Partie II
Modèles adaptatifs du contrôle moteur: schémas de contrôle et architectures adaptatives

Chapitre 4
Des schémas de contrôle

4.1 Introduction

4.1.1 Le contrôle adaptatif d'un système de dynamique inconnue

De nombreuses études théoriques proposent des hypothèses concernant le fonctionnement des structures neuronales responsables de l'apprentissage du geste moteur [9, Atkeson, 1989] . Ces travaux abordent en fait le même problème, celui de l'apprentissage par un système adaptatif, du contrôle d'un objet dont la dynamique est a priori inconnue. Cette problématique est également largement abordée par les roboticiens travaillant dans le domaine du contrôle adaptatif des robots manipulateurs [230, Slotine et Li, 1991] . Notons tout d'abord que la plupart des systèmes automatiques que nous utilisons dans la vie courante ne sont pas adaptatifs. Bien souvent, la commande des actuateurs (moteurs à courant continu des véhicules téléguidés, des portes ou des volets automatisés par exemple) est réalisée par un servocontrôleur, contrôleur en boucle fermée, dont les paramètres sont des constantes. Cependant, les paramètres de ces contrôleurs ont été précisément choisis par les concepteurs des systèmes automatisés et ils sont donc intrinsèquement porteurs d'une certaine «connaissance» du système contrôlé. Dans les systèmes comportant un **contrôle adaptatif**, les paramètres évoluent dans le temps de façon a améliorer autant que possible les performances et la précision du contrôle. Lorsque cette évolution des paramètres est autonome et régie par une règle de correction dépendant d'une mesure de ces performances (règle d'apprentissage), cette adaptation se fait par **apprentissage**. La connaissance acquise par apprentissage concernant les propriétés dynamique du système inconnu est «stockée» dans une structure formelle (rassemblant l'ensemble des paramètres) jouant le rôle d'un approximateur de la dynamique cherchée. Cette structure peut être simplement l'ensemble des matrices nécessaires à l'écriture, sous forme d'une équation différentielle (description de Lagrange), d'un modèle simplifié du système. Mais elle peut emprunter différentes formes ou «architectures», par exemple:

- un filtre de Kalman [80, Gerdes et Happee, 1994] [94, Gusev et Semenov, 1992] [263, Wolpert et al., 1995] ,
- un réseau de neurone à couches utilisant la rétropropagation du gradient [87, Gomi et Kawato, 1993] [112, Jordan et Rumelhart, 1992] [178, Nerrand et al., 1993] ,
- un réseau de fonctions à base radiale («Radial basis functions») [213, Sanner and Slotine, 1992] [207, Ronco et Gawthrop, 1998] ,
- un contrôleur utilisant les règles de la logique floue [28, Bouslama et Ichikawa, 1993] [128, Kosko, 1992] ,
- un contrôleur utilisant des tables de mémoires [11, Atkeson et Reinkensmeyer, 1990] .

Les approches décrites dans ce chapitre, dérivées en faite des théories de l'automatique, tentent de résoudre les mêmes problèmes:

- Puisque la dynamique du système à contrôler est inconnue, il est a priori impossible d'obtenir l'erreur en commande correspondant à une erreur de trajectoire,
- Ces modèles étant adaptatifs, la dynamique du contrôle se combine avec la dynamique de l'apprentissage. La stabilité de l'ensemble du système (système inconnu + contrôleur adaptatif) est d'autant plus difficile à démontrer.
- l'application de ces modèles à la physiologie implique de montrer la stabilité de l'algorithme de contrôle pour un système hautement non linéaire.

La structure formelle rassemblant les paramètres du contrôleur sera désignée sous le terme «**filtre adaptatif**» qui ne préjuge en rien de l'architecture de cette structure. Un filtre adaptatif (au sens classique) peut utiliser de deux façons différentes les connaissances qu'il acquière concernant la dynamique du système lors de l'apprentissage (voir figure 2). Il peut d'abord effectuer une prédiction sur l'évolution de la sortie de ce système à partir d'une copie de la commande envoyée à celui-ci. Dans ce cas, son rôle est de fournir une estimation de la **dynamique directe** du système (figure 2-A). Mais il peut également donner une estimation de la commande à appliquer pour obtenir l'évolution désirée de la sortie. Il joue alors le rôle d'un estimateur de la **dynamique inverse** du système (figure 2-B). Dans les deux cas et notamment lorsque le système à contrôler est non-linéaire, l'estimation donnée par le filtre peut dépendre non seulement de la commande (dynamique directe) ou de la trajectoire désirée (dynamique inverse) mais aussi des informations concernant les variables d'état du système ou une prédiction sur celles-ci. Ainsi, les différents types de filtres adaptatifs sont d'une manière générale des **approximateurs** de fonctions.

Dans ce chapitre, nous ne discuterons pas l'efficacité ou la validité physiologique de tel ou tel type d'approximateur, mais nous comparerons **les schémas de contrôle proposés** en les étudiant dans le même cadre

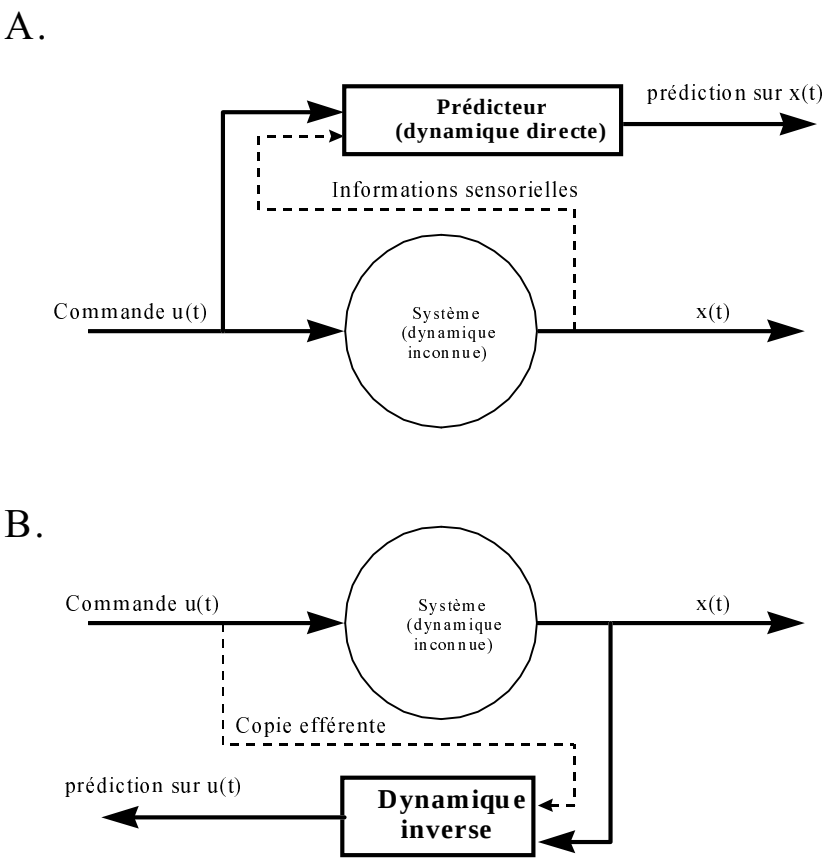

Figure 2. Deux formes de modèles internes de la dynamique d'un système inconnu. A) Dynamique directe: le modèle doit prédire au mieux l'évolution du système en fonction des commandes qui lui sont envoyées. B) Dynamique inverse: le modèle devine la commande envoyée au système d'après son évolution.

théorique avec le même vocabulaire formel. Les modèles retenus dans cette étude sont les modèles adaptatifs les plus connus, pouvant être considérés comme génériques. Il n'est en effet pas nécessaire de produire une description exhaustive des travaux publiés dans ce domaine car s'il proposent une grande variété de modèles de filtres adaptatifs, ils sont souvent intégrés dans l'un des schémas de contrôle étudiés ici. L'ordre retenu permet d'aborder progressivement les différences existant entre ces approches. Nous concluerons cette étude par une discussion de la validité physiologique et neuro-physiologique de ce type d'approche.

4.1.2 Définition d'un schéma de contrôle adaptatif

Soit un système inconnu dont la dynamique peut être décrite par l'équation suivante :

$$\frac{dx}{dt} = f(u, x)$$

où u est la commande envoyée au système, et x le vecteur d'état composé des variables d'état du système $x = \left[x^{(0)}, x^{(1)}, x^{(2)}, ..., x^{(n)}\right]$. Sachant que l'on veut faire suivre à la sortie $x(t)$ une trajectoire désirée $x_d(t)$ dans l'espace des états du système, contrôler ce système c'est trouver une fonction h de l'état instantané du système $[x(t), x'(t)]$ [21] et de la trajectoire désirée $x_d(t)$ donnant la commande $\widetilde{u}(t)$ telle que la trajectoire produite soit «*la plus proche possible*» de la trajectoire désirée. Ce «*plus proche possible*» sous entend la formulation d'un **critère** permettant d'évaluer la qualité du contrôle c'est à dire le type d'erreur que l'on souhaite minimiser dans de la tâche assignée au contrôleur. D'une manière générale, h peut s'écrire

$$\widetilde{u}(t) = h\left(..., \int x(t)dt, x(t), \frac{dx}{dt}(t), ..., \int x_d(t)dt, x_d(t), \frac{dx_d}{dt}(t), \boldsymbol{\varpi}\right) \tag{4.6}$$

ou ω est un vecteur représentant l'ensemble des paramètres de la fonction h. Ces paramètres peuvent être constants (commande non adaptative) ou évoluer au cours du temps (commande adaptative) de façon à optimiser le critère choisi. On peut décrire cette fonction comme un ensemble d'opérations unitaires (addition, multiplications, retards...) réalisées sur des flux de données (les variables citées dans l'équation (4.6)). Les opérations élémentaires peuvent être regroupées en «blocs» remplissant une fonction précise (par exemple un bloc servocontrôleur, un bloc filtre adaptatif, un bloc comparateur etc...). Un schéma de contrôle décrit l'organisation des différents blocs et les flux de données qui entrent et sortent de chaque bloc.

Un schéma de contrôle adaptatif sera ainsi défini par

1. le critère minimisé lors de la tâche,
2. une règle d'apprentissage indiquant la façon de calculer les modifications des paramètres stockant la

[21] $x'(t)$ désigne ici la dérivée temporelle du vecteur des variables d'état du système.

connaissance acquise par apprentissage de la dynamique du système en fonction des erreurs (fonction obtenue à partir du critère précédent),

3. la façon dont cette connaissance est utilisée pour le calcul de la commande (sans faire d'hypothèse sur l'architecture adaptative employée).

En fait, un schéma de contrôle adaptatif est ici une description générale de l'algorithme de commande ne précisant pas le type de filtre adaptatif employé. Nous faisons également l'hypothèse certainement abusive que ce filtre est capable d'approximer n'importe quelle fonction linéaire ou non de ses entrées.

4.1.3 Notations employées

Par souci de simplification, nous ne développerons les calculs que pour un système monodimensionnel à une seule entrée et une seule sortie (système SISO). L'entrée du système est notée u, sa sortie $x(t)$ et la trajectoire désirée $x_d(t)$. Une dérivée par rapport au temps de la variable α sera notée par α', α'' pour la dérivée première et seconde ou bien par $\alpha^{(n)}$ pour une dérivée d'ordre supérieure. Certaines parties des démonstrations font appel au calcul des variations (à un niveau élémentaire) et une petite variation d'une variable α sera notée $\delta\alpha$. Quant à la dérivée partielle de α par rapport à une autre variable β elle sera notée $\frac{\partial\alpha}{\partial\beta}$. La valeur estimée d'une variable ou d'un paramètre α sera notée $\widehat{\alpha}$ et sa valeur optimale $\widetilde{\alpha}$. Un filtre adaptatif (jouant le rôle d'approximateur d'une fonction) sera représenté par une fonction $\Psi(w, ...)$ dépendant d'un ensemble (vecteur) des paramètres $w = [w_1, w_2, ...w_n]$ et d'autres variables, par exemple uniquement la trajectoire désirée dans le cas d'un approximateur de la dynamique inverse. La forme transposée d'un vecteur V ou d'une matrice M sera noté V^T ou M^T. Pour simplifier les développement théoriques, le système étudié sera parfois un système limité à l'ordre 2, dont la dynamique sera décrite par les équations différentielles suivantes :

$$\begin{cases} \frac{dx}{dt} = x' \\ \frac{dx'}{dt} = f(x, x', u) \end{cases} \tag{4.7}$$

Lorsque l'équation décrivant la dynamique du système est inversible, on notera la dynamique inverse $u = h(x, x', x^{\cdot})$.Les démonstrations des auteurs, bien que remaniées et simplifiées autant que possible, sont hélas nécessaires à la discussion finale. Les développements théoriques qui suivent concernent souvent des systèmes monodimensionnels mais les conclusions obtenues peuvent la plupart du temps être étendues au cas multidimensionnel.

4.1.4 Les modèles étudiés

Les modèles retenus ici sont des modèles génériques, souvent cités dans la littérature, dont nombre d'autres travaux se sont inspirés. Le premier n'est utilisé ici que pour introduire la démarche théorique utilisée.

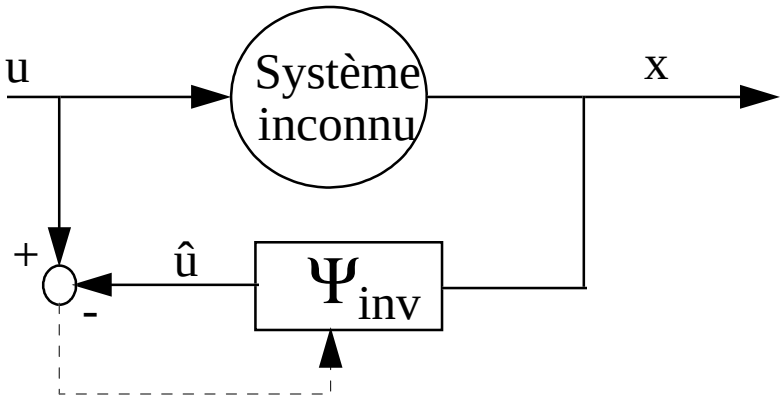

Figure 3. Estimation directe de la dynamique inverse

4.2 Estimation directe de la dynamique inverse

4.2.1 Principe

Un raisonnement simple permet de proposer une méthode d'identification de la dynamique inverse d'un système inconnu: si la commande $u(t)$ est fournie au système qui génère en sortie la trajectoire $[x(t), x'(t), x''(t)]$, alors le filtre adaptatif optimal recevant comme consigne $[x_d = x, x'_d = x', x''_d = x'']$ doit proposer comme réponse $\widehat{u}(t) = u(t)$. Dans le cas contraire, l'erreur en sortie du filtre est naturellement $\widehat{u}(t) - u(t)$. Autrement dit, en générant une commande aléatoire $u(t)$, il doit être possible d'identifier une fonction Ψ telle que $\Psi(x(u), x'(u), x''(u))$ soit l'identité. Cette méthode constitue historiquement le premier schéma de contrôle proposé pour expliquer les capacités d'auto-adaptation des systèmes sensori-moteurs [4, Albus, 1975] [11, Atkeson et Reinkensmeyer, 1990] [129, Kuperstein et Rubinstein, 1989] [199, Psaltis et Sideris, 1987] [154, Martinets et al. 1990] [169, Miller et al., 1990] .

4.2.2 Stabilité de l'apprentissage

On suppose qu'il existe un ensemble de paramètres $\widetilde{w}$ pour lesquels le filtre adaptatif est capable de reproduire exactement la dynamique inverse du système. L'erreur motrice produite par un filtre non parfaitement adapté peut s'écrire :

$$\Delta u = \widehat{u}(t) - u(t) = \Psi(w, x, x', x'') - h(x, x', x'') \tag{4.8}$$

L'optimisation du filtre peut se faire en utilisant la loi de modification des paramètres suivante:

$$\frac{dw}{dt} = -\alpha.\Delta u.\frac{\partial \Psi}{\partial w} \tag{4.9}$$

où α est une constante positive permettant d'agir sur la vitesse d'apprentissage.

Pour étudier la stabilité de l'apprentissage, introduisons une fonction $V(t)$ définie comme:

$$V(t) = \frac{1}{2}\Delta u^2 \tag{4.10}$$

Pour une trajectoire réalisée quelconque $x(t)$, une petite variation δw des valeurs des paramètres w autour de $\widetilde{w}$ et une petite variation δt du temps entraîne une variation de la fonction $V(t)$ qui peut s'écrire :

$$dV = \left(\frac{\partial V}{\partial w}\right)^T .dw + \frac{\partial V}{\partial t}\Big)_{w=cste} .dt$$

et donc,

$$\frac{dV}{dt} = \left(\frac{\partial V}{\partial w}\right)^T .\frac{dw}{dt} + \frac{\partial V}{\partial t}\Big)_{w=cste} \tag{4.11}$$

Sous le terme $\frac{\partial V}{\partial t})_{w=cste}$ sont regroupées les variations de V qui ne sont pas induites par l'évolution temporelle du vecteur des poids w.Or, d'après la définition (4.10) et d'après (4.8),

$$\frac{\partial V}{\partial w} = \Delta u^T.\frac{\partial \Psi}{\partial w}$$

En choisissant la règle d'apprentissage (4.9) alors,

$$\left(\frac{\partial V}{\partial w}\right)^T .\frac{dw}{dt} = -\alpha.\Delta u^2.\frac{\partial \Psi}{\partial w}^T .\frac{\partial \Psi}{\partial w} \leq 0$$

La dérivée temporelle de $V(t)$ (autour de la solution optimale) est donc composée de deux termes, dont l'un est toujours de signe négatif. Le second terme n'a aucune raison de possèder un signe constant. Ce terme perturbe donc la procédure d'apprentissage et ce d'autant plus que le modèle interne de la dynamique inverse est éloigné de la dynamique du système contrôlé. En effet, si $\widetilde{\varpi}$ est la valeur optimale du vecteur des paramètres de Ψ, l'erreur motrice peut s'exprimer comme suit:

$$\Delta u = \Psi(w, x, x', x'') - h(x, x', x'') = \Psi(w, x, x', x'') - \Psi(\widetilde{w}, x, x', x'')$$

$$= \Delta\Psi(x, x', x'')$$

et donc,

$$\frac{dV}{dt}\Big)_{w=cste} = \Delta\Psi.\left[\frac{\partial\Delta\Psi}{\partial x}.x' + \frac{\partial\Delta\Psi}{\partial x'}.x'' + \frac{\partial\Delta\Psi}{\partial x''}.\frac{d(x'')}{dt}\right]$$

Le signe de cette dernière expression dépend donc des dérivées temporelles de la trajectoire jusqu'à l'ordre 3. Cependant, si la vitesse d'évolution des paramètres du filtre adaptatif est faible devant celle de la variable $x(t)$ et de ses dérivées, on peut espérer que ce terme soit en moyenne nul sur un intervalle de temps Δt (de l'ordre de la constante de temps d'évolution des paramètres). Dans ce cas,

$$\int_{\Delta t} \frac{dV}{dt}.dt = \overline{V'(t)} = -\alpha\left(\Delta u^T.\frac{\partial\Psi}{\partial w}\right)^T.\Delta u^T.\frac{\partial\Psi}{\partial w} \leq 0$$

La fonction $\overline{V(t)}$ obeit aux critères définissant une fonction de Lyapunov. En effet, elle est définie positive, possède des dérivées partielles continues, et sa dérivée temporelle est négative ou nulle. Par conséquent, d'après le théorême de stabilité globale de Lyapunov, la solution optimale constitue un point d'équilibre asymptotiquement stable. En d'autres termes, la dynamique inverse h est un attracteur stable pour le filtre adaptatif: si l'état initial de Ψ est proche de cet attracteur, alors le filtre convergera vers un état donnant la dynamique inverse du système $(\widehat{u} = h(x, x', x''))$.

4.2.3 Discussion

Ce modèle possède trois caractéristiques principales:

1. L'algorithme d'apprentissage proposé conduit à une minimisation d'une erreur Δu exprimée dans l'espace de la commande et non pas dans l'espace de la tâche. Par conséquent, si la dynamique du système inconnu est localement non linéaire, une petite erreur faite dans le calcul de la commande peut conduire à une erreur de trajectoire très importante. C'est pourquoi ce type d'apprentissage n'est pas «orienté tâche» («goal directed»).
2. La phase d'apprentissage doit être séparée du suivi de trajectoire. Il n'y a aucun moyen d'apprendre la dynamique et de contrôler le système simultanément.
3. Le système de contrôle ne réagira pas aux perturbations puisqu'il fonctionne entièrement en boucle ouverte. Si le modèle inverse n'est pas optimal, le système de contrôle va accumuler des imprécisions et la trajectoire produite peut diverger très vite de la trajectoire désirée.

Notons également la contrainte existant entre rapidité et stabilité de l'apprentissage. En effet, si la constante de temps $(1/\alpha)$ est trop faible, alors le second terme de l'équation (4.11) perturbe l'apprentissage et empêche le système de converger, ou bien le fait converger vers un modèle dynamique inverse erroné.

4.3 Le Modèle de Jordan et Rumelhart (the distal teacher)

4.3.1 Principe

La méthode d'inversion directe possède deux principaux inconvénients:

- elle n'autorise pas un contrôle et un apprentissage simultanés car durant la phase d'apprentissage la commande fournie au système est aléatoire.
- l'algorithme d'apprentissage minimise une erreur exprimée dans l'espace des commandes et non dans l'espace de la tâche. Or, si le modèle inverse obtenu n'est pas parfait, une petite erreur dans l'espace des commandes peut conduire à l'obtention de larges erreurs de trajectoires.

Le modèle de Jordan et Rumelhart [112, Jordan et Rumelhart, 1992] permet d'obtenir une représentation interne de la dynamique du système en optimisant une erreur exprimée dans l'espace de la tâche. Pour cela, leur schéma de contrôle utilise deux filtres adaptatifs Ψ_D et Ψ_I destinés à apprendre respectivement une représentation de la dynamique directe (filtre prédictif) et de la dynamique inverse du système contrôlé. L'apprentissage du contrôle se fait en deux étapes. La première est similaire à celle employée dans l'estimation de la dynamique inverse: la génération de commandes aléatoires produit des trajectoires (dans l'espace de la tâche) que le modèle direct Ψ_D apprend à prédire. Dans un second temps, les propriétés dynamiques du système ainsi intégrées dans le modèle direct sont utilisées pour l'apprentissage de la dynamique inverse Ψ_I.

4.3.2 Règle d'apprentissage et stabilité du contrôle

Appelons $\widehat{x}(t)$ la prédiction faite par le modèle direct, $x = F(u)$ une fonctionnelle représentant la dynamique du système inconnu, et $F^{-1}(x) = H(x)$ son inverse représentant la dynamique inverse. On supposera qu'un apprentissage supervisé préliminaire a permis d'apprendre la dynamique directe du système et que $\Psi_D(w_D, u) \simeq F(u)$. Le critère retenu par les auteurs est le suivant :

$$V(t) = \frac{1}{2}(x - x_d)^T(x - x_d) = \frac{1}{2}e^T.e$$

Si seul le modèle inverse est soumis à l'apprentissage, la dérivée temporelle de $V(t)$ peut s'écrire:

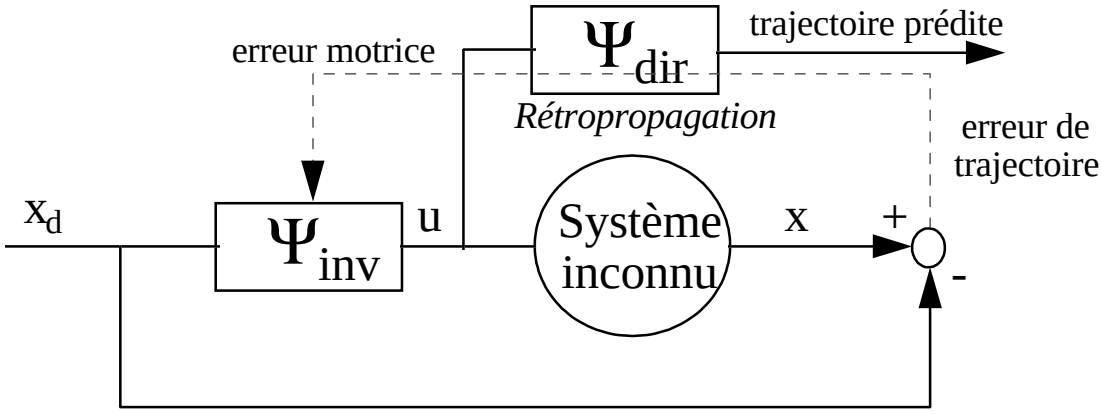

Figure 4. Modèle adaptatif de Jordan et Rumelhart («supervised learning with a distal teacher», voir texte pour les explications)

$$\frac{dV}{dt} = \left(\frac{\partial V}{\partial w_I}\right)^T . \frac{dw_I}{dt} + \frac{\partial V}{\partial t}\Bigg)_{w=cste} \tag{4.12}$$

et donc

$$\left(\frac{\partial V}{\partial w_I}\right)^T . \frac{dw_I}{dt} = e. \left(\frac{\partial x}{\partial w_I}\right)^T . \frac{dw_I}{dt}$$

Or

$$\frac{\partial x}{\partial w_I} = \left(\frac{\partial x}{\partial u}\right)^T . \frac{\partial u}{\partial w_I}$$

Mais le premier facteur $\frac{\partial x}{\partial u}$ est a priori inconnu. Les auteurs proposent de substituer à ce terme l'estimation $\frac{\partial \widehat{x}}{\partial u}$ qui peut en être faite à partir du modèle direct Ψ_D. Ainsi,

$$\left(\frac{\partial V}{\partial w_I}\right)^T . \frac{dw_I}{dt} = e. \left(\frac{\partial \widehat{x}}{\partial u} . \frac{\partial u}{\partial w_I}\right)^T . \frac{dw_I}{dt} = e. \frac{\partial \Psi_D}{\partial u} . \left(\frac{\partial u}{\partial w_I}\right)^T . \frac{dw_I}{dt}$$

En choisissant comme règle d'apprentissage $\frac{dw_I}{dt} = -\alpha . e . \frac{\partial \Psi_D}{\partial u}^T . \left(\frac{\partial u}{\partial w_I}\right)$ on obtient le terme de valeur toujous négative ou nulle

$$\left(\frac{\partial V}{\partial w_I}\right)^T . \frac{dw_I}{dt} = -\alpha . e^2 . \frac{\partial \Psi_D}{\partial u}^2 . \left(\frac{\partial u}{\partial w_I}\right)^T . \frac{\partial u}{\partial w_I}$$

Si l'on suppose, comme dans le modèle précédent, que le second terme de l'équation (4.12) est en moyenne nul sur une fenêtre de temps de l'ordre de la constante de temps d'évolution des paramètres w_I, alors $\overline{V'(t)} \leq 0$ et $V(t) \geq 0$ et l'on peut espérer que le critère $V(t)$ tende vers 0 au fur et à mesure de l'apprentissage.

4.3.3 Discussion

Le modèle de Jordan et Rumelhart est bien orienté-tâche mais il conserve deux inconvénients majeurs:

- le modèle direct n'intervient pas dans le contrôle et ne permet donc pas de réagir aux perturbations éventuelles.
- l'apprentissage du modèle direct doit être séparé dans le temps de celui du modèle inverse.

De plus, d'un point de vue plus pratique, l'apprentissage ne permet pas d'obtenir un filtre adaptatif direct Ψ_D modélisant parfaitement le système considéré. Il est donc légitime de se demander si les imperfections de Ψ_D ne vont pas perturber de façon dramatique l'apprentissage du modèle inverse Ψ_I et la stabilité du contrôle

résultant.

4.4 Le Modèle de Kawato (feedback-error learning)

4.4.1 Principe

Kawato propose d'utiliser comme estimation de l'erreur faite sur la commande, la sortie d'un contrôleur de type P.I.D (combinaison linéaire de l'erreur de trajectoire et de ses dérivées ou intégrales successives). La commande $u(t)$ est séparée en deux composantes que sont respectivement la sortie du contrôleur (u_{fb}) et celle d'un filtre adaptatif ($u_{ff} = \Psi(w, x_d, x_d^{(1)}, x_d^{(2)}, ..., x_d^{(n)})$. Cette seconde composante étant une commande en boucle ouverte puisqu'elle ne dépend que de la trajectoire désirée et non de l'erreur. La sortie du contrôleur u_{fb} est considérée comme une estimation de l'erreur motrice et la loi d'apprentissage supervisée proposée est la suivante :

$$\Delta w_i = \alpha.(\frac{d\Psi}{dw_i}).u_{fb} \tag{4.13}$$

où Δw_i représente le changement de valeur de l'un des paramètres du modèle Ψ après une itération de calcul. Nous reprendrons la démonstration proposée par l'auteur dans [118, Kawato, 1990] en la simplifiant (système unidimensionnel de dynamique inversible). Cette démonstration fait appel au calcul des variations effectué sur un intervalle de temps fini. L'introduction d'une fonctionnelle $F(u) = x - x_d = \delta x = e$ est nécessaire pour étudier les conséquences d'une petite variation «autour de la commande idéale» $u_d(t)$ sur l'erreur de poursuite $e(t)$. F est définie comme une fonctionnelle de l'espace $C[0,T]$ de la commande motrice u vers l'espace bidimensionnel $C^2[0,T]$ de l'erreur e. Une variation δu de la commande autour de la commande optimale entraîne l'apparition d'une erreur de trajectoire $e(t)$ qui peut être calculée comme suit:

$$F'(u).\delta u = \delta x = e$$

où $\delta u = u - \widetilde{u}$ décrit l'erreur motrice, différence entre la commande appliquée et la commande optimale. Pour obtenir une relation entre erreur motrice δu et erreur dans l'espace de la tâche $e(t)$, il suffit (si cela est possible) d'inverser la relation précédente:

$$\delta u = F'^{-1}.e$$

La même relation peut être appliquée à la description de la dynamique du système en termes de mécanique Lagrangienne telle qu'elle est faite dans l'équation (4.7):

$$\begin{cases} \frac{d(\delta x)}{dt} = \delta x' \\ \frac{d(\delta x')}{dt} = \delta x'' = \frac{\partial f}{\partial x}.\delta x + \frac{\partial f}{\partial x'}.\delta x' + \frac{\partial f}{\partial u}.\delta u \end{cases}$$

Par conséquent, la relation entre erreur motrice et erreur dans l'espace de la tâche sera donnée par :

$$\delta u = \frac{\delta x'' - \frac{\partial f}{\partial x}.\delta x - \frac{\partial f}{\partial x'}.\delta x'}{\frac{\partial f}{\partial u}} \tag{4.14}$$

Utiliser cette relation pour calculer la commande optimale est une application d'un algorithme d'optimisation numérique bien connu : la méthode de Newton (Raphson-Newton, méthode de descente du gradient à l'ordre 2). Malheureusement, il est impossible d'appliquer la relation (4.14) car les termes dépendant de la dynamique sont a priori inconnus. Kawato propose d'utiliser une approximation de l'opérateur F'^{-1} par un contrôleur P.D.A., une combinaison linéaire de l'erreur et de ses dérivées, soit :

$$\delta u = K_A.e'' - K_P.e - K_D.e' \tag{4.15}$$

avec

$$u = u_{ff} - u_{fb} = u_{ff} - \left(K_A.e'' - K_P.e - K_D.e'\right)$$

Les gains K_P, K_D et K_A étant calculés de façon à ce que l'équation différentielle (4.15) soit stable (une condition nécessaire est $[K_P \leq 0, K_D \leq 0, K_A \geq 0]$). L'équation (4.15) décrit en fait la dynamique d'un système mécanique classique : un système composé d'une masse, d'un amortisseur, et d'un ressort. Ce système mécanique «virtuel» est substitué au système inconnu pour le calcul de la relation entre erreur motrice et erreur dans l'espace de la tâche. L'équation (4.15) fournit en fait une approximation de la dynamique inverse du système supposant que cette dynamique est proche de celle d'un système masse/ressort/amortisseur.

