Parmi les paramètres indispensables à l'élaboration de la commande à appliquer au temps t, on peut donc d'ores et déjà mentionner $I(t-1), I(t-2), \vec{u}_C(t-1)$ et $\vec{u}_C(t-2)$. Pour une application donnée, c'est-à-dire pour une position de référence donnée, I_{ref} est constant. Nous ne sommes donc pas obligés de la mentionner. Cependant, on pourrait éventuellement le faire apparaître en remplaçant par exemple les paramètres $I(t-1)$ et $I(t-2)$ par $[I(t-1)-I_{ref}]$ et $[I(t-2)-I_{ref}]$ respectivement.

Dans une approche où l'on considère la matrice $\mathrm{B}(t)$ constante, il ne semble pas nécessaire d'ajouter d'autres paramètres en entrée du réseau. Si au contraire on envisage un approche à $\mathrm{B}(t)$ non constante, d'autres paramètres peuvent être ajoutés. Mentionnons par exemple $I(t-3)$ et $\vec{u}_C(t-3)$ qui apparaissent dans l'approche de Papanikolopoulos ainsi que dans notre approche d'estimation de ε_C filtré avec confiance.

On peut envisager d'autres moyens d'estimer les paramètres "manquants" $\varepsilon_C^i(t-1)$ et $\varepsilon_C^i(t-2)$. Une petite astuce décrite dans l'Annexe B permet par exemple de faciliter l'estimation de $\vec{u}_C(t)$ à $\mathrm{B}(t)$ non constante, sans estimer directement les paramètres ε_C. De plus cette procédure n'ajoute qu'un seul paramètre (scalaire) σ aux entrées du réseau. Notons cependant qu'elle est peu précise et permet seulement d'éviter les gros écarts en profondeur.

En résumé, on peut dire que les paramètres indispensables à une estimation neuronale sont:

$$I(t-1), I(t-2), \vec{u}_C(t-1), \vec{u}_C(t-2)$$

auxquels on peut éventuellement ajouter

$$I(t-3), \vec{u}_C(t-3), I_{ref}, \sigma$$

Les vecteurs $\vec{u}_C(.)$ comportent 6 composantes (3 rotations, 3 translations). Pour N points caractéristiques, les vecteurs $I(.)$ comportent $2N$ composantes. Nous avons donc un minimum de $4N+12$ entrées pour 6 sorties (les 6 composantes du $\vec{u}_C(t)$). Le nombre maximum d'entrées peut aller jusqu'à $4N+12+(4N+7) = 8N+19$. En vérité cette valeur n'est pas exhaustive car on pourrait encore ajouter $I(t-4), \vec{u}_C(t-4)$,

etc. Cependant il est clair que la taille du réseau importe, comme nous allons le constater ici.

4.2 Approche par le CMAC

4.2.1 Travaux antérieurs

Des recherches visant à utiliser le réseau d'Albus [1] ont déjà été menées par le passé [5], [6]. C'est Miller qui s'y est particulièrement intéressé. Dans un premier article [5], il relate des expériences conduites avec un robot 3 axes chargé de s'asservir sur un objet (un anneau) posé sur un convoyeur. Il s'agit d'un asservissement référencé capteur, puisque le robot doit déplacer la caméra qui est à l'extrémité de son dernier axe, de façon à ce que l'image de l'anneau vu par la caméra soit constamment égale à l'image de référence. Cette image de référence représente l'anneau placé au centre de l'image caméra, de façon à ce que le centre de cet anneau soit au centre optique de la caméra, et que le diamètre de l'anneau corresponde à la taille z_0 sous laquelle il est vu lorsque la caméra est à une distance d_o fixée de l'objet.

Dans un premier temps l'objet est fixe[9]. Les paramètres d'entrée (voir figure (6)) sont alors au nombre de 6: les 3 positions angulaires des axes au début de l'itération considérée, et les 3 changements désirés en position (s_x, s_y, s_z). Ces trois derniers paramètres sont calculés comme suit: considérons la figure (6) représentant le plan image à l'instant considéré. Le centre de l'anneau n'est pas exactement au centre de l'image. Soient dx et dy les écarts au centre en pixels, suivant les axes x et y. Soit $dz = z - z_0$ l'écart entre le diamètre apparent (en pixels) de l'anneau, par rapport au diamètre de référence z_0. Le vecteur $[dx, dy, dz]^T$ est transformé de manière à former un vecteur de norme unitaire $[sx, sy, sz]^T$ tel que:

$$\begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} = \rho \begin{bmatrix} sx \\ sy \\ sz \end{bmatrix}$$

où ρ est la norme du vecteur $[dx, dy, dz]^T$. Les sorties sont formées par les trois composantes du vecteur commande $\vec{u}_C$ à envoyer au robot à l'itération suivante. Ce vecteur est multiplié par ρ en sortie pour tenir compte de l'amplitude du mouvement à effectuer.

Cette opération de normalisation à 1 *avant* de passer dans le CMAC suivie d'une "remise à niveau" (multiplication par ρ) *après* être sorti du CMAC s'explique

[9] en fait le convoyeur sur lequel est posé l'objet est lui-même fixe

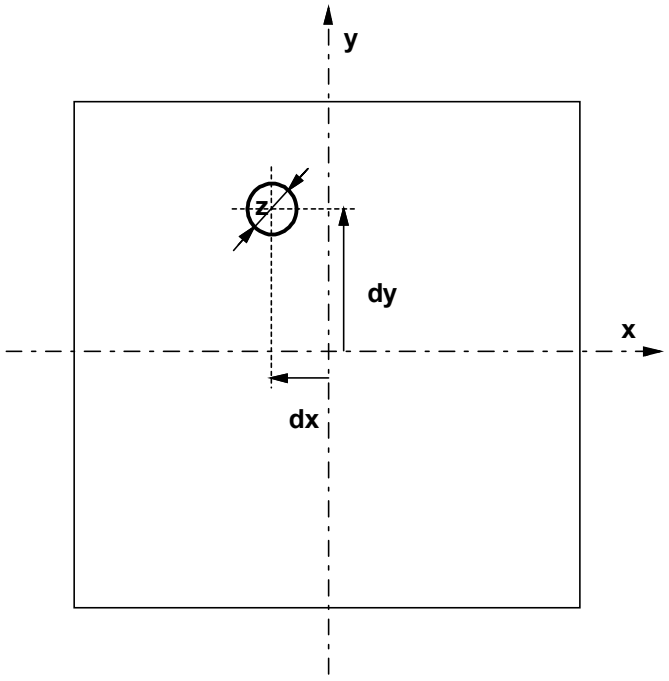

Figure 6: *Vue de l'anneau par la caméra à l'instant t*

par la façon dont le CMAC fonctionne. Rappelons qu'en entrée les composantes sont mises sous forme binaire ou $L - aire$ (voir Chapitre 1 Section 4). Ici, Miller choisit $L = 100$ pour chacune des composantes d'entrée. En bornant chaque sx, sy ou sz entre -1 et 1 (c'est l'effet de la normalisation à 1 du vecteur), Miller facilite la représentation des entrées. En effet, sans normalisation, les bornes de L-arisation imposées par l'opérateur pourraient s'avérer trop grandes (jamais atteintes) ou trop petites (souvent dépassées). En normalisant le vecteur d'entrée, Miller amène une meilleure "occupation du terrain". Il lui suffit ensuite de multiplier le vecteur sortie par la norme ρ pour redonner au problème sa dimension en amplitude.

