Table des matières

Remerciements

C'est absurde, repartit Ulrich avec force. J'ai dit que ce qui comptait, ce n'était pas un faux pas, mais le pas qui suit ce faux pas. Mais qu'est-ce qui compte après le pas suivant ? Sans doute, bien sûr, celui qui suit ? Et après le énième pas, le pas $n+1$? Cet homme devrait donc vivre privé de fin et de décision, privé même, somme toute, de réalité. Pourtant il est bien vrai que c'est toujours le pas suivant qui compte. La vérité est que nous ne disposons d'aucune méthode pour traiter comme il faudrait cette série infatigable.

Robert Musil,
L'homme sans qualités

Ce mémoire présente le travail de trois années de thèse passées sous la direction scientifique du Professeur Robert Azencott au sein du groupe DIAM du Centre des Mathématiques et de Leurs Applications (CMLA) de l'Ecole Normale Supérieure de Cachan, du laboratoires de Probabilités de l'Université de Paris VI, et du Groupe Logiciel du Département Electronique de la Direction de la Recherche de Renault.

L'objectif de ce travail était l'étude des potentialités des fonctions de la classe des réseaux de neurones formels (RNF) à contrôler des systèmes dynamiques non linéaires. Un objectif applicatif à Renault est le contrôle de la combustion dans les cylindres d'un moteur à essence.

Je veux d'abord remercier Robert Azencott pour avoir dirigé mes recherches avec tant d'attention. Il a su en effet me guider vers des directions fructueuses tout en me laissant libre dans la suite de leur exploration. En me faisant sentir les pré-requis pour le développement de ces idées, il m'a amené

autant vers l'apprentissage de domaines des mathématiques qui m'étaient peu connus, que de méthodologies de travail auxquels je n'avais pas encore été formé.

L'écoute dont j'ai bénéficié lorsque je suis allé exposer les principaux résultats de ce travail de thèse à l'université de West Viriginia ainsi qu'aux congrès ICANN98, NOLTA98, ICANN99 et ISIC99 me conduit à penser que les directions prises tout au long de ces trois années peuvent répondrent à certaines de questions que la communauté étudiant les RNF et le contrôle intelligent se pose aujourd'hui. A ces occasions, les discussions avec les autres scientifiques ont été très fructueuses et ont contribuées à l'enrichissement de mes travaux.

D'autre part, mon intégration au sein du Groupe Logiciel de Renault est un des facteur qui m'ont permis de comprendre certaines implications industrielles des avancées de l'électronique et de l'informatique. Je pense notamment aux problématiques issues de la finesse sans cesse croissante des capteurs physiques disponibles ainsi que des nouveaux défis de la conception de logiciels distribués. Je dois de nombreux éléments de ces réflexions à Philippe Dubois et à François Ougier.

Ces remerciements ne seraient pas complets s'ils ne s'adressaient pas aussi à tout ceux qui m'ont donné le goût, les compétences et l'énergie nécessaires à la réalisation de ces travaux ; ils se reconnaîtront.

Ils ne seraient pas non plus complets s'ils ne mentionnaient pas la mémoire de mon père, décédé avant le début de ces travaux, dont le caractère méditerranéen et le métier de psychiatre sont sans doute les premiers responsables de mon goût immodéré pour la conjugaison du formalisme et de la méthode expérimentale dans le but de comprendre des phénomènes par nature hors d'atteinte. Mon attirance pour l'activité de chercheur provient en grande partie de cela.

Introduction

Contrôle de systèmes dynamiques non linéaires et apprentissage de réseaux de neurones formels

L'étude des capacités de contrôle des réseaux de neurones formels (RNF) est passionnante à plus d'un titre. D'abord parce qu'elle se situe à la croisée de plusieurs théories liées depuis longtemps.

En effet, en 1948 dans **cybernetica**, Norbert Wiener [Wiener, 1948] posa autour de la notion de feed-back des concepts qui participèrent à l'avancée de la théorie du contrôle comme à celle des RNF. Il est facile de se représenter pourquoi ces deux champs de connaissances sont proches :

- d'un coté, la théorie mathématique du contrôle s'intéresse aux moyens de déterminer comment amener les trajectoires d'un système dynamique paramétré vers une référence,
- d'un autre celle des RNF regroupe des techniques et résultats mathématiques permettant de choisir parmi une classe de fonctions celle dont le graphe est le plus proche d'une référence. Typiquement, le cheminement qui mène au choix d'une fonction particulière au sein une classe peut être vu comme une trajectoire dans le bon espace.

Bien sûr, ces définitions sont volontairement simplifiées afin de mettre en avant leurs points communs ; la partie I de ce mémoire les précise et les complète.

La "proximité apparente" de ces deux théories n'est pas le seul intérêt de leur étude conjointe, les enjeux applicatifs qui y sont liés y ajoutent beaucoup.

Enjeux applicatifs

L'industrie d'aujourd'hui développe des moyens de connaissance très fins des processus qu'elle veut contrôler au plus juste. Dans l'industrie automobile par exemple, si la résolution ou la simulation d'équations provenant de la physique et de la chimie des moteurs reste hors de portée des puissances de

calcul actuelles, des appareillages de mesure à haute fréquence de l'état d'un moteur en régime transitoire commencent à être disponibles. Une grande attention est donc portée aux les méthodes d'approximation du comportement d'un moteur, afin de les confronter à ces données.

Les modèles linéaires étant désormais trop grossiers pour rendre compte des données disponibles, le choix se porte vers les solutions non linéaires, qui sont très diverses. La théorie des RNF, qui étudie différents types de modèles (comme les perceptrons multicouche, machines de Boltzmann, cartes auto-organisatrices, support vector machines, et réseaux bayésiens) à la fois pour leurs nonlinéarités, leur robustesse, et leur efficacité statistique, apporte des débuts de réponses à cette problématique.

Intérêt des perceptrons affines par morceaux

Bien sûr, ce travail n'est pas le premier dans le domaine de l'utilisation conjointe des RNF et du contrôle et il ne prétend pas y mettre un point final. Néanmoins j'espère qu'il pose de bonnes questions et que les réponses qu'il apporte permettront d'aller plus en avant dans les résultats et les applications qui en découlent.

Les relations qu'il explicite entre les perceptrons affines par morceaux (comme la partie II l'expose, il s'agit d'une forme particulière de perceptrons mutlicouche) et des fonctions affines par morceaux ouvrent de nombreuses possibilités applicatives. Au dela des résultats obtenus dans le cadre de cette thèse, il est possible de prolonger l'étude de cette forme de perceptrons en faisant par exemple des liens avec les systèmes hybrides (plusieurs systèmes dynamiques avec des fonctions de bascule des uns aux autres) ou l'étude plus poussée de l'évolution des paramètres de cette famille de modèles empiriques pendant leur mise au point. Dans le cadre de leur utilisation pour le contrôle, il paraît envisageable de ramener le comportement d'un système dynamique contrôlé par une telle fonction à une chaîne de Markov suffisamment simple pour en obtenir des résultats intéressants (cf. section 6.2).

Les difficultés recontrées habituellement lors de l'utilisation de RNF comme contrôleurs non linéaires sont de trois ordres :

– Lors de la **phase d'initialisation** : un contrôleur initialisé aléatoirement conduira dans la plupart des cas le système contrôlé dans un état (ou une zone de son espace d'état) inintéressant ou inexploitable (par exemple pour une réaction chimique : un état où elle n'a plus lieu ; ou bien pour un moteur à combustion : un calage ou la casse du vi-

lebrequin). Or la méthode d'initialisation des RNF la plus largement répandue est justement aléatoire.
- Il existe de nombreuses **méthodes de mise au point des paramètres** d'un RNF, et dans le cas de cette mise au point dans une boucle fermée, la durée des trajectoires à observer avant mise à jour est un paramètre difficile à choisir. En effet, cette durée varie suivant le point de départ de la trajectoire.
- Une fois les paramètres réglés à partir de l'observation de certaines trajectoires, le problème se pose de la **généricité du contrôleur obtenu** : quelles seront ses performances dans des zones de l'espace d'état peu visitées par ces trajectoires ?

C'est à ces problématiques que ce travail apporte des réponses. Il contient des preuves théoriques ainsi que des méthodologies de mise en place automatique d'un contrôleur émulé par un RNF en boucle fermée avec un système dynamique dont on connait un modèle partiel. Il s'agit de la conception automatique d'un contrôleur émulé par un RNF, et pas celle d'un contrôleur adaptatif, domaine pour lequel il existe une autre gamme de méthodes (comme le CMAC [Kuvayev & Sutton, 1997]), ni celle de l'utilisation d'un RNF pour émuler le comportement d'un sous-système utilisé dans une vaste stratégie de contrôle (par exemple un contrôle prédictif).

Le plan de ce mémoire suit le cours de cette introduction : la partie I présente les éléments de contrôle non linéaire et linéaire nécessaires à la compréhension de la suite avant de passer en revue les différentes techniques de contrôle "neuronal" les plus communément utilisées.

La partie II positionne les perceptrons affines par morceaux (Piecewise Affine Perceptrons : PAP) au milieu d'autres classes de RNF et dans le cadre du contrôle. Lors de cette présentation, une méthode d'initialisation des PAP est exposée, elle repose sur la relation très forte qui existe entre eux et les fonctions affines par morceaux (section 5.1). Cette partie contient aussi des résultats de stabilité sur le contrôle engendré par les PAP, expose comment ils permettent de ramener certains systèmes dynamiques à des chaînes de Markov, et détaille finalement un algorithme d'apprentissage adapté au contrôle par réseaux de neurones.

La dernière partie consiste en l'application des résultats et de la méthodologie obtenus au contrôle de la combustion d'un moteur à essence.

Enfin, la conclusion ouvre vers différents prolongements possibles de ces travaux.

Première partie

Du contrôle aux réseaux de neurones formels

Chapitre 1

Systèmes dynamique et problématique du contrôle

1.1 Systèmes dynamiques

Les systèmes dynamiques formalisent les processus d'évolution d'un système. L'ensemble de tous les états d'un processus est appelé *espace des phases* [Arnold, 1974]. On le notera $\mathcal{X}$. Un tel processus peut être déterministe (lorsque son état présent détermine totalement ses états passés et futurs) ou stochastique (si ce n'est pas le cas).

Définition 1 (Système dynamique continu ou discret) *Un processus est différentiable si son espace des phases est muni d'une structure de variété différentiable, et que ses changements d'état au cours du temps sont définis par des fonctions différentiables. Il peut alors être formalisé par le système dynamique continu suivant ($x \in \mathcal{X}$ est l'état du système, t indice le temps) :*

$$\dot{x}_t = f(t, x_t) \tag{1.1}$$

A contrario, lorsque les changements du processus ne peuvent pas être définis par des fonctions différentiables, il peut être formalisé par le système dynamique discret (ou système aux différences) suivant :

$$x_{t+1} = f(t, x_t) \tag{1.2}$$

On note aussi (1.2) en utilisant l'opérateur "avance" σ (formellement $\sigma x_t = x_{t+1}$) ce qui allège les notations en permettant de se passer de l'indiçage par le temps :

$$\sigma x = f(x)$$

La *trajectoire* d'un système dynamique issue d'un point x_{t_0} est le lieu de l'évolution du système passant par x_{t_0} en t_0.

La plupart des systèmes physiques livrés à eux-mêmes sont modélisables par des systèmes dynamiques. L'étude des systèmes dynamiques est l'objet d'une vaste littérature sans cesse en évolution.

L'étude des systèmes dynamiques vise notamment à déterminer le lieu des trajectoires d'un tel système.

Une notion capitale dans ce cadre est de savoir si, pour un système donné, deux états "proches" à un instant t de l'évolution du système vont rester proches, voire se confondre, ou s'éloigner, ou même si des situation de proximité et d'éloignement vont s'alterner, et si oui à quel rythme.

1.2 Fonctions de Lyapunov

Les méthodes de Lyapunov permettent de répondre à certaines questions de stabilité de systèmes dynamiques.

Dans cette section, le système considéré est différentiable et on le ramène au cas où $x \equiv 0$ en est une solution (Le passage d'un système générique $\dot{y} = f(t, y)$ avec $y \equiv y_0$ comme solution à un tel système se fait par le changement de variable : $x = y - y_0$ et $F(t, x) = f(t, x) - f(t, y_0)$).

Définition 2 (Stabilité de Lyapunov) *La solution $x(t) \equiv 0$ de (1.1) est dite* **lyapunov-stable** *si quel que soit ε positif strictement, il existe $\delta(\varepsilon, t_0)$ positif strictement tel que :*

Si $t \geq t_0$ et x_0 vérifie $||x_0|| \leq \delta(\varepsilon, t_0)$, alors toute trajectoire $\zeta_t(t, x_0)$ de (1.1) issue de x_0 en t_0 vérifie :

$$||\zeta_t(t, x_0)|| \leq \varepsilon$$

Définition 3 (Stabilité uniforme) *La solution $x(t) \equiv 0$ de (1.1) est dite* **uniformément stable** *par rapport au temps si quelque soit ε strictement positif, il existe $\delta(\varepsilon)$ strictement positif (indépendant de t_0) tel que : pour tout $t \geq t_0$ et x_0 tel que $||x_0|| \leq \delta(\varepsilon)$, toute trajectoire $\zeta_t(t, x_0)$ de (1.1) issue de x_0 en t_0 vérifie :*

$$||\zeta_t(t, x_0)|| \leq \varepsilon$$

Par exemple, le système :

$$\dot{x} = \frac{x}{t},\ x(0) = x_0$$

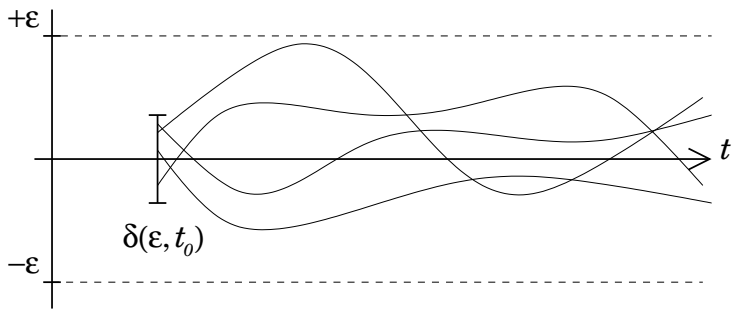


FIG. 1.1 – Illustration de la stabilité au sens de Lyapunov.

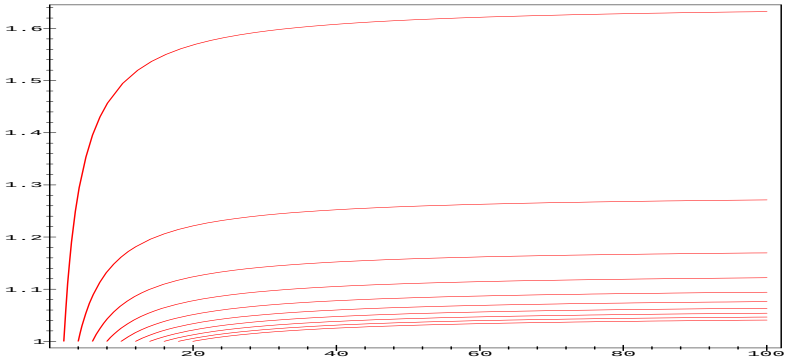


FIG. 1.2 – Trajectoires dans $t \times x$ des solutions de $\dot{x} = x/t$, avec x_0 successivement égal au nombres pairs inférieurs ou égaux à 20 (10 trajectoires), et $x_0 = 1$.

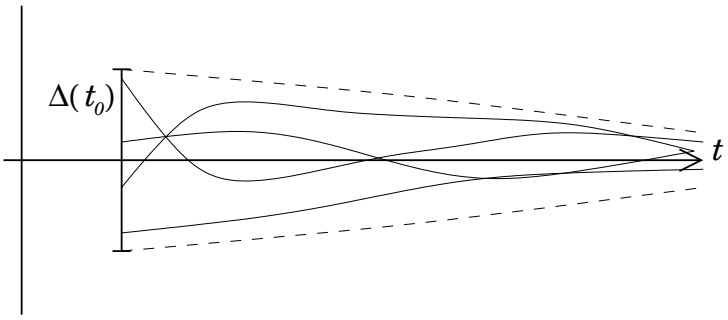


FIG. 1.3 – Illustration de la stabilité uniforme.

qui a pour solution $x = x_0 \exp(1/t_0 - 1/t)$ (figure 1.2), **n'est pas uniformément stable mais est lyapunov-stable** ($\delta(\varepsilon, t_0) = \varepsilon \exp(-1/t_0)$).

Définition 4 (Stabilité asymptotique) *La solution $x(t) \equiv 0$ de (1.1) est dite* **asymptotiquement stable** *si et seulement si :*

1. *elle est lyapunov-stable*
2. *quel que soit t_0 strictement positif, il existe $\Delta(t_0)$ strictement positif tel que :*
$$||x|| \leq \Delta(t_0) \Rightarrow \lim_{t \to \infty} \zeta_t(t_0, x_0) = 0$$

Le **domaine d'attraction** de $x(t) \equiv 0$ est l'ensemble des x_0 tels que :

$$\lim_{t \to \infty} \zeta_t(t_0, x_0) = 0$$

Définition 5 (Fonction de Lyapunov) *Une fonction à valeurs réelles positives, continuement différentiable $V(t, x)$ telle que $V(t, 0) \equiv 0$ est dite* **fonction de Lyapunov**.

La quantité :

$$\dot{V}(t, x) = \frac{\partial V}{\partial t} + \langle \nabla_x V, F \rangle$$

où F est la fonction de (1.1) est appellée **dérivée de V le long des trajectoires de F**. On reconnaît la formulation de la dérivée de Lie de V le long des solutions de (1.1).

Théorème 1 (Premier théorème de Lyapunov) *Si il existe une fonction de Lyapunov et une fonction ω de $\mathbb{R}_+$ dans $\mathbb{R}_+$ telle que $\omega(0) = 0$ et (sauf en $x = 0$) :*

$$\omega(||x||) < V(t, x) \text{ et } \dot{V}(t, x) \leq 0$$

(V est définie positive et $\dot{V}$ est semi-définie négative).
Alors la solution $x(t) \equiv 0$ de (1.1) est lyapunov-stable.

Preuve, cf Afanasev [Afanasev *et al.* , 1996] et [Lyapunov, 1947], [LaSalle, 1961].

Si il existe **en outre** une fonction ω' de $\mathbb{R}_+$ dans $\mathbb{R}_+$ telle que :

$$V(t,x) \leq \omega'(||x||)$$

Alors (1.1) est uniformément stable par rapport au temps. (cf [Sontag, 1990a]).

Théorème 2 (Deuxième théorème de Lyapunov) *Si il existe une fonction de Lyapunov V, et trois fonctions ω_1, ω_2 et ω_3 de $\mathbb{R}_+$ dans $\mathbb{R}_+$ telles que :*

$$\begin{array}{rcccl} \omega_1(||x||) & \leq & V(t,x) & \leq & \omega_2(||x||) \\ & & \dot{V}(t,x) & \leq & -\omega_3(||x||) \end{array} \tag{1.3}$$

Alors les trajectoires du système (1.1) sont uniformément asymptotiquement stables.

La fonction V peut être interprétée comme une **fonction d'énergie** ; le second théorème de Lyapunov est extrêmement utile pour démontrer qu'un système est stable.

1.3 Système dynamique contrôlé

1.3.1 Problématique

La théorie du contrôle est l'étude des trajectoires d'un objet dont la position au cours du temps est déterminée par un système d'équations différentielles ou aux différences (on parle alors de système à temps continu ou à temps discret), stochastique[1] ou non.

Ce système d'équations a en outre la particularité d'être paramétré par un ensemble de "variables de contrôle". L'espace dans lequel évolue l'objet étudié est appelé "espace d'état" (ici noté $\mathcal{X}$) et celui des variables de contrôle "espace de contrôle" (ici noté $\mathfrak{U}$).

Exemple 1.3.1. *Dynamique d'un couple de points du plan.*

Un exemple très simple est un couple de points (x, y) et (X, Y) d'un plan cartésien qui s'y meuvent sans frottement et auxquels des forces (u, v) et

[1] Pour le contrôle stochastique, se référer par exemple à [Levada, 1998].

(U, V) sont appliquées. Les forces sont les variables de contrôle des trajectoires des points (la notation $\dot{x}$ correspond à la dérivée de la variable x par rapport au temps) :

$$(1.4) \qquad \text{point 1} : \begin{cases} \dot{x} &= x' \\ \dot{x}' &= u \\ \dot{y} &= y' \\ \dot{y}' &= v \end{cases} \qquad \text{point 2} : \begin{cases} \dot{X} &= X' \\ \dot{X}' &= U \\ \dot{Y} &= Y' \\ \dot{Y}' &= V \end{cases}$$

Ici $\mathcal{X} = I\!R^8$, $\mathfrak{U} = I\!R^4$, (x, y) sont les coordonnées cartésiennes du premier point ((X, Y) celles du second point), (u, v) est la force appliquée sur le premier point ((U, V) sur le second).

Par exemple, en prenant :

$$(1.5) \qquad \begin{cases} u = -U &= \sin(t/\pi) \\ v = -V &= \cos(t/\pi) \end{cases}$$

La figure 1.4 montre la trajectoire partant du point : $(x_0, y_0, X_0, Y_0) = (-1, 0, 0, 0, 1, 0, 0, 0)$.

Au delà de l'observation de ce genre de systèmes et de leur étude comportementale, la théorie du contrôle s'intéresse à la relation entre le choix des valeurs des variables de contrôle au cours du temps (ici (u, v, U, V)) et les trajectoires du système.

Si on a comme objectif de faire atteindre aux trajectoires de (1.4) le lieu $(0, 0)$, il est par exemple possible d'appliquer "naïvement" les contrôles suivants : $u = -\frac{1}{10}x$ et $v = -\frac{1}{10}y$. Les trajectoires ne seront alors pas celles désirées (figure 1.5).

En y ajoutant une composant transverse :

$$(1.6) \qquad \begin{cases} u &= -\frac{1}{10}x - \frac{1}{20}y \\ v &= -\frac{1}{10}x - \frac{1}{20}y \end{cases}$$

les trajectoires sont alors beaucoup plus "mélangées" (figure 1.6).

Pour mieux observer ces trajectoires (qui sont en dimension 4) on peut choisir d'effectuer une analyse en composante principales pour ne faire figurer que les trois axes les plus explicatifs (figure 1.7). Ces couples de valeurs

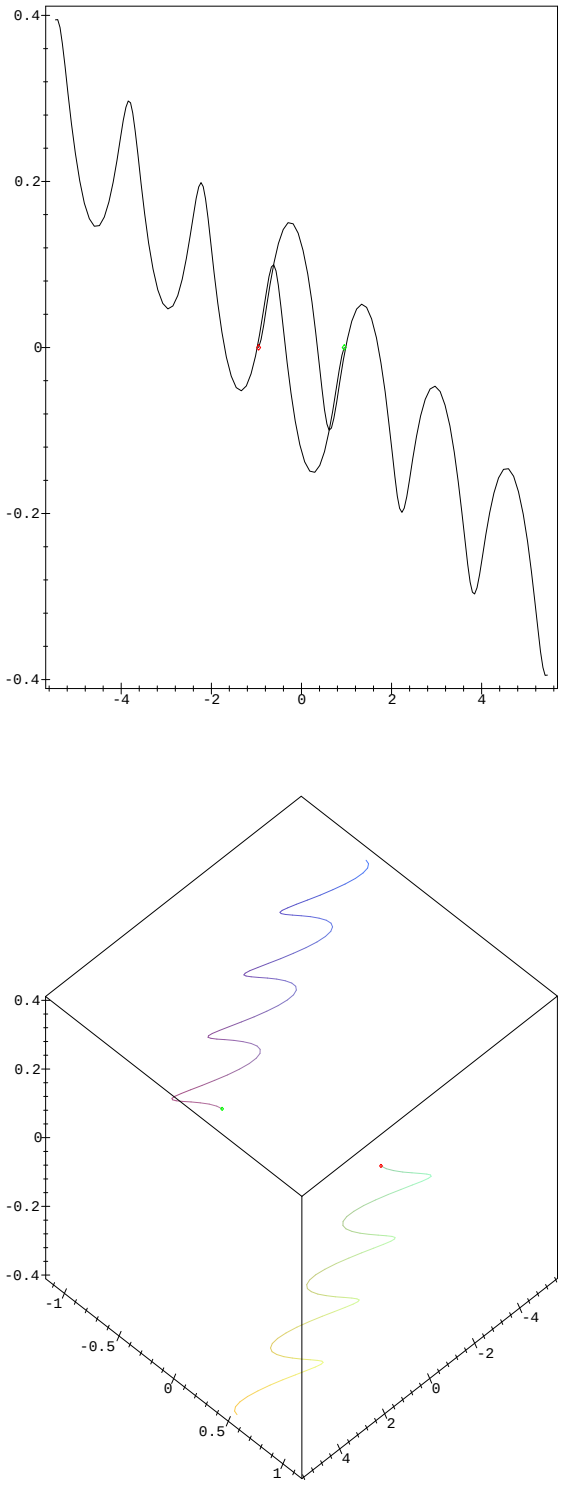


FIG. 1.4 – Une trajectoire de (1.4) pour le contrôle (1.5) dans le plan des positions et dans l'espace constitué des positions et de la vitesse horizontale.

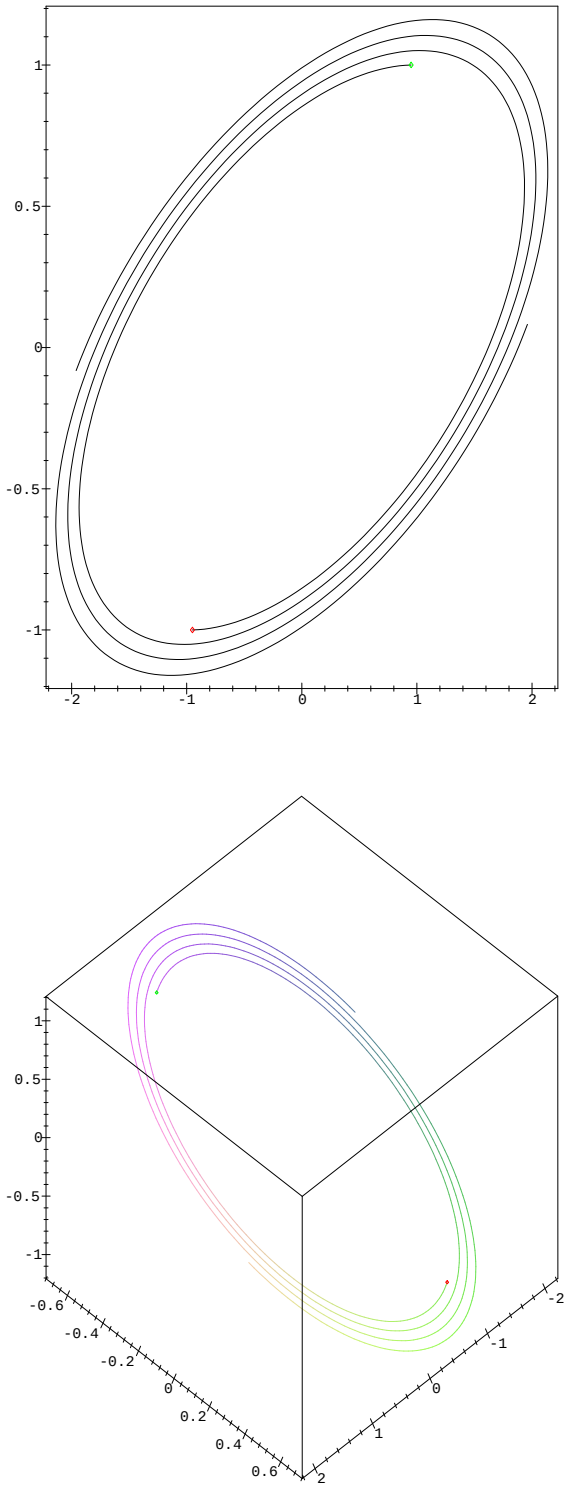


FIG. 1.5 – Une trajectoire de (1.4) pour le contrôle $u = -\frac{1}{10}x$ et $v = -\frac{1}{10}y$ dans le plan des positions et dans l'espace constitué des positions et de la vitesse.

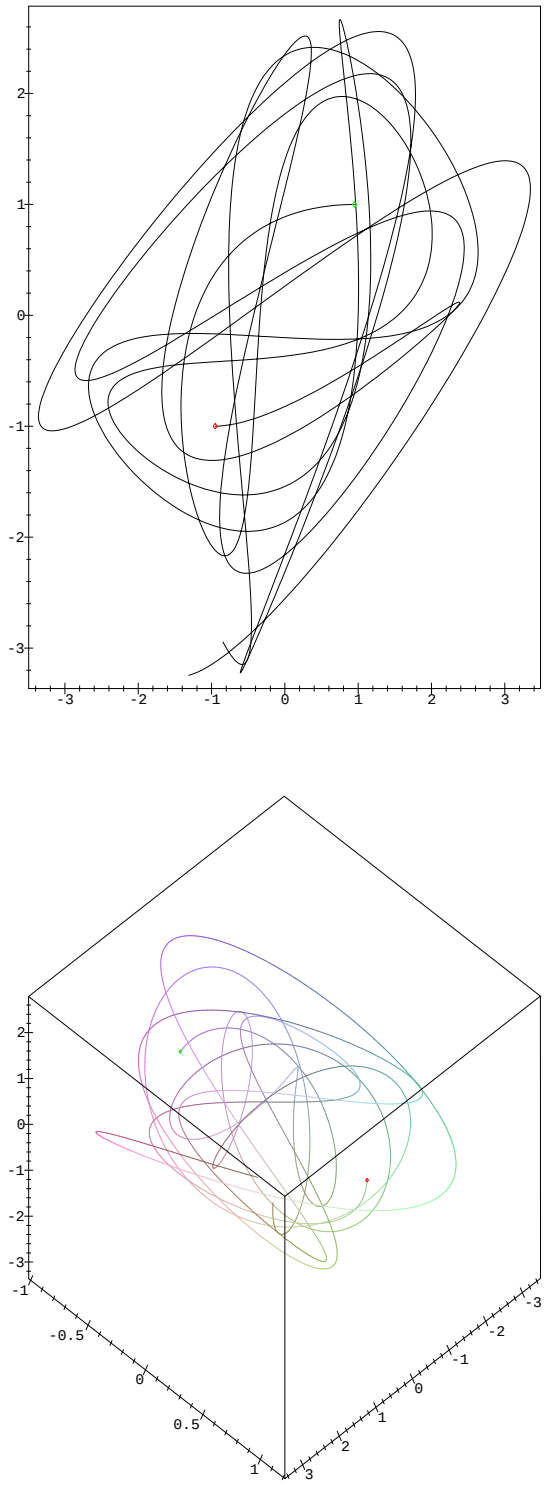


FIG. 1.6 – Une trajectoire de (1.4) pour le contrôle (1.6) dans le plan des positions et dans l'espace constitué des positions et de la vitesse.

propres - vecteurs propres sont :

$$\begin{aligned}(\lambda_1, v_1) &= 25.18653266,\\ &\quad (0.9485333583, 0.01844895912, 0.3035996626, 0.08815525848)\\ (\lambda_2, v_3) &= 0.001094294776,\\ &\quad (-0.2886635095, 0.4608144757, 0.8080783184, 0.2265674915)\\ (\lambda_3, v_4) &= 0.0000003713961,\\ &\quad (0.08369546674, 0.5557366350, -0.05563094508, -0.8252618127)\end{aligned}$$

La diversité des trajectoires associées à ce simple exemple montre combien le cadre général demande de résultats[2].

◇ *Fin de l'exemple 1.3.1*

Ce mémoire se place dans le cadre suivant :

- le contrôle est en boucle fermée : la variable u de l'équation $\dot{x} = f(x, u)$ dépend de la valeur de x,
- cette boucle fermée est réalisée par les fonctions de la classe des perceptrons affines par morceaux ($u = \Psi_\Theta(x)$ où Θ est un ensemble de paramètre qui défini la forme de la fonction Ψ),
- nous allons chercher les éléments de la classe de fonctions choisie (i.e. les valeurs des paramètres Θ) qui garantissent certaines propriétés sur les trajectoires du système contrôlé (en notant $\mathcal{X}$ l'espace d'état dans lequel vivent les réalisations de x, l'ensemble $\mathbf{X}$ de ces trajectoires est constitué des éléments de $\mathcal{X}^{\mathbb{R}}$ dans le cas continu et de $\mathcal{X}^{\mathbb{N}}$ dans le cas discret),
- les propriétés des trajectoires recherchées vont s'exprimer comme des parties de $\mathbf{X}$, qui seront définies en utilisant une fonction $\mathcal{C}$ de $\mathbf{X}$ dans $\mathbb{R}^+$ (dite *fonction de coût*) ; ainsi le sous-ensemble de $\mathbf{X}$ qui nous intéresse est celui qui minimise $\mathcal{C}$ sur $\mathbf{X}$, ce qui nous place dans le cadre du *contrôle optimal*.

Ces éléments sont précisés dans la section 2.1.3 en ce qui concerne le contrôle et dans le chapitre 5.1 en ce qui concerne la classe des réseaux de neurones.

1.3.2 Contrôle non linéaire

Plus formellement, la théorie du contrôle non linéaire regroupe les méthodes et résultats mathématiques permettant d'étudier l'évolution au cours

[2] Une étude de la complexité de ces trajectoire via un exemple a priori simple est fournie dans [Libre & Ponce, 1996].

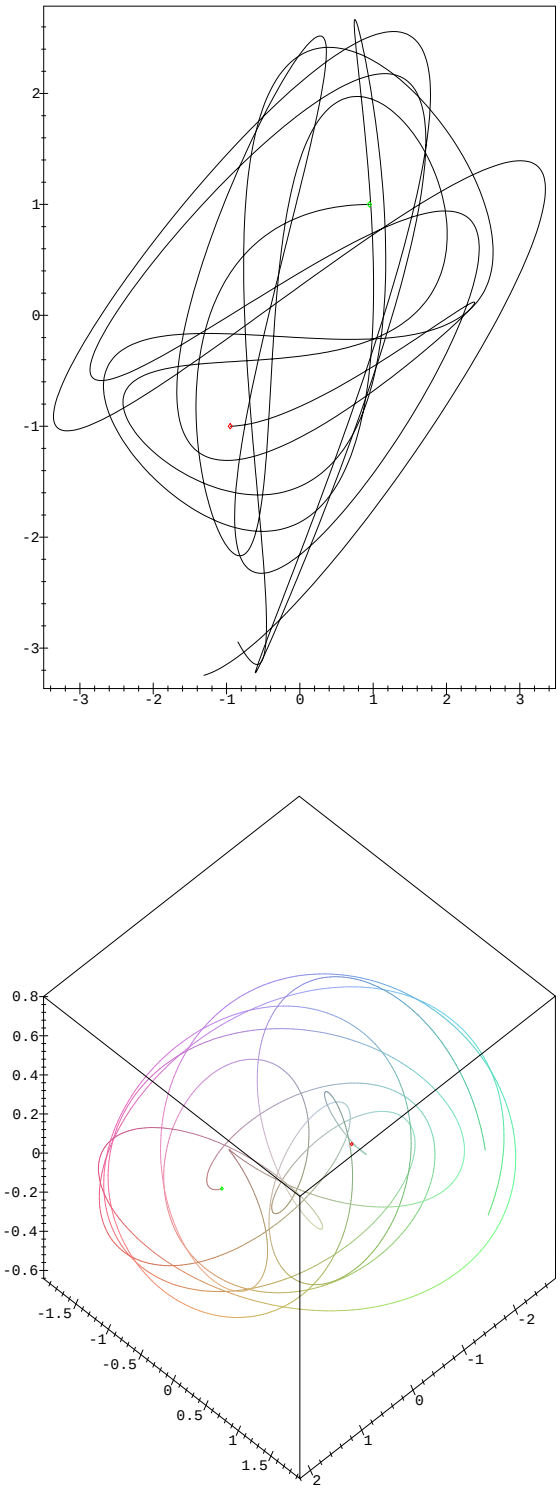


FIG. 1.7 – Une trajectoire de (1.4) pour le contrôle (1.6) dans le plan des positions et dans l'espace constitué des trois axes les plus explicatifs parmi les 4 existants.

du temps de processus dont l'état est à la position x_t de l'espace d'état (ou espace des phases) $\mathcal{X}$ à l'instant t, et dont les transitions sont gouvernées par une fonction qui dépend du temps et qui est paramétrée par un *contrôle* u_t (qui appartient à l'espace des contrôles admissibles $\mathcal{U}$).

Définition 6 (Système contrôlé déterministe continu ou discret) *Lorsque $\mathfrak{T} = \mathbb{R}$ que $\mathcal{X}$ est muni d'une structure de variété différentiable et qu'il existe une fonction différentiable de cet espace telle que les évolutions de x soient gouvernées par :*

$$\dot{x} = f(x, u, t) \tag{1.7}$$

alors on dit que le système contrôlé est continu. On définie en outre ξ une partie de $\mathcal{X}$ dont sont issus les états initiaux du systèmes (si on ne précise par ξ c'est qu'il est égal à $\mathcal{X}$).

Lorsque $\mathcal{X}$ ne peut pas être munie d'une structure de variété différentielle, alors on formalise l'évolution du processus par l'équation aux différences suivante :

$$x_{n+1} = f(x_n, u_n, n) \tag{1.8}$$

on définie aussi ξ comme le lieu des états initiaux du système.

Dans le cas de l'équation aux différences, l'opérateur d'avance σ est parfois utilisé ($\sigma x_n = x_{n+1}$), il permet d'écrire (1.8) sous la forme :

$$\sigma x = f(x, u, n)$$

D'autre part, il existe des versions stochastiques des équations (1.8) et (1.7).

Comme l'introduction l'a souligné, la théorie du contrôle se ramifie en de nombreuses branches traitant chacune d'un aspect bien spécifique de l'évolution des systèmes (1.7) et (1.8). En voici une brève description tirée principalement de [Sussmann, 1989], [Bellman, 1961] et [Aubin *et al.* , 1980] :

- **La contrôlabilité, l'observabilité, et la réalisation minimale** de systèmes non linéaires [Coron, 1994]. Il s'agit de l'étude des relations entre les espaces $\mathcal{X}$ et $\mathfrak{U}$: quelles parties de $\mathcal{X}$ sont parcourues par le système lorsque le contrôle prend ses valeurs dans une partie de $\mathfrak{U}$? et réciproquement ?
- **La recherche de chemins minimaux**. L'objectif est dans ce cas de déterminer quelles valeurs de x_0 et de u au cours du temps permettent de minimiser une fonction de coût calculée le long des trajectoires de (1.7) ou (1.8) [Sussmann, 1989].

- **La stabilisation par bouclage**, qui est l'étude des fonctions u de la forme $g(x,t)$ (ou $g(x_n,n)$) permettant qu'un point donné de $\mathcal{X}$ soit un équilibre de (1.7) ou (1.8). [Sontag, 1990b]
- **L'équivalence de systèmes** par difféomorphismes et par bouclage. Qui consiste en l'étude des similarités entre systèmes de type (1.7) ou (1.8) modulo certains changements de variables [Celikovsky, 1995].
- **Les systèmes hybrides**, qui sont une collection de systèmes de type (1.7) et (1.8) reliés entre eux par des conditions de passage. Ce type de système est particulièrement adapté à la modélisation de programmes informatiques à base d'agents.
- **Le contrôle optimal** traite des méthodes permettant d'assurer que l'ensemble des trajectoires partant d'une zone de $\mathcal{X}$ minimise une fonction de coût pendant un intervalle de temps fixé. [Sussmann, 1998]

1.3.3 Stabilité et robustesse

Une des principales questions soulevées par l'application de la théorie du contrôle à des cas réels est que le système (1.8) est une approximation d'un système réel à contrôler. En d'autres termes on est alors dans une situation où la fonction $f(x,u)$ de (1.8) est une approximation d'une fonction $\mathcal{F}(x,u)$ avec une relation du type :

$$\int_{x\in\mathfrak{X}}\int_{u\in\mathfrak{U}} ||f(x,u)-\mathcal{F}(x,u)||^2\,dxdu < \varepsilon$$

ou :

$$||f(x,u)-\mathcal{F}(x,u)|| < \varepsilon,\ \forall x\in\mathfrak{X}, \forall u\in\mathfrak{U}$$

Cette relation ne donne que peu de garanties sur le comportement du système dynamique correspondant puisque l'on a au mieux [cf Sontag p347], lorsque le même contrôle est appliqué à f et $\mathcal{F}$ (en notant ζ une trajectoire associée au système issu de f et ξ à celui issu de $\mathcal{F}$) et lorsque $f(x,g(x))$ est lipschitzienne (ie : il existe k_t tel que $||f(x,g(x))-f(y,g(y))|| \le k_t\,||x-y||$) :

$$||\zeta(t)-\xi(t)|| \le ||\zeta_0-\xi_0|| + \int_{t_0}^{t} k_\tau\cdot||\zeta(\tau)-\xi(\tau)||\,d\tau + \int_{t_0}^{t}\varepsilon(\tau)d\tau$$

pour tout t (cf [Sontag, 1990a] p347).

Ce qui est une majoration assez brutale de $\xi(t)$, trajectoire du système réel contrôlé par g construite à partir de f. Certaines informations supplémentaires sur le système permettent néanmoins de conclure.

Par exemple le système très simple :

$$\dot{x} = x + u^3,\ x \in I\!R,\ u \in I\!R \tag{1.9}$$

peut être stabilisé vers zéro par le contrôle :

$$u = -(2|x|)^{\frac{1}{3}} \tag{1.10}$$

Le système contrôlé $\dot{x} = -x$ tends vers zéro, **même lorsqu'il est soumis à de petites perturbations** : en effet, si $f(x,u) = x+u^3$ est remplacée par $\mathcal{F}(x,u) = x + u^3 + o(x)$ (où $o(x)$ tends vers 0 lorsque x tends vers zéro), u défini par (1.10) est encore un contrôle efficace.

Dans la mesure où le système engendré $f(\cdot, g(\cdot))$ est convergent et asymptotiquement stable, la question est donc ici de savoir le comportement de celui engendré par $\mathcal{F}(\cdot, g(\cdot))$.

1.3.4 Méthode de Lyapunov

Le choix d'une fonction de contrôle $u = g(x,t)$ ramène l'étude d'un système de type (1.7) ou (1.8) à celle d'un système dynamique non contrôlé, de la forme :

$$\dot{x} = f(x, g(x,t), t)$$

ou

$$\sigma x = f(x, g(x,n), n)$$

Les quelques notions et notations suivantes sont très utiles pour l'étude de ce genre de systèmes.

Le système étudié est alors celui de l'équation (1.1) :

$$\dot{x} = f(t,x)$$

On peut alors utiliser les résultats exposés dans la section 1.2, le second théorème de Lyapunov est extrêmement utile pour démontrer qu'un système contrôlé est stable. Il est même possible de l'utiliser pour **construire** un contrôle stable.

En effet, si on construit une fonction de Lyapunov $V(t,x)$ telle que la condition (1.3) soit vérifiée, et en écrivant explicitement sa dérivée le long des trajectoires du système à contrôler où le contrôle u est remplacé par

une fonction paramétrée par un ensembles de variables Ω, la condition (1.3) donne des inégalités sur les Ω. Si ces inégalités sont vérifiées, alors le contrôle sera stable.

Pour ce faire, on retient une famille paramétrée de fonction de Lyapunov, les inégalités (1.3) donnent alors des contraintes sur les paramètres et les contrôles qui permettent de conclure à la stabilité du système. La vérification de telles contraintes définie explicitement les contrôles en fonctions des variables d'état : cela permet de définir un contrôle stable.

Bien sûr cela n'est pas toujours possible, et le choix de la famille paramétrée de fonctions "candidates" à vérifier les conditions de Lyapunov est rarement simple.

Voici, dans ces grandes lignes, une illustration de cette méthodologie appliquée à l'exemple (1.4) du début de cette partie :

Exemple 1.3.4. *Utilisation de fonctions de Lyapunov pour un contrôle de trajectoire.*

Il s'agit de déterminer les forces à appliquer aux deux masses ponctuelles se déplaçant sur un plan[3] ; exemple (1.4). Les deux masses ont pour coordonnées respectives (x, y) et (X, Y). Les contrôles sont les vecteurs (u, v) et (U, V) :

$$\text{point 1 :} \left\{ \begin{array}{rcl} \dot{x} & = & x' \\ \dot{x}' & = & u \\ \dot{y} & = & y' \\ \dot{y}' & = & v \end{array} \right. \qquad \text{point 2 :} \left\{ \begin{array}{rcl} \dot{X} & = & X' \\ \dot{X}' & = & U \\ \dot{Y} & = & Y' \\ \dot{Y}' & = & V \end{array} \right.$$

L'objectif est de les amener respectivement dans deux régions circulaires T_1 et T_2 du plan des positions par :

$$\begin{array}{rcl} T_1 & = & \left\{(a,b) \in I\!R^2 \,/\, (a-c_{11})^2 + (b-c_{12})^2 \leq r_1^2\right\} \\ T_2 & = & \left\{(a,b) \in I\!R^2 \,/\, (a-c_{21})^2 + (b-c_{22})^2 \leq r_2^2\right\} \end{array} \tag{1.11}$$

tout en évitant les régions AT_1 et AT_2 (elles aussi du plan des positions) qui dépendent des positions des points :

$$\begin{array}{l} AT_1(t) = \left\{(a,b) \in I\!R^2 \,/\, (a-X(t))^2 + (b-Y(t))^2 \leq r_3^2\right\} \\ AT_2(t) = \left\{(a,b) \in I\!R^2 \,/\, (a-x(t))^2 + (b-y(t))^2 \leq r_4^2\right\} \end{array} \tag{1.12}$$

[3]Cet exemple est une refonte de celui de Vanualaila et Nakagiri [Vanualaila *et al.* , 1998].

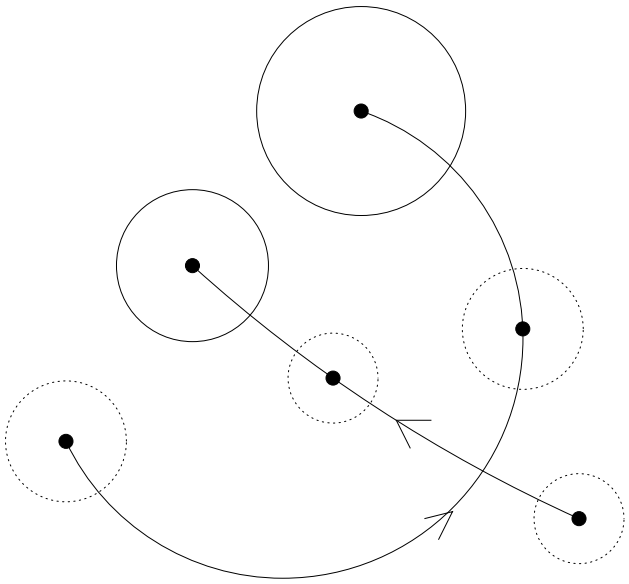


FIG. 1.8 – Idée de la situation dans le plan des positions : Les deux zones d'objectif sont en traits pleins, les trajectoires sont figurées par des flêches, autour de chaque point figure une "zone de sécurité" en pointillés.

ainsi si les deux masses (x, y) et (X, Y) évitent les régions $AT_1(t)$ et $AT_2(t)$ à chaque instant, elles ne se "percuteront" pas (confère figure 1.8).

En construisant la fonction d'énergie $L(x, y, X, Y)$ à l'aide des fonctions intermédiaires suivantes :

– l'énergie $\mathfrak{V}_0$ est construite à partir de T_1 par :

$$\mathfrak{V}_0(x, y) = \frac{1}{2}\left((x - c_{11})^2 + (y - c_{12})^2 + \dot{x}^2 + \dot{y}^2\right)$$

(plus $\mathfrak{V}_0$ est petit et plus (x, y) est proche de T_1 avec une vitesse petite).

– l'énergie $\mathfrak{V}_1$ est construite à partir de T_2 par :

$$\mathfrak{V}_1(x, y) = \frac{1}{2}\left((x - c_{21})^2 + (y - c_{22})^2 - r_2^2\right)$$

($\mathfrak{V}_1$ devient négative lorsque (x, y) est dans T_2).

– l'énergie $\mathfrak{V}_2$ est construite à partir de AT_1 :

$$\mathfrak{V}_2(x, y, X, Y) = \frac{1}{2}\left((x - X)^2 + (y - Y)^2 - r_3^2\right)$$

($\mathfrak{V}_2$ est négative lorsque (x, y) est dans un rayon r_3 de (X, Y)).

– l'énergie F qui s'annule lorsque l'objectif est atteind :

$$F(x,y) = \frac{1}{2}\left((x-c_{11})^2 + (y-c_{12})^2\right)$$

– en utilisant deux constantes **strictement positives** β_{11} et β_{12} on construit alors :

$$\mathfrak{V}(x,y,X,Y) = \mathfrak{V}_0 + F \cdot \sum_{k=1,2} \frac{\beta_{1k}}{\mathfrak{V}_k}$$

qui est définie, continue et positive sur :

$$D(\mathfrak{V}) = \left\{(x,y,X,Y) \in I\!R^4 \,/\, \mathfrak{V}_1 > 0 \text{ et } \mathfrak{V}_2 > 0\right\}$$

– respectivement pour la masse de coordonnées (X,Y) :

$$\mathfrak{W}(X,Y,x,y) = \mathfrak{W}_0 + G \cdot \sum_{k=1,2} \frac{\beta_{2k}}{\mathfrak{W}_k}$$

Finalement :

$$L(x,y,X,Y) = \mathfrak{V}(x,y,X,Y) + \mathfrak{W}(X,Y,x,y)$$

On constate que si il existe $\gamma_{11}, \gamma_{14}, \gamma_{22}$ et γ_{24} positifs strictement tels que :

$$\left\{ \begin{array}{lcl|lcl} u+f_2 & = & -\gamma_{12}\cdot\dot{x} & v+f_4 & = & -\gamma_{14}\dot{y} \\ U+g_2 & = & -\gamma_{22}\cdot\dot{X} & V+g_4 & = & -\gamma_{24}\dot{Y} \end{array} \right. \tag{1.13}$$

alors L sera une fonction de Lyapunov strictement décroissante sur les trajectoires de système formé des deux masses ponctuelles.

La condition (1.13) déterminée à partir d'une fonction d'énergie "plausible" L découlant des contraintes du programme de contrôle défini par $(T_1, T_2, AT_1, AT_2, \text{etc})$, permet donc de déterminer des valeurs pour (u,v,U,V) qui garantissent le respect de ces contraintes.

Les figures 1.9 et 1.10 sont issues de cet exemple avec les valeurs suivantes pour les paramètres :

$$\left\{ \begin{array}{lll} c_{11}=\frac{1}{2}, & c_{12}=\frac{1}{2}, & c_{21}=0 \\ c_{22}=0, & r_1=\frac{1}{3}, & r_2=\frac{1}{3} \\ r_3=\frac{1}{3} & & \end{array} \right. \text{ et } \left\{ \begin{array}{lll} \beta_{11}=\frac{1}{2}, & \beta_{12}=\frac{1}{2}, & \beta_{21}=\frac{1}{3} \\ \beta_{22}=\frac{1}{2}, & \gamma_{12}=1, & \gamma_{14}=1 \\ \gamma_{22}=1, & \gamma_{24}=1 & \end{array} \right.$$

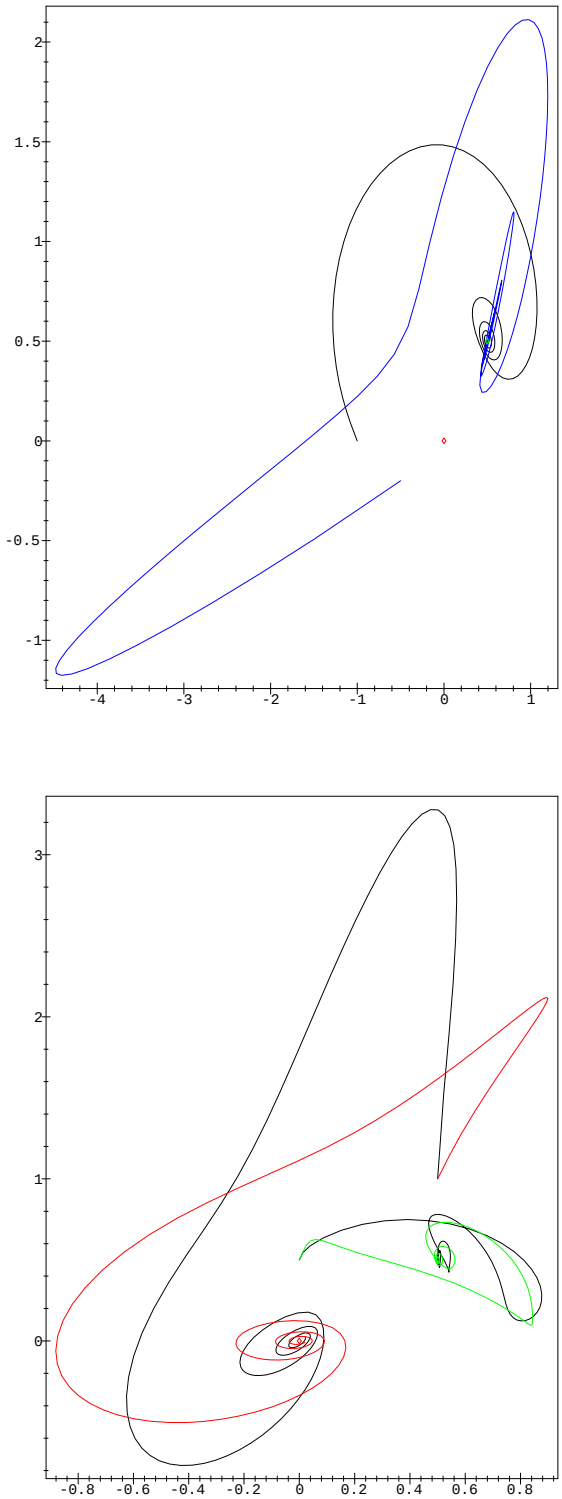


FIG. 1.9 – Deux trajectoires superposées dans le cas d'un seul point (à gauche) et de deux (à droite).