Ce schéma a été présenté comme une hypothèse sur le fonctionnement des circuits moteurs et en particulier de la boucle cortico-cerebelleuse [120, Kawato et Gomi, 1992] et de l'adaptation du réflexe opto-cinétique et du réflexe vestibulo-oculaire [121, Kawato et Gomi, 1993] .

4.4.2 Stabilité du contrôle

Cette démonstration est restreinte aux cas où la fonction f décrivant la dynamique du système est inversible (d'inverse $h(x, x', x'') = u$) et appliquée à un système monodimensionnel. On suppose en outre que le filtre adaptatif Ψ est capable de fournir la solution exacte de la dynamique inverse du système inconnu pour un

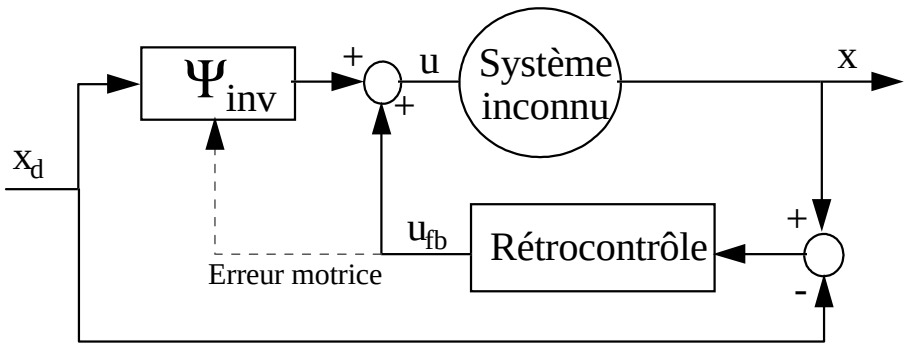

Figure 5. Modèle adaptatif de Kawato (”feedback error learning”)

ensemble de paramètres optimaux $\widetilde{w}$.

4.4.2.1 Résumé de la démonstration proposée par l'auteur

L'auteur propose d'étudier la stabilité du contrôle en choisissant le critère suivant :

$$V = \frac{1}{2} u_{im}^2$$

avec $u_{im} = u_{ff} - \widetilde{u}$ où $\widetilde{u}$ désigne la commande produite par un modèle inverse de la dynamique optimal ($\widetilde{u} = h(x_d, x'_d, x''_d) = \Psi(\widetilde{w}, x_d, x'_d, x''_d)$). Démontrer la convergence vers 0 de la fonction $V(t)$ permettrait de montrer que la procédure d'apprentissage aboutit à l'obtention d'un modèle dynamique inverse optimal. On peut écrire,

$$u_{im} = -u_{fb} + h(x, x', x'') - h(x_d, x'_d, x''_d) = \Delta h - u_{fb}$$

mais aussi

$$u_{im} = \Psi(w, x, x, x'') - \Psi(\widetilde{w}, x_d, x'_d, x''_d)$$

La fonction $V(t)$ ne dépend que de variables «externes» (w et x_d, x'_d, x''_d) du système de contrôle et sa dérivée temporelle est

$$\frac{dV}{dt} = \frac{\partial V}{\partial w}^T . \frac{dw}{dt} + \left(\frac{\partial V}{\partial x_d} . \frac{dx_d}{dt} + \frac{\partial V}{\partial x'_d} . \frac{dx'_d}{dt} + \frac{\partial V}{\partial x''_d} . \frac{dx''_d}{dt} \right) \tag{4.16}$$

Cette expression est composée de deux termes respectivement l'apprentissage et à la cinématique de la consigne. Le premier terme de cette expression peut s'écrire :

$$\frac{\partial V}{\partial w}^T . \frac{dw}{dt} = u_{im} . \frac{\partial u_{im}}{\partial w}^T . \frac{dw}{dt}$$

En appliquant la règle d'apprentissage proposée par l'auteur, on obtient

$$\frac{\partial V}{\partial w}^T . \frac{dw}{dt} = \alpha . u_{im} . \frac{\partial \Psi}{\partial w}^T . \frac{\partial \Psi}{\partial w} . u_{fb}$$

Les termes u_{im} et u_{fb} n'ont a priori aucune raison d'avoir des signes différents. Cependant, l'auteur suppose, pour pouver la stabilité de l'algorithme, que u_{im} est proportionnel à $-u_{fb}$ et propose les deux justifications suivantes :

1) Si les gains du contrôleur P.D.A. sont suffisamment grands, alors $u_{im} = \Delta h - u_{fb} \approx -u_{fb}$ et l'expression ci-dessus est de signe négatif.

2) on peut également développer le terme Δh de la façon suivante :

$$\Delta h = h(x, x', x'') - h(x_d, x'_d, x''_d) = \frac{\partial h}{\partial \zeta}.\delta\zeta + \frac{\partial h}{\partial \zeta'}.\delta\zeta' + \frac{\partial h}{\partial \zeta''}.\delta\zeta''$$

avec $\delta\zeta = x - x_d = e$. Or si les coefficients du contrôleur P.I.A. sont choisis de façon à ce qu'ils constituent une bonne approximation de l'opérateur (4.14) alors $\Delta h \approx -u_{fb}$ et par conséquent $u_{im} \approx -2.u_{fb}$.

Dans les deux cas on peut alors écrire

$$\frac{\partial V}{\partial w}^T.\frac{dw}{dt} \approx -\alpha.\lambda.u_{im}^2.\frac{\partial \Psi}{\partial w}^T.\frac{\partial \Psi}{\partial w} \tag{4.17}$$

avec $0 \leq \lambda \leq 1$.

La dérivée de la fonction $V(t)$ peut donc se décomposer en deux termes (cf expression 4.16) dont l'un est de signe toujours négatif, le signe du second terme restant indéterminé. Cependant, si l'on suppose que la trajectoire désirée possède des «bonnes propriétés» (x_d processus stationnaire) et si l'on raisonne sur une fenêtre temporelle glissante Δt suffisamment grande devant les variations de la trajectoire désirée telle que

$$\int_{\Delta t} \left(\frac{\partial V}{\partial x_d}.x'_d + \frac{\partial V}{\partial x'_d}.x''_d + \frac{\partial V}{\partial x''_d}.x_d^{(3)} \right) dt = 0$$

alors

$$\int_{\Delta t} V'(t)dt = \overline{V'(t)} = -\alpha.\lambda. \int_{\Delta t} \left(u_{im}^2.\frac{\partial \Psi}{\partial w}^T.\frac{\partial \Psi}{\partial w} \right) dt \leq 0$$

Il faut noter que la dimension de la fenêtre temporelle est liée à la constante de temps de l'apprentissage (approximativement égale à $1/\alpha$). En d'autre termes, pour que l'algorithme converge, il est nécessaire que l'apprentissage soit très lent devant les variations temporelles de x_d et de ses dérivées.

La fonction $\overline{V(t)}$ possède des dérivées partielles continues, est définie positive et a pour borne inférieure 0 et une dérivée négative ou nulle. On peut par conséquent dire que le point $\overline{u_{im}} = 0$ est un point d'équilibre globalement asymptotiquement stable.

4.4.2.2 Conséquences de la convergence asymptotique de V(t)

Nous avons montré que u_{im} tendait asymptotiquement vers 0. Mais ceci n'implique pas forcément la convergence asymptotique de l'erreur de poursuite $e(t)$ vers 0. On peut écrire

$$u_{im} = \Delta h - u_{fb} = -K_P.e(t) - K_D.e'(t) + K_A.e''(t) + \Delta h(t) \tag{4.18}$$

Ceci amène l'auteur à supposer également que la solution de l'équation différentielle (4.18) pour $u_{im}(t) = 0$ tend asymptotiquement vers 0 lorsque t tend vers l'infini. Cette proposition n'est vrai que si les gains du contrôleur P.D.A. sont convenablement choisis, et si le terme «perturbant» Δh (reflétant les imprécisions du modèle dynamique inverse) est d'amplitude négligeable devant les autres termes.

4.4.3 Discussion

Le développement ci-dessus nous permet de faire les remarques suivantes:

- Comme le modèle de Jordan et Rumelhart, le modèle de Kawato est orienté-tâche («goal-directed»).
- Il présente de plus l'avantage de permettre l'apprentissage du modèle inverse lors de l'accomplissement de la tâche (tâche de poursuite par exemple).

Cependant,

- la démonstration de la stabilité de l'algorithme suppose des hypothèses fortes sur les propriétés de la trajectoire désirée.
- le contrôleur en boucle fermée doit approximer de manière correct le jacobien inverse F'^{-1} ce qui suppose une connaissance, même grossière et établie a priori, des caractéristiques dynamiques du système contrôlé.
- au cours de l'apprentissage, la convergence est perturbée par les imprécisions issues du modèle dynamique en cours d'adaptation. Cette difficulté ne peut être contournée qu'en choisissant une vitesse d'apprentissage délibérément faible (comme pour les modèles précédents).

Ce schéma de contrôle, qui en pratique fonctionne bien, à été ensuite généralisé, donnant d'autre rôles au filtre adaptatif Ψ. Dans le modèle présenté ici, celui-ci admet en entrée des informations concernant uniquement la trajectoire désirée du système. La commande $u_{ff} = \Psi(w, x_d, x'_d, x''_d)$ est donc générée en boucle ouverte. Mais le filtre adaptatif peut également fonctionner en boucle fermée ($u_{ff} = \Psi(w, x, x', x'')$) ou dans un mode de contrôle mixte («Non Linear Regulator Learning», $u_{ff} = \Psi(w, x_d, x'_d, x''_d, x, x')$). Les démonstrations concernant ces modèles, très semblables à celle proposée ici, peuvent être trouvées dans Gomi et Kawato [87,

Gomi et Kawato, 1993] .

4.5 Robustesse du contrôle

4.5.1 Influence des imprécisions du modèle interne sur la stabilité du contrôle adaptatif

Dans les paragraphe précédents, nous avons montré que la convergence de l'apprentissage et l'efficacité du contrôle étaient garantis à condition que l'effet des imprécisions sur la connaissance de la dynamique du système contrôlé ne soit «pas trop perturbant». Plus précisément, la dérivée de la fonction de Lyapunov $V(t)$ associée à chaque schéma de contrôle a été systématiquement séparée en deux termes

$$\frac{dV}{dt} = \left[\frac{\partial V}{\partial w}^T . \frac{dw}{dt}\right] + \frac{dV}{dt}\Big)_{w=cste} \tag{4.19}$$

Or, si l'on a vu que l'on pouvait déterminer le signe du premier terme de cette équation en choisissant une loi d'apprentissage adaptée, le signe du second terme ne peut pas être maîtrisé. La convergence de l'apprentissage n'est alors garantie que si ce second terme de l'équation (4.19) possède les «bonnes propriétés» décrites précédemment. Cependant, cette contrainte, même si elle est vérifiée, nécessite le choix d'une faible vitesse d'apprentissage. Par conséquent, les modèles proposés ici ne présentent pas de réelle garantie de robustesse vis-à-vis des imprécisions de modélisation de la dynamique inconnue et imposent un apprentissage lent.

Il existe cependant différentes approches théoriques se proposant d'aborder spécifiquement ce problème de robustesse du contrôle des systèmes dynamiques et nous avons choisi de développer ici celle utilisant des **variables composites** (Slotine et Li, 1991). En effet, cette approche propose des hypothèses très intéressantes pour les physiologistes et pourraient en particulier permettre d'expliquer des traits caractéristiques majeures du contrôle des mouvement chez les vertébrés. Nous développerons cette approche en deux étapes. Dans un premier temps, nous rappelerons que l'utilisation de variables composites permet de prendre en compte et de maîtriser les incertitudes concernant le contrôle non adaptatif d'un système dont les propriétés dynamiques ne sont pas parfaitement connues [230, Slotine et Li, 1991] . Nous verrons ensuite comment cette approche peut être utilisée dans le cadre du contrôle adaptatif pour augmenter à la fois sa robustesse et la rapidité de l'apprentissage.

4.5.2 Les variables composites

La définition d'une variable composite est très simple puisqu'il s'agit d'une combinaison linéaire de variables mesurables et de leurs dérivées. Dans le développement suivant, nous utiliserons en fait seulement des vari-

ables composites combinaison de l'erreur exprimée dans l'espace de la tâche et de ses dérivées. D'autres combinaisons présentent un intérêt hors du cadre de cette discussion comme par exemple une combinaison linéaire de la force et de la vitesse lorsque l'on veut contrôler un système en présence d'un retard bidirectionnel important dans la transmission des signaux des capteurs et des signaux de commande (en téléopération par exemple). Si l'on définit ici la variable composite $s(t)$ comme étant

$$s(t) = \sum_{i=0}^{n-1} \lambda_i . e^{(i)} \tag{4.20}$$

alors cette relation est également une équation différentielle d'ordre $(n-1)$. Le contrôle par surface glissante («sliding control») propose d'utiliser comme critère minimisé lors de la tâche une fonction du carré de cette variable, usuellement

$$V(t) = \frac{1}{2} s(t)^2$$

Cette fonction est définie positive et l'on peut, comme dans les démonstration précédente, essayer de proposer une loi de commande qui détermine le signe de sa dérivée temporelle de façon à assurer la convergence asymptotique de $V(t)$. Si $V(t)$ converge vers 0, il est possible de choisir les coefficients λ_i de la variable composite de façon à ce que l'équation (4.20) soit stable et implique donc une convergence asymptotique de l'erreur $e(t)$ vers 0 (on suppose de plus que $\lambda_{n-1} = 1$ pour simplifier les calculs). Appliquons ce principe à un système d'ordre quelconque dont la dynamique est définie par l'équation différentielle

$$x^{(n)} = f(x, x^{(1)}, x^{(2)}, ..., x^{(n-1)}) + u \tag{4.21}$$

La fonction f peut être non-linéaire mais le système doit être à entrée séparable (la commande agit de façon additive). On suppose également que l'on dispose d'une estimation imparfaite de la dynamique du système $\widehat{f}$. La dérivée temporelle de $s(t)$ s'écrit:

$$\frac{dV}{dt} = s . \frac{ds}{dt}$$

Or, d'après (4.20),

$$\frac{ds}{dt} = \sum_{i=1}^{n} \lambda_{i-1} . e^{(i)}$$

et par conséquent

$$\frac{ds}{dt} = x^{(n)} - x_d^{(n)} + \sum_{i=1}^{n-1} \lambda_{i-1}.e^{(i)}$$

En utilisant la relation (4.21)

$$\frac{ds}{dt} = u + f - x_d^{(n)} + \sum_{i=1}^{n-1} \lambda_{i-1}.e^{(i)}$$

Si l'on partage la commande u en deux composantes distinctes $\widehat{u}$ et u_s telles que $u = \widehat{u} + u_s$ avec $\widehat{u} = x_d^{(n)} - \sum_{i=1}^{n-1} \lambda_{i-1}.e^{(i)} - \widehat{f}$, alors l'équation précédente devient

$$\frac{ds}{dt} = u_s + (f - \widehat{f}) = u_s + \Delta f \tag{4.22}$$

cette équation, qui n'est autre que le second terme de la relation (4.19), peut être considérée comme une équation du premier ordre où le terme Δf agit comme une perturbation sur la commande u_s. Ainsi, l'introduction de la variable composite $s(t)$, a permis de transformer un problème de contrôle d'un système d'ordre n par le contrôle d'un système de premier ordre (problème beaucoup plus simple). Remarquons cependant que le problème de la robustesse du contrôle n'est toujours pas résolu en raison de l'existence d'une perturbation $\Delta f = (\widehat{f} - f)$ issue d'une imprécision sur la connaissance des propriétés dynamiques du système. En pratique il est cependant possible de borner cette imprécision, c'est à dire de trouver une fonction du temps $\eta(t)$ (ou même constante) telle que

$$|\Delta f(t)| \leq \eta(t)$$

Puisque le nouveau système (4.22) est du premier ordre, il est possible de choisir une stratégie de contrôle en tout ou rien du type «si l'erreur $s(t)$ est négative, alors il suffit de la corriger en produisant une commande h positive d'amplitude assez grande (et vice-versa)». On peut donc proposer la commande

$$u_s = -k.signe(s) \tag{4.23}$$

Dans ce cas,

$$\frac{dV}{dt} = s.\left[-k.signe(s) + \Delta f\right] = -k.\left|s\right| + \Delta f.s$$

En choisissant une fonction k d'amplitude «suffisamment grande» comme par exemple $k = \eta + \gamma$ ou γ est une constante positive, alors

$$\frac{dV}{dt} \leq -\gamma . |s|$$

Ainsi, appliquer la commande totale $u = x_d^{(n)} - \sum_{i=1}^{n-1} \lambda_{i-1}.e^{(i)} - \widehat{f} - k.signe(s)$ au système permet à la dérivée de la fonction $V(t)$ d'être toujours de signe négatif, ce qui assure la convergence asymptotique de $V(t)$ vers 0, et par conséquent, la convergence asymptotique de $s(t)$ vers 0. Remarquons ici que la commande produite est composée de deux termes de natures différentes. Le premier ($\widehat{u}$) est assimilable à un contrôleur fonctionnant de manière continue alors que le second ($u_s = -k.signe(s)$) introduit dans la commande une composante non-linéaire importante (et notamment des hautes fréquences).

Dans le chapitre précédent (modèle de Kawato), le contrôle est également basé sur la minimisation d'un critère fonction lui aussi d'une variable composite (du second ordre) constituée de la sortie du contrôleur P.D.A. soit

$$u_{fb} = s(t) = K_A.e''(t) - K_P.e(t) - K_D.e'(t)$$

Le critère minimisé est le même que celui choisit ici soit $V(t) = \frac{1}{2}s(t)^2$. Cependant, alors que la méthode décrite dans ce chapitre suppose que l'on connaisse une approximation de la dynamique du système contrôlé, cette estimation est obtenue par apprentissage dans le modèle utilisant un «feedback error learning». Autrement dit, dans le modèle de Kawato, seule le signe du premier terme de l'equation (4.19) est contrôlé alors que c'est le signe du second terme que le contrôleur décrit ci dessus maîtrise. Il est donc naturel de supposer que les deux approches peuvent être associées de façon à produire un schéma de contrôle adaptatif robuste permettant un apprentissage plus rapide.

4.6 Un modèle synthétique

4.6.1 Un contrôle adaptatif robuste

Le premier intérêt de l'utilisation d'une variable composite est de ramener le contrôle d'un système d'ordre quelconque à celui d'un système d'ordre 1 (cf. equation 4.22). On peut donc essayer d'imaginer un système de contrôle permettant cette réduction d'ordre mais qui soit également adaptatif. Prenons l'exemple d'un système dynamique inversible du second ordre, dont la dynamique inverse est définie par la relation

$$u = h(x, x', x'')$$

En notations de Laplace, si ce système est linéaire, on peut lui attribuer une fonction de transfert F telle que $X = F(U)$ où U est la transformée de Laplace de la commande $u(t)$ et X celle de la trajectoire $x(t)$. Essayons de ramener le problème du contrôle de ce système au contrôle d'un système de premier ordre de dynamique

$$u = \alpha.x + \beta.x'$$

en introduisant une boucle de rétroaction (linéaire) $r(x, x', x'')$ dans la commande u. Cette commande est donc la somme de deux termes

$$u = u_s + r(x, x', x'')$$

et la fonction de transfert en boucle fermée du système s'écrit

$$X = \frac{F}{1 - R.F}.U_s$$

Or, si p est l'opérateur de Laplace,

$$\frac{F}{1 - R.F} = \frac{1}{\alpha + \beta.p}$$

si et seulement si

$$R = F^{-1} - (\alpha + \beta.p) = H - (\alpha + \beta.p)$$

Ce calcul appliqué au cas particulier d'un système linéaire montre que l'on peut simplifier un problème de contrôle en introduisant une boucle de retour qui utilise une estimation de la dynamique inverse du système et une variable $y(t) = \alpha.x + \beta.x'$ combinaison de la sortie du système et de sa dérivée. Notons bien que si nous avons choisi ici une variable composite du premier ordre, il est possible d'utiliser une variable composite d'ordre quelconque ou même une fonction non-linéaire de x, x',..., $x^{(n)}$. L'intérêt de cette méthode est de forcer le système global (système dynamique + rétrocontrôle) à adopter une dynamique particulière.

Essayons de développer une extension de ce schéma au contrôle adaptatif de systèmes non linéaires en remplacant la boucle de rétroaction linéaire $R(x, x', x'')$ par un filtre adaptatif $\Psi(x, x', x'')$. L'équation décrivant la dynamique en boucle ouverte peut s'écrire

$$u_s + \Psi(w, x, x', x'') - \left(\alpha.x + \beta.x'\right) = h(x, x', x'')$$

ou encore

$$u_s = y(t) + (h - \Psi)$$

Le second terme de la relation précédente agit donc comme une perturbation sur le contrôle. Le rôle de la procédure d'apprentissage est de minimiser cette perturbation et donc de trouver les paramètres optimaux du filtre $\widetilde{w}$ tels que $\Psi(\widetilde{w}, x, x'', x'') = h(x, x', x'')$. Pour trouver une règle d'apprentissage adaptée, il est nécessaire de proposer une fonction potentielle (fonction de Lyapunov) dont les propriétés permettent d'assurer la stabilité et la convergence de l'apprentissage. Proposons la fonction

$$V(t) = \frac{1}{2}(w - \widetilde{w})^T(w - \widetilde{w})$$

La dérivée temporelle de cette fonction est

$$V'(t) = (w - \widetilde{w})^T.\frac{dw}{dt} = \Delta w^T.\frac{dw}{dt}$$

Le terme Δw est inconnu, cependant on peut remarquer que

$$u_s - y(t) = h - \Psi = \Psi(\widetilde{w}, x, x', x'') - \Psi(w, x, x', x'')$$

or

$$\Psi(\widetilde{w}, x, x', x'') - \Psi(w, x, x', x'') = -\frac{\partial \Psi}{\partial w}^T(w, x, x', x'').(w - \widetilde{w}) = -\frac{\partial \Psi}{\partial w}^T.\Delta w \tag{4.24}$$

Si l'on choisit comme règle d'apprentissage

$$\frac{dw}{dt} = (u_s - y)\frac{\partial \Psi}{\partial w} \tag{4.25}$$

alors

$$V'(t) = (u_s - y).\Delta w^T.\frac{\partial \Psi}{\partial w} = -(u_s - y)^2$$

La fonction $V(t)$ possède une dérivée de valeur négative ou nulle ce qui garantie la stabilité globale asymptotique du point $w = \widetilde{w}$. Notons cependant que si le filtre adaptatif est très éloigné du filtre optimal, la relation (4.24) n'est plus vraie. En particulier, au début de l'apprentissage, le système est placé uniquement

sous le contrôle de la variable composite $y(t)$ qui doit donc être capable d'assurer un contrôle stable du système durant une période suffisante pour que le filtre adaptatif se rapproche du modèle inverse du système. Notons bien que la règle d'apprentissage (4.25) est très proche de celle proposée par Kawato[22].

4.6.2 Contrôle adaptatif par surfaces glissantes

Pour appliquer ce schéma de contrôle à la poursuite, il suffit d'introduire une entrée supplémentaire dans le système en posant

$$s(t) = e'(t) + \lambda.e(t) = (x' - x'_d) + \lambda.(x - x_d)$$

en utilisant comme boucle de rétrocontrôle $y(t) = s'(t)$ (la dérivée temporelle de $s(t)$ et non $s(t)$) avec donc

$$u = u_s + \Psi - s'(t)$$

L'équation de la dynamique en boucle ouverte du système devient alors

$$u_s = s'(t) + (h - \Psi)$$

Cette équation est en tous points semblable à la relation (4.22) proposée dans le cadre du contrôle par surface glissante non adaptatif. En choisissant comme fonction potentielle

$$V(t) = \frac{1}{2}\left[s^2 + (w - \widetilde{w})^T(w - \widetilde{w})\right]$$

et comme règle d'apprentissage

$$\frac{dw}{dt} = (u_s - s').\frac{\partial \Psi}{\partial w}$$

alors on peut utiliser exactement la même méthode («switching control») que dans le chapitre précédent si l'on peut borner l'incertitude $(h - \Psi)$ concernant la dynamique inverse du γ contrôlé. Dans ce cas, il suffit de définir la commande u_s comme suit

$$u_s = -k(t).signe(s)$$

[22] Ce développement, légèrement modifié, peut constituer une démonstration (plus concise que celle proposée par l'auteur) de la pertinence du «feedback error learning».

pour s'assurer que $V'(t) \leq 0$. Ainsi, le choix de la fonction potentielle ci-dessus permet de prouver à la fois la convergence du modèle interne Ψ vers le modèle dynamique inverse de l'objet et la convergence de la trajectoire du système vers la trajectoire désirée. Il faut noter que, grâce à l'apprentissage, l'incertitude $(h - \Psi)$ tend vers 0 et que donc l'amplitude des mouvements de correction ($k(t)$) peut être choisie comme une fonction décroissante du temps.

4.6.3 Contrôle par rétrocontrôle adaptatif de l'erreur robuste

En nous inspirant du contrôle par surface glissante adaptatif [230, Slotine et Li, 1991] , essayons de la même manière d'intégrer dans le schéma de contrôle général proposé précédemment le modèle de Kawato. Proposons une fonction potentielle $V(t)$ suivante

$$V(t) = \frac{1}{2}\left[\alpha \int_0^t e(z).dz + \beta e(t) + \gamma.e'(t)\right]^2 + \frac{1}{2}(w - \widetilde{w})^T(w - \widetilde{w})$$

où α, β et γ sont les coefficients du contrôleur P.I.A. introduit par Kawato

Dans ce cas,

$$\frac{dV}{dt} = s(t).\left[\alpha.e(t) + \beta e'(t) + \gamma.e''(t)\right] + \Delta w^T.\frac{dw}{dt} = s(t).s'(t) + \Delta w^T.\frac{dw}{dt}$$

Notons bien ici que selon les termes utilisés par l'auteur $s'(t) = u_{fb}(t)$. Selon les notations utilisées ici, on obtient donc une dynamique globale du système décrite par la relation suivante:

$$u_s = s'(t) + (h - \Psi)$$

là aussi tout à fait semblable à la relation (4.22).

En choisissant comme règle d'apprentissage :

$$\frac{dw}{dt} = (u_s - s').\frac{\partial \Psi}{\partial w}$$

Alors

$$\frac{dV}{dt} = s(t).\left[u_s + (\Psi - h)\right] - (u_s - s')^2 \frac{\partial \Psi}{\partial w}^T.\frac{\partial \Psi}{\partial w}$$

et donc, si l'on peut borner l'incertitude $(\Psi - h)$, il est possible de choisir une fonction $k(t)$ telle que si

$$u_s = -k.signe(s)$$

alors

$$\frac{dV}{dt} \leq -\gamma.\,|s| - (u_s - s')^2 \frac{\partial \Psi}{\partial w}^T . \frac{\partial \Psi}{\partial w} \leq 0$$

Cette démonstration, tout à fait similaire à la précédente permet d'assurer la convergence globale asymptotique à la fois de l'erreur de poursuite $e(t)$ et du filtre adaptatif vers la dynamique inverse du système.

4.6.4 Conclusion

Le schéma de contrôle proposé dans ce chapitre permet de montrer que l'approche proposé par Kawato et l'approche robuste du contrôle des systèmes non-linéaires par surface glissantes relèvent de la même démarche. Nous pouvons voir que finalement, les deux approches diffèrent seulement par le choix de la fonction potentielle minimisée par la commande adaptative:

- Modèle de Kawato : $s(t) = \alpha \int_0^t e(z).dz + \beta e(t) + \gamma.e'(t)$
- Modèle de contrôle par surface glissantes : $s(t) = e'(t) + \lambda.e(t)$

L'algorithme d'apprentissage force donc le système global à adopter une dynamique différente dans les deux cas. En fait, comme nous l'avons vu précédemment, n'importe quelle combinaison linéaire de l'erreur et de ses dérivées[23] peut être utilisée. Si on choisit une dynamique D_g (dynamique globale) imposée par une fonction du type

$$s(t) = D_g(... \int\int\int e.dt, \int\int e.dt, \int e.dt, e, e^{(1)}, e^{(2)}, e^{(3)} ...)$$

alors la boucle de rétrocontrôle à utiliser est

$$y(t) = \frac{ds}{dt}$$

et la dynamique en boucle ouverte du système global devient

$$\frac{ds}{dt} = u_s + (\Psi - h)$$

[23] Respectant cependant certaines conditions de stabilité (voir l'annexe dédiée au contrôle par surfaces glissantes pour les détails).