Voyons maintenant la taille du réseau CMAC utilisé par Miller pour cette expérience; d'après les indication d'Albus, il faudrait choisir un nombre d'unités actives C tel que:

$$C \geq \frac{N \ln L}{\ln 99}$$

Ici, pour $N = 6$ entrés et $L = 100$ niveaux, on s'attendrait à ce que Miller choisisse C de l'ordre de 6 ou 7, d'où un nombre de poids de l'ordre de 600 à 700 par sortie. Miller ne semble pas partager les vues d'Albus sur ce point, car il utilise $C = 32$ unités actives par entrée, et 16.384 poids par sortie, ce qui est considérablement plus important que les suggestions d'Albus. En fin d'article, Miller présente de bons résultats pour son système, à la fois en apprentissage et en généralisation.

Dans le même article, Miller relate un deuxième type d'expérience. Il se place maintenant dans un cas où le convoyeur sur lequel est posé l'objet, se déplace lentement. L'objet possède donc désormais un mouvement propre. Pour résoudre ce problème, Miller ajoute 3 nouveaux paramètres aux 6 entrées précédemment utilisées pour son système: ce sont les variations des paramètres image à l'itération précédente. Il y a toujours les 3 mêmes sorties pour le CMAC.

Comme dans la première expérience, Miller utilise beaucoup plus d'unités actives et de poids que ne le conseille Albus. Pour cette expérience, chaque vecteur d'entrée sélectionne $C = 128$ unités, et il y a 65.536 poids par sortie. Sans vouloir contester les expériences de Miller, un tel nombre de poids semble quand même vraiment important. D'un autre côté, le CMAC n'est pas basé sur un principe d'adaptation à une base de données, comme peut l'être le perceptron multicouche, mais il fonctionne plutôt comme une mémoire à cases: chaque vecteur d'entrée active un groupe de cases différent, apportant ainsi une réponse différente. De ce fait, plus la dimensionnalité des entrées est importante, plus le nombre de cases mémoire doit être élevé, ce qui explique le grand nombre de poids utilisé par Miller.

Dans un deuxième article [6], Miller utilise un robot à 4 axes, également chargé de suivre un objet fixe sur un convoyeur en mouvement. L'objet en question est un rasoir jetable dont la position sur l'image caméra peut être définie par 4 paramètres (voir figure (7)).

Les quatre paramètres sont les coordonnées X et Y (en pixels) du centroïde du rasoir sur l'image, la longueur Z (en pixels) du rasoir, et son orientation $R = Z\cos\theta$ par rapport à l'horizontale. La tâche du système d'asservissement est d'amener le rasoir dans une position de référence par rapport à la caméra. Pour cela, on cherche à la fois à rendre la taille du rasoir égale à celle qu'il a lorsque la distance caméra/objet vaut la distance de référence d_0, et à mettre le manche du rasoir en position parallèle à l'axe horizontal. Du point de vue des paramètres image, cela donne $Z = Z_0$ et $R = Z_0$. Une fois dans cette position, Miller cherche à faire suivre une trajectoire[10] prédéfinie au centroïde du rasoir.

Un CMAC possédant 12 entrées et 4 sorties est chargé de cet asservissement visuel. Les 12 entrées sont: les quatre positions des axes du robot au début de l'itération, les quatre coordonnées (X, Y, Z, R) du rasoir dans l'image, et les quatre changements désirés de ces paramètres (dX, dY, dZ, dR). dX (respectivement dY) est l'écart entre la coordonnée actuelle X (respectivement Y) et la coordonnée désirée X_0 (respectivement Y_0) du centroïde dans l'image. De même, $dZ = Z - Z_0$, où Z_0 est la longueur désirée du rasoir (correspondant à une distance d_0 entre l'objet

[10] tâche de haut niveau

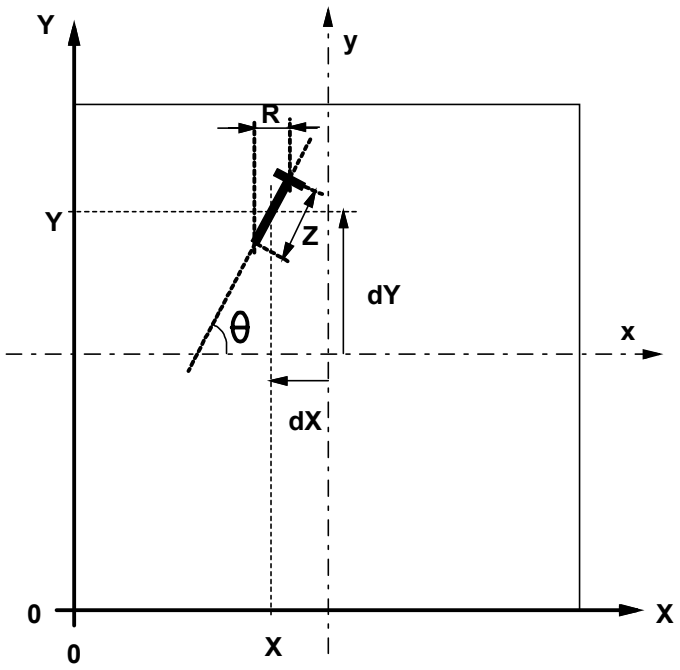

Figure 7: *Vue du rasoir par la caméra à l'instant t*

et la caméra). Enfin on a $dR = R - Z_0$. Les sorties sont les commandes à envoyer au robot à l'instant suivant.

Miller utilise $C = 64$ unités actives par entrée, et 16.384 poids par sortie. Ici encore, nous constatons que pour obtenir des résultats satisfaisants, Miller utilise un CMAC de grandes dimensions.

En réalité, malgré les bons résultats obtenus par Miller dans les différentes expériences qu'il présente, cette approche pourrait montrer certaines limites. Tout d'abord, notons que Miller n'a étudié que les cas "relativement" simples d'un robot à 3 ou 4 degrés de liberté seulement, au lieu de 6 degrés de liberté en ce qui nous concerne. Cela entraîne une simplification considérable du problème, car il n'y a pas de rotation 3D dans les problèmes de Miller. Tout au plus y a-t-il, dans le problème à 4 degrés de liberté, possibilité de rotation autour de l'axe des Z. Mais cette rotation s'effectue de toute façon *sans* déformation de l'objet, puisque l'objet est plan, posé sur un support plan, et que la cameŕa reste (par construction du dispositif) constamment perpendiculaire au plan objet. De ce fait Miller n'a pas abordé le problème des rotations 3D, alors que ce sont *justement* les rotations 3D qui posent les plus grandes difficultés dans ce problème d'asservissement. Cette partie reste donc pour l'instant un problème ouvert.