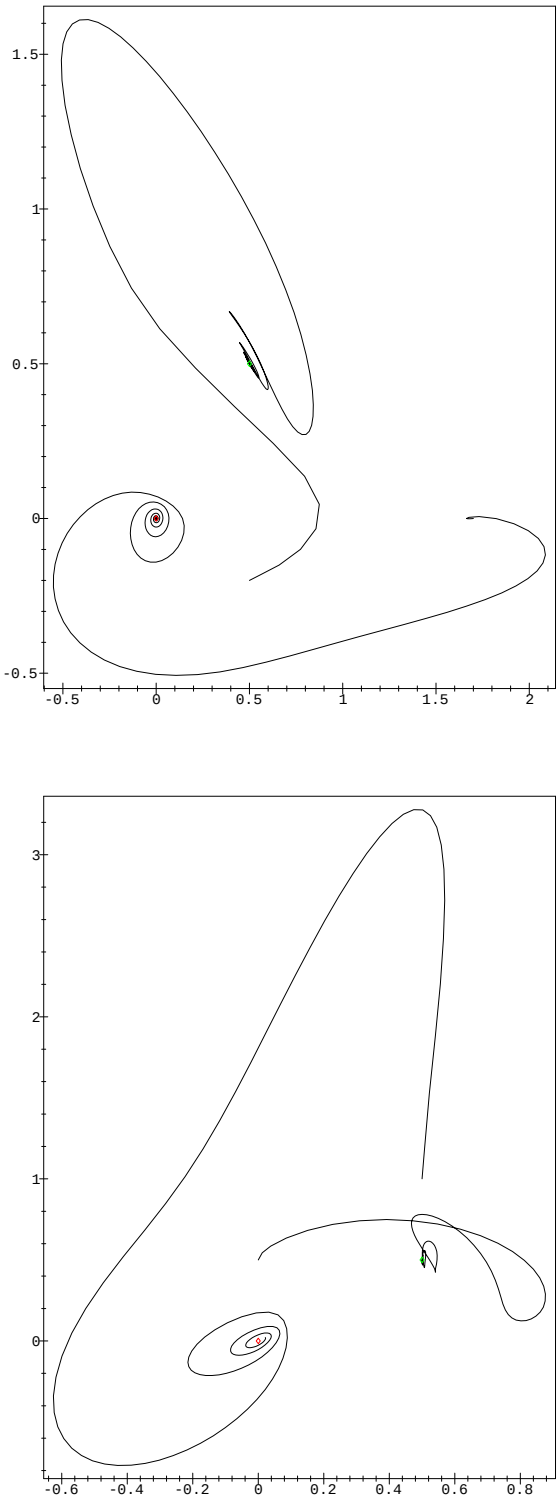


FIG. 1.10 – Diverses trajectoires de deux points.

◇ . . . . . *Fin de l'exemple 1.3.4*

Bien sûr, cette méthodologie fait dépendre le contrôle de la fonction de Lyapunov choisie et de la forme de la fonction paramétrée u. Et d'autre part, l'absence de solution aux inégalités issues de (1.3) ne signifie pas qu'il n'existe pas de contrôle stable.

Il existe des versions de ces théorèmes pour les systèmes à temps discret [Lino, 1995], [Clarke *et al.* , 1999], [Fomin, 1991], confère surtout Blanchini [Blanchini, 1995].

Dans la suite de ce mémoire, une classe spécifique de fonctions de Lyapunov sera utilisée ; il s'agit des **fonctions de Lyapunov quadratiques par morceaux**, introduites en 1997 par Johansson et Rantzer [Johansson & Rantzer, 1997] et [Rantzer & Johansson, 1997].

La théorie du contrôle non linéaire regroupe de nombreuses autres méthodes et résultats qui ne sont pas cités ici. Seuls ont été exposés les résultats directement utilisés dans ce mémoire.

Cela ne veut pas dire non plus que les interactions entre les RNF et le contrôle se limitent aux notions présentées. De nombreuses autres approches existent, la plupart d'entre elles sont citées au cours de la partie 4.

Chapitre 2

Contrôle optimal

2.1 Principe du contrôle optimal

Dans [Lions, 1968], J.L. Lions donne la définition suivante du contrôle optimal déterministe :

Définition 7 (Contrôle optimal) *La théorie du contrôle optimal (déterministe) se fait à partir des données suivantes :*

1. **un contrôle** u *"à notre disposition" dans un ensemble* $\mathcal{U}$ *(l'ensemble des "contrôles admissibles") ;*
2. **l'état** $x(u)$ *du système à contrôler qui est donné, pour* u *choisi, par la résolution de l'équation :*

$$\Lambda y(u) = \text{fonction donnée de } u$$

où Λ *est l'opérateur (supposé connu) qui "représente" le système à contrôler (*Λ *est le "modèle" du système) ;*

3. **l'observation** $y(u)$ *qui est une fonction de* $x(u)$ *(supposée connue) ;*
4. **la fonction de coût** $\mathcal{J}(u)$ *("fonction économique") qui est définie à partir d'une fonction* $y \to \mathcal{I}(y)$ *numérique positive sur "l'espace des observations" par :*

$$\mathcal{J}(u) = I(y(u)) \tag{2.1}$$

On cherche (problème de Calcul des Variations) :

$$\inf \mathcal{J}(u), u \in \mathcal{U}$$

Il ajoute que les objectifs de la théorie du contrôle optimal sont l'obtention de conditions nécessaires (ou nécessaires et suffisantes) pour le ou les extrema (ou minima) ; l'étude de la structure et des propriétés des équations exprimant ces conditions (elles font intervenir le "modèle" Λ) ; et l'obtention d'algorithmes constructifs, susceptibles d'applications numériques pour l'approximation du (ou d'un) $u \in \mathcal{U}$ réalisant le inf et qui est alors dit "**contrôle optimal**".

La version stochastique de cette définition donne celle du **contrôle optimal stochastique**.

Il s'agit donc des problèmes de contrôle dont l'objectif est de minimiser un critère le long des trajectoires du système contrôlé. La théorie du contrôle optimale est très riche, l'article [Sussmann, 1989] de Sussman en fait un vaste tour d'horizon en mettant l'accent sur les réponses apportées par la géométrie différentielle.

Le recours aux Réseaux de Neurones Formels rentre assez naturellement dans ce cadre du contrôle optimal puisque ceux-ci peuvent résoudre des problèmes de minimisation (confère partie 3.2.2).

En robotique (particulièrement dans le cadre du "Q-learning" [Dayan, 1995], utilisé notamment pour le contrôle d'agents informatiques) , le contrôle optimal est associé à la notion suivante :

> L'action la plus performante à court terme (à l'instant d'après dans le cas des systèmes discrets, ou après un intervalle de temps δt dans le cas des systèmes continus) n'est pas nécessairement optimale sur le long terme (à l'infini ou après un intervalle de temps très long T).

Ce qui est une interprétation assez naturelle des expressions (2.6) et (2.7). Dans le cadre de la mise au point d'un contrôleur à l'aide d'informations sur le résultat de ses actions, cela implique que corriger une erreur constatée à l'instant t nécessite de revoir toutes les actions de contrôle entre 0 et t, et pas seulement en $t-1$ ou $t-\delta t$.

Comme nous le verrons dans la section 7.3, la satisfaction de ce principe impacte directement l'algorithme d'apprentissage d'un RNF utilisé dans le cadre du contrôle optimal en boucle fermée.

2.1.1 Boucle fermée et boucle ouverte

Lorsque le modèle du système à contrôler n'est pas exact, on a souvent recours à un contrôle en boucle fermée qui peut compenser cet écart lorsque le contrôleur sera utilisé avec le système réel.

Le contrôle en boucle fermée correspond à la notion de "feed-back", largement utilisée par Norbert Wiener [Wiener, 1948] pour introduire les bases de la cybernétique. L'idée sous-jacente est d'utiliser l'information d'écart à la position désirée du système afin de l'y amener.

Le choix des variables à inclure dans le feedback est délicat lorsqu'il s'agit de contrôler un système réel. Il résulte souvent du savoir faire du concepteur du contrôleur et est limité par les mesures disponibles (les dérivées de certaines variables ne le sont pas toujours).

Ce type de contrôle est particulièrement utilisé pour des contrôleurs dont l'esprit est d'**imiter les systèmes biologiques** : il corrige en effet ses actions uniquement à partir d'observations d'écarts (l'écart à la position désirée, l'écart à la vitesse désirée, à la quantité d'énergie désirée, etc). Cette technique de correction des actions à partir de l'erreur commise est d'ailleurs aussi utilisée pour la rétropropagation du gradient dans le cadre de la mise au point des paramètres des réseaux de neurones formels (cf section 4).

Si dans ce mémoire le cas traité explicitement est celui du contrôle en boucle fermée, les réseaux de neurones formels sont aussi utilisé pour effectuer du contrôle en boucle ouverte, via des techniques évoquées section 4.2.2.

2.1.2 Programmation dynamique

Pour résoudre ce type de problématique lorsque la fonction de coût (2.1) est de la forme :

$$\mathcal{J}(t,T,x,u) = \int_t^T q(\tau,x,u)d\tau + p(x(T)),\ t \in [T_0,T] \tag{2.2}$$

et que les trajectoires suivent :

$$\dot{x} = f(t,x,u)$$

on construit la fonction :

$$V(t,x) = \min_u \{\mathcal{J}(t,T,x,u)\} \tag{2.3}$$

Le problème est alors ramené à l'élaboration d'une stratégie de construction de u. L'objectif est de construire le contrôle u qui réalise ce minimum.

Pour cela, on écrit (2.3) sous la forme :

$$V(t,x) = \min_u \left\{ \int_t^T q(\tau, x, u) d\tau + p(x(T)) \right\}$$

lorsque le contrôle u est optimal, on a :

$$V(t,x) = \int_t^T q(\tau, x, u^*) d\tau + p(x(T))$$

(remarquons que nécessairement $V(T,x) = p(x(T))$) et en dérivant chaque membre de cette égalité par rapport à t, on obtient :

$$\partial_t V(t,x) + \partial_x V(t,x) \cdot f(t,x,u^*) = -q(t,x,u^*)$$

V est donc solution de l'équation aux dérivées partielles :

$$\left\{ \begin{array}{lcl} \partial_t V(t,x) & = & -q(t,x,u^*) - \partial_x V(t,x) \cdot f(t,x,u^*) \\ V(T,x) & = & p(x(T)) \end{array} \right.$$

dont une autre formulation est (cf Sontag [Sontag, 1998b]) :

$$\partial_t V(t,x) = -\min_u \left\{ q(t,x,u) + \partial_x V(t,x) \cdot f(t,x,u) \right\}, \; V(T,x) = p(x(T))$$

qui est appelée équation de **Hamilton-Jacobi-Bellman** (HJB). Sa version à temps discret est l'équation de **programmation dynamique** :

$$V(t,x) = \min_u \left\{ q(t,x,u) + V(t+1, f(t,x,u)) \right\}, \; V(T,x) = p(x(T))$$

Afin d'appréhender au mieux le cas où (HJB) a une solution unique, on introduit le **hamiltonien** associé au problème de contrôle :

$$\mathcal{H}(t,x,u,\lambda) = q(t,x,u) + \langle \lambda, f(t,x,u) \rangle$$

où λ est dans $\mathbb{R}^n$, $n = \dim \mathcal{X}$.

La solution recherchée (lorsqu'elle existe) est donc $\operatorname{argmin}_u \mathcal{H}(t,x,u,\lambda)$ qui s'exprime en fonction des données du problème et de λ. En la notant $u^*(t,x,\lambda)$ on a le résultat suivant :

Théorème 3 *Si u^* existe et que V continuement différentiable existe tel que $u^*(t,x,\partial_x V(t,x))$ soit un contrôle admissible (tel que le système contrôlé admette une solution), alors :*

"V satisfait l'équation de Hamilton-Jacobi-Bellman"

équivaut à :

"u^* est le contrôle optimal du problème dont le coût est $\mathcal{J}$"

Une preuve de ce théorème se trouve par exemple dans Sontag [Sontag, 1998b]. Pour plus de précisions, consulter [Felgenhauer, 1996].

Cette approche permet de construire des fonctions u qui minimisent le critère de coût sur les trajectoires du système. Elle est fréquemment utilisée dans le cas des systèmes linéaires, lorsque l'équation (HJB) est simple à résoudre.

En particulier. Prenons le cas où l'objectif est de minimiser le critère :

$$\mathcal{J}(\tau_0, \tau_1, x, u) = \sum_{k=\tau_0}^{\tau_1} u_k^2 + x_{\tau_1}^2$$

le long des trajectoires du système dynamique (à temps discret) mono dimensionnel :

$$\sigma x = ax + u \tag{2.4}$$

en utilisant le contrôle $u = cx$.

On cherche donc à résoudre :

$$V(t,x) = \min_u \left\{ u^2 + V(t+1, ax+bu) \right\},\ V(\tau_1, x) = x_{\tau_1}^2 \tag{2.5}$$

La minimisation se fait de τ_1 à τ_0, **en partant de la fin de la trajectoire**.

La forme de (2.5) implique que si $V(t+1,x)$ est quadratique en x, $V(t,x)$ l'est aussi, donc on cherche une suite de (π_i) telle que :

$$V(t,x) = \pi_t \cdot x^2$$

ce qui ramène (2.5) à :

$$V(t,x) = \min_u \left\{ u^2 + \pi_{t+1} \cdot (ax+bu)^2 \right\}$$

qui est atteint lorsque :

$$u = \underbrace{-\frac{\pi_{t+1}}{1+\pi_{t+1}} a}_{c}\, x$$

et implique que :

$$\pi_t = \frac{\pi_{t+1}}{1+\pi_{t+1}}$$

2.1.3 Un paradigme de contrôle

Voici les hypothèses qui encadrent les cas traités ici :

Forme de la fonction de contrôle :
Ici la fonction de contrôle u est de la forme $u = g(x)$ où g est prise dans la classe des réseaux de neurones formels (RNF), plus particulièrement des Perceptrons Affines par Morceaux (*Piecewise Affine Perceptron* : PAP, cf section 5). Indépendamment de l'utilisation des réseaux de neurones, cela place la problématique dans la catégorie du **contrôle en boucle fermée**. Il faut noter que l'utilisation de fonctions de la classe des RNF n'est pas très restrictive (confère section 5.2).

Contraintes du contrôle :
L'objectif de contrôle est de minimiser un critère global $\mathcal{I}$ qui dépend de la trajectoire contrôlée (solution de $\dot{x} = f(x, g(x))$ ou de $\sigma x = f(x, g(x))$) et de l'ensemble de départ de x. $\mathcal{I}$ s'exprime comme :

$$\mathcal{I} = \int_{x_0 \in \Omega_0} \sum_{n=0}^{\infty} L_n(x_n, u_n) d\xi(x_0) \tag{2.6}$$

ou pour le cas continu :

$$\mathcal{I} = \int_{x_0 \in \Omega_0} \int_{t=0}^{\infty} L_t(x(t), u(t)) d\xi(x_0) \tag{2.7}$$

Dans les deux cas la fonction L va de $\mathfrak{X} \times \mathfrak{U}$ dans $\mathbb{R}_+$; dans les cas les plus classiques il s'agit d'une forme quadratique ($L(x, u) = x'Qx + u'Ru$).

Cela place la problématique dans le cadre du **contrôle optimal** [Sussmann, 1989], [Kirk, 1970], [Aubin *et al.* , 1980].

Stabilité et robustesse :
La fonction $f(x, u)$ des équations (1.8) ou (1.7) peut être considérée comme l'approximation d'une fonction $\mathcal{F}$ d'un système réel ; nous regarderons comment un contrôle solution d'une approximation f peut être une solution "convenable" pour le système réel $\mathcal{F}$ (c'est-à-dire quelles formes prennent les trajectoires solution lorsque $f(x, u)$ est remplacée par $\mathcal{F}(x, u)$). Cela fait appel à des notions de **stabilité et de robustesse**.

2.2 Contrôle linéaire optimal : un point de vue local

2.2.1 Définitions

La théorie du contrôle linéaire traite des systèmes qui peuvent se formaliser sous une des deux formes suivantes :

$$x_{n+1} = A_n \cdot x_n + B_n \cdot u_n \tag{2.8}$$

pour les systèmes à temps discret, et pour ceux à temps continu :

$$\dot{x} = A_t \cdot x + B_t \cdot u \tag{2.9}$$

Les deux équations sont parfois bruitées de façon additive.

Les variables x et u appartiennent respectivement à des espaces vectoriels $\mathfrak{X}$ et $\mathfrak{U}$ et A et B sont des opérateurs linéaires.

Pour décrire l'objectif de contrôle, on définie un critère de contrôle : $I_t(x, u)$ de $\mathfrak{X} \times \mathcal{U}$ dans $\mathbb{R}^+$ à partir d'une suite de jauges.

Définition 8 (Jauge) *Une jauge est une fonction $g : \mathbb{R}^n \to \mathbb{R}$ qui vérifie pour tout x et y de $\mathbb{R}^n$ et tout λ de $\mathbb{R}^+$:*
i) $g(x) \geq 0$
ii) $g(\lambda x) = \lambda\, g(x)$
iii) $g(x + y) \leq g(x) + g(y)$

Remarque : une norme est une jauge symétrique (telle que $g(x) = g(-x)$).

Définition 9 (Contrôle linéaire optimal) *Lorsque le système (2.8) est muni de la fonction de coût (ou critère de performance) construite à partir d'une suite de jauges $(I_k)_k$:*

$$\mathcal{J}_{K_0,K_1}(u) = \sum_{k=K_0}^{K_1} I_k(x_k, u_k) \tag{2.10}$$

ou que le système (2.9) est muni de :

$$\mathcal{J}_{T_0,T_1}(u) = \int_{t=T_0}^{T_1} I_t(x_t, u_t)dt$$

pour toutes les trajectoires du système (1.8), et que l'on s'attache à choisir le u qui minimise $\mathcal{J}$, on est dans le cadre du **contrôle linéaire optimal**.

2.2.2 Méthode de Riccati

Dans le cas où le système à contrôler peut être ramené à :

$$\dot{x} = A(t,x) + B(t,x) \cdot u$$

et que le coût à minimiser est quadratique (2.2) avec :

$$q(t,x,u) = u' \cdot R(t,x) \cdot u + Q(t,x)$$

où R est $\mathcal{C}^1$ symétrique définie positive et Q est $\mathcal{C}^1$ non négative.

Le hamiltonien associé à ce problème est :

$$\begin{array}{rcl} \mathcal{H}_L(t,x,u,\lambda) & = & u'R(t,x)u + Q(t,x) + \langle \lambda, A(t,x) + B(t,x)u \rangle \\ & = & u'Ru + \lambda' A + \lambda' Bu + Q \end{array}$$

et pour déterminer le contrôle u qui le minimise, on utilise le fait que R soit réelle symétrique définie positive (donc inversible) pour factoriser $\mathcal{H}_L$:

$$\mathcal{H}_L(t,x,u,\lambda) = \left(u + \frac{1}{2}R^{-1}B'\lambda\right)' R \left(u + \frac{1}{2}R^{-1}B'\lambda\right) - \frac{1}{4}\lambda' B R^{-1} B' \lambda + Q$$

donc $u^* = \arg\min_u \mathcal{H}_L = -\frac{1}{2}R^{-1}B'\lambda$.

Si d'autre part on cherche V tel que $\partial_t V = -2x'P(t)$ avec P symétrique définie positive, on obtient le contrôle bouclé suivant :

$$u^*(t,x,\partial_x V) = -R^{-1}B'Px$$

qui est admissible lorsque par exemple :

$$\left\{ \begin{array}{rcl} A(t,x) & = & \mathbf{A}_t \cdot x \\ B(t,x) & = & \mathbf{B}_t \\ R(t,x) & = & \mathbf{R}_t \\ Q(t,x) & = & x' \cdot \mathbf{Q}_t \cdot x \\ p(x) & = & x' \cdot S \cdot x \end{array} \right.$$

En outre, dans ce cas, l'équation (HJB) devient :

$$u'\mathbf{R}u + x'\mathbf{Q}x - 2x'P(\mathbf{A}x + \mathbf{B}u) + x'\dot{P}x = 0,\ P(T) = S \tag{2.11}$$

qui équivaut à :

$$x'\dot{P}x = x'(P\mathbf{B}\mathbf{R}^{(} - 1)\mathbf{B}'P - (P\mathbf{A} + \mathbf{A}'P) - \mathbf{Q})x$$

ce qui donne l'**équation différentielle de Riccati** :

$$\dot{P} = P\mathbf{B}\mathbf{R}^{-1}\mathbf{B}'P - P\mathbf{A} - \mathbf{A}'P - \mathbf{Q},\ P(T) = S$$

et pour le problème à temps discret :

$$\sigma P = \mathbf{A}'(P - P\mathbf{B}(\mathbf{R} + \mathbf{B}'P\mathbf{B})^{-1}\mathbf{B}'P)\mathbf{A} + \mathbf{Q}$$

et le contrôle optimal est alors $u^* = -(\mathbf{R} + \mathbf{B}'P\mathbf{B})^{-1}\mathbf{B}'P\mathbf{A}x$.

Il y a de nombreuses façons d'obtenir ces équations de Riccati ([Willems, 1971], pour une approche de toute la théorie du contrôle linéaire par les équations de Riccati : [Cheremensky & Fomin, 1996]), nommées d'après **Jacopo Francesco Riccati**, mathématicien vénitien [Bittani *et al.* , 1989] qui étudia aux alentours de 1720 les équations différentielles de la forme [Riccati, 1742] :

$$\begin{cases} \dot{x} &= \alpha \cdot x^2 + \beta \cdot t^m \\ \dot{x} &= \alpha \cdot x^2 + \beta \cdot t + \gamma \cdot t^2 \end{cases}$$

2.2.3 Diversité des méthodes

Dans le cadre du contrôle linéaire, diverses approches peuvent être utilisées pour garantir la stabilité de la trajectoire. En utilisant les transformées de Laplace pour les systèmes continus et transformées en z pour les systèmes discrets, on aboutit en effet au concept de placement de pôle (i.e. déterminer les racines des dénominateurs des fractions rationnelles provenant de la transformée du système), ce qui amène aux critères de Nyquist, Hurwitz-Routh pour les systèmes continus et par exemple Schur-Cohn (Schur, 1917, Cohn 1922, Marden 1966), Jury (Jury 1962, Blanchard 1961) ou Bishop (1960) pour les systèmes discrets.

Ces méthodes sont couramment utilisées en automatique. Ce mémoire n'y faisant pas particulièrement référence, seule cette section les évoque[1].

2.3 Contrôle par Intégrateur Linéaire Quadratique (LQI)

2.3.1 Construction d'un LQI

Cette méthode consiste à construire un sur-système de la forme suivante à partir d'un système linéaire discret de type (2.8) :

$$X_{n+1} = \mathbf{A}_n \cdot X_n + \mathbf{B}_n \cdot u \tag{2.12}$$

[1]Pour un panorama simple : [Fomin, 1991], [Azzo, 1988], [Faure, 1986] ou [Arimoto, 1966].

avec :

$$X_n = \begin{pmatrix} x_n - x^* \\ \sum_{k=0}^{n}(x_k - x^*) \\ x^* \end{pmatrix}, \mathbf{A} = \begin{bmatrix} A & 0 & -\mathrm{Id} \\ A & \mathrm{Id} & -\mathrm{Id} \\ 0 & 0 & \mathrm{Id} \end{bmatrix}, \mathbf{B} = \begin{bmatrix} B \\ B \\ 0 \end{bmatrix}$$

avec x^* une constante de $\mathbb{R}^d$ (une cible) et de lui associer le coût suivant :

$$\bar{\mathcal{J}}_K = \sum_{k=0}^{K} X_k' \, \mathbf{Q}_{k-1} \, X_k + u_{k-1}' \, R_{k-1} \, u_{k-1} \tag{2.13}$$

avec :

$$\mathbf{Q} = \begin{bmatrix} Q & 0 & 0 \\ 0 & Q_1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

où Q_1 est une matrice symétrique définie positive que l'on fixe arbitrairement en fonction du problème (elle pondère l'importance que l'on veut donner aux écarts concrets).

L'intérêt de cette méthode est que amener le système (2.12) en $X = 0$ revient à amener le système (1.8) en $x = x^*$. Ainsi il est possible d'utiliser la même matrice K issue de la résolution des équations de Riccati (2.11) pour amener le système (1.8) tour à tour en plusieurs points $x^{*,1}$, $x^{*,2}$, etc (sous la condition bien entendu que le passage d'un $x^{*,n}$ à un autre ne soit pas trop rapide) [Rauch, 1997].

2.3.2 Illustration : contrôle LQI d'un modèle de moteur essence

Un **moteur à combustion à essence** peut, au prix de quelques simplifications, être modélisé par un système dont l'espace d'état a deux dimensions (commande d'air arrivant au moteur, couple moyen instantané) et l'espace de contrôle deux (commande d'air, rendement d'avance à l'allumage) [Rauch, 1996], [Rauch, 1997].

On utilisera un sur-système type LQI et les matrices de coût :

$$Q = 10 \cdot \mathrm{Id},\ Q_1 = \mathcal{D}\mathrm{iag}((10, 100)),\ R = \mathcal{D}\mathrm{iag}((10^5, 10^4))$$

Ce qui donne par linéarisation autour du point d'équilibre :

$$u_0 = \begin{pmatrix} 0 \\ 2.8332 \end{pmatrix},\ x_0 = \begin{pmatrix} 0 \\ 3.8332 \end{pmatrix}$$

Le système :

$$\sigma X = \begin{bmatrix} 0.30020 & 0 & 0 & 0 & -1.0 & 0 \\ 0.30020 & 0 & 0 & 0 & 0 & -1.0 \\ 0.30020 & 0 & 1.0 & 0 & -1.0 & 0 \\ 0.30020 & 0 & 0 & 1.0 & 0 & -1.0 \\ 0 & 0 & 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0 \end{bmatrix} \cdot X + \begin{bmatrix} 0.69981 & 0 \\ 0.69981 & 1.0 \\ 0.69981 & 0 \\ 0.69981 & 1.0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot u$$

on obtient en résolvant numériquement l'équation de Riccati associée :

$$K = - \begin{bmatrix} 0.0074287 & 0 & 0.0095901 & 0.0079365 & -0.0 & -1.0074 \\ 0.034306 & 0 & -0.0084051 & 0.091486 & -1.0 & 0.96569 \end{bmatrix}$$

Le système est alors simulé autour d'un couple point de départ, point de consigne pour donner un ensemble de trajectoires (t_i), la figure 2.1 montre l'évolution du critère I_k en fonction du temps.

2.4 Du linéaire au non linéaire

2.4.1 Stabilité locale et linéarité locale

L'intérêt pour l'étude des systèmes linéaires (ou localement linéarisés) provient du *principe de linéarisation*. Avant de l'exposer, il convient de définir la notion de contrôle asymptotique :

Définition 10 (Système asymptotiquement contrôlable) *Un système est dit asymptotiquement contrôlable lorsque pour tout point x^*, il existe un contrôle tel que le système contrôlé tende à l'infini vers x^*. Il est localement asymptotiquement contrôlable lorsque pour tout voisinage $\mathcal{V}$ de x_0 il existe un voisinage $\mathcal{W}$ de x_0 tel que toute trajectoire contrôlée partant de $\mathcal{W}$ soit asymptotiquement contrôlable vers x_0 sans quitter $\mathcal{V}$.*

Théorème 4 (Principe de linéarisation) *Lorsque (x_0, u_0) est un équilibre pour (1.7) et que (2.15) est asymptotiquement contrôlable alors (1.7) l'est localement.*

Il existe en outre une matrice F telle que :

$$\dot{x} = f(x, u_0 + F \cdot (x - x_0))$$

soit localement asymptotiquement contrôlable.

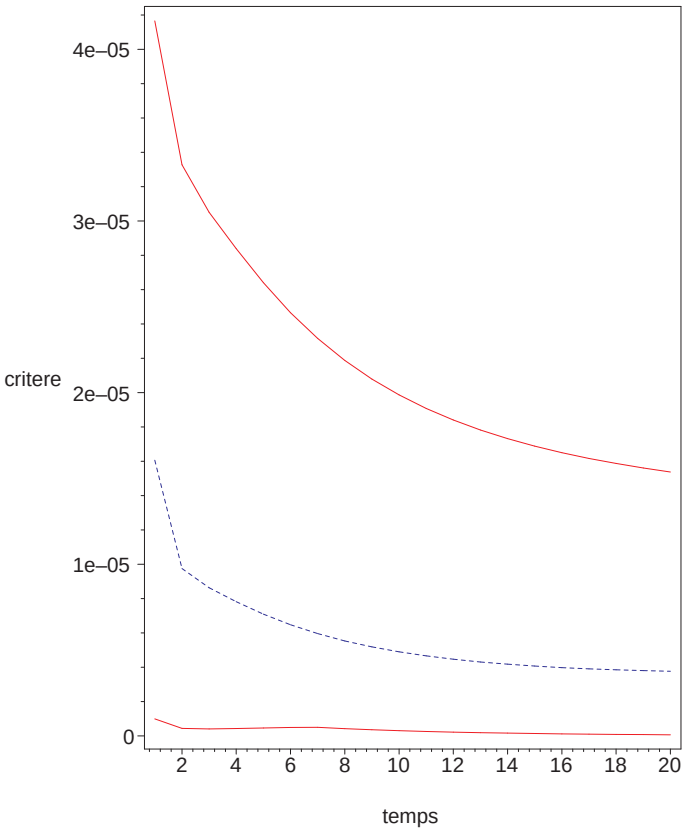


FIG. 2.1 – Cette illustration montre l'évolution de la moyenne, du minimum et du maximum du critère à chaque instant (20 instants en abscisse) calculés sur 150 trajectoires dont les conditions initiales sont différentes.

Une preuve de ce théorème par le théorème de Lyapunov se trouve dans [Sontag, 1990a] ; F est la matrice qui contrôle asymptotiquement le système linéaire :

$$\dot{x} = Ax + BFx$$

Utilisation du principe de linéarisation. A partir d'un système comme (1.7) ($\dot{x} = f(x,u)$) et lorsque f admet un développement de Taylor au point (x_0, u_0) à l'ordre un, on peut écrire :

$$f(x,u) = f(x_0,u_0) + \partial_x f(x_0,u_0)\cdot(x-x_0) + \partial_u f(x_0,u_0)\cdot(u-u_0) + o(||x-x_0||, ||u-u_0||) \tag{2.14}$$

Si on choisit (x_0, u_0) tel que $f(x_0, u_0) = x_0$ (i.e. c'est un équilibre pour (1.7)), et en notant $\delta x = x - x_0$ et $\delta u = u - u_0$, (2.14) devient :

$$\sigma\dot{\delta x} = \partial_x f(x_0,u_0)\cdot\delta x + \partial_u f(x_0,u_0)\cdot\delta u + o(||x-x_0||, ||u-u_0||)$$

si le système original est tel que le terme $o(||x-x_0||, ||u-u_0||)$ peut effectivement être négligé[2], et que c'est l'étude autour de (x_0, u_0) qui nous intéresse, alors on se ramène à l'étude plus simple de :

$$\sigma\delta x = \partial_x f(x_0,u_0)\cdot\delta x + \partial_u f(x_0,u_0)\cdot\delta u \tag{2.15}$$

qui est souvent notée de la façon suivante :

$$\sigma\delta x = A\delta x + B\delta u$$

2.4.2 Les systèmes hybrides et le hard-switching

Le comportement d'un système linéaire pouvant être décrit très simplement [Arnold, 1980] et la construction comme l'étude d'un contrôleur linéaire bouclé associé étant très largement étudiée, l'étude d'un système non linéaire est souvent ramenée à celui de systèmes linéarisés autour de points d'équilibres choisis en fonction du problème sous jacent.

Il s'agit alors d'étudier autour de ces points de référence différents systèmes linéaires. Afin de contrôler le système global, un automate détermine dans quelle zone de l'espace d'état le système se trouve, et lui applique le contrôleur linéaire correspondant (cf Narendra [Narendra, 1994]).

[2] par exemple la présence de bruit (si le système non linéaire initial est bruité : $\dot{x} = f(x,u) + \mathcal{B}$) ou plus simplement s'il s'agit en réalité d'un $o(||x-x_0||, ||u-u_0||)^n$, $n > 1$.

Cette méthodologie parfois appelée du "**hard switching**" est apparentée aux systèmes dits "**hybrides**" (ensemble de systèmes dynamiques continus parmi lesquels on bascule par un système discret) et laisse ouvertes les questions suivantes :

- où positionner exactement les coupures entre les différentes régions de l'espace où le système est linéarisé ? sans une approche critique de ce point, on risque de se trouver dans une situation où le contrôleur est optimal pour chaque zone linéaire, mais où la position de chacune des zones est arbitraire (illustration : figure 2.2).
- quel est le devenir des résultats de stabilité lorsqu'on applique à un système non linéaire un contrôle obtenu à partir de sa linéarisation ?
- d'autre part, cette méthodologie soulève des problèmes de convergence relevant de l'étude des systèmes hybrides (continus **et** discrets). Le système contrôlé est en effet décrit par plusieurs systèmes dynamiques (résultant chacun de l'application d'un contrôleur local au système générique) et par des "règles de transitions" entre eux. L'étude de tels systèmes est à ses débuts et très riche en applications (cf le workshop "systèmes hybrides" du congrès ISIC/ISAS'00).

Nous verrons dans la suite (parties 5, 8 et 6) comment le contrôle par Perceptron Affine Par morceaux apporte une réponse à chacun de ces points.

2.5 Les autres problématiques du contrôle optimal

Dans *Optimal Control* [Sussmann, 1989], H. Sussmann liste les différents champs de la théorie du contrôle :

- la contrôlabilité, l'observabilité et la réalisation minimale de systèmes non linéaires,
- la recherche de chemins minimaux,
- la stabilisation par bouclage,
- l'équivalence de système par difféomorphismes et par bouclage,
- la séparation de sources,
- la poursuite,
- le contrôle optimal.

Les applications des résultats obtenus aujourd'hui sont très vastes ; par exemple :

- le contrôle de trajectoires de véhicules ; par exemple d'avions [Magni & Bennani, 1997] [Calise & Rydyk, 1998],

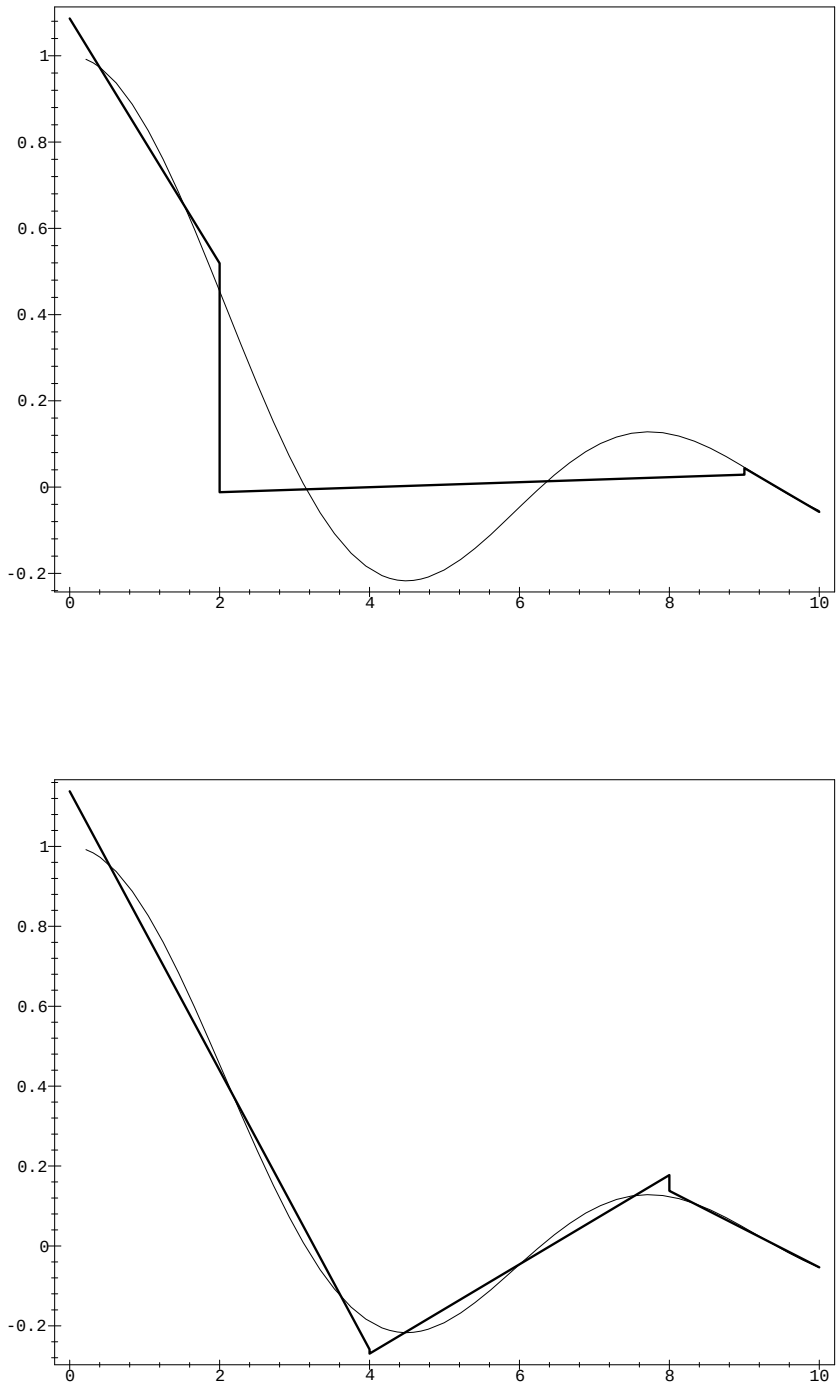


FIG. 2.2 – linéarisation par MCO de la fonction $\frac{\sin x}{x}$: les deux interpolations sont optimales mais dans un cas de découpage est adéquat (en bas) et dans l'autre pas (en haut).

– la robotique [Ram & Santamaria, 1993],
– ou l'utilisation optimale de ressources naturelles. [Clark, 1985].

Pour reprendre les phases du développement de la théorie du contrôle exposées par K.S. Narenda[3] lors du congrès ISIC/ISAS99[4] :

– **Les prémisses** : James Watt pour le contrôle des gouvernails (1788) et D. Papin pour le régulateur de pression (1681) [Papin, 1681].
– **Le contrôle mécanique** : mi-19ème siècle à début 20ème avec Airy (1840) [Airy, 1840] , Maxwell (1868), Viyshnegradsky (1876), Lyapunov (1892), Stodola (1893) [Bissell, 1989], et Hurwitz (1895).
– **Le contrôle des fluctuations**[5] : Nyquist (amplification par bouclage : 1932 [Nyquist, 1932]) et Wiener (pour les antennes : 1941-45).
– **L'ère moderne**, où la théorie du contrôle est à l'intersection des besoins technologiques[6], des méthodes mathématiques, et des méthodes informatiques avec un objectif de contrôle stable, précis, rapide et robuste. Il y a désormais de nombreuses disciplines en théories du contrôle que l'on peut lister ainsi (par couple avec la discipline académique et les applications industrielles) :

Discipline académique	*Application(s) industrielle(s)*
contrôle linéaire	contrôle avec un espace d'état de grande dimension
contrôle non linéaire "classique"	contrôle en présence de dynamiques non linéaires
contrôle optimal	contrôle sous contraintes
théorie des jeux	contrôle via des capteurs distribués
contrôle stochastique	contrôle en environnement bruité
contrôle robuste - (auto) adaptatif	contrôle avec des modèles de mauvaise qualité et des perturbations extérieures
contrôle intelligent	contrôle dans de multiples espaces d'état

A mes yeux, l'importance des résultats de la théorie du contrôle provient des deux champs applicatifs suivants :

– D'une part, la finesse de l'information disponible sur la plupart des processus industriels a considérablement augmenté. Ceci est du notamment à la mise sur le marché de capteurs de plus en plus précis. Dans le domaine de l'automobile par exemple, il n'est possible que depuis peu

[3] K.S. Narendra a participé à un exposé clair des RNF dans le cadre du contrôle [Narendra, 1994] ; pour un autre panorama historique, consulter [Sussmann, 1998].

[4] Cette liste de points et l'exposé du "hard switching control" provient de prises de notes durant l'exposé du Pr. Narendra et d'une discussion avec lui lors du déroulement de ce congrès.

[5] confère par exemple [Bateman, 1945], [Gille *et al.* , 1958] ou [Kalman, 1960].

[6] Pour un panorama, se référer à [Bellman, 1961].

de connaître dynamiquement la composition des polluants (jusque là, on ne connaissait que des volumes moyens à intervalles de temps assez grands).
Pour profiter de la précision de ces nouvelles mesures, il est nécessaire de recourir à des méthodes non linéaires, la finesse des méthodes classiques (souvent linéaires) n'étant plus suffisante.
Une méthode intermédiaire fréquemment utilisée est de découper la zone à analyser en plusieurs parties, et de s'attacher à résoudre la problématique dans chacune de ces parties, avec des outils classiques. Cette méthodologie, qui empêche une prise en compte globale de la problématique, ne peut souvent atteindre que des optima locaux.
Cet état de fait dans les domaines industriels implique que toute amélioration des méthodologies se répercute positivement sur les processus, que ce soit en terme de coût, de qualité, ou de délais.

– D'autre part, les outils informatiques peuvent manipuler des données de plus en plus nombreuses (en terme de données comme en terme de lignes de programme). Cela a mené les ingénieurs informaticiens vers des concepts de modularité des programmes et vers la conception de modules contenant des petites parties d'algorithmes, séparés par des interfaces standardisées.
Si cette approche a eu le mérite de permette la croissance du volume de programmes produits, elle conduit inévitablement à une perte de robustesse du système global : une petite perturbation d'une des briques composant une application peut causer des dégâts très importants.
Ce nouveau type de problème a conduit à la notion d'agent logiciel : un composant situé entre deux interfaces. Un agent peut être un algorithme unitaire comme le processus de pilotage d'une automobile dans un flux de véhicules. L'optimisation locale (en ayant recours à des processus d'auto-organisation) comme globale (via des techniques de recherches d'optimum) de ces agents est un nouveau défi pour la communauté du contrôle.

Chapitre 3

Les réseaux de neurones formels

3.1 Introduction

L'objectif de ce chapitre est d'exposer les particularités des Perceptrons Affines Par morceaux (Piecewise Affine Perceptrons : PAP) ainsi que de positionner leur utilisation relativement à celle d'autres familles réseaux de neurones formels.

Avant tout il convient de consacrer quelques lignes aux réseaux de neurones en général et à leur historique.

C'est en développant sa théorie sur le contrôle par rétroaction dans Cybernetica que Norbert Wiener posa des notions qui participèrent à la formation de la discipline des réseaux de neurones formels [Wiener, 1948]. La **première implémentation** d'un réseau de neurones formels est vraisemblablement due à Minsky et Edmonds qui assemblèrent en 1951 à l'aide de 300 tubes à vide et d'un pilote automatique de bombardier B24 une machine baptisée SNARC qui simulait l'évolution d'un rat dans un labyrinthe [Crevier, 1993].

On peut considérer un réseau de neurones formels (RNF) comme une formalisation très simplifiée d'un ensemble d'une certaine catégorie de neurones biologiques. On trouve en effet une analogie entre les phénomènes neuronaux biologiques et les réseaux de neurones formels : les unités de base du système (neurones) reçoivent une information locale et la transmettent autour de zones spécifiques du réseau (dendrites). Les autres unités présentes dans le voisinage peuvent à leur tour relayer cette information qui se combine ainsi jusqu'à atteindre une zone de sortie où elle est lue (confère illustration 3.1). Le phénomène d'apprentissage provient de ce que le mode de transmission de l'information de chaque unité se modifie en fonction de son état d'activa-

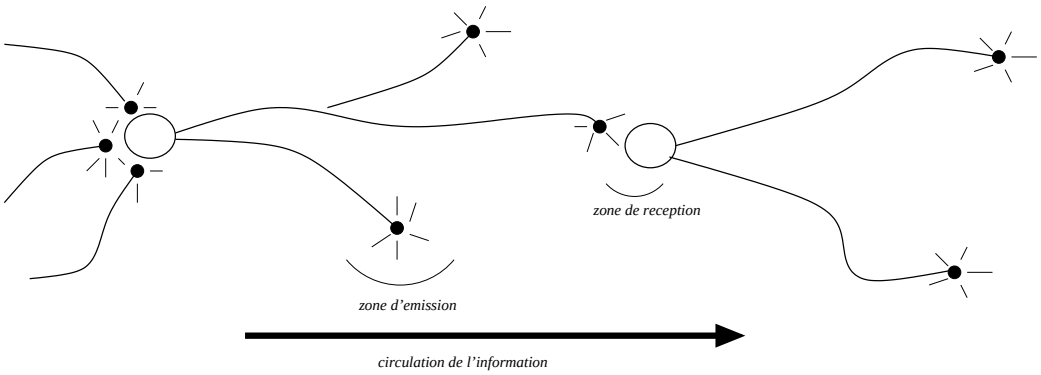


FIG. 3.1 – Représentation symbolique de relais de l'information dans un réseau de neurones biologiques.

tion et d'une grandeur locale (interprétable comme une fonction croissante de l'erreur commise localement). Le résultat de ces modifications maximise (ou minimise) un critère global : il s'agit d'un phénomène d'auto organisation (i.e. les modifications locales d'un système à partir d'informations locales engendrent une optimisation globale).

Un tel réseau est simultanément relais (en transfert) et interpréteur (en apprentissage) de l'information qu'on lui fournit, et pour peu que l'ensemble des grandeurs locales optimisées ait une signification globale, ce système va *évoluer* vers un état optimal vis à vis de ce critère global.

Les figures 3.2 et 3.3 (d'après Szentágithai [SZENTÁGITHAI, 1978] et [SZENTÁGITHAI, 1983] et Eccles [ECCLES, 1989-1992]) montrent combien la complexité et la diversité des cellules du cerveau est loin des classes de fonctions des réseaux de neurones formels. L'analogie n'est que lointaine et provient d'une simulitude fonctionnelle de certains réseaux de cellules nerveuses vus à grande échelle.

Définitions.
Un réseau de neurones formels émule une fonction mathématique dépendant d'un jeu de paramètres W qui associe à tout point x de l'espace d'entrée $\mathcal{X}$ un point u de l'espace de sortie $\mathcal{U}$ tel que :

$$\begin{array}{rcccc} \Psi & : & \mathcal{W}\times\mathcal{X} & \longrightarrow & \mathcal{U} \\ & & (W,x) & \longmapsto & \Psi(W,x) \end{array}$$

qui peut être vue aussi comme :

$$\begin{array}{rcccc} \Psi_W & : & \mathcal{X} & \longrightarrow & \mathcal{U} \\ & & x & \longmapsto & \Psi_W(x) \end{array}$$

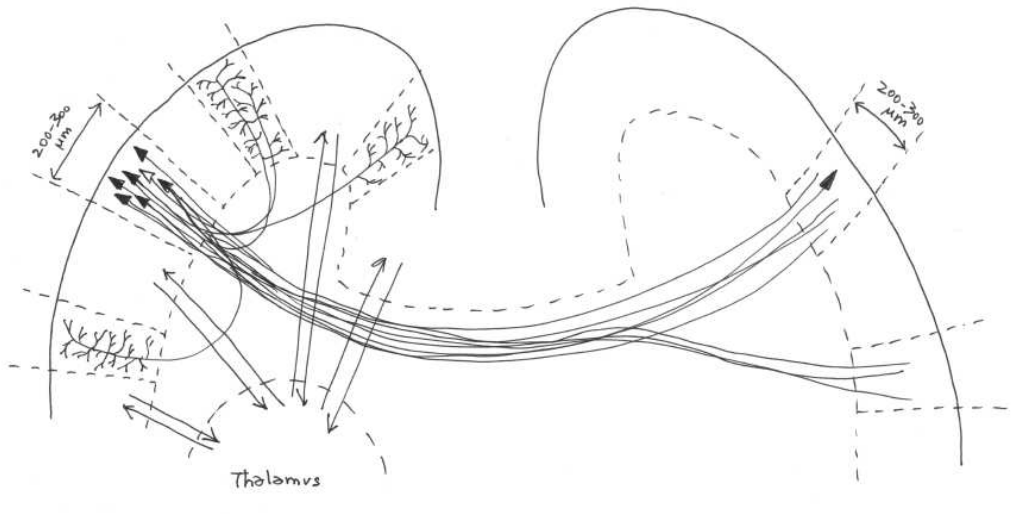


FIG. 3.2 – Illustration du principe général de la connectivité cortico-corticale. Les connexions relient entre elles des colonnes verticales perpendiculaires à la surface du crâne. Les connexions intérieures à un hémisphère sont dues aux cellules de la couche III (cf figure 3.3). Les connexions d'un hémisphère à l'autre sont dues aux couches II-VI.

Suivant la valeur de l'ensemble des paramètres W, Ψ_W va émuler une fonction particulière. La diversité des fonctions émulables par une famille de RNF est une des raisons importantes de leur utilisation.

Apprentissage.
Déterminer le jeu de paramètres qui permet à Ψ_W d'émuler une fonction prédéterminée f revient à réaliser le minimum d'un certain critère de distance $\mathcal{I}$ entre Ψ_W et f qui dépend des paramètres W et de l'information disponible sur la fonction f.

Lorsqu'il est possible de construire itérativement une suite (W_n) des paramètres à partir de $\mathcal{I}$ telle que $\Psi_{\lim_{n\to\infty} W_n} = f$, alors on dira que la suite (W_n) est l'apprentissage de f pour Ψ_W suivant le critère $\mathcal{I}$. L'analogie entre les réseaux de neurones biologiques et une classe de RNF est d'autant plus forte que l'algorithme d'optimisation des paramètres W a une version locale.

Cette partie va d'abord présenter les perceptrons multi couches avant de s'attarder sur les perceptrons affines par morceaux (PAP) et d'en énoncer les principales caractéristiques.

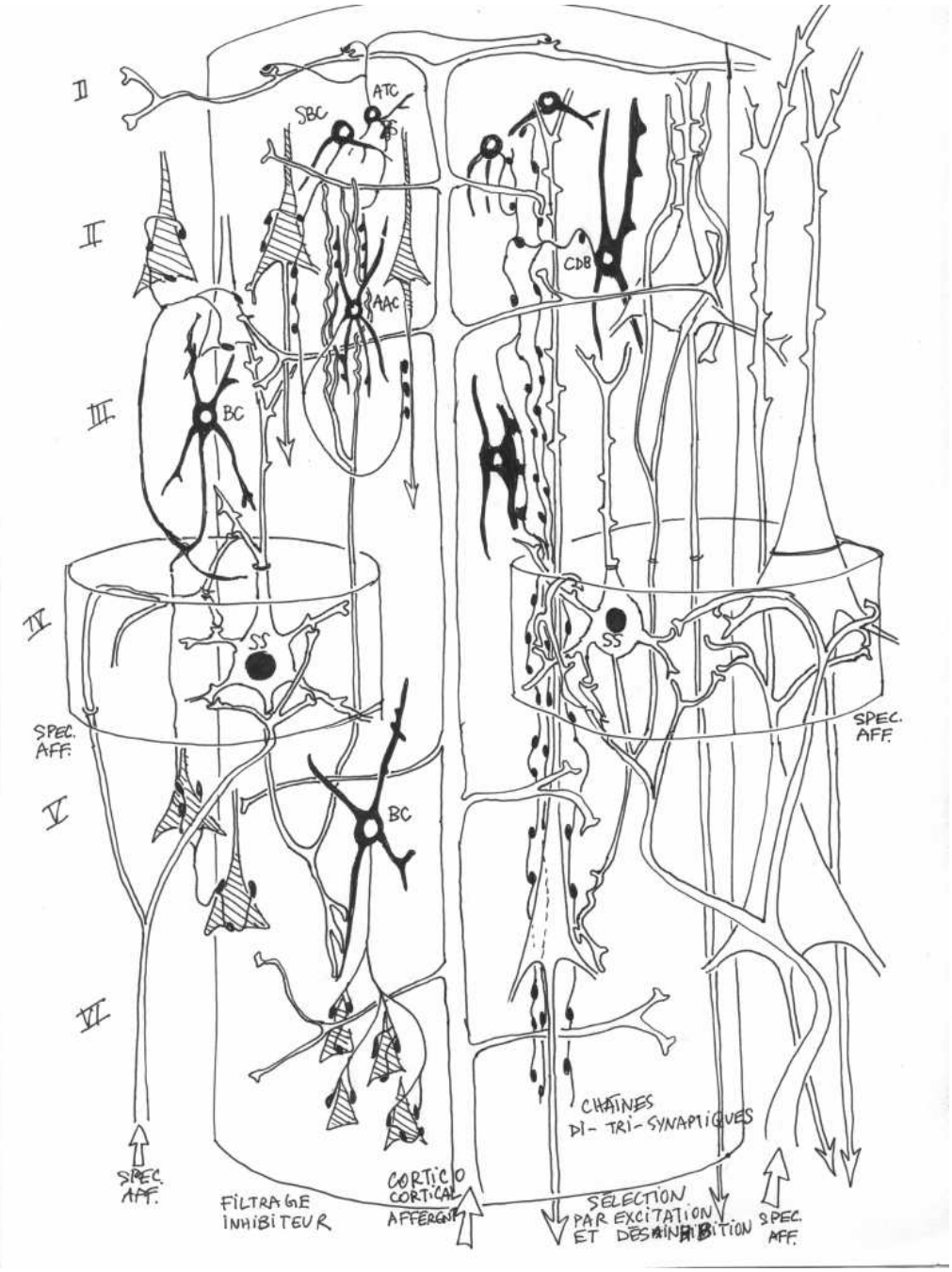


FIG. 3.3 – Connectivité des neurones dans une des colonnes reliant plusieurs couches du cortex (cf figure 3.2). Le cylindre vertical de 300μm figurant au centre a une intersection avec deux zones applaties situées dans la couche IV. Le neurone cortico-cortical afférent (bas de la figure) envoie des terminaisons de façon hétérogène dans toutes les parties de ce module. Dans la couche I, l'étalement des fibres s'étend bien au delà du module. Dans la partie droite de la figure, l'impulsion doit, pour atteindre les cellules pyramidales, franchir les interneurones d'excitation (SS) ou de désinhibition (Cellules à Double Bouquet de Ramon y Cajal). Les CDB agissent spécifiquement sut d'autre interneurones d'inhibition (en noir). Le coté gauche de la figure montre l'action des interneurones d'inhibition comme une filtration qui agit sur les cellules pyramidales ici en pointillé. Les cellules qu'on a des raison de considérer comme inhibiteurs sont en noir (cellules en panier des couches profondes [BC], petites cellules en panier de la couche II [SBC], cellules à axones en touffe [ATC], et une cellule très spécialisée qui relie les axones entre eux et agit sur les segments initiaux des cellules pyramidales [AAC]).