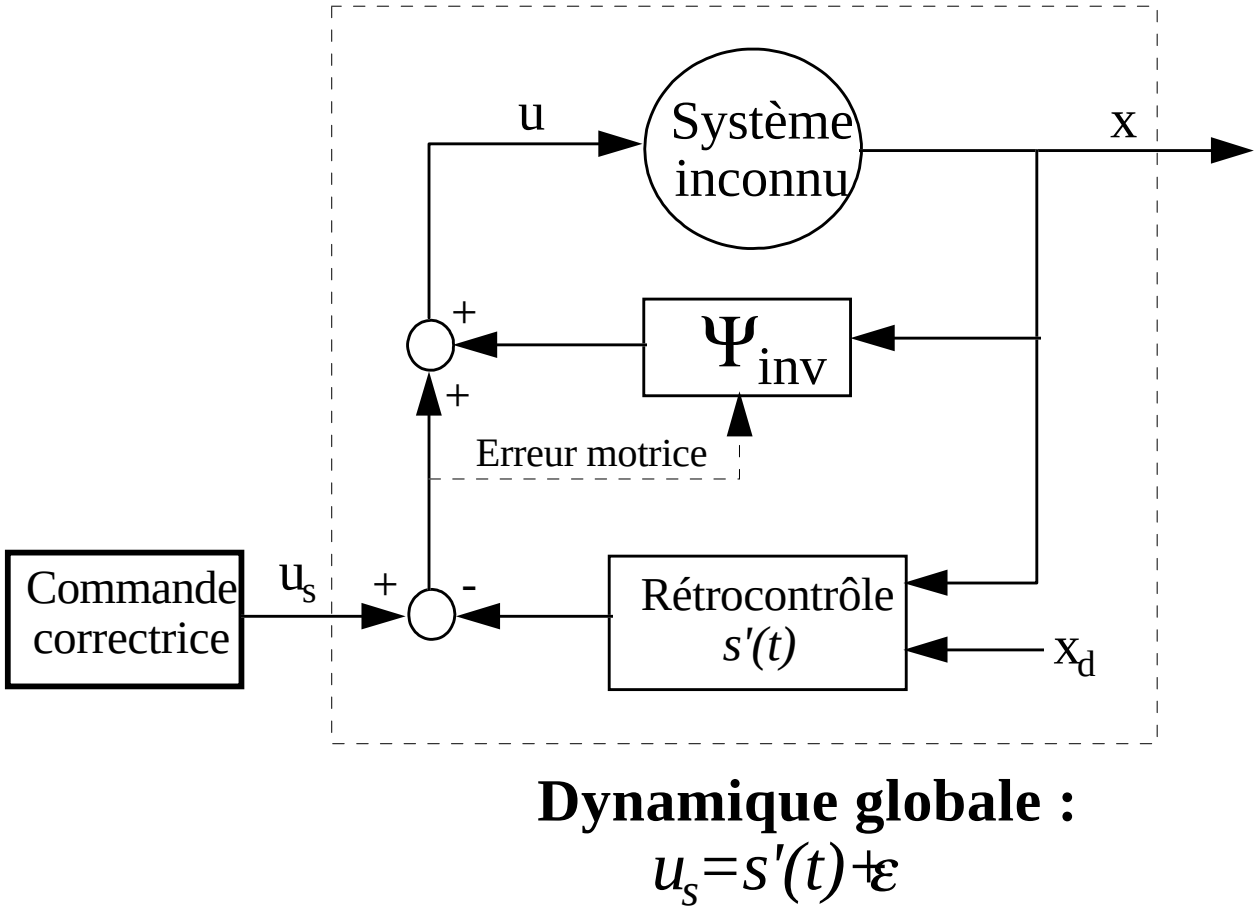

Figure 6. Modèle générique proposé pour un contrôle adaptatif robuste (IMRAC)

En appliquant la règle d'apprentissage

$$\frac{dw}{dt} = (u_s - s').\frac{\partial \Psi}{\partial w}$$

alors la convergence du filtre adaptatif vers la dynamique inverse du système est assurée. Si de plus l'incertitude $(\Psi - h)$ peut être bornée, alors la robustesse du contrôle adaptatif est garantie.

4.7 Discussion

Par rapport aux modèles d'apprentissage présentés au début de cette étude, l'approche développée ci-dessus présente l'avantage d'assurer une robustesse plus grande du contrôle adaptatif, et également la possibilité d'adapter et de contrôler le système simultanément. Le seul élément adaptatif du schéma de contrôle joue le rôle d'un estimateur de la dynamique inverse du système fonctionnant en boucle fermée. Deux éléments du contrôle jouent un rôle fondamental dans cette stratégie: la variable composite et l'ajout d'une commande en

boucle ouverte u_s assurant la stabilité du contrôle.

4.7.1 Rôle de la variable composite

Le choix de la variable composite $s(t)$ détermine le comportement en boucle ouverte du système global. Or, bien que cette variable composite aie une influence décisive sur le contrôle, nous n'avons donné pour l'instant aucune indication sur le choix des paramètres qui la définissent et aucune idée sur son éventuel rôle ou origine physiologique.

La variable composite peut d'abord être considérée comme un rétrocontrôleur dont le rôle est important au début de l'apprentissage lorsque l'estimation de la dynamique inverse donnée par le filtre adaptatif est incorrecte. La nécessité d'assurer ce rôle est en soi une première contrainte qui limite le choix des paramètres de s. Cette contrainte implique que $s(t)$ doit être «suffisamment proche» de la dynamique inverse du système[24]. Cependant, si, après apprentissage, la dynamique du filtre adaptatif est proche de la dynamique inverse du système, il est possible de modifier les paramètres de $s(t)$ sans pour autant rendre nécessaire une autre phase d'apprentissage. Cette caractéristique est très intéressante du point de vue du roboticien, car il est ainsi possible de contrôler de façon indépendante l'impédance du système mécanique [87, Gomi et Kawato, 1993] . Cette impédance mécanique peut être définie

- dans l'espace des articulations si $s(t)$ est définie comme une combinaison linéaire des erreurs angulaires et de leurs dérivées.
- dans l'espace de la tâche si $s(t)$ est définie comme une combinaison linéaire de l'erreur et de ses dérivées exprimée en coordonnées cartésiennes.

4.7.2 Rôle de la commande en boucle ouverte

Pour que la démonstration du chapitre précédent soit valable, il est nécessaire que le décourt temporel de $u_s(t)$ soit indépendant de celui des variables d'état du système contrôlé ou bien de l'erreur de trajectoire (commande en boucle ouverte). Indépendamment de la notion de robustesse du contrôle, cette commande peut être en fait une contrainte imposée par le milieu extérieur, comme par exemple un couple ou une force d'interaction issues d'un contact avec un objet extérieur [87, Gomi et Kawato, 1993] ou bien encore de l'effet de la gravité sur les mouvements des segments. Cependant, nous avons vu qu'il était possible de se servir de cette commande supplémentaire pour améliorer la robustesse du contrôle. Puisque l'introduction de la variable composite et du filtre

[24] Dans le cadre du contrôle par surface glissante adaptatif, il est possible d'utiliser une variable composite dont la dynamique est très différente de la dynamique inverse du système. Mais c'est au prix d'une forte activité de la commande correctrice u_s, et d'une vitesse d'apprentissage très importante. Ceci peut avoir comme effet d'empêcher la convergence des paramètres w du filtre. Ce n'est absolument par gênant du point de vue du contrôle (qui reste stable). Le filtre adaptatif Ψ fluctue alors tout au long du suivi de trajectoire.

adaptatif nous a permis de ramener le problème au contrôle d'un système du premier ordre, plusieurs stratégies peuvent être adoptées. Nous en avons déjà présenté plus particulièrement une («switching law», 4.23) qui présente cependant un inconvénient majeur: elle demande une très haute activité de commande pouvant impliquer une dépense d'énergie importante et exciter des fréquences de résonance élevées dans certains systèmes mécaniques. Un autre mode de fonctionnement intéressant pourrait être l'emploi d'un stratégie impliquant **un contrôle intermittent** de la valeur de la variable composite. Supposons que la valeur absolue de l'incertitude sur la dynamique puisse être bornée par une constante K positive. Alors

$$u_s - K < s'(t) < u_s + K$$

Si aucune commande supplémentaire n'est appliquée ($u_s = 0$), l'incertitude sur la valeur de la variable composite $s(t)$ augmente linéairement avec le temps. Si $s(t = 0) = 0$,

$$0 - K.t < \int_0^t s'(\tau).d\tau < 0 + K.t$$

Si le système de contrôle ne peut produire une commande quelconque de manière instantanée (comme cela est requis pour appliquer un «switching control») et qu'il lui faut un temps Δt donné pour le faire, il est cependant possible d'appliquer une stratégie consistant à maintenir la valeur absolue de $s(t)$ en dessous d'un certain seuil s_s. En l'occurence, si au temps $t = t_1$, $s(t_1) > s_s$, alors il suffit de produire une commande u_s telle que

$$\int_{t=t_1}^{t=t_1+\Delta t} u_s(\tau).d\tau = s(t_1)$$

pour que la valeur de $s(t + \Delta t)$ soit au dessous du seuil. Evidemment, la valeur du seuil est liée à l'incertitude sur la dynamique. Il est nécessaire, pour que cette stratégie fonctionne, que

$$|s_s| >> K.\Delta t$$

Si l'incertitude sur la dynamique est importante, cette contrainte implique soit une durée Δt très brève (ce qui correspond au «switching control», qui est a priori la stratégie optimale), soit un seuil très élevé et donc une faible qualité du suivi de trajectoire.

4.7.3 Théorie du contrôle adaptatif des systèmes non linéaires et physiologie du

mouvement

Les schémas de contrôle étudiés dans ce chapitre ont été présentés soit dans le strict cadre de la robotique des manipulateurs, soit dans une perspective physiologique comme support à l'explication des propriétés adaptatives du système moteur. Ainsi, le modèle de Kawato a été utilisé pour modéliser les propriétés d'adaptation des réflexes vestibulo-oculaire et optocinétique [86, Gomi et KAwato, 1992] , [120, Kawato et Gomi, 1992] [121, Kawato et Gomi, 1993] . Pour les auteurs, le cervelet joue un rôle crucial dans l'apprentissage du contrôle des mouvement et l'adaptation des réflexes. C'est un filtre adaptatif (Ψ) qui apprend grâce au mécanisme de la LTD («long term depression») modulant l'efficacité synaptique des connections entre fibres parallèles et cellules de Purkinje. Bien que cette proposition soit soutenue par un grand nombre de données anatomiques et electrophysiologiques [228, Shidara et al., 1993] , il reste cependant difficile d'attribuer ainsi une fonctionnalité particulière aux différentes aires impliquées dans le contrôle des mouvements pour deux raisons principales:

- Les voies afférentes et efférentes à ces différentes aires sont parfois mal connues.
- Les signaux produits par les neurones de ces structures, même s'ils peuvent être corrélés à l'évolution de variables physiquement mesurables (position ou vitesse de l'oeil, erreur visuelle, force appliquée) peuvent très bien être exprimés dans des référentiels internes dont la détermination nécessiterait l'enregistrement simultané de l'activité d'un grand nombre de cellules.

Sur le plan comportemental, certaines des hypothèses proposées peuvent cependant être testées expérimentalement. En effet, le modèle synthétique proposé dans le dernier chapitre possède des caractéristiques particulières et identifiables. Il implique (dans sa forme *contrôle intermittent*) la superposition de deux commandes aux caractéristiques différentes,

- une commande «continue», fortement corrélée à une combinaison de l'erreur et de ses dérivées exprimées dans un référentiel lié à la tâche.
- une commande en boucle ouverte, intermittente et de nature impulsionnelle (génératrice de hautes fréquences) responsable du maintien de la stabilité du contrôle en dépit des incertitudes concernant la dynamique des segments corporels ou des objets manipulés.

Or, on trouve une telle combinaison de mouvements rapides et intermittents et de mouvement plus lents et continus dans plusieurs comportements moteurs comme par exemple les réflexes vestibulo-oculaires et optocinétiques, les mouvements des yeux (combinaison de saccades et poursuites) et dans le contrôle postural. Même si l'identification du modèle proposé, qui comporte des propriétés non-linéaires importantes, reste problématique, il reste possible de vérifier la plausibilité de quelques unes de ses propriétés principales:

- la combinaison d'une commande continue et d'une commande intermittente de type «impulsionnelle»,
- l'existence d'une variable composite minimisée lors de la tâche,
- le déclenchement de mouvements correcteurs lorsque cette variable composite dépasse, en valeur absolue, un seuil relativement fixe,
- une diminution de l'amplitude et de la fréquence de ces corrections au fur et à mesure de l'apprentissage, la dynamique du système controlé (segments corporels ou objet manipulé) étant plus précisément connue.

La troisième partie de cette thèse, à caractère spécifiquement expérimental, sera justement consacrée à la validation d'un modèle exploitant ces propositions, dans le cadre d'une tâche impliquant le contrôle visuo-manuel d'un système dynamique aux propriétés inconnues des sujets.

Il reste cependant que nous avons volontairement «laissé dans l'ombre» un composant essentiel de ces schémas de contrôle. Les «boîte noires» Ψ sont des éléments essentiels car elles sont à la source même des propriétés adaptatives du contrôle. C'est pourquoi les deux derniers chapitres de cette seconde partie sont destinés à la présentation de deux modèles d'architecture de calcul distribuées susceptibles de jouer ce rôle. Le développement de ces deux modèles a été guidé par un souci de rigueur et d'efficacité pratique, mais également par la préoccupation de rechercher des solutions porteuses d'une forte pertinence biologique.

Chapitre 5
Le codage par population de fonctions

5.1 Le neurone formel

5.1.1 Le neurone support du traitement de l'information dans le système nerveux

Ce sont les progrès de l'histologie, puis la naissance de l'électrophysiologie qui ont donné naissance à la «doctrine neuronale». Celle-ci fait du neurone le type cellulaire reponsable de la circulation et du traitement de l'information dans le système nerveux depuis les récepteurs cellulaires jusqu'aux effecteurs. Ce postulat est né de quelques observations maintenant anciennes:

- Dans une région donnée du système nerveux, il n'existe pas de variabilité continue dans l'organisation des arbres dendritiques, dans la taille et l'aspect des soma. La population neuronale d'une aire peut être classée selon quelques types neuronaux qui diffèrent par l'aspect, la taille ou la disposition des neurones ou bien par la structure de leur arbre dendritique [35, Cajal, 1911] .
- L'information (la polarisation électrique de la membrane) circule dans un neurone depuis la surface post-synaptique vers l'axone et ses collatérales dans un seul sens. Cette «polarisation dynamique» du neurone a conduit rapidement à le considérer comme un élément associatif, sa sortie (caractérisée par la présence ou l'absence d'un potentiel d'action) dépendant de façon causale de ses entrées [35, Cajal, 1911] . Cette vision du neurone comme étant le noeud fonctionnel d'un réseau éventuellement réverbérant, réalisant une fonction d'entrée-sortie a atteint son apogée avec le modèle de McCulloch et Pitts [157, McCulloch et Pitts, 1943] .
- La réactivité du neurone à une entrée donnée peut être modifiée. La découverte des bases cellulaires de l'apprentissage [25, Bliss et Lomo, 1973] [40, Castelluci et al., 1978] [238, Spencer et al., 1966] a permis de soutenir l'hypothèse que l'adaptation ou l'apprentissage avait pour source la capacité du neurone à

apprendre une fonction particulière.

Ces observations, et elles seules, ont été à l'origine de la notion de «neurone canonique» et du développement des travaux concernant les réseaux de neurones formels. Dans cette approche, le neurone ne réalise qu'un seul type de fonction («fonction canonique»), et la connectivité du réseau, éventuellement acquise par apprentissage, en règle les paramètres. Ainsi, le modèle de neurone formel le plus classiquement utilisé réalise le seuillage d'un produit scalaire. Si l'on note $s(t)$ sa sortie et $e(t)$ ses entrées

$$s(t) = f(\sum_{j=1}^{j=k} \varpi_j . e_j(t) + \xi) \tag{5.26}$$

où $f(x) = 1/(1+e^{-x})$, ξ est le seuil de sensibilité du neurone et ϖ_j définit la connectivité du neurone avec des récepteurs sensoriels ou bien d'autres neurones.

5.1.2 La modélisation en neurophysiologie: deux approches complémentaires

Que peut apporter la modélisation à l'étude de la physiologie du mouvement ? Son rôle premier est d'aider à la validation théorique d'hypothèses émises d'après des études expérimentales c'est à dire de proposer un cadre formel qui permet d'expliquer les résultas obtenus. Citons par exemple l'analyse du codage par population de la force ou de la direction du mouvement qui a donnée lieu à de nombreuses études expérimentales où la théorie joue un grand rôle [79, Georgopoulos et al., 1993] [215, Schwartz, 1993] [247, Tanake et Nakayama, 1995] . Mais les modèles peuvent aussi être directement source d'hypothèses soumises ensuite au crible de l'expérimentation. C'est le cas par exemple du «minimum jerk model» [67, Flash et Hogan, 1985] et du «minimum torque-change criterion» [253, Uno et Kawato, 1989] , proposant tous les deux des contraintes formelles sur la genèse de la cinématique des mouvements. En ce qui concerne plus particulièrement le rôle de la modélisation dans l'étude fonctionnelle des réseaux de neurones, deux approches complémentaires coexistent. La première démarche utilise l'étude des propriétés biochimiques et même moléculaires des neurones et les données expérimentales concernant leur connectivité pour simuler le fonctionnement d'un réseau formel plus facilement manipulable que son alter ego biologique. Une seconde approche consiste à construire un réseau à partir d'un ou de plusieurs modèles de neurone minimal possédant une fonctionnalité mathématique locale (du type de celle décrite dans l'equation 5.26). Les simulations permettent de valider cette architecture neuronale. Elle doit être organisée de façon à respecter des critères de pertinence biologique et être capable de reproduire une fonction du système nerveux central. Si cette architecture fonctionne, elle peut alors constituer une hypothèse intéressante pouvant être testée expérimentalement en particulier si les simulations permettent de faire des prédictions sur le comportement de la population de neurones biologiques concernée.

5.1.3 Des contraintes spécifiques à l'approche dynamique

Un système dynamique est par définition un système dont l'évolution à un instant donné dépend de ses entrées mais aussi de ses états antérieurs. La reproduction de la dynamique d'un système par un modèle connexioniste doit donc également respecter ces propriétés. D'autre part, les membres polyarticulés des mammifères (les systèmes qui nous intéressent ici) ont une géométrie et une dynamique hautement non linéaires. Ce sont ces deux caractéristiques (systèmes à mémoire d'état, non linéarités) qui rendent complexe l'étude et la modélisation du contrôle des mouvements. Le problème de la redondance géométrique et dynamique des effecteurs (non traité dans ce chapitre) est également source de complexité. Du point de vue de la Robotique, le contrôle des mouvements des membres correspond au «pire» des cas: le contrôle adaptatif d'un système non linéaire géométriquement et dynamiquement redondant. Dans cette étude théorique, nous proposons deux hypothèses différentes concernant la simulation ou le contrôle, par *un filtre adaptatif* de systèmes dynamiques articulés non linéaires. Dans une première partie, nous essayerons de mettre en correspondance d'une part les observations expérimentales concernant le codage par population de neurones et les propriétés «multimodales» des neurones et d'autre part des propositions théoriques issues de la Robotique (réseau de gaussiennes, fonctions à bases radiales) ou de modèles neurophysiologiques (modèle de la mémoire dynamique). Cette première approche correspond à l'apprentissage (ou approximation) de la dynamique d'un système non linéaire par *linéarisation locale*. Notre seconde proposition, une architecture à base de champs mnémoniques, est l'expression d'une intuition théorique. Elle constitue une approche novatrice car l'architecture proposée utilise des unités (les «mnémons») possédant des propriétés de mémorisation plutôt que des propriétés calculatoires du types de celles décrites dans l'équation (5.26). Ces deux études répondent au même souci de trouver des méthodes d'approximation de fonctions dynamiques non linéaires utilisant un *algorithme d'apprentissage local*, sans rétropropagation d'erreur.

5.2 Linéarisation locale: multimodalité et codage par population

5.2.1 Multimodalité et codage par population

Le contrôle des mouvements nécessite l'apprentissage, par le système nerveux central, de transformations non linéaires complexes. Une dynamique directe ou inverse peut être reproduite grâce à des filtres adaptatifs tels que le filtre de Kalman (essentiellement utilisé pour les systèmes linéaires) ou les réseaux de neurones formels. Les réseaux de neurones sont très souvent utilisés dans les modélisation de différentes fonctions du système nerveux central (vision, mouvement, posture) car il sont supposés posséder des caractéristiques proches de leurs équivalents biologiques. Ces travaux utilisent en grande majorité l'architecture du perceptron multi-couches apprenant par retropropagation d'erreur. Si les propriétés d'approximateur universel de tels réseaux

sont reconnues (bien que discutées), ils utilisent un algorithme d'apprentissage nécessitant une propagation complexe de l'erreur entre les différentes couches. Or, cette circulation d'information, du moins dans sa version originelle, apparaît comme peut réaliste au regard des connaissances accumulées sur les propriétés individuelles des neurones et de leurs populations.

Les neurones de nombreuses structures sensorielles et motrices possèdent deux propriétés communes et majeures. Il est d'abord clair que ces neurones ne réagissent pas aux variations d'un seul type de signal sensoriel mais possèdent une sensibilité à des signaux issus de différentes modalités ou bien issus de récepteurs sensoriels qui différent par leur type ou leur localisation organique. La fréquence de décharge des neurones du colliculus supérieur, par exemple, est influencée à la fois par des stimuli d'origine visuelle, auditive ou somatosensorielle [259, Wallace et al., 1993] . L'activité neuronale dans le cortex moteur est modifiée par des signaux proprioceptifs provenant de groupes musculaires et non d'un muscle isolé [65, Fetz et Cheney, 1980] . La réponse de neurones des aires motrices a pu être corrélée avec la direction des mouvements effectués mais aussi avec la force appliquée [214, Schmidt et al., 1975] [44, Cheney et Fetz, 1980] .

D'autre part, l'activité de ces neurones dépend des variations des entrées sensorielles de la façon suivante: leur décharge est maximale pour une combinaison particulière des entrées (un «stimulus préféré»). La fréquence de décharge diminue ensuite lorsque la stimulation s'éloigne de ce stimulus préféré. La relation entre taux de décharge et la stimulation, la **courbe d'accord** du neurone («tuning curve»), est d'allure variable selon les aires et les types neuronaux, mais comporte souvent un unique maximum. Cette propriété existe dès les premiers stades de transmission des informations sensorielles, par exemple pour les afférences vestibulaires primaires [7, Angelaki, 1991] ou pour les fibres du nerf auditif [124, Kiang, 1965] . Mais on la trouve dans les aires supérieures en particulier au niveau cortical. Les cellules du cortex visuel ont une selectivité à l'orientation du stimulus ou à la direction de mouvement de ce même stimulus ([182, Orban, 1984] ; [183, Orban, 1991] ; revues). Les neurones de l'aire MT, intervenant dans la perception des vitesses dans l'espace visuel, présentent une direction de vitesse préférée [172, Movshon et al., 1985] . Un grand nombre de données expérimentales suggère également l'existence de ce type de sélectivité dans les aires corticales impliquées dans le contrôle des mouvements. C'est le cas pour le cortex moteur et le cortex premoteur, dont les neurones semblent sensibles à la direction du mouvement effectué et déchargent de façon maximale pour une direction donnée. Mais on trouve une réponse neuronale sélective à une orientation également dans le cervelet [72, Fortier et al., 1993] , le colliculus supérieur [255, Van Gisbergen et Tax, 1987] [130, Kutz et al., 1997] [237, Sparks et al., 1997] et également dans le cortex préfrontal en particulier dans le champ oculomoteur frontal (le «frontal eye field», [76, Funahashi et al., 1990]).

Ce mode de fonctionnement individuel des neurones a conduit expérimentalistes et théoriciens à considérer que le système nerveux central procédait à un codage distribué des informations issues des récepteurs

sensoriels (le «codage par population»). En effet, il est possible de reconstituer, à partir des fréquences de décharge individuelles des neurones, la stimulation correspondante par une opération très simple effectuée au niveau de la population [78, Georgopoulos et al., 1986] . Si l'on note par exemple θ la direction du mouvement, $\widehat{\theta}_i$ la direction préférée du $i^{ème}$ neurone d'une population de n cellules et $g_i(\theta)$ sa courbe d'accord, alors il est possible de retrouver la direction θ du mouvement en calculant une estimation $\tilde{\theta}$ obtenue par l'expression suivante:

$$\widetilde{\theta} = \sum_{i=1}^{i=n} g_i(\theta).\widehat{\theta}_i \tag{5.27}$$

L'estimateur (5.27) implémente un type de codage particulier communément appelé «population coding» mais également «channel coding» par la communauté travaillant sur la physiologie du système visuel [233, Snippe and Koenderink, 1992] . D'autres hypothèses ont été envisagées pour expliquer les caractéristiques en terme de résolution, d'hyperacuité ou seuil de discrimination des systèmes sensoriels. On peut citer les alternatives les plus simples comme d'abord le codage par intensité («intensity coding» ou «rate coding»), ou chaque récepteur code une composante de la stimulation dans l'espace des percepts possibles. Mais certains auteurs proposent également l'hypothèse d'un codage par étiquettage («coarse coding» ou «labelled line coding» ou encore «place coding»). Selon ce principe, les neurones codent un percept dans l'espace des percepts possibles en déchargant en «tout ou rien» lorsque le stimulus est localisé dans leur champ récepteur. Autrement dit, si r_i représente la réponse (normalisée) du $i^{ème}$ neurone de la population et R_i son champ récepteur et x le stimulus, alors

$$\begin{aligned} r_i(x) &= 1 \text{ si } x \in R_i \\ r_i(x) &= 0 \text{ si } x \notin R_i \end{aligned} \tag{5.28}$$

Les caractéristiques de ce type de codage, appliqué à des neurones possédant de larges champs récepteurs, permettent de proposer une explication des phénomènes d'hyperacuité observés pour les récepteurs sensibles aux stimuli tactiles, auditifs et visuels [63, Eurich et Schwegler, 1997] . Cependant, en particulier en ce qui concerne le contrôle des mouvements, on n'observe pas de neurones présentant le comportement en tout ou rien décrit par la propriété (5.28). En effet, un grand nombre de neurones présente une sensibilité à la position de la stimulation dans le champ récepteur. Les propriétés de l'estimateur (5.27) ont été étudiées dans le cadre des travaux concernant le cortex moteur, mais également dans un cadre théorique plus général, en considérant l'équation (5.27) comme un cas particulier d'approximateur de la fonction identité. Il est important de résumer les caractéristiques de ce codage car, outre sa forte plausibilité biologique, il présente de nombreux avantages

théoriques par rapport aux autres hypothèses citées précédemment.

5.2.2 Propriétés du codage par population

Les propriétés de l'estimateur ont été étudiées dans le cas monodimensionnel, en relation avec les résultats obtenus sur le codage de la direction des mouvements [246, Tanaka, 1994] [247, Tanaka et Nakayama, 1995] , mais aussi dans le cas multidimensionnel en ce qui concerne notamment la perception visuelle [233, Snippe and Koenderink, 1992] . Dans le cas multidimensionnel, l'équation (5.27) reste valide, mais $\widehat{\theta}_i$, $\widetilde{\theta}$ et θ sont des vecteurs. La première propriété de ce codage est sa précision. Cette précision dépend de la taille de la population de neurones, de la distribution des directions préférées et des fonction g_i (définissant en particulier la forme et la taille des champs récepteurs). La précision est le plus souvent estimée par l'erreur r.m.s. définie par la relation

$$E_{rms} = \frac{1}{m}\sqrt{\sum_{k=1}^{k=m}\left(\theta_k - \widetilde{\theta}_k\right)^2}$$

où m est le nombre d'observations. Pour plusieurs types de courbes d'accord, les simulations numériques semblent montrer que la précision dépend linéairement de l'inverse de la racine carré du nombre d'unités neuronales impliquées dans le codage [246, Tanaka, 1994] . Ce résultat a été obtenu pour différents degrés de variabilité des décharges neuronales et pour des degrés variables d'uniformité de la distribution des directions préférées. L'erreur r.m.s augmente linéairement avec la variabilité des réponses neuronales et avec le degré de non uniformité de la distribution des directions préférées. Il faut noter cependant que ces observations ont été obtenues par simulation numérique (dont la valeur démonstrative est très limitée) et qu'elles concernent une classe particulière de fonction $g_i(\theta, \widehat{\theta}_i)$. La courbe d'accord est approximée par une fonction cosinusoidale de la différence entre la direction perçue et la direction préférée ($g_i = cos(\theta - \widehat{\theta}_i)$). Par conséquent, la largeur du champ récepteur est fixe (défini de 0 à 2π) et la relation (5.27) s'apparente à un produit scalaire normalisé entre le vecteur représentant la direction du mouvement et le vecteur de direction de mouvement préférée. D'autres études décrivent les propriétés théoriques du codage par population effectué par une population de neurones dont la courbe de réponse est une gaussienne à symétrie radiale

$$g_i(\theta, \widehat{\theta}_i) = A_i.e^{-\frac{\left(\theta - \widehat{\theta}_i\right)^2}{2\sigma_i^2}} \tag{5.29}$$

La constante A_i représente le gain spécifique de la courbe d'accord et σ_i la largeur du champ récepteur du neurone correspondant. Dans le cas multidimensionnel, l'équation (5.29) devient

$$g_i(\theta, \widehat{\theta}_i) = A_i.e^{-\frac{(\theta-\widehat{\theta}_i)^T(\theta-\widehat{\theta}_i)}{2\sigma_i^2}} \tag{5.30}$$

L'influence du bruit et de la largeur du champ récepteur (lorsque celle-ci est la même pour tous les neurones) ont été étudiés par le biais de simulations numériques dans le cas (5.29) appliqué au codage de la direction de mouvement [247, Tanaka et Nakayama, 1995] . Les auteurs ont montré que E_{rms}, en l'absence de bruit croît de façon monotone avec la largeur du champ récepteur en suivant une loi de puissance en deçà d'une valeur critique, et une loi exponentielle au delà de cette même valeur. Cette dépendance de E_{rms} vis à vis de σ n'est plus monotone si la réponse des neurones est bruitée: dans ce cas, on obtient une valeur minimum de E_{rms} pour une valeur particulière de σ (appelée «largeur optimale» par les auteurs). Cette relation entre l'erreur d'estimation et la largeur du champ récepteur a été étudiée de manière théorique et plus générale et dans le cadre multidimensionnel [233, Snippe and Koenderink, 1992] . Dans cette étude s'intéressant plus particulièrement à l'analyse formelle des capacités «psychophysiques» d'un observateur virtuel utilisant l'estimateur (5.27), les auteurs ont montré qu'il était possible de prédire le seuil de discrimination de cet observateur (en présence de bruit) et que le phénomène d'hyperacuité (observation de seuils de discrimination faibles devant la largeur des champs récepteurs) était une de ses propriétés naturelles.