Un autre point que n'aborde pas l'approche de Miller, du fait même de sa simplicité,

est l'estimation de la profondeur des points de l'objet. En effet, dans ces approches à 3 et 4 degrés de liberté, la profondeur des points est directement *calculable* par le système. Ainsi, puisque l'objet est plan et que la caméra est toujours perpendiculaire au plan de l'objet (par construction du dispositif expérimental), la taille apparente du rasoir (ou de l'anneau) dans l'image est inversement proportionnelle à la profondeur des points. Il est de fait très facile de calculer la profondeur des points: il n'y a donc pas besoin de l'*estimer*. Au contraire, dans notre approche à 6 degrés de liberté, les profondeurs ne peuvent pas être calculées directement: elles ne peuvent être obtenues que par estimation.

4.2.2 Approche proposée

Dans le problème qui nous intéresse, on a vu que les paramètres importants étaient $I(t-1), I(t-2), \vec{u}_C(t-1), \vec{u}_C(t-2)$ ainsi que d'autres paramètres supplémentaires éventuels ($I(t-3), \vec{u}_C(t-3), \sigma$, etc). Pour les quatre vecteurs considérés, cela fait en tout $4N+12$ entrées, sachant que le nombre de points caractéristiques N doit être au moins supérieur à 3. En général, on prend $N \geq 5$, ce qui fait ici un nombre d'entrées supérieur ou égal à 32 pour ce problème avec un robot 6 axes. C'est environ trois fois plus important que le nombre de paramètres d'entrées utilisé par Miller lors de son expérience avec un robot 4 axes.

Si on calcule le nombre minimal d'unités actives par entrée (suivant les instructions d'Albus), on trouve:

$$C \geq \frac{(4N+12)\ln L}{\ln 99}$$

Si on sépare les paramètres d'entrées en $L = 20$ niveaux chacun, on arrive à $C \geq 21$ pour $N = 5$ points caractéristiques, et $W = 2100$ poids par sortie. Le véritable problème qui se pose alors, est que le nombre d'entrées possibles est très important. De ce fait, la base d'apprentissage du CMAC doit être très grande pour pouvoir tenir compte de la grande diversité des entrées. Même en construisant une grande base d'apprentissage, on n'est pas sûr que *tous* les poids seront adaptés au mieux (parfois certains poids ne sont jamais sélectionnés lors de l'apprentissage).

D'autre part, pour améliorer les performances, on pourrait être tenté d'augmenter le nombre de niveaux de quantification des entrées, et de prendre L aussi grand que possible: $L = 100$ comme Miller, par exemple. Mais pour L très grand, on arrive à un nombre de poids si grand qu'il semble vraiment improbable que la base d'apprentissage arrive à atteindre tous ces poids, aussi importante soit-elle. On peut alors raisonnablement penser qu'en phase de test, les poids qui n'ont pas été sélectionnés à l'apprentissage ne donneront pas des réponses intéressantes.

Nous avons contruit une base d'aprentissage de 2000 exemples. Le réseau CMAC présente d'excellentes performances sur sa base d'apprentissage. Peu d'itérations sont d'ailleurs nécessaires pour faire converger le réseau. Toutefois, en phase de test, les résultats obtenus sont très médiocres: le système diverge facilement.

Nous avons essayé d'expliquer ces résultats. Il ne nous semble pas que le CMAC soit bien adapté pour des problèmes de grandes dimensions. Notons par exemple qu'en suivant les conseils d'Albus nous obtenons un réseau déjà grand, et qui pourtant ne semble pas suffire pour couvrir l'étendue du problème. Si nous avions suivi l'exemple de Miller, qui partitionne les entrées sur $L = 100$ niveaux, nous aurions 100^{32} entrées possibles! Quant au choix du nombre d'unités actives par entrée, avec $C = 21$ nous avons déjà un adressage complexe. Si on prenait $C = 128$ unités actives par entrée (comme Miller dans son premier article avec un robot 3 axes), on arriverait à un système vraiment lourd. En outre, avec un nombre de poids par sortie avoisinant les 65.000 (cf Miller dans [5]), on aurait un CMAC réellement gigantesque. Par rapport au nombre de poids utilisés, il ne nous semble pas que le CMAC soit une réponse idéale au problème que nous nous posons. Devant un tel choix il semble préférable de revenir à la résolution "classique" du problème par le calcul. Les bons résultats obtenus par Miller avec le CMAC sont probablement dûs au fait qu'il s'est restreint à un problème très simplifié (3 ou 4 degrés de liberté seulement).

4.3 Approche par un PMC

4.3.1 Structure du réseau

En même temps que nous avons lancé nos recherches en direction du CMAC, nous avons étendu nos investigations vers une solution utilisant un perceptron multicouche [9], [10]. C'est aussi à cause des bons résultats obtenus par le PMC (avec pourtant très peu de poids en comparaisons du CMAC) que nous avons décidé de ne pas poursuivre les recherches sur le CMAC.

Nous avons d'abord axé nos recherches sur un modèle simple de PMC: un modèle deux couches, ne comportant donc qu'une seule couche de poids adaptatifs. Nous avons également utilisé les entrées les plus simples: $I(t-1), I(t-2), \vec{u}_C(t-1)$ et $\vec{u}_C(t-2)$, en nous disant que nous voulions créer un réseau neuronal ayant au moins les performances d'un système "classique" considérant la matrice B constante. Or pour B constante, on sait que la formule de la commande se simplifie beaucoup: l'équation (27) devient

$$\vec{u}_C(t) = \mathrm{B}^+ \left[I_{ref} - 3I(t-1) + 2I(t-2) \right] + 2\vec{u}_C(t-1) - \vec{u}_C(t-2)$$

On observe qu'il est possible d'inclure I_{ref} dans les vecteurs $I(t-1)$ et $I(t-2)$ de façon astucieuse, sans avoir à augmenter le nombre de paramètres d'entrée. En effet, si on nomme $J(t-1) = I(t-1) - I_{ref}$ et $J(t-2) = I(t-2) - I_{ref}$, alors on a:

$$\vec{u}_C(t) = \mathrm{B}^+ \left[2J(t-2) - 3J(t-1)\right] + 2\vec{u}_C(t-1) - \vec{u}_C(t-2)$$

c'est-à-dire que dans le calcul de $\vec{u}_C(t)$ n'apparaissent plus que quatre vecteurs dépendant du temps (B est considérée constante donc il ne semble pas nécessaire de l'inclure dans les entrées).

Les nouveaux vecteurs entrée sont $J(t-1), J(t-2), \vec{u}_C(t-1)$ et $\vec{u}_C(t-2)$, avec $J(t-i) = I(t-i) - I_{ref}$, pour $i \in \{1,2\}$. On a exactement $4N+12$ entrées et 6 sorties. Pour un perceptron à deux couches, cela fait $6 \times (4N+12)$ poids. Pour $N = 5$ par exemple, on arrive à $6 \times 32 = 192$ poids, chiffre à comparer à $6 \times 2100 = 12.600$ poids (au minimum!) pour le CMAC. Il y a de façon évidente une différence d'ordre de grandeur énorme entre les deux réseaux, qui nous pousse à utiliser un PMC plutôt qu'un CMAC.

4.3.2 Base d'apprentissage

Notre base d'apprentissage est composée de couples entrée/sortie où chaque sortie est formée par le vecteur $\vec{u}_C(t)$ adapté au quadruplet $(J(t-1), J(t-2), \vec{u}_C(t-1), \vec{u}_C(t-2))$ qui lui a donné naissance[11]. Nous allons décrire ici la façon dont les exemples de cette base d'apprentissage sont générés.