3.2 Les perceptrons multicouches

3.2.1 Définitions et notations

De façon qualitative, un perceptron multi couches de $\mathbb{R}^d$ dans $\mathbb{R}^a$ peut être représenté par le graphe orienté de la figure 3.4.

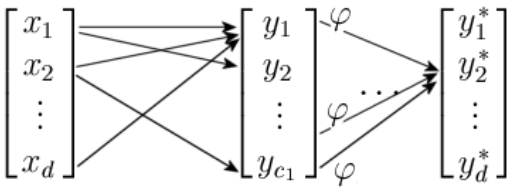


FIG. 3.4 – Représentation d'un perceptron multicouches

Cette représentation doit être lue ainsi :

– chaque scalaire y_i (appelé "activation" du neurone numéro i de la première couche cachée) est la somme pondérée des x_j localisés sur la "couche précédente" (les x_j sont les "entrées" du perceptron) :

$$y_i = \sum_{j=1}^{d} w_{ij} x_j$$

– ceci se répète pour chaque couche k à la différence qu'une fonction d'activation φ_k entre en action (en notant $y^{(k)}$ le vecteur des activations de cette couche) :

$$y_i^{(k)} = \sum_{j=1}^{c_k} w_{ij}^{(k)} \varphi_k(y_j^{(k-1)})$$

L'analogie avec un "réseau de cellules" est la suivante : chaque "cellule" $y_i^{(k)}$ reçoit la somme pondérée des activités des cellules qui lui sont reliées. Les activités sont "normalisées" par l'utilisation d'une fonction d'activation (souvent une sigmoïde) qui contraint les activations transmises aux cellules à rester dans $[0, 1]$.

Plus une pondération $w_{ij}^{(k)}$ est grande en valeur absolue et plus l'activité de $y_i^{(k)}$ est corrélée à celle de $y_j^{(k-1)}$ (positivement ou négativement suivant le signe de $w_{ij}^{(k)}$).

Définition 11 (Perceptrons) *Les perceptrons à n couches cachées de $\mathbb{R}^d$ dans $\mathbb{R}^a$ sont les fonctions de la forme :*

$$\Psi_W(x) = \Phi_n\left(W^{(n)} \cdot \Phi_{n-1}\left(W^{(n-1)} \cdot \cdots \Phi_1\left(W^{(1)} \cdot x + b^{(1)}\right) + b^{(2)}\right) + \cdots\right) + b^{(n)}\right) \tag{3.1}$$

où x est dans $\mathbb{R}^d$ et :

$$\forall k \in \{1, \ldots, n\},\ W^{(k)} \in \mathcal{M}(c_k, c_{k-1}), b^{(k)} \in \mathbb{R}^{c_k},\ c_1 = d,\ c_n = a$$

et où Φ_k est une fonction de $\mathbb{R}^{c_k}$ dans lui-même, qui à $y^{(k)} = (y_1^{(k)}, \ldots, y_{c_k}^{(k)})$ associe :

$$\Phi_k(y^{(k)}) = (\varphi_k(y_1^{(k)}), \ldots, \varphi_k(y_{c_k}^{(k)}))$$

avec φ_k la fonction d'activation de la $k$ème couche du perceptron.

Classiquement, les fonctions d'activation φ_k sont choisies parmi l'identité, la tangente hyperbolique, et $(1 + e^{-x})^{-1}$. Dans la suite, les exemples seront pris avec $\varphi_k = \tanh$ pour tous les k sauf mention explicite du contraire.

Les c_k sont le nombre d'unités (ou neurones) de la couche k et l'ensemble $W = (W^{(k)}, b^{(k)}, 1 \leq k \leq n)$ est l'ensemble des paramètres du réseau. On appelle $W^{(k)}$ les poids et $b^{(k)}$ les seuils (ou biais) de la $k$ème couche de Ψ_W.

On a le résultat suivant [Minsky, 1964] :

Théorème 5 (Approximation universelle) *Quelle que soit f une fonction $\mathcal{C}^\infty$ bornée d'un compact de $\mathbb{R}^d$ dans $\mathbb{R}^a$, et quel que soit le réel ε strictement positif, il existe un entier c_2 et un jeu de paramètre $W = (W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)})$ associés au perceptron Ψ_W tels que :*

$$\int_{x \in \mathbb{R}^d} ||f(x) - \Psi_W(x)||\ dx < \varepsilon \tag{3.2}$$

Ce résultat a fait des perceptrons une classe de fonctions intéressantes pour l'identification de fonctions, d'autant plus que l'on remarque qu'ils sont relativement peu gourmands en paramètres [1].

D'autre part, leur succès est du à la méthode de rétropropagation du gradient qui permet de construire une suite de poids $W(0), W(1), \ldots$ convergeant vers un minimum local de la distance entre Ψ_W et la fonction à émuler f. Ce processus permet notamment de construire $(W^{(i)}, b^{(i)})(n)$ en fonction de $(W^{(i)}, b^{(i)})(n-1)$ et $(W^{(i+1)}, b^{(i+1)})(n-1)$ seulement.

[1] ie : *à non linéarité égale*, le nombre de paramètres d'un perceptron est moins élevé que celui d'un polynôme par exemple.

3.2.2 Rétropropagation

Si on dispose d'un ensemble de couples $(x_i, y_i)_{i\in I}$ sur $\mathbb{R}^d \times \mathbb{R}^a$ tels que $y_i = f(x_i)$ pour tout i, et que l'on cherche (à c_2 fixé) les valeurs des éléments de W qui minimisent :

$$\mathcal{I}(W) = \frac{1}{\mathrm{Card}(I)} \sum_{i\in I} ||y_i - \Psi_W(x_i)||^2 \tag{3.3}$$

on peut construire la suite (W_n) par :

$$\left\{ \begin{array}{lll} \forall w \in W, & w_{n+1} & = & w_n - \gamma_n \cdot \partial_w \mathcal{I}(W)|_{W_n} \\ & W_0 & = & \xi_0 \end{array} \right. \tag{3.4}$$

où (γ_n) est une suite de pas qui tends vers zéro à l'infini dont la somme diverge.

Cette suite converge vers un paramétrage W_* tel que $\mathcal{I}(W_*)$ soit un minimum local de $\mathcal{I}(W)$. (refs)

De plus, l'équation (3.4) est très facilement parallèlisable. On le voit en posant $\varepsilon_W(x, y) = y - \Psi_W(x)$. On aura alors :

$$\mathcal{I}(W) = \frac{1}{\mathrm{Card}((x_i)_i)} \sum_i \varepsilon_W(x_i, y_i) \cdot \varepsilon_W(x_i, y_i)')$$

car si on calcule d'une part :

$$\begin{array}{rcl} \partial_{W_{ab}^{(n)}} \Psi_W(x)_i & = & \varphi_n'(y_i^{(n)}) \cdot \varphi_{n-1}(y_b^{(n-1)}) \cdot \delta_{ai} \\ \delta(W_{ab}^{(n)}) & = & 2\varepsilon_W(x, y)_a \cdot \varphi_n'(y_a^{(n)}) \cdot \varphi_{n-1}(y_b^{(n-1)}) \end{array}$$

et d'autre part :

$$\begin{array}{rcl} \partial_{W_{ab}^{(n-1)}} \Psi_W(x)_i & = & \varphi_n'(y_i^{(n)}) \cdot W_{ia}^{(n)} \cdot \varphi_{n-1}'(y_a^{(n-1)}) \cdot \varphi_{n-2}(y_b^{(n-2)}) \\ \delta(W_{ab}^{(n-1)}) & = & 2 \underbrace{\sum_{i=1}^{c_n} \varepsilon_W(x, y)_i \cdot \varphi_n'(y_i^{(n)}) \cdot W_{ia}^{(n)}}_{\varepsilon_W(x,y)_a^{(n-1)}} \cdot \varphi_{n-1}'(y_a^{(n-1)}) \cdot \varphi_{n-2}(y_b^{(n-2)}) \end{array}$$

on obtient la formule générale suivante pour la couche k, en posant $\varepsilon_W(x, y)^{(n-1)}$ l'erreur rétropropagée à la couche $n-1$:

$$\delta(W_{ab}^{(k)}) = 2\,\varepsilon_W(x, y)_a^{(k)} \cdot \varphi_n'(y_a^{(k)}) \cdot \varphi_{k-1}(y_b^{(k-1)})$$

Remarque. si les dérivées partielles de $\mathcal{I}$ sont définies presque partout, alors la suite des poids va converger presque surement vers un minimum local de $\mathcal{I}$. D'autre part sa version stochastique a des propriétés particulières ; pour ces deux points confère par exemple [Benveniste *et al.* , 1987] ou [Freidlin & Ventcel, 1998].

Les nombreuses contributions à cet algorithme s'attache à augmenter ses capacités à atteindre le minimum global de $(\mathcal{W})$. Ces raffinements sont de plusieurs ordres :

- choix des valeurs de la suite γ
- modification de $\mathcal{I}$
- modification de la direction pointée par le gradient,

3.3 Autres types de réseaux que les perceptrons utilisés pour le contrôle

Deux familles de réseaux de neurones autres que les perceptrons (présentés plus loin) sont souvent utilisés dans le cadre du contrôle par RNF. Il s'agit des réseaux RBF et des réseaux récurrents.

Les premiers sans doute parce qu'ils ont une expression explicitement locale et que leur composition en deux couches hétérogènes (une couche de neurones à activités gaussiennes suivi par une couche linéaire ou sigmoïdale) en fait des candidats a priori simples pour le contrôle adaptatif, assez proche du contrôle flou (cf sections 4.2.1 et 4.3.1, ainsi que [Bahrami, 1993] [Tzirkel-Hancock & Fallside, 1991]).

Les seconds sont vraisemblablement utilisés car ils ont une structure bouclée temporellement, qui laisse entendre que des phénomènes de même nature que ceux émulés par un PID pourront être émulés (l'émulation de la dérivée ou de l'intégrale d'une des variables observables).

3.3.1 Réseaux à fonction à base radiale (RBF)

Les réseaux à fonction à base radiale (Radial Basis Function networks : RBF, cf Poggio) sont les fonctions de $\mathbb{R}^d$ dans $\mathbb{R}$ qui peuvent s'écrire de la forme :

$$\mathfrak{R}_W(x) = \sum_b w_b \exp\left(-\frac{||x - m_b||^2}{2\sigma_b^2}\right) \tag{3.5}$$

où $W = (w_b, m_b, \sigma_b)_b$ sont ses paramètres ($w_b \in I\!R$, $m_b \in I\!R^d$, $\sigma_b \in I\!R$). Un indice b est appelé neurone ou unité de $\mathfrak{R}$. Pour un point x de l'espace de départ $I\!R^d$ donné, les termes de la somme de (3.5) sont d'autant plus petits que x est loin de m_b.

Si on éloigne les centres en prenant par exemple :

(3.6)

$$\begin{aligned}(m_b, w_b, \sigma_b)_b \;=\; & \left(\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, 2/3, 1/25\right), \left(\begin{pmatrix} 1/2 \\ -1/4 \end{pmatrix}, 1/3, 1/25\right),\right. \\ & \left(\begin{pmatrix} -1/2 \\ -1/2 \end{pmatrix}, 1/2, 1/25\right), \left(\begin{pmatrix} 1/2 \\ 3/4 \end{pmatrix}, 1/4, 1/25\right), \\ & \left.\left(\begin{pmatrix} -3/4 \\ 1/2 \end{pmatrix}, 3/4, 1/25\right)\right)\end{aligned}$$

on obtient comme graphe de $\mathfrak{R}$ le premier dessin de la figure 3.5.

Si au contraire on les rapproche (on qu'on élargisse leurs bases), on obtient localement des combinaisons linéaires des gaussiennes "activées" par la présence de l'entrée à proximité. Par exemple on peut prendre :

(3.7)

$$\begin{aligned}(m_b, w_b, \sigma_b)_b \;=\; & \left(\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, 2/3, 1/8\right), \left(\begin{pmatrix} 1/2 \\ -1/4 \end{pmatrix}, 1/3, 1/8\right),\right. \\ & \left(\begin{pmatrix} -1/2 \\ -1/2 \end{pmatrix}, 1/2, 1/8\right), \left(\begin{pmatrix} 1/2 \\ 3/4 \end{pmatrix}, 1/4, 1/8\right), \\ & \left.\left(\begin{pmatrix} -3/4 \\ 1/2 \end{pmatrix}, 3/4, 1/8\right)\right)\end{aligned}$$

Un des avantages des réseaux RBF est leur comportement local très intuitif. Ils sont souvent utilisés pour effectuer des classifications ou même pour des tâches de contrôle qui s'apparentent à de la classification.

Ils sont aussi utilisés dans le contrôle pour générer un contrôleur adaptitif, car il est assez naturel de générer des règles de "glissement" des centres avec l'évolution du système à contrôler.

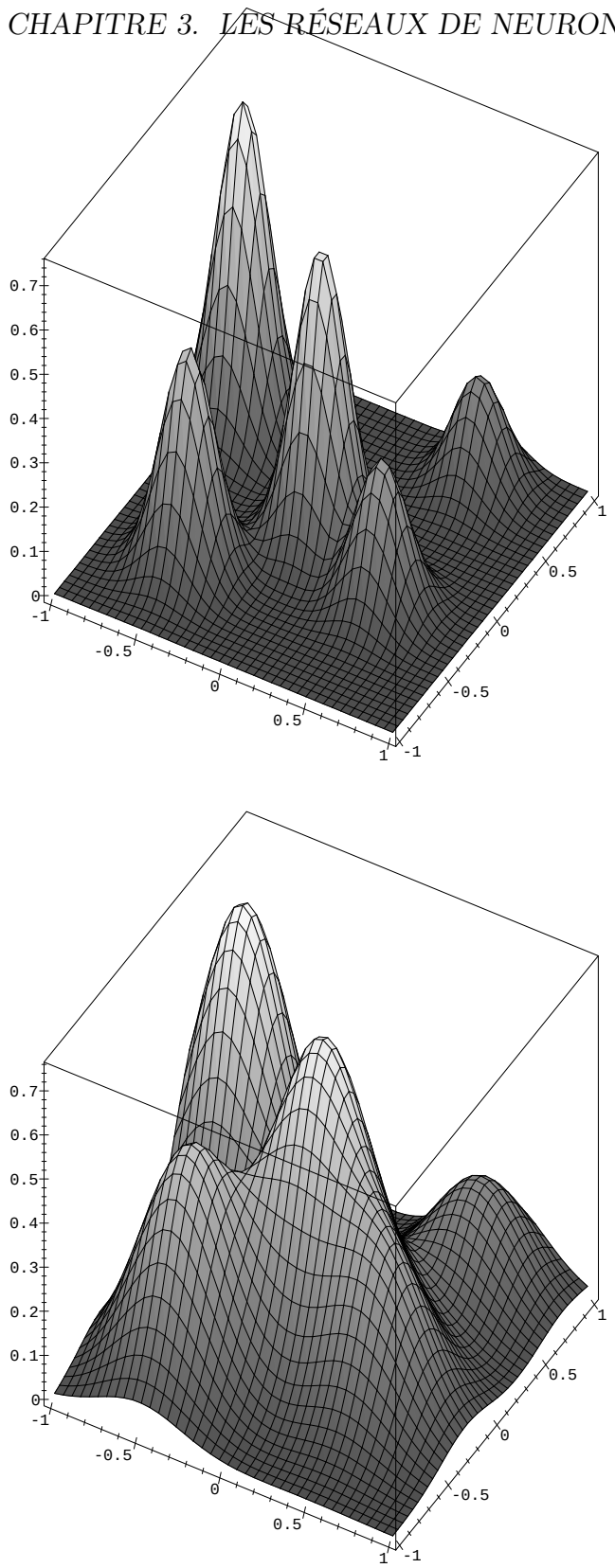


FIG. 3.5 – Le graphe du réseau RBF dont les paramètres dont ceux de (3.6) à gauche et ceux de (3.7) à droite.

Néanmoins, il faut noter que les propriétés locales des Perceptrons Affines Par morceaux exposées plus loin encouragent à les utiliser dans le contexte du contrôle aussi bien que les réseaux RBF.

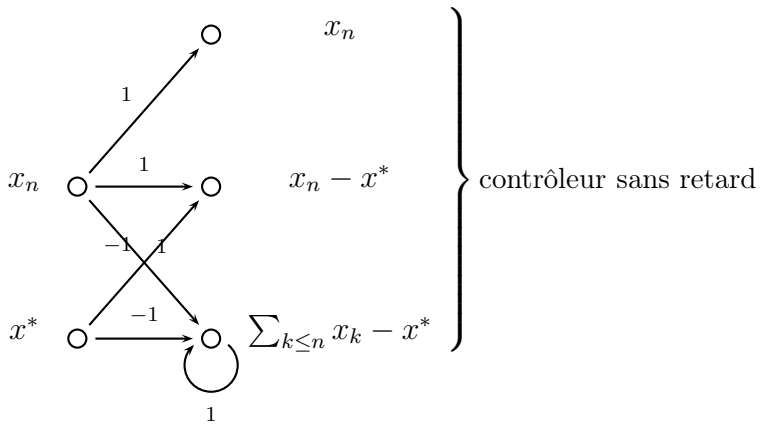


FIG. 3.6 – Un LQI remplace l'espace d'état x_n par celui, trois fois plus grand : $(x_n, \sum_{k\leq n} x_k - x^*, x_n - x^*)$ où x^* est la consigne à atteindre.

3.3.2 Les réseaux récurrents

Les réseaux récurrents offrent, a priori, plus de possibilités que les perceptrons, ils sont en effet constitués d'unités totalement connectées les unes aux autres et n'ont aucune contrainte sur les connections entre leurs unités (il n'est pas nécessaire que le graphe de leurs connections soit acyclique).

La formulation d'un réseau réccurent fait appel à des opérateurs retards. Par exemple, pour formaliser un perceptron à une couche cachée bouclé sur lui-même et de sortie $\mathfrak{C}_W(x)$, on peut écrire :

$$\left\{ \begin{array}{lcl} \mathfrak{C}_W(x) & = & \Phi_2\left(W^{(2)} \cdot \Phi_1(y^{(1)}) + b^{(2)}\right) \\ \sigma y^{(1)} & = & W^{(1)} \left[\begin{array}{c} y^{(1)} \\ x \end{array} \right] + b^{(1)} \end{array} \right. \tag{3.8}$$

Le bouclage possible entre leurs neurones leur permet en outre d'émuler un opérateur retard, voire intégral ou dérivé. Il est par exemple très simple d'imaginer un graphe de connections qui envoie l'état x_n d'un système dynamique sur l'entrée d'un contrôleur LQI : $(x_n, x_n - x^*, \sum_{k\leq n} x_k - x^*)$, avec x^* une consigne (figure 3.6).

Dans le cadre du contrôle, ceux qui ont été utilisés sont des réseaux de type perceptrons avec quelques bouclages ou retards supplémentaires (pour les retards, confère par exemple [UNGERER, 1999] et pour les bouclages [KU, 1995]).

Les réseaux de Jordan, souvent utilisés en contrôle [VENUGOPAL & PANDYA, 1994] [PUSKORIUS & FELDKAMP, 1993], sont compris dans la classe des fonctions de type (3.8). Plus simplement ce sont des réseaux pour lesquels la sortie est réinjec-

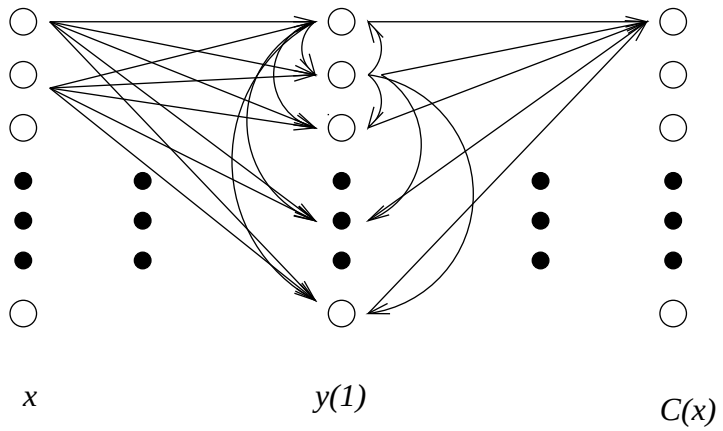


FIG. 3.7 – diagramme réprésentant synthétiquement les RNF récurrents de (3.8) (les biais sont omis et certains liens ne sont pas dupliqués).

tée en entrée :

$$\left\{\begin{array}{lcl} y^{(1)} & = & W^{(1)}\left[\begin{array}{c} x' \\ x \end{array}\right] + b^{(1)} \\ \mathfrak{C}_W(x) & = & \Phi_2\left(W^{(2)} \cdot \Phi_1(y^{(1)}) + b^{(2)}\right) \\ \sigma x' & = & \Phi_2\left(W^{(3)} \cdot \Phi_1(y^{(1)}) + b^{(3)}\right) \end{array}\right. \tag{3.9}$$

Bien entendu dans les deux cas il est nécessaire d'initialiser certaines valeurs des variables internes des réseaux récurrents.

Un des effets principaux de l'utilisation des réseaux récurrents est qu'il créent des retards. Il est en effet possible de considérer un tel réseau récurrent comme une fonction non récurrente de toutes les entrées présentées au réseau :

$$\mathfrak{C}_W(x_n) = f(x_n, x_{n-1}, \ldots, x_0) \tag{3.10}$$

Appelons *ordre* du réseau le nombre n, qui peut être égal à l'infini.

Si un système est contrôlable, il l'est par une fonction du type $g(x_n)$. Même s'il est parfois nécessaire de considérer un sur-système (par exemple confère 2.3), cela revient alors à manipuler un espace d'entrée plus grand du type $X_n = (x_n, x_{n-1}, \ldots, x_0)'$ dans lequel le contrôle bouclé s'exprime bien par $g(X_n)$.

Etant donné qu'on peut avoir un système non contrôlable qui admet un sur-système qui lui est contrôlable, utiliser un RNF récurrent revient en quelque sorte à laisser l'algorithme d'apprentissage déterminer lui-même la

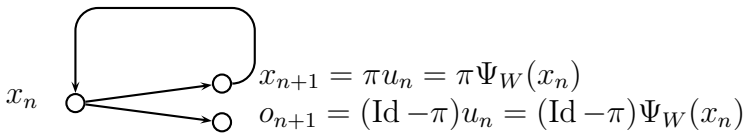


FIG. 3.8 – Un PAP bouclé sur lui-même

dimension du sur-espace d'état que l'on va utiliser. La phase d'apprentissage du réseau de neurones récurrent va donc devoir simultanément estimer les bonnes valeurs des paramètres du RNF, mais aussi la bonne dimension de l'espace d'état. Il s'agit d'une double tâche extrêmement difficile.

D'autre part, il faut noter qu'en traitant le cas des PAP en boucle fermée, il est possible de considérer qu'on traite de réseaux récurrents d'une forme particulière puisque, si on pose $f(x, u) := \pi u$ où π est une projection (afin d'identifier le contrôle u à un sous ensemble des neurones du réseau récurrent), on traite de réseaux bouclés sur eux-même :

$$\left\{ \begin{array}{lcl} x_{n+1} & = & \pi \Psi_W(x_n) \\ o_n & = & (\mathrm{Id} - \pi) \Psi_W(x_n) \end{array} \right. \tag{3.11}$$

L'apprentissage d'un réseau récurrent est donc en quelque sorte un problème de contrôle, mais pour lequel la fonction à contrôler est l'identité (figure 3.8). Il s'agit donc d'un problème complexe ne serait-ce que par les questions de contrôlabilité (donc d'apprentissabilité) que cela soulève. Un début de réponse est notamment amorçé par Sontag dans [SONTAG & QIAO, 1998], [SONTAG, 1998a] et [SONTAG & SUSSMANN, 1997].

Les réseau récurrents ont aussi été utilisés pour émuler des contrôleurs [KU, 1995], [VENUGOPAL & PANDYA, 1994].

Chapitre 4

Contrôle par réseaux de neurones formels : état de l'art

4.1 Liens entre les RNF et le contrôle

Les fonctions de la classe des réseaux de neurones formels sont de plus en plus utilisées dans le cadre du contrôle d'un système dynamique.

D'abord parce qu'il y a une relation naturelle entre les RNF et le contrôle par boucle : la plupart des processus de détermination des paramètres d'un RNF relèvent eux-mêmes de la théorie du contrôle .

Formalisme 4.1 : *Réseaux de neurones formels, définitions et apprentissage.*

Il est en effet possible de poser la problématique de l'apprentissage d'un RNF comme suit :

- Soit Ψ_W une fonction de $\mathcal{E}$ dans $\mathcal{F}$ paramétrée par une variable W à valeurs dans $\mathcal{W}$. On lui associe une "fonction d'activité" $\tilde{\Psi}_W$ de $\mathcal{E}$ dans $\mathcal{G}$ (à un instant t, $\tilde{\Psi}_{W_t}(x_t)$ est l'*activité* du réseau). Il est important que Ψ_W émule un ensemble fonctionnel vaste‡ lorsque W parcourt $\mathcal{W}$.
- La problématique est alors de choisir W afin que Ψ_W émule une fonction $f : \mathcal{E} \to \mathcal{F}$ particulière, connue uniquement par un ensemble de couples $(x_t, y_t)_{t\in\mathbb{N}}$ vérifiants $y_t = f(x_t)$‡.
- On dispose d'une fonction d'erreur $\mathfrak{E}$ qui permet de mesurer l'écart‡ entre deux points de $\mathcal{F}$; on pose $\epsilon_t = \mathfrak{E}(y_t, \Psi_{W_t}(x_t))$ qui est un réel positif.
- Une "fonction d'apprentissage" $\mathfrak{A}$ de $\mathcal{W} \times (\mathbb{R}^+ \times \mathcal{G})^{\mathbb{N}}$ dans $\mathcal{W}$ est alors

utilisée ainsi :

$$\begin{cases} a_{t+1} &= \tilde{\Psi}_{W_t}(x_{t+1}) \\ \epsilon_{t+1} &= \mathfrak{E}(y_{t+1}, \Psi_{W_{t+1}}(x_{t+1})) \\ w_{t+1} &= \mathfrak{A}(w_t, (\epsilon_1, a_1), \ldots, (\epsilon_{t+1}, a_{t+1})) \end{cases}$$

Il s'agit bien d'un système d'équations aux différences qui régit la trajectoire de W_t au cours du temps. Il est contrôlé par le choix de la fonction $\mathfrak{A}$.

Une classe de RNF est entièrement déterminée par Ψ_W, $\tilde{\Psi}_W$ et $\mathfrak{A}$ (souvent $\tilde{\Psi}$ est suffisamment explicite pour être omise).

Construire une classe de RNF, c'est choisir $\mathfrak{A}$ de telle sorte que la limite‡ des Ψ_{W_t} lorsque $t \to \infty$ soit f.

Les concepts marqués du signe ‡ (l'importance de la taille de l'espace fonctionnel parcouru par les Ψ_W ; l'égalité de y_t avec $f(x_t)$ —car il peut y avoir du bruit— ; la "mesure d'écart" entre les éléments de $\mathcal{F}$ et la "limite" des W_t) sont à prendre qualitativement, ils seront précisés au fur et à mesure de ce mémoire.

Fin du formalisme 4.1

Cette section, sans anticiper les quelques explications sur différentes classes de RNF fournies dans la partie 5, passe en revue les utilisations déjà faites des RNF en contrôle. Ces utilisations sont séparées ici en quatre familles, exposées à la suite :

- les utilisations de RNF comme **sous-systèmes d'un processus de contrôle** très vaste [1], en utilisant essentiellement leurs capacités d'approximateurs (contrôle prédictif, par modèle inverse, etc)
- le cas où le cadre de contrôle repose sur des résultats de **géométrie différentielle**, les RNF sont alors utilisés pour accéder aux dérivées partielles de certains sous-systèmes.
- l'approche basée sur le **contrôle par renforcement**, très répandue dans le contexte du contrôle par agents.
- l'utilisation de RNF pour générer un **contrôleur optimal en minimisant un critère global** sur les trajectoires du système à contrôler.

En parallèle à cette typologie (pour un tour d'horizon : [Agarwal, 1997], [Warwick, 1995], [Chen, 1993], [Chen, 1995]), il ne faut pas mettre de coté les travaux étudiant la mise en place pratique de contrôles neuronaux en répondant à des questions du type : comment fixer des règles permettant de décider qu'une période d'apprentissage est nécessaire ? faut-il construire plu-

[1] Pour un tour d'horizon avec une vaste bibliographie confère : [Sontag, 1992b] [Heermann, 1992] [Chaudhuri, 1995] [Goutte & Ledoux, 1996] .

sieurs contrôleurs neuronaux spécialisés et dans ce cas comment gérer les passage de l'un à l'autre ("hard switching") ?

4.1.1 Notions préliminaires

Avant de chercher un réseaux de neurones formels de la forme $u = g(x)$ qui contrôle (1.7) ou (1.8), il faut s'assurer qu'il existe au moins une fonction qui contrôle le système. Quelques résultats permettent de le garantir.

Une des premières questions qui se pose lors de la détermination d'une fonction de contrôle par bouclage ($u = g(x)$) devant amener le système (1.7) (ou (1.8)) vers un équilibre particulier est la relation entre la forme de g et la possibilité du contrôle.

Pour reprendre l'exemple (1.9), la fonction de contrôle choisie $-(2|x|)^{\frac{1}{3}}$ n'est pas différentiable en zéro. Si l'on se restreignait aux fonction analytiques, le contrôle serait nécessairement de la forme $u = \mathcal{O}(x)$, ce qui implique que le système contrôlé serait de la forme $\dot{x} = x + \mathcal{O}(x)$, qui est instable [Sontag, 1992a].

4.1.2 Présentation de l'état de l'art

La section suivante va présenter les principales gammes d'utilisations des réseaux de neurones formels dans le cadre du contrôle. Elles ont été regroupées en deux familles :

- l'utilisation des RNF comme **approximateurs**, puis la construction d'une stratégie de contrôle *autour* de cet approximateur,
- l'utilisation des RNF comme **contrôleurs**.

Chaque famille d'utilisation sera présentée suivant le même formalisme :

- une présentation qualitative de la méthode,
- une formalisation mathématique qui y correspond,
- éventuellement quelques remarques sur les points forts et les faiblesses de la méthodologie,
- une illustration.

Afin de ne pas trop s'étendre sur les variantes de chaque méthode, les références bibliographiques retenues sont celles qui contiennent elles-mêmes de larges bibliographies auxquelles le lecteur pourra se référer.

4.2 Les réseaux de neurones en tant qu'approximateurs

4.2.1 Le contrôle prédictif

Principe. Le contrôle prédictif (souvent ramené aux acronymes GPC et LPC : *Generic Predicitive Control* et *Linear Predictive Control*), repose sur la recherche d'une trajectoire optimale à partir de *simulations* utilisant un modèle du comportement du système et de son contrôleur (par exemple [Ronco & Gawthrop, 1999][2] et [Lino, 1995]).

Dans certains cas la solution du programme d'optimisation est explicite, et il n'est alors pas nécessaire de recourir à de réelles simulations.

A chaque instant, le système (réel plus simulateur) permet d'envisager des approximations des trajectoires du système pour toute stratégie de contrôle ; c'est la stratégie la plus performante (suivant un critère qui définit le contrôle) qui sera retenue.

Formalisme. x est la variable d'état, le système dynamique règle les trajectoires d'un système du type :

$$x_{n+1} = f(x_n, u_n)$$

ou bien

$$\dot{x} = f(x, u)$$

u est la variable de contrôle et x^* est un état de consigne à atteindre ; le contrôle est uniquement déterminé en fonction de l'état et de la consigne :

$$u = g(x, x^*)$$

On dispose d'un modèle de f :

$$\Psi_W(x, u) \sim f(x, u)$$

qui est par exemple obtenu en minimisant un écart entre f et Ψ_W :

$$\min_W \int_{x \in \mathfrak{X}, u \in \mathcal{U}} \text{écart}\left(f(x,u), \Psi_W(x,u)\right) du\, dx$$

et d'un critère à minimiser :

$$\mathcal{J}(t_0, t) = \sum_k I(x_k, x_k^*)$$

On va utiliser Ψ_W pour choisir u :

[2] Ronco et Gawthrop ont une approche intéressante dans [Ronco & Gawthrop, 1998], entre le contrôle prédictif et la "feedback linearization".

1. on génère une collection $(u^{(i)})_{1\leq i\leq I}$ (suite de I contrôles) en utilisant la fonction g (qui est souvent une *méthodologie* de génération de contrôle),
2. on simule les I trajectoires $x^{(i)}$ correspondantes en utilisant Ψ_W à la place de f :
$$x_{n+1}^{(i)} = \Psi_W(x_n^{(i)}, u_n^{(i)}),\ \forall 1 \leq i \leq I$$
3. on calcule les valeurs du critère pour chaque $x^{(i)}$,
4. on décide d'utiliser le $u^{(k)}$ auquel correspond le moindre coût.

Dans les cas simples pour Φ_W et $\mathcal{J}$, il est parfois possible d'obtenir immédiatement $u^{(k)}$ à partir de x_0 et x^*.

Remarque. Cette méthode donne de bons résultats applicatifs (Neumerkel, Kruger & Hidiroglu) et permet, à l'aide en outre d'une "stratégie d'apprentissage" adéquate, de générer un contrôleur adaptatif. La classe de RNF la plus largement utilisée dans ce cadre est celle des réseaux RBF.

4.2.2 Modèle inverse

Principe. Une autre utilisation des RNF en contrôle qui repose avant tout sur leur capacité d'approximation est de construire un **modèle inverse** ([Ashhab *et al.* , 1998], [Calise & Rydyk, 1998] ou [Kambhampati *et al.* , 1996]) du système à contrôler. Il s'agit d'une fonction qui associe à chaque couple (x_1, x_2) de l'espace d'état un contrôle $u^{1\to 2}$ qui, appliqué au système, permettra de l'amener de x_1 à x_2.

Il sera ainsi possible :
- de contrôler directement le système : l'amener de x_0 à x^* se fait en appliquant le modèle inverse à ces deux points,
- éventuellement de construire une trajectoire de x_0 à x^* en plusieurs étapes (la plupart du temps par optimisation combinatoire).

Formalisme. En conservant la notation f de la sous section précédente, l'ensemble W des paramètres du modèle inverse $\Psi_W(x, y)$ est construit via le programme de minimisation suivant :

$$\min_W \int_{x_1\in\mathfrak{X}, x_2\in\mathfrak{X}} \text{écart}\left(x_2, f(x_1, \Psi_W(x_1, x_2))\right) dx_1\, dx_2$$

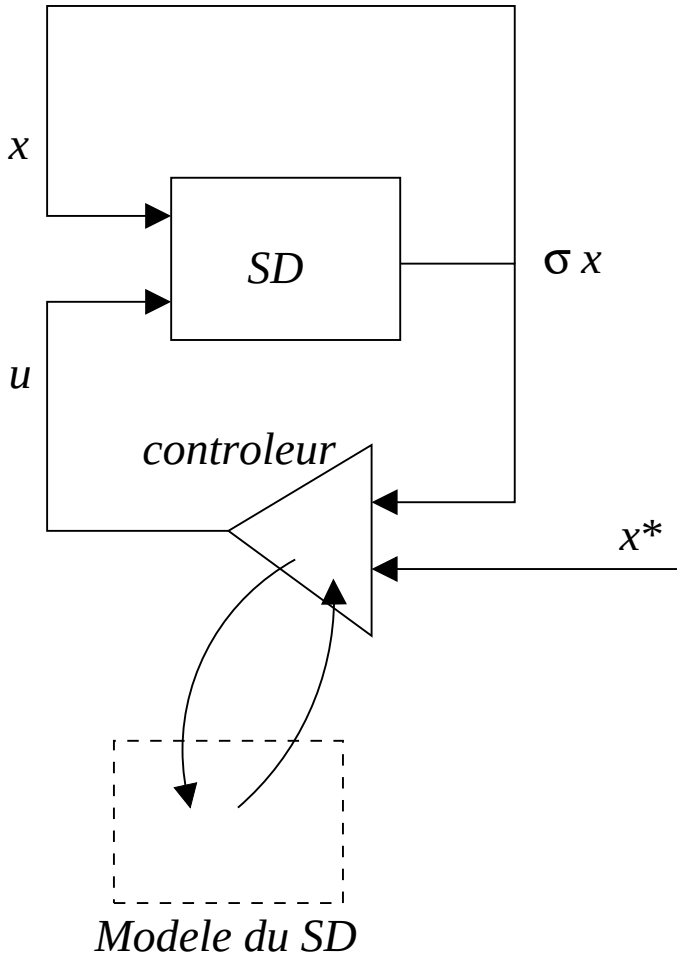


FIG. 4.1 – Contrôle prédictif : le carré SD est le système dynamique à contrôler, le triangle le contrôleur ; celui-ci utilise l'état du système et une consigne pour déterminer (en fonction d'un modèle du système à contrôler : le carré en pointillés) le contrôle à appliquer.

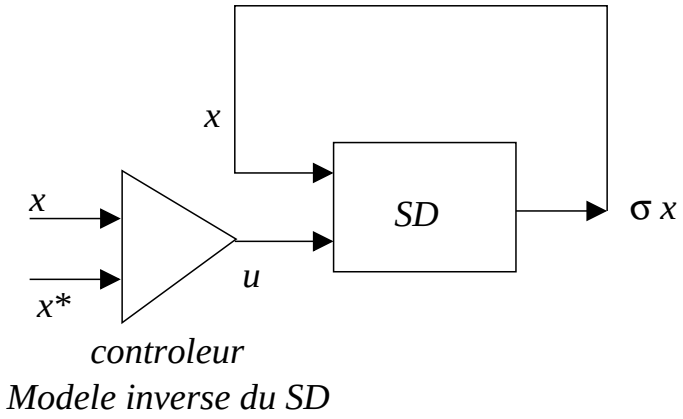


FIG. 4.2 – Contrôle par modèle inverse.

Remarque. Cette méthode soulève le problème de l'existence d'un tel Ψ_W, et des informations nécessaires à sa construction. Ces deux points reposent sur l'inversibilité de f. D'autre part il est assez délicat de faire directement intervenir la fonction de coût du contrôle dans la construction de Ψ_W.

Une autre difficulté est rencontrée lorsqu'il n'est pas possible d'amener le système à l'état de référence (x^*) en une action de contrôle. Dans ce genre de cas $\Psi_W(x, x^*)$ n'existe pas. Il est alors nécessaire de construire un chemin d'objectifs $x_0^*, x_1^*, \ldots$ réalisables, ce qui peut être une question difficile en soi.

4.2.3 Accéder aux dérivées partielles d'un système à contrôler

Les résultats de géométrie différentielle en théorie du contrôle étant souvent très utiles mais nécessitant la connaissance de dérivées partielles du système à contrôler, les RNF sont parfois utilisés pour estimer ces grandeurs.

Le plus souvent le système à contrôler est modélisé par un RNF différentiable, dont les dérivées partielles sont ensuite calculées et utilisées pour celles du système original. (cf par exemple [FREGENE & KENNEDY, 1999]).

Remarques. Il faut associer deux remarques à cette méthode :

- l'application de ce genre de résultats nécessite des hypothèses sur la classe du système à contrôler (par exemple qu'il soit semi-linéaire) qui doivent être vérifiées.
- obtenir une "bonne approximation" d'une fonction ne donne aucune garantie de proximité entre la dérivée de l'estimateur et celle de la

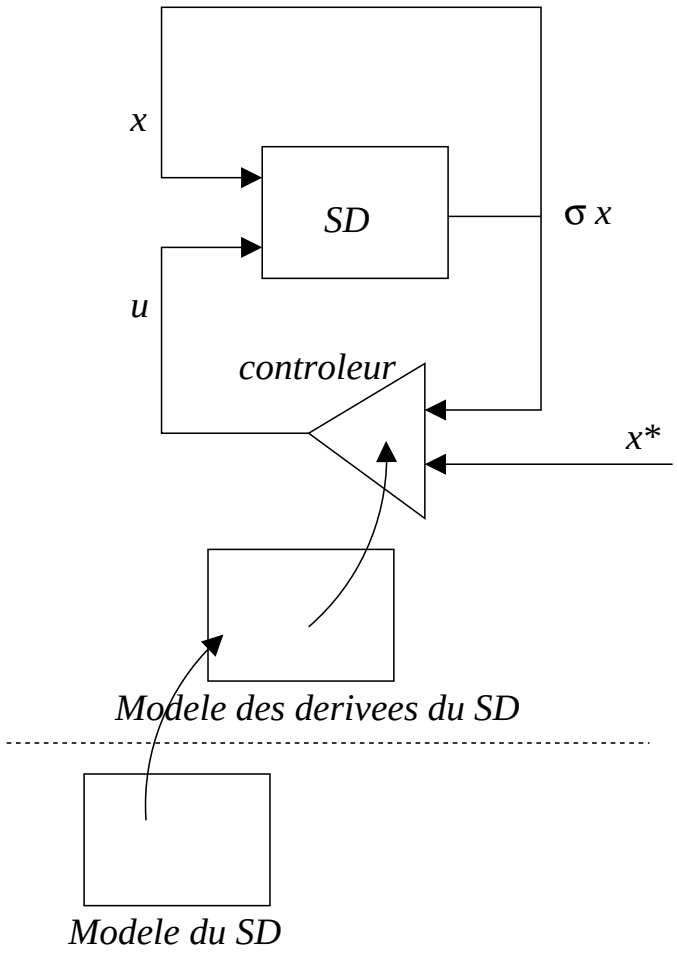


FIG. 4.3 – Contrôle par approximation des d'erivées du système à contrôler.

fonction originale (confère figure 4.4). La meilleur approche est sans doute d'approximer directement les dérivées partielles à utiliser par la suite. La difficulté est alors d'accéder à leurs valeurs.

On peut trouver dans [Sontag, 1992a] de bonnes pistes vers les hypothèses nécessaires à une telle utilisation.

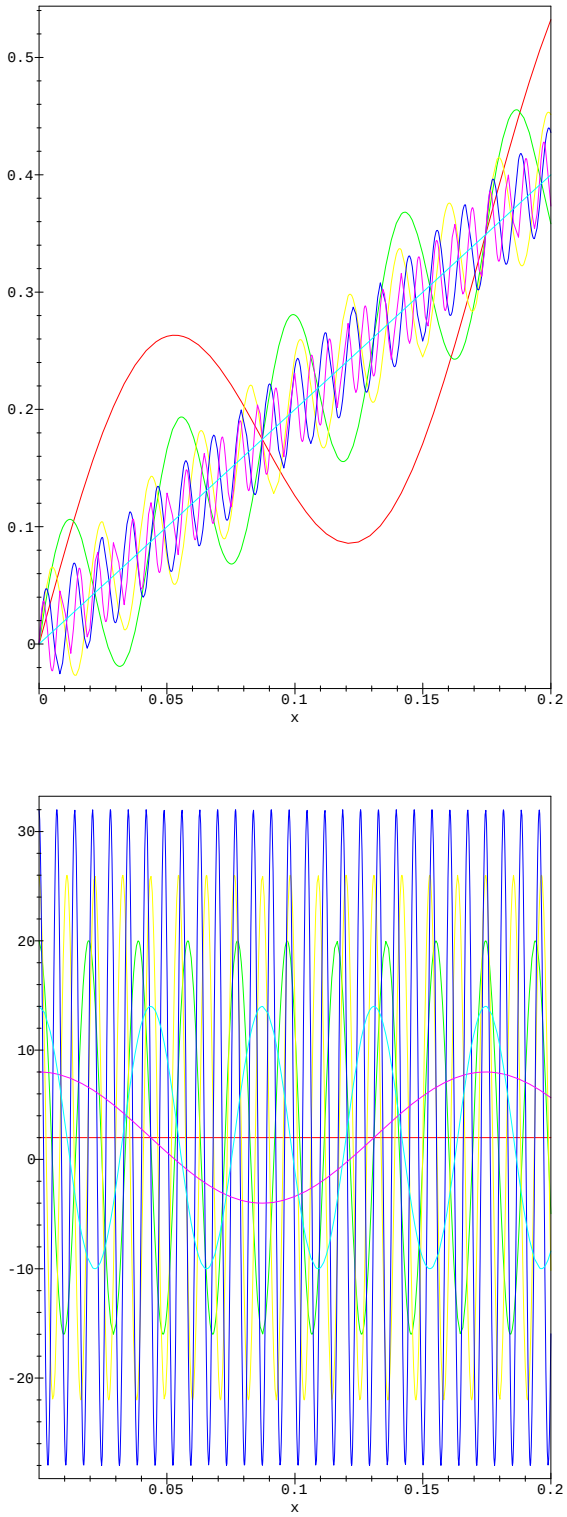


FIG. 4.4 – Une fonction linéaire qui reste dans un tube de taille ε autour de f peut avoir des dérivées très éloignées de celles de f. Ici $f_a(x) = \sin(a^2x)/a+bx$ et $g(x) = bx$. On a $\lim_{a\to\infty}\{g(x) - f_a(x), \forall x \in I\!R\} = \lim_{a\to\infty}[-1/a, 1/a] = \{0\}$ et $\lim_{a\to\infty}\{g'(x) - f'_a(x), \forall x \in I\!R\} = \lim_{a\to\infty}[-a, a] = I\!R$.

4.3 Les réseaux de neurones en tant que contrôleurs

4.3.1 Imitation d'un contrôleur classique

Principe. Une utilisation des réseaux de neurones qui semble contourner les difficultés inhérentes à l'apprentissage d'un contrôleur est de mettre au point un contrôleur avec une méthode classique (typiquement un contrôleur PID[3]), puis de réaliser un RNF qui l'émule, et enfin d'utiliser les capacités d'apprentissage des RNF afin d'émuler un contrôle adaptatif (la première phase d'un tel programme (émulation d'un contrôleur PID) a d'ailleurs été une des solutions mises en place au Groupe de Recherche Logiciel de Renault, avant le commencement de mes travaux de thèse chez Renault sous contrat CIFRE).

Formalisme. Il s'agit donc simplement de construire un RNF Ψ_W qui émule le plus correctement possible un contrôleur g construit par ailleurs :

$$\min_W \int_{x\in\mathfrak{X},x^*\in\mathfrak{X}} \text{écart}\left(g(x,x^*),\Psi_W(x,x^*)\right)\right) dx\,dx^*$$

Remarque. Les principaux inconvénients inhérents à cette approche sont :
- qu'il est nécessaire de construire un contrôleur classique,
- que la problématique de l'adaptation "en ligne" d'un contrôleur sont au moins aussi complexes que ceux de sa mise au point automatique (sauf en ce qui concerne l'initialisation).

On peut citer [Gorinevsky & Feldkamp, 1996], [Tzirkel-Hancock & Fallside, 1991], [Unar & Murray-smith, 1995] et [Bahrami, 1993].

Ce mélange de contrôle classique (pour la mise au point) suivi d'une adaptation conduit assez rapidement au contrôle flou (fuzzy control, [Qiao & Mizumoto, 1996] ou [Huang & Huang, 1996]).

4.3.2 Contrôle par renforcement

Principe. Cette méthodologie se découpe en plusieurs étapes :

1. initialisation (souvent aléatoire) d'un contrôleur paramétré, la structure du contrôleur doit permettre d'établir une relation *de bon sens* entre ses paramètres et le comportement du système. Par exemple : lorsque

[3]Proportionnel Intégrale Dérivée

le paramètre w_1 du contrôleur augmente, la troisième coordonnée des trajectoires du système sont de plus en plus grandes, etc.

2. observation de trajectoires du système contrôlé et association à intervalle régulier des *anomalies* de la trajectoire (trop "vers le haut", ou trop "à gauche"),
3. application d'un correctif aux paramètres du contrôleur, ce correctif est basé sur les relations de bon sens entre les variations des trajectoires et celles de paramètres, ainsi que son l'amplitude des anomalies observées.

Formalisme. On se donne un système à contrôler : $\sigma x = f(x, u)$ et un contrôleur $u = \Psi_W(x, x^*)$. On dispose en outre de règles de renforcement des paramètres $\Delta_w(w, x_n, u_n, x_{n+1}, x^*)$, qui associe une variation à appliquer au paramètre w à un quintuplet constitué de la valeur d'un paramètre, de l'état du système à l'instant n, du contrôle qu'on lui applique à cet instant, de son état à l'instant suivant et de l'état où on voudrait être.

A chaque instant n, on opère :

1. génération et application du contrôle : $u_n = \Psi_W(x_n, x^*)$, $x_{n+1} = f(x_n, u_n)$,
2. calcul du renforcement à appliquer à chaque paramètre w :

$$\delta w = \Delta_w(w, x_n, u_n, x_{n+1}, x^*)$$

3. modification des paramètres : $w(n+1) = w(n) + \delta w$.

La plupart du temps Δ_w est d'expression extrêmement simple, souvent de la forme (α_w est une constante différente pour chaque paramètre w, et non pour chaque valeur de w) :

$$\Delta_w(w, x_n, u_n, x_{n+1}, x^*) = \alpha_w \; ||u_n|| \; ||x_{n+1} - x^*||$$

Remarque. La mise au point d'un contrôleur par renforcement nécessite un opérateur de "critique" de la réussite du contrôle (cf V. Gullapalli, [Keerthi & Ravindran, 1995], [Kuvayev & Sutton, 1997] [McCallum, 1992], [Sheppard *et al.* , 1994], [Horng, 1998]), comme l'illustre la figure 4.5, surtout : [Gullapalli, 1991].

Cette approche cherche avant tout à engendrer des algorithmes légers en temps de calcul et donc n'utilise l'information qu'à très court terme seulement. Son utilisation est très répandue en robotique [Thrun & Mitchell, 1993], [Barto & Anandan, 1985]. Cette approche frappe l'esprit par son aspect immédiatement *adaptatif*, presque descriptible comme l'émulation d'un processus darwinien (le contrôleur est parfois un algorithme génétique, ce qui donne

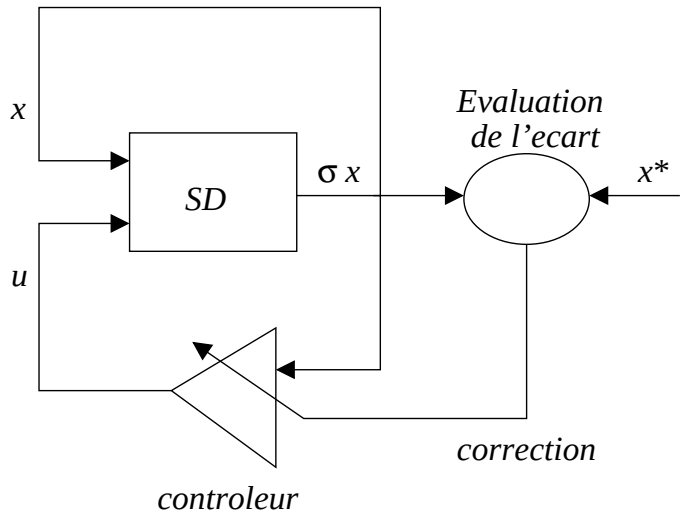


FIG. 4.5 – Contrôle par renforcement.

une légère variante de l'approche) : c'est sans doute pour cela que le milieu de la robotique y recours fréquemment.

La nécessité de connaître les relations qualitatives entre les paramètres et le comportement du système ainsi que l'initialisation souvent aléatoire du contrôleur limitent l'utilisation du contrôle par renforcement :

- aux système à faible dimensionnalité,
- à un petit nombre de paramètres "clefs" du contrôleur permettant d'obtenir un contrôle adaptatif à partir d'un contrôleur construit plus classiquement.