5.2.3 Approximation utilisant un codage par population sur un réseau de gaussiennes :

Le codage par population est considéré dans les travaux cités ci-dessus avant tout comme un estimateur robuste d'une variable physiquement signifiante. Il est cependant difficile de trouver une discussion de l'utilisation de cette information distribuée par le système nerveux central dans le but de produire ou de contrôler une action, un mouvement. C'est précisément cette question que nous abordons dans ce chapitre. En effet, une extension de ce codage par population présente des propriétés intéressantes du point de vue du contrôle des mouvements: une population de neurones présentant un stimulus préféré permet en fait un échantillonnage spatial de la variable considérée. Le principe discuté dans ce chapitre est le codage par population, non pas d'une variable physiquement signifiante, mais d'une transformation sensorimotrice. Autrement dit, il est possible d'utiliser le principe du codage par population afin d'approximer une fonction complexe (non linéaire) par une population de fonctions aux propriétés plus simples (linéaires ou semi-linéaires).

Prenons l'exemple d'une fonction $f(\theta)$ de $\Re$ dans $\Re$. On peut produire une estimation $\widetilde{f}$ de f en utilisant, par analogie avec (5.27), l'expression

$$\widetilde{f}(\theta) = \sum_{i=1}^{i=n} (a_i \cdot \theta + b_i) \cdot g_i(\theta) = \sum_{i=1}^{i=n} f_i(\theta) \cdot g_i(\theta) \tag{5.31}$$

Nous avons remplacé dans l'estimateur (5.27) la fonction constante sous-entendue $f_i(\theta) = \theta_i$ par une fonction affine définie par les deux coefficients a_i et b_i. Autrement dit, nous associons à chaque neurone accordé sur une valeur particulière de θ, une transformation affine particulière définissant **une approximation affine locale** de la fonction f (voir **Annexe A**, pour plus de détails).

Pour étudier l'influence de la courbe d'accord $g_i(\theta)$ choisie, intéressons-nous au propriétés spectrales (dans un premier temps du point de vue spatial) de cet approximateur. Ceci peut se faire facilement si l'on restreint l'ensemble des fonctions $g_i(\theta)$ possibles à une famille de fonction obtenues par décalage spatial d'une même et seule fonction radiale soit $g_i(\theta) = g_0(\theta - i \cdot \Delta\theta)$. La définition de l'estimateur (5.27) devient

$$\widetilde{f}(\theta) = \sum_{i=1}^{i=n} \theta_i \cdot g_0(\theta - i \cdot \Delta\theta) \tag{5.32}$$

Or l'expression ci-dessus est exactement équivalente à une approximation discrète d'une **convolution spatiale** de deux fonctions. La première fonction est une fonction en escalier h définie par

$$h(\theta) = \theta_i \text{ pour } \theta \in [i \cdot \Delta\theta, (i+1) \cdot \Delta\theta[$$

La seconde fonction est la fonction à base radiale $g_0(\theta)$. Le codage par population permet donc d'approximer la fonction identité en utilisant une approximation par morceaux (courbe en escalier) filtrée. Les propriétés spectrales du filtre correspondent à celles de la transformée de Fourier de la fonction $g_0(\theta)$. Ainsi, si la fonction à base radiale est une gaussienne, la convolution conduira à un filtrage passe-bas de la fonction en escalier $h(\theta)$. Ce raisonnement est également valable pour l'approximateur (5.31) mais la fonction $h(\theta)$ est alors la fonction linéaire par morceau

$$h(\theta) = a_i \cdot \theta + b_i \text{ pour } \theta \in [i \cdot \Delta\theta, (i+1) \cdot \Delta\theta[$$

Il faut par conséquent noter que le choix de la fonction de base $g_0(\theta)$ conditionne la résolution fréquentielle de l'approximateur.

5.2.4 Cas des systèmes dynamiques non linéaires: codage spatio-temporel

Ce pincipe peut s'appliquer pour approximer la dynamique d'un système non linéaire dont les paramètres dépendent de ses variables d'état. La fonction $h(\theta)$ décrit alors la dynamique locale de ce système. Prenons l'exemple d'une variable $y(t)$ dont la dynamique par rapport à une entrée $x(t)$ s'exprime sous la forme:

$$y(t) = a(x) \cdot x(t) + b(x) \cdot \dot{x}(t) + c(x) \cdot \ddot{x}(t)$$

Les paramètres de la dynamique de $y(t)$ sont ici eux-mêmes fonction de l'état de l'entrée $x(t)$ (la dynamique décrite est donc non-linéaire). On peut dans ce cas appliquer le principe de codage par population (5.31) à une population de fonctions décrivant une dynamique linéaire:

$$\widetilde{y}(x(t)) = \widetilde{f}(x(t)) = \sum_{i=1}^{i=n} \left(a_i \cdot x(t) + b_i \cdot \frac{\partial x}{\partial t}(t) + c_i \cdot \frac{\partial^2 x}{\partial t^2}(t) \right) \cdot g_i(x(t)) = \sum_{i=1}^{i=n} f_i(x, \dot{x}, \ddot{x}) \cdot g_i(x(t)) \quad (5.33)$$

Chacune des fonctions locales $h_i(x) = a_i \cdot x(t) + b_i \cdot \frac{\partial x}{\partial t}(t) + c_i \cdot \frac{\partial^2 x}{\partial t^2}(t)$ représente une approximation linéaire locale de la dynamique non linéaire globale à approximer. La sortie du modèle $\tilde{y}(t)$ est la somme des estimations fournies par chaque fonction linéaire locale f_i, pondérée par l'activité $g_i(t)$ du neurone «codant l'état» qui lui est associé.

Nous définirons cette population de neurones responsables du codage par population de l'état du système par le terme «**carte de contexte**». Leur activité dépend ici d'une variable d'état du système mais elle peut également être influencée, comme nous le verrons en particulier pour le premier exemple, par des variables n'étant pas des variables d'état (c'est pourquoi nous préférerons ce terme à celui d'«espace d'état» ou de «carte d'état»). Le rayon de la fonction à base radiale définit le «**champ récepteur**» de ces neurones (voir figure 7).

On peut étendre ce principe à une dépendance multidimensionnelle des paramètres de la dynamique, par exemple une dépendance par rapport à $x(t)$ et à $\dot{x}(t)$. Dans ce cas, il faut utiliser un codage bidimensionnel du type (5.30). Il faut noter que cette décomposition en fonctions linéaires ne peut se faire que pour des fonctions «douces» ne variant pas de façon trop brusque dans le domaine d'approximation de leurs paramètres. Pour une fonction donnée, deux paramètres jouent un rôle essentiel dans la capacité du réseau à reproduire les variations de la fonction. Ce sont la résolution (densité de gaussiennes utilisée pour échantillonner l'espace des paramètres) et la largeur de la fonction à base radiale (notée σ). Cette question est abordée dans la discussion ainsi que dans l'annexe qui expose plus en détail les propriétés de ce type d'estimateur.

5.2.5 Construction d'un observateur prédictif

5.2.5.1 Principe

Nous avons appliqué ce modèle à la construction adaptative d'un observateur prédictif d'un système dynamique complexe. Celui-ci est composé d'un bras mécanique à deux degrés de liberté mis en mouvement par quatre muscles (deux pour chaque articulation) modélisés par de simples ressorts exponentiels. Il faut noter en effet que la stabilité d'un tel système mécanique ne peut être assurée par des ressorts linéaires classiques [221, Shadmehr and Arbib, 1992] . Les paramètres mécaniques du bras sont ceux cités dans [253, Uno et al., 1989] . La force exercée par le muscle i est donnée par la relation suivante

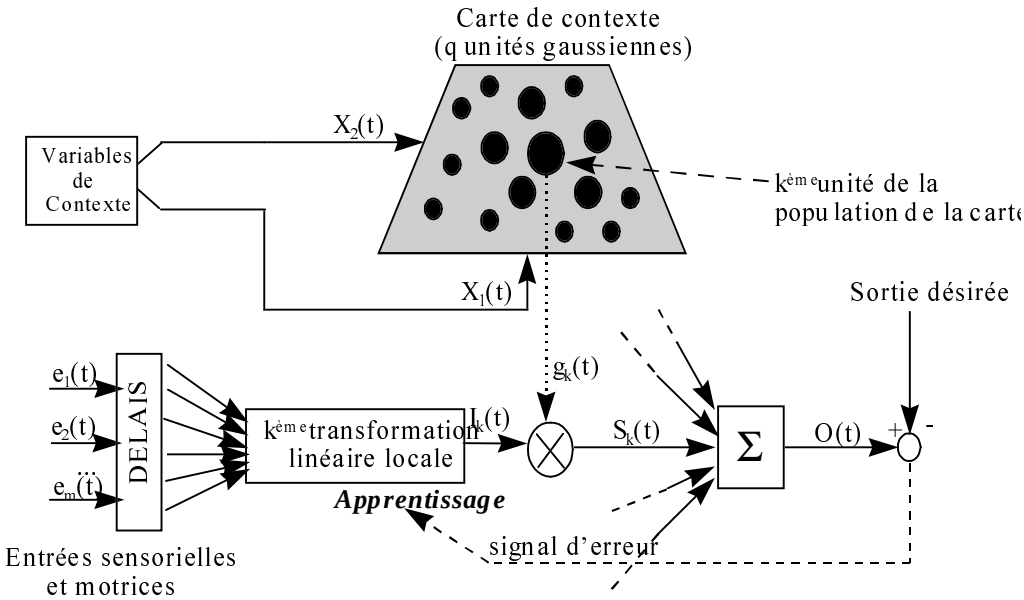

Figure 7. Schéma explicatif de l'approximation d'une fonction non linéaire par une population de fonctions linéaires (ici un approximateur de dimension 2). L'approximateur est constitué de deux modules: une carte de contexte codant par population un couple de variables (X_1, X_2) et une population de neurones dont la réponse est une combinaison des entrées sensorielles et motrices (population de fonctions linéaires). Chaque unité de la population de fonction propose une approximation locale f_k de la fonction. La sortie globale du réseau $O(t)$ est la somme des S_k. Les S_k correspondent à la sortie f_k de chaque module linéaire pondérée par l'activité g_k d'un neurone de la carte de contexte qui lui est associé.

$$f(a_i, l_i) = A_i e^{K_i.(a_i+l_i)} \tag{5.34}$$

où les constantes A_i et K_i permettent de paramétrer la rigidité et la force au repos (longueur et activité nulle) du muscle. Un tel muscle n'est pas capable de «pousser» et se contracte si l'«activité musculaire» a_i est non nulle (dans les simulations suivantes a_i est toujours positif). Deux autres paramètres interviennent dans le calcul de la longeur du muscle (fonction des angles articulaires), ce sont la position de ses deux points d'insertion.

Le rôle de l'observateur prédictif est de prédire dans un futur proche l'évolution du bras articulé (dans un référentiel cartésien ou polaire) à partir de la connaissance des activations musculaires et des déplacements angulaires actuels et passés (voir figure 8). Autrement dit, cet observateur doit réaliser une prédiction sur l'évolution d'un système dynamique multidimensionnel (6 entrées et deux sorties) en effectuant de plus une transformation de coordonnées (transformation également non linéaire). Les équations décrivant la dynamique et la géométrie de ce système sont très complexes et nous ne prétendons pas ici fournir un prédicteur optimal en les utilisant. Le but de ces simulations est avant tout de chercher si l'on peut obtenir une prédiction de bonne qualité en partant d'un nombre limité d'hypothèses avec un modèle le plus simple possible. Pour cela, nous exprimerons de façon restrictive la dynamique directe du système par une relation non linéaire bidimensionnelle posée **a priori** du type:

$$\begin{aligned} y_1(t) &= f_1\left[\overline{a}_1(t), \overline{a}_2(t), \overline{a}_3(t), \overline{a}_4(t), \overline{\theta}_1(t), \overline{\theta}_2(t)\right] \\ y_2(t) &= f_2\left[\overline{a}_1(t), \overline{a}_2(t), \overline{a}_3(t), \overline{a}_4(t), \overline{\theta}_1(t), \overline{\theta}_2(t)\right] \end{aligned} \tag{5.35}$$

où

- $y_1(t)$ et $y_2(t)$ représentent des variables observables telles que les coordonnées cartésiennes ou polaires de l'extrémité du bras dans un plan,
- les $a_i(t)$, $i \in [1, 4]$ sont les vecteurs décrivant les activités musculaires actuelles et passées (sur un intervalle de temps discrétisé fini),
- les vecteurs $\overline{\theta}_1(t)$ et $\overline{\theta}_2(t)$ donnent de la même façon et sur la même fenêtre temporelle les valeurs actuelles et passées des positions angulaires des deux articulations du bras.

On peut écrire

$$\begin{aligned}
\widetilde{y_1}(t+\tau) - y_1(t) &= \frac{\partial f_1}{\partial \overline{a}_1}^T \frac{\overline{\delta a}_1}{\partial t} + \frac{\partial f_1}{\partial \overline{a}_2}^T \frac{\overline{\delta a}_2}{\partial t} + ... + \frac{\partial f_1}{\partial \overline{\theta}_1}^T \frac{\overline{\delta \theta}_1}{\partial t} + \frac{\partial f_1}{\partial \overline{\theta}_2}^T \frac{\overline{\delta \theta}_2}{\partial t} + o_1(t) \qquad (5.36)\\
\widetilde{y_2}(t+\tau) - y_2(t) &= \frac{\partial f_2}{\partial \overline{a}_1}^T \frac{\overline{\delta a}_1}{\partial t} + \frac{\partial f_2}{\partial \overline{a}_2}^T \frac{\overline{\delta a}_2}{\partial t} + ... + \frac{\partial f_2}{\partial \overline{\theta}_1}^T \frac{\overline{\delta \theta}_1}{\partial t} + \frac{\partial f_2}{\partial \overline{\theta}_2}^T \frac{\overline{\delta \theta}_2}{\partial t} + o_2(t)
\end{aligned}$$

Les fonctions $o_1(t)$ et $o_2(t)$ représentent ici le résidu de l'erreur de prédiction observée sur chaque variable. En effet, si on considere que la dynamique du système est effectivement décrite par l'équation (5.35), la prédiction sur l'évolution des deux variables y_1 et y_2 ne sera bonne que pour un horizon τ infiniment petit. Pour des horizons grands, l'évolution du système dépend également des commandes musculaires futures (information non disponible) et la prédiction devient alors très hasardeuse. Il faut noter que les dérivées partielles du type $\frac{\partial f_1}{\partial \overline{a}_1}$ dépendent à priori de toutes les variables $\left\{\overline{a}_1(t), \overline{a}_2(t), \overline{a}_3(t), \overline{a}_4(t), \overline{\theta}_1(t), \overline{\theta}_2(t)\right\}$. Pour tenir compte de cette dépendance éventuelle, il serait nécessaire de définir la fonction d'accord sur un nombre de variables très important et par conséquent de réaliser un approximateur travaillant sur un espace de dimension déraisonnablement grande. C'est pourquoi nous avons apporté une seconde restriction à la résolution de ce problème en postulant que les dérivées partielles des fonctions $f_1(t)$ et $f_2(t)$ ne dépendent que des deux variables $\theta_1(t)$ et $\theta_2(t)$ exprimant la position angulaire instantanée du bras.

Compte tenu de ces restrictions, la dynamique de l'observateur prédictif, construit comme une somme d'approximation linéaires locales de la dynamique (5.36) peut s'écrire:

$$\begin{aligned}
\widetilde{\Delta y_1}(t) &= \sum_{i=1}^{i=n}\sum_{j=1}^{j=n} \left[\overline{\alpha}_{1,1}^{i,j}\frac{\overline{\delta a}_1}{\partial t} + \overline{\alpha}_{1,2}^{i,j}\frac{\overline{\delta a}_2}{\partial t} + \overline{\alpha}_{1,3}^{i,j}\frac{\overline{\delta a}_3}{\partial t} + \overline{\alpha}_{1,4}^{i,j}\frac{\overline{\delta a}_4}{\partial t} + \overline{\alpha}_{1,5}^{i,j}\frac{\overline{\delta \theta}_1}{\partial t} + \overline{\alpha}_{1,6}^{i,j}\frac{\overline{\delta \theta}_2}{\partial t}\right] \cdot g_{i,j}(\theta_1, \theta_2)\\
\widetilde{\Delta y_2}(t) &= \sum_{i=1}^{i=n}\sum_{j=1}^{j=n} \left[\overline{\alpha}_{2,1}^{i,j}\frac{\overline{\delta a}_1}{\partial t} + \overline{\alpha}_{2,2}^{i,j}\frac{\overline{\delta a}_2}{\partial t} + \overline{\alpha}_{2,3}^{i,j}\frac{\overline{\delta a}_3}{\partial t} + \overline{\alpha}_{2,4}^{i,j}\frac{\overline{\delta a}_4}{\partial t} + \overline{\alpha}_{2,5}^{i,j}\frac{\overline{\delta \theta}_1}{\partial t} + \overline{\alpha}_{2,6}^{i,j}\frac{\overline{\delta \theta}_2}{\partial t}\right] \cdot g_{i,j}(\theta_1, \theta_2)
\end{aligned}$$

avec $\widetilde{\Delta y_1}(t) = \widetilde{y_1}(t+\tau) - y_1(t)$ et $\widetilde{\Delta y_2}(t) = \widetilde{y_2}(t+\tau) - y_2(t)$.

Les fonctions à base radiale utilisées lors des simulations sont des fonctions gaussiennes

$$g_{i,j}(\theta_1, \theta_2) = \exp\left(-p_0.\left[(\theta_1 - \theta_i)^2 + (\theta_2 - \theta_j)^2\right]\right)$$

le paramètre p_0 permettant de contrôle leur rayon («champ récepteur»).

Pour simplifier l'écriture des équations associées à ce modèle, il est possible d'écrire de façon plus générale (sans expliciter les variables d'entrée)

$$\begin{aligned}\widetilde{\Delta y_1}(t) &= \sum_{i=1}^{i=n}\sum_{j=1}^{j=n}\left[\sum_{k=1}^{k=n_e}\alpha_{1,k}^{i,j}.e_k\right]\cdot g_{i,j}(\theta_1,\theta_2) \\ \widetilde{\Delta y_2}(t) &= \sum_{i=1}^{i=n}\sum_{j=1}^{j=n}\left[\sum_{k=1}^{k=n_e}\alpha_{2,k}^{i,j}.e_k\right]\cdot g_{i,j}(\theta_1,\theta_2)\end{aligned} \tag{5.37}$$

si l'on nomme n_e le nombre d'entrées considérées.

Remarquons de plus que le système d'équations (5.35) est introduit comme une fonction statique des activités musculaires et des angles articulaires. Or le système modélisé est un système dynamique et les sorties $y_1(t)$ et $y_2(t)$ dépendent non seulement des activités musculaires et des angles articulaires mais aussi de leurs dérivées successives. Il est possible de prendre en compte ces opérations de dérivation dans le modèle en introduisant des entrées supplémentaires correspondant aux variables $e_k, k \in [1, n_e]$ précédemment introduites retardées d'un multiple entier d'un retard τ. Ainsi, le modèle d'observateur prédictif final prend en entrée les variables suivantes

- les variations des activités musculaires $\left\{\frac{\overline{\delta a}_1}{\partial t}(t), \frac{\overline{\delta a}_2}{\partial t}(t), \frac{\overline{\delta a}_3}{\partial t}(t), \frac{\overline{\delta a}_4}{\partial t}(t)\right\}$ mais aussi les variations des activités musculaires précédentes (sur un intervalle de temps fini et avec un échantillonnage fixe de période τ) soit$\left\{\frac{\overline{\delta a}_1}{\partial t}(t-\tau), \frac{\overline{\delta a}_2}{\partial t}(t-\tau), \frac{\overline{\delta a}_3}{\partial t}(t-\tau), \frac{\overline{\delta a}_4}{\partial t}(t-\tau)\right\}$, $\left\{\frac{\overline{\delta a}_1}{\partial t}(t-2\tau), \frac{\overline{\delta a}_2}{\partial t}(t-2\tau), \frac{\overline{\delta a}_3}{\partial t}(t-2\tau), \frac{\overline{\delta a}_4}{\partial t}(t-2\tau)\right\}$, ...,$\left\{\frac{\overline{\delta a}_1}{\partial t}(t-n_\tau\tau), \frac{\overline{\delta a}_2}{\partial t}(t-n_\tau\tau), \frac{\overline{\delta a}_3}{\partial t}(t-n_\tau\tau), \frac{\overline{\delta a}_4}{\partial t}(t-n_\tau\tau)\right\}$,
- les vitesses angulaires $\left\{\frac{\overline{\delta\theta}_1}{\partial t}(t), \frac{\overline{\delta\theta}_2}{\partial t}(t)\right\}$ ainsi que leurs valeurs précédentes $\left\{\frac{\overline{\delta\theta}_1}{\partial t}(t-\tau), \frac{\overline{\delta\theta}_2}{\partial t}(t-\tau)\right\}$, $\left\{\frac{\overline{\delta\theta}_1}{\partial t}(t-2\tau), \frac{\overline{\delta\theta}_2}{\partial t}(t-2\tau)\right\}$,..., $\left\{\frac{\overline{\delta\theta}_1}{\partial t}(t-n_\tau\tau), \frac{\overline{\delta\theta}_2}{\partial t}(t-n_\tau\tau)\right\}$.

5.2.5.2 Règle d'apprentissage

Représenter une fonction dynamique en utilisant une population de fonctions permet d'utiliser une règle d'apprentissage locale. Toutes les unités de calcul reçoivent le même signal d'erreur et adaptent localement leurs paramètres. L'apprentissage ne nécessite aucune rétropropagation complexe d'information. En ce qui concerne cet observateur prédictif, le signal d'erreur est donné par la différence entre la prédiction $\left[\widetilde{\Delta y_1}(t), \widetilde{\Delta y_2}(t)\right]$ réalisée au temps t et les variations observées au temps $(t+\tau)$. Si l'on définit les signaux d'erreurs comme

$$\begin{aligned}\varepsilon_1(t-\tau) &= \widetilde{\Delta y_1}(t-\tau) - (y_1(t) - y_1(t-\tau)) \\ \varepsilon_2(t-\tau) &= \widetilde{\Delta y_2}(t-\tau) - (y_2(t) - y_2(t-\tau))\end{aligned}$$

et que l'algorithme d'apprentissage cherche à minimiser les variables $J_1 = \varepsilon_1(t)^2$ et $J_2 = \varepsilon_2(t)^2$, alors

l'application de la règle de Widrow-Hoff donne la règle

$$\alpha_{1,k}^{i,j} \longleftarrow \alpha_{1,k}^{i,j} - \sigma \cdot \frac{\delta J_1}{\delta \alpha_{1,k}^{i,j}}(t-\tau) \tag{5.38}$$

avec

$$\frac{\delta J_1}{\delta \alpha_{1,k}^{i,j}}(t-\tau) = \varepsilon_1(t-\tau) \cdot g_{i,j}(\theta_1(t-\tau), \theta_2(t-\tau)) \cdot e_k(t-\tau)$$

et σ réel positif définissant le pas de la descente de gradient. L'application de cette règle ne pouvant se faire qu'au temps t (lorsque les valeur de $y_1(t)$ et $y_2(t)$ sont connues), la modification des poids est effectuée seulement à ce instant là. Autrement dit, le processus d'apprentissage s'effectue avec un déphasage de τ secondes sur l'évolution du bras articulé. Une transposition au domaine biologique de cette contrainte impliquerait donc que **l'horizon de prédiction de l'observateur corresponde au délai de conduction du signal d'erreur** $\varepsilon_1(t-\tau) \cdot g_{i,j}(\theta_1(t-\tau), \theta_2(t-\tau)) \cdot e_k(t-\tau)$.

5.2.5.3 Simulations

Le système dynamique constitué du bras articulé mis en mouvement par 4 muscles élastiques est trop complexe pour que nous proposions un schéma de contrôle incluant notre observateur prédictif. Nous avons par conséquent choisi de tester ses capacités a posteriori. Nous avons généré des mouvements aléatoires du bras en produisant des séries temporelles $[a_1(t), a_2(t), a_3(t), a_4(t)]$ pseudo aléatoires. Chaque signal d'activité musculaire a été obtenu par un mélange de sinusoides dont les composantes fréquentielles sont choisies dans une bande de fréquences limitée. Durant les simulations, les activités musculaires, ainsi que la trajectoire du bras, sont enregistrées dans un fichier. Nous avons donc finalement obtenu une base de données (la base de test) contenant le décours temporel des activités musculaires et la cinématique du bras ceci pour plusieurs mouvements.

La qualité de la prédiction est estimée à partir de l'erreur quadratique moyenne, sous la forme d'un **pourcentage de variance exprimée (P.V.E.)**. Ces mesures sont données pour chacune des variables d'intérêt Δy_1 et Δy_2 par les formules

$$\begin{aligned} P.V.E_{\Delta y1} &= 100 \cdot \left(1. - \frac{Var(\varepsilon_1)}{Var(\Delta y_1)}\right) \\ P.V.E_{\Delta y1} &= 100 \cdot \left(1. - \frac{Var(\varepsilon_1)}{Var(\Delta y_1)}\right) \end{aligned}$$

qui expriment le pourcentage de variance expliquée par l'observateur par rapport au pourcentage de variance qui serait expliqué par un modèle utilisant comme prédiction simplement les valeurs moyennes de

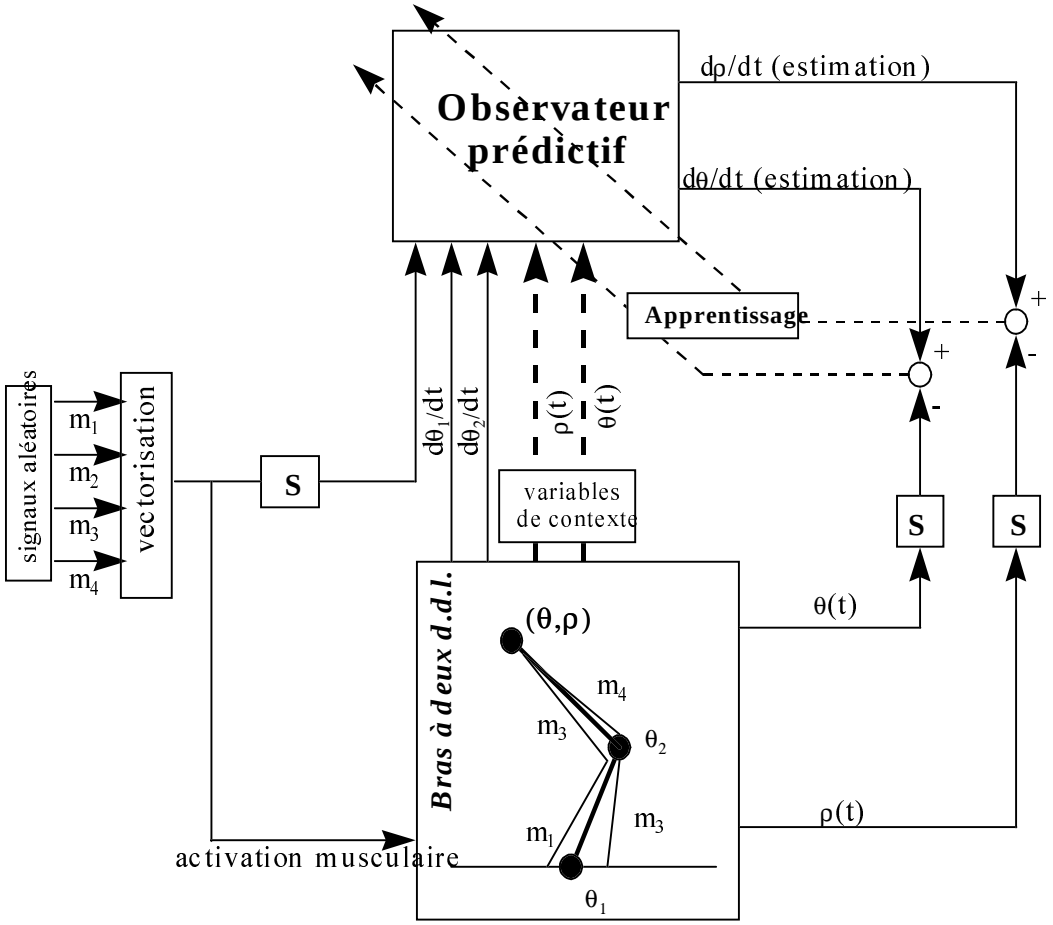

Figure 8. Schéma illustrant la simulation d'un observateur prédictif appliqué à la prédiction des déplacements d'un bras articulé à deux degrés de liberté (la trajectoire du bras est ici représentée en coordonnées polaires). Les mouvements du bras sont aléatoires et les variables d'état utilisées par l'observateur prédictif sont les coordonnées polaires $[\theta(t), \rho(t)]$ de l'extrémité du bras. Les variables sensorielles constituant les entrées (motrices et sensorielles) de l'observateur sont les variations d'activité musculaire $[\dot{a}_1, \dot{a}_2, \dot{a}_3, \dot{a}_4]$ et les vitesses angulaires des articulations $\dot{\theta}_1$ et $\dot{\theta}_2$.