Avant de commencer les calculs, rappelons que pour tout couple de référentiels R_0 et R_1, la matrice de transformation entre R_0 et R_1 est la matrice 4×4 définie par:

$$M_{R_0}^{R_1} = \begin{pmatrix} & R & & T \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

où R est la matrice 3×3 de rotation entre les axes de R_0 et ceux de R_1, et T est le vecteur de translation entre l'origine de R_0 et l'origine de R_1.

$I(t-2)$ est la projection des points caractéristiques de l'objet sur le plan image à l'instant $t-2$. Par conséquent, pour calculer $I(t-2)$, nous avons besoin de $M_{co}(t-2)$, la matrice de transformation entre le référentiel objet et le référentiel caméra (voir figure (8)). On suppose que $I(t-2)$ et I_{ref} sont proches [12], donc $M_{co}(t-2)$ peut être considérée comme une version "bruitée" de la matrice correspondant à la position de

[11] avec $J(t-i) = I(t-i) - I_{ref}$, pour $i \in \{1,2\}$

[12] c'est-à-dire que le système d'asservissement n'a pas de dysfonctionnements graves

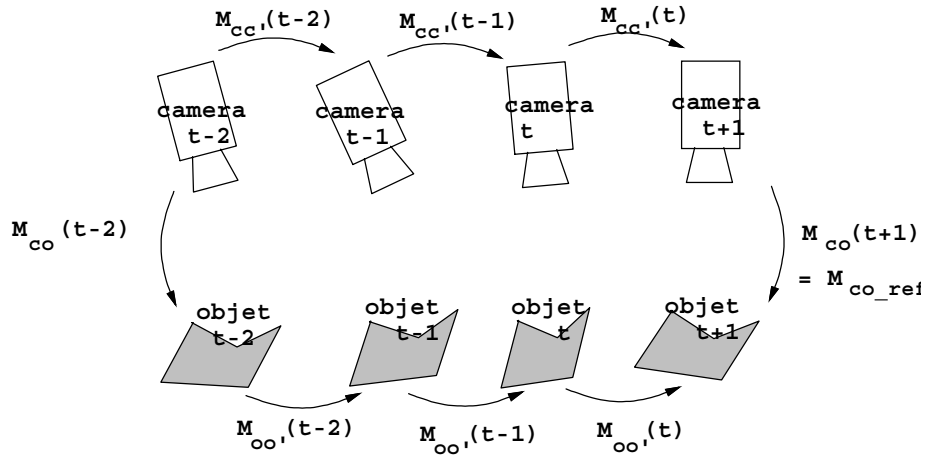

Figure 8: *Evolution des positions relative entre l'objet et la caméra au cours du temps*

référence M_{co_ref}:

$$M_{co}(t-2) = M_{co\_ref} M_u(t-2)$$

où $M_u(t-2)$ est une matrice aléatoire de "bruit" générée par le processus suivant: un vecteur "bruit" de rotation et de translation $\vec{n}_u(t-2)$ est généré aléatoirement par tirage uniforme. C'est à partir de ce vecteur $\vec{n}_u(t-2)$ que l'on calcule la matrice $M_u(t-2)$ en utilisant la formule de Rodrigues [11] (voir Annexe C pour ce calcul).

Une fois que cette position de départ est calculée, un premier vecteur vitesse de translation et de rotation de l'objet $U_0(t-2)$ est tiré au hasard (uniformément). La matrice de transformation de l'ancien vers le nouveau référentiel objet, $M_{oo'}(t-2)$, est calculée à partir de $U_0(t-2)$ grâce à la formule de Rodrigues.

La caméra est alors supposée suivre ce mouvement de façon à se maintenir en position de référence relativement à l'objet: la représentation idéale de la matrice $M_{co}(t-1)$ est donc M_{co_ref} et on a:

$$M_{cc'}^{ideal}(t-2) = M_{co}(t-2) M_{oo'}(t-2) M_{co\_ref}^{-1} \tag{36}$$

De cette matrice idéale $M_{cc'}^{ideal}(t-2)$ on tire la commande de la caméra à l'instant $t-2$: $U_C(t-2)$. Cette opération est effectuée en utilisant l'inverse de la formule de Rodrigues. Toutefois, puisque les mouvements réels sont rarement parfaits, on ajoute un petit vecteur aléatoire de bruit à $U_C(t-2)$:

$$\vec{u}_C(t-2) = U_C(t-2) + \vec{n}_C(t-2)$$

C'est à partir du vecteur (bruité, mais réaliste) $\vec{u}_C(t-2)$ que nous recalculons la vraie

matrice $M_{cc'}(t-2)$ (toujours à l'aide de la formule de Rodrigues). L'équation (36) est alors réarrangée pour faire apparaître la nouvelle relation entre les référentiels objet et caméra:

$$M_{co}(t-1) = M_{cc'}^{-1}(t-2)M_{co}(t-2)M_{oo'}(t-2)$$

De la matrice $M_{co}(t-1)$ nous tirons $I(t-1)$ par simple projection des points caractéristiques de l'objet dans la plan image.

L'intégralité de ce processus est répétée à l'itération suivante (allant du temps $t-1$ au temps t). Cependant, cette fois-ci une contrainte est appliquée, qui provient de simples considérations physiques: pour un objet se mouvant à une vitesse raisonnable, et pour un échantillonnage temporel suffisant, il semble probable que la vitesse instantanée de l'objet ne puisse pas varier de manière importante entre deux itérations. En conséquence on suppose que le nouveau vecteur vitesse de l'objet $U_0(t-1)$ obéit à l'égalité suivante:

$$U_0(t-1) = U_0(t-2) + \vec{d_0}(t-1)$$

où $\vec{d_0}(t-1)$ est un vecteur de bruit tiré au hasard entre des limites valant le dixième des bornes utilisées lors du tirage de $U_0(t-2)$.

Sans répéter toute la procédure, disons que de $U_0(t-1)$ on tire $M_{oo'}(t-1)$ par Rodrigues. On calcule alors une matrice "idéale"

$$M_{cc'}^{ideal}(t-1) = M_{co}(t-1)M_{oo'}(t-1)M_{co\_ref}^{-1}$$

de laquelle on extrait $U_C(t-1)$ par l'inversion de la formule de Rodrigues. A ce vecteur commande "idéal" $U_C(t-1)$ on ajoute un petit vecteur de bruit:

$$\vec{u}_C(t-1) = U_C(t-1) + \vec{n}_C(t-1)$$

d'où le calcul du vrai $M_{cc'}(t-1)$. On en tire

$$M_{co}(t) = M_{cc'}^{-1}(t-1)M_{co}(t-1)M_{oo'}(t-1)$$

On recommence une dernière fois tout le processus (intervalle allant de t à $t+1$): un vecteur de bruit $\vec{d_0}(t)$ est tiré au hasard et on a

$$U_0(t) = U_0(t-1) + \vec{d_0}(t)$$

d'où le calcul de la matrice de mouvement de l'objet $M_{oo'}(t)$. Remarquons ici que puisque le réseau est supposé être entraîné à répondre parfaitement aux divers mouvements de l'objet, aucun bruit supplémentaire ne sera ajouté au vecteur

commande “idéal" de la caméra extrait de la matrice

$$M_{cc'}(t) = M_{co}(t)M_{oo'}(t)M^{-1}_{co\_ref}$$

De ce fait $\vec{u}_C(t)$ est calculé directement à partir de $M_{cc'}(t)$ par l'inversion de la formule de Rodrigues.