Notons d'autre part que les relations qualitatives entre les paramètres du contrôleur et le comportement des trajectoires du système sont des expressions qualitatives des dérivées partielles du système couplé (système à contrôlé plus contrôleur) par rapport aux paramètres du contrôleur. On peut rapprocher l'approche de la section suivante (où on effectue une réelle minimisation) à celle-ci en l'exprimant comme une de ses extensions. En effet, pour des systèmes aux dérivées partielles d'expression simple et de faible inertie, l'approche de la section suivante peut se ramener à un contrôle par renforcement.

4.3.3 Contrôle par réseau de neurones via une modification de la dynamique d'évolution de ses poids

Un point de vue complémentaire au précédent consiste à considérer l'évolution simultanée de l'état du système à contrôler et du contrôleur (émulé par un RNF). Par exemple en notant Θ l'ensemble des paramètres du RNF et x l'état du système :

$$\left\{ \begin{array}{rcl} \dot{x} & = & f(x, \Psi_\Theta(x)) \\ \dot{\Theta} & = & g(x, \Theta) \end{array} \right. \tag{4.1}$$

Ce type d'analyse a été menée hors du contexte du contrôle par Hirch et Benaïm dans le cadre de l'étude de la convergence des réseaux à mémoire associative ([Hirsch & Smith, 2003], [Benaïm & Hirsch, 1999], [Lehalle, 1994]).

Dans le cadre du contrôle, l'objectif est de choisir g afin de garantir la convergence du système couplé. La théorie de la viabilité [VIAB] offre la possibilité de contraindre la dynamique des paramètres Θ afin de garantir cette convergence. Ceci a été réalisé par Aubin et Seube [Aubin, 1996] dans le cas du contrôle d'un véhicule sous-marin. L'état était de dimension 6 et l'espace de contrôle de dimension 3.

L'idée est d'obtenir une condition sur Θ qui permette à chaque instant et pour des valeurs connues de toutes les variables de (4.1) de savoir quelle nouvelle valeur lui attribuer.

Cette méthode a plusieurs caractéristiques : la première est qu'elle est pragmatique par héritage direct de cet aspect de la théorie de la viabilité (la convergence recherchée est une convergence à horizon fini, rendue accessible par la prise en compte à chaque instant des bassins d'attraction du système couplé), la seconde qu'elle est très gourmande en temps de calcul (car la plupart du temps la construction des bassins doit se faire numériquement). Néanmoins ce point de vue a l'avantage de mêler la dynamique du RNF à celle du système permettant ainsi de considérer la convergence du système global.

4.3.4 Réglage d'un contrôleur par minimisation d'une fonction de coût

Les travaux présentés dans ce mémoire se placent essentiellement dans le cadre de la minimisation d'une fonction de coût. Il s'agit alors de régler les paramètres d'un RNF utilisé comme contrôleur afin de minimiser un critère qui est défini dans l'espace de **toutes les trajectoires sur toutes leurs**

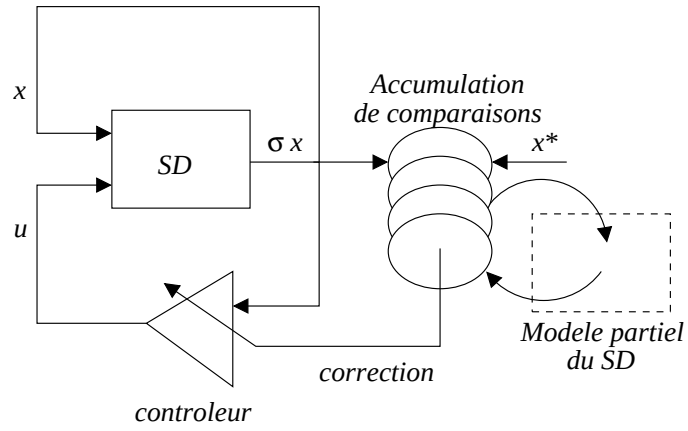


FIG. 4.6 – Contrôle par minimisation d'un critère via une descente de gradient.

durées.

La méthode de mise au point des paramètres d'un tel RNF par descente du gradient le long du critère à minimiser est naturelle dans ce cadre.

Les auteurs ayant précédemment traité cette approche se sont avant tout attachés à décrire l'**algorithme de mise à jour des paramètres** qui découle d'une descente du gradient. Ce mémoire ne se focalise pas sur ce point même si le chapitre 7.3 de la partie 7 expose ce type de calcul pour en déduire une **formule de récurrence** permettant la mise au point des paramètres assez simplement, il expose aussi un point de vue global amenant une **stratégie d'apprentissage** et pas seulement une descente de gradient.

Une fois une telle "règle de mise au point" choisie, il reste de nombreux aspects sans réponse, et c'est essentiellement sur ceux-là que ce travail met l'accent. Il s'agit alors de la difficulté particulière de l'**initialisation** d'un RNF dans ce contexte, du choix de la **durée d'observation** des trajectoires, et de la stabilité d'un contrôleur obtenu par **minimisation du risque empirique** plutôt que du risque théorique (qui est inaccessible). Il s'agit aussi du choix de la "stratégie d'adaptation", point qui n'est ici qu'évoqué dans la partie 6.

Ce dernier aspect est étudié pour une grande partie par Narendra[4] (ISIC/ISAS99,

[4]Ronco développe dans [RONCO, 1997] une approche similaire ; il utilise de nombreux contrôleurs locaux, mais de façon moins explicite que Narendra.

[Levin & Narendra, 1993], [Narendra & Mukhopadhyay, 1994]) qui opte pour le "**hard switching**" : lorsque le système contrôlé n'a plus le comportement désiré, le RNF est considéré comme inadapté. Il est alors *dupliqué* et sa nouvelle version est soumise à l'optimisation de ses paramètres, puis associée à la zone de l'espace d'état où elle se trouve. L'espace d'état est ainsi peu à peu pavé de contrôleurs adaptés localement[5].

Effet de bascule entre contrôleurs locaux. Il y a alors émergence de situations de transitions entre contrôleurs. Le fait que la "compatibilité" des contrôleurs entre eux ne soit pas une des contraintes du problème peut engendrer des situations de sauts brusques (d'où le nom de "hard switching") d'une zone à l'autre, phénomènes très difficilement maîtrisés aujourd'hui.

Il semble que peu d'auteurs ont recours à une modélisation de ce genre de phénomènes par des chaînes de Markov, approche qui semble pourtant riche et pourrait être une des voies possibles de poursuite du travail exposé ici.

Les résultats obtenus pour ce type de système peuvent s'appliquer aux réseaux récurrents assez simplement. Dans ce cadre il est possible d'utiliser toute la puissance de la théorie du contrôle non linéaire pour répondre à des questions difficiles sur l'apprentissage et le comportement de tels réseaux ([Sontag & Qiao, 1998], [Sontag, 1998a] et [Sontag & Sussmann, 1997]).

4.4 Bilan

Les nombreuses approches présentées dans ce chapitre sont le prolongement naturel de l'étude des réseaux de neurones formels : le passage de leur propre "mise au point" par feed-back à leur utilisation en tant que feed-back. La richesse de ces familles de fonctions ressort dans de telles situations. Il y a en quelque sorte une "amplification" de la moindre déviation à un comportement espéré à chaque passage de la boucle.

Les travaux présentés dans ce mémoire prolongent certaines de ces approches (essentiellement celles décrites dans la dernière section) tout en mettant l'accent sur une classe particulière de RNF : les perceptrons affines par morceaux (PAP : Piecewise Affine Perceptrons) qui ont l'avantage d'apporter un éclairage au comportement des perceptrons multicouches plus classiques.

[5] auxquels ils serait sans doute possible d'adjoindre des contrôleurs linéaires classiques "spécialisés" dans certaines zones.

Le fait de conserver une dynamique des poids “classique” permet de faire en sorte que les résultats obtenus soient directement exploitables non seulement dans le cadre du contrôle, mais aussi dans le contexte de l'utilisation de perceptrons comme estimateur d'une fonction-réponse.

Deuxième partie

Perceptrons affines par morceaux : une famille de RNF particulièrement riche

Chapitre 5

Des réseaux de neurones classiques aux PAP : une famille de fonctions globales localement explicites

5.1 Dimensionnement des perceptrons

5.1.1 Nombre d'unités cachées

Une des difficultés du choix d'une classe de RNF (comme pour toute autre famille de modèles) réside dans celui de la complexité maximale qu'elle peut émuler (figure 5.1 ; les illustrations de cette figure ont été réalisées avec un petit package MapleV programmé pour l'occasion, celui-ci permet de réaliser des descentes de gradient simples et d'initialiser un réseau de neurones en répartissant "uniformément" ses neurones sur un hypercube). Cette complexité est très liée au nombre de neurones cachés $c_2, c_3, \ldots$

Cet aspect est essentiellement développé via le concept de dimension de Vapnik-Cervonenkis ([Girosi, 1995]) qui fait plus généralement l'objet de la théorie de l'apprentissage : il s'agit d'une mesure de la capacité d'une classe de fonction à émuler des non linéarités (cf section 7.4.2). Dans le cadre des classifieurs, cette grandeur est très naturelle : il s'agit du maximum de points séparables de toutes les façons possibles par la classe de fonctions. La classe des droites du plan a une dimension de Vapnik-Cervonenkis de 3 car il est impossible de séparer 4 points de toutes les façon possibles par une droite dans le plan (une séparation impossible est par exemple celle des quatre sommets d'un rectangle qui ont un label alterné lorsqu'on parcours le périmètre

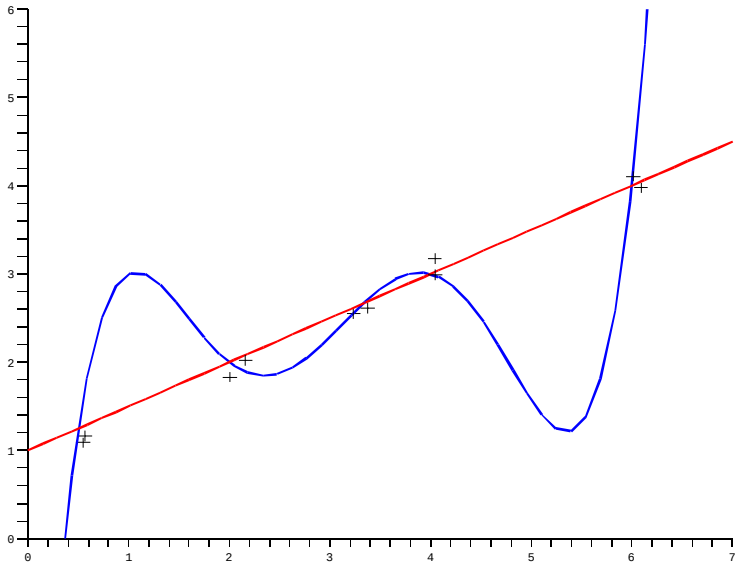


FIG. 5.1 – Le résultat de deux apprentissages sur un nuage de points. Modèle linaire (en rouge) : $c_2 = 1$ n'est pas assez grand (underfitting) ; modèle fortement non linéaire : c_2 est trop grand (overfitting).

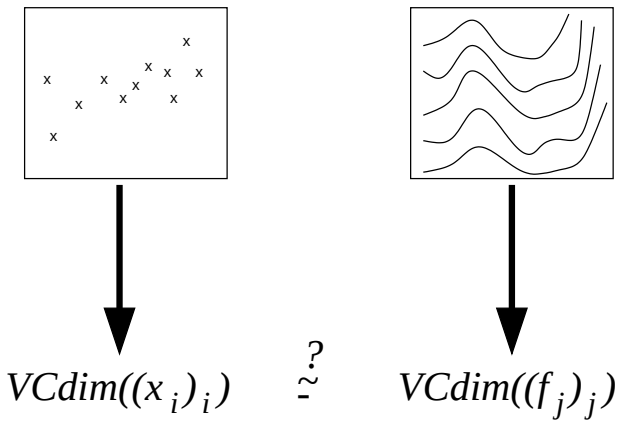


FIG. 5.2 – Il est parfois possible de calculer la VCdim d'un tirage aléatoire de valeurs d'une fonction d'un part, et de la classe de RNF choisie en fonction d'un de ses paramètres (par exemple le nombre d'unités cachées c_2) d'une autre ; la comparaison de ces deux grandeurs guide dans le dimensionnement de la classe de fonction finalement utilisée.

du rectangle).

Connaissant la VCdim de la fonction à émuler et une relation entre c_2 et la VCdim de la famille des perceptrons associée, on peut choisir c_2 afin d'avoir les deux dimensions du même ordre (figure 5.2).

Il existe de nombreuses méthodes ad hoc pour traiter cette difficulté comme surévaluer c_2 puis ajouter à $\mathcal{I}(W)$ un terme proportionnel à $||W||$ afin d'éviter que Ψ_W soit trop chahutée (confère section 5.7), ou éliminer les unités estimées inutiles par une méthode statistique (pruning).

Dans le cas des réseaux de neurones particulièrement utilisés dans ce mémoire (les perceptrons affines par morceaux), on verra plus loin qu'il est possible de relier directement le nombre de leurs neurones cachés c_2 à la nonlinéarité maximale qu'ils sont capable d'émuler. Ceci peut permettre de choisir un c_2 adapté au problème posé.

5.1.2 Partition de Voronoï

Pour mieux comprendre l'action d'un perceptron à une couche cachée, une première méthode consiste à faire varier empiriquement certains de ses paramètres. On remarque assez rapidement quatre types d'action (figure 5.3) qui seront confirmés par l'étude des PAP (section 5.1) :

– une variation de $W^{(1)}$ change localement la pente de la fonction émulée

et la décale partout ailleurs.
- une variation de $b^{(1)}$ translate localement la fonction émulée dans l'espace d'entrée.
- une variation de $W^{(2)}$ a un effet de même nature qu'une variation de $W^{(1)}$.
- une variation de $b^{(2)}$ décale la valeur de la sortie de la fonction émulée.

D'autre part, si on prend des tangentes hyperboliques comme fonctions d'activation (le raisonnement est similaire pour les autres fonctions), son comportement "presque linéaire" dans $[-1, 1]$ et "presque constant" en dehors (confère figure 5.3) met en avant une série de zones sur chacune desquelles le comportement de Ψ_W a une pente *plutôt constante*.

La figure 5.5 représente l'action d'un perceptron à une couche cachée de 4 unités de $\mathbb{R}^2$ dans $\mathbb{R}$, dont les poids sont choisis arbitrairement :

$$W^{(1)} = 2\begin{bmatrix} 1 & 1 \\ -5/3 & -1 \\ 1 & -1 \\ -1 & 4/3 \end{bmatrix} \quad , \quad b^{(1)} = \begin{bmatrix} 2.5 \\ 2 \\ 1.5 \\ 1 \end{bmatrix}$$
$$W^{(2)} = \begin{bmatrix} 0.2 & 0.1 & 0.15 & 0.1 \end{bmatrix} \quad , \quad b^{(2)} = [0.05]$$

5.1.2.1 Développement limité d'un perceptron multi-couches

Calcul de la Dérivée première.

Le gradient d'une telle fonction fournit une information sur sa pente. Pour un perceptron de $\mathbb{R}$ dans $\mathbb{R}$ à une couche cachée de formule générale (3.1), sa formulation est :

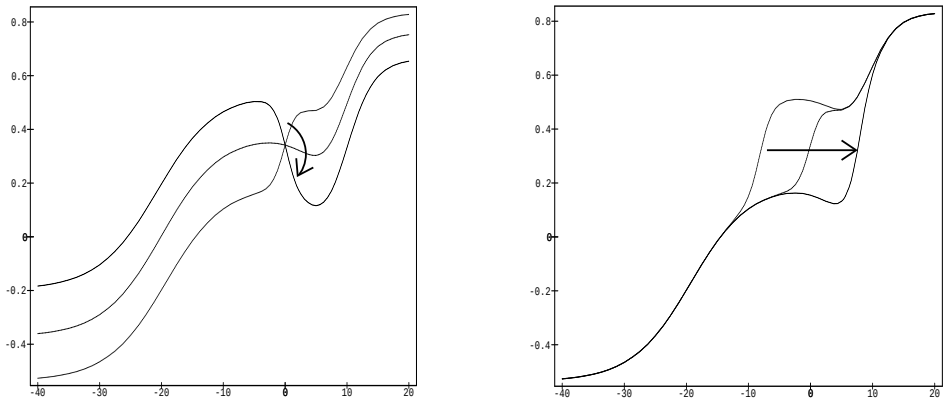
$$\partial_x \Psi_W(x) = \varphi'(y^{(2)}) \cdot W^{(2)} \cdot \mathcal{D}_{\varphi'}(y^{(1)}(x)) \cdot W^{(1)} \tag{5.1}$$

avec la notation :

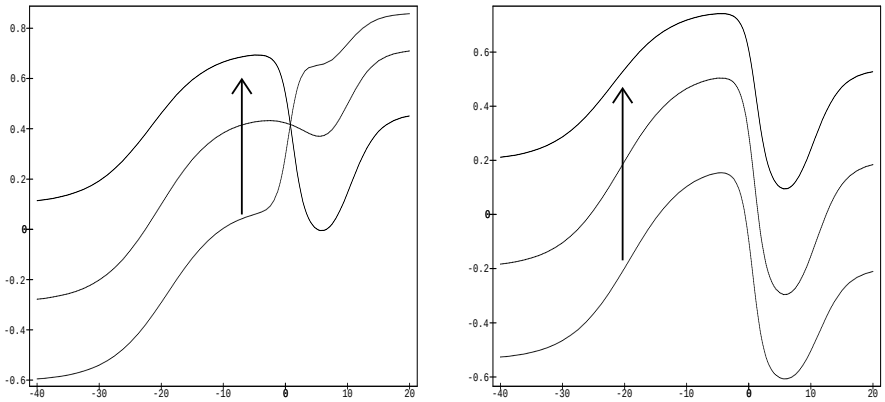
$$\mathcal{D}_g(z) = \begin{bmatrix} g(z_1) & & 0 \\ & \ddots & \\ 0 & & g(z_n) \end{bmatrix}$$

Les extréma de Ψ_W correspondent aux points de $\mathbb{R}^d$ qui annulent $\partial_x \Psi_W$.

Pour l'exemple particulier de la figure 5.5, le carré de la norme de $\partial_x \Psi_W$ est représenté à droite.



$W^{(1)}$ $b^{(1)}$



$W^{(2)}$ $b^{(2)}$

FIG. 5.3 – Pour un perceptron de $\mathbb{R}$ dans $\mathbb{R}$, chacune de ces figures illustre la variation sur son graphe causée par une variation de $W^{(1)}$, $b^{(1)}$, $W^{(2)}$ ou $b^{(2)}$. La ligne centrale est le graphe du perceptron, les deux autres lignes sont les graphes du même perceptron pour deux valeurs différentes de ses paramètres.

On peut caluler l'action au premier ordre de Ψ_W, qui est liée à celle de $\partial_x \Psi$ par :

$$\Psi_W(x^* + h) = \Psi_W(x^*) + \partial_x \Psi_W|_{x^*} \cdot h + o(||h||)$$

et au second ordre :

(5.2)
$$\Psi_W(x^* + h) = \Psi_W(x^*) + \partial_x \Psi_W|_{x^*} \cdot h + \tfrac{1}{2} h' \cdot H_x(\Psi_W)(x^*) \cdot h + o(||h||^2)$$

avec $H_x(\Psi_W)(x^*)$ la matrice **hessienne** de Ψ_W par rapport à x et calculée en x^* :

$$H_x(\Psi_W)(x^*) = [h_{ij}]_{ij},\ h_{ij} = \left.\frac{\partial^2 \Psi_W(x)}{\partial_{x_i} \partial_{y_j}}\right|_{x^*}$$

Cette formule implique que plus la norme de $H_x(\Psi_W)(x^*)$ est proche de zéro, et plus Ψ_W est proche de son approximation affine :

$$\Psi_W(X) = \Psi_W(x^*) - \partial_x \Psi_W|_{x^*} \cdot x^* + \partial_x \Psi_W|_{x^*} \cdot X$$

La figure 5.6 montre les variations du carré de la norme de $H_x(\Psi_W)(x^*)$ en fonction de x^*. Des zones de $\mathbb{R}^2$ sur lesquelles $H_x(\Psi_W)(x^*)$ est "faible" se dégagent.

5.1.2.2 Découpage de l'espace de départ

Une poursuite "naturelle" de ces remarques est de tenter de ramener l'action de Ψ_W à une collection d'actions locales, les plus simples possibles.

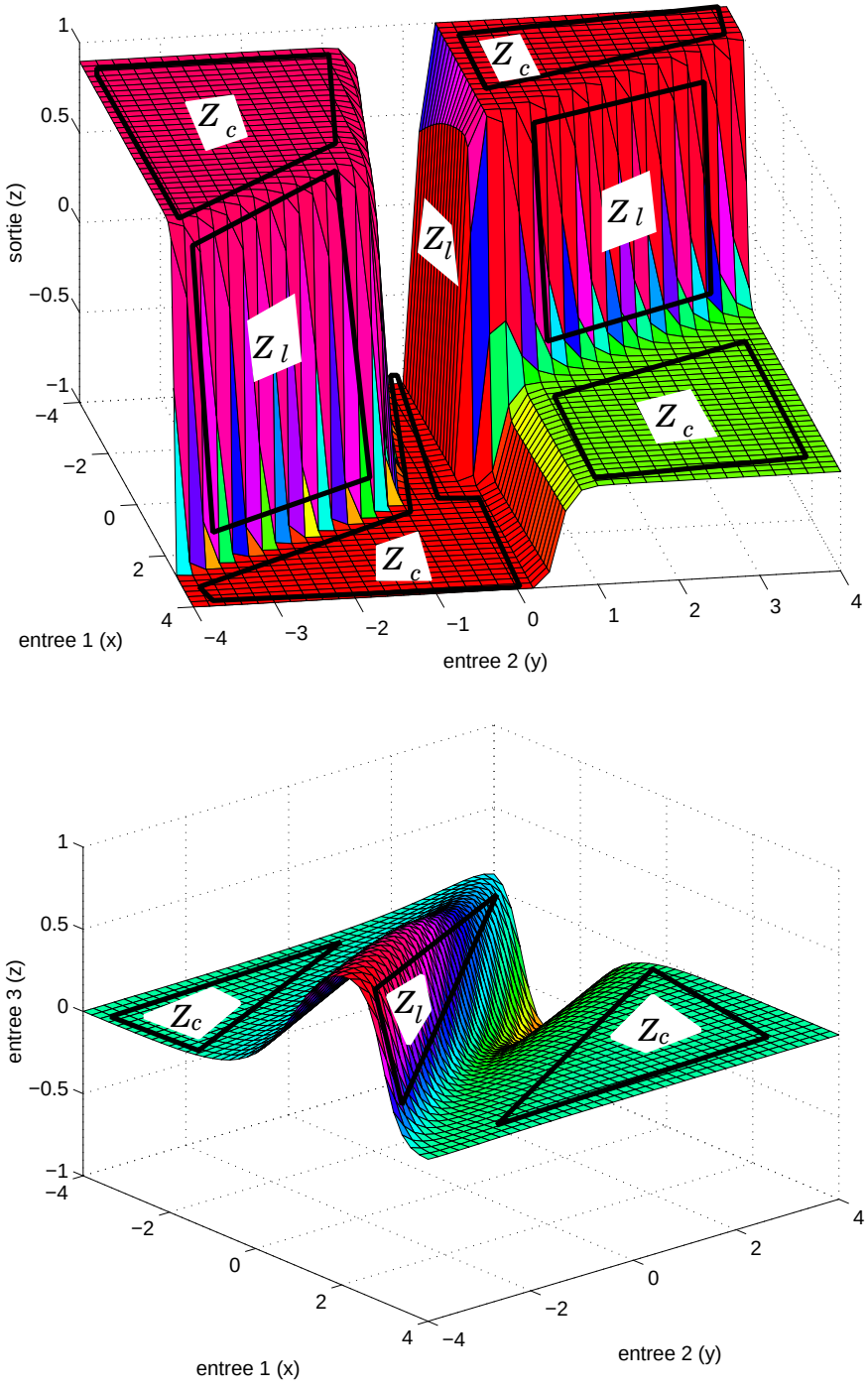
L'espace de départ peut alors être découpé en deux types de zones : celles où l'action de Ψ_W est presque linéaire, et les autres. Les zones du premier type sont centrées sur les points qui annulent $H_x(\Psi_W)(x^*)$.

Le formalisme le plus adéquat pour ce genre de représentation est le partitionnement en cellules de Voronoï [Moller, 1994].

Définition 12 (Cellule de Voronoï) *Pour un ensemble de points isolés $(x_i)_i$ de $\mathbb{R}^d$, la cellule de Voronoï associée à x_j est la zone de $\mathbb{R}^d$ dont les points sont plus proches de x_j que de tout autre point de $(x_i)_{i \neq j}$. Il s'agit d'une cellule polyhédrale dont les bords sont les médiatrices entre x_j et les autres points.*

Cette définition (cf [Moller, 1994] et [Preparata, 1985]) s'étend simplement aux ensembles de points non isolés pourvus qu'ils forment une suite d'ensembles isolés chacun connexe.

FIG. 5.4 – Zones et hyperplans frontières pour deux perceptrons de $\mathbb{R}^2$ dans $\mathbb{R}$ dont les poids ont été pris au hasard : on devine aisément les zones sur lesquelles le comportement du perceptron et "presque affine".



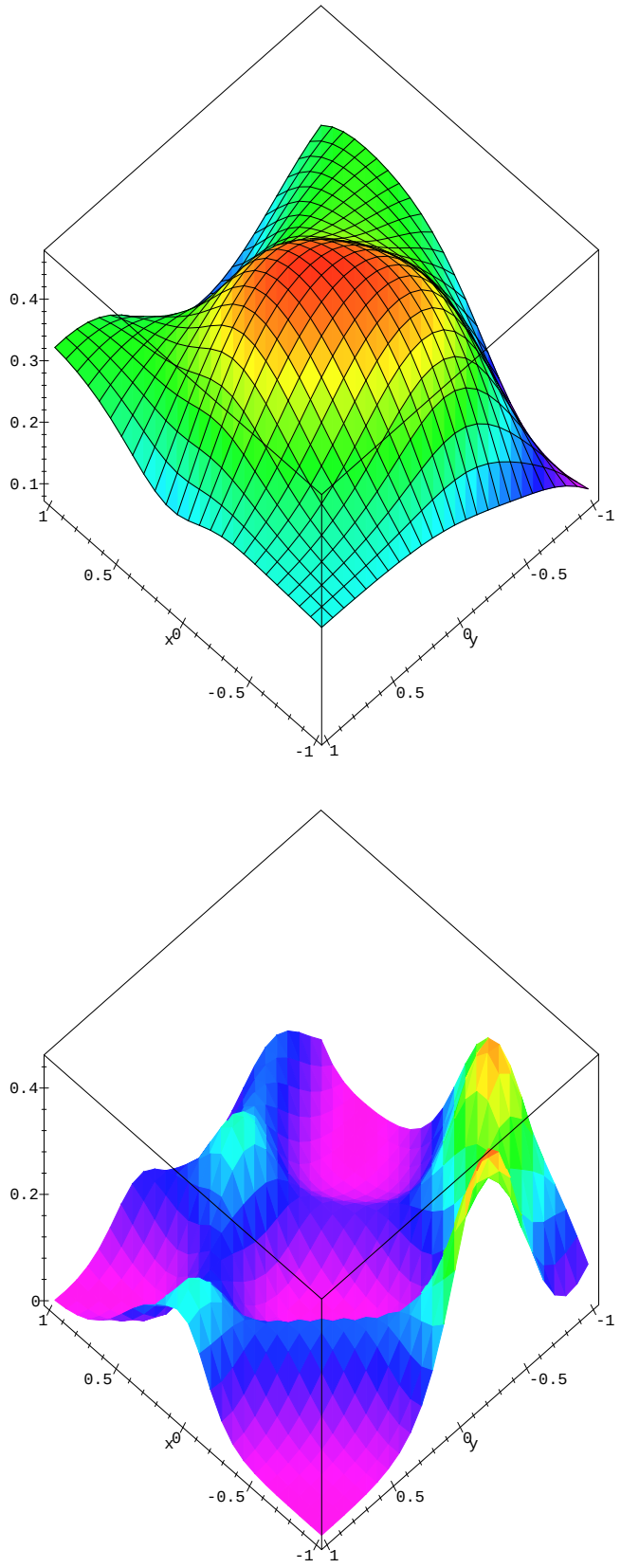


FIG. 5.5 – Action d'un perceptron à une couche cachée de $\mathbb{R}^2$ dans $\mathbb{R}$. L'illustration du haut montre le graphe de $\Psi_W(x)$ en fonction de x (le plan horizontal est l'espace d'entrée et la verticale l'espace de sortie) ; celle du bas montre le graphe de $||\partial_x(\Psi_W)(x^*)||^2$.

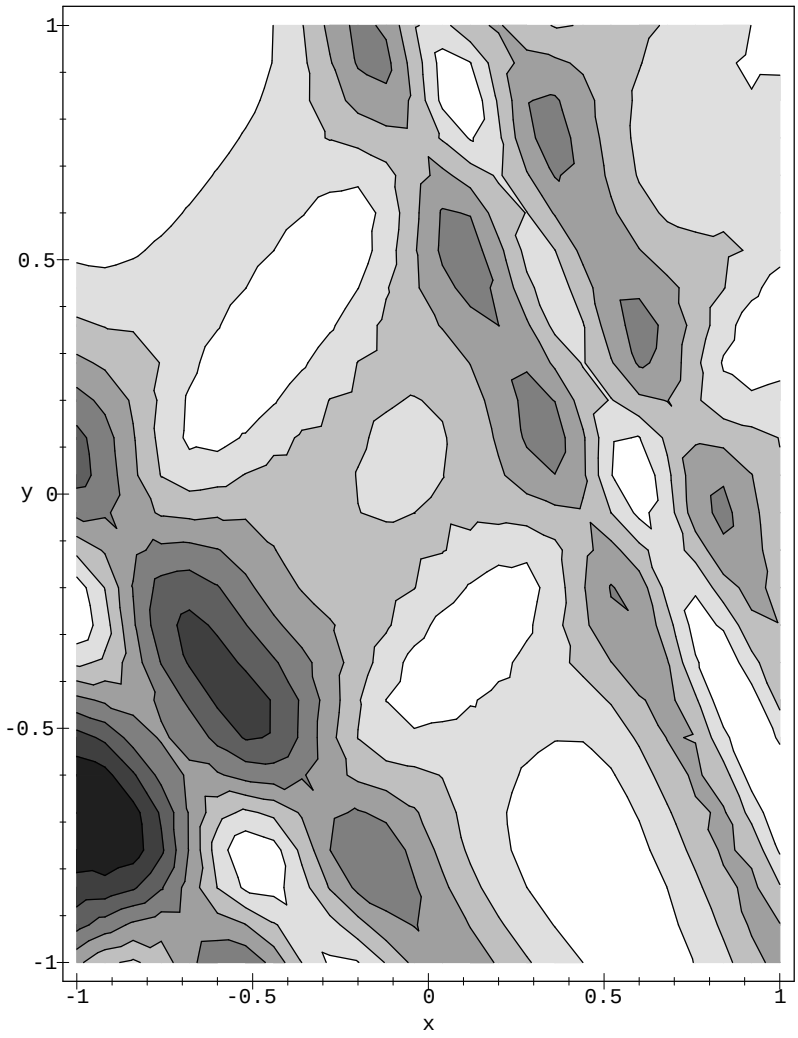


FIG. 5.6 – Les variations du carré de la norme de $H_x(\Psi_W)(x^*)$ en fonction de x^*.

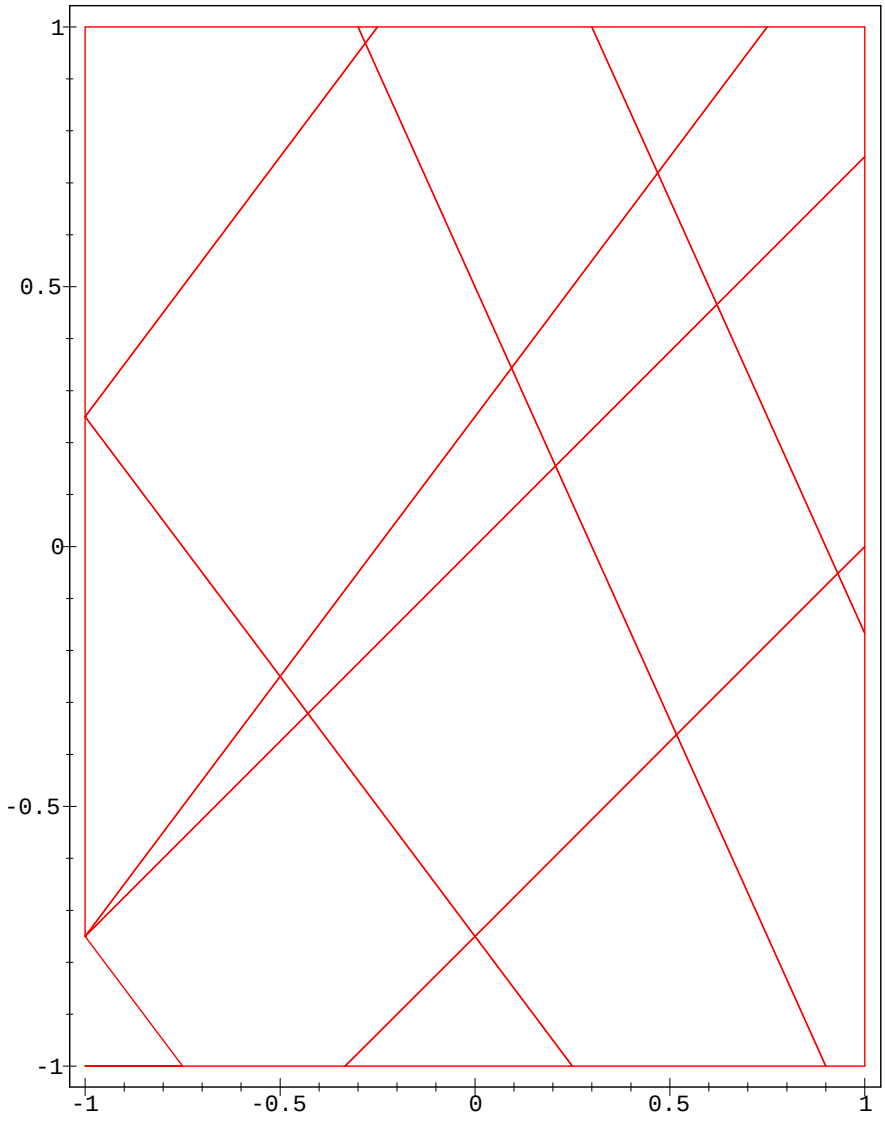


FIG. 5.7 – Une partition de Voronoï de $\mathbb{R}^2$.

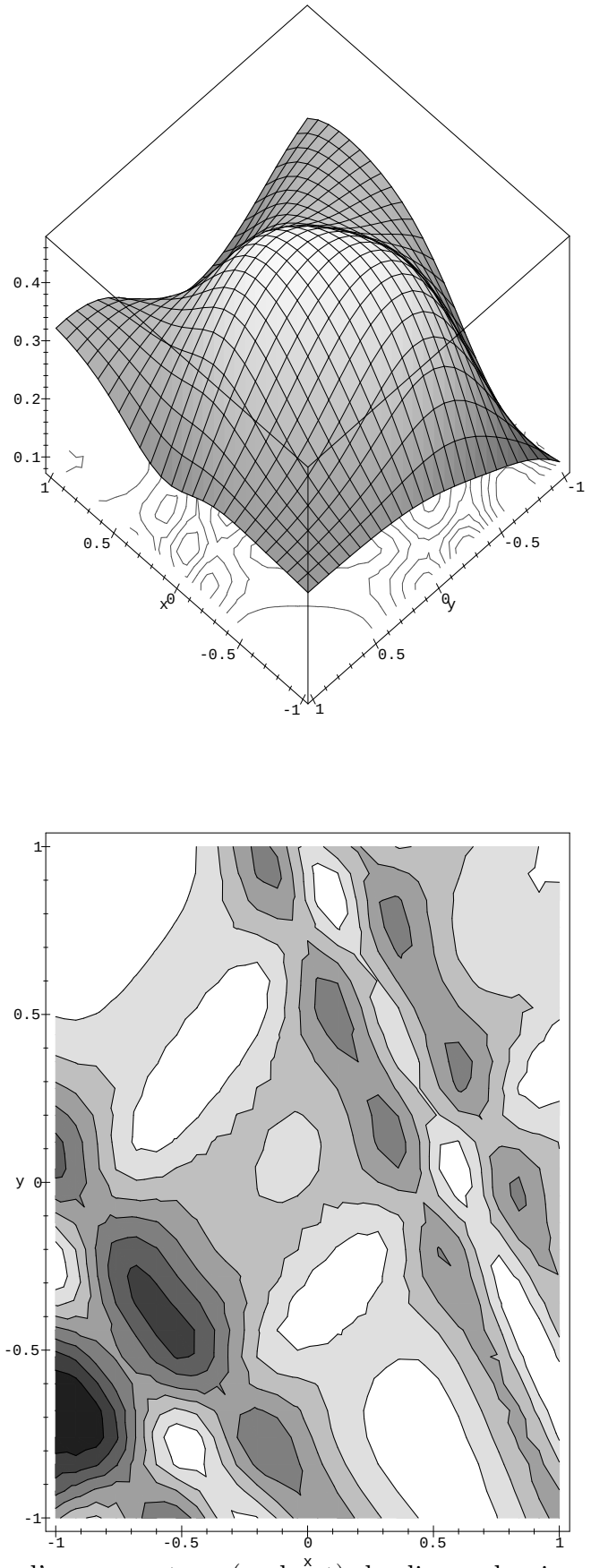


FIG. 5.8 – Le d'un perceptron (en haut), les lignes de niveaux du carré de la norme de sa hessienne (en bas) : plus la norme de la hessienne est petite et plus le comportement du perceptron est proche de son approximation linéaire.

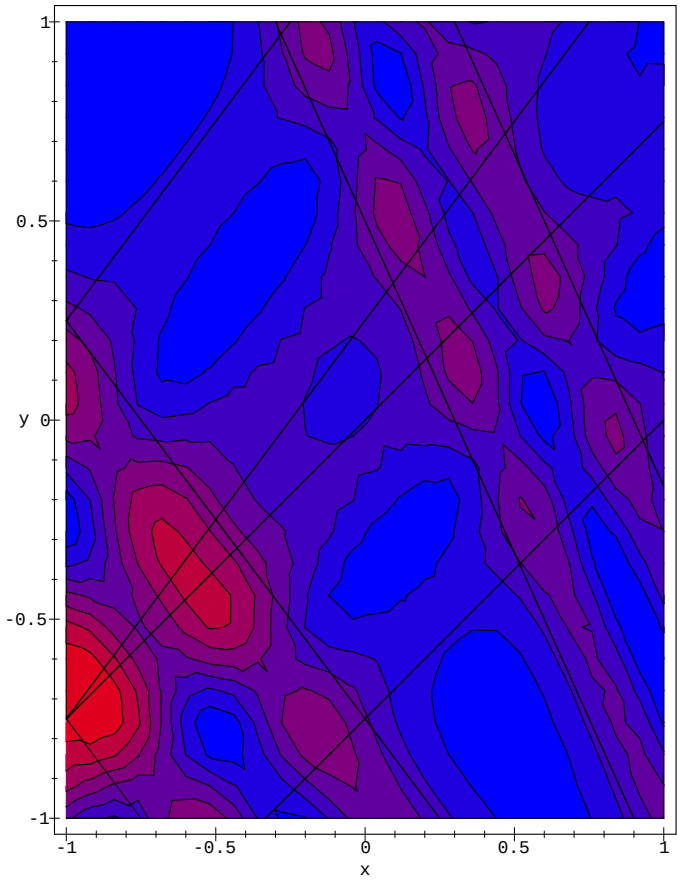


FIG. 5.9 – La norme de la hessienne d'un perceptron superposé avec le découpage en cellules polyhédrales engendré par sa version affine par morceau (i.e. son développement limité à l'ordre 1).

La figure 5.7 montre une partition de Voronoï réalisée à l'aide du package MapleV développé pendant le travail de thèse.

L'illustrations du haut de la figure 5.8 présente l'action d'un perceptron à une couche cachée de 4 neurones de $\mathbb{R}^2$ dans $\mathbb{R}$ ainsi que les lignes de niveaux de la norme de la hessienne associée. Celle du bas superpose les lignes de niveaux du carré de la norme de la hessienne et la partition de Voronoï associée aux minima locaux du carré de la norme de la hessienne.

L'interprétation de ces figures est la suivante : plus un point est proche du centre d'une cellule et plus l'action de Ψ_W sur lui est affine ; plus il est proche des bords et moins c'est le cas.

Ce type d'étude mène à une **meilleure connaissance de l'action des MLP**. Et plutôt que de regarder l'action d'un MLP de forme générale (3.1) et de la linéariser localement en dégageant les cellules de Voronoï, il est possible de considérer une forme "linéarisée" du MLP dès le début du raisonnement, quitte à revenir à la forme générale par la suite.

La "version linéaire" d'un MLP est celle pour laquelle la fonction d'activation φ est affine par morceaux. Le nombre de morceaux peut varier : plus il est nombreux et plus cette "version linéaire" est proche de sa version originale (figure 5.10).

C'est justement ce type d'approche que les Perceptrons Affines Par morceaux (PAP) permettent de formaliser. Comme les sections et chapitres suivants vont l'exposer, les propriétés de cette classe de fonctions la rendent particulièrement intéressante.

Pour garder une relation très forte avec les MLP, une étude des écarts entre un MLP classique et le PAP associé est nécessaire. Une telle approche n'est pas faite dans ce mémoire. Elle permettrait de généraliser explicitement certaines des propriétés des PAP aux MLP. Il est néanmoins possible d'utiliser certains des résultats sur les PAP avec des MLP (comme l'initialisation).

5.2 Définition et comportement des PAP

Les Perceptrons Affines Par morceaux (PAP) sont des perceptrons multicouches dont les fonctions d'activation sont **continues et affines par morceaux**. La forme standard correspond au graphe (2) de la figure 5.10, mais il est tout à fait possible de considérer des PAP aux fonctions d'activation constituées de plus de brisures.

Les résultats exposés ici concernent les PAP avec fonctions d'activation

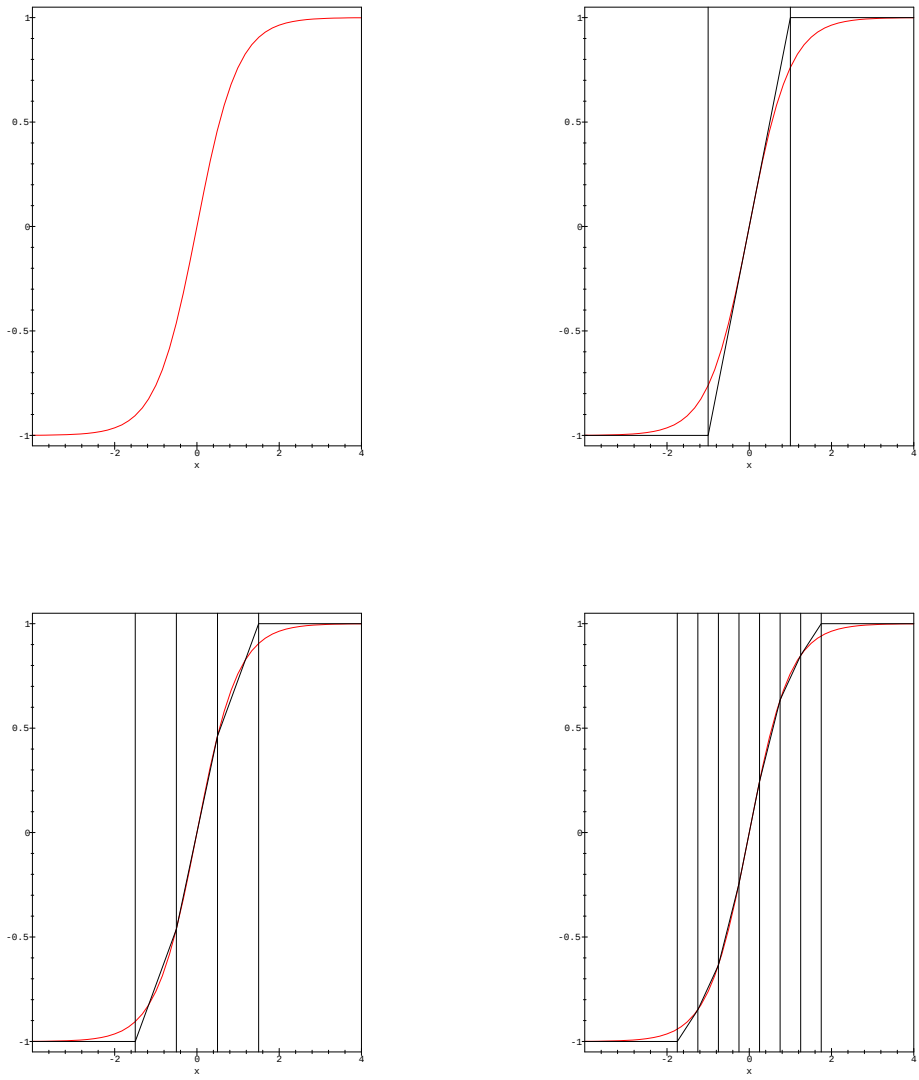


FIG. 5.10 – (1) Fonction d'activation en tanh (perceptron classique) ; (2) fonction d'activation affine sur trois morceaux (PAP classique) ; (3-4) fonction d'activation affine sur 5 et 10 morceaux.

affines sur trois morceaux, ils sont facilement généralisables aux PAP dont les fonctions d'activation sont affines sur plus de morceaux.

Définition 13 (Perceptron Affine par Morceaux (PAP)) *Les percep-*

trons affines par morceaux à n couches cachées de $\mathbb{R}^d$ dans $\mathbb{R}^a$ sont les fonctions de la forme :

$$\Psi_W(x) = \Phi\left(W^{(n)} \cdot \Phi\left(W^{(n-1)} \cdot \ldots \Phi\left(W^{(1)} \cdot x + b^{(1)}\right) + b^{(2)}\right) + \cdots\right) + b^{(n)}\right) \tag{5.3}$$

où x est dans $\mathbb{R}^d$ et :

$$\forall k \in \{1, \ldots, n\},\ W^{(k)} \in \mathcal{M}(c_k, c_{k-1}), b^{(k)} \in \mathbb{R}^{c_k},\ c_1 = d,\ c_n = a$$

et où Φ est une fonction de $\mathbb{R}^{c_k}$ dans lui-même, qui à $y^{(k)} = (y_1^{(k)}, \ldots, y_{c_k}^{(k)})$ associe :

$$\Phi(y^{(k)}) = (\varphi(y_1^{(k)}), \ldots, \varphi(y_{c_k}^{(k)}))$$

avec φ la fonction d'activation du perceptron qui est de la forme : $\varphi(a) = a$ si a est compris entre -1 et 1, $\varphi(a) = -1$ si $a < -1$ et $\varphi(a) = 1$ si $a > 1$.

Les PAP considérés par la suite ont **exactement une couche cachée** sauf précision du contraire. Le nombre d'unités de cette couche cachée est noté c (plutôt que c_2 ; on a en outre $c_1 = d$ et $c_3 = a$).

La première remarque à faire sur les PAP est que le profil de leur fonction d'activation donne lieu à un comportement homogène par zone de l'espace, un peu de façon similaire au découpage en cellules de Voronoï construit dans le chapitre précédent. Ici :

$$\left.\begin{array}{c}\text{l'unité } k \text{ est saturée} \\ \text{par l'entrée } x\end{array}\right\} \Leftrightarrow \underbrace{\left|\left|[W^{(1)} \cdot x + b^{(1)}]_k\right|\right|}_{\left|\left|y_k^{(1)}\right|\right|} < 1 \tag{5.4}$$

Plus précisément, chaque unité k de la couche cachée associe à l'entrée x du PAP la valeur de $\varphi(\langle w, x\rangle + \beta)$ avec w le $k$ème vecteur ligne de la matrice de poids $W^{(1)}$ et β la $k$ème coordonnée du vecteur de seuils $b^{(1)}$. Or :

$$\varphi(\langle w, x\rangle + \beta) = \begin{cases} -1 & \text{si} \quad x \text{ est dans le demi espace de frontière orthogonale à } w \text{ et contenant } \tilde{x}^- \text{ tel que : } \langle w, \tilde{x}^-\rangle + \beta = -1 \\ +1 & \text{si} \quad x \text{ est dans le demi espace de frontière orthogonale à } w \text{ et contenant } \tilde{x}^+ \text{ tel que : } \langle w, \tilde{x}^+\rangle + \beta = +1 \\ \langle w, x\rangle + \beta & \text{sinon} \quad \text{c'est-à-dire entre ces deux demi-espaces} \end{cases} \tag{5.5}$$

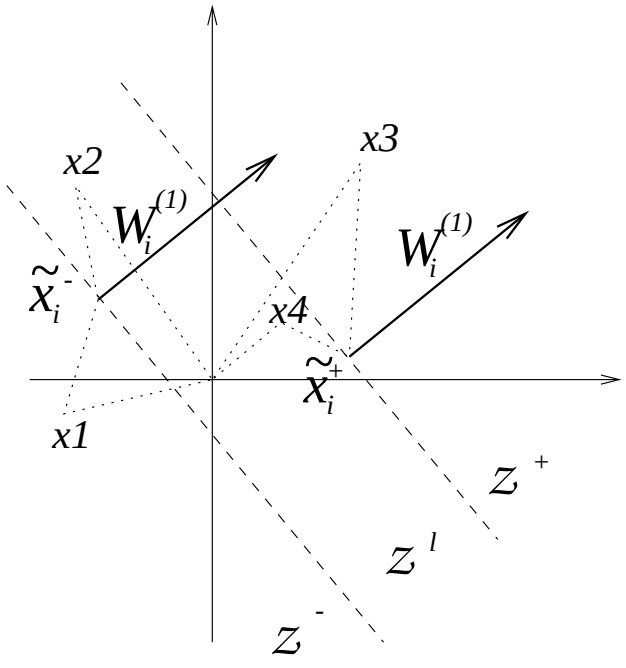
En notant $\mathcal{H}_k^-$ et $\mathcal{H}_k^+$ les hyperplans orthogonaux à la $k$ème ligne de $W^{(1)}$ et passant respectivement par les points $\tilde{x}^-$ et $\tilde{x}^+$; l'action de la première

couche d'un PAP est la combinaison d'actions affines sur chacun des découpages engendrés par les hyperplans $(\mathcal{H}_k^-, \mathcal{H}_k^+)_k$ sur $\mathbb{R}^d$. Ce découpage produit des cellules polyhédrales.

On définit $\mathcal{Z}_k^-$ (respectivement $\mathcal{Z}_k^+$) le demi-espace de frontière $\mathcal{H}_k^-$ (respectivement $\mathcal{H}_k^+$) et passant par $\tilde{x}^-$ (respectivement $\tilde{x}^+$) tel que si x appartient à $\mathcal{Z}_k^-$, $\left\langle W_k^{(1)}, x - \tilde{x}^- \right\rangle < 0$ (respectivement $\left\langle W_k^{(1)}, x - \tilde{x}^+ \right\rangle > 0$). On dira que $\mathcal{Z}_k^-$ est **à gauche** de $\mathcal{H}_k^-$ et que $\mathcal{Z}_k^+$ est **à droite** de $\mathcal{H}_k^+$. $\mathcal{Z}_k^0$ est la bande d'espace comprise entre $\mathcal{H}_k^-$ et $\mathcal{H}_k^+$.

Définition 14 (Partition initiale) *La partition initiale ou partition primaire d'un PAP est composée des cellules polyhédrales provenant des intersections des* $2c$ *demi-espaces* $\mathcal{Z}_k^-$ *et* $\mathcal{Z}_k^+$ *et des* c *bandes d'espaces* $\mathcal{Z}_k^0$ $(1 \leq k \leq c)$.

FIG. 5.11 – Illustration des positions respectives de $\mathcal{Z}_k^-$, $\mathcal{Z}_k^0$, $\mathcal{Z}_k^+$, $W_k^{(1)}$, $\mathcal{H}_k^-$, $\mathcal{H}_k^+ \tilde{x}^-$ et $\tilde{x}^+$. Les points $x1 \ldots 4$ figurent diverses configurations de points dans le plan.



Ces demi-espaces et bandes d'espaces sont formellement caractérisés par

les équivalences suivantes :

$$(5.6)\quad \forall 1 \leq k \leq c \begin{cases} \tilde{x}^+ \text{ est défini par} & : & \left\langle W_k^{(1)}, \tilde{x}^+ \right\rangle = 1 - b_k^{(1)} \\ \tilde{x}^- \text{ est défini par} & : & \left\langle W_k^{(1)}, \tilde{x}^- \right\rangle = -1 - b_k^{(1)} \\ x \in \mathcal{Z}_k^+ & \Leftrightarrow & \left\langle W_k^{(1)}, x - \tilde{x}^+ \right\rangle > 0 \\ x \in \mathcal{Z}_k^- & \Leftrightarrow & \left\langle W_k^{(1)}, x - \tilde{x}^- \right\rangle < 0 \\ x \in \mathcal{Z}_k^0 & \Leftrightarrow & \begin{cases} \left\langle W_k^{(1)}, x - \tilde{x}^- \right\rangle > 0 \\ \left\langle W_k^{(1)}, x - \tilde{x}^+ \right\rangle < 0 \end{cases} \end{cases}$$

Ces relations permettent d'obtenir des résultats comme :

Propriété 1 (Largeur des bandes d'espaces : $\mathcal{Z}_k^0$) *La largeur $l_1(k)$ de la bande d'espace associée à la kème unité de la première couche cachée est :*

$$l_1(k) = \frac{2}{\left|\left|W_k^{(1)}\right|\right|}$$

où $W_k^{(1)}$ est le kème vecteur ligne de la matrice de poids $W^{(1)}$.