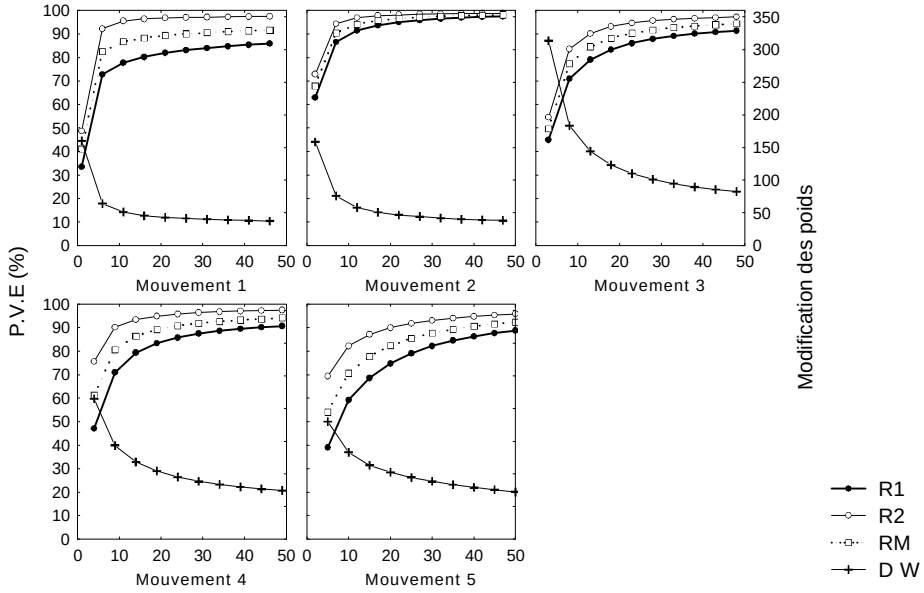

Figure 9. Résultat de la prédiction dans un référentiel cartésien de l'évolution d'un bras articulé mis en mouvement par quatre muscles. R1 et R2 représentent le pourcentage de variance expliquée (P.V.E.) par le modèle (sur la prédiction des variations en y_1 et en y_2). RM est le pourcentage de variance expliquée moyen. DW représente une mesure des variations des poids synaptiques du réseau dues à l'apprentissage.

Δy_1 et de Δy_2. Ces mesures sont directement reliées aux coefficients de corrélation des regressions linéaires de $\widetilde{\Delta y_1}$ en fonction de Δy_1 et de $\widetilde{\Delta y_2}$ en fonction de Δy_2. En effet,

$$\rho_{\Delta y1} = \sqrt{1. - \frac{Var(\varepsilon_1)}{Var(\Delta y_1)}}$$

$$\rho_{\Delta y1} = \sqrt{1. - \frac{Var(\varepsilon_1)}{Var(\Delta y_1)}}$$

Le nombre de synapses adaptatives présentes dans le réseau dépend donc du nombre de fonctions à base radiale n et du nombre de retards utilisés n_τ pour la prise en compte des dérivées successives des entrées. Ainsi $n_e = 6 + n_\tau \cdot 6$ et le nombre total de synapses est

$$n_s = n^2 * ne$$

Par exemple, un réseau comportant une grille de $6 \cdot 6 = 36$ gaussiennes, avec 3 retards doit ainsi apprendre $36 \cdot (6 + 4 \cdot 6) = 1080$ valeurs flottantes.

5.2.5.4 Résultats

Dans l'exemple choisi les résultats obtenus portent sur la prédiction de cinq mouvements répétés chacun dix fois seulement. Les mouvement, qui durent 40 secondes et sont échantillonnés à 1kHz, sont présentés au réseau selon une séquence régulière (mouvement 1, puis 2, 3, 4 et 5, répétée dix fois). La carte de contexte comprend 25 gaussiennes réparties de façon régulière sur une grille de 5 par 5. Pour chaque signal d'entrée (les quatres activités musculaires et les deux vitesses angulaires articulaires), le réseau reçoit 3 copies de ces signaux retardées (respectivement de 150, 300 et 450 ms). L'horizon de prédiction est de 400 ms. Le pourcentage de variance expliquée moyen dépasse en fin de simulation dans tous les cas 90 % (ce qui correspond à un coefficient de régression de 0.949). La prédiction de l'évolution de y_1 (c'est à dire de l'abscisse de l'extrémité du bras) semble clairement plus difficile à apprendre comme l'atteste l'évolution du pourcentage de variance expliquée R1 sur la figure (9), toujours inférieur à R2. La décroissance régulière, d'allure exponentielle, de la mesure des variations des poids synaptiques montre que l'apprentissage est régulier et tout à fait efficace. Il faut noter que les courbes d'évolution temporelle des différents pourcentages de variance expliquée sont strictement monotone. Cette observation montre les capacités de généralisation du réseau de gaussiennes. En effet, ce réseau pourrait a priori apprendre une dynamique propre à chaque mouvement de façon indépendante, et «désapprendre» ensuite lorsque un mouvement différent lui est présenté.

5.2.6 Application au contrôle d'un système non-linéaire monodimensionnel

5.2.6.1 Principe

Dans le chapitre précédent, nous avons présenté une application du filtre adaptatif proposé à la prédiction de l'évolution d'un système dynamique non linéaire. Cette seconde application de l'approximation par populations de fonctions est destinée au contrôle adaptatif d'un système dynamique également non linéaire. Le réseau de populations de fonctions dynamiques est utilisé comme un filtre adaptatif inclu dans un schéma de contrôle du type «apprentissage d'erreur en retour» («feedback error learning»). Le système contrôlé est une version simplifiée (monoarticulaire) du modèle de bras articulé utilisé dans les simulations précédentes. La tâche demandée au contrôleur est une tâche de poursuite: il doit imprimer à l'articulation une trajectoire angulaire aussi proche que possible de celle d'une cible. Cette simulation permet de montrer qu'un réseau de gaussiennes utilisant le principe du codage par population peut être utilisé pour compenser les non-linéarités d'un système dynamique.

Le système contrôlé est donc une version monoarticulaire du bras simulé (la seconde articulation étant bloquée comme si ce bras restait tendu). Ses caractéristiques mécaniques sont les suivantes:

- masse $m = 2$ kg,
- longueur $l = 0.6$ m,
- coefficient de viscosité $b = 0.08$ kg.m^2.s^{-1}.

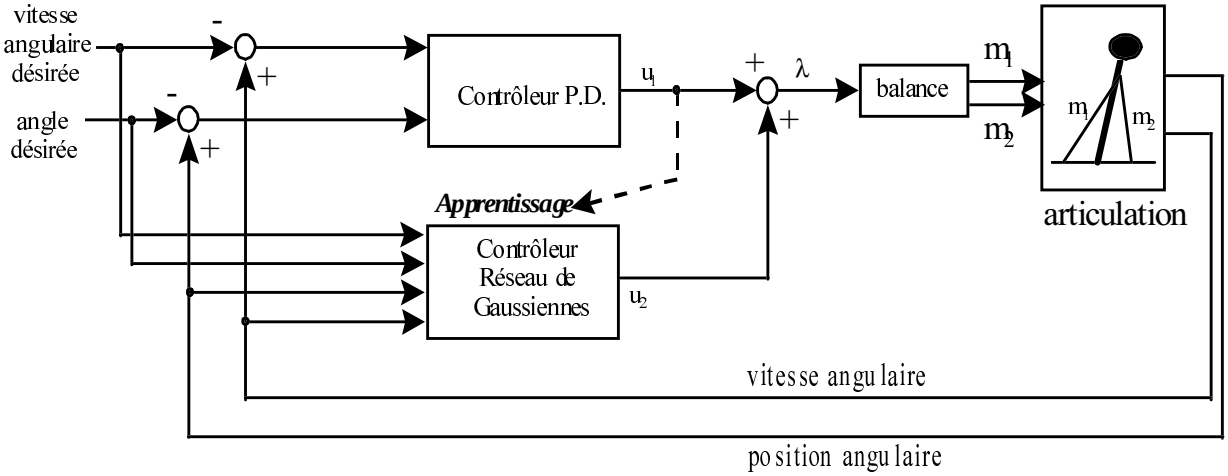

Figure 10. Schéma de contrôle utilisé pour tester la capacité d'un réseau de gaussienne à modéliser la dynamique d'un système nonlinéaire

Ce bras tendu est soumis à l'influence de la gravité. Les caractéristiques des muscles sont données par $A = 1.5$ et $K = 2$ (voir équation 5.34).

Le schéma de contrôle choisi utilise deux modules complémentaires. Le premier, non adaptatif, est simplement un contrôleur proportionnel-dérivé (PD) dont les paramètres sont fixes (servo-contrôleur) et choisis de façon arbitraire. Sa sortie est définie par la relation

$$u_{PD}(t) = K_p.e(t) + K_d.e'(t)$$

avec

- $e(t) = \theta(t) - \theta_d(t)$, différence entre la position angulaire instantanée et la position angulaire désirée,
- $e'(t) = \theta'(t) - \theta'_d(t)$, erreur de vitesse angulaire,
- $K_p = 0.9$ et $K_d = 0.1$, gains en position et en vitesse du servocontrôleur.

Utilisé seul, ce contrôleur linéaire ne peut assurer de façon générale la stabilité du contrôle. La commande $u_{PD}(t)$ qu'il produit est utilisée comme signal d'erreur pour un second module possédant des propriétés adaptatives. Ce second module est tout à fait identique, dans son architecture, à l'observateur prédictif décrit dans l'exposé du modèle précédent (voir 5.37). Cependant,

- la carte de contexte n'est pas ici définie uniquement dans un intervalle de positions angulaires instantanées, mais dans une région bidimensionnelle de l'espace des phases $\left[\theta(t), \theta'(t)\right]$ du système

monoarticulaire considéré ici. l'activité des neurones de cette carte est décrite par la relation $n_{ij}(t) = e^{\left[-\gamma.\left((\theta(t)-\theta_{i,j})^2+(\theta'(t)-\theta'_{i,j})\right)\right]}$ où $\left[\theta_{i,j}, \theta'_{i,j}\right]$ représente le «contexte préféré» du neurone situé sur la $i^{ème}$ ligne et $j^{éme}$ colonne de la carte et γ la pente ou sensibilité de la courbe d'accord. Autrement dit, la commande $\Psi(t)$ produite par ce module adaptatif est une somme de fonctions dynamiques localement linéaires, pondérée par les activités observées dans la carte de contexte. La fonction dynamique choisie à un instant t donné dépend donc de la position angulaire et de la vitesse angulaire du bras à cet instant.

- ces fonctions dynamiques sont une combinaison linéaire des positions et vitesses angulaires et des positions et vitesses angulaires désirées (la consigne).

La commande totale, calculée comme la somme des sorties des deux modules ($u(t) = u_{PD}(t) + \Psi(t)$) est utilisé comme un signal d'activation des muscles. La contraction des deux muscles n'est pas indépendante: la commande $u(t)$ est distribuée à chacun des deux par un mécanisme de «balance» (on ne s'intéresse pas ici à la résolution de la redondance dynamique articulaire). Seul la valeur absolue de $u(t)$ est considérée et son signe détermine vers quel muscle ce signal est dirigé (le muscle antagoniste reçoit dans ce cas une activation nulle). Ce système très simple permet de conserver une certaine plausibilité biologique à notre modèle car la commande motrice provoque uniquement la contraction musculaire et ne peut produire de forces de dilatation.

5.2.6.2 La simulation

Nous avons choisi de comparer le comportement de deux modèles de contrôle dans une tâche de poursuite. Le premier modèle ne comporte que la partie non adaptative du schéma de contrôle décrit précédemment (le servocontroleur produisant $u_{PD}(t)$). Le second modèle est le modèle complet, comportant un second module adaptatif Ψ. Ces deux systèmes commandent le même modèle de bras et doivent poursuivre une trajectoire composée d'un mélange de cinq sinusoides dont les fréquences et les déphasages ont été tirés au hasard (avec une limite supérieure en fréquence de 1,5 Hz). La carte adaptative comporte $6 \times 6 = 36$ neurones disposés selon une grille régulière, et la pente de la fonction d'accord des neurones d'états est 2.

5.2.6.3 Résultats

Les résultats obtenus concernent une simulation simultanée des deux modèles durant 70 secondes. L'apprentissage est déclenché après les 6 premières secondes, et arrêté au bout de 40 secondes. Le fait marquant de cette simulation est la divergence du modèle non adaptatif aux environs de la 45ème seconde de simulation (figure 11, graphique du haut). Le modèle adaptatif n'est pas déstabilisé alors que l'apprentissage n'est plus actif (figure 11, en bas). De plus, la trajectoire du système adaptatif suit très étroitement celle de la cible, même lorsque le filtre Ψ n'apprend plus. Cette simulation montre qu'une adaptation assez rapide permet au second système

- d'améliorer de façon très importante la qualité de la poursuite,

- de réduire le déphasage observé entre la consigne et la trajectoire produite, c'est à dire de faire montre de capacités d'anticipation,
- d'augmenter la stabilité de la commande.

5.3 Application au modèle de la mémoire dynamique

5.3.1 La mémoire dynamique

Le modèle de la mémoire dynamique ([58, Droulez et Berthoz, 1991]) propose une hypothèse originale concernant la nature des transformations spatiotemporelles impliquées dans le contrôle des mouvements des yeux. La commande centrale a pour origine les neurones tecto-reticulo-spinaux mais aussi les neurones «bursters» prémoteurs du colliculus qui tous semblent produire un signal de commande corrélé à la vitesse de déplacement de l'oeil. Se pose donc la question du maintien, par le système nerveux central, d'une représentation de la cible à atteindre pendant le mouvement. Cette mémoire de la cible peut fonctionner par le biais d'une opération d'intégration temporelle du signal de commande en vitesse (une «copie efférente» de la commande motrice). C'est précisément ce que propose le modèle de la mémoire dynamique: l'intégrateur comprend une représentation de la cible à atteindre sous la forme d'une carte dont les neurones possèdent des champs récepteurs rétinotopiques. L'intérêt de ce modèle est donc de montrer que cette représentation peut être maintenue au cours du mouvement par un processus d'intégration non linéaire d'une copie efférente de la commande en vitesse de l'oeil. D'autre part, les motoneurones enregistrés dans les noyaux oculomoteurs ont une activité à la fois tonique et phasique. Une autre phase d'intégration du signal provenant des neurones tecto-réticulo-spinaux est nécessaire pour expliquer la présence de la composante tonique de la décharge des motoneurones durant les saccades. Le modèle de la mémoire dynamique peut également être invoqué pour réaliser cette autre opération d'intégration (réalisée par l'«intégrateur oculomoteur final» [57, Droulez et Berthoz, 1991]).

Pour une carte de dimension un (intervenant par exemple contrôle des saccades horizontales), si la distribution de l'activité dans la carte (la représentation) est décrite par une fonction $f(x,t), x \in \Re, t \in [0,+\infty]$ ou t représente le temps et x la position, alors on peut faire correspondre un déplacement de la distribution d'activité de la carte à un déplacement dx de la position de la cible sur la rétine en un temps dt si la différentielle totale reste nulle (contrainte de rigidité) c'est à dire

$$\forall x, f(x+dx, t+dt) = f(x,t) \tag{5.39}$$

Un développement de Taylor du membre gauche de cette équation donne

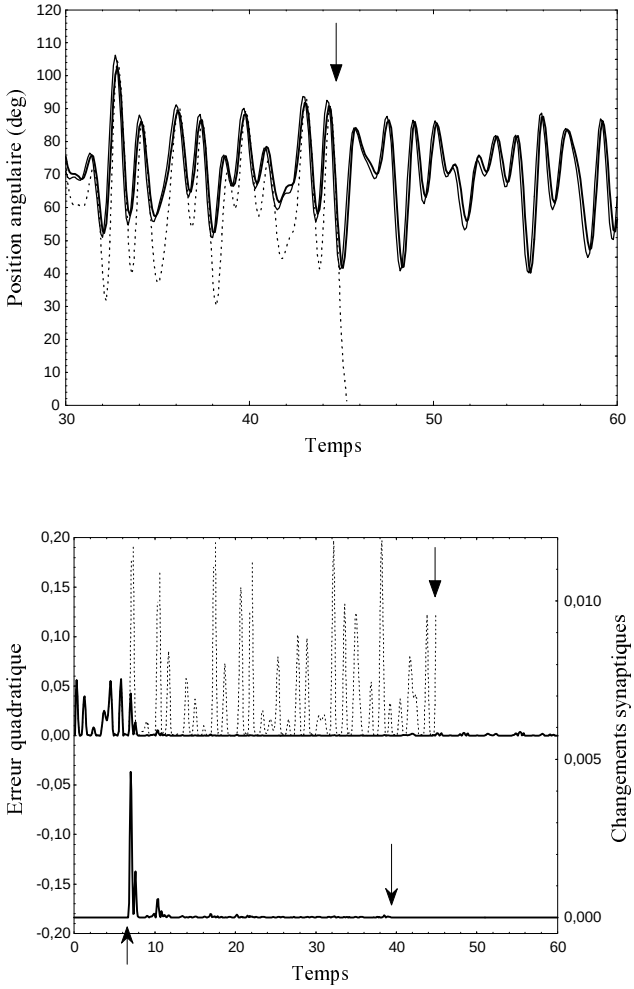

Figure 11. Résultats de la simulation. En haut, un extrait de la trajectoire du modèle non adaptatif (tirets noirs), de celle du modèle adaptatif (courbe en gras) et de la trajectoire désirée (courbe en traits fins). La flèche verticale indique le décrochage du modèle non adaptatif dont la trajectoire diverge ensuite. La figure du bas, échelle de gauche, décrit l'évolution de l'erreur quadratique (carré de la sortie du P.D.) en fonction du temps de simulation. Une flèche verticale indique l'instant du décrochage. La courbe en gras associée à l'échelle de droite montre l'évolution du carré des modifications des poids synaptiques entre le début et la fin de la période d'apprentissage (indiqués par les deux flèches du bas).

$$f(x+dx,t+dt) = f(x,t) + \frac{\delta f}{\delta x}dx + \frac{\delta f}{\delta t}dt \tag{5.40}$$

Ceci implique donc, si l'on compare les relations (5.39) et (5.40) que

$$\frac{\delta f}{\delta t} = \frac{\delta f}{\delta x} \cdot \frac{dx}{dt} = \frac{\delta f}{\delta x} \cdot \dot{x} \tag{5.41}$$

Cette dernière équation montre que les variations temporelles et spatiales de la fonction $f(x,t)$ sont liées par une relation simple impliquant la vitesse de déplacement de la représentation. Si l'on passe d'une représentation continue à une représentation discrète en décrivant l'activité de n neurones équidistribués sur la carte comme $n_i(t) = f(i \cdot \mu, t), \mu \in \Re$. Alors

$$n_i(t+dt) = n_i(t) + \frac{\delta f}{\delta t}(x,t) \cdot dt = n_i(t) + \frac{\delta f}{\delta x}(x,t) \cdot \dot{x} \cdot dt$$

Si la cible est fixe et si l'on note V la vitesse de déplacement horizontale de l'oeil, alors $\dot{x} = -V$ et par conséquent

$$n_i(t+dt) = n_i(t) - \frac{\delta f}{\delta x}(x,t) \cdot V \cdot dt \tag{5.42}$$

L'équation (5.42) montre que la variation d'activité temporelle du neurone ($\frac{n_i(t+dt)-n_i(t)}{dt}$) peut être calculée à partir de la variation d'activité locale ($\frac{\delta f}{\delta x}(x,t)$) dans la carte et de la commande en vitesse V. Or la variation locale d'activité peut être approximée à partir des activités des neurones voisins. On peut par exemple écrire

$$\widehat{\frac{\delta f}{\delta x}(x,t)} \cong \frac{n_{i+1}(t) - n_i(t)}{\mu}$$

ou d'une manière plus générale

$$\widehat{\frac{\delta f}{\delta x}(x,t)} \cong \sum_{j=1}^{j=n} \varpi_{i,j} n_j(t) \tag{5.43}$$

les poids $\varpi_{i,j}$ du réseau pouvant être acquis par apprentissage [57, Droulez et Berthoz, 1991] . D'une manière générale, l'opération réalisée par l'équation (5.43) est une **convolution de l'activité locale dans la carte par un filtre dériveur** (filtre passe-bande). La qualité de l'estimation du gradient dépend donc des

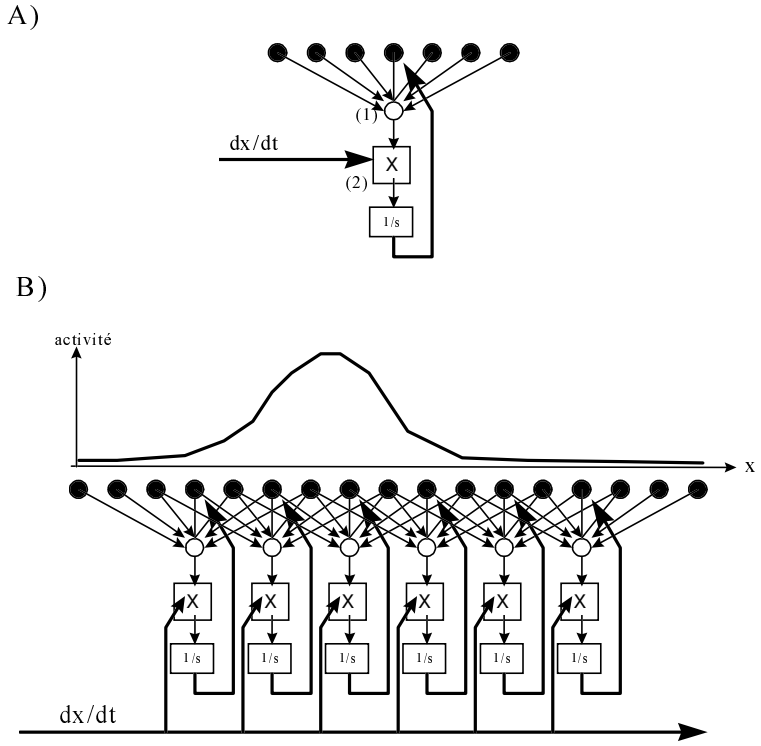

Figure 12. Principe de la mémoire dynamique. A) Unité fonctionnelle comportant deux éléments principaux. Le premier (1) effectue une somme pondérée des activités des neurones voisins (ici les six voisins les plus proches) et produit une estimation du gradient spatial local d'activité. Le second élément (2) multiplie ce gradient par le signal de vitesse. Le signal résultant est ensuite intégré. B) Assemblage des unités fonctionnelles (par souci de clarté, seule une sur deux est représentée).

propriétés de ce filtre et des fréquences spatiales présentes dans la carte codant la variable x. Comme il n'existe pas de filtre aux qualités parfaites, chaque calcul de gradient introduit une erreur dans le processus d'intégration. La mémoire dynamique ne peut donc réaliser une intégration pendant un temps infini, et l'activité dans la carte doit être régulièrement «rafraîchie» par les informations provenant des capteurs sensoriels (informations rétiniennes par exemple).

Ainsi, l'évolution temporelle de chacun des neurones de la carte peut donc s'exprimer en fonction de la commande V et des activités des neurones voisins. Cette architecture permet de combiner une intégration temporelle et un changement de référentiel par la construction d'un réseau récurrent dont une caractéristique est de nécessiter des «neurones produits» (voir figure 12).

Contrairement aux deux modèles présentés précédemment, le modèle de la mémoire dynamique est un réseau à connexions récurrentes. Or s'il est clair que le réseau effectue globalement une intégration linéaire, la fonction réalisée par chacune des unités neuronales est non linéaire. Nous allons essayer de développer cette idée en l'appliquant à un codage par population et en étendant son principe à des systèmes dynamiques d'ordre

plus élevé, linéaires ou non linéaires. Nous restreindrons cette étude à un système monodimensionnel (une seule variable d'intérêt) bien que tous les résultats obtenus puissent s'appliquer à des systèmes de dimension plus élevée.

5.3.2 Apport du codage par population

5.3.2.1 Calcul exact des poids intervenant dans le calcul du gradient spatial

Dans ce modèle, la distribution d'activité initiale $f(x,t)$ est donnée par l'image rétinienne. On peut cependant appliquer le même raisonnement à une variable codée par une population de neurones à fonction d'accord. Prenons l'exemple d'une population de n neurones à fonctions d'accord gaussiennes $n(y,y_i)$ de largeur σ. L'activité de la population code la valeur d'une variable $y(t)$ selon la loi

$$\widehat{y} = \sum_{i=1}^{i=n} y_i \cdot e^{-\frac{(y-y_i)^2}{\sigma^2}} = \sum_{i=1}^{i=n} y_i \cdot n_i(y,y_i) \tag{5.44}$$

La variation temporelle du i-ème neurone de la carte est donné par la relation

$$\frac{dn_i}{dt} = \frac{dn_i(t)}{dy} \cdot \frac{dy}{dt} = -2\frac{(y-y_i)}{\sigma^2} \cdot n_i \cdot \frac{dy}{dt}$$

Or, dans cette équation, la variable $y(t)$ peut être remplacée par son estimation (5.44) et finalement, l'évolution temporelle d'un neurone de la population peut être décrite par l'équation différentielle suivante

$$\frac{dn_i(t)}{dt} = -\frac{2}{\sigma^2}\left[\sum_{j=1}^{j=n} y_j \cdot n_j \cdot n_i - y_i \cdot n_i\right] \cdot \dot{y}$$

Cette dernière équation nous donne donc les poids optimaux de la matrice d'interconnection des neurones de la carte. Le comportement de celle-ci est donc décrit par $(2n+1)$ équations différentielles non linéaires du premier ordre couplées entre elles. Notons que nous avons effectué un calcul direct de la connectivité mais elle peut également être apprise par le réseau. L'équation ci-dessus montre que l'architecture optimale est un réseau récurrent composé d'une couche d'unités sigma-pi produisant la sortie $\Pi_i(t)$ avec des poids qui ont pour valeur

$$\varpi_{i,j} = -\frac{2}{\sigma^2} \cdot y_j$$

Avec ce formalisme, la dynamique de chaque neurone de la couche s'écrit

$$\frac{dn_i(t)}{dt} - \frac{2}{\sigma^2} \cdot y_i \cdot n_i(t) \cdot \dot{y}(t) = -\Pi_i(t) \cdot \dot{y}(t)$$

Si la variable $\dot{y}(t)$ est elle aussi codée par une population de n neurones possédant une fonction d'accord $r(\dot{y}, \dot{y}_i)$ définie par des vitesses préférées, il est possible de rempacer $\dot{y}(t)$ par son estimation $\widehat{\dot{y}}(t) = \sum_{i=1}^{i=n} \dot{y}_i \cdot r(\dot{y}, \dot{y}_i)$. De même, la valeur de y peut être remplacée par son estimation $\hat{y}(t) = \sum_{i=1}^{i=n} y_i \cdot n_i$, ce qui permet d'obtenir la dynamique suivante pour chaque élément $n_j, j \in [1, n]$ du réseau :

$$\frac{dn_j(t)}{dt} - \frac{2}{\sigma^2} \cdot n_j(t) \cdot \left(\sum_{i=1}^{i=n} y_i n_i \right) \cdot \left(\sum_{i=1}^{i=n} \dot{y}_i r(\dot{y}, \dot{y}_i) \right) = -\Pi_j(t) \cdot \sum_{i=1}^{i=n} \dot{y}_i \cdot r(\dot{y}, \dot{y}_i)$$

.Ce calcul est propre à l'utilisation d'une fonction d'accord gaussienne, mais n'oublions pas que le calcul direct de $\frac{dn_i(t)}{dy}$ peut être remplacé par une estimation locale du gradient dans la carte.

5.3.2.2 Réalisation d'un filtre du premier ordre

Il est possible de réaliser un système dynamique du premier ordre en utilisant à la fois le principe de la mémoire dynamique et celui du codage par population. Si la dynamique du filtre peut être décrite par l'équation

$$\dot{y} = \Phi(y) + u \tag{5.45}$$

alors un principe très proche de celui décrit dans le chapitre précédent permet d'obtenir une mémoire simulant la dynamique de $y(t)$. Si l'on considère que l'on peut approximer localement la fonction $\Phi(y)$ il est possible d'associer à chaque neurone de la carte n_i une constante a_i telle que $\Phi(y_i) \backsimeq a_i$ et d'obtenir une estimation de la vitesse de déplacement de la représentation :

$$\widehat{\dot{y}}(t) = \sum_{i=1}^{i=n} n_i(t) \cdot a_i + u(t)$$

Ainsi, la dynamique du système (5.45) peut être reproduite par une carte neuronale dont l'évolution est décrite par la dynamique locale

$$\frac{dn_p}{dt} = \sum_{j=1}^{j=n} \varpi_{p,j} n_j(t) \cdot \left(\sum_{i=1}^{i=n} a_i n_i(t) + u(t) \right)$$

Ce modèle, qui utilise pour approximer la fonction $\Phi(y)$ le principe de codage par population, n'est rien d'autre qu'une version récurrente du modèle décrit dans la section précédente de ce chapitre. Il est par exemple très facile d'obtenir un comportement dynamique du type «intégrateur à fuite» linéaire en utilisant ce principe. La fonction de transfert d'un tel intégrateur est du type

$$\dot{y} = -\frac{1}{\tau}y + \beta u$$

ou τ représente la constante de temps du filtre et β son gain. Pour obtenir une mémoire dynamique se comportant comme cet intégrateur, il suffit de poser

$$a_i = -\frac{1}{\tau \cdot \sum_{i=1}^{i=n} n_i(t)}$$

et d'appliquer le gain β à l'entrée $u(t)$.