Nous avons maintenant en main tous les paramètres (obtenus de manière aléatoire) nécessaires à l'entraînement du réseau: les entrées $J(t-2)$ et $J(t-1)$ (obtenues via $I(t-2)$ et $I(t-1)$), ainsi que $\vec{u}_C(t-2)$ et $\vec{u}_C(t-1)$; et également la sortie $\vec{u}_C(t)$. Le choix des paramètres dans des directions aléatoires garantit l'adaptativité du réseau à n'importe quel mouvement de l'objet. En outre, le fait que la vitesse de l'objet puisse varier (les vitesses $U_0(.)$ aux temps $t, t-1$ et $t-2$ sont différentes) devrait donner la possibilité de suivre un objet même lorsque sa vitesse varie (mouvements non-uniformes).

4.3.3 Possibilités d'extension

Nous venons de montrer comment construire la base d'apprentissage pour un PMC recevant $J(t-1), J(t-2), \vec{u}_C(t-1), \vec{u}_C(t-2)$ en entrée et fournissant $\vec{u}_C(t)$ en sortie. Nous avions précédemment mentionné un PMC à deux couches au départ. Il est bien évident que cette même base peut servir à entraîner des réseaux à trois couches (voire plus).

Nous pouvons aussi imaginer de revenir encore plus en arrière dans la création de la base d'apprentissage et d'ajouter explicitement $I(t-3)$ et $\vec{u}_C(t-3)$ aux entrées du réseau (les algorithmes “classiques" supposant B non constante n'utilisaient $I(t-3)$ et $\vec{u}_C(t-3)$ que de manière implicite, pour l'estimation de ε_C). Le plus intéressant, en réalité, est que contrairement aux approches dites “classiques", notre démarche n'est pas figée: nous pouvons utiliser $I(t-3)$ sans $\vec{u}_C(t-3)$ ou réciproquement; nous pouvons utiliser l'astuce de l'Annexe B et ajouter le paramètre scalaire σ aux entrées; nous pouvons même utiliser $I(t-4)$ et $\vec{u}_C(t-4)$ si nous le désirons. Il nous est aussi possible de changer la forme ou l'expression des paramètres d'entrée, si nous pensons qu'une autre forme peut mieux convenir (utilisation d'un algorithme d'Analyse en Composantes Principales, par exemple, pour éliminer les dimensions superflues ou n'apportant que très peu d'énergie).

5 Résultats et comparaisons

La première partie des expérimentations s'intéresse aux mouvements uniformes:

l'objet est en mouvement de translation ou de rotation à une vitesse constante, et nous utilisons le critère d'erreur suivant:

$$e_{im} = \frac{1}{m}\sum_{n=1}^{m}\sqrt{\frac{1}{N}\|I(n) - I_{ref}\|^2}$$

e_{im} est calculé comme la moyenne du carré de l'erreur entre la position courante et la position de référence des $N = 5$ points caractéristiques sur m instants d'échantillonnage. L'objet utilisé est schématisé en figure (9). Il est important de

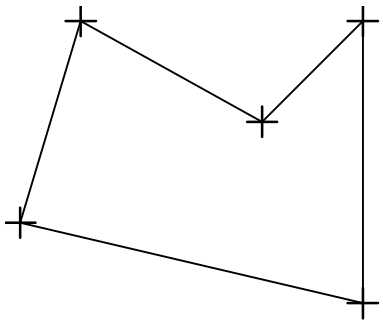

Figure 9: *Positions des points caractéristiques de l'objet.*

noter que, dans le but de présenter des expériences plus réalistes, les pixels dans l'image $I(n)$ sont arrondis à la valeur *paire* la plus proche (au lieu de l'arrondi habituel à l'entier le plus proche). De plus, un bruit uniforme tiré aléatoirement entre -1 et 1 est ajouté sur les coordonnées en pixels, de façon à approcher au mieux les défauts d'un traitement d'image réaliste. Afin de montrer les performances des méthodes dans plusieurs types de conditions, plusieurs vitesses d'objet ont été testées. La figure (10) représente l'erreur moyenne en pixels en fonction de la vitesse de translation de l'objet le long de l'axe des x (en cm/s). La fréquence d'échantillonnage est de 4 échantillons par seconde, et chaque point sur la courbe représente une moyenne sur $m = 200$ instants d'échantillonnage. La même expérience est réalisée pour une rotation uniforme de l'objet autour de l'axe des x dans son propre repère. Comme dans le cas de la translation, on peut noter que les résultats du réseau de neurones sont très prometteurs, puisqu'il sont légèrement meilleurs que ceux de l'approche classique à B constante. La méthode à B non constante avec amélioration de l'estimateur en Z_C et filtrage avec confiance montre elle aussi de bonnes performances.

On s'intéresse aussi au cas où les mouvements de l'objet sont non uniformes. Dans la première expérience de mouvements non uniformes, la position de l'objet selon l'axe des x est une fonction sinusoïdale de la forme

$$x_{pos}(n) = A\ sin(2\pi f n)$$

où n est le nombre d'itérations. L'amplitude A est fixée à 30cm, alors que la fréquence f varie. Pour chaque fréquence, un essai est réalisé sur 1000 instants

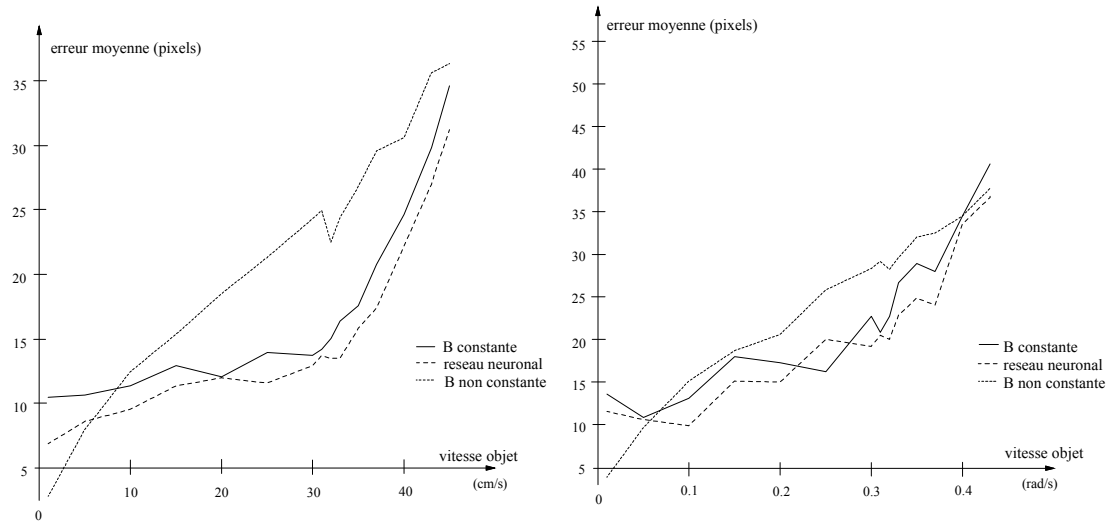

Figure 10: *Performances comparées des trois méthodes d'asservissement dans le cas de vitesses objet uniformes (a) en translation, (b) en rotation.*

d'échantillonnage. Les courbes montrant les erreurs moyennes en pixels sur 1000 instants d'échantillonnage sont présentées en figure (11a). La validité de l'approche neuronale pour des rotations non uniformes est également illustrée: des rotations