Ce résultat vient naturellement des deux premières relations de 5.6 : pour tout x^+ appartenant à $\mathcal{H}_k^+$ et x^- à $\mathcal{H}_k^-$, on a :

$$(5.7)\qquad \left.\begin{array}{rcl} \left\langle W_k^{(1)}, x^+ \right\rangle & = & 1 - b_k^{(1)} \\ \left\langle W_k^{(1)}, x^- \right\rangle & = & -1 - b_k^{(1)} \end{array}\right\} \Rightarrow \left\langle W_k^{(1)}, x^+ - x^- \right\rangle = 2$$

D'autre part, $x^+ - x^-$ peut se décomposer en : $x^+ - x^- = x_{\bar{w}}^{\pm} \cdot \bar{W} + x_w^{\pm} \cdot W_k^{(1)}$, où $\bar{W} \perp W_k^{(1)}$, $\left|\left|\bar{W}\right|\right| = 1$, et $x_{\bar{w}}^{\pm}$ et $x_w^{\pm}$ sont des scalaires. On a alors d'une part :

$$\begin{array}{rcl} \left|\left|x^+ - x^-\right|\right|^2 & = & \left|\left|x_{\bar{w}}^{\pm} \cdot \bar{W} + x_w^{\pm} \cdot W_k^{(1)}\right|\right|^2 \\ & = & (x_{\bar{w}}^{\pm})^2 + (x_w^{\pm})^2 \cdot \left|\left|W_k^{(1)}\right|\right|^2 \end{array}$$

et d'autre part, suite à (5.7) :

$$x_w^{\pm} = \frac{2}{\left|\left|W_k^{(1)}\right|\right|^2}$$

ce qui donne :

$$\left|\left|x^+ - x^-\right|\right|^2 = (x_{\bar{w}}^{\pm})^2 + \frac{4}{\left|\left|W_k^{(1)}\right|\right|^2}$$

qui est minimal lorsque $x_{\bar{w}}^{\pm} = 0$ (ce qui est toujours possible), on a dans ce cas :

$$\left|\left|x^+ - x^-\right|\right| = d(\mathcal{H}_k^+, \mathcal{H}_k^-) = l_1(k) = \frac{2}{\left|\left|W_k^{(1)}\right|\right|}$$

□

Cette propriété montre que les variations de la norme d'un vecteur ligne de $W^{(1)}$ font varier la largeur de la bande d'espace concernée par ce vecteur ligne, conformément à l'illustration de la figure 5.12.

Propriété 2 (Centre d'une unité cachée) *L'hyperplan $\mathcal{H}_k^0$ situé entre les hyperplans $\mathcal{H}_k^+$ et $\mathcal{H}_k^-$ est l'hyperplan othogonal à $W_k^{(1)}$ qui passe par le point :*

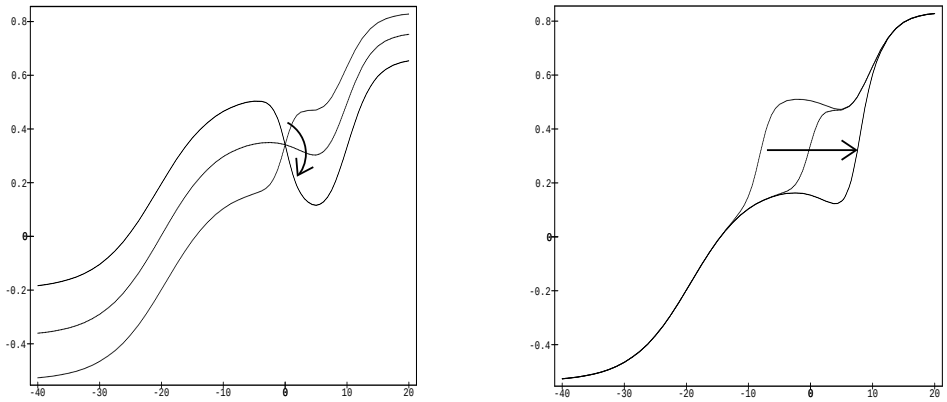
$$x^{(m)} = \frac{b_k^{(1)}}{\left|\left|W_k^{(1)}\right|\right|} \cdot \frac{W_k^{(1)}}{\left|\left|W_k^{(1)}\right|\right|}$$

Cette propriété montre que les variations d'une coordonnée du vecteur de biais $b^{(1)}$ translatent l'action concernée par cette coordonnée, conformément à l'illustration de la figure 5.12.

Ce découpage peut être décrit plus précisément encore, en effet la partition initiale d'un PAP est constituée de toutes les cellules polyhédrales de dimension d des c couples d'hyperplans parallèles : $(\mathcal{H}^+(k), \mathcal{H}^-(k))$. Le nombre maximum de cellules de ce type résultant d'une telle partition peut être calculé.

Pour cela, les définitions et le résultat suivants sont utilisés :

Définition 15 (Cellules polyhédrales et ensemble d'hyperplans) *Soit $\mathcal{E}$ un ensemble de n hyperplans d'un espace vectoriel E de dimension d, on appelle* **cellule de dimension** *p provenant des intersections des éléments de $\mathcal{E}$ tout ensemble P de points de E dont les frontières sont des hyperplans de $\mathcal{E}$, tel que toute intersection non vide de P avec un hyperplan de $\mathcal{E}$ soit une frontière* **et** *tel que le sous espace vectoriel engendré par P dans E soit de dimension p.*



$W^{(1)}$ $b^{(1)}$

FIG. 5.12 – Action d'une variation de $W^{(1)}$ (à gauche) et de $b^{(1)}$ (à droite) en dimension 1.

Par abus de langage : la cellule polyhédrale engendrée par un ensemble d'hyperplans $\mathcal{E}$ sera aussi appelée *intersection de* $\mathcal{E}$.

Définition 16 (Famille générique d'hyperplans) *Un ensemble d'hyperplans est dit générique lorsque tout sous-ensemble de k hyperplans a une intersection de dimension* $(d-k)$ *; lorsque* $(d-k)$ *est négatif, alors leur intersection est vide.*

Théorème 6 (Partition de l'espace par des hyperplans, R.C.Buck) *Le nombre de cellules de dimension p provenant des intersections de n hyperplans constituant une famille* **générique** *dans un espace de dimension d est :*

$$M_d(p,n) = \begin{pmatrix} n \\ d-p \end{pmatrix} \cdot F_p(n+p-d), \text{ avec } F_d(n) = \sum_{k=0}^{d} \begin{pmatrix} n \\ k \end{pmatrix} \tag{5.8}$$

La preuve se trouve dans [BUCK, 50, Nov, 1943] et dans [EDELSBRUNNER, 1987].

L'ensemble des hyperplans $\mathcal{H}^+(k)$ et $\mathcal{H}^-(k)$ lorsque $1 \leq k \leq c$ n'est pas **générique**. En revanche, une notion similaire pour les couples d'hyperplans parallèles peut être construite. En premier lieu il est nécessaire de détailler la définition 16 :

Proposition 1 (Description d'une famille générique) *Un ensemble* $\mathcal{E}$ *de n hyperplans dans un espace de dimension d est générique si et seulement si :*

(i) aucun couple d'éléments de $\mathcal{E}$ ne sont parallèles.
(ii) soit $\mathcal{B}_k$ l'intersection de k éléments quelconques de $\mathcal{E}$, alors aucun $(k+1)$ème élément de $\mathcal{E}$ ne contient $\mathcal{B}_k$; et si $d > k$ tout $(k+1)$ème élément de $\mathcal{E}$ intersecte $\mathcal{B}_k$.

Cette équivalence se démontre en deux temps (seuls les cas où $d \leq 2$ sont non triviaux) :

Le fait que les propriétés (i) et (ii) décrivent des ensembles **génériques** se démontre par récurrence sur k :

1. la propriété est vraie pour $k = 2$, car comme les éléments de $\mathcal{E}$ ne sont pas parallèles entre eux :

$$\forall \mathcal{H}_1, \mathcal{H}_2 \in \mathcal{E}, \dim(\mathcal{H}_1 \cap \mathcal{H}_2) = d - 2$$

2. si il existe $k < d$ tel que :

$$\forall \mathcal{H}_1, \ldots, \mathcal{H}_k \in \mathcal{E}, \dim(\mathcal{H}_1 \cap \ldots \cap \mathcal{H}_k) = d - k$$

alors comme $k < d$ la propriété (ii) implique que comme tout $(k+1)$ème élément de $\mathcal{E}$ intersecte $\mathcal{H}_1 \cap \ldots \cap \mathcal{H}_k$:

$$\dim((\mathcal{H}_1 \cap \ldots \cap \mathcal{H}_k) \cap \mathcal{H}) = d - (k+1)$$

et dès que $k = d$, c'est que $\mathcal{H}_1 \cap \ldots \cap \mathcal{H}_k$ est un point, et comme par (ii) $\mathcal{H}$ ne contient pas ce point :

$$(\mathcal{H}_1 \cap \ldots \cap \mathcal{H}_k) \cap \mathcal{H} = \emptyset$$

Puis le fait qu'une famille **générique** vérifie (i) et (ii) vient aussi par récurrence :

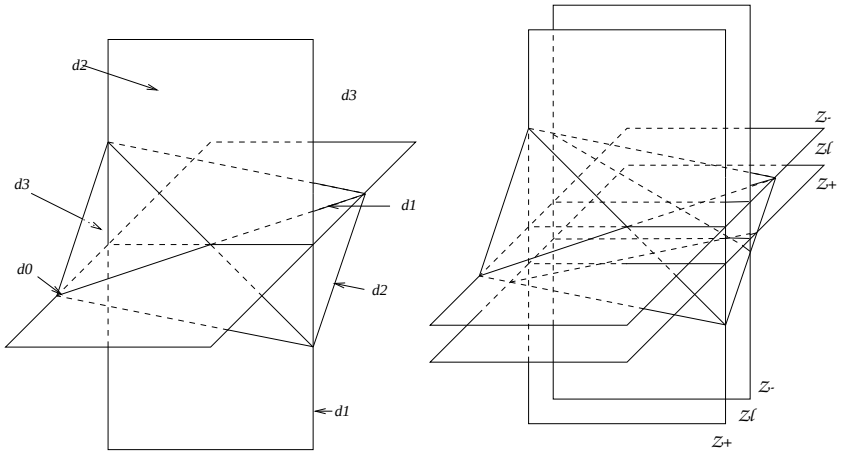
- si deux éléments $\mathcal{H}$ et $\mathcal{H}'$ de $\mathcal{E}$ sont parallèles, alors $\dim(\mathcal{H} \cap \mathcal{H}') = 0$.
- et d'autre part soient $\mathcal{H}_1, \ldots, \mathcal{H}_k$ k éléments quelconques de $\mathcal{E}$, alors si $k < d$: $\dim(\mathcal{H}_1 \cap \ldots \cap \mathcal{H}_k) = d-k$ et $\dim(\mathcal{H}_1 \cap \ldots \cap \mathcal{H}_k \cap \mathcal{H}) = d-(k+1)$ implique que $\mathcal{H}$ intersecte $\mathcal{H}_1 \cap \ldots \cap \mathcal{H}_k$, et si $k \geq d$: si il existait un élément de $\mathcal{E}$ (hors les $\mathcal{H}_1, \ldots, \mathcal{H}_k$) qui contienne $\mathcal{H}_1 \cap \ldots \cap \mathcal{H}_k$, $\dim(\mathcal{H}_1 \cap \ldots \cap \mathcal{H}_k)$ ne serait pas vide.

□

Il est important de noter le théorème suivant :

Théorème 7 (Ensembles génériques d'hyperplans) *Soit $\mathfrak{G}_n$ l'ensemble des familles non génériques d'au plus K hyperplans de $\mathbb{R}^n$ et $\mathfrak{H}_n$ l'ensemble des familles d'au plus K hyperplans de $\mathbb{R}^n$. Alors la mesure de Lebesgue de $\mathfrak{G}_n$ dans $\mathfrak{H}_n$ est nulle.*

FIG. 5.13 – intersection d'hyperplans — intersection d'hyperplans parallèles.



Soit $\mathcal{G}$ un élément de $\mathfrak{G}_n$. Tout hyperplan $\mathcal{H}$ de $\mathcal{G}$ peut être factorisé par une forme linéaire φ et un scalaire α tels que

$$x \in \mathcal{H} \Leftrightarrow \langle \varphi, x \rangle = \alpha$$

Comme $\mathcal{G}$ est dans $\mathfrak{G}_n$, il a au moins une des deux propriétés suivantes :

(i) il existe deux couples (φ, α) et (φ', α') de $\mathcal{G}$ tels que :

$$\varphi - \frac{\alpha}{\alpha'}\,\varphi' = 0 \tag{5.9}$$

(ii) il existe une combinaison de $k+1$ hyperplans ($k < d$) de $\mathcal{G}$ d'intersection vide. Cela implique qu'il existe un système de $k+1$ équations linéaires à d inconnues sans solutions. Comme ces $k+1$ hyperplans ne sont pas parallèles deux à deux, cela veut dire que le déterminant de Cramer associé à ce système d'équations linéaires est nul :

$$\exists(\varphi_1, \ldots, \varphi_{k+1}) \,/\, \det(\varphi_1, \ldots, \varphi_{k+1}) = 0 \tag{5.10}$$

On peut représenter $\mathfrak{H}_n$ comme l'espace vectoriel de toutes les coordonnées des φ et des α qui factorisent les hyperplans d'un de ces éléments. Il s'agit d'un espace vectoriel de dimension $K \times d$.

Par exemple un ensemble de deux hyperplans de $\mathbb{R}^d$ a $2(d+1)$ coordonnées non nulles dans cet espace vectoriel.

La description de $\not\emptyset_n$ dans cet espace vectoriel est l'ensemble généré par toutes les équations polynômiales de type (5.9) et (5.10) : il s'agit d'un fermé de Zariski (famille engendrée par des égalités polynômiales, [Azencott & Berthelor, 1965]), il est donc de Lebesgue-mesure nulle. □

Notons que cette nature particulière des familles d'hyperplans *non génériques* (le fait qu'elles soient de mesure de Lebesgue nulle) permet d'aborder les utilisations numériques des familles d'hyperplans comme si elles étaient toutes *génériques*. En effet, si les valeurs numériques (coordonnées) qui définissent une famille d'hyperplans lui donne la propriété d'être non générique, nous savons désormais qu'une perturbation numériquement "imperceptible" de ces coordonnées engendrera une famille qui, elle, sera générique.

Par analogie avec la définition d'une famille générique d'hyperplans :

Définition 17 (Famille générique de couples d'hyperplans parallèles) *Une famille $\mathcal{E}$ de n couples d'hyperplans $(\mathcal{H}_+, \mathcal{H}_-)$ d'un espace de dimension d où $\mathcal{H}_+$ est parallèle à $\mathcal{H}_-$ est dit* **générique** *lorsque :*

(i) deux éléments de deux couples distincts ne sont pas parallèles.

(ii) si $\mathcal{B}_k$ est l'intersection des bandes contenues entre les hyperplans de k couples, alors aucune des bandes contenues entre les deux éléments d'un $(k+1)$ème couple de $\mathcal{E}$ ne contient $\mathcal{B}_k$; et si $d > k$ toute bande située entre les deux hyperplans d'un $(k+1)$ème couple de $\mathcal{E}$ intersecte $\mathcal{B}_k$.

Par héritage du théorème 7 on a la propriété suivante :

Corollaire 1 (Mesure de Lebesgue des famille non génériques d'hyperplans) *Les familles non génériques de couples d'hyperplans dans l'ensemble des familles de couples d'hyperplans est de mesure de Lebesgue nulle.*

Théorème 8 (Nombre de cellules de la partition initiale d'un PAP) *Le nombre maximal de cellules de la partition initiale d'un PAP de $\mathbb{R}^d$ dans $\mathcal{E}$ ayant N neurones cachés est de :*

$$B_d(N) = \sum_{k=0}^{d} \left(\begin{array}{c} N \\ k \end{array} \right) 2^k \tag{5.11}$$

Le nombre de cellules de cette partition sur lesquelles il y a p unités cachées $(1 \leq p \leq d)$ non saturées et $(N-p)$ constantes est de :

$$T_d(N,p) = \left(\begin{array}{c} N \\ d-p \end{array} \right) \sum_{k=0}^{p} \left(\begin{array}{c} N+p-d \\ k \end{array} \right) \tag{5.12}$$

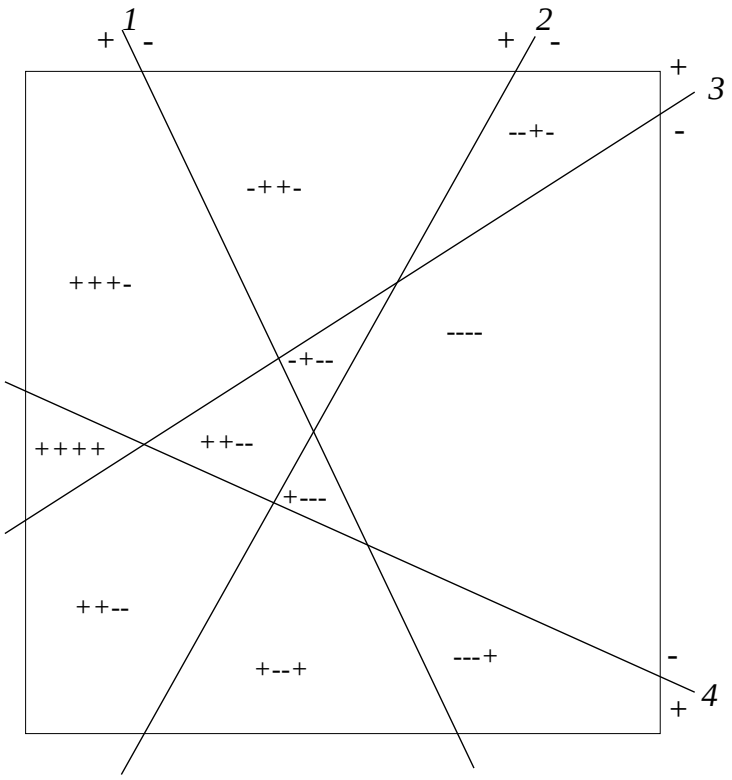


FIG. 5.14 – Vecteurs de positions : quatre droites du plan numérotées et orientées arbitrairement et les vecteurs de position associés à chaque cellule de dimension 2 générée par leur intersection. Les abréviations + et − sont utilisées pour +1 et −1. Les vecteurs de positions des points sur les droites ne sont pas positionnés ; on a par exemple : $p(x_1) = (0 + +-)$ et $p(x_2) = (- - -0)$.

Les bornes $B_d(N)$ et $T_s(N,p)$ sont atteintes.

Ceci est illustré par la figure 5.14. On remarque que pour cet exemple le nombre maximum de cellules de dimension 2 n'est pas $2^4 = 16$ (soit toutes les combinaisons de + et − possibles) mais (conformément au théorème 6) :

$$M_2(2,4) = \begin{pmatrix} 4 \\ 0 \end{pmatrix} \left(\begin{pmatrix} 4 \\ 0 \end{pmatrix} + \begin{pmatrix} 4 \\ 1 \end{pmatrix} + \begin{pmatrix} 4 \\ 2 \end{pmatrix} \right) = 11$$

Remarque.
Ce résultat est particulièrement intéressant car il permet d'avoir un point de vue local extrêmement simple sur l'action d'un PAP : si un PAP émule

une fonction non linéaire, on peut exprimer explicitement son action locale (il s'agit d'une fonction affine sur une cellule polyhédrale que l'on connaît exactement à partir des valeurs des paramètres du PAP). D'autre part, il relie le nombre de neurones cachés du PAP avec le nombre maximal de non linéarités que celui-ci peut émuler. Il est donc possible d'inverser ce calcul et choisir le nombre de neurones cachés en fonction du nombre de non linéarités que l'on a besoin d'émuler. Ceci permet de répondre en partie au problème largement connu du dimensionnement des PAP (et par extension des perceptrons classiques).

Avant de démontrer ce théorème, notons que le nombre maximal de cellules de la partition initiale d'un PAP de $\mathbb{R}^d$ dans $\mathcal{E}$ ayant N neurones cachés est le nombre de cellules de dimension d résultant des intersections de N couples d'hyperplans parallèles $(\mathcal{H}_+(i), \mathcal{H}_-(i))$ formant une famille **générique**.

Ce théorème 8 provient du théorème 5.8. Il est nécessaire de poser quelques notations :

Définition 18 (Vecteurs de position)

- *Soit $\mathcal{E}$ un ensemble de k hyperplans arbitrairement ordonnés et orientés d'un espace vectoriel E de dimension d, on note $p(x) = (p_i(x))_{i\in\{1,\dots,k\}}$ le* **vecteur de position** *de x : la $i$ème coordonnée $p_i(x)$ de $p(x)$ vaut 1 si x est d'un coté du $i$ème hyperplan de $\mathcal{E}$, -1 s'il est de l'autre coté, et 0 s'il appartient à cet hyperplan.*
- *Soit $\mathcal{E}$ un ensemble de k couples d'hyperplans parallèles de E espace vectoriel de dimension d. Pour tout x point de E, on note $\tilde{p}(x) = (\tilde{p}_i(x))_{i\in\{1,\dots,k\}}$ le* **vecteur de position** *de x. La $i$ème coordonnée $\tilde{p}_i(x)$ de $p(x)$ vaut 1 si x est dans le demi-espace de frontière $\mathcal{H}_+(i)$, premier hyperplan du $i$ème couple de $\mathcal{E}$, qui ne contient pas $\mathcal{H}_-(i)$ (second hyperplan du même couple) ; 0 si x est entre $\mathcal{H}_+ + (i)$ et $\mathcal{H}_-(i)$; et -1 sinon.*

- notons $\mathcal{A}$ l'ensemble des cellules constituées par toutes les intersections des $((\mathcal{H}_+(i), \mathcal{H}_-(i))_{i\in\{1,\dots,N\}}$ dans l'espace $\mathbb{R}^d$.
- notons $\mathcal{A}^+$ l'ensemble des cellules constituées par toutes les intersections des $(\mathcal{H}_+(i))_{i\in\{1,\dots,N\}}$ dans $\mathbb{R}^d$.
- on assimile $\tilde{p}$ à la fonction qui associe le vecteur de position dépendant de $\mathcal{A}+$: $\tilde{p}(x)$ à tout x de E .
- on assimile p à la fonction qui associe le vecteur de position dépendant de $\mathcal{A}^+$: $p(x)$ à tout x de E.

En montrant que l'image de $\mathcal{A}$ par $\tilde{p}$ et celle de $\mathcal{A}^+$ par p sont identiques, et comme p et $\tilde{p}$ sont des bijections, on démontre que $\mathcal{A}$ et $\mathcal{A}^+$ ont le même nombre d'éléments.

De cela il vient la formule :

$$T_d(N,p) = M_d(d-p,N)$$

(et donc la formule (5.11)) et le calcul :

$$B_d(N) = \sum_{p=0}^{d} T_d(N,p)$$

Montrons que $\tilde{p}(\mathcal{A}) = p(\mathcal{A}^+)$. La preuve se fait par récurrence sur le nombre k de couples d'hyperplans dont les intersections constituent les éléments de $\mathcal{A}$:

1. pour $k = 1$ on a bien :
$$p(\mathcal{A}^+) = \{(1),(0),(-1)\} = \tilde{p}(\mathcal{A})$$
2. si il existe K tel que, quel que soient les $\mathcal{A}^+$ et $\mathcal{A}$ issus de $k < K$ hyperplans ou couples d'hyperplans : $p(\mathcal{A}^+) = \tilde{p}(\mathcal{A})$, soit alors un ensemble de $(k+1)$ couples d'hyperplans. Soient $\mathcal{A}_k^+$ et $\mathcal{A}_k$ les ensembles de cellules générés par les k premiers d'entre eux.
Ajoutons le $(K+1)$ème couple $(\mathcal{H}_+, \mathcal{H}_-)$; dès qu'un élément de $\mathcal{A}_k^+$ est coupé par $\mathcal{H}^+$, l'élément correspondant de $\mathcal{A}_k$ est coupé par $\mathcal{H}_+$ et $\mathcal{H}_-$ par définition de la simplicité de l'ensemble des hyperplans et couples d'hyperplans dont les intersections constituent $\mathcal{A}_{k+1}$ et $\mathcal{A}_{k+1}^+$.
On a donc bien le diagramme suivant :

$$\begin{array}{rcccc} \tilde{p} & : & \mathcal{A} & \overset{\text{bijection}}{\longrightarrow} & \tilde{p}(\mathcal{A}) \\ & & & & \big|\big|_{\text{égal}} \\ p & : & \mathcal{A}^+ & \overset{\text{bijection}}{\longrightarrow} & p(\mathcal{A}^+) \\ \\ \Rightarrow \tilde{p}^{-1} \circ p & : & \mathcal{A}^+ & \overset{\text{bijection}}{\longrightarrow} & \mathcal{A} \end{array}$$

Equivalence des résultats de Buck et d'Edelsbrunner. Il suffit de montrer l'égalité suivante (dont la partie droite est la formule de Buck [Buck, 50, Nov, 1943] et on trouve la partie gauche dans [Edelsbrunner, 1987]) :

$$\sum_{i=0}^{k} \binom{d-i}{k-i}\binom{n}{d-i} = \sum_{i=0}^{k} \binom{n}{d-k}\binom{n+k-d}{i} \tag{5.13}$$

on a :

$$\sum_{i=0}^{k} \left(\begin{array}{c} d-i \\ k-i \end{array} \right) \left(\begin{array}{c} n \\ d-i \end{array} \right) = \sum_{i=0}^{k} \frac{n!}{(d-k)!\,(k-i)!\,(n-d+i)!}$$

le changement de variable muette i en $(k-i)$ donne

$$\sum_{i=0}^{k} \frac{n!}{(d-k)!\,i!\,(n+k-d-i)!}$$

qui est égal à

$$\sum_{i=0}^{k} \left(\begin{array}{c} n \\ d-k \end{array} \right) \left(\begin{array}{c} n+k-d \\ i \end{array} \right)$$

□

Tous les résultats démontrés jusqu'ici permettent de cerner exactement le comportement d'une couche de PAP.

Pour éclaircir la suite, il est nécessaire de définir les matrices et vecteurs de saturation d'un PAP associés à une cellule particulière :

Définition 19 (Matrice et vecteur de saturation d'un PAP associés à une de ses cellules) *La matrice de saturation (respectivement le vecteur de saturation) de la partition initiale du PAP :*

$$\Psi(x) = \Phi(W^{(2)}\,\Phi(W^{(1)}x + b^{(1)}) + b^{(2)})$$

associée à sa cellule $\mathcal{C}^{(i)}$ est la matrice $\overline{W^{(1)}}^{<i}$ (respectivement le vecteur $\overline{b^{(1)}}^{<i}$) dont la ligne ℓ (respectivement l'élément ℓ) vaut :

- *la ℓème ligne de $W^{(1)}$ (respectivement le ℓème élément de $b^{(1)}$) si $-1 < W_\ell^{(1)}x + b_\ell^{(1)} < +1$ pour x dans $\mathcal{C}^{(i)}$,*
- *une ligne de zéros (respectivement -1 lorsque $W_\ell^{(1)}x + b_\ell^{(1)} \leq -1$ et $+1$ lorsque $+1 \leq W_\ell^{(1)}x + b_\ell^{(1)}$ pour x dans $\mathcal{C}^{(i)}$) dans le cas contraire.*

On a alors :

$$x \in \mathcal{C}^{(i)} \Rightarrow \Phi(W^{(1)}x + b^{(1)}) = \overline{W^{(1)}}^{<i}x + \overline{b^{(1)}}^{<i}$$

Définition 20 (Partition terminale) *La partition terminale d'un PAP est la partition de son espace d'entrées en cellules polyhédrales telle que sur chacune de ces cellules il existe une fonction affine qui coïncide avec le PAP.*

Théorème 9 (Nombre de cellules de la partition finale d'un PAP) *Le nombre maximal de cellules de la partition finale d'un PAP de $\mathbb{R}^d$ dans $\mathbb{R}$ ayant N neurones cachés est :*

$$B_d^f(N) = 3 \sum_{k=0}^{d} \left(\begin{array}{c} N \\ k \end{array} \right) 2^k \tag{5.14}$$

Cette borne est atteinte.

En utilisant les notation de la définition 19, le PAP $\Psi(x) = \Phi(W^{(2)}\,\Phi(W^{(1)}x + b^{(1)}) + b^{(2)})$ peut s'écrire sur une cellule $\mathcal{C}^{(i)}$ comme :

$$\Psi(x) = \Phi(W^{(2)}\overline{W^{(1)}}^{<i}x + W^{(2)}\overline{b^{(1)}}^{<i} + b^{(2)})$$

Comme Ψ est à valeurs dans $\mathbb{R}$, $W^{(2)}$ est une matrice ligne et $b^{(2)}$ un scalaire. L'action de la dernière couche sur $\mathcal{C}^{(i)}$ est donc affine sur trois morceaux : $\mathcal{F}^+$, $\mathcal{F}^0$, et $\mathcal{F}^-$ définis par :

- $\mathcal{F}^-$ est le demi espace contenant les points x tels que :
$$W^{(2)}\overline{W^{(1)}}^{<i}x + W^{(2)}\overline{b^{(1)}}^{<i} + b^{(2)} \leq -1$$
- $\mathcal{F}^-$ est la bande d'espace contenant les points x tels que :
$$-1 \leq W^{(2)}\overline{W^{(1)}}^{<i}x + W^{(2)}\overline{b^{(1)}}^{<i} + b^{(2)} \leq +1$$
- $\mathcal{F}^+$ est le demi espace contenant les points x tels que :
$$+1 \leq W^{(2)}\overline{W^{(1)}}^{<i}x + W^{(2)}\overline{b^{(1)}}^{<i} + b^{(2)}$$

La dernière couche va donc au plus séparer chaque cellule en 3 sous cellules (et ce nombre est atteint pour un PAP dont la partition initiale serait constituée de cellules dont les frontières seraient toutes parallèles et telles que les vecteurs $(W^{(2)}\overline{W^{(1)}}^{<i})'$ soient tous colinéaires à ces frontières).

□

5.3 Représentation des fonctions continues affines par morceaux par les PAP

Une des caractéristiques principales des PAP est qu'ils émulent les fonctions continues affines par morceaux (CAM). Cette section contient la démonstration qu'ils les émulent **toutes**.

Ce résultat nécessite d'abord que deux définitions soient posées :

Définition 21 (Fonction continue affine par morceaux) *Une fonction CAM de $\mathbb{R}^d$ dans $\mathbb{R}^a$ est une fonction continue f et une partition $(\mathcal{C}^{(i)})_{i\in I}$ de $\mathbb{R}^d$ telles que :*

(*i*) *chaque cellule $\mathcal{C}^{(i)}$ est définie $x \in \mathbb{R}^d$ est dans $\mathcal{C}^{(i)}$ si et seulement si $g_k^{(i)}(x) \geq 0$ pour tout k de I_k (où les $g^{(i)}$ sont continues et $g_k^{(i)}(x) = 0$ définit une variété de dimension $d-1$),*

(*ii*) *il existe I fonctions affines $f_i(x) = A_i x + B_i$ telles que $f(x) = f_i(x)$ sur $\mathcal{C}^{(i)}$.*

Définissons en outre une autre famille de fonction affines par morceaux qui est a priori un sous ensemble du précédent :

Définition 22 (Fonction continue affine par polyhèdres) *Une fonction continue affine par polyhèdres (CAP) de $\mathbb{R}^d$ dans $\mathbb{R}^a$ est une fonction continue f et une partition $(\mathcal{C}^{(i)})_{i\in I}$ de $\mathbb{R}^d$ telles que :*

(*i*) *chaque cellule $\mathcal{C}^{(i)}$ est polyhédrale c'est-à-dire définie par : x est dans $\mathcal{C}^{(i)}$ si et seulement si $\left\langle W_k^{(i)}, x - x_k^{(i)} \right\rangle \geq 0$ pour tout k de I_k (où les $W_k^{(i)}$ sont des vecteurs de $\mathbb{R}^d$ et les $x_k^{(i)}$ des points de $\mathbb{R}^d$),*

(*ii*) *il existe I affinités $f_i(x) = A_i x + B_i$ telles que $f(x) = f_i(x)$ sur $\mathcal{C}^{(i)}$.*

(*iii*) *les cellules vérifient un ensemble de contraintes linéaires qui assurent que la juxtaposition des affinités forme une fonction continue.*
Il s'agit d'une égalité par frontière :
Si $\left\langle W_k^{(i)}, x - x_k^{(i)} \right\rangle = 0$ et $\left\langle W_n^{(i)}, x - x_n^{(i)} \right\rangle \geq 0$ pour les $n \neq k$ est la frontière entre les cellules $\mathcal{C}^{(i)}$ et $\mathcal{C}^{(j)}$ issue de la définition de la cellule $\mathcal{C}^{(i)}$; il existe alors une frontière $\left\langle W_\ell^{(j)}, x - x_\ell^{(j)} \right\rangle = 0$ et $\left\langle W_m^{(j)}, x - x_m^{(j)} \right\rangle \geq 0$ pour les $m \neq l$ issue de la définition de $\mathcal{C}^{(j)}$ telle que :

$$\begin{cases} W_\ell^{(j)} &= -W_k^{(i)} \\ x_\ell^{(j)} &= x_k^{(i)} \end{cases}$$

et surtout, on a pour tout point x de cette frontière :

$$A_i x + B_i = A_j x + B_j$$

Qualitativement : les fonctions CAM sont affines sur des cellules aux frontières curvilignes, et les fonctions CAP le sont sur des cellules polyhédrales (dont les frontières sont un ensemble d'hyperplans de son espace de départ).

Le théorème principal de cette section est alors le suivant :

Théorème 10 (Représentation des fonctions continues affines par morceaux par les PAP) *Pour toute fonction continue affine par morceaux (CAM) $\mathcal{A}$ de $\mathbb{R}^d$ dans $[-1,1]^a$ et pour tout hypercube $\mathcal{H}$ fixé de $\mathbb{R}^d$, il existe au moins un PAP émulant exactement $\mathcal{A}$ sur $\mathcal{H}$.*

Le démonstration se fait en plusieurs étapes, la première éclaircissant la nature des fonctions CAM.

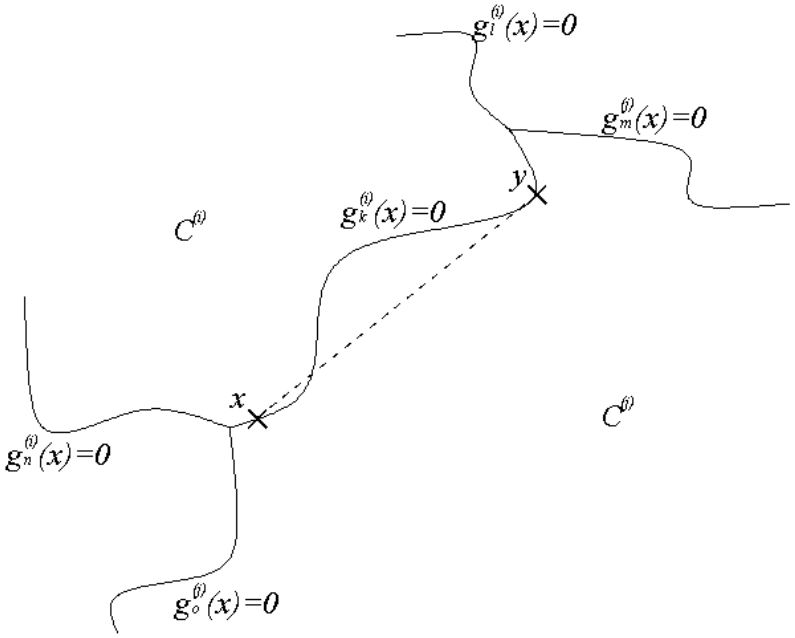


FIG. 5.15 – Deux cellules mitoyennes d'une fonction CAM

Proposition 2 *Toute fonction continue affine par morceaux (CAM) est continue affine par polyhèdres (CAP).*

En effet, soit Q une fonction CAM. Choisissons une des frontières de Q commune à deux cellules $\mathcal{C}^{(i)}$ et $\mathcal{C}^{(j)}$ définie par $g_k^{(i)}(x) = 0$ et $g_\ell^{(i)}(x) \geq 0$ pour $\ell \neq k$, cette frontière est de dimension $d-1$. Soient x et y deux points distincts de cette frontière.

A priori l'intersection entre la frontière et la droite (x, y) est réduite aux points x et y, en revanche les points du segment $[x, y]$ appartiennent soit à $\mathcal{C}^{(i)}$ soit à $\mathcal{C}^{(j)}$. D'autre part, comme ils sont communs aux deux cellules $\mathcal{C}^{(i)}$ et $\mathcal{C}^{(j)}$, x et y vérifient :

$$\left\{ \begin{array}{rcl} A_i x + B_i & = & A_j x + B_j \\ A_i y + B_i & = & A_j y + B_j \end{array} \right. \tag{5.15}$$

Et les points de $[x, y]$ qui sont de la forme : $z = \alpha x + (1-\alpha)y$ ($\alpha \in [0, 1]$) vérifient aussi (par linéarité de (5.15) et de leur définition) : $A_i z + B_i = A_j z + B_j$.

Donc *tout le segment* $[x, y]$ *appartient à la frontière entre* $\mathcal{C}^{(i)}$ *et* $\mathcal{C}^{(j)}$.

Comme la frontière est de dimension $d-1$, il est possible de choisir d points $x_1, \ldots, x_d$ de cette frontière tels que les vecteurs $v_r = x_r - x_1$ ($1 < r \leq d$) soient $d-1$ vecteurs indépendants. Dans la partie de l'espace $g_\ell^{(i)}(x) \geq 0$ pour $\ell \neq k$, le sous espace $\mathcal{F}_{i,j}$ de dimension $d-1$ engendré par ces vecteurs a ses points z qui vérifient :

$$A_i z + B_j = A_j z + B_j$$

Ce sous espace est factorisable par une forme linéaire. Il existe donc un vecteur $V_{i,j}$ tel que : $x \in \mathcal{F}_{i,j} \Leftrightarrow \langle V_{i,j}, z \rangle = 0$. C'est ce vecteur qui va faire office de $W_k^{(i)}$ et $-W_\ell^{(j)}$ dans la définition des fonctions CAP qui s'applique désormais à Q.

□

C'est la contrainte de continuité qui fait hériter aux frontières des cellules compartimentant l'action de Q la nature linéaire de cette action.

Cela illustre que pour construire une fonction CAM, il ne suffit pas de positionner des actions affines en diverses locations de l'espace de départ, la contrainte de continuité implique des relations fortes entre le positionnement des frontières d'une cellule et la fonction qui s'y applique ainsi que celle qui s'applique aux cellules adjacentes.

Ce sont ces relations qui vont permettre de construire pour n'importe quelle fonction CAM un PAP qui l'émule exactement.

Définition 23 (Fonction CAP non triviale) *Une fonction CAP non triviale est une fonction CAP pour laquelle* $\sum_i I_i > 2I$ *(i.e. chaque cellule a en moyenne au moins deux frontières).*

Les fonctions CAP triviales n'ont aucun intérêt dans le cadre du contrôle ou de l'émulation de fonctions par réseaux de neurones. En effet seules les CAP dont les cellules ont très majoritairement plus de 2 frontières vont véritablement émuler des fonctions ayant des non linéarités.

Proposition 3 (Représentation des fonctions CAM par les PAP) *Toute fonction CAM non triviale sur un hypercube centré en zéro et de rayon* $\mathcal{K}$ *de* $\mathbb{R}^d$ *dans* $[-1, 1]$ *est exactement émulable par un PAP.*

Soit Q une fonction CAM. Sa définition nous fournit I cellules polyhédrales, chacune spécifiée par I_i inégalités du type :

$$x \in \mathcal{C}^{(i)} \Leftrightarrow \forall k \in I_i, \left\langle W_k^{(i)}, x - x_k^{(i)} \right\rangle \geq 0$$

Construisons une matrice $\mathcal{M}$ et un vecteur $\mathcal{B}$ de $\mathcal{J} = \sum_i I_i$ lignes telle que chaque ligne $\mathcal{M}_\ell$ de $\mathcal{M}$ soit proportionnelle à un des $W_k^{(i)}$ et chaque élément de $\mathcal{B}$ vérifie :

$$\left\{ \begin{array}{lcl} \mathcal{M}_\ell & = & \alpha_\ell W_k^{(i)} \\ \mathcal{B}_\ell & = & 1 - \mathcal{M}_\ell \, x_k^{(i)} \end{array} \right. \tag{5.16}$$

Pour le neurone associé, la saturation à $+1$ se produit en $\tilde{x}^+$ vérifiant :

$$\begin{array}{lll} & \mathcal{M}_\ell \, \tilde{x}^+ + \mathcal{B}_\ell & = 1 \\ \Leftrightarrow & \alpha_\ell W_k^{(i)} (\tilde{x}^+ - x_k^{(i)}) + 1 & = 1 \\ \Leftrightarrow & \left\langle W_k^{(i)}, \tilde{x}^+ - x_k^{(i)} \right\rangle & = 0 \end{array}$$

Afin de ne pas perturber la construction du PAP émulant Q, faisons en sorte que l'autre frontière engendrée par la ligne ℓ soit rejetée hors de l'hypercube qui nous intéresse.

Un point $\tilde{x}^-$ de cette seconde frontière est défini par $\alpha_\ell W_k^{(i)} (\tilde{x}^- - x_k^{(i)}) = -2$ et est situé à une distance $2/\left|\left|\mathcal{M}_\ell\right|\right|$ de $\tilde{x}^+$. Si cette distance est plus grande que $2\mathcal{K}$, l'inégalité triangulaire nous garantit que $\tilde{x}^-$ sera hors de l'hypercube centré sur zéro et de rayon $\mathcal{K}$. Assurons nous donc que $2/\left|\left|\mathcal{M}_\ell\right|\right| > 2\mathcal{K}$, i.e. que :

$$|\alpha_\ell| < \left(\mathcal{K} \left|\left| W_k^{(i)} \right|\right| \right)^{-1}$$

Si on a contraint la définition de Q à ne comporter que des vecteurs $W_k^{(i)}$ de norme égale à 1 (ce qui est tout à fait possible), alors cela revient à prendre α_ℓ indépendant de ℓ toujours égal à $\mathcal{K}^{-1}$.

Nous disposons donc d'une matrice $\mathcal{M}$ et d'un vecteur $\mathcal{B}$ tels que les frontières définies par la partition initiale du perceptron Ψ qui a pour sa première couche $\mathcal{M}$ comme matrice de poids $\mathcal{M}$ et $\mathcal{B}$ comme vecteur de seuils coïncident avec les cellules de la partition de Q.

En utilisant les définitions 19 de matrices et de vecteurs de saturation, et en tenant compte que la proposition 3 ne concerne que les fonctions CAM à valeurs dans $[-1, 1]$, on cherche une matrice ligne $\mathfrak{L}$ (qui a $J = \sum_i I_i$ éléments) et un réel β tels que la fonction Q soit émulée par :

$$\Psi(x) = \Phi(\mathfrak{L} \, \Phi(\mathcal{M}x + \mathcal{B}) + \beta)$$

Sur chaque cellule $\mathcal{C}^{(i)}$ (il y en a I), l'action de Q est $A_i x + B_i$. Pour que Ψ émule Q il suffit de résoudre les deux équations suivantes[1] :

$$\left\{ \begin{array}{rcl} \mathfrak{L}\,\overline{\mathcal{M}}^{<i} & = & A_i \\ \mathfrak{L}\,\overline{\mathcal{B}}^{<i} + \beta & = & B_i \end{array} \right. \tag{5.17}$$

Au total nous avons donc un système de $2I$ équations linéaires pour $\sum_i I + 1$ inconnues à résoudre. Ce qui est évident dès que :

(i) chaque cellule a au moins en moyenne deux frontières, ce qui est le cas pour les fonctions CAM non triviales,

(ii) **et** qu'il n'y a pas deux équations issues de (5.17) dont les membres de gauche soient proportionnels et pas les membres de droite (déterminant de Cramer nul pour le système à résoudre).

Le processus de construction de la matrice $\mathcal{M}$ et du vecteur $\mathcal{B}$ décrit jusqu'à présent peut assez naturellement engendrer des situations de type (ii). Il est néanmoins facile de les contourner car en ajoutant une ligne à la matrice $\mathcal{L}$, on ajoute une inconnue à $\mathcal{L}$ (on passe de $\sum_i I + 1$ à $\sum_i I + 2$ inconnues). Dès qu'on a deux équations dont les membres de gauche sont proportionnels (par exemple le premier et égal à γ fois le second), il suffit donc d'ajouter une ligne à $\mathcal{M}$ dont les coordonnées m_k et $m_{k'}$ qui vont se retrouver dans les équations incriminées ne vérifient pas $m_k = \gamma\, m_{k'}$ et de telle sorte que les frontières de la bande d'espace que cette nouvelle ligne engendre soient en dehors de l'hypercube qui nous intéresse (ce qui exige seulement de contrôler la norme de cette nouvelle ligne de $\mathcal{M}$, ce qui peut toujours être fait indépendamment de la contrainte $m_k \neq \gamma\, m_{k'}$).

On obtient finalement une matrice $\mathcal{M}$ et un vecteurs $\mathcal{B}$ avec plus de lignes, mais qui assurent que (ii) est contournée.

□

La proposition 3 prouve le théorème 10. □

Remarque. La restriction du résultat aux fonctions CAM non triviales peut être contournée par le fait que les cellules à une seule frontière (seule leur présence en grande proportion peut générer des CAM triviaux) sont nécessairement "au bord" de l'hypercube sur lequel on travaille. Il est donc possible de ramener toute fonction CAM triviale à une version non triviale en restreignant le rayon de l'hypercube considéré. Les seules fonctions CAM

[1] Le problème de saturation sur la dernière couche ne se pose pas : au plus cela transforme quelques unes de ces égalités en inégalités, ce qui est plus facile à résoudre.

triviales échappant à cette transformation sont les fonctions à deux cellules, particulièrement inintéressantes.

5.4 Initialisation d'un perceptron

5.4.1 Représentation et initialisation : cas du contrôle non linéaire

La motivation de ces résultats a été de pouvoir générer l'initialisation d'un PAP capable de contrôler un système dynamique donné pendant une durée suffisante à une première itération d'apprentissage par descente de gradient.

En effet, la méthode d'initialisation classique des perceptrons (aléatoire) ne permet pas d'observer l'action de contrôle généré assez longtemps pour faire quoi que ce soit. C'est une des raisons pour lesquels les réseaux de neurones sont assez peu utilisés pour contrôler des système dynamiques dans un espace de dimension supérieure à 6 ou 7.

Il est en effet assez simple d'initialiser un PAP émulant localement quelques fonctions affines en quelques points de l'espace d'état du système à contrôler.

C'est cette méthode qui a été utilisée dans un cadre applicatif : trois zones de l'espace d'état d'un moteur essence ont été isolées (proche du "calage", régime transitoire et point de fonctionnement classique du régime ralenti) et un contrôleur LQI a été réalisé en utilisant un modèle linéaire frustre du moteur autour de chacune de ces zones.

Un PAP a ensuite été initialisé afin d'émuler ces trois contrôleurs "locaux", il a servi de point d'initialisation pour la suite.

Deux points forts ressortent de cette méthodologie d'initialisation :

- les experts connaissent souvent plusieurs modèles linéaires localement fiables du système à contrôler,
- on contrôle le dimensionnement du PAP non pas par des variables intermédiaires aux problèmes étudiés (taille de la couche cachée ou écart type d'un tirage aléatoire par exemple), mais par des grandeurs directement accessibles (partitionnement de l'espace d'état, pentes, etc).

Bien entendu ces résultats, s'il sont exacts pour les PAP, sont encore qualitativement juste pour les perceptrons plus classiques. La méthode d'initialisation décrite ici d'étend à cette plus large famille de fonctions.

5.5 Les directions possibles

Dans la mesure où l'objectif ici est de construire un PAP destiné à émuler un contrôle non linéaire, l'intérêt de l'utilisation des PAP dans d'autres directions n'a pas été exploré.

Néanmoins, il y a un gain à les utiliser dans d'autres cadres puisque :

- en utilisant la description en cellules de l'espace de départ, il est possible de positionner un neurone dans une zone précise de l'espace (par exemple en cas d'underfitting).
- il est aussi possible que la traduction en termes statistiques de l'identification par PAP mène à des résultats assez clairs. Par exemple en terme de robustesse face à des données bruitées.
- d'autre part, il est envisageable de transférer certaines propriétés des PAP aux perceptrons plus classiques : par exemple d'initialiser un perceptron avec les valeurs calculées pour un PAP.

Chapitre 6

Apport des PAP à l'étude de systèmes dynamiques : comportement des PAP en boucle fermée

6.1 Comportement asymptotique du système dynamique généré par un PAP en boucle fermée

De nombreux auteurs utilisent des RNF pour émuler un système dynamique (par exemple [Tan & Yu, 1993] qui utilisent un réseau RBF, ou encore [Polycarpou & Ioannou, 1991]et [Pazman, 1993]). Ces RNF sont par la suite utilisés notamment dans le cadre de la mise en place d'un contrôle prédicitif du système émulé par le RNF.

Le but de ce chapitre est d'étudier les trajectoires du système dynamique de $\mathbb{R}^2$ constitué du PAP (6.2) à quatre unités cachées bouclé sur lui-même.

L'espace d'état $\mathcal{X}$ est donc l'hypercube centré en zéro et d'arête de longueur 2 de $\mathbb{R}^2$, et le système dynamique est :

$$x_{n+1} = \Psi_W(x_n) \tag{6.1}$$

où $\Psi_W(x_n)$ le PAP de formule explicite :

$$\Psi_W(x) = \Phi\left(W^{(2)}\,\Phi\left(W^{(1)} \cdot x + b^{(1)}\right) + b^{(2)}\right) \tag{6.2}$$

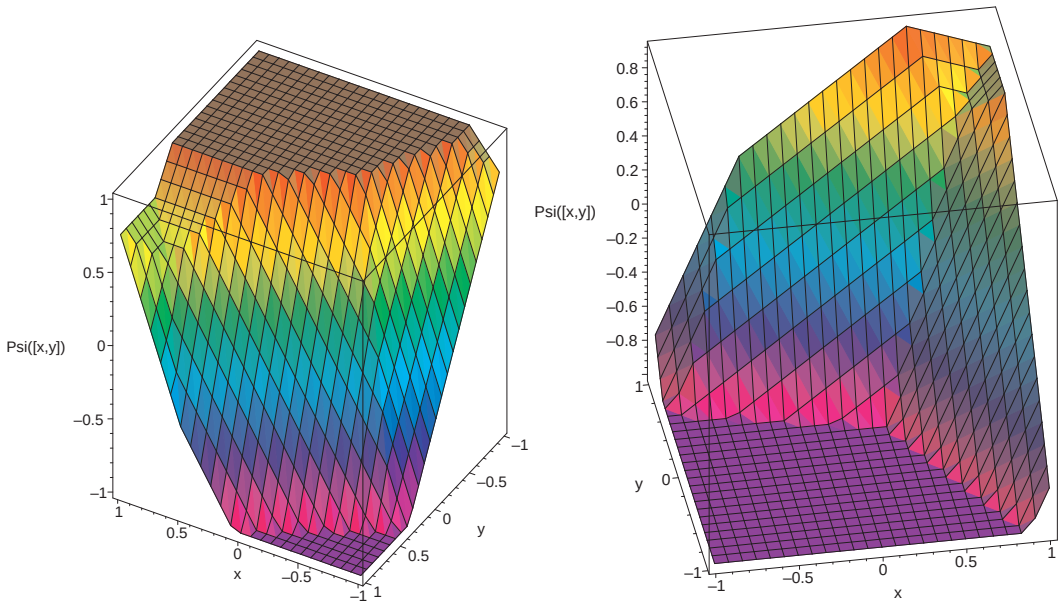


FIG. 6.1 – Schéma du PAP étudié : il va de $\mathbb{R}^2$ dans $\mathbb{R}^2$, sa première sortie est à gauche et la seconde à droite.

avec :

$$W1 = \begin{bmatrix} 4/3 & -4/3 \\ 1 & 1 \\ 4/7 & 0 \end{bmatrix} b1 = \begin{bmatrix} 1 \\ -1/2 \\ -1/7 \end{bmatrix} W2 = \begin{bmatrix} 1 & -1 & 7/4 \\ -1 & 1 & 7/4 \end{bmatrix} b2 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{6.3}$$

Ces matrices et vecteurs ont été choisis afin de générer des séparations claires de l'espace.

On va voir tout au long de cette partie comment l'étude d'un système dynamique émulé par un PAP peut se ramener à l'étude d'une chaîne de Markov. Au delà de l'obtention de résultats comportementaux sur le système dynamique sous jacent, cela préfigure les opportunités qu'ouvrent l'utilisation des PAP dans les systèmes hybrides en ramenant les composantes continues d'un tel système dans un espace d'état discret, simplifiant ainsi son étude.