5.3.3 Extension à l'intégration de systèmes dynamiques linéaires et non linéaires

5.3.3.1 Mémoire dynamique du second ordre

La mémoire dynamique réalise l'intégration d'une équation différentielle du premier degré. Ce principe peut être étendu à l'intégration d'une équation différentielle d'ordre plus élevé. Considérons par exemple le système dynamique décrit par l'équation

$$\ddot{y} + b\dot{y} + ky = u \tag{5.46}$$

L'intégration de ce système du second ordre peut très simplement se faire par deux étapes d'intégration successives utilisant deux mémoires dynamiques du premier odre fonctionnant en série. Si l'on nomme $f(x,t)$ la carte représentant la variable $y(t)$ et $g(x,t)$ la carte représentant la variable $\dot{y}(t)$, alors l'application de la contrainte de rigidité à ces deux représentation donne le système suivant

$$\begin{aligned} \frac{\delta f}{\delta t}(x,t) &= -\frac{\delta f}{\delta x}(x,t) \cdot \dot{y}(t) \\ \frac{\delta g}{\delta t}(x,t) &= -\frac{\delta g}{\delta x}(x,t) \cdot \ddot{y}(t) \end{aligned} \tag{5.47}$$

les deux cartes placées en série réalisant alors l'intégration double du signal $\ddot{y}(t)$. Si l'on veut intégrer l'équation (5.46) alors il est nécessaire de définir l'opération consistant à produire la sommation de plusieurs mémoires dynamiques. Supposons ainsi que $u(t)$ soit représenté dans une carte $h(x,t)$. L'équation (5.46) impose $\ddot{y} = u - b\dot{y} - ky$. Une première solution consiste à reconstruire le signal $\ddot{y}(t)$ à partir du contenu des trois cartes $h(x,t)$, $g(x,t)$ et $f(x,t)$. Par exemple, si ces trois cartes utilisent un codage par population, alors

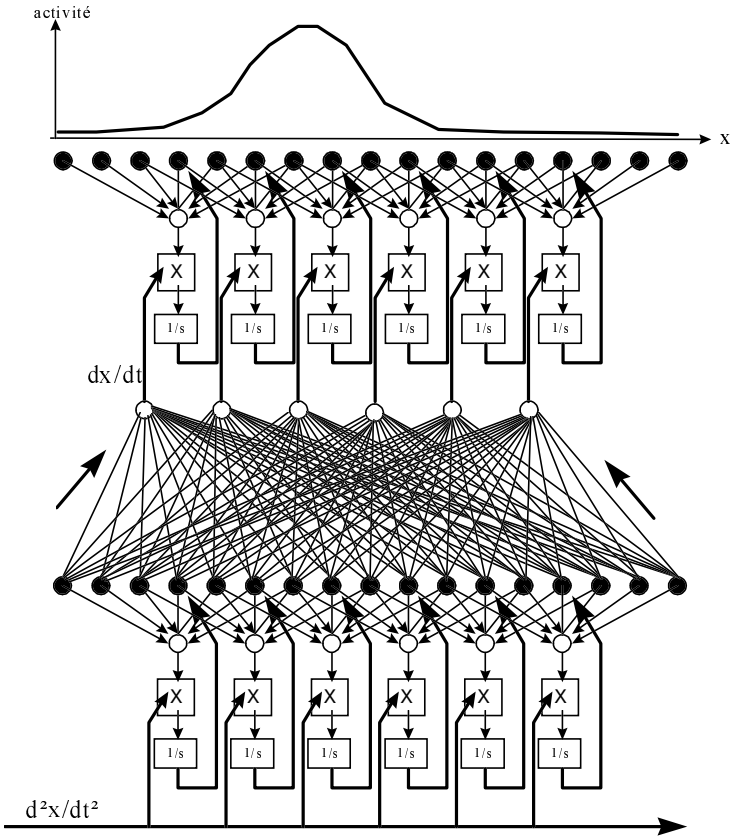

Figure 13. Exemple de réalisation d'un intégrateur du second ordre utilisant deux mémoires dynamiques placées en série. La première mémoire intègre le signal d'accélération. Tous les neurones de sortie de cette première carte sont connectés à chaque neurone d'entrée de la couche intermédiaire qui réalise, via un codage par population, l'estimation de la vitesse de déplacement qui est ensuite intégrée par la seconde mémoire dynamique.

$$\ddot{y}(t) = \int_{\Lambda_h} xh(x,t)dx - b\int_{\Lambda_g} xg(x,t)dx - k\int_{\Lambda_f} xf(x,t)dt$$

où Λ représente le domaine de définition spatial de chacune trois cartes.

Mais on peut également chercher à n'utiliser qu'une seule mémoire dynamique permettant d'intégrer cette équation différentielle, c'est à dire de maintenir une représentation de la variable $y(t)$ dans une carte monodimensionnelle $f(x,t)$ en acceptant comme seule entrée la commande u. Cette équation étant d'ordre 2, il est nécessaire de travailler sur une expression de la différentielle de f prenant en compte le second ordre. La différentielle de f s'écrit alors

$$df(x,t) = \frac{\delta f}{\delta x}\cdot dx + \frac{\delta f}{\delta t}\cdot dt + \frac{1}{2}\left[\frac{\delta^2 f}{\delta x^2}\cdot dx^2 + \frac{\delta f^2}{\delta t^2}\cdot dt^2 + 2\frac{\delta^2 f}{\delta x \delta t}\cdot dx\cdot dt\right]$$

le déplacement de la représentation est donné par

$$dx = \dot{y} \cdot dt + \frac{1}{2}\ddot{y} \cdot dt^2 + o(dt^3)$$

et la différentielle de f s'exprime sous la forme du polynôme en dt suivant (on néglige les termes de degré supérieur à 2)

$$df = dt \cdot \left[\frac{\delta f}{\delta x}\dot{y} + \frac{\delta f}{\delta t}\right] + \frac{dt^2}{2} \cdot \left[\frac{\delta^2 f}{\delta t^2} + \frac{\delta^2 f}{\delta x^2}\dot{y}^2 + 2\frac{\delta^2 f}{\delta x \delta t}\dot{y} + \frac{\delta f}{\delta x}\ddot{y}\right]$$

la contrainte de rigidité imposant $df = 0$, alors nécessairement

$$\begin{aligned} \frac{\delta f}{\delta t} &= -\frac{\delta f}{\delta x}\dot{y} \\ \frac{\delta^2 f}{\delta t^2} &= -\frac{\delta^2 f}{\delta x^2}\dot{y}^2 - 2\frac{\delta^2 f}{\delta x \delta t}\dot{y} - \frac{\delta f}{\delta x}\ddot{y} \end{aligned}$$

L'écriture de ce système peut se «simplifier» en

$$\frac{\delta^2 f}{\delta t^2} = \frac{\frac{\delta^2 f}{\delta x^2}}{\left(\frac{\delta f}{\delta x}\right)^2}\left(\frac{\delta f}{\delta t}\right)^2 - \frac{\delta f}{\delta x}\ddot{y} \tag{5.48}$$

Ce développement peut s'obtenir à partir du système (5.47) en imposant $g(x,t) = \frac{\delta f}{\delta t}$. La dynamique locale des neurones d'une telle «mémoire dynamique» du second ordre sera du type

$$z_i^2 \ddot{n}_i = z_i' \dot{n}_i^2 - z_i^3 \ddot{y}$$

ou z_i et z_i' sont respectivement des estimations locales du gradient spatial et de la courbure de $f(x,t)$ qui peuvent être obtenues à partir de l'activité des neurones voisins. Ce court développement montre que la suppression d'une représentation intermédiaire (ici la carte $g(x,t)$) conduit à une complexification de la fonction de transfert du neurone qui doit alors obéir à une dynamique locale fortement non linéaire. Cette équation décrit en réalité l'équivalent d'un processus physique de diffusion. La construction de réseaux de neurones récurrents régulant la diffusion de l'information dans des représentations de type topographique est une voie prometteuse pouvant bénéficier du savoir-faire théorique des physiciens dans ce domaine [75, Funahashi et Nakamura, 1993] [147, MacGregor, 1987] [266, Yuasa et al., 1997] .

5.3.3.2 Chaînage de mémoires dynamiques

Le principe du codage par population de neurones permet de modéliser des systèmes plus complexes (d'ordre plus élevé) par le chaînage de mémoires dynamiques réalisant des intégrations successives. Ainsi, un intégrateur du second ordre peut être construit en plaçant deux mémoires dynamiques en série: la vitesse de déplacement de la «montagne d'activité» dans la population des neurones de la seconde carte est obtenue par un codage par population de la vitesse dans la première carte (voir figure 13). Le chaînage implique que tous les neurones de sortie d'une mémoire dynamique projettent sur tout les neurones d'entrée de la mémoire dynamique réalisant l'intégration suivante (voir figure 13).

5.3.3.3 Exemples de simulation

On peut obtenir un système dynamique du second ordre en chaînant deux mémoires dynamiques du premier ordre. Nous avons par exemple simulé un filtre du second ordre dont la fonction de transfert est du type $H(s) = \frac{\beta}{(s+\frac{1}{\tau})^2}$ ou τ et β représentent respectivement une constante de temps et le gain du filtre. Le modèle utilisé comporte deux mémoires dynamiques du premier ordre identiques se comportant comme des intégrateurs à fuite de même constante de temps τ (τ=20 ms). Les poids des neurones permettant l'estimation du gradient spatial d'activité dans la carte ne sont pas appris mais calculés afin que ce gradient soit obtenu par un filtre dérivateur sinusoïdal de largeur fixe (le calcul utilise ici les 10 plus proches voisins de chaque neurone). La figure (14) représente la réponse impulsionnelle des deux mémoires dynamiques. Le principe du chaînage fonctionne bien car le déplacement de la représentation dans les deux cartes monodimensionnelles suit la loi attendue.

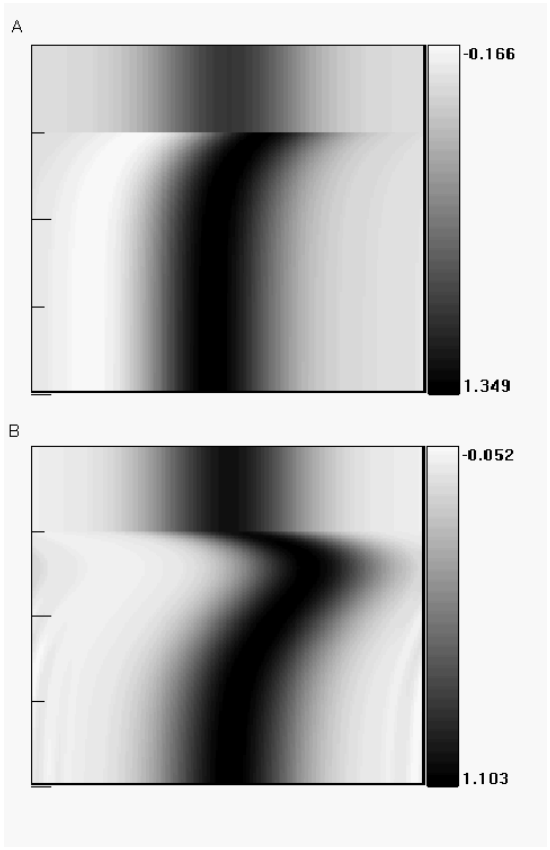

Figure 14. Evolution temporelle de l'activité des neurones de deux mémoires dynamiques (200 neurones chacune) placées en séries se comportant comme des intégrateurs à fuite. L'axe vertical est l'axe temporel et chaque graduation mineure représente 10 ms. L'entrée présentée à la première carte (A) est une impulsion. La carte répond par un déplacement instantané suivit d'un retour au centre selon une décroissance exponentielle. La seconde carte, identique à la première, utilise la valeur codée par la population de neurones de la première carte comme un signal de vitesse et réalise l'intégration de ce signal.

5.4 Discussion du modèle

Nous avons proposé dans ce chapitre une architecture formelle utilisant le principe de codage par population et montré ses capacités à jouer le rôle d'observateur prédictif (équivalent à un modèle dynamique direct) d'un système non linéaire multidimensionnel. Nous avons également montré qu'un tel filtre permettait le contrôle adaptatif d'un système dynamique non linéaire monodimensionnel. Enfin, nous avons réintroduit le modèle de la mémoire dynamique en montrant comment l'on pouvait utiliser le principe de codage par population pour chaîner des mémoires dynamiques et ainsi obtenir des modèles dynamiques internes «autonomes» de systèmes d'ordres supérieurs à l'unité. Si nous nous sommes principalement inspiré de résultats expérimentaux concernant les propriétés des populations neuronales pour proposer ce modèle, il faut noter que celui-ci appartient au domaine de l'approximation de fonctions non linéaires par des réseaux utilisant des fonctions à bases radiales. Ce suport théorique appréciable ainsi que les nombreuses observations expérimentales associées font de ce modèle (et des modèles proches) une hypothèse très plausible pour l'internalisation de la dynamique des segments corporels par le système nerveux central. Cette forte plausibilité théorique sera le thème de cette discussion qui n'oubliera pas cependant de souligner certaines limitations de ce type d'architecture.

D'un point de vue didactique, ce modèle montre bien que le codage par population ne se résume pas à une représentation interne distribuée et robuste de variables dans un référentiel donné mais qu'il permet une simplification de la génération de transformations sensori-motrices non linéaires par la sélection d'une population de «programmes moteurs» adaptés. Ces programmes ne sont pas, comme cela a été proposé dans certains modèles, des représentations de paramètres statiques du mouvement (comme la direction d'un mouvement du bras [79, Georgopoulos et al., 1993] , la direction et l'amplitude d'une saccade [194, Pouget et Sejnowsky, 1997] [236, Sparks et al., 1976]), mais des fonctions sensori-motrices dynamiques. Autrement dit, l'architecture proposée permet la sélection et l'apprentissage de la relation perception-action adaptée au «contexte» sensoriel et moteur codé dans la carte de contexte.

5.4.1 Généralisation du modèle à d'autres types de courbes d'accord

La qualité principale de cette architecture est sa capacité à réaliser l'approximation de fonctions non linéaires en utilisant une seule couche d'unités de calcul. Cette propriété, qui permet l'utilisation de règles d'apprentissage simples et locales, est due au caractère multimodal de la sensibilité des neurones de la carte d'état et à l'introduction d'une opération mathématiques simple qu'est la multiplication (opération absente de l'architecture des perceptrons multicouches canoniques). Les simulations présentées utilisent une base de fonctions à courbes d'accord radiales de type «gaussiennes». Il faut cependant souligner que d'autres types de fonction auraient pu être utilisées comme par exemple des fonctions sigmoïdes ou encore des «chapeaux mexicains». En effet, les fonctions gaussiennes de la carte de contexte sont utilisées comme une base de fonctions, c'est à

dire un ensemble de fonctions à partir duquel la fonction à recontruire peut être approximée. C'est grâce à un principe similaire que des fonctions réelles de $\Re$ dans $\Re$ peuvent être reconstruites à partir d'une combinaison linéaire de sinus et de cosinus dont l'expression des coefficients donne la transformée de Fourier de la fonction considérée. Dans notre modèle, les fonctions d'accord de la carte de contexte jouent le même rôle que les fonction sinus et cosinus de la transformée de Fourier. D'après les propriétés des approximateurs utilisant des bases de fonctions non linéaires, il est facile de montrer

- que ce modèle peut utiliser d'autres type de fonctions d'accord que les fonctions gaussiennes,
- que ce modèle permet de réaliser, par la combinaison «en cascade» de cartes d'états sensorielles ou motrices monomodales, des transformations non linéaires indépendantes et en particulier des changements de référentiels.

Une propriété essentielle de ce principe d'approximation est le fait que le produit de deux bases de fonctions est une base de fonction (pour la démonstration, voir par exemple [194, Pouget et Sejnowsky, 1997]). Pour illustrer ce fait, prenons l'exemple d'une fonction f de $\Re^2$ dans $\Re$ approximée par l'expression (voir (5.30))

$$\widetilde{f}(x,y) = \sum_{i=1}^{i=n} \sum_{j=1}^{j=n} a_{i,j} \cdot e^{-\frac{(x-x_i)^2+(y-y_j)^2}{2\sigma^2}} \tag{5.49}$$

La carte de contexte associée à cet approximateur est bidimensionnelle et comporte des fonctions d'accord gaussiennes. Les propriétés de la fonction exponentielle nous permettent de réécrire cette équation sous la forme:

$$\widetilde{f}(x,y) = \sum_{i=1}^{i=n} \sum_{j=1}^{j=n} a_i \cdot e^{-\frac{(x-x_i)^2}{2\sigma^2}} \cdot e^{\frac{-(y-y_j)^2}{2\sigma^2}} = \sum_{i=1}^{i=n} \sum_{j=1}^{j=n} a_{i,j} \cdot B_i(x) \cdot B_j(y) \tag{5.50}$$

Autrement dit, un approximateur utilisant une carte de contexte bidimensionnelle peut être obtenu par la combinaison, terme à terme, de deux cartes d'états monodimensionnelles indépendantes utilisant deux bases de fonctions $B_i, i \in [1,n]$ et $B_j, j \in [1,n]$ différentes. D'un point de vue pratique: le calcul des n^2 termes exponentiels de l'équation (5.49) peut être limité au calcul des $2n$ termes exponentiels de l'équation (5.50) ce qui réduit de façon importante le temps de calcul de $\widetilde{f}$. Ce résultat peut être étendu à des fonctions de dimension plus élevée, mais également à des bases de fonctions de types différents ce qui participe de la pertinence biologique du modèle. Pour soutenir cette affirmation, nous nous inspirerons du modèle des «gain fields» proposé dans le cadre de l'étude du contrôle des mouvements des yeux [6, Andersen et al., 1985] . Ce modèle est

issu de l'observation d'une modulation, par un signal corrélé à la position des yeux, de l'amplitude de la réponse des neurones du cortex pariétal postérieur à la présence d'un stimulus dans leur champ récepteur (champ rétinotopique fixe). Chaque champ récepteur de ces neurones est accordé sur une position particulière de la rétine pour laquelle la réponse de la cellule est maximale. Ce maximum varie quasi-linéairement avec la position de l'oeil dans l'orbite. La réponse visuelle de chaque cellule de cette région du cortex peut être modélisée par le produit d'une gaussienne (représentant la courbe d'accord du champ récepteur) et d'une sigmoïde (représentant la courbe d'accord de la décharge fonction de la position de l'oeil). Si l'on se limite à un modèle monodimensionnel, la fréquence de décharge d'une cellule (désignée par $n_{i,j}$) dépend à la fois de la position du stimulus sur la rétine et de la position de l'oeil selon la relation (voir [194, Pouget et Sejnowsky, 1997])

$$n_{i,j} = a_{i,j} \cdot e^{\frac{-(rx-rx_i)^2}{2\sigma^2}} \cdot \frac{1}{1+e^{\frac{-(ex-ex_j)}{T}}} = a_{i,j} \cdot B_i^G(rx) \cdot B_j^S(ex)$$

où

- rx et ex représentent respectivement la position du stimulus lumineux sur la rétine et la position de l'oeil dans l'orbite (exprimés en degrés),
- rx_i et ex_j représentent respectivement le centre du champ récepteur rétinotopique (la zone de réponse maximale) et le seuil de la fonction sigmoïde,
- σ et T désignent la largeur du champ récepteur et la pente de la sigmoïde,
- $a_{i,j}$ est l'amplitude de l'activité «de base» de la cellule $n_{i,j}$ (c'est à dire pour un stimulus visuel hors du champ récepteur et une position de l'oeil très inférieure au seuil),
- les indices G et S dans $B_i^G(rx)$ et $B_j^S(ex)$ désignent le type de fonction canonique (gaussienne ou sigmoïde) à partir desquelles les bases de fonctions sont construites.

Les propriétés individuelles de ces neurones donnent à leur population la capacité de générer des fonctions linéaires ou non linéaires de la position de l'oeil dans l'orbite et de la position d'un stimulus lumineux sur la rétine. Il leur est en particulier possible de coder les coordonnées du stimulus visuel dans un référentiel craniotopique. Il suffit pour cela d'adapter les coefficients $a_{i,j}$ tels que la fonction $\widetilde{f}$ donnée par

$$\widetilde{f}(rx,ex) = \sum_{i=1}^{i=n}\sum_{j=1}^{j=n} n_{i,j} = \sum_{i=1}^{i=n}\sum_{j=1}^{j=n} a_{i,j} \cdot B_i^G(rx) \cdot B_j^S(ex) \tag{5.51}$$

approxime la fonction $f(rx,ex) = rx + ex$. Si ce principe est ici appliqué à une transformation statique d'informations sensorielles et motrices, il peut être étendu à l'approximation de fonctions dynamiques comme nous l'avons montré précédemment. La décomposition de la carte de contexte en une combinaison de

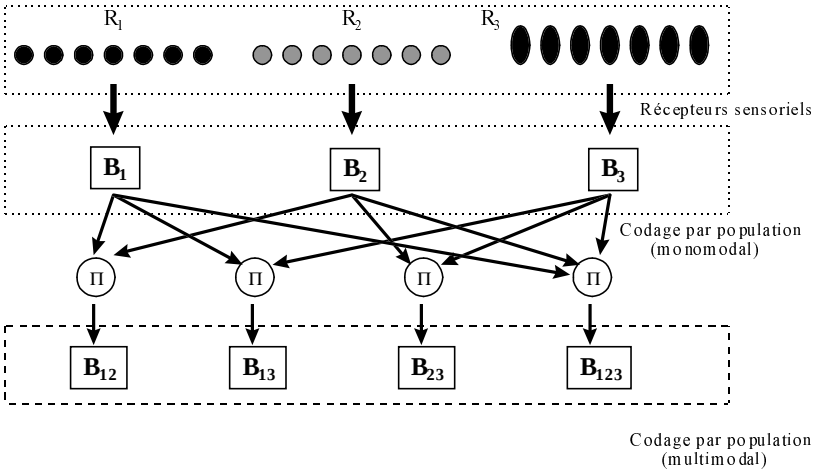

Figure 15. Illustration de l'utilisation hypothétique de combinaisons de bases de fonctions par le système nerveux central pour approximer des transformations sensorimotrices non linéaires. Les populations de neurones B_1, B_2 et B_3 réalisent un codage par population monomodal (à partir des signaux délivrés par une population de récepteurs sensoriels spécifiques). Les populations B_{12}, B_{13}, B_{23} et B_{123} effectuent un codage multimodal: les activités des neurones de ces populations sont obtenues par la multiplication terme à terme de toutes les activités des neurones des populations à codage monomodal correspondantes. Les fonctions remplies par les neurones de chacune des populations à codage multimodal constituent alors elles aussi des bases de fonctions qui peuvent ainsi être utilisées pour la réalisation de transformations sensorimotrices non linéaires indépendantes. Par exemple, la population B_{23} peut être impliquée dans la génération d'une commande motrice fonction des activités des populations de récepteurs sensoriels R_2 et R_3.

codages indépendants à l'avantage de permettre à une population de neurones sensibles à une variable signifiante particulière d'être impliquée dans le calcul de plusieurs transformations sensorimotrices indépendantes. Cette approche peut ainsi conduire à la proposition d'un modèle global hypothétique de la façon dont le système nerveux peut tirer partie de nombreuses populations de neurones ou de récepteurs sensoriels, présentant des courbes de sensibilité de types différents, pour réaliser des transformations indépendantes (voir figure 15).

5.4.2 Amélioration du modèle par utilisation d'une multirésolution

Nous avons vu dans ce chapitre, que la carte de contexte réalise en fait un filtrage par convolution (en l'occurrence passe-bas, voir équation (5.32) par exemple) des coefficients de toutes les approximations linéaires locales. Le choix du rayon des gaussiennes σ, en partie arbitraire dans les simulations présentées, conditionne les fréquences de coupure de ce filtre et donc la qualité de l'approximation. La résolution de la carte de contexte (et donc le nombre de fonctions radiales utilisées) est liée au choix de leurs rayons: si l'on choisit un rayon faible afin de permettre au réseau d'approximer les composantes fréquentielles élevées de la fonction à approximer, alors la distance séparant les centres des gaussiennes doit être réduite. Le coût à payer est donc finalement une augmentation du nombre de gaussiennes dans cette carte. Or, le défaut principal de cette ap-

proche est l'explosion combinatoire du nombre de fonctions avec l'augmentation du nombre de variables prises en compte dans la carte de contexte. Si l'on utilise pour chaque dimension le méme nombre n de fonctions de base (comme dans l'équation 5.51) alors, le nombre total de fonctions (ou de combinaisons de ces fonctions) à considérer est n^d (où d est la dimension de la carte de contexte).

Mais le pavage régulier de la carte de contexte par les fonctions gaussiennes est un choix arbitraire non optimal. En effet, les composantes fréquentielles de la fonction à approximer peuvent varier selon les régions de la carte de contexte. On peut donc penser limiter le problème de l'explosion combinatoire du nombre de gaussiennes en faisant varier la densité et le rayon des champs récepteurs en fonction des composantes fréquentielles spatiales locales de la fonction a approximer [173, Mussa-Ivaldi, 1992] [174, Mussa-Ivaldi et Giszter, 1992] . Ainsi, lorsque la fonction à approximer «varie peu» dans une région de la carte de contexte, il est possible de réduire la densité des gaussiennes et d'augmenter leurs rayons. A l'opposé, si cette fonction présente localement de fortes variations, la taille des champs récepteurs doit être réduite et leur densité conjointement augmentée. Mais comme ces fluctuations des composantes spectrales de la fonction à approximer sont a priori inconnues, il est nécessaire de faire varier la résolution de la carte de contexte lors de l'apprentissage. Introduire un tel processus, c'est considérer que les rayons σ et les centres des gaussiennes sont des paramètres eux aussi sujets à une adaptation. Trois dynamiques se superposent alors:

- la dynamique du contrôle ou de la prédiction (évolution temporelle des variables d'entrée et de sortie du filtre),
- la dynamique de l'apprentissage des coefficients des gaussiennes (évolution temporelle des coefficients),
- la dynamique d'une résolution adaptative (évolution temporelle des centres et des rayons des champs récepteurs de la carte).

Si l'on dispose d'une première loi d'apprentissage concernant l'adaptation des coefficients des gaussiennes (équation 5.38), il est également nécessaire de trouver une loi permettant de faire varier la résolution des gaussiennes de façon pertinente.

Le parti pris qui nous semble le meilleur propose d'obtenir une estimation locale des composantes fréquentielles de la fonction codée par le filtre à partir des coefficients des applications linéaires locales liées aux gaussiennes [38, Cannon et Slotine, 1995] . En effet, supposons que la carte de contexte soit de type multirésolution et donc qu'elle comporte différentes populations de gaussiennes différant par leur rayon et leur espacement. Dans ce type de représentation (de type gamme d'ondelettes) les gaussiennes dont le rayon est important ont une densité plus faible que celles dont le rayon est plus petit (voir schéma 16).

Les auteurs [38, Cannon et Slotine, 1995] proposent un algorithme permettant d'ajouter ou de retirer de la carte des fonctions à base radiales à chaque pas de calcul. Cet algorithme gère une population de fonctions

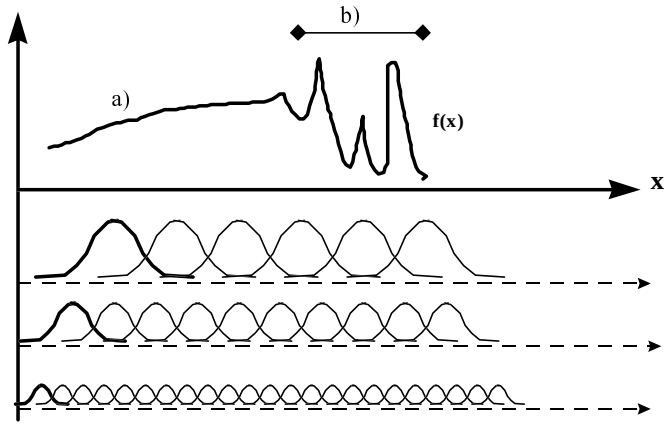

Figure 16. Approche multirésolution: la carte de contexte, ici monodimensionnelle, est échantillonnée par trois populations de fonctions gaussiennes différant par leur densité et la largeur de leur rayon. L'estimation de la fonction à approximer f(x) en un point x de l'axe horizontal est obtenue par une somme des coefficients associés à chaque gaussienne pondérés par l'activité de la gaussienne correspondante en ce point. Dans la région a) de la carte pédominent les basses fréquences. On s'attend alors à ce que les coefficients des gaussiennes à large rayon soient plus important en valeur absolue que les coefficients des gaussiennes à faible rayon. Dans la région b) comportant plus de hautes fréquences, le résultat est inverse. On peut se baser sur une comparaison de l'amplitude des coefficients associés aux gaussiennes pour supprimer de la carte de contexte les gaussiennes dont les coefficients sont localement faibles.

à bases radiales (de type «chapeau mexicain») dans laquelle la probabilité de survie de chaque individu est définie par un indice de «fitness» correspondant à la norme des coefficients associés à la fonction à base radiale considérée. Avec les notations utilisées dans ce travail, l'estimateur $\Psi(x) = \hat{f}(x)$ est défini par

$$\widehat{f}(x) = \sum_{j \in J_t^+ \sqcup J_t^-} \hat{c}_j(t) B_j^M(x) + \hat{c}(t)$$

où les ensembles J_t^+ et J_t^- représentent deux populations de fonctions évoluant lors de l'apprentissage avec

$$\begin{aligned} J_t^- &= \{j; |\hat{c}_j(t)| \leq \lambda \text{ et } \frac{d}{dt}\hat{c}_j(t)^2 \leq 0\} \\ J_t^+ &= \{j; |\hat{c}_j(t)| > \lambda \text{ ou } \frac{d}{dt}\hat{c}_j(t)^2 > 0\} \end{aligned}$$

Le paramètre λ joue le rôle de seuil sur la valeur de la fitness $|\hat{c}_j(t)|$. Toutes les fonctions utilisées sont issues d'une population générative J_t^0 contenant des fonctions à bases radiales qui diffèrent par la position de leur centre et par la taille de leur champ récepteur. La population J_t^+ représente les fonctions qui satisfont au critère choisi. La population J_t^- représente l'ensemble des fonctions ne satisfaisant pas au critère choisi et qui peuvent donc être supprimées de l'approximateur. A chaque itération, une fonction est retirée de J_t^- si cet ensemble n'est pas vide et une fonction tirée de J_t^0 est introduite à sa place selon une heuristique que nous ne décrirons pas ici. Ce processus n'est pas à proprement parler supervisé et est assez proche, dans son principe d'une optimisation par algorithme génétique. Il faut cependant souligner la grande rigueur dont font montre les auteurs dans leur choix des paramètres du modèle et leur constant souci de justifier leur choix par une démarche analytique poussée.