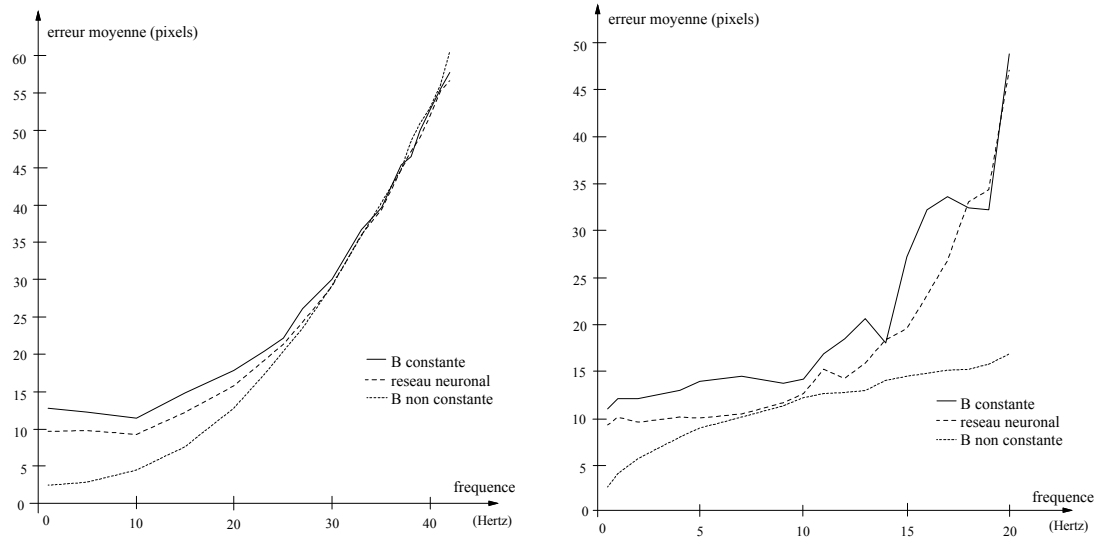

Figure 11: *Performances comparées des trois méthodes d'asservissement dans le cas de vitesses objet non uniformes (a) en translation sinusoïdale, (b) en rotation sinusoïdale.*

sinusoïdales autour de l'axe des x sont montrées en figure (11b). L'angle de rotation

instantané θ est tel que:

$$\theta(n) = \frac{\pi}{3} sin(2\pi f n)$$

Les mouvements sinusoïdaux (donc non uniformes) de l'objet semblent être aussi bien maîtrisés par le réseau de neurones que les mouvements uniformes. En ce qui concerne la translation non uniforme, l'approche neuronale se comporte mieux que l'approche à B constante, et cette tendance est visible surtout lorsque les déplacements ne sont pas trop grands. Notons cependant que dans ce cas, l'erreur relative croît beaucoup plus vite lorsque l'erreur absolue augmente. Pour des déplacements plus importants, les trois méthodes semblent se valoir. Enfin, mentionnons que la méthode améliorée à B non constante que nous proposons montre des performances sensiblement meilleures que les deux autres méthodes, ce qui semble normal puisqu'elle peut en principe mieux s'adapter à tout changement des paramètres.

En ce qui concerne la rotation non uniforme (sinusoïdale), on voit que, comme pour le translation non uniforme, le réseau de neurones fait légèrement mieux que la méthode à B constante. Par ailleurs, on peut noter la bonne performance de notre approche améliorée à B non constante: alors que la rotation non uniforme semble a priori le cas le plus difficile à traiter, cette approche montre une erreur assez faible, et, surtout, cette erreur a une pente beaucoup moins importante que les autres approches.

6 Conclusion

Dans ce Chapitre dédié à la commande référencée capteur, nous avons voulu mettre à plat les équations régissant le système, afin de mettre en valeur les paramètres importants. Nous avons alors étudié plusieurs solutions existantes: la solution à B constante, simple d'emploi, a montré à la Section précédente qu'elle peut fournir des performances relativement convenables. L'approche à B non constante qui était proposée par Papanikolopoulos ne nous a pas semblé convenir, car elle nécessite de connaître des paramètres difficilement appréhendables. Nous proposons donc une approche améliorée à B non constante, avec un estimateur stable sur la profondeur des points. En outre, nous proposons de pondérer les coûts des mouvements de la caméra par rapport aux erreurs en pixels sur l'image, de manière à décourager les grands mouvements. La méthode a été testée avec succès pour des mouvements uniformes et non uniformes. Sa supériorité sur l'approche à B constante est cependant plus marquée dans le cas de mouvements non uniformes.

Enfin, nous avons cherché à mettre en œuvre deux méthodes neuronales différentes. La première, basée sur l'utilisation du CMAC d'Albus, a été employée avec succès par Miller pour des mouvements à 3 et 4 degrés de liberté. Par contre, lorsque l'on cherche à l'employer pour des mouvements à 6 degrés de liberté comme ici, les calculs

de dimensionnalité montrent que le CMAC adapté à ce problème devrait être de très grande taille si l'on désire des performances acceptables.

L'approche utilisant un Perceptron MultiCouche est plus aisée: nous mettons en œuvre un tel modèle auquel nous envoyons les paramètres que nous avons jugés essentiels à la résolution du problème. La base d'entraînement du perceptron est composée de mouvements bruités, pour simuler ce qui peut se produire dans la réalité. Les courbes d'erreur présentées à la Section précédente montrent que cette approche réagit relativement mieux que l'approche à B constante. Cependant, on note que la meilleure approche reste la méthode à B non constante utilisant un estimateur avec confiance de la profondeur des points de l'objet.

Il faudrait sans doute chercher à représenter les paramètres image de façon plus concise en entrée du réseau de neurones. Par exemple, on pourrait songer à remplacer les vecteurs $I(t-1)$ et $I(t-2)$ par les coefficients de la transformation qui transforme $I(t-1)$ en $I(t-2)$. En jouant sur de tels concepts, peut-être serait-il possible de rendre les informations que reçoit le réseau plus pertinentes et mois bruitées. C'est un axe de recherche à ne pas négliger.

Annexe A: Instabilité de l'estimateur proposé à l'équation (30)

L'équation (30) propose un estimateur du paramètre $\varepsilon_C^i(t)$ représentant la profondeur du point P de l'objet dans le repère caméra:

$$\hat{\varepsilon}_C^i(t) = \; H_i^+(t)\left\{\mathrm{B}_i(t-1)\vec{u}_C(t-1) - \mathrm{B}_\omega^i(t)\vec{\omega}(t) + I_i(t+1) - 2I_i(t) + I_i(t-1)\right\}$$

avec $\vec{u}_C(.) = [\vec{V}(.)\ \vec{\omega}(.)]^T$.