6.1.1 Partitions de l'espace de départ en cellules polyhédrales

Cette section comporte des illustrations provenant d'un "package" informatique pour le logiciel Maple réalisé par mes soins dans le cadre de cette

thèse. Les principales API de ce package sont brièvement décrites en fin de section.

Les cellules peuvent être devinées en dimension faible en traçant le graphe de Ψ_W : figure 6.1.

On peut d'autre part calculer les équations des hyperplans (ici des droites) correspondants à la partition initiale engendrée par Ψ_W :

$$\left\{\begin{array}{rcr} -x+y & = & -1 \\ -x+y & = & 1 \\ y-1 & = & -1 \\ y-1 & = & 1 \\ y+1 & = & -1 \\ y+1 & = & 1 \\ 2x-1 & = & -2 \\ 2x-1 & = & 2 \end{array}\right. \tag{6.4}$$

Partition terminale. Pour chacune des cellules polyhédrales de la figure 6.2 (en haut), l'action de $x \mapsto \Phi(W^{(2)} \cdot x + b^{(2)})$ va de nouveau générer un découpage (même figure mais illustration d'en bas).

6.1.2 Action du PAP cellule par cellule

Il reste à voir l'action de Ψ_W sur chacune des cellules. Les illustrations de la figure 6.2 montrent chaque cellule de la partition terminale et son image, superposées.

La transformation de 4 cellules a été retenue pour figurer dans ce mémoire : il s'agit des cellules numérotées 3, 5, 7 et 9 en raison de leur rôle dans la chaîne de Markov engendrée par les transitions de cellule à cellule. En effet la cellule 3 fait partie de la distribution limite, les cellules 5 et 7 sont représentatives des cellules majoritairement présentes et la cellule 9 est celle qui communique avec le plus d'autres cellules.

Cellule 9. Les deux illustrations relatives à la cellule 9 (comme pour chaque cellule) sont :

En haut : les frontières de la partition finale du PAP (en rouge) et l'image de la cellule numéro 9 par la première couche du PAP (en noir). Des traits pointillés joignent les sommets de $\mathcal{C}^{(9)}$ et son image par la première couche du PAP : $W^{(1)}\mathcal{C}^{(9)} + b^{(1)}$.

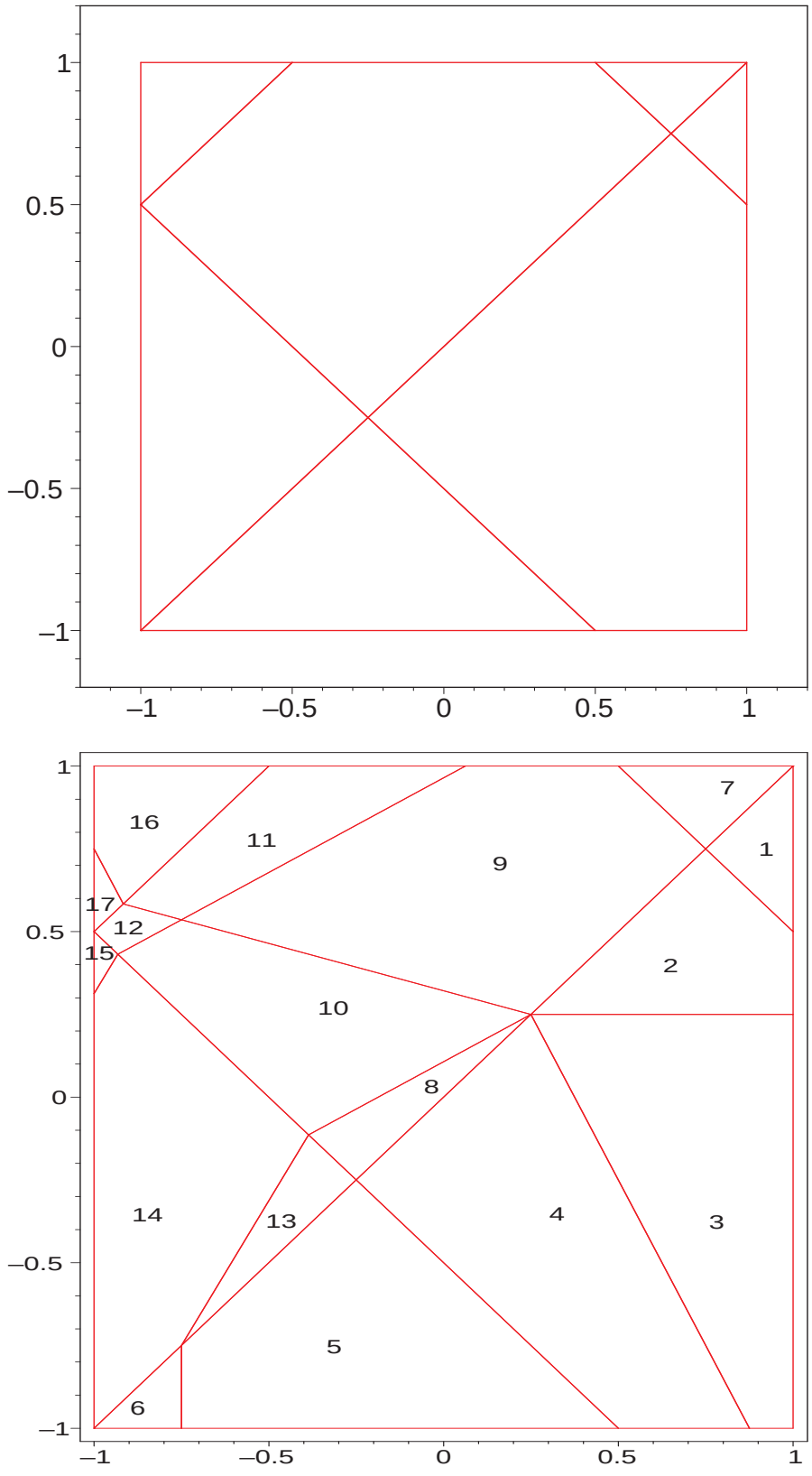


FIG. 6.2 – Partition primaire (en haut) et partition terminale (en bas) du PAP. Les cellules de la partition terminales sont numérotées.

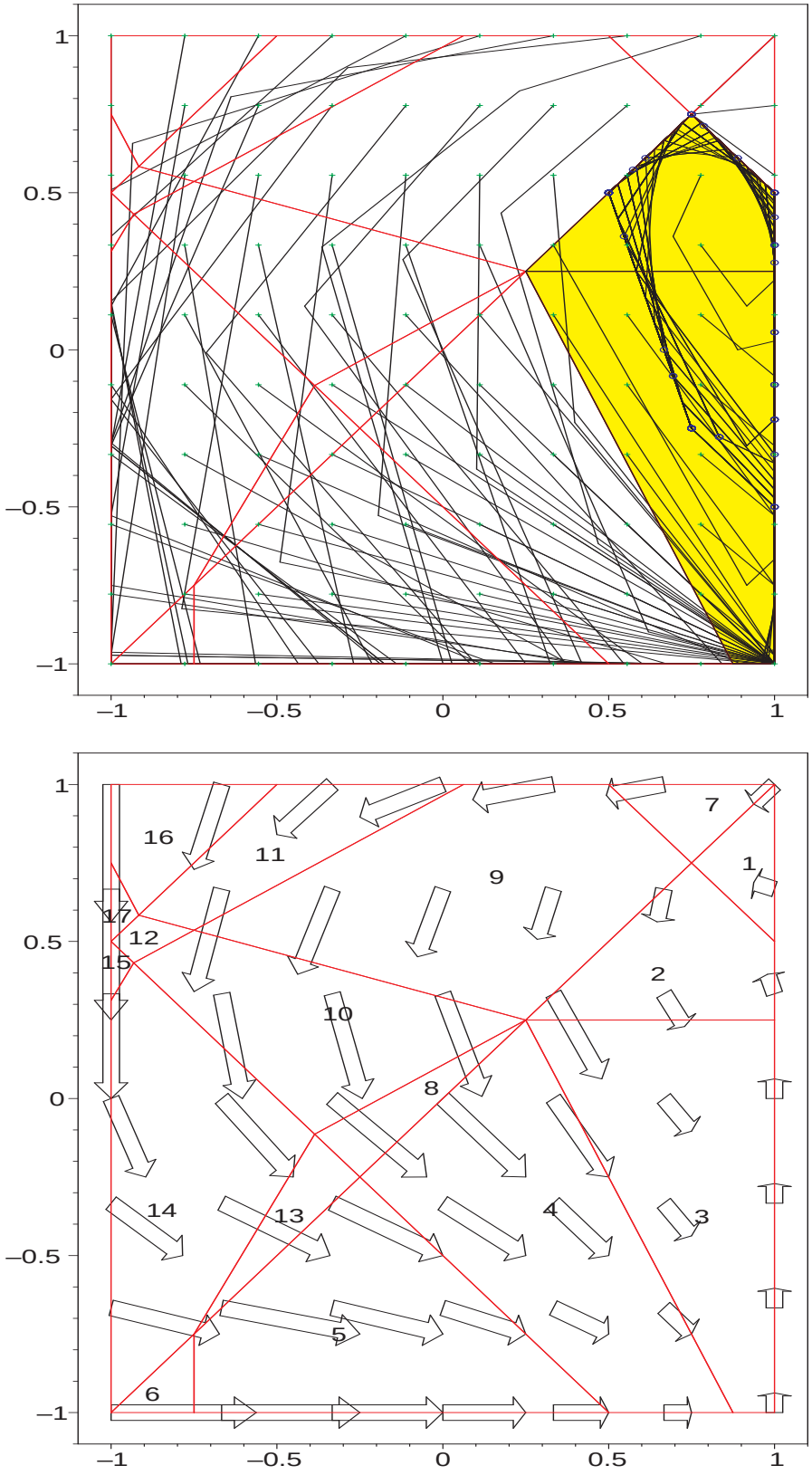


FIG. 6.3 – En haut : superposition des cellules de la partition terminale du PAP et des trajectoires de son bouclage ; les cellules qui contiennent les fins de trajectoires sont en jaune. En bas : le champ de vecteurs généré par le système dynamique du PAP bouclé.

En bas : les frontières de la partition finale du PAP (en rouge) et l'image de la cellule numéro 9 par le PAP (en noir). Des traits pointillés joignent les sommets de $\mathcal{C}^{(9)}$ et son image par le PAP : $\Psi_W(\mathcal{C}^{(9)})$.

L'action de la première couche du PAP l'envoie sur les cellules 1, 2, 7, 8, 13 et sur une partie des cellules 9, 10, 3, 4 et 5.

L'action totale du PAP la fait recouvrir un encore plus grand nombre de cellules.

Cellule 3. On voit donc que la première couche envoie $\mathcal{C}^{(3)}$ hors de $\mathcal{X}$: l'image finale de $\mathcal{C}^{(3)}$ est sur une droite le long de la frontière de $\mathcal{C}^{(3)}$ et $\mathcal{C}^{(2)}$ commune à $\mathcal{X}$.

Cellule 5. La cellule 5 est envoyée sur un point au coin de $\mathcal{C}^{(3)}$.

Cellule 7. La cellule 7 est envoyé sur les cellules 9 et 11.

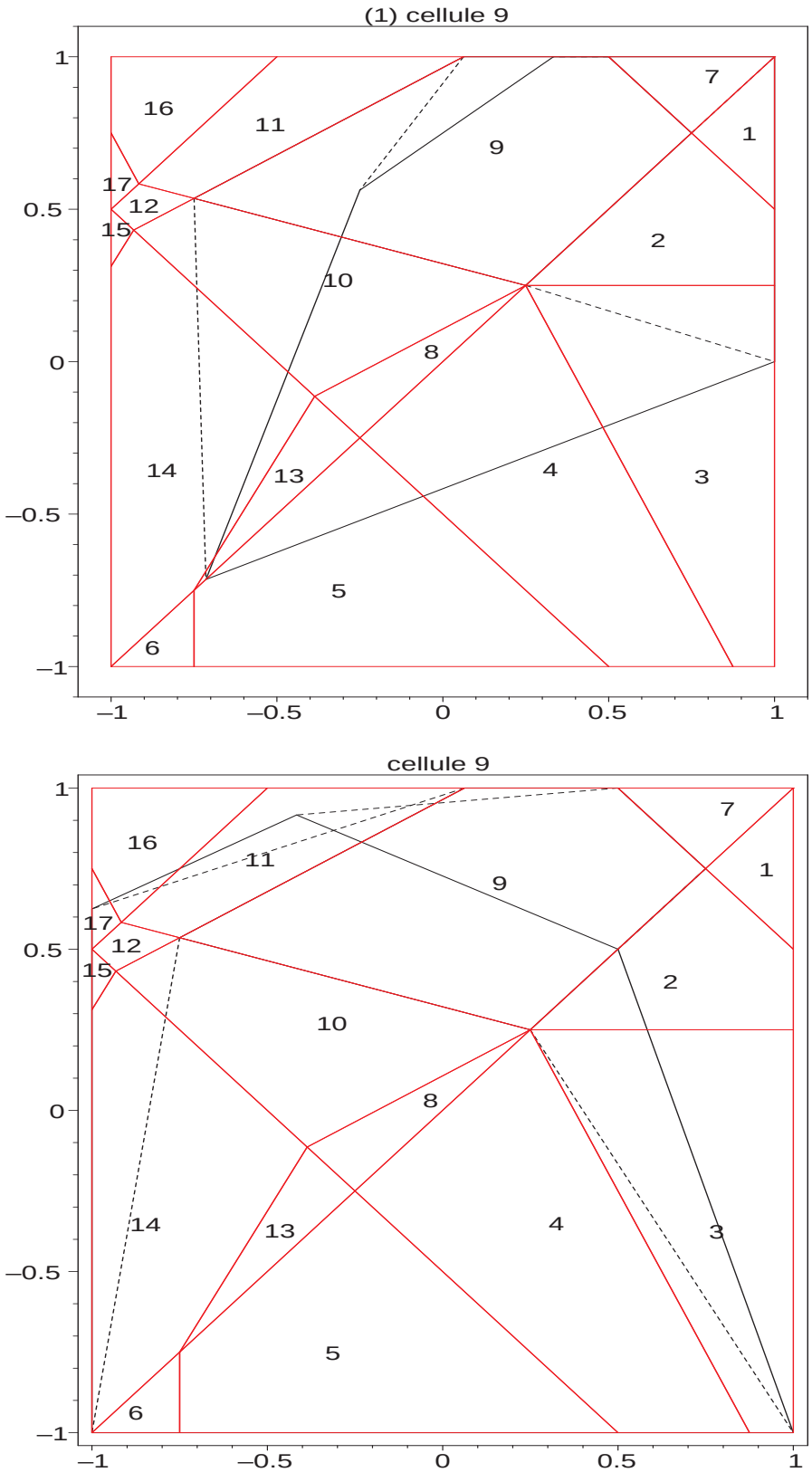


FIG. 6.4 – Action de la première couche affine du PAP (en haut) et du PAP tout entier (en bas) sur la cellule 9 de sa partition terminale.

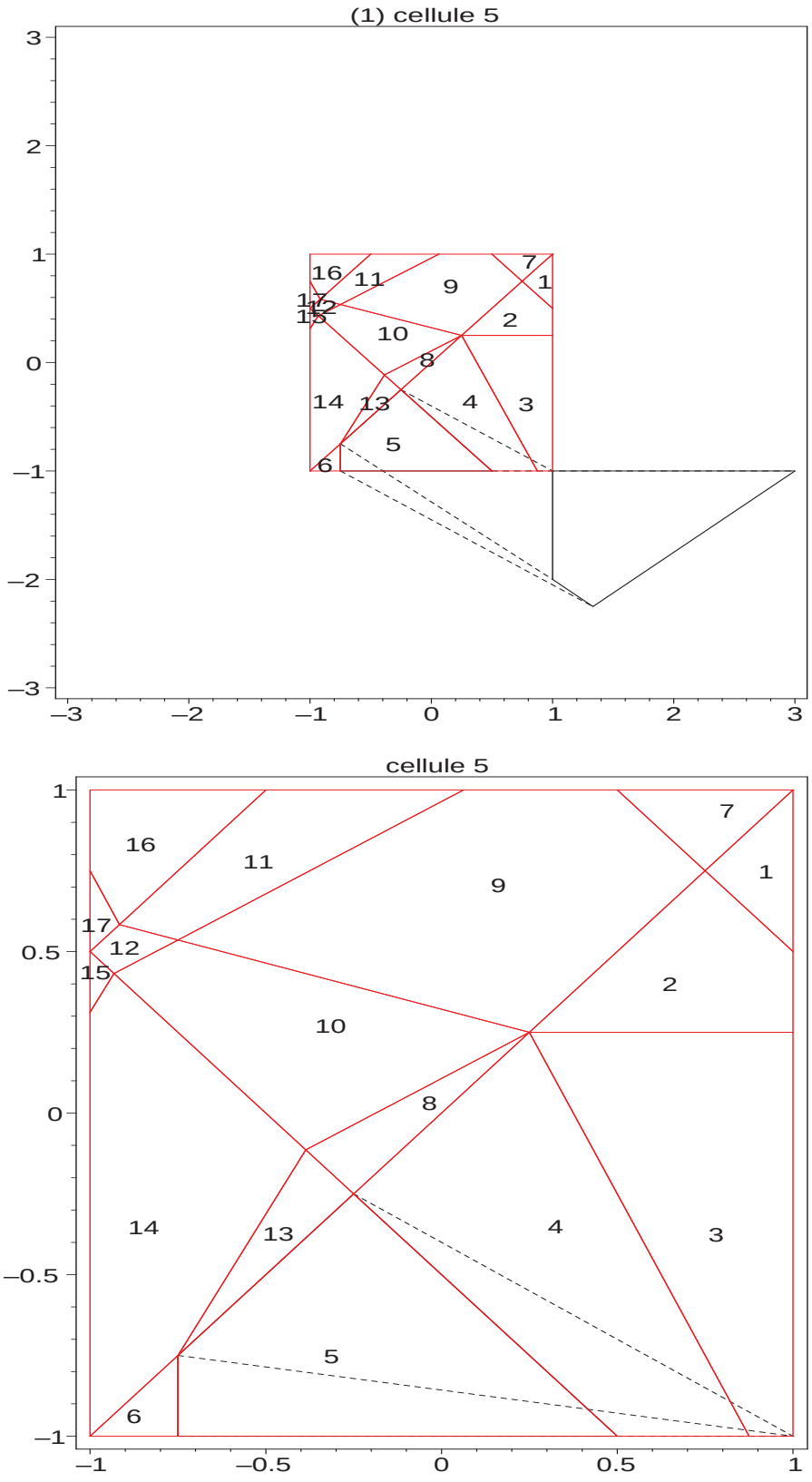


FIG. 6.5 – Action de la première couche affine du PAP (en haut) et du PAP tout entier (en bas) sur la cellule 5 de sa partition terminale.

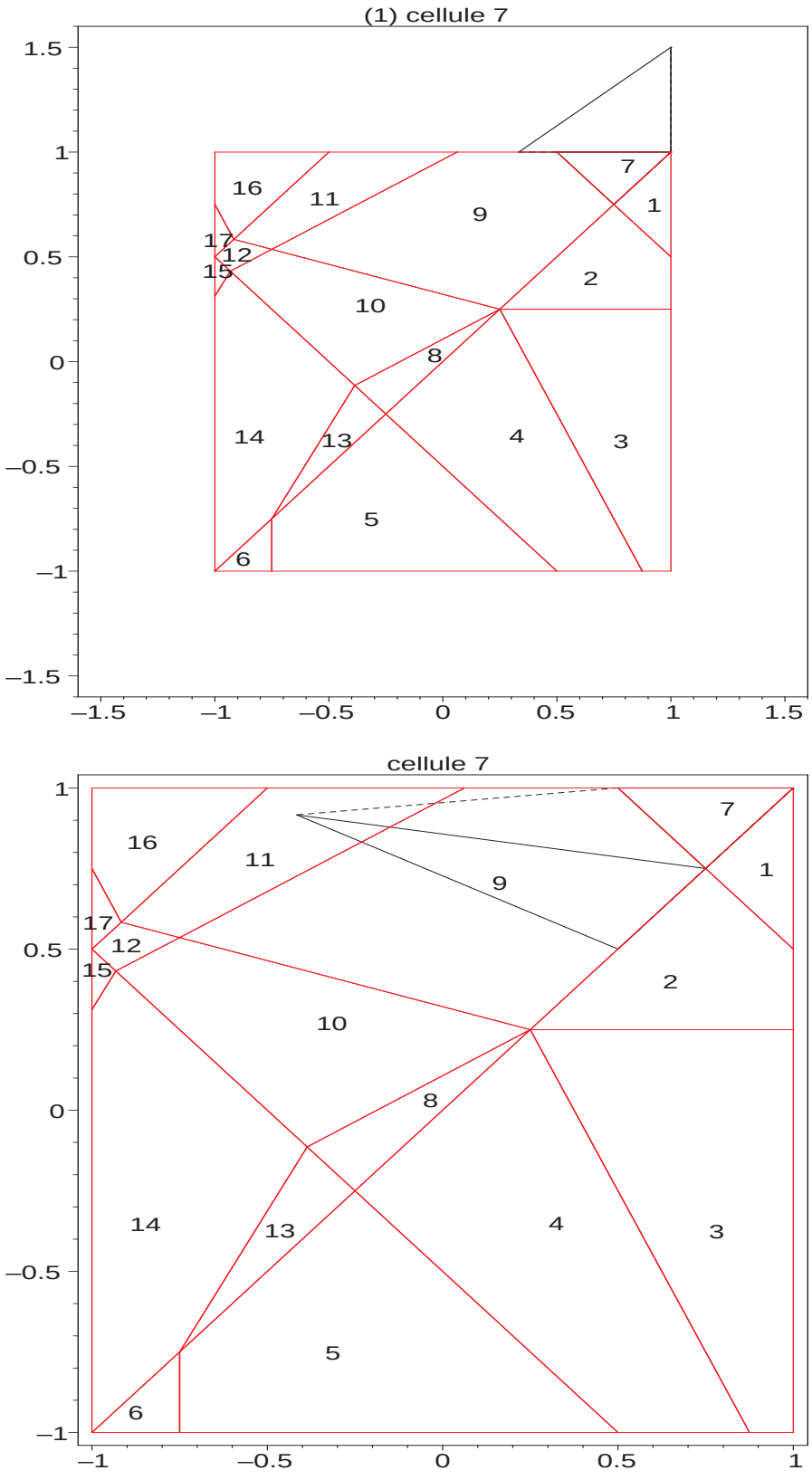


FIG. 6.6 – Action de la première couche affine du PAP (en haut) et du PAP tout entier (en bas) sur la cellule 7 de sa partition terminale.

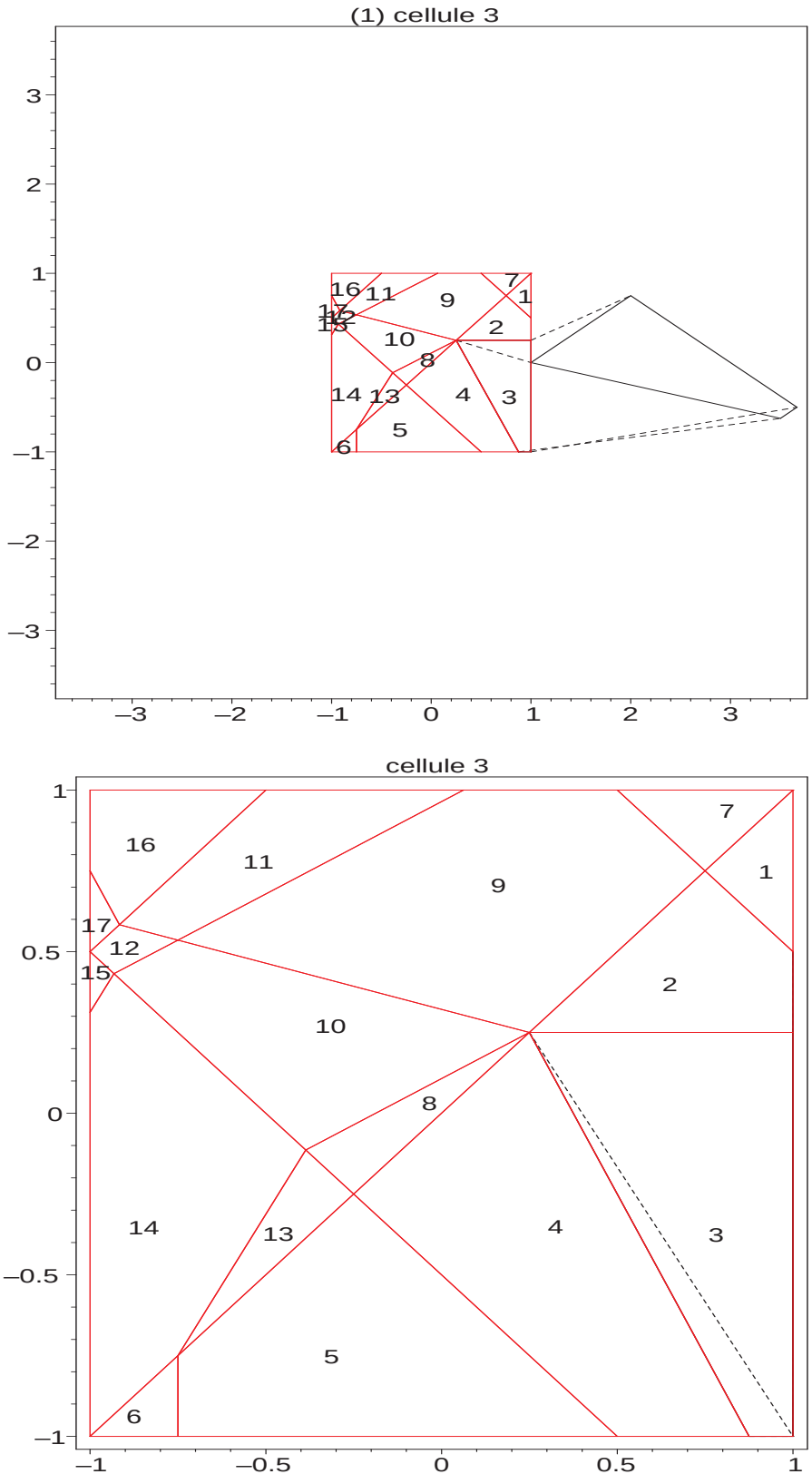


FIG. 6.7 – Action de la première couche affine du PAP (en haut) et du PAP tout entier (en bas) sur la cellule 3 de sa partition terminale.

Les illustrations et les calculs de cette section sont réalisés avec un code MapleV (développé pendant le travail de thèse) qui permet de manipuler très simplement des PAP en faible dimension[1].

Il est constitué des primitives suivantes :

- `hyperCC(`x`,`y`)` qui renvoie un hypercube[2] de rayon $y-x$ centré en zéro.
- `primaryPart(`F`,` $W^{(1)}$`,` $b^{(1)}$`)` qui renvoie la liste des polyhèdres résultants du "découpage" de F (une liste de cellules polyhédrales) par la première couche d'un PAP de poids de première couche $W^{(1)}$ et $b^{(1)}$.
- `secondaryPart(`F`,` $W^{(1)}$`,` $b^{(1)}$`,` $W^{(2)}$`,` $b^{(2)}$`)` qui renvoie la liste des polyhèdres résultants du "découpage" de F par le PAP de poids $W^{(1)}$, $b^{(1)}$, $W^{(2)}$, et $b^{(2)}$.
- `PPForDraw(`F`)` qui affiche la liste de cellules polyhédrales F.

Ces fonctions permettent de générer le découpage en cellules polyhédrales induit par un PAP, et de l'afficher. Pour l'exemple 6.2, elles ont généré les illustrations 6.1 à 6.7.

[1] Le code complet permet de manipuler des PAP en dimension quelconque, mais son utilisation est plus complexe.

[2] Les composants de ce cube sont exprimés dans un format défini pour l'occasion, qui permet un affichage et une manipulation simples.

6.2 Approximation par une chaîne de Markov

6.2.1 Généralités et conjectures

Une fois que l'on a identifié les cellules polyhédrales d'un PAP et que l'on veut étudier son bouclage sur lui-même, il est assez naturel de tenter de le simplifier ainsi :

Définition 24 (Simplification d'un PAP bouclé sur lui-même) *Soit un PAP* $\Psi_W(x) = \Phi(W^{(2)}\,\Phi(W^{(1)}x + b^{(1)}) + b^{(2)})$ *bouclé sur lui-même par l'équation aux différences :*

$$x_{n+1} = \Psi_W(x_n),\ x_0 = \xi_0 \in \mathcal{E}_0 \tag{6.5}$$

Ψ_W *génère une partition terminale* $\{\mathcal{C}^{(i)}\}_{1\leq i\leq I}$ *en* I *cellules polyhédrales de l'hypercube* $\mathfrak{C}$ *(on a* $\mathcal{E}_0 \subset \mathfrak{C}$*). On note pour tout couple* $(\mathcal{C}^{(i)}, \mathcal{C}^{(j)})$ *de cellules (distinctes ou non) de cette partition :*

$$P_{ij} = Prob(x_{n+1} \in \mathcal{C}^{(j)} | x_n \in \mathcal{C}^{(i)}) = \frac{aire(\Psi_W(\mathcal{C}^{(i)}) \cap \mathcal{C}^{(j)})}{aire(\Psi_W(\mathcal{C}^{(i)}) \cap \mathfrak{C})} \tag{6.6}$$

On associe alors à Ψ_W *la chaîne de Markov dont les états sont les* $\mathcal{C}^{(i)}$*, les probabilités de transition les* P_{ij}*, et la distribution initiale :*

$$\mu_0(i) = Prob(\text{état initial} = \mathcal{C}^{(i)}) = \frac{aire(\mathcal{E}_0 \cap \mathcal{C}^{(i)})}{aire(\mathcal{E}_0)} \tag{6.7}$$

On pourra noter $\mathcal{M}(\Psi_W)$ *la matrice des* P_{ij} *et* $\mu_0(\Psi_W)$ *le vecteur des* $\mu_0(i)$*.*

On remarque que la matrice $P = [P_{ij}]_{ij}$ est bien une matrice stochastique : tous ses éléments sont positifs et la somme de chacune de ses lignes est égale à 1 (puisque les $(\mathcal{C}^{(j)})_j$ forment une partition de l'hypercube) :

$$\sum_j P_{ij} = \frac{\sum_j aire(\Psi_W(\mathcal{C}^{(i)}) \cap \mathcal{C}^{(j)})}{aire(\Psi_W(\mathcal{C}^{(i)}) \cap \mathfrak{C})} = 1$$

L'étude du comportement asymptotique d'une telle chaîne de Markov se ramène à l'étude de la matrice $\mathcal{M}(\Psi_W)$ (pour un point de vue compatible avec la théorie de la dynamique symbolique, cf chapitre sept de [Kitchens, 1998]).

Il est certainement possible de démontrer un résultat similaire à la conjecture suivante :

Conjecture 1 (Relation entre un PAP bouclé et la chaîne de Markov associée) *Si pour tout $x_0 = \xi_0 \in \mathcal{E}_0$ le système $x_{n+1} = \Psi_W(x_n)$ (6.5) converge vers une limite x_∞ alors le support de la distribution limite de la chaîne de Markov de matrice de transition $\mathcal{M}(\Psi_W)$ sur la partition terminale $(\mathcal{C}^{(i)})_{1\leq i\leq I}$ de Ψ_W et de distribution initiale $\mu_0(\Psi_W)$ (définies par (6.6) et (6.7)) est un ensemble $(\mathcal{C}^{(j)})_{j\in J}$ de cellules dont l'intersection contient x_∞.*

Les principales difficultés rencontrées pour démontrer ce type de propriétés se situent aux frontières entre cellules de la partition terminale de Ψ_W. On peut en revanche facilement obtenir la propriété suivante :

Proposition 4 (Relation entre la convergence d'un PAP bouclé et de la chaine de Markov associée dans un cas simple) *Si le système $x_{n+1} = \Psi_W(x_n)$, $x_0 = \xi_0 \in \mathcal{E}_0$ converge vers une limite x_∞ qui est à l'intérieur de la cellule $\mathcal{C}^{(j)}$ de la partition terminale de Ψ_W et que le support de la distribution limite de la chaîne de Markov associée est réduit à une seule cellule $\mathcal{C}^{(k)}$, alors $\mathcal{C}^{(j)} = \mathcal{C}^{(k)}$.*

En effet, si le support de la distribution limite de $\mathcal{M}(\Psi_W)$ est réduit à une seule cellule $\mathcal{C}^{(k)}$, aucune trajectoire de $\mathcal{M}(\Psi_W)$ issue de cette cellule ne la quitte (cf par exemple la proposition 6 p139 de [Pallu de la Barrière, 1965]), il est donc impossible que toutes les trajectoires de $x_{n+1} = \Psi_W(x_n)$ issues de $\mathcal{C}^{(k)}$ arrivent dans une cellule $\mathcal{C}^{(j)} \neq \mathcal{C}^{(k)}$. Nécessairement $\mathcal{C}^{(j)} = \mathcal{C}^{(k)}$. □

Mais l'objet de ce mémoire n'est pas de s'attarder sur les propriétés des chaînes de Markov associées aux PAP bouclés sur eux-mêmes. Il s'agit juste de mettre en avant dans ce chapitre qu'il est possible d'obtenir des résultats de ce type en faisant appel aux résultats théoriques sur la convergence des chaînes de Markov.

6.2.2 Reprise de l'exemple du système dynamique engendré par un PAP bouclé sur lui-même

Il est possible (mais assez technique, surtout en dimension quelconque) de calculer les P_{ij} explicitement en fonction des valeurs des matrices de poids d'un PAP.

La solution qui a été retenue est de les approcher par simulation : n^d points (notés $(x_i)_{1\leq i\leq n^d}$) sont équirépartis dans un espace de dimension d (suivant une grille), on approche alors P_{ij} par $\hat{P}_{ij}$:

$$\hat{P}_{ij} = \frac{\text{Card}\{x_k \in \Psi_W(\mathcal{C}^{(i)}) \cap \mathcal{C}^{(j)}\}}{\text{Card}\{x_k \in \Psi_W(\mathcal{C}^{(i)}) \cap \mathfrak{C}\}}$$

$\hat{P}_{ij}$ se rapproche évidemment de P_{ij} lorsque n augmente. L'avantage de cette méthode est qu'il suffit d'implémenter une fonction qui permet de savoir si un point de l'espace appartient à une cellule polyhédrale.

Pour l'exemple de cette partie et avec $n = 20$ (c'est-à-dire $20^2 = 400$ points), on obtient l'approximation suivante pour la matrice de transition :

$$\mathcal{M}(\Psi_W) = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{5}{11} & \frac{6}{11} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{27} & \frac{26}{27} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{2}{3} & \frac{1}{3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{8}{9} & 0 & \frac{1}{9} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & \frac{1}{62} & \frac{4}{31} & \frac{11}{62} & \frac{11}{62} & 0 & 0 & \frac{3}{62} & \frac{7}{62} & \frac{3}{31} & \frac{3}{62} & 0 & \frac{1}{62} & \frac{5}{31} & 0 & \frac{1}{62} & 0 \\
0 & 0 & \frac{1}{10} & \frac{4}{15} & \frac{17}{30} & \frac{1}{15} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{16}{21} & \frac{1}{7} & 0 & \frac{2}{21} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{4}{53} & \frac{15}{53} & \frac{31}{53} & \frac{3}{53} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0
\end{bmatrix} \tag{6.8}$$

La distribution limite associée est donnée par le vecteur propre associé à la valeur propre 1 :

$$\mu_\infty = \begin{bmatrix} 0 & \frac{11}{173} & \frac{162}{173} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Cela implique que les cellules qui contiennent les limites du système dynamique formé par le PAP bouclé sur lui-même contient les cellules 2 et 3. Ce résultat est parfaitement cohérent avec la figure 6.3 issue de simulation numériques sur le PAP bouclé.

6.3 Conclusion

L'étude rapide de la dynamique d'un PAP bouclé sur lui-même permet de faire la transition entre les propriétés des PAP relatives à la description "statique" de son action (essentiellement autour de la partition de l'espace qu'il génère) exposées précédemment et les propriétés dynamiques des PAP relatives à leur utilisation dans le cadre du contrôle qui sont exposées dans les chapitres suivants.

D'une part, la description explicite de l'action d'un PAP en termes de fonctions affines associées à des cellules polyhédrales partitionnant l'espace de départ d'un PAP permet d'accéder à une compréhension claire de son action. En effet, les sections précédentes montrent que la connaissance des valeurs des poids d'un PAP permet de déterminer clairement et explicitement la nature de son action par : ses partitions initiales et terminales et les valeurs de ces matrices de saturation sur chaque cellule engendrée : les PAP sont loin d'être des "*boîtes noires*" et fournissent des pistes permettant d'expliciter les perceptrons plus classiques.

D'autre part, le recours à un exemple de PAP bouclé sur lui-même rend compte de la façon dont les PAP peuvent être utilisés dans le cadre du contrôle. En effet, l'étude du système dynamique constitué par un PAP bouclé sur lui-même peut se séparer en deux parties :

- l'étude de leur comportement au sein d'une cellule de la partition terminale qu'ils engendrent : il s'agit d'une étude *locale*. Dans ce cadre un PAP se ramène à une fonction affine,
- l'étude (plus macroscopique) des transitions entre les cellules de leur partition terminale.

Il est vraisemblable que la conjoncture 1 puisse être démontrée, il serait alors possible d'étudier simplement un système dynamique par la méthodologie suivante :

1. approximation du système par un PAP,
2. étude de la chaîne de Markov engendrée par le PAP,
3. détermination de l'ensemble des cellules de la partition terminale du PAP qui est le support à la distribution limite de la chaîne de Markov,
4. conclusion : les limites éventuelles du système original sont inclues dans cet ensemble.

Cette méthodologie permettrait d'obtenir rapidement des résultats "*macroscopique*" sur la convergence de systèmes dynamiques. Cela serait surtout

utile dans le domaine du contrôle pour une vérification a posteriori :

1. le système à contrôler est approximé par un PAP Ψ_1,
2. le contrôle est émulé par un PAP Ψ_2,
3. on construit le PAP du système asservi : $\Psi = \Psi_1 \circ \Psi_2$,
4. on applique la méthodologie précédente,
5. on sait dans quelles zones de l'espace d'état le système contrôlé peut être amené (i.e. support à la distribution limite de la chaîne de Markov).

Les sections suivantes vont exposer en quoi il est possible d'obtenir des résultats plus directement (sans passer par la construction de la chaîne de Markov sous jacente).

Des résultats de convergence de systèmes émulés par des PAP sont importants dans la mesure où les PAP représentent les fonctions continues affines par morceaux : il y en a nécessairement qui convergent et d'autres pas. Les résultats obtenus sur les PAP sont en outre directement applicables aux fonctions continues affines par morceaux, fréquemment utilisées dans la cadre du contrôle flou et de la robotique.

Chapitre 7

Apprentissage dans une boucle fermée

7.1 Contrôle optimal par RNF

On considère le système dynamique :

$$x_{n+1} = F(x_n, u_n), x_0 = \xi_0, x \in I\!R^d, u \in I\!R^a, x_0 \in \mathcal{X}_0 = Supp(\xi_0) \tag{7.1}$$

contrôlé par la valeur de $u_n = \Psi_W(x_n)$ où Ψ_W est un réseau de neurones à une couche cachée :

$$\Psi_W(x) = \Phi_2\left(W^{(2)} \cdot \Phi_1\left(W^{(1)} \cdot x + b^{(1)}\right) + b^{(2)}\right) \tag{7.2}$$

où Φ_1 et Φ_2 sont des fonctions vectorielles d'activation presque partout différentiables (p/r à une mesure de Lebesgue sur leurs espaces de départs respectifs : $I\!R^c$ et $I\!R^a$).

On note :

$$\begin{aligned} y^{(1)} &= W^{(1)} \cdot x + b^{(1)} \\ y^{(2)} &= W^{(2)} \cdot \Phi_1\left(y^{(1)}\right) + b^{(2)} \end{aligned}$$

On veut que les trajectoires issues de (7.1) minimisent le critère $I\!I$ défini à partir de plusieurs fonctions de "coût" :

Coût instantané d'un contrôle noté $I(x)$, qui est une jauge ; dans le cadre du contrôle à coût quadratique, on prendra :

$$I(x) = x'Qx + u'Ru,\ u = \Psi_W(x) \tag{7.3}$$

avec Q et R deux matrices symétriques définies positives.

Coût d'une trajectoire issue d'un point x_0, défini à l'aide d'une famille de pondérations $(\rho_k)_k$, et noté :

$$\mathcal{I}_n(x_0) = \sum_{k=0}^{n} \rho_k \, I(x_k) \tag{7.4}$$

Espérance du coût de contrôle obtenu à l'aide d'une mesure $d\xi_0$ sur l'espace des trajectoires initiales qui correspond à la distribution de probabilité d'apparition des points initiaux :

$$\mathit{I\!I}_n = \int_{x_0 \in \mathcal{X}_0} \mathcal{I}_n(x_0) d\xi_0(x_0) \tag{7.5}$$

Coût du contrôle qui est la limite de l'espérance du coût de contrôle lorsque l'on suit les trajectoire à l'infini :

$$\mathit{I\!I} = \lim_{n \to \infty} \mathit{I\!I}_n \tag{7.6}$$

on peut aussi écrire directement $\mathit{I\!I}$ comme :

$$\mathit{I\!I} = \lim_{n \to \infty} \int_{x_0 \in \mathcal{X}_0} \left(\sum_{k=0}^{n} \rho_k \left(x_k' Q x_k + u_k' R u_k \right) \right) d\xi_0(x_0)$$

7.2 Apprentissage en boucle fermée

7.2.1 Ordre de l'apprentissage

Cet ensemble de contraintes défini un système dynamique paramètré par les poids W de Ψ_W : $W^{(1)}$, $b^{(1)}$, $W^{(2)}$, et $b^{(2)}$. On veut attribuer à W des valeurs telles que le critère (7.6) soit minimal. Pour cela, on peut choisir des valeurs arbitraires pour les w pris dans l'ensemble des coordonnées de $W^{(1)}$, $b^{(1)}$, $W^{(2)}$, ou $b^{(2)}$ puis les faire évoluer suivant la dynamique (k est une variable muette qui correspond aux itération de l'algorithme d'*apprentissage*) :

$$w_{k+1} = w_k - \gamma \cdot \partial_w \mathit{I\!I} \tag{7.7}$$

En pratique, il est impossible de calculer $\mathit{I\!I}$, on utilise donc un de ses estimateurs :

$$\widehat{\mathit{I\!I}}_N = \frac{1}{\mathrm{Card}(X_0)} \sum_{x_0 \in X_0} \mathcal{I}_N(x_0) \tag{7.8}$$

On remplace en effet $\int_{\mathcal{X}_0} f(x) d\xi_0(x)$ par $\sum_{x \in X_0} f(x)$ où les points de X_0 sont choisis dans $\mathcal{X}_0$ selon ξ_0. On ne dispose en effet généralement pas de la loi ξ_0 mais il est toujours possible d'observer ses réalisations.

Le passage du *risque théorique* $\int_{\mathcal{X}_0} f(x)d\xi_0(x)$ au *risque empirique* $\sum_{x\in X_0} f(x)$ n'est pas sans une perte quantifiée notamment par Vapnik et Chervonenkis (section 7.4.2).

On prend la définition suivante :

Définition 25 (ordre de l'apprentissage)
*L'***ordre** *de l'apprentissage est l'entier N de l'expression (7.8).*

On sait que le résultat de cet algorithme va aboutir à des valeurs pour les w qui minimisent localement $I\!I$. La dynamique (7.7) peut être améliorée par l'intervention d'un moment ([Rumelhart *et al.* , 1986] ou [Pearlmutter, 1992]) ce qui donnerait :

$$w_{k+1} = w_k - \gamma_k \cdot \partial_w I\!I|_{W_k} - \alpha\gamma_{k-1} \cdot \partial_w I\!I|_{W_{k-1}} \tag{7.9}$$

ou bien par l'utilisation de la hessienne de $I\!I$:

$$w_{k+1} = w_k - \gamma \cdot (\mathrm{Hess}_w(I\!I))^{-1}\, \partial_w I\!I \tag{7.10}$$

ou par toute autre méthode locale ou globale d'accélération de la convergence ([Jondarr, 1996] ou [Dreisietl, nov 1992]).

Nous nous limiterons à (7.7) pour des raisons de simplicité de calcul, mais ceux-ci pourraient être menés sur les formulations avec moment ou avec hessienne.

D'autre part, l'aspect stochastique de l'algorithme [Jabri & Flower, 1992] dû à l'utilisation d'un estimateur de $I\!I$ plutôt que sa véritable valeur (on fait intervenir "à la Monte-Carlo" un tirage aléatoire selon ξ_0 de points de départ x_0) laisse penser que les trajectoires de (7.7) sortiront des minima locaux de $I\!I$ vue comme une fonction de w, mais ce travail ne s'attardera pas sur cet aspect intéressant qu'il faudrait aussi développer.

Initialisation des poids du contrôleur. Comme on l'a exposé plus haut, une des grosses difficultés du contrôle par RNF réside dans l'initialisation des poids du contrôleur. En effet la méthode d'initialisation des poids d'un RNF habituellement utilisée est aléatoire ; or l'asservissement d'un système par un contrôleur "aléatoire" provoque dans la plupart des cas (en fait : dès que la dimension de l'espace d'état est d'au moins 10) une "catastrophe". Cela peut être une casse ou un calage dans le cas des moteurs, ou bien l'explosion ou l'arrêt total dans le cadre de réacteurs chimiques : dans tous les cas (et même si on se trouve dans des situations où la "casse" est "gratuite", ce qui peut être obtenu en utilisant des simulateurs) il n'est pas possible d'observer le comportement du système sur plus de quelques itérations, ce qui implique que l'ordre de l'apprentissage soit plus petit de ce nombre d'itérations "observées".

Dans le cas des PAP, les résultats précédents nous permettent d'initialiser un PAP à partir d'une partition de l'espace d'état et de fonctions affines sur chaque cellule de cette partition. **On ne va donc pas initialiser les** w **aléatoirement** mais on va utiliser un jeu de valeurs qui minimisent le problème linéaire correspondant, ou un ensemble de problèmes linéaires locaux correspondants (cf section 5.4) . Le système (7.7) est donc positionné assez proche d'un minimum important.

Cette étape est capitale et ouvre aux PAP des domaines de contrôle jusqu'à lors refusés aux autres réseaux de neurones (dimension de l'espace d'état assez grande).

7.2.2 Contribution des paramètres du RNF au coût de contrôle

Pour utiliser l'algorithme d'évolution du système (7.7), il est nécessaire de pouvoir calculer $\partial_w \widehat{I\!I}_N$. Ceci se fait par l'utilisation des relations suivantes :

$$\partial_w \widehat{I\!I}_N = \frac{1}{\mathrm{Card}(X_0)} \sum_{x_0 \in X_0} \partial_w \mathcal{I}_N(x_0) \tag{7.11}$$

$$\partial_w \mathcal{I}_N(x_0) = \sum_{k=0}^{N} \rho_k \partial_w I(x_k) \tag{7.12}$$

En notant $\omega_n = \partial_w x_n$ (avec $n > 0$) pour simplifier les écritures et en utilisant :

$$\partial_w u_n = \partial_w \Psi_W(x_n) = \partial_w \Psi_W|_{x_n} \cdot \partial_w x_n$$

on obtient :

$$\partial_w I(x_n) = 2 \left(x_n Q + \Psi_W(x_n)' R \, \partial_w \Psi_W|_{x_n} \right) \omega_n \tag{7.13}$$

et :

$$\begin{aligned} \omega_{n+1} &= \partial_w F(x_n, \Psi_W(x_n)) \\ &= \left(\partial_x F|_{x_n} + \partial_u F|_{x_n} \cdot \partial_w \Psi_W|_{x_n} \right) \omega_n \qquad (7.14) \\ \omega_1 &= \partial_u F|_{x_0} \cdot \partial_w \Psi_W|_{x_0} \qquad (7.15) \end{aligned}$$

Grâce à ces équations, on peut calculer ω_n pour tout n, puis en déduire $\partial_w \widehat{I\!I}_N$. On voit qu'il est nécessaire de connaître $\partial_x F$ et $\partial_u F$ d'une part (nous y reviendrons en partie dans la section 7.4.1), et $\partial_w \Psi_W$ d'autre part.

7.2.3 Etape élémentaire de l'apprentissage

Le calcul de $\partial_w \Psi_W$ nécessite d'être effectué, même si il correspond à une étape élémentaire.

Pour simplifier les écritures, on notera pour un vecteur y de $\mathbb{R}^d$, $\mathcal{D}\mathrm{iag}_l(y)$ la matrice :

$$\mathcal{D}\mathrm{iag}_l(y) = \begin{bmatrix} \phi_l'(y_1) & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \phi_l'(y_d) \end{bmatrix}$$

- lorsque $w = b^{(2)}$:
$$\partial_{b^{(2)}} \Psi_W(x) = \mathcal{D}\mathrm{iag}_2(y^{(2)})$$
- lorsque $w = W^{(2)}_{.l}$, la $l$ème ligne de $W^{(2)}$:
$$\partial_{W^{(2)}_{.l}} \Psi_W(x) = \phi_1(y^{(1)}_l)\ \mathcal{D}\mathrm{iag}_2(y^{(2)})$$
- lorsque $w = b^{(1)}$:
$$\partial_{b^{(1)}} \Psi_W(x) = \mathcal{D}\mathrm{iag}_2(y^{(2)})\, W^{(2)}\ \mathcal{D}\mathrm{iag}_1(y^{(1)})$$
- lorsque $w = W^{(1)}_{.l}$, la $l$ème ligne de $W^{(1)}$ (à vérifier...) :
$$\partial_{W^{(1)}_{.l}} \Psi_W(x) = x_l\ \mathcal{D}\mathrm{iag}_2(y^{(2)})\, W^{(2)}\ \mathcal{D}\mathrm{iag}_1(y^{(1)})$$

Il faut noter que l'on a choisi d'exprimer les différentielles par rapport aux vecteurs lignes des matrices $W^{(1)}$ et $W^{(2)}$ plutôt que par rapport à leurs vecteurs colonnes car une ligne est associé à un neurone (cf section 5.3).

Ces quatre expressions permettent de calculer tous les éléments nécessaires à l'implémentation de la récurrence (7.7) de mise à jour des paramètres du réseaux.

7.3 Conception de l'algorithme d'apprentissage

7.3.1 Squelette de l'apprentissage

Les calculs précédents permettent l'implémentation d'un algorithme de mise à jours itérative des paramètres d'un réseau de neurones destiné au contrôle d'un système dynamique donné. Dans ce cadre l'implémentation à été faite d'abord sous l'environnement de programmation mathématique MapleV[1] puis dans le langage de programmation Java[2].

[1] release 2, ©maplesoft 1992.
[2] JDK 1.1.5

Algorithm 1 Apprentissage : principe de l'algorithme.

`entrées`

un modèle du système à contrôler
un modèle des dérivées du système à contrôler
le nombre de trajectoires à simuler ($\mathrm{Card}(X_0)$)
les frontières de X_0 (qui est limité à un hypercube)
l'*ordre* de l'apprentissage (la durée de chaque trajectoire N de $\widehat{\mathit{I\!I}}_N$)
les pondérations $(\rho_i)_i$ et les matrices Q et R qui définissent $\mathcal{I}$
un jeu de paramètres $\{W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}\}$ initial pour le contrôleur
le nombre de fois qu'il faut itérer l'apprentissage
une suite $(\gamma_k)_k$ de pas pour l'apprentissage

`algorithme`

faire autant de fois que l'on veut itérer l'apprentissage (variable k)
  mettre à zéro des variables $\delta(W_l^{(1)})_l$, $\delta(b^{(1)})$, $\delta(W_l^{(2)})_l$, $\delta(b^{(2)})$
  faire pour chaque x_0 choisit arbitrairement dans X_0
    {initialisation des $\omega(w)$: équation (7.15)}
    $\omega(w) \leftarrow \partial_u F|_{x_0} \cdot \partial_w \Psi_W|_{x_0}$
    {mise à jour des $\delta(w)$: équations (7.13)}
    $\delta(w) \overset{+}{\leftarrow} 2\rho_0 \left(x_0' Q x_0 + \Psi_W(x_0)' R\, \partial_w \Psi_W|_{x_n}\right) \omega(w)$
    faire pour chaque itération n de 1 à l'ordre de l'apprentissage
      {calcul de l'état suivant}
      $x_n \leftarrow F(x_{n-1}, \Psi_W(x_{n-1}))$
      {mise à jour des $\omega(w)$: équation (7.14)}
      $\omega(w) \leftarrow \left(\partial_x F|_{x_n} + \partial_u F|_{x_n} \cdot \partial_w \Psi_W|_{x_n}\right) \omega(w)$
      {mise à jour des $\delta(w)$: équation (7.13)}
      $\delta(w) \overset{+}{\leftarrow} 2\rho_n \left(x_n' Q x_n + \Psi_W(x_n)' R\, \partial_w \Psi_W|_{x_n}\right) \omega(w)$
    fin faire
  fin faire
  {mise à jour des nouvelles valeurs des w}
  $w \overset{+}{\leftarrow} -\gamma_k \frac{1}{\mathrm{Card}(X_0)} \delta(w)$
fin faire

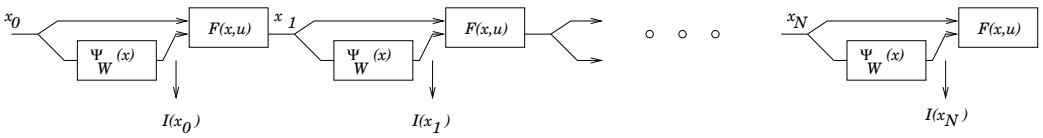


FIG. 7.1 – N itérations du système contrôlé.