5.4.3 Limitations de cette approche et pertinence biologique

Le principal inconvénient de ce type de filtre adaptatif est l'augmentation exponentielle du nombre de neurones dans la carte de contexte avec la dimension de la fonction à approximer. Ce problème peut être résolu en partie par l'utilisation d'un processus adaptatif permettant de faire varier localement la résolution de la carte de contexte. Mais il est d'autre part difficile de fixer a priori l'étendue du domaine de définition de la carte de contexte. En effet, cette carte perd totalement son utilité si les neurones la composant sont sensibles à des configurations sensorielles ou motrices peu fréquentes. La gestion de l'étendue de la sensibilité de cette carte, comme la gestion de sa résolution, doit se faire de façon adaptative[25].

[25] Nous avons pour cela utilisé une méthode très simple et efficace dans les simulations présentées. Le domaine de définition de la carte de contexte évolue dans le temps avec les extrema des variables de contexte. Rien ne montre cependant que cette procédure ne nuit pas à la stabilité du système dans le cas général.

Malgré ces faiblesses, ce modèle est cependant très séduisant du point de vue de la neurobiologie pour plusieurs raisons

- il autorise un apprentissage supervisé sans traitement complexe du signal d'erreur (une seule couche de traitement).
- il fait appel à des unités formelles fonctionnant sur un principe similaire au codage par population proposé par les neurophysiologistes.
- Plusieurs filtres adaptatifs de ce type peuvent travailler indépendamment en combinant les informations fournies par plusieurs ensembles d'unités opérant des codage par population monomodaux.
- Sous la forme «mémoire dynamique», ce modèle permet la génération de modèles internes autonomes de processus dynamiques,

Si l'on imagine que le système nerveux central utilise ce principe, les différentes populations de récepteurs sensoriels sont une source de fonctions non linéaires dans laquelle il lui est possible de puiser afin de générer, de façon éventuellement plastique, de multiples cartes de contexte. Cependant, une des caractéristiques de cette architecture peut naturellement intriguer. Que penser d'un modèle nécessitant la réalisation du produit de l'activité de deux unités neuronales? Un grand nombre de neurophysiologistes ont été séduit par les modèles du type «perceptron multicouche» parce qu'ils mobilisent des unités de calcul semilinéaires dont les activités sont simplement additionnées. Ce succès est sans doute en partie dû au fait que les perceptrons ne remettent pas en question le stéréotype du «neurone sommateur de ses entrées». Ce modèle canonique du neurone est sans doute plus bousculé encore dans le chapitre suivant, qui propose un modèle de filtre adaptatif fonctionnant sur un principe tout à fait différent. Nous reviendrons cependant sur la plausibilité biologique de ce modèle (en particulier sur le problème de la multiplication des activités neuronales) et proposerons avec prudence d'identifier les structures neuronales susceptibles de fonctionner sur ce principe dans la conclusion générale.

Chapitre 6
Les champs mnémoniques

6.1 Introduction

Comme nous l'avons vu dans le chapitre précédent, des algorithmes d'apprentissage simples peuvent être utilisés pour construire des modèles internes prédictifs ou bien pour apprendre la dynamique inverse de systèmes non linéaires complexes. Le type de filtre adaptatif non-linéaire Ψ le plus utilisé, le modèle du perceptron multicouche, convertit une fonction complexe (d'ordre élevé) en fonction du premier ordre par rapport à la sortie d'une couche «cachée» constituant une représentation intermédiaire (représentation "interne"). Mais l'apprentissage de cette première étape du traitement de l'information peut être problématique, en particulier lorsque le nombre de signaux d'entrée est important (lorsque l'ordre du problème posé est élevé). Dans ce paragraphe, nous proposons une approche différente, prenant directement en compte l'ordre du problème et évitant ainsi l'étape problèmatique de la construction d'une étape intermédiaire de traitement de l'information. L'approche proposée diffère également de celle adoptée dans le paragraphe précédent, car nous ne cherchons pas ici à linéariser localement une fonction non linéaire, mais utilisons des unités de calcul, les *mnémons*, qui sont des opérateurs non-linéaires par construction. Afin de soutenir notre argumentation, nous avons également comparé les performances des champs mnémoniques à celles des perceptrons multicouches pour quelques problèmes génériques. Cette comparaison est faite en termes de mémoire requise, de vitesse de calcul et d'apprentissage ainsi qu'en terme de précision. La dernière partie de cette étude est consacrée à la résolution d'un problème intéressant plus particulièrement le contrôle de la dynamique des mouvements: le contrôle d'un modèle de bras articulé à deux degrés de liberté. Ce dernier exemple démontre la capacité des champs mnémoniques à approximer des fonctions dynamiques non-linéaires d'ordre élevé. Il nous permettra également de discuter en conclusion la plausibilité biologique de cette approche nouvelle.

6.2 Mnémons et champs mnémoniques

6.2.1 Le mnémon

Contrairement au neurone canonique décrit par l'équation (5.26) et aux fonctions à bases radiales décrites dans le chapitre précédent, un mnémon ne peut traiter que des signaux binaires. Dans sa forme originelle, le mnémon n'est rien d'autre qu'une *table de mémoire*, capable d'associer une réponse à une stimulation exprimée sous la forme d'un vecteur, dont les composantes ne peuvent prendre que la valeur 1 ou la valeur 0. Un mnémon est entièrement décrit par la donnée de cette table et de son *champ récepteur* qui spécifie quels sont les bits d'information qui seront traités par ce mnémon parmi les informations disponibles. Plaçons nous d'emblée dans le cadre de l'approximation d'une fonction booléenne f dont l'ensemble de départ est $\Omega_n = [0,1]^n$ à valeurs dans $[0,1]$. L'ordre maximal de la fonction f est donné directement par la dimension n de l'espace de départ. Une table de mémoire comprenant 2^n cases contenant chacune un unique bit d'information peut décrire totalement la fonction f. Un mnémon ne traite qu'une partie de l'espace Ω_n des entrées possibles et son champ récepteur est défini par la donnée d'un sous espace Φ_k de dimension k de Ω_n. Prenons par exemple le cas particulier $n = 6$ et $k = 3$. Le champ récepteur du mnémon est le sous ensemble de Ω_6 comportant les vecteurs dont les trois composantes correspondent aux second, troisième et sixième bits du vecteur E des entrées:

$$E = [e_1, e_2, e_3, e_4, e_5, e_6] \rightarrow V_{k=3} = [e_2, e_3, e_6] = [\varepsilon_1, \varepsilon_2, \varepsilon_3]$$

Définition 1 : Le champ récepteur d'un mnémon est un sous espace Φ_k de Ω_n.

Le mnémon associe au vecteur V courant une adresse (par exemple $a(V) = e_2 \cdot 2^0 + e_3 \cdot 2^1 + e_6 \cdot 2^2$), et produit une sortie $S[a]$ choisie dans une table de mémoire de taille 8 bits. Si, par exemple $E = [0,1,0,0,1,1]$, alors $V = [1,0,1]$, $a(V) = 5$ et la sortie S correspondra au bit mémorisé dans la sixième case de la table du mnémon.

Définition 2 : Un mnémon d'ordre k est suite de 2^k valeurs (binaires ou réelles) ordonnées. L'entrée du mnémon V est donnée par l'état de son champ récepteur et sa sortie S est obtenue par une bijection (un code) de Φ_k sur $S[j]$, $j \in [1, 2^k]$.

Ainsi, un mnémon d'ordre k utilise une mémoire de 2^k bits. Cette valeur peut être comparée au $32 \cdot (k+1)$ bits nécessaires pour définir les poids et le seuil d'un neurone formel classique (en virgule flottante, simple précision) traitant aussi k signaux d'entrée. Une unité mnémonique consomme ainsi moins de mémoire qu'un neurone formel tant que le nombre d'entrées est inférieur ou égal à 8.

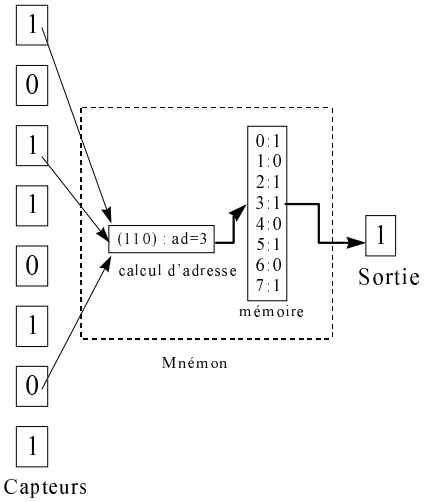

Figure 17. Description schématique du fonctionnement d'une unité mnémonique. Le mnémon est ici d'ordre 3 et est porteur d'une mémoire de $2^3 = 8$ bits. La sortie est le contenu de l'adresse obtenue à partir des signaux appartenant au champ récepteur de l'unité (limité aux premier, troisième et septième capteurs).

6.2.2 Les champs mnémoniques

En général, la dimension de l'espace de départ est trop grande pour utiliser des mnémons dont le champ récepteur prend en compte toutes les entrées. Par exemple, identifier une fonction booléenne dont l'entrée est donnée par une image noir et blanc de 16 pixels de coté nécessite une table de mémoire contenant plus de bits que de nombre de particules dans l'univers ! L'ordre des mnémons est donc limité par un phénomène d'explosion combinatoire. Mais on peut s'attendre à ce que dans la plupart des cas, il suffise d'utiliser une partie seulement des entrées pour obtenir une bonne approximation de la réponse désirée. De plus, on peut améliorer encore significativement cette approximation en utilisant simultanément plusieurs échantillons indépendants des signaux d'entrée. Ceci peut être réalisé en utilisant simultanément plusieurs mnémons d'ordre limité dont les champs récepteurs sont différents. Un tel ensemble de mnémon définit un *champ mnémonique* :

Définition 3 : Un champ mnémonique est un ensemble de m mnémons dont les champs récepteurs sont différents. La sortie Ψ du champ mnémonique est une somme[26] (éventuellement pondérée) des sorties S_i, $i \in [1, m]$ des mnémons qui le composent:

$$\Psi = \sum_{i=1}^{i=m} S_i[a(V_i)] \tag{6.52}$$

L'utilisation de plusieurs mnémons fonctionnant en parallèle permet de contourner le problème de capacité mémoire issu de l'explosion combinatoire mais également de doter le champ mnémonique d'une capacité de généralisation.

6.2.3 Règles d'apprentissage

D'après la définition 3, la sortie du champ mnémonique peut être directement reliée à la sortie de chacun des mnémons. Par conséquent, dans le cadre d'un apprentissage supervisé, il n'y a aucune rétropropagation de l'erreur à appliquer. Si l'on cherche à minimiser un critère quadratique de l'erreur (sur un ensemble de l exemples) tel que

$$J = \sum_{j=1}^{j=l} \left(\Psi_j - \widehat{\Psi}_j\right)^2 = \sum_{j=1}^{j=l} \left(\sum_{i=1}^{i=m} S_i\left[a(V_i^j)\right] - \widehat{\Psi}_j\right)^2 = \sum_{j=1}^{j=l} \Delta_j^2$$

avec $\widehat{\Psi}_j$ réponse attendue du champ mnémonique, alors il est possible d'appliquer une règle d'apprentissage du type descente de gradient (Δ-rule):

[26] La sortie du champ mnémonique peut éventuellement être obtenue à partir des sorties des mnémons par une fonction différente d'une simple somme (processus du type «winner-take-all» ou encore par un vote ou une fonction seuillée de la somme). Les méthodes proposées dans la suite de ce chapitre peuvent facilement être adaptées à d'autres types de fonctions de «prise de décision».

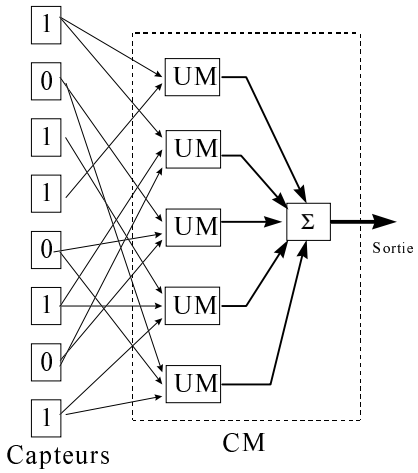

Figure 18. Illustration schématique de la définition d'un champ mnémonique. Un champ mnémonique est composé de plusieurs mnémons (UM) d'ordre limité fonctionnant en parallèle. La sortie globale du champ mnémonique (CM) est simplement la somme des sorties de toutes les unités mnémoniques.

$$S_i(t+1) = S_i(t) - \alpha \cdot \Delta_j \cdot S_i(t) \tag{6.53}$$

où t est le numéro de l'itération et α le pas du gradient. Mais d'autres algorithmes d'apprentissage, supervisés ou non peuvent être utilisés de manière similaire.

L'efficacité d'un mnémon dépend également du champ récepteur qui lui est associé. Il est ainsi important que ce champ prenne en compte des signaux qui sont pertinents du point de vue de la sortie désirée. Dans le cas contraire, le mnémon n'apprendra rien d'autre que la moyenne de la sortie désirée calculée sur tous les exemples qui lui sont présentés. On peut penser utiliser également une procédure d'apprentissage permettant d'adapter le champ récepteur de façon à ce qu'il sélectionne les signaux intéressants parmis les informations fournies. Malheureusement il est a priori impossible de savoir comment modifier ces champs récepteurs à partir de la base d'exemple. Dans cette étude, nous avons choisi

- soit un tirage aléatoire des champs récepteurs des mnémons, laissés inchangés durant toute la simulation,
- soit une procédure d'optimisation non supervisée des champs fonction d'un score attribué à chaque mnémon. Le score choisi est ici le nombre de réponses correctes du mnémon. Les mnémons qui ont les plus faibles performances sont éliminés et remplacés par de nouvelles unités dont le champ récepteur est tiré au hasard.

6.2.4 Approximation de fonctions continues: les mnémons stochastiques

Les champs mnémoniques, tels qu'ils ont été présentés jusqu'ici, ne peuvent être utilisés que pour approximer des fonctions définies de $[0,1]^n$ dans $[0,1]$ (fonction booléenne) ou de $[0,1]^n$ dans $\Re$. Comment utiliser un champ mnémonique pour approximer une fonction de de $\Re^n$ dans $\Re$? La méthode la plus simple, en tout cas du point de vue de l'implémentation, serait d'utiliser comme entrée le codage binaire des variables de l'espace de départ. En particulier pour des raisons de plausibilité biologique, nous proposons un modèle de *mnémon stochastique*, permettant l'approximation de fonctions f définies de $\Re^n$ dans $\Re$. Les entrées d'un mnémon stochastique ε_1, ε_2, ..., ε_k obéissent à une loi de probabilité fonction des variables continues dont f dépend. Ces entrées peuvent être considérées comme la sortie de k capteurs stochastiques indépendants.

Définition 4: un capteur stochastique est une variable aléatoire booléenne dont l'état r dépend d'une variable déterministe x selon la loi de probabilité conditionnelle suivante

$$\begin{aligned} p(X_{/x}) &= p(r = 1_{/x}) = p(x) \\ p(\overline{X}_{/x}) &= 1 - p(x) \end{aligned}$$

$p(x)$ étant une fonction de $\Re$ dans [0..1].

Définition 5: un mnémon stochastique est un mnémon dont les k entrées sont les sorties de k capteurs

stochastiques $r_j(x)$, $j \in [1,k]$ indépendants. La sortie d'un mnémon stochastiques est une probabilité, c'est à dire la probabilité conditionnelle que le mnémon soit actif pour la configuration d'entrée courante du champ récepteur.

Considérons le cas d'une fonction $y = f(x)$ de $\Re$ dans $\Re$. La stratégie proposée consiste à choisir k capteur stochastiques, m mnémons dont l'entrée est la sortie des capteurs stochastiques, et à appliquer une règle d'apprentissage du type de celle décrite par l'équation (6.53). Il est bien sûr inutile dans le cadre de ce problème simple, de considérer que la sortie des mnémons est une probabilité. On utilisera plutôt des tables de valeurs réelles correspondant directement aux valeurs estimées de $f(x)$ fournies par les mnémons. Il est cependant judicieux de déterminer le type de capteur adapté à la résolution d'un problème donné.

6.3 Quelques propriétés attendues des champs mnémoniques

6.3.1 Une justification de l'intérêt des champs mnémoniques

Dans le cas général, on a bien évidemment toujours avantage à utiliser un seul mnémon d'ordre maximal. L'introduction des champs mnémoniques permet de tenter d'approximer une fonction non linéaire avec plusieurs mnémons d'ordres raisonnables. Mais encore faut-il montrer que l'augmentation du nombre de mnémons dans un champ mnémonique permet d'accroître la qualité de l'approximation. Nous précisons dans les lignes qui suivent, pour des mnémons déterministes, les conditions pour lesquelles l'augmentation du nombre de mnémons permet d'améliorer la qualité de l'approximation. Ce développement permet également d'obtenir une règle d'apprentissage utilisable dans un algorithme ou un champ mnémonique est construit de façon itérative.

Soit f une fonction de $[0,1]^m$ dans $\Re$, que l'on essaye d'approximer avec un champ mnémonique $CM^{(n-1)}$ comportant $(n-1)$ mnémons d'ordre k ($k < m$). Nous supposons disposer d'un ensemble Ω de n_e exemples sur lesquels l'erreur quadratique d'approximation est calculée. La sortie du champ mnémonique est simplement donnée par la somme des sorties de tous les mnémons. On peut définir l'erreur réalisée par le champ mnémonique comme étant

$$J_{n-1} = \frac{1}{2} \sum_{j=1}^{j=n_e} \left(f(j) - \sum_{i=1}^{i=n-1} f_i(j) \right)^2 = J_{n-1} = \frac{1}{2} \sum_{j=1}^{j=n_e} \left(\Delta_j^{n-1} \right)^2$$

où

- n_e désigne le nombre d'exemples de la base de test,
- $f(j)$ la valeur de la fonction f pour le $j^{ième}$ exemple,
- $f_i(j)$ la valeur en sortie du $i^{ème}$ mnémon pour le $j^{ème}$ exemple.

Si l'on rajoute un $n^{ième}$ mnémon au champ mnémonique, l'erreur réalisée par celui-ci sera

$$J_n = \frac{1}{2}\sum_{j=1}^{j=n_e}\left(f(j) - \sum_{i=1}^{i=n} f_i(j)\right)^2 = \frac{1}{2}\sum_{j=1}^{j=n_e}\left(\Delta_j^{n-1} - f_n(j)\right)^2 \tag{6.54}$$

Il nous faut donc préciser à quelle condition l'introduction de ce nouveau mnémon permet de diminuer l'erreur réalisée par le champ mnémonique, c'est à dire à quelle condition $J_n < J_{n-1}$. Les mnémons d'ordre k ont une table de mémoire comportant 2^k valeurs réelles. Pour chacun des mnémons, il est possible de partitionner l'ensemble des exemples présentés en 2^k sous ensembles définis comme suit :

Définition 6: on nommera Ω_p^i, $p \in \left[1, 2^k\right]$, l'ensemble (éventuellement nul) des exemples «concernés par l'adresse p du $i^{ème}$ mnémon», c'est à dire l'ensemble des exemples dont la présentation aboutit à la sélection du contenu $S_i(p)$ de la $p^{ème}$ case mémoire comme sortie du $i^{ème}$ mnémon.

Pour un mnémon particulier, les Ω_p^i, $p \in \left[1, 2^k\right]$ forment une partition en sous-ensembles disjoints de l'ensemble Ω des exemples, autrement dit $\forall i \in [1, n], \bigcup_p \Omega_p^i = \Omega$ et $\forall s \in \left[1, 2^k\right], \forall q \in \left[1, 2^k\right]$ avec $q \neq s$ alors $\Omega_s^i \bigcap \Omega_q^i = \emptyset$. Ces sous-ensembles sont disjoints car, pour les mnémons déterministes, la présentation d'un même exemple ne peut aboutir à la sélection de cases mémoires différentes.

Définition 7: on nommera Γ_p^i «contribution de l'adresse p du $i^{ème}$ mnémon à l'erreur» la quantité définie par $\Gamma_p^i = \sum_{j \in \Omega_p^i}\left[\Delta_j^{n-1} - S_i(p)\right]^2$.

$S_i(p)$ désigne comme dans (6.52) le contenu de la $p^{ième}$ case mémoire du $i^{ème}$ mnémon. Il est bien évident que l'on a

$$J_n = \frac{1}{2}\sum_{p=1}^{p=2^k}\Gamma_p^i, \forall i \in [1, n] \tag{6.55}$$

et que

$$J_{n-1} = \frac{1}{2}\sum_{p=1}^{p=2^k}\sum_{j \in \Omega_p^i}\left[\Delta_j^{n-1}\right]^2, \forall i \in [1, n] \tag{6.56}$$

Ce changement de notations nous permet en particulier d'écrire pour le $n^{ième}$ mnémon rajouté à $CM^{(n-1)}$:

$$\frac{\partial J_n}{\partial S_n(p)} = \sum_{j \in \Omega_p^n}\left[\Delta_j^{n-1} - S_n(p)\right]$$

Pour tout p tel que $card(\Omega_p^n) \neq 0$, La valeur optimale $\tilde{S}_n(p)$ de la $p^{ième}$ case mémoire du contenu du $n^{ième}$ mnémon est celle qui donne $\frac{\partial J_n}{\partial S_n(p)} = 0$ soit

$$\tilde{S}_n(p) = \frac{\sum\limits_{j \in \Omega_p^n} \Delta_j^{n-1}}{card(\Omega_p^n)} \tag{6.57}$$

qui est «la valeur moyenne de l'erreur réalisée par $CM^{(n-1)}$ sur l'ensemble des exemples concernés par l'adresse p du $n^{ième}$ mnémon». Notons bien que l'équation (6.57) nous donne une règle d'apprentissage optimale pour la construction d'un champ mnémonique par ajout itératif de mnémons supplémentaires. Quel est alors le gain obtenu sur l'erreur si l'on ajoute ce $n^{ième}$ mnémon en obéissant à la règle (6.57) ?

On peut écrire

$$\sum_{j \in \Omega_p^n} \left[\Delta_j^{n-1} - S_n(p)\right]^2 = \sum_{j \in \Omega_p^n} \left[\left(\Delta_j^{n-1}\right)^2 - \tilde{S}_n(p)^2 + 2\widetilde{S}_n(p)\left(\tilde{S}_n(p) - \Delta_j^{n-1}\right)\right]$$

D'après (6.55),

$$J_n = \frac{1}{2}\sum_{p=1}^{p=2^k}\sum_{j \in \Omega_p^n}\left(\Delta_j^{n-1}\right)^2 - \frac{1}{2}\sum_{p=1}^{p=2^k}\sum_{j \in \Omega_p^n}\tilde{S}_n(p)^2 + \sum_{p=1}^{p=2^k}\tilde{S}_n(p)\sum_{j \in \Omega_p^n}\left(\tilde{S}_n(p) - \Delta_j^{n-1}\right)$$

et donc, puisque d'après (6.57) le dernier terme est nul et en utilisant l'expression (6.56)

$$J_n = J_{n-1} - \frac{1}{2}\sum_{p=1}^{p=2^k}\sum_{j \in \Omega_p^n}\tilde{S}_n(p)^2 \tag{6.58}$$

On peut donc affirmer que l'application de la règle (6.57) pour initialiser le nouveau mnémon garantie $J_n \leq J_{n-1}$. En remplaçant dans (6.58) le terme $\tilde{S}_n(p)$ par sa valeur on obtient :

$$J_n - J_{n-1} = \frac{1}{2}\sum_{p=1}^{p=2^k}\frac{(\sum\limits_{j \in \Omega_p^n}\Delta_j^{n-1})^2}{card(\Omega_p^n)} \tag{6.59}$$

Ce dernier résultat nous permet d'écrire «**qu'il suffit que la moyenne des erreurs réalisées par le champ $CM^{(n-1)}$ sur l'ensemble des exemple concernés par une seule adresse du nouveau mnémon ne soit pas nulle pour que l'on obtienne une diminution de l'erreur quadratique par l'ajout de ce mnémon**».

6.3.2 Mnémons et probabilités conditionnelles

D'un point de vue probabiliste, chaque élément de la table de mémoire d'un mnémon (mnémon booléen) peut être vue comme la probabilité pour le mnémon d'être actif étant donnée la configuration courante des entrées du champ récepteur. En effet, considérons l'évenement $(S = 1)$ définit par «le mnémon donne 1 en sortie» et l'évenement $(V = V_i)$ comme étant «la configuration courante du champ récepteur est la $i^{ème}$ configuration parmis la liste ordonnée des 2^k configurations possibles». On peut associer à ces évenements les probabilités suivantes :

- $p(S = 1)$: probabilité que la sortie du mnémon soit 1 sur un ensemble de tirage des configurations d'entrées.
- $p(V = V_i)$: probabilité d'observer la $i^{ème}$ configuration sur un ensemble de tirage des configurations d'entrées.

  Les évenements $(V = V_i)$ sont des évènements mutuellements exclusifs et l'on a également:

$$p\left(\bigcup_{i=1}^{i=2^k} (V = V_i)\right) = 1$$

  par conséquent,

$$p(S = 1) = p\left([S = 1] \cap \left[\bigcup_{i=1}^{i=2^k} (V = V_i)\right]\right) = p\left(\bigcup_{i=1}^{i=2^k} ([S = 1] \cap [V = V_i])\right)$$

  et donc, d'après la définition des probabilités conditionnelles,

$$p\left(\bigcup_{i=1}^{i=2^k} ([S = 1] \cap [V = V_i])\right) = \sum_{i=1}^{i=2^k} p\left([S = 1]_{/[V=V_i]}\right) \cdot p\left(V = V_i\right) = \sum_{i=1}^{i=2^k} S\left[a(V_i)\right] \cdot p\left(V = V_i\right) \quad (6.60)$$

  (voir équation 6.52).

6.3.3 Champs mnémoniques stochastiques et neurones Sigma-Pi généralisés

Considérons un mnémon stochastique dont l'entrée est donnée par k capteurs stochastiques de loi de probabilité $p_j(x)$, $j \in [1, k]$ (x étant une variable à valeurs dans $\Re$). D'après l'équation 6.60, la probabilité que la sortie du mnémon soit 1 pour une valeur particulière de x est donnée par

$$p(S = 1) = \sum_{i=1}^{i=2^k} S\left[a(V_i)\right] \cdot p\left(V = V_i\right)$$

Or, puisque les capteurs sont indépendants,

$$p\left(V = V_i\right) = \prod_{j=1}^{j=k} p(r_j = e_j)$$

et donc

$$p(S = 1) = \sum_{i=1}^{i=2^k} S\left[a(V_i)\right] \cdot \prod_{j=1}^{j=k} p(r_j = e_j) \tag{6.61}$$

Il n'existe pas de développement simplifié de la relation (6.61) dans le cas général.. Cependant, elle montre qu'un mnémon stochastique effectue une combinaison non linéaire des sorties des capteurs stochastiques. Prenons l'exemple d'un mnémon d'ordre 3, associé à trois capteurs r_1, r_2 et r_3.

$$\begin{aligned} p(r_1 &= 1, r_2 = 1, r_3 = 1) = p_1(x) \cdot p_2(x) \cdot p_3(x) \\ p(r_1 &= 1, r_2 = 0, r_3 = 0) = p_1(x) \cdot (1 - p_2(x)) \cdot (1 - p_3(x)) \\ &\quad etc... \end{aligned}$$

Par conséquent, la procédure de mémorisation réalisée par le mnémon est, exprimée en termes probabilistes, *l'équivalent de l'opération réalisée par une unité sigma-*Π *généralisée acceptant comme entrée les* $p_j(x)$, $j \in [1, k]$. En effet, la définition d'une telle unité sigma-Π serait la suivante:

$$S(x) = \sum_{i_1, i_2, \ldots i_k \in [0,k]} \alpha_{i_1, i_2, i_3 \ldots i_k} \cdot \prod_{j=1}^{j=k} p_j^{(i_j)} + \beta$$

où les $\alpha_{i_1, i_2, i_3 \ldots i_k}$ et β sont des constantes réelles (poids et seuil du réseau). Mais, du point de vue pratique, les mnémons stochastiques ont le très grand avantage de **remplacer le calcul de ces multiples produits, très coûteux en temps de calcul, par une simple opération de mémorisation**.

On peut donner cependant une expression plus simple de (6.61) dans le cas d'un mnémon stochastique dont les entrées sont données par k capteurs stochastiques indentiques. Dans ce cas, les configurations d'entrées comportant le même nombre de bits actifs ont la même probabilité d'être observées. La probabilité d'observer une configuration comportant i bits actifs sur k est $C_k^i p^i \overline{p}^{(k-i)}$ (distribution binomiale). Par conséquent, la probabilité pour un tel mnémon d'être actif peut être représentée par la relation suivante

$$p(S=1) = \sum_{i=0}^{i=k} \alpha_i C_k^i p^i \overline{p}^{(k-i)}$$

où les α_i sont des constantes réelles. Dans ce cas particulier, le mnémon rempli la même fonction qu'une unité sigma-Π limitée réalisant **une approximation polynomiale** (d'ordre k) de la fonction $\widehat{\Psi}$. Notons cependant que ce type de mnémon n'est pas avantageux en termes de capacité mémoire. En effet, l'information y est stockée de façon redondante: toutes les configurations d'entrée comportant i bits actifs adressent des zones mémoires différentes mais comportant exactement la même information. Autrement dit, la solution consistant à utiliser des capteurs stochastiques identiques est une mauvaise solution car elle conduit à une inutile dépense de mémoire, à une diminution de la résolution de l'approximation effectuée et de l'ordre réel des mnémons.