Dans le cas où la caméra n'effectue pas de rotation, cette équation se simplifie dès lors que $\vec{\omega}(t) = \vec{\omega}(t-1) = \vec{0}$:

$$\hat{\varepsilon}_C^i(t) = \; H_i^+(t)\left\{\hat{\varepsilon}_C^i(t-1)H_i(t-1) + I_i(t+1) - 2I_i(t) + I_i(t-1)\right\}$$

Si de plus, le mouvement de la caméra est une simple translation sur l'axe des x, alors on a:

$$\vec{V}(t) = \begin{bmatrix} x_i^\bullet(t) \\ 0 \\ 0 \end{bmatrix}$$

d'où

$$H_i(t) = G_i(t)\vec{V}(t) = \begin{bmatrix} -Tx_i^\bullet(t) \\ 0 \end{bmatrix} \quad \text{et} \quad H_i(t-1) = \begin{bmatrix} -Tx_i^\bullet(t-1) \\ 0 \end{bmatrix}$$

On calcule la pseudoinverse de $H_i(t)$:

$$H_i^+(t) = \frac{1}{Tx_i^\bullet(t)} \begin{bmatrix} -1 & 0 \end{bmatrix}$$

d'où:

$$H_i^+(t)H_i(t-1) = \frac{x_i^\bullet(t-1)}{x_i^\bullet(t)}$$

Notre équation se simplifie alors en:

$$\hat{\varepsilon}_C^i(t) = \hat{\varepsilon}_C^i(t-1)\frac{x_i^\bullet(t-1)}{x_i^\bullet(t)} - \frac{1}{Tx_i^\bullet(t)}\left[x_i(t+1) - 2x_i(t) + x_i(t-1)\right]$$

avec $\hat{\varepsilon}_C^i(.) = \varepsilon_C^i(.) + \Delta\varepsilon_C^i(.)$.

Si cette équation d'estimation est *exacte*, alors cela signifie que

$$\varepsilon_C^i(t) = \varepsilon_C^i(t-1)\frac{x_i^\bullet(t-1)}{x_i^\bullet(t)} - \frac{1}{Tx_i^\bullet(t)}\left[x_i(t+1) - 2x_i(t) + x_i(t-1)\right]$$

En retranchant cette équation à l'équation précédente, on obtient alors

$$\boxed{\Delta\varepsilon_C^i(t) = \frac{x_i^\bullet(t-1)}{x_i^\bullet(t)}\Delta\varepsilon_C^i(t-1)}$$

où $\Delta\varepsilon_C^i(.)$ est l'erreur commise sur l'estimation de $\varepsilon_C^i(.)$.

Si on se trouve dans un mouvement d'accélération dirigé vers la droite, on a $0 < x_i^\bullet(t) < x_i^\bullet(t-1)$. On a alors:

$$\Delta\varepsilon_C^i(t) > \Delta\varepsilon_C^i(t-1)$$

c'est-à-dire que l'erreur commise sur ε_C^i croît de plus en plus. Cette situation apparaît aussi dans un cas de mouvement de décélération dirigé vers la gauche, puisque l'on a alors $x_i^\bullet(t-1) < x_i^\bullet(t) < 0$.

On en conclut que ce système est instable.

Annexe B: Un nouveau moyen pour paramétriser la profondeur des points de l'objet

On a vu, dans tout ce chapitre, que l'estimation de la profondeur des points de l'objet (distance à la caméra) était relativement difficile à réaliser avec une seule caméra.

On propose ici un moyen simple (mais peu précis) permettant de fournir un paramètre proportionnel à la profondeur.

Considérons la figure (12) représentant un point de l'objet d'abscisse X_C et sa

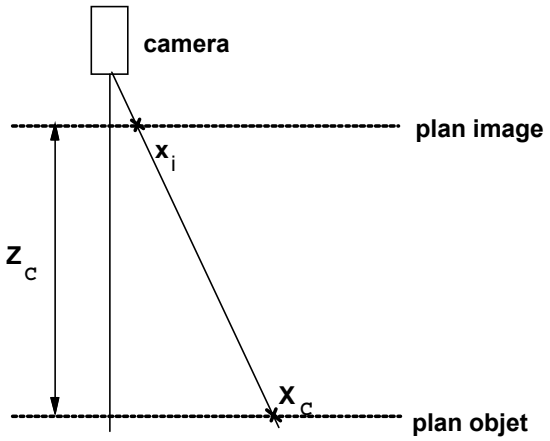

Figure 12: *Vue de profil du système*

projection x_i dans l'image. D'après la relation de Chasles, on a, pour tout point caractéristique de l'objet:

$$x_i = \frac{f}{S_X}\frac{X_C}{Z_C}$$

Si on suppose que la profondeur des points caractéristiques est à peu près la même pour tous les points, alors le barycentre des points vérifie la même relation, d'où:

$$\sigma_{x_I}^2 = \left(\frac{f}{S_X}\right)^2 \frac{\sigma_{X_C}^2}{Z_C^2}$$

avec

$$\sigma_{x_I}^2 = \frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N} \quad \text{où} \quad \bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$

De ce fait Z_C vérifie

$$Z_C = \frac{f}{S_X}\frac{\sigma_{X_C}}{\sigma_{x_I}}$$

Or l'objet est rigide, donc σ_{X_C} est constante. Si l'objet reste constamment perpendiculaire à la caméra, on peut donc écrire

$$Z_C \equiv \frac{1}{\sigma_{x_I}}$$

En réalité cette relation de proportion existe autant sur l'axe des x que sur l'axe des y, donc pour améliorer la fiabilité de cet estimateur on écrit, de façon plus générale:

$$Z_C \equiv \frac{1}{\sigma_I}$$

où

$$\sigma_I = \sqrt{\sigma_{x_I}^2 + \sigma_{y_I}^2}$$

Ainsi, à chaque fois que l'on disposera des paramètres images $I(t)$, il sera possible de calculer le paramètre σ_I proportionnel à l'inverse de Z_C. Ce paramètre ne peut pas facilement être utilisé dans une méthode classique, mais il est utilisable de manière directe dans un PMC, en tant que paramètre d'entrée supplémentaire. Il faut toutefois ne pas oublier que la relation entre Z_C et σ_I n'est qu'approximative, et est d'autant moins vraie que l'objet s'écarte de la position de référence dans le repère caméra.

Le paramètre σ_I représente l'écart-type de la distance dans l'image entre les points caractéristiques et leur barycentre.