L'algorithme 1 présentant l'algorithme utilisé est évidemment simplifié : par exemple, pour des raisons de temps de calcul, certaines grandeurs intermédiaires sont stockées temporairement afin de ne pas être calculées plusieurs fois (comme les $\partial_w \Psi_W|_{x_n}$). D'autre part, il existe une version de l'algorithme qui mémorise la valeur du critère au fur et à mesure des itérations et des trajectoires. Celle-ci, si elle est plus gourmande en mémoire, permet de "surveiller" l'action de l'apprentissage.

Pour des raisons de simplicité, seul un algorithme dont le pas $(\gamma_k)_k$ est indépendant de la valeur du critère à minimiser à été implémenté. Usuellement, on a pris pour une constante g :

$$\gamma_k = \frac{g}{1+k}$$

7.3.2 Limites naturelles d'un apprentissage en boucle fermée

Cet algorithme est naturellement limité par deux aspects :

Minimisation du risque empirique. On utilise comme critère :

$$\hat{I\!I}_N = \frac{1}{\mathrm{Card}(X_0)} \sum_{x_0 \in X_0} \mathcal{I}_N(x_0)$$

plutôt que

$$I\!I_N = \int_{x_0 \in \mathcal{X}_0} \mathcal{I}_N(x_0) d\xi_0(x_0)$$

Minimisation à horizon fini. On utilise $\widehat{I\!I}_N$ plutôt que $\lim_{n\to\infty} \widehat{I\!I}_n$.

La première limitation est discutée dans la section 7.4.2, la seconde correspond au tronquage de la série des $(\rho_n I(x_n))_n$ en N, on peut l'illustrer par la figure 7.1.

Pour avoir une idée de ce qui est perdu (en terme de quantité rétro propagée) lors de l'abandon de :

$$r_N(x_0) = \sum_{k=N+1}^{\infty} \rho_k \, I(x_k) \tag{7.16}$$

(x_0 signifie que les $(x_k)_k$ sont les trajectoires de (7.1) issues de x_0) il suffit d'étudier le comportement de la norme de $\partial_w r_n(x_0)$. Or, la suite d'inégalités :

$$\begin{aligned} ||\partial_w r_n(x_0)|| &\leq \sum_{k=n+1}^{\infty} \rho_k \, ||\partial_w I(x_k)|| \\ &\leq \sum_{k=n+1}^{\infty} 2\rho_k \, \left|\left|\left(x_k Q + \Psi_W(x_k)' R \, \partial_w \Psi_W|_{x_k}\right)\right|\right| \cdot ||\omega_n|| \\ &\leq \sum_{k=n+1}^{\infty} 2\rho_k \, \left|\left|\left(x_k Q + \Psi_W(x_k)' R \, \partial_w \Psi_W|_{x_k}\right)\right|\right| \cdot \\ &\qquad \underbrace{\left(\prod_{m=1}^{k} \left|\left| \partial_x F|_{x_m} + \partial_u F|_{x_m} \cdot \partial_w \Psi_W|_{x_m} \right|\right|\right)}_{\Pi_k(w, X_0)} ||\omega_1|| \end{aligned} \tag{7.17}$$

met en avant le rôle du produit $\Pi_k(w, X_0)$ dans l'importance de la part du critère non prise en compte lorsqu'on se limite à $\mathcal{I}_k(x_0)$. Ce produit joue aussi un grand rôle dans la convergence de la dynamique des poids w puisque :

Propriété 3 (Convergence de l'apprentissage.)
Posons :

$$\delta F_W(x_m) = \partial_x F|_{x_m} + \partial_u F|_{x_m} \cdot \partial_w \Psi_W|_{x_m}$$

qui est la différentielle par rapport à x, prise en x_m, de $F(x, \Psi_W(x))$ vue comme une fonction de x.

Si pour tous les points initiaux x_0 , chaque trajectoire suivant (7.7) passe au plus un nombre de fois borné par un entier J par un état où

$$||\delta F_W(x_m)|| \geq 1 \tag{7.18}$$

et si de plus les trajectoires restent dans un compact.

Alors l'accroissement des poids du RNF pendant l'apprentissage est borné.

La condition "les trajectoires de l'apprentissage ne passent qu'un nombre de fois borné par J par un état $\bar{x}$ pour lequel $||\delta F_W(\bar{x})|| \geq 1$" est existentielle : elle ne permet pas de vérifier quoi que ce soit empiriquement.

En effet, il se peut que δF_W ait de large zones sur lesquelles sa norme soit plus grande que un ; mais cela importe peu si les trajectoires de l'apprentissage n'y passent pas (ou presque pas). Il se peut au contraire qu'il n'y ait qu'une toute petite zone sur laquelle sa norme soit très grande, mais que les trajectoires de l'apprentissage y passent très fréquemment : cela entraînera une divergence des poids.

Cette proposition a donc surtout une valeur illustrative des phénomènes qui entrent en jeu dans le cadre de la dynamique des poids pendant l'apprentissage.

L'information qu'apporte cette proposition provient de la forme de δF_W : on voit qu'il y a un phénomène de "compensation" possible entre $\partial_u F|_{x_m}$ et $\partial_w \Psi_W|_{x_m}$. Plus le système dynamique est "naturellement convergent" (i.e. plus $\left|\left|\partial_u F|_{x_m}\right|\right|$ est petite) et plus la contrainte sur les poids du RNF peut être relaxée (i.e. plus $\left|\left|\partial_w \Psi_W|_{x_m}\right|\right|$ peut être grande).

La démonstration de cette propriété est élémentaire ; écrivons l'accroissement d'un des poids w :

$$\partial_w I\!I = \int_{x_0 \in \mathcal{X}_0} \left(\sum_{n>0} \rho_n 2(x_n Q + \Psi_W(x_n)' \, R \, \partial_w \Psi_W|_{x_n}) \cdot \prod_{m \leq n} \delta F_W(x_m) \cdot \partial_u F|_{x_0} \cdot \partial_w \Psi_W|_{x_0} \right) d\xi_0(x_0)$$

Les hypothèses de la proposition nous permettent de borner la norme de $\partial_w I\!I$ ainsi (σ est une correspondance entre les J_m premier entiers ($J_m \leq J$) et les indices de x pour lesquels $||\delta F_W||$ est plus grande que 1 ; $\mathbf{B}, \mathcal{M}$ et $q < 1$ sont des majorants) :

$$\begin{aligned} ||\partial_w I\!I||^2 &\leq \sum_{n>0} \int_{x_0 \in \mathcal{X}_0} \mathbf{B}^2 \cdot \prod_{j=1}^{J_m} \left|\left|\delta F_W(x_{\sigma(j)})\right|\right|^2 \prod_{1 \leq m \geq n}^{m \neq \sigma(\{1,\ldots,J_m\})} \left|\left|\delta F_W(x_{\sigma(j)})\right|\right|^2 d\xi_0(x_0) \\ &\leq \sum_{n>0} \mathbf{B}^2 \mathcal{M}^{2J} q^{2(n-J)} \int_{x_0 \in \mathcal{X}_0} d\xi_0(x_0) \\ &\leq B^2 \frac{\mathcal{M}^{2J}}{q^{2J}} \frac{1}{1-q^2} \end{aligned}$$

L'existance d'un majorant $\mathbf{B}$ à $\left|\left|\rho_n 2(x_n Q + \Psi_W(x_n)' \, R \, \partial_w \Psi_W|_{x_n})\right|\right|$ provient du fait que les x restent dans un compact. □

Il est vraisemblable qu'une étude plus poussée dans cette direction permette d'établir des résultats de convergence conjointe du système asservi et de l'apprentissage.

Conjecture 2 (Convergence conjointe du système asservi et de son apprentissage) *Si pour tous les points de l'ensemble initial (sauf un sous en-*

semble de ξ_0-mesure nulle) le système asservi correspondant à chaque jeu de poids utilisé pendant l'apprentissage converge, alors l'apprentissage converge.

7.3.3 Critère d'arrêt

Pour l'instant nous ne nous sommes intéressés qu'à des situations où le nombre de passes de l'algorithme (la durée de la trajectoire des paramètres w du réseau de neurones) ainsi que l'ordre de l'apprentissage sont décidés arbitrairement.

7.3.3.1 Ordre de l'apprentissage

La "modification" d'un des poids w à chaque itération de l'apprentissage est $\partial_w I\!I$; dans la réalité, il n'est pas possible d'attendre un temps infini avant de choisir de modifier un poids après une observation de durée infini : il est donc nécessaire d'arrêter l'apprentissage après une durée d'observation finie.

Cette durée est appelée *ordre de l'apprentissage* ; habituellement elle est fixée arbitrairement, et est identique pour toutes les trajectoires observées. On définit donc $I\!I_N$ qui est la version tronquée après N observations de l'évolution de la trajectoire.

Dans le cadre de ce nouvel algorithme d'apprentissage, l'ordre de l'apprentissage est différent pour chaque trajectoire. Il correspond à un instant pour lequel, pour la trajectoire observée, il est probable que $||\partial_w I\!I||^2 - ||\partial_w I\!I_N||^2$ soit petit.

Pour cela on écrit :

$$\partial_w I\!I = \partial_w I\!I_N \quad + \quad \int_{x_0 \in \mathcal{X}_0} \left(\sum_{n>N} \rho_n 2(x_n Q + \Psi_W(x_n)' \, R \, \partial_w \Psi_W|_{x_n}) \cdot \right.$$
$$\left. (\prod_{m \leq N} \delta F_W(x_m))(\prod_{N<m\leq n} \delta F_W(x_m)) \cdot \partial_u F|_{x_0} \cdot \partial_w \Psi_W|_{x_0} \right) d\xi_0(x_0)$$

ce qui permet d'obtenir :

$$|\partial_w I\!I||^2 \leq ||\partial_w I\!I_N||^2 + \int_{x_0 \in \mathcal{X}_0} \left(\prod_{m \leq N} ||\delta F_W(x_m)||^2 \right) \sum_{n>N} \cdots d\xi_0(x_0) \tag{7.19}$$

C'est de cette inégalité qu'on va déterminer l'ordre $N(x_0)$ associé à la trajectoire issue de x_0, on va le prendre de telle sorte que :

$$\prod_{m \leq N(x_0)} ||\delta F_W(x_m)|| < \varepsilon \tag{7.20}$$

Pour un réel positif fixé ε, c'est à ce moment que l'on va décider de stopper le calcul de la trajectoire.

Cette méthode permet de contrôler l'ordre de l'apprentissage à l'aide du paramètre ε plutôt que de se fixer un ordre maximum identique pour toutes les trajectoires : on va apprendre "plus longtemps" sur les trajectoires qui ont "longtemps" un grand coût et moins longtemps sur les trajectoires qui optimisent vite le critère à minimiser.

7.3.3.2 Nombre de passes de l'algorithme

Le nombre de passes de l'algorithme est le nombre de points de départ (pris dans $\mathcal{X}_0$) pour l'itération de la dynamique (7.7) :

$$w_{k+1} = w_k - \gamma_k \cdot \partial_w I\!I$$

Il a été fixé arbitrairement.

La problématique de "sur-apprentissage" n'est pas de même nature que celle rencontrée habituellement lors de l'apprentissage de RNF. En effet, on ne dispose pas d'une base de données de cardinal fini ; pour chaque passe de l'apprentissage on tire un nouveau point dans $\mathcal{X}_0$ en utilisant la même loi que celle qui sera utilisé lors de l'utilisation réelle du contrôleur.

La seule limite à la capacité d'apprendre totalement par cette méthode est issue du recours au risque empirique plutôt qu'au risque théorique lors de la minimisation (section 7.4.2).

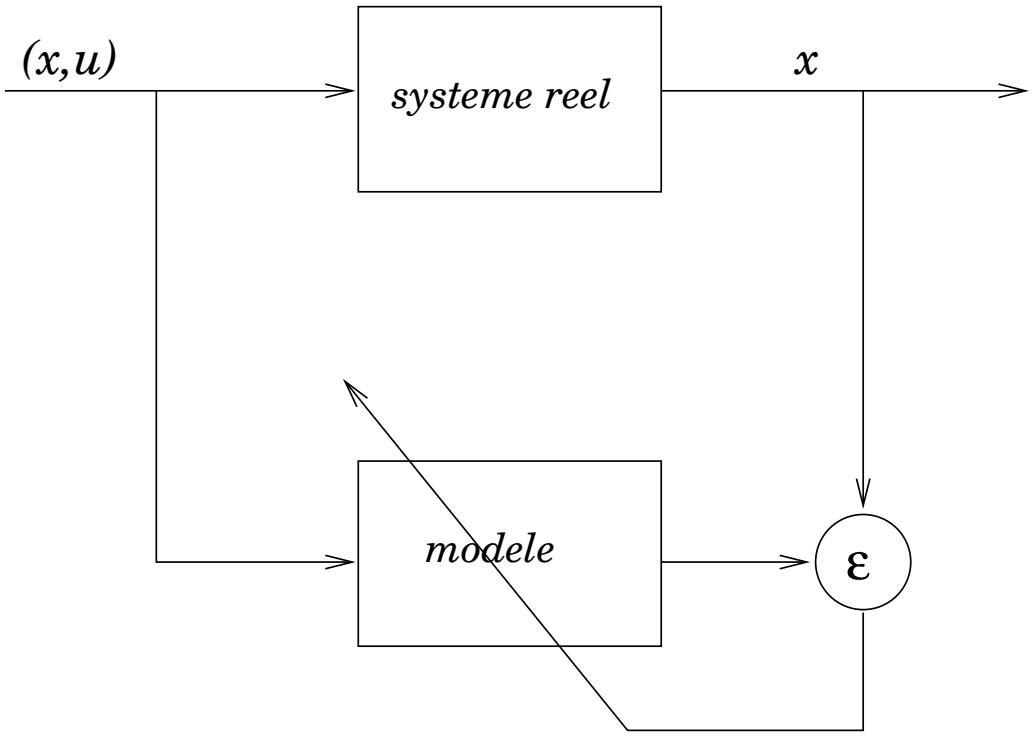


FIG. 7.2 – Conception d'un modèle du système à contrôler.

7.4 Connaissances nécessaires sur le système à contrôler

7.4.1 Connaissances des dérivées partielles du système à contrôler

Bien sûr, cet algorithme nécessite l'accès aux dérivées partielles du système à contrôler, cela veut dire que dans la réalité il est nécessaire de disposer d'un modèle du système. En pratique, il faudra donc procéder en plusieurs phases (figures 7.2 et 7.3) :

1. construction d'un modèle (en effet en pratique le système à contrôler est rarement connu explicitement),
2. utilisation du modèle pour entraîner un contrôleur

A la fin de la première phase, nous obtenons un modèle qui n'est pas exact. Reste à savoir en quoi son utilisation peut tout de même conduire à un contrôleur exact. Que ce soit directement lors de la phase 2, ou bien lors

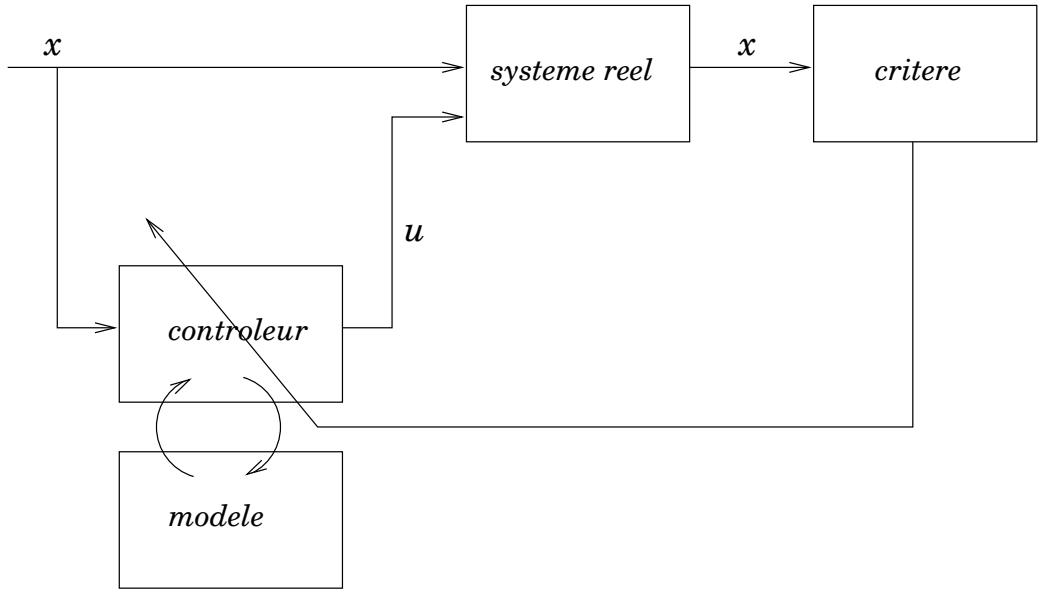


FIG. 7.3 – Conception d'un contrôleur à partir d'un modèle du système à contrôler.

d'une phase intermédiaire *d'utilisation du modèle seul pour la conception du contrôleur* (figure 7.4).

C'est une problématique qui se pose souvent en pratique : quel écart peut-on se permettre entre le modèle *interne* (ie : le modèle du système à contrôler) et la réalité pour obtenir un contrôleur du système réel après apprentissage ?

Cette question trouve en partie une réponse dans la section 8.2, spécifique à l'utilisation de perceptrons affines par morceaux. Il y est fait un lien entre l'écart du modèle à la réalité et l'importance des forces qui mènent le système à l'équilibre : plus il est possible d'obtenir un contrôle stable et plus on peut se permettre un écart important entre le modèle de référence et la réalité.

7.4.2 Conséquences de la minimisation du risque empirique plutôt que du risque théorique

7.4.2.1 Généralités

Plusieurs fois dans les chapitres précédents il a été question de l'erreur commise en minimisant $\widehat{I\!I}$ plutôt que $I\!I$.

Il paraît en effet naturel que plus on dispose d'une classe de fonctions $Q(x, \lambda)$ (dépendant du paramètre $\lambda \in \Lambda$) permettant d'émuler un profil complexe, plus le minimum atteint sur un échantillon de points $(x_1, \ldots, x_\ell)$ d'un

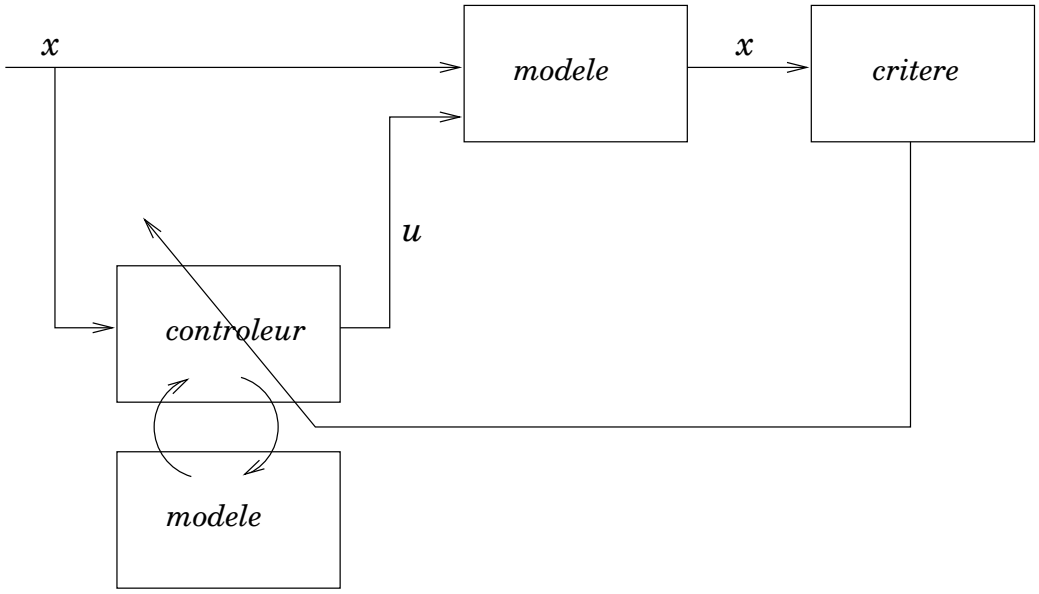


FIG. 7.4 – Phase intermédiaire lors de laquelle seul le modèle est utilisé.

ensemble $\mathcal{X}$ pour un critère fixé n'est pas valable sur tout $\mathcal{X}$.

Ceci a été formalisé par Vapnik et Chervonenkis autour de la quantification d'une mesure de la capacité d'une classe de fonctions à émuler des non linéarités : la dimension de Vapnik Chervonenkis (notée VCdim). Le recours à cette grandeur permet d'encadrer le *risque théorique* (minimum atteint pour un critère théorique) par des bornes dépendant du *risque empirique* (minimum atteint pour la version empirique de ce critère) et de la VCdim de la classe de fonctions utilisée.

Ces bornes ont d'abord été développées dans le cadre de classifieurs, et sont peu à peu étendues à d'autres gammes de classes de fonctions.

7.4.2.2 Cas des classifieurs

Définition 26 (Dimension de Vapnik - Chervonenkis) *La VCdim d'une classe de classifieurs $Q(z, \lambda)$ de $\mathcal{Z}$ dans $\{0, 1\}$ paramétrée par $\lambda \in \Lambda$ est le nombre maximum h de vecteurs quelconques $(z_1, \ldots, z_h)$ qui peuvent être séparés en deux classes de 2^h façons par $Q(z, \lambda)$.*

Dans ce cadre, il existe de nombreux encadrements plus ou moins fins du

risque théorique par le risque empirique ; citons l'un d'entre eux :

$$P\left(\sup_{\lambda\in\Lambda}\left\|\int Q(z,\lambda)dF(z)-\frac{1}{\ell}\sum_{i=1}^{\ell}Q(z_i,\lambda)\right\|>\varepsilon\right)\leq 4\exp\left(\ell\left(\frac{h\left(\ln(2\ell/h)+1\right)}{\ell}-\varepsilon^2\right)\right)$$

Ce type de résultat exige une réflexion sur la VCdim de la classe de fonction que l'on utilise dans le cadre d'un apprentissage qui minimise un critère fixé. Dans le cas des RNF, la VCdim croît en fonction du nombre de neurones utilisés.

Remarque.
Si l'on cherche à utiliser les PAP comme des classifieurs, la démonstration du théorème 8 qui fourni le nombre de cellules polyhédrales de la partition initiale d'un PAP peut sans doute être prolongée pour calculer la VCdim de familles de PAP.

7.4.2.3 Cas de l'apprentissage d'un contrôleur

Le passage de la minimisation du risque théorique au risque empirique a été clairement identifié section 7.3.2 : il s'agit du remplacement de l'intégrale des coûts des trajectoires sur l'ensemble des points initiaux muni d'une distribution de probabilité ξ_0 par une somme sur un échantillon de points initiaux choisis suivant la distribution ξ_0.

Comme la remarque précédente le souligne, il y a un lien fort entre la VC-dimension d'une famille de fonctions et le nombre de non linéarités qu'elle est capable d'émuler. Dans le cadre des PAP cette grandeur est pilotée par le nombre d'unités cachées (comme le montre le théorème 8 : une famille de PAP est l'ensemble des PAP ayant un nombre fixé d'unités cachées).

De façon qualitative, cela veut dire que si on choisit de contrôler un système dynamique par un PAP avec un "trop grand nombre" d'unités cachées, alors on pourra se trouver dans une situation où le risque empirique (minimisé par l'algorithme d'apprentissage) est très faible, mais que le risque théorique correspondant (non calculable explicitement) peut être plus grand.

Chapitre 8

Garanties apportées par les PAP dans le cadre du contrôle

8.1 Objectif : contrôle optimal

L'apprentissage d'un contrôle optimal par RNF repose sur l'optimisation d'une fonction de coût en certains points seulement de l'espace qui nous intéresse. Typiquement, alors que le critère à minimiser est de la forme :

$$\mathcal{C}(W) = \int_{\Omega} \mathbf{C}(\Psi_W, \omega)\, d\omega$$

Dans le cadre du contrôle optimal et pour un système dynamique discret, le critère à minimiser est le suivant (reprise de l'équation (2.6)) :

$$\mathcal{I} = \int_{x_0 \in \Omega_0} \sum_{n=0}^{\infty} L_n(x_n, u_n) d\xi(x_0) \tag{8.1}$$

C'est-à-dire lorsqu'on met en place un contrôle en boucle fermée par RNF :

$$\mathcal{I}(W) = \int_{x_0 \in \Omega_0} \sum_{n=0}^{\infty} L_n(x_n, \Psi_W(x_n)) d\xi(x_0)$$

L'espace des points qui sont mis en jeux par l'optimisation est donc $\Omega = \Omega_0^{I\!N}$. On se contente pourtant d'opérer une optimisation sur un sous-ensemble de Ω de la forme $\bar{\Omega}_0^N$ où $\bar{\Omega}_0 \subset \Omega_0$ et $N \ll \infty$.

Mais *le RNF obtenu ne risque-t-il pas d'être surtout efficace sur un ensemble de départ de trajectoires $\hat{\Omega}_0$ qui est un voisinage de $\bar{\Omega}_0$ et pour une durée finie des trajectoires ?*

Cette interrogation "pragmatique" met souvent un frein à l'utilisation des RNF dans le cadre du contrôle :

- d'une part car elle est très en deçà de ce qu'affirment les résultats théoriques du contrôle linéaire et une partie du contrôle non linéaire,
- d'autre part car dans un contexte industriel, ce type de proposition est "inquiétante" dans la mesure où elle n'apporte que peu de garanties sur l'efficacité de la mise en place d'un contrôle par RNF.

J'ai senti ces deux aspects dans le cadre de directions de la recherche de grands groupes (entre autres Renault, PSA, Volvo, Siemens) : les bureaux d'étude en automatique utilisaient essentiellement des techniques de contrôle linéaires justement pour être rassurés sur l'efficacité du contrôle obtenu, les équipes opérationnelles n'étant pas prêtes à mettre en place un contrôle risquant de casser un moteur de prototype.

Pourtant la mise en place d'un contrôle non linéaire par RNF me parait tout à fait envisageable dans l'industrie automobile. En effet, l'utilisation de techniques de contrôle linéaire pour asservir le système fortement non linéaire qu'est un moteur de voiture en pensant que cela va être parfaitement efficace paraît extrêmement douteux. En effet, rien ne garantit (il est possible d'obtenir ce genre de garanties, mais cela demande de très bien connaître le comportement non linéaire d'un moteur : d'avoir au moins une idée précise de la distance entre la réalité non linéaire et le modèle linéaire utilisé) que les non linéarités non prises en compte par l'approximation linéaire du système ne vont pas le conduire dans des zones d'instabilité chronique. L'efficacité observée sur des cas concrets des contrôleurs linéaires mis en place impliquait que le comportement du genre de moteur contrôlé peut se ramener sans commettre une trop grande erreur à un comportement affine par morceaux : et les PAP sont justement très adaptés à ce genre de situation.

Suite aux travaux expérimentaux menés sur le contrôle par PAP dans le cadre de cette thèse, je suis convaincu que les PAP offrent le même genre de garanties que les contrôleurs linéaires.

De façon générale, il faut constater la faible intégration des RNF dans les domaines où ils sont utilisés : il y a par exemple assez peu de résultats statistiques sur les RNF alors que ce sont des estimateurs très répandus. Dans le cadre des perceptrons, alors que l'on sait qu'il s'agit d'approximateurs universels (c'est-à-dire que pour toute fonction lisse f et pour tout ε il existe un perceptron $\Psi(f, \varepsilon)$ qui émule f à ε près), nous manquons de résultats sur le comportement statistique (par exemple le biais et l'écart type) du nombre de neurones cachés, des matrices de poids et vecteurs de seuils obtenus pour un procédé de minimisation donné.

Le formalisme associé aux PAP (surtout par l'intermédiaire des matrices et vecteurs de saturation), permet justement d'utiliser sur les PAP une partie des techniques utilisées dans les preuves des propriétés de modèles linéaires. Le fait que les PAP mettent en jeux plusieurs cellules polyhédrales complique les preuves en question, mais les PAP ont la qualité de donner un point d'accroche à l'adaptation de la plupart des propriétés des modèles linéaires aux PAP (et par leur intermédiaire, aux perceptrons en général).

Mon idée fût donc de tenter d'obtenir pour les PAP des garanties sur la stabilité du contrôle qu'ils génèrent propres à les mettre au moins au niveau de contrôleurs linéaires.

Pour cela je me suis imprégné des résultats de stabilité des contrôles linéaires et non linéaires qui me semblaient proches des PAP, dans l'objectif de trouver des techniques de preuves qui s'appliqueraient à ces derniers :les résultats en contrôle flou [Leung *et al.* , 1998] et en contrôle par morceaux [Rockafellar, 1988], [Rockafellar, 1988] ainsi que d'autres sur des fonctions de Lyapunov construites par morceaux (comme [Blanchini, 1995], [Blanchini & Miani, 1996], ou [Byrnes, 1998]) ainsi que les lectures de [Jagannarthan, 1996], [Polycarpou, 1996], [Levin, 1993], [Levin & Narendra, 1996], [Byrnes *et al.* , 1997], [Freeman *et al.* , 1998], [Ledyaev & Sontag, 1997], [Hirsh, 1974], et [Wredenhagen & Belanger, 1994].

Les systèmes linéaires par morceaux sont étudiés par quelques auteurs (dont E.D. Sontag [Sontag, 1981]) ; la lecture de ces études donne à penser que les PAP devraient offrir certaines garanties de stabilité, mais que cela serait plus difficile à obtenir pour les systèmes dynamiques discrets que pour leurs homologues continus. En effet, les systèmes discrets "permettent" de "sauter" d'une cellules polyhédrale à une autre non nécessairement mitoyenne, ce qui complique les démonstrations.

Finalement je décidais de m'appuyer sur les travaux de Johansson et Rantzer ([Johansson & Rantzer, 1997], [Rantzer & Johansson, 1997] et [Johansson & Rantzer, 1996]). L'un d'entre eux permet en effet de résoudre directement la question des systèmes dynamiques continus, et donne donc des directions efficaces pour obtenir un résultat sur les homologues discrets.

Ce chapitre expose ces résultats.

8.2 Stabilité d'un contrôle par PAP

8.2.1 Résultats de Rantzer et Johansson et existence de matrices frontières

Les résultats de cette section proviennent directement de la transposition des travaux de Anders Rantzer et Michael Johansson en ce qui concerne la stabilité des systèmes continus et sont le prolongement de leur transposition dans l'univers des PAP en ce qui concerne la stabilité des discrets.

Le principal des preuves de leurs travaux est exposé dans deux rapports techniques intitulés *Piecewise linear quadratic optimal control* [Rantzer & Johansson, 1997] et *Computation of piecewise quadratic lyapunov functions for hybrid systems* [Johansson & Rantzer, 1996].

Les notations utilisées dans ces deux rapports sont différentes entre elles, elles diffèrent aussi de notations issues des PAP.

Rappelons alors le théorème 9 de [Rantzer & Johansson, 1997] qui concerne les approximations affines par morceaux de systèmes lisses :

Théorème 11 (Johanssen et Rantzer, 1997) *Soit $x(t) \in I\!R^n$ une trajectoire $\mathcal{C}^1$ par morceaux du système $\dot{x} = f(x)$ et supposons qu'il existe un découpage en cellules polyhédrales décrit par des matrices frontières $\bar{A}_i$, $i \in I_0$) tel que :*

$$||f(x) - (A_i x + a_i)|| \leq \varepsilon_i \, ||x|| \Leftrightarrow x \in \mathcal{C}^{(i)}, \text{ on note } \bar{A}_i = \begin{bmatrix} A_i & a_i \\ 0 & 0 \end{bmatrix}$$

Si il existe des $\gamma_i > 0$, des matrices symétriques U_i, $\bar{U}_i$, V_i et $\bar{V}_i$ dont les éléments sont tous positifs ou nuls, et une matrice T telle que $\bar{P}_i = \bar{E}_i' T \bar{E}_i$ and $P_i = E_i' T E_i$ satisfassent pour les i de I_0 :

$$E_i' U_i E_i < P_i < \gamma_i \, Id \tag{8.2}$$

$$-E_i' V_i E_i > A_i' P_i + P_i A_i + 2\varepsilon_i \gamma_i \, Id \tag{8.3}$$

et pour les autres indices :

$$\bar{E}_i' \bar{U}_i \bar{E}_i < \bar{P}_i < \gamma_i \, Id \tag{8.4}$$

$$-\bar{E}_i' \bar{V}_i \bar{E}_i > \bar{A}_i' \bar{P}_i + \bar{P}_i \bar{A}_i + 2\varepsilon_i \gamma_i \, Id \tag{8.5}$$

alors $x(t)$ tends exponentiellement vers 0.

La démonstration se reprend facilement : on construit $\mathcal{V}(x) = \bar{x}'\bar{P}_i\bar{x}$ pour x dans $\mathcal{C}^{(i)}$, alors (8.2) et (8.4) impliquent :

$$(\bar{E}_i\bar{x})'\bar{U}_i(\bar{E}_i\bar{x}) < \mathcal{V}(x) < \gamma_i\bar{x}'\bar{x}$$

et d'autre part les deux autres inégalités impliquent elles (en notant $\tilde{a}_i(x) = \left[\ f(x) - (A_ix - a_i) \quad , 0\ \right]'$) :

$$\begin{aligned}
\frac{d}{dt}\mathcal{V}(x) &= \frac{d}{dt}(\bar{x}'\bar{P}_i\bar{x}) \\
&= \bar{x}'(\bar{A}_i'\bar{P}_i + \bar{P}_iA_i)\bar{x} - 2\bar{x}'P_i\tilde{a}_i(x) \\
&< -\bar{x}'(2\varepsilon_i\gamma_i\,\mathrm{Id} + \bar{E}_i'V_i\bar{E}_i)\bar{x} + 2\,||x||\,\gamma_i\varepsilon_i\,||x|| \\
&\leq -(\bar{E}_i\bar{x})'\bar{V}_i(\bar{E}_i\bar{x}) \\
&\leq -\delta\,||x||^2 \\
&\leq -\frac{\delta}{\gamma_i}\,\mathcal{V}(x)
\end{aligned}$$

□

Ce passage à une approximation de f par une fonction affine par morceaux est justifié par le théorème 10 de la même publication :

Théorème 12 (Johanssen et Rantzer, 1997) *Soit $f \in \mathcal{C}^1(X, I\!R^n)$ telle que $\partial f/\partial x$ soit borné sur X. Supposons que le système $\dot{x} = f(x)$ soit globalement asymptotiquement stable, alors pour une partition suffisamment fine en cellules polyhédrales, il existe une solution $\gamma_i, U_i, \bar{U}_i, V_i, \bar{V}_i$ et T aux inégalités (8.2) à (8.5).*

Les cellules polyhédrales sont définies par Rantzer et Johansson via des matrices que nous appellerons "matrices frontières" :

Propriété 4 (Existence de matrices frontières) *Soit le PAP Ψ_W défini par $\Psi_W(x) = \Phi(W^{(2)}\,\Phi(W^{(1)}x + b^{(1)}) + b^{(2)})$. Alors il existe des matrices $(G_i, i \in I)$, $(D_i, i \in I)$ et $(F_i, i \in I)$ appelées matrices frontières associées à Ψ_W telles que :*

$$\left\{ \begin{array}{ll} G_ix + D_i \geq 0 & , \quad \textit{pour } x \in \mathcal{C}^{(i)} \\ F_i \begin{pmatrix} x \\ 1 \end{pmatrix} = F_j \begin{pmatrix} x \\ 1 \end{pmatrix} & , \quad \textit{pour } x \in \mathcal{C}^{(i)} \cap \mathcal{C}^{(j)} \end{array} \right. \tag{8.6}$$

et il existe des matrices A_i et B_i telles que :

$$\Psi_W(x) = A_ix + B_i, \ \textit{lorsque } x \in \mathcal{C}^{(i)} \tag{8.7}$$

Cette propriété vient immédiatement de l'existence d'un partition terminale pour les PAP.

Il est possible d'exprimer explicitement les matrices G, D et F en fonction de $W^{(1)}$, $b^{(1)}$, $W^{(2)}$ et $b^{(2)}$ à partir des équations (5.6) appliquées non seulement à la partition initiale mais aussi à la partition terminale. Ces relations explicites ne sont pas exprimées ici car l'application des équations (5.6) à chaque cellule de la partition initiale génère une condition du type "si l'hyperplan $\mathcal{H}$ coupe la cellule $\mathcal{C}^{(i)}$" par unité de la dernière couche du PAP et par cellule de la partition primaire. □

On va noter I_0 le sous ensemble de I qui contient les indices des cellules qui contiennent le point zéro (on remarque que $i \in I_0$ implique $e_i = 0$ et $f_i = 0$).

Il est ainsi possible d'utiliser les résultats de Rantzer et Johansson.

8.2.2 Système dynamique à temps continu

Exemple 8.2.2. *(équivalent PAP de l'exemple de [Rantzer & Johansson, 1997]).* On construit le PAP de matrices et vecteurs de poids :

$$W^{(1)} = \frac{1}{10}\begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \quad , \quad b^{(1)} = \frac{1}{10}\begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$$

$$W^{(2)} = 10\begin{bmatrix} -5 & -2 & -4 \\ -1 & 20 & -2 \end{bmatrix} \quad , \quad b^{(2)} = \begin{bmatrix} 3 \\ 21 \end{bmatrix}$$

donc les trajectoires sont celles de :

$$\dot{x} = \begin{cases} \begin{bmatrix} -5 & -4 \\ -1 & -2 \end{bmatrix} x, & \text{lorsque } x_1 < 0 \\ \\ \begin{bmatrix} -2 & -4 \\ 20 & -2 \end{bmatrix} x, & \text{lorsque } x_1 \geq 0 \end{cases} \tag{8.8}$$

on va donc pouvoir générer la fonction de Lyapunov suivante :

$$\mathcal{V}(x) = \begin{cases} x' \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix} x, & \text{lorsque } x_1 < 0 \\ \\ x' \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix} x + 7x_1^2, & \text{lorsque } x_1 \geq 0 \end{cases} \tag{8.9}$$

qui nous garantie la convergence de (8.8). En effet en notant :

$$P = \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}, \eta = 7, A_1 = \begin{bmatrix} -5 & -4 \\ -1 & -2 \end{bmatrix}, A_2 = \begin{bmatrix} -2 & -4 \\ 20 & -2 \end{bmatrix}$$

on vérifie (avec $C = [1, 0]$) que :

$$\begin{array}{rcl} P & > & 0 \\ A_1'P + PA_1 & < & 0 \\ P + \eta\, C'C & > & 0 \\ A_2'(P + \eta C'C) + (P + \eta C'C)A_2 & < & 0 \end{array}$$

◇ ..... *Fin de l'exemple 8.2.2*

Théorème 13 *Les théorèmes 11 et 12 s'appliquent directement aux PAP dans les circonstances suivantes :*

- *les PAP bouclés sur eux-mêmes : $\dot{x} = \Psi_W(x)$, dans ces circonstances, les ε_i sont nuls et le résultat est plus direct,*
- *les PAP en tant que contrôleurs : $\dot{x} = f(x, \Psi_W(x))$.*

Dans ce second cas, la démarche est de remplacer d'abord $f(x, y)$ par un PAP $\Psi_{W'}$, avec une erreur de ε_i sur chaque cellule $\mathcal{C}^{(i)}$. On obtient alors le système dynamique suivant : $\dot{x} = \Psi_{W'}(x, \Psi_W(x))$ qui est un PAP à trois couches cachées. Il se ramène néanmoins à une fonction continue affine par morceaux explicite en fonction des W' et W. On peut la nommer $\Psi_{W' \cup W}$, on est alors ramené à l'étude de $\dot{x} = \Psi_{W' \cup W}(x)$ qui approche f à ε_i près sur chaque $\mathcal{C}^{(i)}$: on est bien dans le cadre du théorème 11. □

Appliqué aux PAP, ce résultat est d'abord de nature existentiel : il existe un système d'inégalités matricielles "à la Riccati" pour le contrôle par PAP.

Mais il s'agit aussi d'un résultat constructif qui permet de vérifier la stabilité d'un système contrôlé par un PAP. En effet, le calcul des matrices frontières associées à une composition de PAPs (comme $\Psi_{W' \cup W}$) peut se faire explicitement moyennant un peu de programmation combinatoire[1]. Il est alors possible de vérifier numériquement si le système est stable ou non.

[1]Dans le cadre de cette thèse, un package MapleV permettant de réaliser des compositions de couches de PAP (totalement automatisé pour les dimensions 1 et 2 seulement).

8.2.3 Système dynamique à temps discret

La problématique en temps discret est rendu plus complexe dans la mesure où les trajectoires peuvent "sauter" d'une cellule polyhédrale à une autre. Lorsqu'on considère le système dynamique suivant (σ est l'opérateur "avance" : $\sigma x_n = x_{n+1}$) :

$$\sigma x = \Psi_W(x) \tag{8.10}$$

Il convient de définir les cellules qui précèdent une cellule donnée :

Définition 27 (Prédécesseurs d'une cellule) *Soit Ψ_W un PAP paramétré et $(\mathcal{C}^{(i)}, i \in I)$ les cellules polyhédrales associées. Les prédéceseurs de la cellule $\mathcal{C}^{(i)}$ selon les trajectoires de (8.10) constituent l'ensemble des indices de cellules $\mathfrak{P}(i)$ défini par la relation :*

$$j \in \mathfrak{P}(i) \Leftrightarrow \left(\exists x \in \mathcal{C}^{(j)} / \sigma x \in \mathcal{C}^{(i)}\right) \tag{8.11}$$

Ceci permet d'obtenir le résultat suivant :

Théorème 14 *Soit le système (8.10) caractérisé par un ensemble de cellules polyhédrales $(\mathcal{C}^{(i)}, i \in I)$ telles que :*

$$x \in \mathcal{C}^{(i)} \Rightarrow \Psi_W(x) = A_i x$$

Supposons qu'il existe des matrices R_i, Q_i et P_i symétriques définies positives telles que :

$$A_i' P_i A_i + Q_i < \inf\left(P_j, j \in \mathfrak{P}(i)\right) \tag{8.12}$$

alors la fonction $\mathcal{V}(x) = x' P_i x$ lorsque $\sigma^{-1}x$ est dans $\mathcal{C}^{(i)}$ est une fonction de Lyapunov pour (8.10).

En effet, $d\mathcal{V}(x) = \mathcal{V}(\sigma x) - \mathcal{V}(x) = x'(A_i' P_i A_i - P_j)x$ avec $x \in \mathcal{C}^{(i)}$ et $\sigma^{-1}x \in \mathcal{C}^{(j)}$. Et (8.12) garantie $d\mathcal{V}(x) < -x' Q_i x$. □

Les hypothèses de ce théorème sont plus difficiles à vérifier car elles nécessitent la construction des ensembles des prédécesseurs de chaque cellule polyhédrale engendrée par un PAP. Ce résultat peut être étendu pour obtenir des formulations similaires à celles de la section précédente.

8.3 Application des critères de stabilité

Les résultats de la section précédente permettent d'étendre aux PAP les propriétés habituellement associées aux contrôleurs linéaires.

En pratique, on peut envisager deux utilisations de ces propriétés :

Une modification de l'algorithme d'apprentissage. L'idée serait de n'accepter une modification des poids d'un PAP que lorsqu'elle conduit à de nouveaux poids assurant la stabilité du contrôle engendré. En effet, si le contrôle généré n'est pas stable, rien ne garantit que la prochaine itération de l'apprentissage ne va pas conduire le système vers un état profondément instable qui va perturber la poursuite de l'adaptation des poids.

Une vérification a posteriori. La mise en œuvre du point précédent est complexe, on peut se contenter de vérifier, en fin d'apprentissage, que le PAP obtenu engendre bien un contrôle stable.

C'est le second point qui a été mis en place dans le cadre de mes travaux de thèse chez Renault pour la mise au point automatique d'un contrôle par PAP. Il a donc été possible d'observer des jeux de poids qui vérifiaient ou pas les inégalités des théorèmes précédents.

Ainsi, la figure 8.1 montre l'évolution du critère sur trois types de trajectoires :

- les trajectoires associées à un PAP initialisé à partir d'un contrôleur linéaire du système à contrôler (bleu),
- les trajectoires associées à un PAP vérifiant les inégalités du théorème 14 (vert),
- les trajectoires associées à un PAP ne vérifiant pas les inégalités du théorème 14 (rouge).

Le critère utilisé est un coût quadratique instantané.

Chaque ensemble de trajectoires est représenté par trois courbes. En effet à chaque instant, le critère à minimiser est calculé sur l'ensemble des trajectoires observées. On conserve le minimum, le maximum et le moyenne des valeurs du critère. Cela conduit donc à la courbe les minima, celle des maxima (toutes deux en pointillés) et celle des moyenne des valeurs du critère (en trait plein).

Cette implémentation montre que dans les cas observés : il y a bien équivalence entre le fait de vérifier les inéquations des théorèmes de stabilité et le fait d'engendrer un contrôle stable.

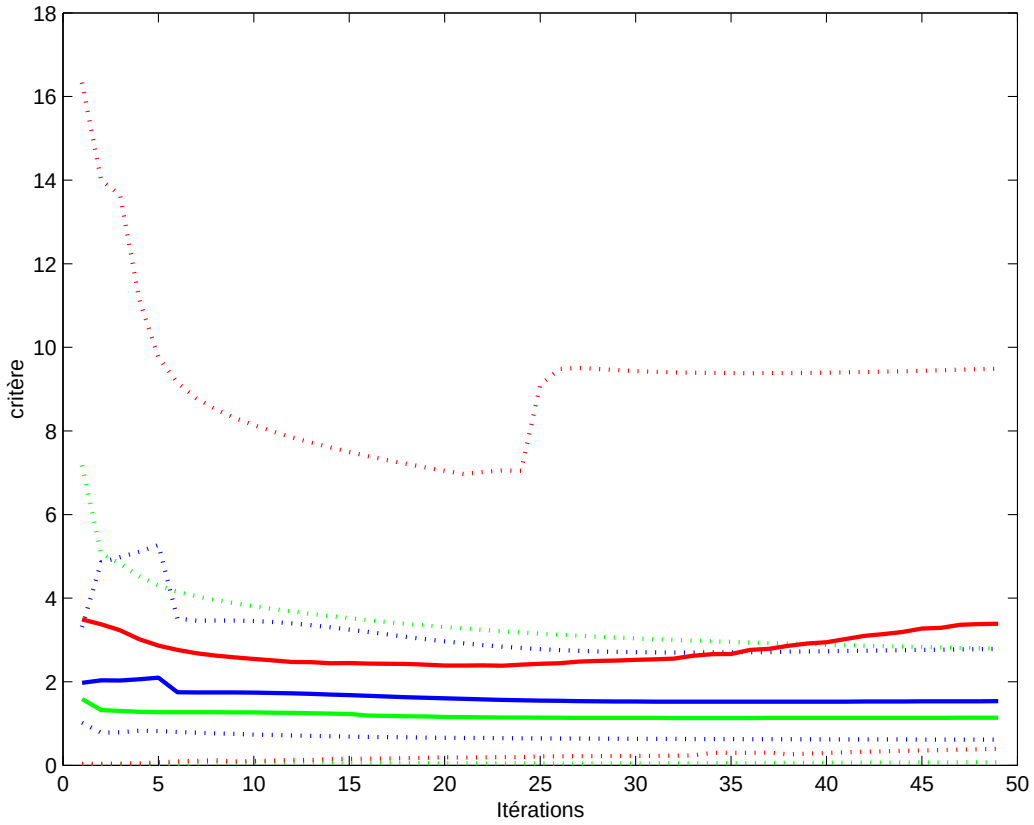


FIG. 8.1 – Trajectoires associées à différents PAPs (cf chapitre 8.3) : les courbes rouges correspondent à un PAP ne vérifiant pas les inéquations du théorème de stabilité, les courbes bleues correspondent à une initialisation émulant un contrôle linéaire stable, et les courbes vertes à un PAP qui vérifie après adaptation les inéquations de stabilité.

Troisième partie

Contexte, enjeux et mise en œuvre

Chapitre 9

Application en contrôle moteur automobile

Dans le cadre d'un contrat CIFRE entre l'ENS et la Direction de la Recherche de Renault, l'objectif était d'étudier les possibilités de concevoir automatiquement des contrôleurs non linéaires.

L'application cible était la conception d'un contrôleur de couple sur une voiture de série (Laguna v1 munie d'un moteur essence F4R).

Avant d'aller plus loin dans la présentation de la mise en place technique des moyens nécessaires à réaliser ce double objectif, il convient de mettre en perspective la situation de l'industrie automobile face à la mise au point automatique de contrôleurs et aux réseaux de neurones formels.

9.1 Utilisation des réseaux de neurones formels

Si les RNF sont assez présents dans de nombreux domaines d'activité, ils sont peu utilisés dans l'industrie automobile (illustration 9.1).

Actuellement leurs utilisations se limitent à de l'identification ou de la classification.

Le travail exposé ici à donné lieu à deux brevets 99 03930 et 97 15802 et 6 plis d'huissiers (cf annexe).

9.1.1 Place des RNF chez Renault.

Depuis 8 ans maintenant le Groupe Logiciel du Département Electronique de la Direction de la Recherche de Renault explore les capacités des RNF en matière de contrôle.

année	nb brevets
1986	8
1987	19
1988	52
1989	97
1990	189
1991	209
1992	267
1993	285
1994	394
1995	450

FIG. 9.1 – Nombre de dépôts de brevets concernant les réseaux de neurones de 1986 à 1995.

Cette thèse sous contrat CIFRE prend essentiellement place après une thèse soutenue par Eric Chapuis suivi d'un stage long de Nicolas Conso qui ont aboutis à l'émulation d'un contrôleur PID préexistant par un perceptron. Cette application, sans aller vers une grande complexité théorique ou technologique, a permis d'ouvrir les esprits aux possibilités des RNF en matière de contrôle.

En effet, cela a permis d'équiper un véhicule d'un "contrôleur neuronal" et de le "montrer" en train de rouler. Il démontrait la faisabilité de la chose, ou en tout cas donnait un aspect concret et matériel à la recherche dans ces directions. Bien sûr il ne s'agissait pas de la construction automatique d'un contrôleur ni de l'utilisation des propriétés d'émulation non linéaire des RNF ; mais le Groupe Logiciel du Département Électronique de la Direction de la Recherche disposait d'un démonstrateur.

En outre, le Groupe Logiciel s'était investi dans un consortium en collaboration avec Volvo, PSA et la scoiété Ricardo dont l'objectif était la réalisation d'un prototype de contrôleur mettant en avant l'aspect adaptatif des RNF. Je suis allé à la dernière réunion de ce consortium, pour la livraison par Ricardo du prototype en question qui avait été réalisé en partie par des ingénieurs d'EDS (Electronic Data Systems Corporation). Ce contrôleur était réalisé à partir d'un réseau de type CMAC, et aucune réflexion n'avait été menée sur la "stratégie d'adaptation", qui est le point capital dans ce contexte : quand est-il nécessaire d'effectuer un nouvel apprentissage ? sur quelles bases et avec quel type d'algorithmique ?

N'ayant pas les mêmes objectifs que les autres constructeurs du consortium, celui-ci a été arrêté en 1999.

La présence du Pr. Azencott au conseil scientifique de la DR de Renault a contribué pour beaucoup à éveiller l'intérêt du Groupe Logiciel et du Département Electronique en général pour les RNF. Les publications des autres constructeurs dans la revue SAE[1], ont aussi participé à maintenir un intérêt sur les RNF.

année	RNF nb publications	RNF & contrôle nb publications
1989	1	1
1993	1	1
1994	1	1
1996	1	1
1997	3	2
1998	47	15
1999	68	12
2000	93	16
2001	106	21
2002	119	30

FIG. 9.2 – Publications SAE traitants des RNF et du contrôle

9.2 Utilisation de perceptrons affines par morceaux hors du contexte du contrôle : le logiciel PiAf

9.2.1 Contexte de l'utilisation de PiAf

Au fur et à mesure de l'avancée des travaux exposés ici, des besoins opérationnels en modèles non linéaires auto adaptatifs ont émergé : autour de la mise au point des moteurs et des stratégies de contrôle (modélisation de certains organes véhicule).

La décision a été prise de réaliser un logiciel d'approximation non linéaire par PAP qui répondrait à une bonne partie de ces problématiques. Celles-ci ont en commun :

– il s'agit de problématiques de modélisation non linéaire encastrée dans une stratégie plus large, souvent déjà implémentée sous MATLAB/SIMULINK,

[1]SAE Transactions, (SAE : Society of Automotive Engineers) dont je suis un des relecteurs depuis 2000.

– les données disponibles ne correspondent pas directement aux entrées et sorties du modèle non linéaire à réaliser mais à celle de la stratégie plus vaste.

Une méthodologie a été dégagée qui a été implémentée à partir de mes spécifications à RENAULT dans le cadre d'un programme Java interfacé avec MATLAB/SIMULINK. Une automatisation de MATLAB/SIMULINK permet de réaliser automatiquement les tâches suivantes :

constitution de la base de données : un module SIMULINK spécifique remplace la fonction à émuler au sein du modèle SIMULINK dans lequel la fonction se placera. Ce module utilise les éléments déjà existants de la stratégie afin de constituer une base de données des entrées et sorties de la fonction à émuler à partir des entrées et sorties de la stratégie dans sa globalité.

apprentissage du PAP sous Java sur la base de données en question : celle-ci est découpée entre une base d'apprentissage et une base de test.

utilisation du PAP après apprentissage : un module SIMULINK spécifique peut utiliser un export des poids du PAP calculés par le programme Java. Ainsi en mode "utilisation", le modèle SIMULINK est indépendant du programme Java qui a permis l'apprentissage du PAP.