En effet, soit un mnémon stochastique recevant m entrées, chacune mesurée par k détecteurs stochastiques identiques. L'ordre de ce mnémon sera donc 2^{m*k}. Cependant, le nombre de zones mémoires dont le contenu est réellement différent est seulement $(k+1)^m$. Par conséquent, on peut définir un ordre équivalent o_{eq} pour un tel mnémon dont la définition est donnée par la formule suivante:

$$2^{o_{eq}} = (k+1)^m$$

ou encore

$$o_{eq} = m \cdot \log_2(k+1)$$

Cette dernière expression montre que l'ordre équivalent augmente linéairement avec le nombre de dimensions, mais qu'il croît malheureusement seulement de façon logarithmique avec l'ordre effectif. Augmenter l'ordre du mnémon pour compenser la dépense inutile de mémoire n'est donc pas une bonne solution, sauf pour des mnémons d'ordre faible pour lesquels la différence en ordre et ordre équivalent n'est pas énorme. Une solution simple mais peu biologiquement pertinente consiste à utiliser au lieu du mnémon un opérateur sélectionnant une zone mémoire en fonction du nombre de bits à 1 dans l'ensemble des signaux fournis par les capteurs stochastiques. Mais il faut souligner que ce résultat décevant ne vaut que pour des mnémons possédant des récepteurs strictements identiques et plaide en faveur de l'introduction de diversité dans les caractéristiques des récepteurs stochastiques. Notons que la diversité des réponses est une caractéristique des récepteurs sensoriels, qui ne sont jamais identiques, même pour une population de récepteurs du même type cellulaire.

6.4 Comparaison avec les perceptrons multicouches

6.4.1 Principales différences théoriques entre neurones formels et mnémons

Sur le plan théorique, plusieurs différences fondamentales existe entre un neurone formel du type de celui décrit par l'équation (5.26) et un mnémon. En ce qui concerne l'approximation de fonction booléennes, le neurone formel canonique réalise une séparation de l'hypercube $[0,1]^n$ (représentant l'ensemble des configurations d'entrées possibles) selon un hyperplan défini par les poids des connections. Il associe à chacun des demi espaces ainsi séparés la réponse 0 ou la réponse 1. Par conséquent, le neurone formel n'est capabale d'approximer qu'une toute petite partie des fonctions booléennes de $[0,1]^n$ sur $[0,1]$. Le pourcentage de fonctions qu'un neurone formel peut réaliser exactement est en fait égal au nombre de configurations dans le cube $[0,1]^n$ séparables par un hyperplan rapporté au nombre de fonctions possible (2^{2^n} !). Le mnémon est lui capable de réaliser exactement n'importe laquelle de ces fonctions.

6.4.2 Application à l'apprentissage de fonctions booléennes

Un seul mnémon d'ordre k peut coder toute fonction booléenne de k entrées avec une mémoire de taille 2^k bits. Les réseaux de neurones conventionnels sont également capables d'apprendre des fonctions booléennes dont les propriétés sont stockées dans les synapses. Un premier élément de comparaison entre mnémons et perceptrons multi-couches consiste en l'estimation de la mémoire minimum nécessaire à un réseau de neurones pour implémenter parfaitement une fonction booléenne d'un ordre donné. D'après ce critère de mémoire, les perceptrons multicouches peuvent faire preuve de leur efficacité si la taille mémoire nécessaire est inférieure ou égale à 2^k bits (moins d'un bit par exemple).

Pour tester cette efficacité, nous avons appliqué un algorithme de rétropropagation standard [257, Vogl et al., 1988] à des perceptrons multi-couches comportant une seule couche cachée de taille variable. Les performances des réseaux de neurones ont été testées sur 50 fonctions booléennes d'ordre $k = 3$ à $k = 10$ choisies au hasard. Conformément à la méthode choisie, les poids du réseau ne sont modifiés qu'après une présentation complète de tous les exemples. L'arrêt de la simulation a lieu soit si le réseau obtient 100 % de réponses correctes ou bien si l'on atteint le nombre maximal de présentation de tous les exemples (5000). Nous avons ensuite tracé le pourcentage de réponses correctes en fonction du nombre de bits stockés dans les poids du réseau divisé par le nombre d'exemples.

Les résultats (figure 19) montrent que des performances parfaites sont obtenues uniquement avec plus de 20 bits par exemple. Si l'on considère uniquement ce critère de mémoire, les réseaux de neurones sont donc beaucoup moins efficaces qu'une simple table de vérité.

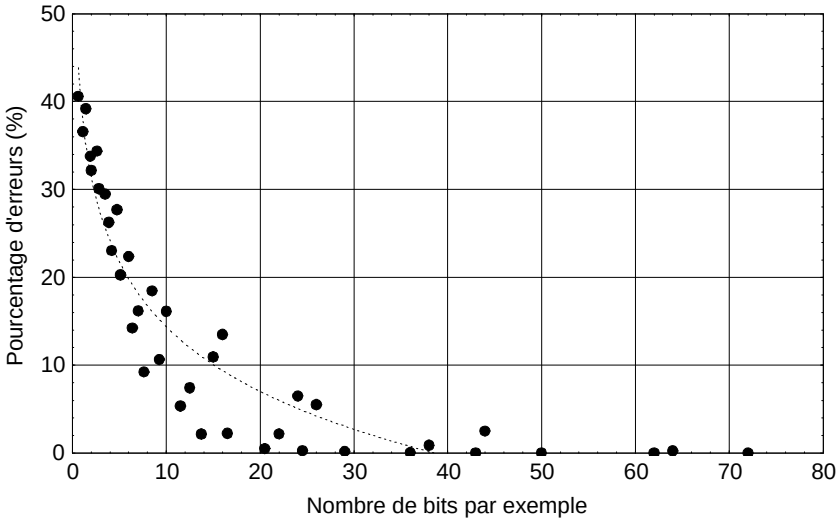

Figure 19. Apprentissage d'une fonction booléenne par un perceptron multicouche (voir texte)

6.4.3 Détection des symétries dans une image

La détection des symétries chez l'Homme est très rapide et fiable, en particulier pour les symétries par rapport aux axes cardinaux (axes horizontal et vertical). Cette tâche a également été citée comme un exemple d'opération relativement complexe qu'un réseau de neurones peut accomplir [219, Sejnowski et al., 1986] . Dans cette courte étude nous avons comparé les capacités d'apprentissage et de généralisation des perceptrons multi-couche et des champs mnémoniques à classer des images noir et blanc (issues d'une base d'exemples) en images symétriques ou non-symétriques. Nous avons utilisé une procédure d'apprentissage supervisé. La détection d'images symétriques par rapport à un axe particulier est un problème d'ordre 2 qui peut être théoriquement résolu sans erreur par un perceptron comportant simplement 2 neurones cachés entièrements connectés à la matrice image d'entrée. En pratique, nous avons cependant observé une erreur résiduelle significative ($>$5 %) qui a été obtenue sur une base de test même pour des tailles d'image réduites (6 par 6 pixels) et pour une base d'apprentissage relativement importante (1024 échantillons).

L'utilisation de champs mnémoniques est particulièrement adaptée à cette tâche. Prenons un simple mnémon possédant un champ récepteur de deux pixels placés de façon symétrique dans l'image. Une seule unité de ce type donnera une réponse correcte pour toutes les images symétriques et pour 50 % des images non symétriques. Un champ mnémonique utilisant une seule de ces unités présente des performances assez pauvres: 75 % de réponses correctes si il y a autant d'images symétriques que d'images non symétriques dans la base de test. Mais si 10 unités comportant des champs récepteurs différents fonctionnent en parallèle, alors la probabilité qu'une image non symétrique active toutes les unités devient très faible (0.1 %).

Pour illustrer cette idée, nous avons réalisé une simulation utilisant un champ mnémonique comportant 30 mnémons d'ordre 4 utilisant une mémoire de seulement 60 octets. Ce champ mnémonique est capable de généraliser un axe de symétrie sur des images de 16 par 16 pixels avec un taux d'erreur de 0,16 % (sur la base de test), avec une base d'apprentissage comportant seulement 256 exemples. La figure (20) résume également plusieurs simulations destinées à comparer les performances de ce champs mnémonique à celles d'un réseau de neurone pour des tailles différentes de la base d'apprentissage. Les performances obtenues sont clairement en faveur du champ mnémonique qui utilise bien moins de mémoire que le perceptron à une couche cachée (296 octets contre 60 pour le champ mnémonique).

6.5 Problèmes d'ordre plus élevé: application au contrôle sensorimoteur

Parmis les différents types d'approximateurs utilisés avec succès pour le contrôle adaptatif de systèmes dynamiques présentant des non-linéarités importantes, certains utilisent, comme les champs mnémoniques, des tables de mémoires [4, Albus, 1975] [10, Atkeson et Reinskenmeyer, 1988] . Dans ces études, l'explosion

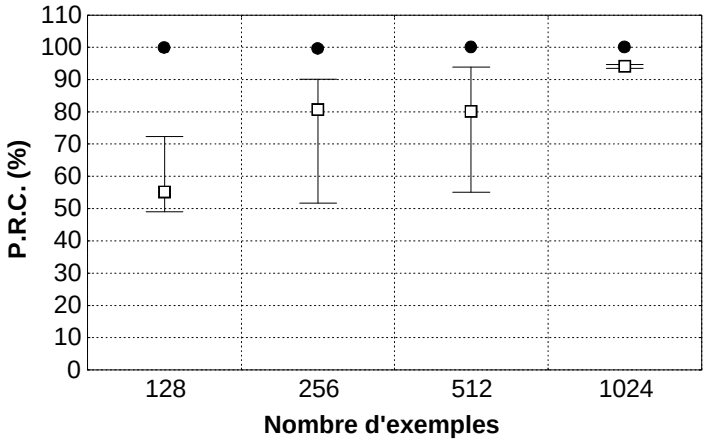

Figure 20. Comparaison des performances d'un perceptron multicouche et d'un mnemon dans une tâche de détection de symétrie

combinatoire est limitée soit par l'utilisation d'une méthode de compression de la mémoire utilisée [4, Albus, 1975] , ou par l'utilisation d'une stratégie d'utilisation du «plus proche voisin» [10, Atkeson et Reinkensmeyer, 1988] . Cependant, de tels méthodes ne peuvent que difficilement être utilisées pour approximer des transformations impliquant une grand nombre de variables. Or une fonction décrivant la dynamique (inverse ou directe) d'un système multidimensionnel possède justement cette caractéristique. Afin de montrer que les champs mnémoniques peuvent être appliqués avec succès au contrôle de tels systèmes, nous proposons de les utiliser pour contrôler la trajectoire d'un modèle de bras articulé à deux degrés de liberté. Nous avons également comparé les performances obtenues par les champs mnémoniques à celles d'un réseau de neurones formels accomplissant la même tâche. Cette comparaison a été faite en termes de vitesse d'apprentissage, pour une même quantité de mémoire utilisée par les deux modèles.

6.5.1 Modèle robotique et schéma de contrôle adaptatif choisis

Cette application des champs mnémoniques au contrôle d'un bras articulé à deux degrés de liberté s'inspire d'un travail de Gomi et Kawato [87, Gomi et Kawato, 1993] , proposant une application de leur schéma de contrôle adaptatif intitulé «feedback error learning» à un contrôleur adaptatif fonctionnant en boucle fermé. Nous avons donc choisi d'utiliser un schéma de contrôle identique ainsi que le même modèle de bras que les auteurs.

La trajectoire désirée est un cercle de 10 cm de diamètre qui doit être tracée à la vitesse de 12.56 radians par seconde (2 Hz) par un bras articulé dont la dynamique est décrite par les équations suivantes:

$$\begin{aligned}\tau 1 &= \left(I_1 + I_2 + 2M_2L_1S_2\cos\theta_2 + M_2\left(L_1\right)^2\right)\ddot{\theta}_1 + \left(I_2 + M_2L_1S_2\cos\theta_2\right)\ddot{\theta}_2 \\ &\quad -M_2L_1S_2\left(2\,\dot{\theta}_1 + \dot{\theta}_2\right)\dot{\theta}_2\,\sin\theta_2 + b_1\,\dot{\theta}_1 \end{aligned} \tag{6.62}$$

$$\tau_2 = \left(I_2 + 2M_2L_1S_2\cos\theta_2\right)\ddot{\theta}_1 + I_2\,\ddot{\theta}_2 + M_2L_1S_2\left(\dot{\theta}_1\right)^2\sin\theta_2 + b_2\,\dot{\theta}_2 \tag{6.63}$$

où z_1 et z_2 représentent les couples appliqués à chacune des articulations, et θ_1 et θ_2 sont les angles articulaires. Les valeurs des paramètres utilisés, choisis aussi proches que possible de la biomécanique du bras humain par les auteurs [253, Uno et al., 1989] , sont les suivantes:

Paramètre	Articulation 1	Articulation 2
M_i (kg) : masse	0,9	1,1
L_i (m) : longueur	0,25	0,35
S_i (m) : distance au centre de masse	0,11	0,15
I_i $(kg.m^2)$: inertie	0,065	0,100
b_i $(kg.m^2/s)$: coefficient de viscosité	0,08	0,08

L'intégration du modèle numérique est réalisée avec l'algorithme de Runge-Kutta (ordre 4) pour un pas de simulation de 0.001 s (25 secondes de simulations soit 25000 pas). Le schéma de contrôle utilise deux boucles de rétroactions fonctionnant en parallèle [87, Gomi et Kawato, 1993] . La première est un servo-contrôleur classique utilisant l'erreur de poursuite et ses dérivées secondes et premières:

$$\tau_c = K_2 \left(\widehat{\ddot{\theta}} - \ddot{\theta} \right) + K_1 \left(\widehat{\dot{\theta}} - \dot{\theta} \right) + K_0 \left(\widehat{\theta} - \theta \right)$$

avec $\theta = [\theta_1, \theta_2]^T$ et $K_0 = [100, 100]$, $K_1 = [7, 7]$, $K_2 = [0.1, 0.1]$. La seconde boucle comporte un module adaptatif acceptant en entrée la trajectoire angulaire et ses dérivées secondes et premières:

$$\tau_n = \Psi \left(\theta, \dot{\theta}, \ddot{\theta}, w \right)$$

où le vecteur w représente l'ensemble des paramètres soumis à apprentissage du filtre Ψ. Dans le cadre de cette étude, ces paramètres sont les poids synaptiques et seuils du réseau de neurones formels ou bien les contenus (à valeur réelle) des cases mémoires des mnémons du champ mnémonique. Le filtre Ψ accepte donc 6 entrées et fournit en sortie deux couples articulaires $\tau_c = [\tau_{c,1}, \tau_{c,2}]^T$. Ces paramètres sont adaptés au cours de l'apprentissage suivant la règle («feedback error learning»)

$$\frac{dw}{dt} = \alpha \left(\frac{\partial \Psi}{\partial w} \right)^T \tau_c \tag{6.64}$$

6.5.2 Description du réseau de neurones et du champ mnémonique utilisés

Le champ mnémonique comporte seulement deux mnémons d'ordre six (5120 bits de mémoire). Chaque variable d'entrée est codée sous forme binaire par cinq capteurs stochastiques identiques possédant tous une réponse stochastique linéaire:

$$p(x) = \frac{x - \min(x(t), t)}{\max(x(t), t) - \min(x(t), t)} \tag{6.65}$$

pour une variable d'entrée donnée x. Cette sensibilité des capteurs évolue temporellement car nous

avons choisi de rendre le système entièrement auto-adaptatif: la valeur maximale et minimale d'une entrée est réactualisée à chaque itération lors de la stimulation. Cette opération est en quelque sorte une procédure d'autocalibration continue. La vitesse d'apprentissage pour le mnémon est paramétrée par le coefficient α de l'équation (6.64). Les sorties des mnémons (le contenu des tables de mémoire) sont des valeurs flottantes en double précision. Le réseau de neurones utilisé est un perceptron à une couche cachée comportant 18 neurones (5248 bits de mémoire). Le réseau de neurone n'utilise pas le système d'autocalibration (6.65). Les poids du réseau sont réactualisés à chaque itération d'après la loi d'apprentissage (6.64) [257, Vogl et al., 1988] .

6.5.3 Résultats

Un aperçu des trajectoires obtenues avant et après apprentissage pour les deux modèles est donné sur la figure (21). Les performances des deux méthodes ont été comparées à la fois en ce qui concerne la qualité du suivi de la trajectoire (mesurée par la racine carrée de l'erreur quadratique ou «erreur R.M.S», mais aussi en terme de vitesse réelle d'apprentissage. Après apprentissage (25000 itérations), l'erreur R.M.S. obtenue avec le champ mnémonique est tout à fait comparable à celle obtenue avec le réseau de neurone (environ 0.015). Mais la vitesse réelle d'apprentissage est beaucoup plus importante pour le champ mnémonique. Ceci se comprend aisément car il n'effectue quasiment qu'un seul type d'opération, en l'occurence particulièrement rapide sur un ordinateur: l'adressage mémoire. Sur la figure (22), le temps réel de simulation est environ 6 fois plus faible pour le système utilisant un champ mnémonique. En réalité, ce rapport des vitesse d'apprentissage est beaucoup plus important, car les 500 s de simulation consommées par le système intégrant un champ mnémonique sont en grande partie due aux opérations nécessaires à l'intégration des équations (6.62) et (6.63). On observe également des différences qualitatives importantes dans le comportement des deux systèmes. La courbe de décroissance de l'erreur réalisée par le champ mnémonique est beaucoup plus «douce». Le réseau de neurone semble en particulier être resté bloqué quelques temps dans un minimum locale (de $t = 350$ s à $t = 1100$ s). La trajectoire finale produite par le champ mnémonique, au comportement stochastique par nature, est plus fluctuante.

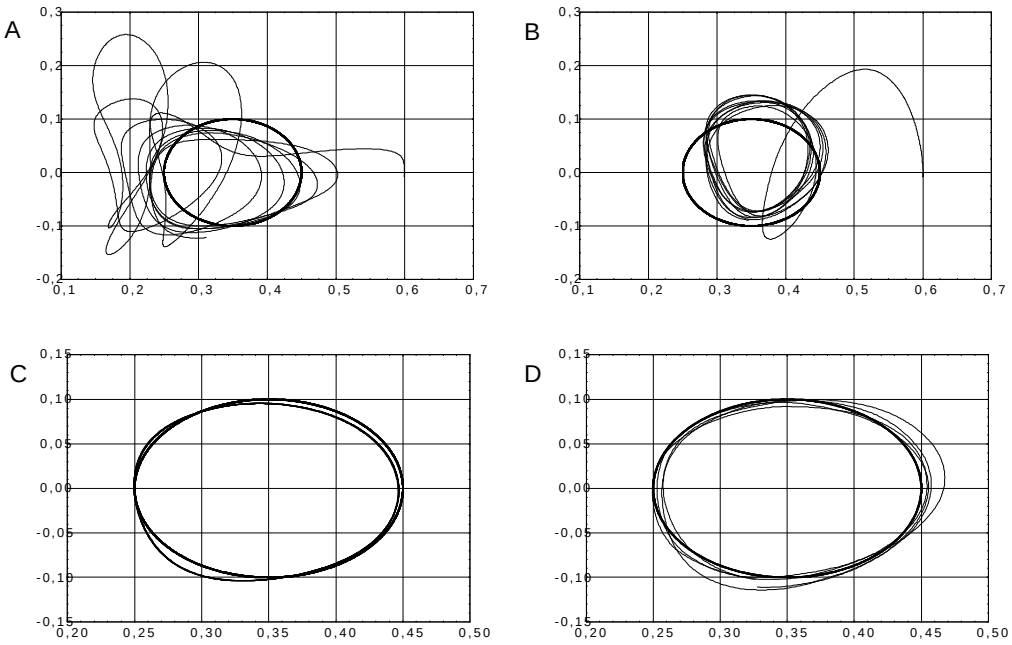

Figure 21. Comparaison de la qualité du contrôle d'un bras articulé à deux degré de liberté et de la qualité de l'apprentissage pour un perceptron multicouche et pour un contrôleur utilisant des champs mnémoniques. Le mouvement à réaliser est un cercle traçé à 2 Hz (deux tours par seconde). A) et C) Mouvement initial et final (après apprentissage) réalisé par le bras contrôlé par un perceptron multicouche. B) et D) Résultats obtenus pour le bras contrôlé par des champs mnémoniques.

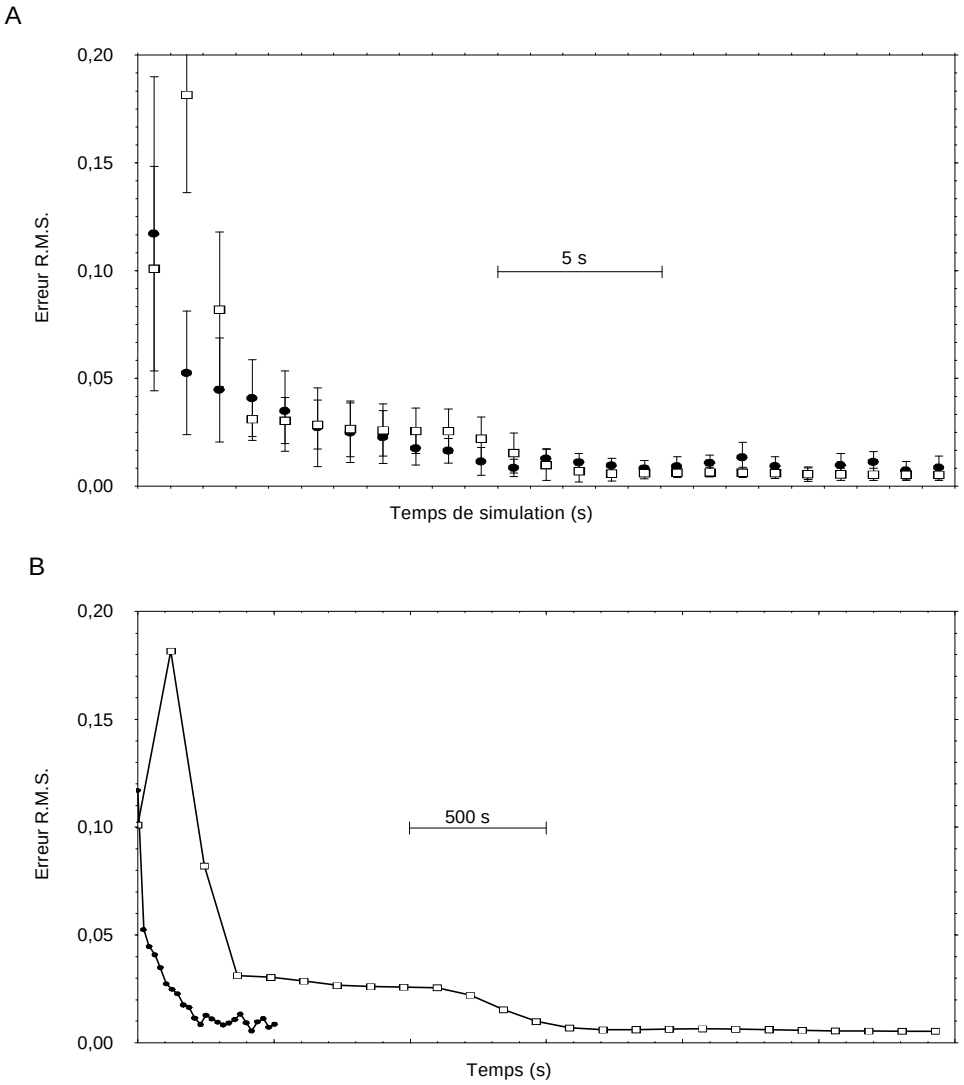

Figure 22. Comparaison de la qualité (A) et de la vitesse de l'apprentissage (temps réel, B) pour le contrôleur utilisant soit un perceptron multicouche soit des champs mnémoniques. La figure (A) représente l'évolution de l'erreur RMS entre trajectoire désirée et obtenue en fonction du temps de simulation. Les performances du MLP sont représentées par des carrés vides alors que des cercles pleins figurent celles des champs mnémoniques. La figure (B) reprend les mêmes mesures et mêmes notations mais en fonction du temps réel de la simulation. Ainsi, le perceptron, pour des performances comparables à celle des champs mnémoniques, accomplit la tâche en un temps six fois plus long.

6.6 Discussion du modèle

Comme le modèle présenté dans le chapitre précédent, auquel il ne prétend nullement s'opposer, le champ mnémonique utilise des unités de calcul fonctionnant en parallèle possédant des caractéristiques non linéaires marquées. En fait, l'on pourrait dire que l'on ne peut pas imaginer d'unité «moins linéaire» que le mnémon. Qu'il soit considéré sous sa forme «table de mémoire» ou sous sa forme «unité sigma-Pi généralisée», le mnémon n'est rien d'autre qu'une transcription des lois de réponse d'un élément de calcul à l'état courant de ses afférences. Ce sont justement ces caractéristiques non linéaires marquées qui permettent de supprimer toute couche intermédiaire, ou «représentation interne» de ce type d'architecture. La suppression des unités cachées permet ainsi, et comme dans le modèle précédent, d'utiliser des lois d'apprentissage supervisées ou non supervisées simples ne nécessitant pas de rétropropagation complexe d'information dans les différentes couches de calcul (comme c'est le cas avec les perceptrons multicouches). Ce chapitre nous a permis de montrer que les performances obtenues avec les champs mnémoniques, aussi bien du point de vue de la quantité de mémoire utilisée que du point de vue de la précision du résultat obtenu n'avait rien à envier à celles des perceptrons multicouches. Soulignons de plus que ces performances ont été obtenues avec une dégradation de l'information donnée en entrée aux mnémons. Dans le dernier exemple, les mnémons d'ordre six n'ont accés qu'à une toute petite partie de l'information contenue dans la valeur des variables d'entrée[27]. Mais, d'un point de vue pratique, les mnémons ont l'avantage de n'être on ne peut plus adaptés à une implémentation sur ordinateur: ils ne mobilisent principalement qu'un seul type d'opération: l'adressage mémoire. Or cette opération est réalisée très rapidement en seulement quelques cycles d'horloge par tous les ordinateurs. C'est pourquoi ils ne peuvent que battre «à plate couture» les perceptrons multicouches en terme de vitesse réelle de calcul. En effet, ce type d'architecture fait largement appel au calcul en virgule flottante (ou fixe) et en particulier à la multiplication, très coûteuse en temps de calcul. Il est cependant évident que l'utilisation des mnémons devient délicate lorsque l'ordre de la fonction à approximer devient très élevé ou lorsque le nombre d'entrées augmente inconsidérément. La capacité mémoire des ordinateurs actuels peut alors être rapidement dépassée[28]. Mais ce problème existe également pour les perceptrons multicouches. Les champs mnémoniques sont d'autre part bien plus simples à mettre en oeuvre que les perceptrons multicouches.

Il reste que certains points théoriques doivent être encore développés. Le premier problème, exposé également dans la discussion sur une adaptation multirésolution du précédent modèle, est la définition d'une procédure d'optimisation des champs récepteurs des mnémons fonctionnant en même temps que l'apprentissage proprement dit. Dans la plupart des études présentées ici, ceux-ci sont tirés au hasard et laissés inchangés durant

[27] Les mnémons ont accès à 6 bits d'information au total sur les 6 variables d'entrées considérées. Avec un calcul flottant en simple précision, $6 \times 32 = 192$ bits sont utilisables par le perceptron multicouche.

[28] Ce problème peut être rencontré dans le cadre d'applications telles que la reconnaissance d'images (de visages) ou la reconnaissance de la parole. Dans ce dernier exemple, si l'on applique aucune procédure de prétraitement du signal échantillonné (à 11 Khz par exemple), le champ mnémonique doit prendre en compte $11000 \times 16 = 176000$ bits en entrée par seconde de signal !

toutes la simulation. La seule procédure d'optimisation que nous pouvons proposer est une substitution par un retirage du champ récepteur des mnémons dont la valeur de sortie change continuellement de signe (pour les mnemons booleens) ou des mnémons ne participant que très peu à la sortie du champ mnémonique (pour les mnémons à valeur réelle, amplitude moyenne de sortie faible par rapport aux autres mnémons du champ). Cette procédure est d'ailleurs très proche de la solution proposée par [38, Cannon et Slotine, 1995] pour les «réseaux d'ondelettes». Le second problème que peuvent poser les champs mnémoniques, en particulier pour des applications de contrôle ou de traitement des signaux, est la discontinuité et l'aspect bruité du signal de sortie. En effet, même si la valeur moyenne du signal produit par un champ mnémonique peut être très proche de la valeur désirée, ce signal est caractérisé par des fluctuations importantes. Ce problème peut être résolu par un filtrage raisonnable des hautes fréquences réalisé soit par un filtre passe-bas adapté ou directement par le système mécanique commandé par le champ mnémonique[29] si celui-ci ne présente pas d'instabilités pour les hautes fréquences.

Enfin, les mnémons stochastiques ont été introduits dans un souci de soutenir la pertinence biologique du modèle. Nous avons cependant montré que cette approche n'était pas optimale du point de vue de la quantité de mémoire mobilisée. Ce modèle est de plus très primaire car les neurones biologiques sont loin d'être stochastiques et montrent dans certains cas une très haute stabilité temporelle de la réponse a un stimulus périodique [13, Bair et Koch, 1996] [218, Segundo et al., 1995] . Ce qui fait en tout cas, du point de vue neurophysiologique, la force de ce modèle, c'est là encore sa capacité à utiliser des règles d'apprentissage locales et le fait qu'il mobilise des unités fonctionnant en tout ou rien. Mais bien évidemment, il est nécessaire , pour admettre que ce type d'architecture soit un hypothèse plausible du fonctionnement de certaines structures neuronales, de remettre là encore en cause le dogme du neurone semilinéaire «sommateur de ses entrées».

[29] C'est le cas dans la dernière application présentée dans ce chapitre.