Annexe C: Formule de Rodrigues

Pour un temps unité, la matrice de transformation M_u associée au vecteur vitesse de rotation et translation $\vec{u} = [\vec{t}, \vec{\omega}]^T$ est

$$M_u = \begin{pmatrix} & R_u & & t \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

où la matrice de rotation R_u est calculée à l'aide de la formule de Rodrigues [11]:

$$R_u = I + \sin\theta\, U + (1 - \cos\theta)\, U^2$$

où

- I est la matrice identité 3×3
- θ est la norme (longueur) du vecteur rotation $\vec{\omega}$:

$$\theta = \|\vec{\omega}\|$$

- U est la matrice 3×3 anti-symétrique définie par:

$$U = \begin{pmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{pmatrix}$$

  telle que

$$\vec{v} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

  soit le vecteur unitaire de direction $\vec{\omega}$:

$$\vec{v} = \frac{\vec{\omega}}{\|\vec{\omega}\|}$$

References

[1] J.S. Albus, "A New Approach to Manipulator Control: Cerebellar Model Articulation Controller (CMAC)" Transactions of the ASME, Journal of Dynamic Systems, Measurements and Control, pp 220-227, Sept 1975

[2] Bernard Espiau, François Chaumette, et Patrick Rives, "A New Approach to Visual Servoing in Robotics", IEEE Transactions on Robotics and Automation, Vol. 8, $n^0$3, June 1992

[3] François Chaumette, "La Relation Vision-Commande: Théorie et Applications à des Tâches Robotiques", Thèse de Doctorat, Université de Rennes I, Juillet 1990

[4] J.T, Feddema & C.S.G. Lee, "Adaptive Image Feature Prediction and Control for Visual Tracking with a Hand-Eye coordinated Camera", IEEE Transactions on Systems, Man and Cybernetics, vol. 20, No 5, pp 1172-1183, 1990

[5] W.T. Miller, "Sensor-Based Control of Robotic Manipulators using a General Learning Algorithm", IEEE Journal of Robotics and Automation, vol. 3, No 2, April 1987

[6] W.T. Miller, "Real-Time Application of Neural Networks for Sensor-Based Control of Robots with Vision", IEEE Transactions on Systems, Man and Cybernetics, vol. 19, No 4, July 1989

[7] N. Papanikolopoulos, B Nelson, & P. Khosla, "Full 3D Tracking using the Controlled Active Vision Paradigm", Proceedings of IEE Symposium on Intelligent Control (ISIC'92), pp 267-274, 1992

[8] N. Papanikolopoulos & P. Khosla, "Visual Tracking of a Moving Target by a Camera mounted on a Robot: a Combination of Control and Vision", IEEE Transactions on Robotics and Automation, vol. 9, No 1, Feb 1993

[9] N. Rondel & G. Burel, "Multi-Layer Perceptrons for Task Visual Servoing in Robotics", IEEE International Conference on Fuzzy Logic and Neural Networks, Funchal, Portugal, 3-6 Oct, 1995

[10] N. Rondel & G. Burel, "Vision Guided Servoing using Neural Networks", IEEE International Conference on Computational Engineering in Systems Applications (CESA96), Invited Paper, Lille, France, July 9-12, 1996

[11] O. Rodrigues, "Des loi géométriques qui régissent les déplacements d'un système solide dans l'espace, et de la variation des coordonnées provenant de ces déplacements considérés indépendamment des causes qui peuvent les produire", Journal de Mathématiques Pures et Appliquées, 1st series, (5), pp 380-440, 1840

Conclusion

Les travaux relatés dans ce mémoire utilisent les techniques neuronales pour la résolution de deux problèmes différents (mais complémentaires dans le cadre de certaines applications):

- le traitement d'antenne, où nous proposons une méthode originale basée sur la mise en œuvre de contraintes sur les poids d'un perceptron multicouche. Nous avons pu par ailleurs démontrer que cette méthode optimise un critère équivalent à celui du Maximum de Vraisemblance, tout en étant beaucoup plus rapide, puisque procédant par descente de gradient. En outre, sous certaines hypothèses, cette approche peut être complétée par la minimisation d'un critère de dépendance sur les sources: cela autorise alors la résolution du cas où le nombre de sources est égal au nombre de capteurs, cas non couvert par les autres méthodes.

- la commande référencée capteur, où nous montrons qu'il est possible d'entraîner un perceptron multicouche à estimer les mouvements à envoyer à un organe de commande (d'un robot, par exemple) pour lui faire suivre un sujet localisé à l'aide de capteurs (caméra, micros, etc), même lorsque les mouvements du sujet sont non uniformes.

Quatre applications du premier problème sont traitées: l'estimation de l'angle d'inclinaison des lignes d'un texte scanné ou écrit de travers, la séparation de signaux dans le cadre des résaux monofréquence TV, la séparation de voix humaines convoluées et mélangées, dans ce qu'on appelle le problème de la "cocktail-party", et enfin la localisation d'une ou deux personnes qui parlent, grâce à une antenne composée de deux microphones.

Le second problème est appliqué à la commande d'un robot 6 axes portant une caméra à l'extrémité de son dernier axe. Ce robot est chargé de suivre un objet en mouvement pas forcément uniforme, de telle façon que l'image fournie par la caméra reste égale à une certaine image de référence correspondant à la position relative désirée entre l'objet et la caméra.

Des extensions aux deux problèmes étudiés sont possibles:

- pour l'estimation d'angles d'arrivée, après avoir traité le cas où le nombre sources est égal au nombre de capteurs, il serait intéressant d'étendre la méthode au cas

où le nombre de sources est supérieur au nombre de capteurs. Le problème est toutefois très difficile car on ne peut pas, à partir d'un vecteur d'observation, générer un vecteur "sources estimées" de dimensionalité supérieure. Il faudrait donc trouver un moyen de contourner l'estimation des sources, par exemple en la remplaçant par une estimation de moments.

- pour la séparation de signaux émanant de réseaux monofréquence TV, une thèse est en cours pour traiter le cas, encore non élucidé, où les signaux sont mélangés et convolués en même temps, ce qui semble être un modèle de mélange plus réaliste.

- pour la commande référencée vision, des recherches sont en cours pour remplacer certaines entrées du perceptron par des paramètres moins nombreux mais contenant de l'information plus pertinente (moins bruitée), de façon à rendre l'estimateur neuronal plus robuste.

Liste des Publications

References

[1] G. Burel & N. Rondel, "Neural Networks for Array Processing: from DOA Estimation to Blind Separation of Sources", IEEE Conference on Systems, Man and Cyberbetics, Invited Paper, Le Touquet, France, 17-20 Oct, 1993

[2] N. Rondel & G. Burel, "Cooperation of Multi-Layer Perceptrons for Angles of Arrival Estimation", IEE 4th International Conference on Artificial Neural Networks, Cambridge, GB, 26-28 Juin, 1995

[3] N. Rondel & G. Burel, "Cooperation of Multi-Layer Perceptrons for the Estimation of Skew Angle in Text Document Images", IEEE 3rd International Conference on Document Analysis and Recognition, pp 1141-1144, Montréal, Canada, 14-16 August, 1995

[4] N. Rondel & G. Burel, "Multi-Layer Perceptrons for Task Visual Servoing in Robotics", IEEE International Conference on Fuzzy Logic and Neural Networks, Funchal, Portugal, 3-6 Oct, 1995

[5] N. Rondel & G. Burel, "Vision Guided Servoing using Neural Networks", IEEE International Conference on Computational Engineering in Systems Applications (CESA96), Invited Paper, Lille, France, July 9-12, 1996

SOMMAIRE

Chapitre 4: Commande référencée capteur 275