Ces trois étapes permettent d'isoler la tâche d'apprentissage hors de SIMULINK. Le programme réalisé l'a été en Java afin de faciliter le portage d'un système d'exploitation à un autre, d'autant que des expérimentations au Groupe Logiciel d la DR de Renault réalisées à l'époque montraient que les temps de calcul en Java étaient comparables aux temps de calcul en C++ pourvu que l'on s'en donne les moyens (utilisation d'un compilateur du Java en code natif au besoin).

Ce logiciel nommé PiAf pour *PIecewise AFfine neural modelization* s'est révélé particulièrement efficace et assez simple à manipuler (surtout en mode interactif) par des ingénieurs de recherche non spécialisés dans l'étude des RNF.

9.2.2 Principe de fonctionnement

Les étapes de l'apprentissage d'un PAP par PiAf exploitaient au mieux les résultats théoriques alors obtenus sur le positionnement des neurones dans l'espace de départ :

Parcours rapide de la base d'apprentissage. Il s'agit dans un premier temps de parcourir les données disponibles afin de dégager des zones

de l'espace sur lesquelles le comportement à émuler est "presque linéaire". Ceci peut se faire par des moyens statistiques classiques : découpage itératif de l'espace avec régressions linéaires et tests de fiabilité de ces régressions. Un PAP émulant ces zones est alors automatiquement construit.

Apprentissage. Une descente de gradient est effectuée afin d'optimiser les poids obtenus à l'issue de l'étape précédente. Cette étape est particulièrement rapide puisque l'initialisation est efficace.

Ajout de neurones. Les erreurs du PAP obtenu sont localisées dans l'espace d'entrée. Un mode interactif ou un seuil d'erreur admissible permet de sélectionner les zones de l'espace associées à des erreurs "inadmissibles" : des neurones sont ajoutés dans ces zones, l'étape précédente est relancée.

Bien sûr, l'étape d'ajout de neurones peut mener à un *overfitting* (la complexité du PAP augmente et dépasse celle de la fonction à émuler), mais il s'est avéré que les erreurs commises *localisées dans l'espace de départ* semblent être assez naturellement interprétées par les experts métiers (ils "savent" que des phénomènes turbulents ou particulièrement bruités se produisent dans telle zone et n'y demandent pas d'ajout de neurones alors qu'une erreur dans une zone connue pour être "lisse" peut se voir attribuer des neurones supplémentaires).

Sans l'avis d'expert, on cours le risque de sur paramétrer le modèle. Le recours à une base de test permet d'éviter que ce sur paramétrage soit trop important.

9.2.3 Méthodologie de parcours rapide d'un ensemble de données

L'objectif de la première étape de PiAf est d'identifier rapidement des "zones homogènes" de la base d'apprentissage. De nombreuses techniques sont disponibles pour effectuer ce genre de tâche.

En notant $(X_i, Y_i)_i$ les points de la base d'apprentissage (on cherche le paramétrage W d'un PAP qui permettent d'émuler $Y_i = \Phi_W(X_i)$) la motivation principale était de disposer d'un algorithme permettant d'isoler des zones de l'espace produit $X \times Y$ qui sont de "bons candidats" pour les cellules polyhédrales induites par les poids du PAP.

Etant donné que sur chacune de ses cellules polyhédrales, le PAP a un comportement affine, cela revient à isoler des zones de l'espace $X \times Y$ pour

lesquelles X et Y sont reliés par une fonction affine. L'algorithmique retenue effectue les opérations suivantes :

(a) l'espace de départ est découpé en "rectangles" (les frontières de ces zones sont orthogonales aux axes de la base naturelle de X),

(b) pour chaque rectangle R :

(1) on effectue un découpage en "sous rectangles" : si X est de dimension n, on dispose de n découpages possibles, un par axe. On a donc n partitions "candidates" $R = R(i) \cup R'(i)$ $(1 \leq i \leq n)$ en coupant R par un hyperplan passant par le milieu de sa frontière selon le $i$ème axe de X.

(2) on effectue une régression $\mathcal{R} : Y = AX + B$ sur tous les points de R (si on n'en dispose pas déjà) plus une sur chaque ensemble de points dans $R(i)$ et dans $R'(i)$ (respectivement notées $\mathcal{R}_i : Y = A_iX + B_i$ et $\mathcal{R}'_i : Y = A'_iX + B'_i$).

(3) on teste $\mathcal{R}$ contre $\mathcal{R}_i \cup \mathcal{R}'_i$ grâce à un test de Fisher sur les résidus et on conserve le résultat le plus efficace.

(c) on dispose d'une nouvelle partition en rectangles de X, on revient donc au point (b).

En pratique, le nombre maximum de passage par le point (b) est un paramètre de l'algorithme, fixé autour de 5. Cela permet d'obtenir un nombre pas trop important de rectangles partitionnant l'espace.

La génération d'un jeu de poids initial pour un PAP émulant chaque régression obtenue sur chaque rectangle se passe ainsi :

(1) Choix des poids de la première couche permettant de découper l'espace selon les rectangles obtenus,

(2) Choix des poids de la seconde couche permettant au PAP d'émuler la fonction obtenue par régression sur chacun des rectangles.

Il est clair que des améliorations éventuelles sont à effectuer essentiellement autour de la méthode de découpage (en se permettant des zones non rectangles). Elles impacteraient l'algorithmique de calcul des poids de la première couche, actuellement grandement facilitée par le fait que les frontières sont positionnées suivant les axes. C'est l'apprentissage qui va ajuster les frontières des cellules au mieux.

9.2.4 Apprentissage par descente de gradient

L'algorithme d'apprentissage utilisé par PiAf est un gradient avec moments. Il peut être modifié conformément à l'état de l'art (recours à un calcul

de hessienne, au gradient conjugué, etc).

Le rôle de cet apprentissage est plus d'ajuster les poids que de les amener vers des valeurs radicalement différentes. En effet, si la première étape n'est pas parvenue à isoler des zones de l'espace sur lesquelles la relation entre X et Y est affine, le recours à un PAP n'est pas véritablement justifié.

Néanmoins, l'amélioration de la descente de gradient permet de "garantir" que les performances du PAP seront "au moins aussi efficaces" que celle d'un perceptron plus classique.

9.2.5 Ajout de neurones cachés

Une fois l'espace partitionné en rectangles et l'apprentissage effectué, le PAP est utilisé sur une base de test. On mesure l'écart entre ses approximations et les valeurs réelles des Y. Sur chaque axe de l'ensemble de départ, on peut identifier un intervalle sur lequel l'erreur est particulièrement importante.

Dans sa version actuelle, PiAf laisse à l'utilisateur la tâche de déterminer la localisation exacte de ces intervalles.

Si l'utilisateur identifie sur l'axe numéro i de l'espace de départ une forte concentration des erreurs dans l'intervalle $[A, B]$, il peut la signaler au système PiAf en ajoutant l'instruction suivante dans un fichier de configuration : `add@i[A,B]`.

Il est alors possible de relancer PiAf qui va :
(a) ajouter un neurone dans cet intervalle,
(b) relancer une descente de gradient.

Le neurone est ajouté assez simplement. En effet en notant w^* son vecteur de poids et b^* son biais, on a $||w^*|| = 2/(B-A)$ et $\langle b^*, w^* \rangle = ||w^*||^2 (A+B)/2$.

Le nouveau neurone est ajouté de façon à avoir (lors de sa création) peu d'influence sur la sortie. Il est uniquement "positionné" dans l'espace de départ mais aucun calcul permettant d'ajuster l'action du PAP compte tenu de l'erreur n'est effectué. C'est l'apprentissage effectué immédiatement après l'ajout du neurone qui va ajuster les valeurs des poids.

C'est cette étape de l'algorithmique qui devrait être principalement améliorée. En premier lieu la localisation de zones d'erreur "importante" sur chaque axe pourrait être réalisée par un algorithme de clustering simple. En deuxième lieu les zones d'erreur pourraient être localisées autrement que

selon chaque axe : une analyse en composantes principales dans l'espace de la base de test dont chacun des points aurait une pondération proportionnelle, l'erreur commise en ce point par le PAP permettrait par exemple de déterminer un "axe principal d'erreur" selon lequel le nouveau neurone serait ajouté.

Si cette capacité à ajouter des neurones afin de diminuer localement l'erreur commise par un PAP est séduisante, il ne faut pas oublier que trop pratiquée elle conduit de façon sûre à un sur-apprentissage. En effet il sera toujours possible d'ajouter des neurones jusqu'à ne plus commettre d'erreur.

Mais le PAP ainsi constitué aura effectué un apprentissage "par cœur" sans aucun intérêt.

Les moyens d'éviter cela sont :

- n'ajouter un neurone qu'après une réflexion sur le sens physique de la présence d'une non linéarité à l'endroit en question,
- aller vers une méthode "*à la bootstrap*" : séparer les données disponibles en deux sous ensembles (base d'apprentissage et de test) plusieurs fois indépendamment et réaliser les étapes actuellement effectuées par PiAf pour chaque couple d'ensembles ainsi généré. Une méthodologie reste à déterminer qui permettrait de "fusionner" les PAP obtenus.

9.2.6 Bilan du développement et de l'utilisation de PiAf

L'intérêt des PAP ne se limite pas au contexte du contrôle : ils sont aussi très utile lorsqu'il s'agit d'émuler une fonction dont on ne connaît que l'action en certains points de l'espace d'entrée. En effet, le positionnement de neurones supplémentaires dans des zones explicites de l'espace (là où l'erreur d'approximation est maximale) fait passer le PAP d'une "boîte noire" (qui réalise une émulation sans dire grand chose sur la façon dont elle y parvient) à une "boîte grise" (qui peut fournir de nombreuses informations sur la façon dont elle parvient à émuler un phénomène et donc sur le phénomène lui-même).

La mise au point de PiAf montre comment le point de vue développé pendant ce travail de thèse donne un caractère d'intelligibilité "local" aux perceptrons. Ceci a été rendu nécessaire par l'objectif d'utiliser des perceptrons dans le cadre du contrôle mais est exploitable dans tous les autres domaines d'utilisation des RNF.

9.2.7 Comparaisons entre les PAP et les autres perceptrons hors du cadre du contrôle

Même si les résultats théoriques obtenus ici concernent les PAP, on peut s'interroger sur la possibilité de leur utilisation pour des perceptrons aux fonctions d'activation plus classiques.

Cela revient à se poser plusieurs questions, qui vont être passées en revue dans cette section ; elles découlent toutes de celle-ci : *quel est l'impact du choix d'une fonction d'activation plutôt qu'une autre ?*

9.2.7.1 Comparaison empirique simple

En premier lieu et comme point de départ possible à cette réflexion, voici le résultat d'une comparaison empirique de trois méthodes :

- un PAP initialisé à partir d'une simplification de la méthode PiAf : les neurones sont placés afin de générer une partition uniforme de l'espace de départ,
- un perceptron à fonctions d'activation en tangente hyperbolique est initialisé exactement de la même façon,
- un perceptron dont la première couche est initialisée aléatoirement mais dont les poids de la seconde couche sont calculé grâce à une méthode "à la Poggio" (afin d'initialiser une matrice $W^{(2)}$ telle que $\Phi(W^{(2)} \cdot \Phi(W^{(1)} X)) = Y$ alors que X, Y et $W^{(1)}$ sont fixés, on prend : $W^{(2)} = \Phi^{-1}(Y) \cdot (\Phi(W^{(1)} X))^{-1}$ sous certaines conditions, cf [Poggio & Girosi, 1986]),

Une fois ces trois perceptrons initialisés, ils sont soumis à une descente de gradient classique.

Remarque sur l'implémentation. Ces simulations proviennent de codes MATLAB en partie réalisés pour MIRIAD *Technologies* dans le cadre de l'implémentation d'estimateurs de densités de probabilité, et en partie spécifiquement implémentés pour cette étude.

Cette comparaison est aussi simple que possible : elle est faite sur un petit jeu de données issues d'une fonction simple, il s'agit d'une ligne brisée bruitée par une gaussienne centrée d'écart type 1/10.

Cent bases de données (chacune de cents point équirépartis) de référence sont générées par réalisations du bruit. A titre indicatif, une de ces bases est représentée figure 9.3. La simplicité de ces bases de données devrait permettre de mieux expliciter les résultat obtenus.

Trois neurones cachés sont attribués à chacun des type de réseaux de neurone, sur lesquels on réalise un apprentissage pour chaque base de données,

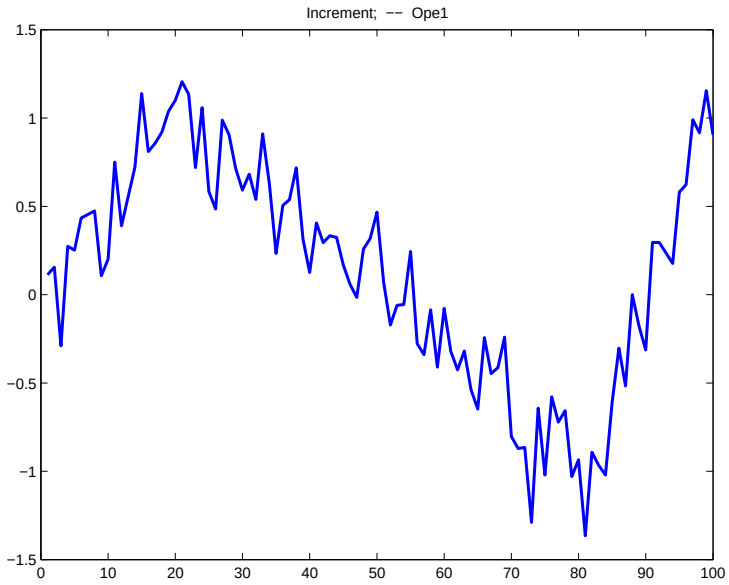


FIG. 9.3 – Une des 100 bases de données de référence pour la comparaison empirique entre PAP et perceptrons.

les apprentissages sont limités à 300 itérations, sachant qu'un critère d'arrêt en cas de diminution trop faible de l'écart à la cible est implémenté. Cent *courbes d'apprentissages* sont ainsi obtenues pour chaque type de perceptron. Une telle courbe d'apprentissage est constituée par l'évolution d'un critère de distance entre le perceptron et l'échantillon de points disponibles, itération par itération (Figure 9.4). Le critère d'apprentissage est proportionnel à l'écart quadratique entre les points de la base de donnée et les sorties du perceptron.

Afin de comparer les courbes d'apprentissage obtenues, on réalise une moyenne (itération par itération) de toutes les courbes d'apprentissage d'un même type de perceptron. Bien que cet indicateur ne soit pas significatif de la totalité du comportement des perceptrons pendant les apprentissages, cela s'avère ici largement suffisant pour aboutir à quelques conclusions (Figure 9.5).

En effet, il est possible de tirer quelques constatations simples des courbes d'apprentissage en question :

- l'initialisation par positionnement des neurones cachés est plus performant qu'une initialisation aléatoire,
- le positionnement équi-réparti des neurones est mieux adapté aux PAP

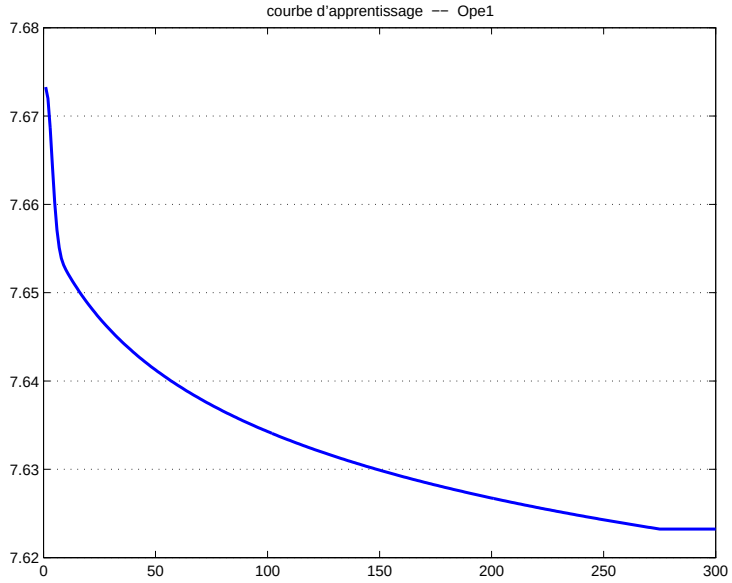


FIG. 9.4 – La courbe d'apprentissage d'un des PAP.

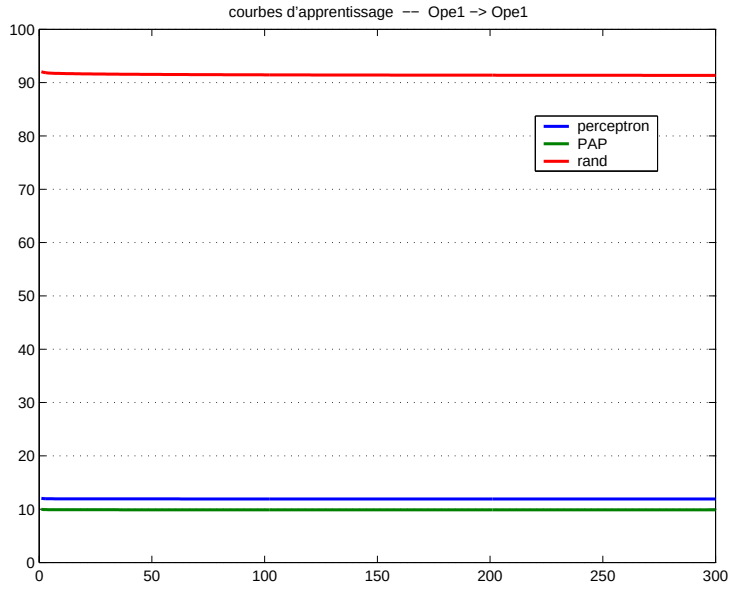


FIG. 9.5 – Les trois courbes d'apprentissage moyennées.

qu'aux perceptrons à activation sigmoïdale,
– à même nombre d'itérations, les PAP sont plus performants que les perceptrons sigmoïdaux (sur cet exemple).

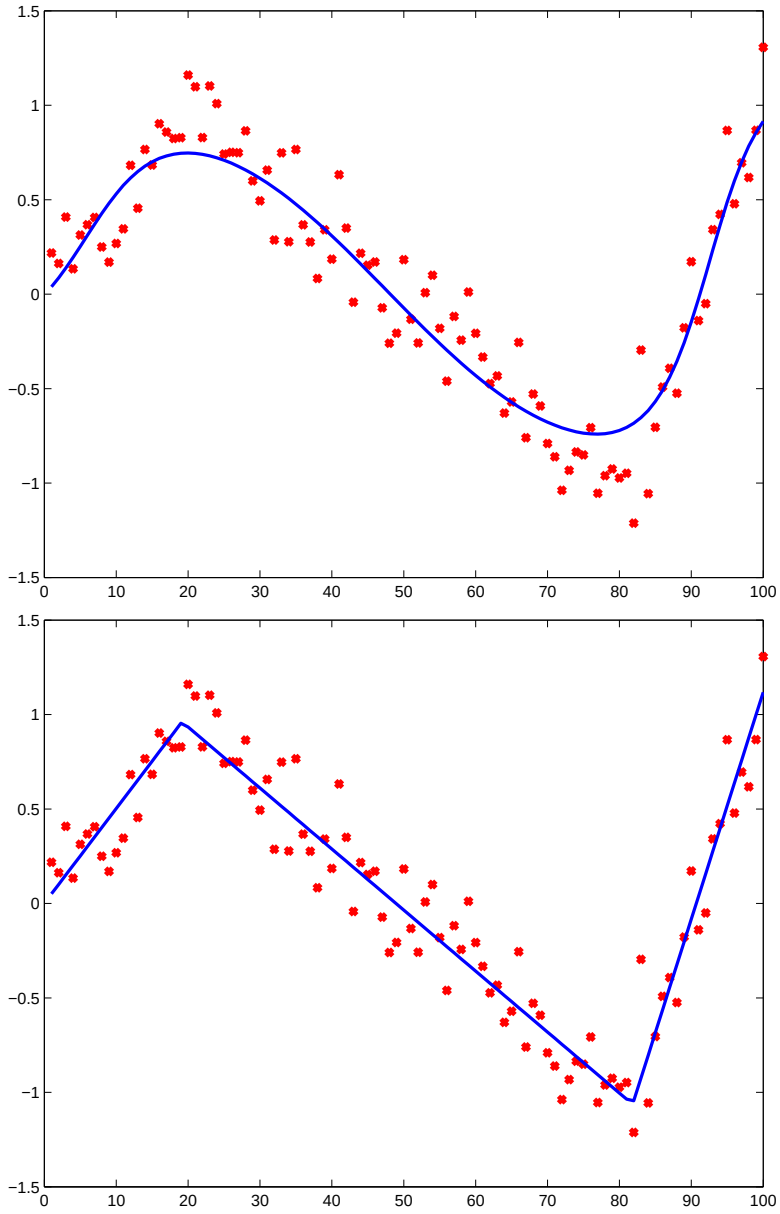


FIG. 9.6 – Exemple du PAP (haut) et du Ψ (bas) en fin d'apprentissage sur le même échantillon de données.

9.2.7.2 Rôle de la fonction d'activation

Il faut se garder de tirer du dernier point une conclusion générale sur la performance relative des PAP et des perceptrons sigmoïdaux, car la forme de sa fonction d'activation prédispose un perceptron à émuler telle ou telle

classe de fonctions. Le cas choisit était issu de lignes brisées, très proches dans leurs formes des perceptrons affines par morceaux, le cas extrême inverse étant celui où la fonction à émuler est beaucoup plus lisse (par exemple un sinus bruité : Figure 9.7) : il est clair qu'un perceptron de la même famille aura plus de faciliter à l'émuler qu'un PAP (qui ne pourra jamais le faire exactement).

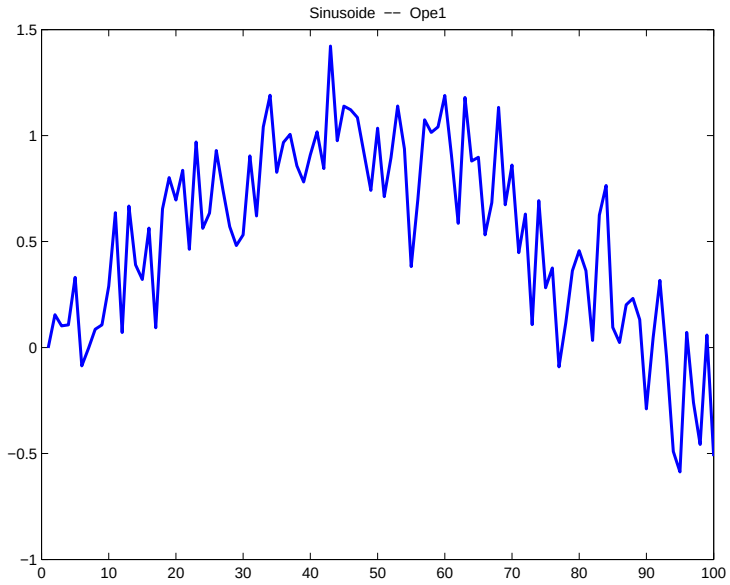


FIG. 9.7 – Un exemple d'une des base sinusoïdales de référence.

Le cas du sinus bruité, traité de façon similaire (100 bases de 100 points équirépartis) inverse l'efficacité relative des PAP et perceptrons sigmoïdaux (Figure 9.8), l'initialisation aléatoire restant moins performante.

Ce qui émerge ici est que l'exploitation d'une connaissance a priori sur la forme de la fonction à émuler apporte un plus lorsqu'elle est prise en compte pour choisir la forme de la fonction d'activation. Par exemple, il vaut mieux choisir un perceptron dont les fonctions d'activations sont $\mathcal{C}^\infty$ si la fonction à émuler est aussi $\mathcal{C}^\infty$. Mais comme la section 9.2.7.4 le souligne, la fonction d'activation ne suffit pas toujours à rendre compte de propriétés connues de la fonction à émuler.

9.2.7.3 Comparer les trajectoires dans l'espace des paramètres

La question de la comparaison de perceptrons de différentes fonctions d'activation ne se pose pas qu'en terme de performance *pas à pas* pendant l'apprentissage, car cette performance n'est que le résultat de la dynamique

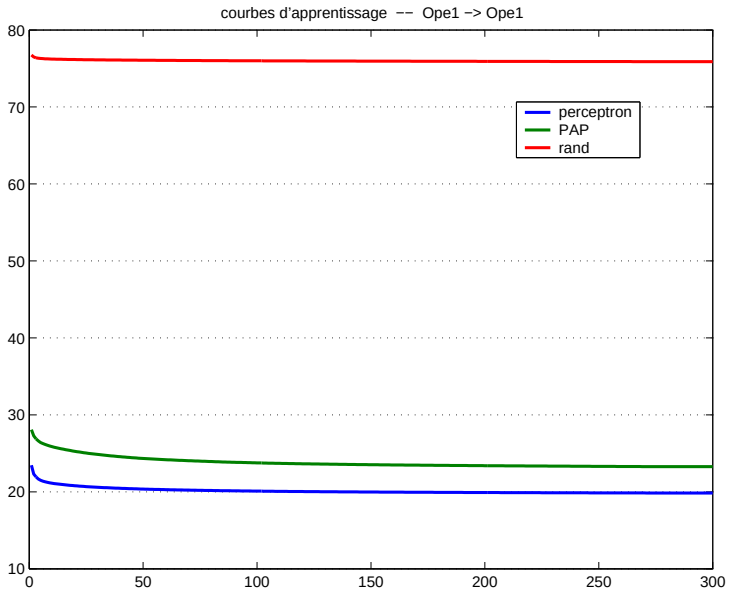


FIG. 9.8 – Les différentes courbes d'apprentissages pour les bases sinusoïdales.

des paramètres du perceptron, qui effectuent une descente de gradient sur une fonction de coût déterminée par la base d'apprentissage.

En effet, les trois perceptrons étudiés dans cette section ont chacun 10 ($= 3 \times 3 + 1$) paramètres ; il est donc tout à fait possible de représenter une paramétrisation d'un des PAP étudiés comme un point de $\mathbb{R}^{10}$. Pendant l'apprentissage, le PAP va effectuer une trajectoire dans cet espace, dont la dynamique est guidée par le gradient de :

$$\mathcal{I} = \int_{(x,y)\in\mathcal{X}\times\mathcal{Y}} \left|\left|\Phi\left(W^{(2)} \cdot \Phi\left(W^{(1)} \cdot x + b^{(1)}\right) + b^{(2)}\right) - y\right|\right|^2 d\mu(x,y)$$

où $\mathcal{I}$ est vue comme une fonction de l'ensemble Θ des paramètres du perceptron (c'est-à-dire $\Theta = \{W^{(1)}, b^{(1)}, W^{(2)}, b^{(2)}\}$). Dans le cas usuel, l'échantillon de couples (x, y) est fini, la mesure μ est donc une somme de diracs.

La dynamique à laquelle est soumise chaque perceptron dépend donc clairement de la forme de ses fonctions d'activation Φ (la figure 9.9 en donne des exemples), la question de la comparaison des PAP et des perceptrons sigmoïdaux devient donc rapidement celle de la comparaison des deux champs de gradients associés. Pour un même échantillon de couples (x, y), y a-t'il des différences notables de trajectoires entre perceptrons sigmoïdaux et PAP ?

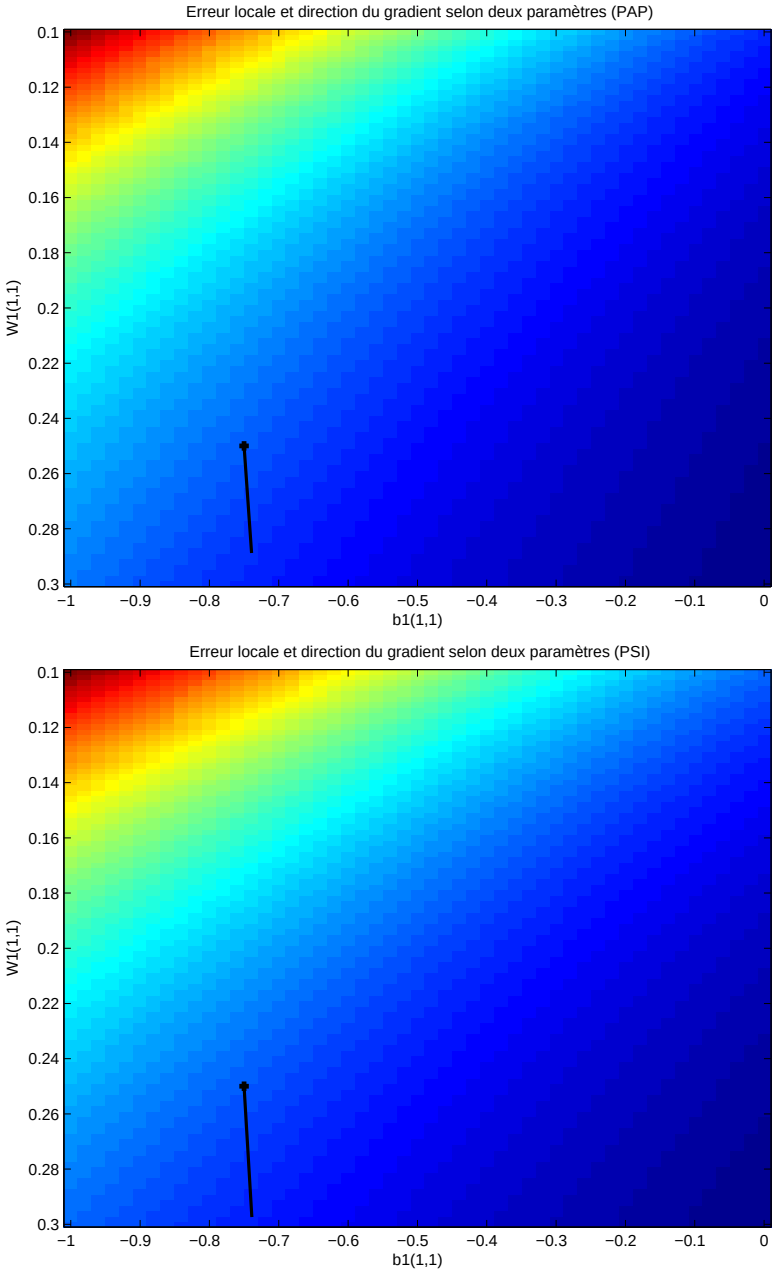


FIG. 9.9 – Exemple de l'erreur locale pour deux paramètres d'un PAP (premières coordonnées de $W^{(1)}$ et de $b^{(1)}$, en haut) et pour les mêmes paramètres du Ψ (en bas) ainsi que la dynamique locale (trait noir ; le point de départ est un point) sur le même échantillon de données issu de la base sinusoïdale.

Une première difficulté consiste à définir la notion de *trajectoires notablement différentes*. Une définition assez peu contraignante de trajectoires proches est :

Définition 28 (Equivalence de trajectoires limites) *Soient deux perceptrons Ψ_1 et Ψ_2 définis tous deux par l'équation :*

$$\Psi_k(x) = \Phi_k\left(W^{(2)} \cdot \Phi_k\left(W^{(1)} \cdot x + b^{(1)}\right) + b^{(2)}\right)$$

sur un ensemble $\mathcal{X}$, et dont les fonctions d'activations Φ_1 et Φ_2 sont différentes. Pour toute valeur θ de Θ (l'ensemble des valeurs possibles pour les paramètres $W^{(1)}, b^{(1)}, W^{(2)}$ et $b^{(2)}$), notons $\Psi_k \triangleleft \theta$ la limite, lorsqu'elle existe, de la trajectoire des paramètres en fin d'apprentissage.

On dira que les trajectoires limites de ces deux perceptrons sont équivalentes lorsque : pour tout point de départ θ_0 de Θ et lorsque les apprentissages en question convergent :

$$\Psi_1 \triangleleft \Psi_2 \triangleleft \theta_0 = \Psi_1 \triangleleft \theta_0$$

et réciproquement en échangeant les rôles de Ψ_1 et Ψ_2

Si on considère les fonctions d'apprentissage empiriques, qui sont toujours munies d'un critère d'arrêt, il faut noter que cette définition est tout à fait pratique et appliquée.

Cas d'un seul neurone. Il ne s'agit pas ici de mener une étude complète sur ce genre de relations entre perceptrons à fonctions d'activations différentes, mais de mettre en avant des raisons pour lesquelles le choix de la fonction d'activation ex nihilo est difficile.

Pour des perceptrons très simples en dimension un à un seul neurone et une seule fonction d'activation, des cas extrêmes se présentent déjà. En effet, de tels perceptrons s'écrivent :

$$\Psi_k(x) = \phi_k(ax + b)$$

Considérons une descente de gradient simple qui suit le critère (on notera $\varepsilon(x, y) = \phi_k(ax + b) - y$) :

$$\mathcal{I}_k = \int_{(x,y)\in\mathcal{X}\times\mathcal{Y}} (\phi_k(ax + b) - y)^2 \, d\mu(x, y) \tag{9.1}$$

un point d'équilibre (a_∞, b_∞) pour l'apprentissage vérifie :

$$\int_{(x,y)\in\mathcal{X}\times\mathcal{Y}} (\phi_k(a_\infty x + b_\infty) - y)\, \phi_k'(a_\infty x + b_\infty) \, d\mu(x, y) = 0 \tag{9.2}$$

C'est la différence entre les dérivées des fonctions d'activation qui va donner lieu à des lieux d'équilibre différents.

Cette constatation permet de construire l'exemple suivant, pour lequel il n'y a pas équivalence des convergences des versions sigmoïdale et affine par morceaux d'un perceptron.

Afin de l'illustrer, le paysage à optimisé a été calculé en entier (ce qui a pu se faire relativement rapidement vu qu'il n'y avait qu'un seul neurone). Le seul paramètre de cette expérimentation est la base d'apprentissage, qui a volontairement été choisie très déséquilibrée (figure 9.10). Afin de permettre des illustrations parlante, seul les poids du neurone de la première couche évoluent ($b_1^{(1)}$ et $W_1^{(1)}$).

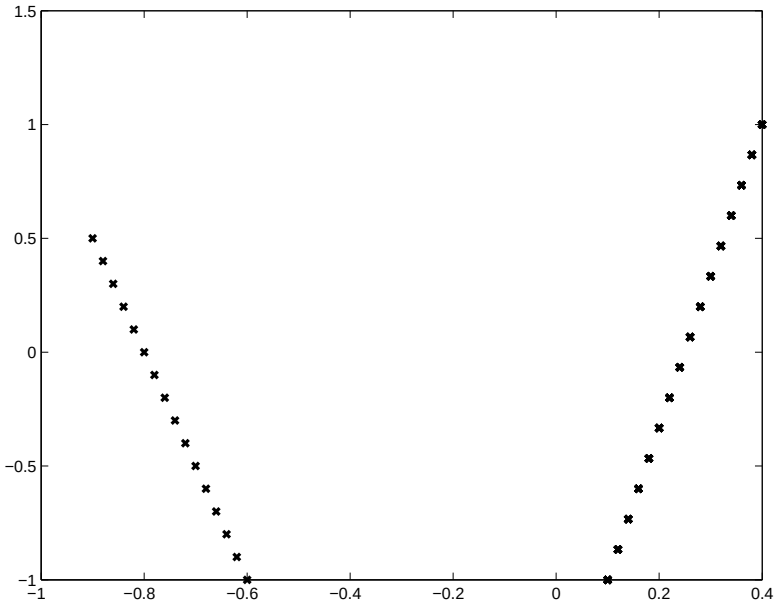


FIG. 9.10 – Base d'apprentissage pour l'exemple.

La figure 9.11 montre le paysage pour un PAP et la figure 9.12 pour un perceptron classique. Le paysage figure "vu de haut" ; il y a aussi une valeur de paramètre (la même : $b_1^{(1)} = -15/10$ et $W_1^{(1)} = -18/10$) et a direction du gradient associé : on vir clairement la bifurcation des trajectoires.

Nous avons donc construit deux perceptrons Ψ_1 et Ψ_2 tels que : pour un point de départ $\theta_0 = (-15/10, -18/10)$, le PAP conduit ses poids dans un bassin d'attraction et le perceptron classique dans un autre. Ensuite, aucun

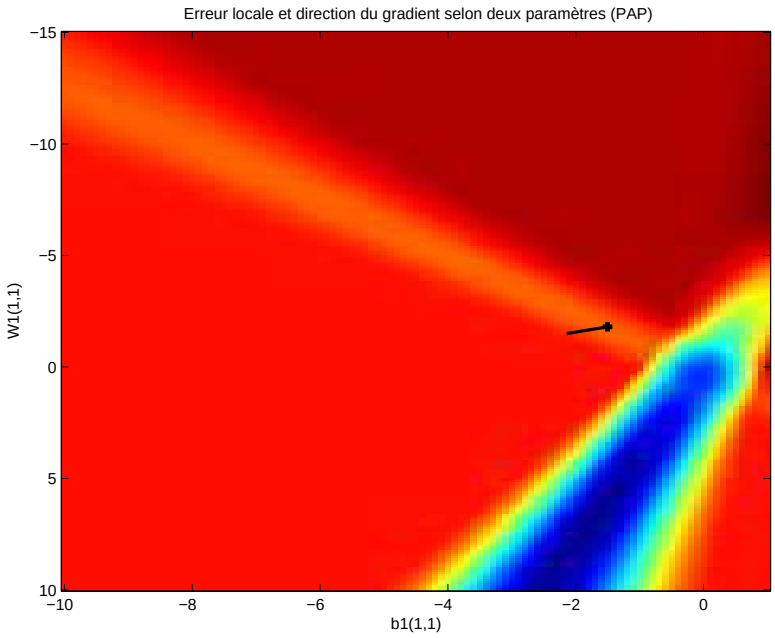


FIG. 9.11 – Paysage d'optimisation pour un PAP : le point de coordonnées $(-15/10, -18/10)$ va rejoindre l'optimum local qui est en haut à gauche.

échange de dynamique ne pourra les en faire sortir.

9.2.7.4 Trajectoires et apprentissage

La section 9.2.7.3 ouvre la possibilité d'intégrer des contraintes dans un apprentissage. Ainsi que la section 9.2.7.2 l'a souligné, ces contraintes peuvent être naturellement issues de propriétés connues de la fonction à émuler. Dans le cas du contrôle par un PAP, et même si la fonction dont le gradient défini le système dynamique des paramètres pendant l'apprentissage est différente (comme la section 7.3 le souligne, elle est issue d'une descente sur un dépliement des trajectoires contrôlées), cela reste un champ de vecteurs dans l'espace des paramètres. Or, les contraintes obtenues en 8.2 qui permettent de garantir la stabilité d'un système dynamique contrôlé par un PAP s'écrivent comme des inégalités sur les matrices de poids du PAP : elles définissent donc des *zones interdites* dans l'espace des paramètres.

Il serait très intéressant de pouvoir s'assurer que les perceptrons issus de l'apprentissage ne sont jamais dans ces zones : cela revient à être certain qu'ils vérifieront bien les propriétés de stabilité qui nous intéressent.

Ce point de vue permet de travailler directement sur le champ engendrant

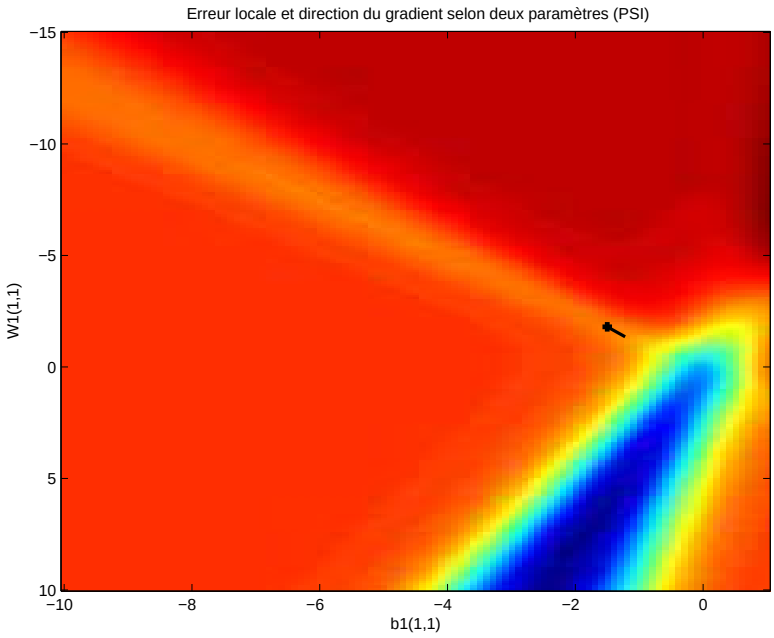


FIG. 9.12 – Paysage d'optimisation pour un perceptron classique : le point de coordonnées $(-15/10, -18/10)$ va rejoindre l'optimum local qui est en bas à droite.

la dynamique afin de rendre les *zones interdites* "repoussantes". Il s'agit là d'un prolongement naturel du travail effectué ici, qui nécessite de gros développements comme le montrent les travaux de ce type déjà existants (par exemple les travaux d'Ishi Amari dans le cadre de la *descente naturelle de gradient* [AMARI., 1998] ou de Nicolas Seube dans la cadre de la *théorie de la viabilité*).

9.3 Première étape du contrôle adaptatif : la pré mise au point

Comme la section 10.1 le détaille, l'énergie disponible dans une automobile provient du moteur de façon directe ou indirecte (via le stockage temporaire dans une batterie). En ce qui concerne les moteurs à essence, c'est l'allumage d'un mélange d'air et de carburant qui produit cette énergie ; pour les moteurs diesel, il y a auto-allumage de carburant à l'état gazeux.

La qualité de la combustion (sa rentabilité en termes énergétiques aussi bien que la quantité des produits de cette réaction —dont les polluants—) dépend à la fois de l'état du moteur (température, vitesse, pressions, etc) et

de paramètres "de contrôle" qui déterminent pour les moteurs à essence la quantité d'air et d'essence présents dans le cylindre et le moment de l'explosion (durée de l'injection, ouverture de vannes d'arrivée d'air, pressions ou instant d'allumage de la bougie).

Avant l'utilisation des systèmes d'injection électronique, la relation entre l'écart et les valeurs des "actuateurs" de contrôle était émulée de façon mécanique (réglage du carburateur). Depuis l'utilisation de calculateurs d'injection, d'abord constitués de composants électriques ou électroniques, puis ayant recours à de véritables micro-contrôleurs munis de mémoire vive et de mémoire morte, cette relation peut prendre la forme que l'on veut.

La principale contrainte est de savoir déterminer son graphe, l'objectif est d'émuler la relation état — contrôle qui permettra d'atteindre la meilleur rentabilité énergétique conjuguée à l'émission d'une pollution minimale. D'autres contraintes peuvent s'ajouter comme améliorer le "confort du conducteur" (diminuer les acoups, le bruit, les vibrations, etc). Bien sûr des notions de coût interviennent selon deux axes :

- lors de l'identification du graphe de cette relation : c'est la mise au point (MAP) du contrôleur. Le temps de MAP et le coût en matériel sont pris en compte.
- lors de son implémentation sur les véhicules de série : la demande de temps machine ou de mémoire est aussi une source de dépenses.

9.3.1 Mise au point du contrôleur

Sans aller trop loin dans le détail, les interactions entre les différentes versions des composants d'un moteur et de ses contrôleurs durant leur conception engendrent de nombreux réglages. C'est par itérations successives que l'on aboutit finalement à un moteur et un contrôleur de série.

Ces allers-et-retours sont directement coûteux et de plus entraînent un délai entre la conception d'un nouveau moteur et sa commercialisation.

Pour de nombreuses raisons historiques, les contrôleurs sont décomposés en multiples sous-systèmes, chacun sous la responsabilité d'une équipe. Aujourd'hui, afin de faciliter le réglage par essais-erreurs et en raison de la faible dimensionnalité des problèmes, ces sous-systèmes sont composés de contrôleurs linéaires, extrêmement simples, dont chaque coefficient peut être modifié manuellement.

La recherche de la rentabilité de la combustion ainsi que les contraintes imposées par l'évolution des normes de pollution conduisent à l'augmentation

de la dimension de l'espace d'état des paramètres du moteur. Citons par exemple :

- pour les moteurs à essence : l'injection directe dans les cylindres (plutôt qu'en amont de la soupape), qui permet de faire varier le profil de la fonction d'injection durant le remplissage des cylindres.
- pour les moteurs à essence encore : les moteurs sans arbre à came qui permettent de choisir de façon indépendante les instants de montée et de descente des soupapes.
- pour les moteurs diesel : l'utilisation d'un rail à haute pression (common rail) en amont des injecteurs.
- l'utilisation de matériel de filtrage (pots catalytiques) et de dépollution (NOx-trap).

Toutes ces démarches visent à augmenter le nombre de moyens d'action sur la combustion en général. Contrairement à une idée reçue, elles ne font pas qu'augmenter le nombre de degrés de liberté des problèmes de contrôle, elles en augmentent aussi la complexité.

Cela amène à deux conclusions, apparues tardivement dans le monde de l'industrie automobile :

- la croissance de la dimension des espaces d'état et de contrôle met un frein au réglage "manuel" des contrôleurs, les solutions iront vers des réglages automatisés ou semi-automatisés.
- les grandeurs à contrôler deviennent de plus en plus proche de la réalité de la physique de la combustion, qui met en jeux des phénomènes fortement non linéaires, le recours à des contrôles sinon non linéaires, ou moins linéaires seulement de façon très locale, est nécessaire.

Une des réponses à cette problématique est ce qu'on appelle "l'ingénierie simultanée". Il s'agit de mettre en place une méthodologie de travail qui permette à toutes les équipes de mise au point d'un véhicule de travailler *simultanément* sur la même version de ce véhicule.

Un des outils importants pour ce type de méthode de travail est de rendre l'équipement accessible via un réseau informatique. Ainsi plusieurs équipes pourraient travailler par exemple sur un moteur disposé sur un tel banc d'essai chacune à partir d'un "terminal" distant ; un serveur de type OPC (OLE for Process Control) utilisera ensuite les ressources matérielles disponibles en les partageant parmi les équipes de réglages.

Une autre approche consiste en la réalisation d'un simulateur du véhicule

à mettre au point. Les différents ingénieurs peuvent alors préparer leurs réglages sur ce simulateur, mis à jour au fur et à mesure des leurs réglages. Une phase de "mise au point fine" reste à réaliser sur le véhicule réel. Elle est alors beaucoup plus courte.

Les RNF peuvent être d'une grande aide dans ce processus. En effet, de nombreux sous-systèmes d'un véhicule ne sont pas encore mis en équations (ou bien les équations disponibles sont trop longues à ajuster ou à simuler pour que ce soit envisageable avec les moyens d'aujourd'hui). Les RNF sont des candidats parfaits pour construire des "modèles empiriques" de tels sous-systèmes.

9.3.2 L'expérience Mac Trucks

En 1999, Mac Trucks, filiale américaine de Renault Véhicules Industriels (RVI), a décidé de construire un tel modèle. Il s'agissait d'un modèle des émissions de polluants d'un moteur de camion (diesel), destiné à être utilisé en boucle fermée. En tant qu'expert Renault en matière de RNF (la Direction de la Recherche de Renault a comme clients internes Renault et RVI) , je suis allé rendre visite au Laboratoire du Pr. C. Atkinson à l'université de West Virginie (WVU) qui dispose de moyens d'essais très performants (par exemple des analyseurs de gaz d'émission dynamiques , c'est-à-dire à la volée) et qui est spécialisé dans la modélisation des émissions de polluants [Traver *et al.* , 1999] [Atkinson *et al.* , 1998]. Le Pr. Atkinson avait déjà utilisé avec succès ce type de modélisation en collaboration avec la société Neuridyn. J'ai donc pu rencontrer Christopher Atkinson afin de l'assister dans le montage du moteur sur le banc d'essai d'abord, et surtout pour confronter nos points de vue sur les différentes utilisations possibles des RNF dans l'industrie automobile, essentiellement dans le cadre de la phase de MAP. Ce genre d' échanges est très riche : d'une part, le Pr. Atkinson a une longue expérience et une expertise poussée dans les potentialités d'amélioration des procédés de contrôle moteur ; d'autre part, le contenu des travaux exposés dans ce mémoire lui a montré le rôle que peuvent remplir les RNF dans ce cadre, que ce soit pour une auto-calibration des contrôleurs ou pour la conception de contrôleurs non linéaires.

Les résultats exposés ici m'ont en outre permis de répondre aux questions que se posaient Mac Trucks et la DR de Renault en ce qui concerne la faisabilité de ce modèle par Neuridyn, suite à un entretien et plusieurs vidéo conférences avec les scientifiques de cette société. Des accords de confidentialité m'empêchent de développer plus en détail le modèle utilisé.

Le simulateur a été réalisé avec succès et est actuellement utilisé à Mac

Trucks pour le pré réglage des camions. En effet, les acheteurs de ce genre de véhicules spécifient souvent un paramètrage précis du contrôle moteur allant avec une charge ou un ratio de consommation particulier.

L'autre réponse à cette préoccupation est la mise au point automatique d'un contrôleur, qui diminue le temps de réglage nécessaire à chaque équipe.

9.3.3 Implémentation en série

Un autre aspect du coût intervenant dans le contrôle moteur est celui du support électronique des algorithmes de régulation : la mémoire vive (nombre de variables) et la mémoire de masse (la taille du programme).

Il faut d'abord préciser que l'arrivée de l'informatique dans l'habitacle des voitures (comme la java-car de Delphi ou l'utilisation de windows CE par PSA) permet d'utiliser un ordinateur "déporté" à l'intérieur du véhicule pour exécuter certains calculs via une interface. Bien sûr, il n'est pas aujourd'hui question de laisser s'exécuter une partie aussi critique que le contrôle moteur hors de calculateurs totalement spécifiés par le constructeur. Cette situation peut évoluer, surtout si les protocoles de communication sécurisés actuellement mis au point pour des mini serveurs de réseaux parviennent jusqu'à l'industrie automobile.

Deux aspects des RNF sont intéressants dans cette direction :

- une fois une fonctionnalité algorithmique mise au point, il est possible de construire un RNF qui l'émule. Ce RNF est alors souvent plus petit en mémoire de masse et plus rapide à l'exécution (pourvu qu'il ne soit pas récurrent) que l'original. On peut voir cela comme une méthode de *compression*.
- la propriété des PAP vue précédemment permet de transformer un PAP en fonction affine par morceau. Il peut donc être codé informatiquement via une table d'interpolation (appelée souvent "cartographie"). Ce type de codage a souvent la réputation d'être plus rapide à l'exécution et surtout est is en œuvre par la plupart des calculateurs déjà sur le marché actuellement).

9.4 Utilisation des PAP en contrôle moteur

9.4.1 Démarche

Comme les chapitres précédents l'ont exposé, la relation bijective qui existe entre les PAP et les fonctions affines par morceaux permet de construire un PAP à partir d'une collection de contrôleurs linéaires locaux, et d'autre part de stocker un PAP sous la forme d'une fonction affine par morceaux.

Une démarche naturellement associée à ces possibilités est la suivante :

1. modélisation du système à contrôler
2. conception de contrôleurs linéaires locaux du système, assez grossiers, autour de points de l'espace d'état "importants".
3. construction d'un PAP émulant localement chacun des contrôleurs locaux.
4. apprentissage en ligne du PAP utilisant un modèle du système à contrôler (via ses dérivées partielles).
5. possibilité de stocker les PAP sous la forme d'une fonction affine par morceaux au cas où le stockage dans le contrôleur soit plus rapide et moins coûteux.

Il faut noter qu'un modèle du système à contrôler est nécessaire. Il peut lui-même être émulé par un RNF mis au point sur une base d'essais.

9.4.2 Véhicule

L'application particulière du contrôle du couple d'un moteur essence F4R d'une Laguna de série s'est faite dans le cadre suivant :

Le véhicule est donc muni d'un calculateur de série modifié (STRV2). Il effectue les mêmes opérations que celui de série et permet en outre à un programme en langage C tournant sur un PC (386) sous MS-DOS de modifier certaines variables : notamment celles qui activent les actionneurs.

Le PC, positionné dans le coffre de la voiture, a d'autre part accès à de nombreuses variables d'état du véhicule. Lors de l'implémentation des algorithmes dans le programme en C (qui se fait sur une station de travail puis est compilé avec un compilateur croisé afin de tourner sur un 386 sous DOS), on dispose des variables d'état nécessaires au contrôleur : le couple est estimé en temps réel, les valeurs de consignes déterminées par le calculateur sont connues.

Il est en outre possible d'enregistrer des grandeurs en binaire sur le disque du PC, puis de les transférer sur une station, de les convertir en ascii, puis de les analyser.

L'intérêt est que nous disposons d'une "structure couple", qui est une des options envisagées dans le futur pour les contrôleurs, et qu'un LQI y avait déjà été implémenté, ce qui permet d'avoir un référentiel. En outre le code de série est lui aussi un référentiel intéressant.

9.4.3 Informatique

Les résultats ont été analysés sur une station de travail sun ultra 30 3D avec les logiciels MapleV reslease 2 (1991) et MATLAB/SIMULINK 5. Le langage de programmation utilisé à été le C pour le code véhicule et Java pour les simulations sur station.

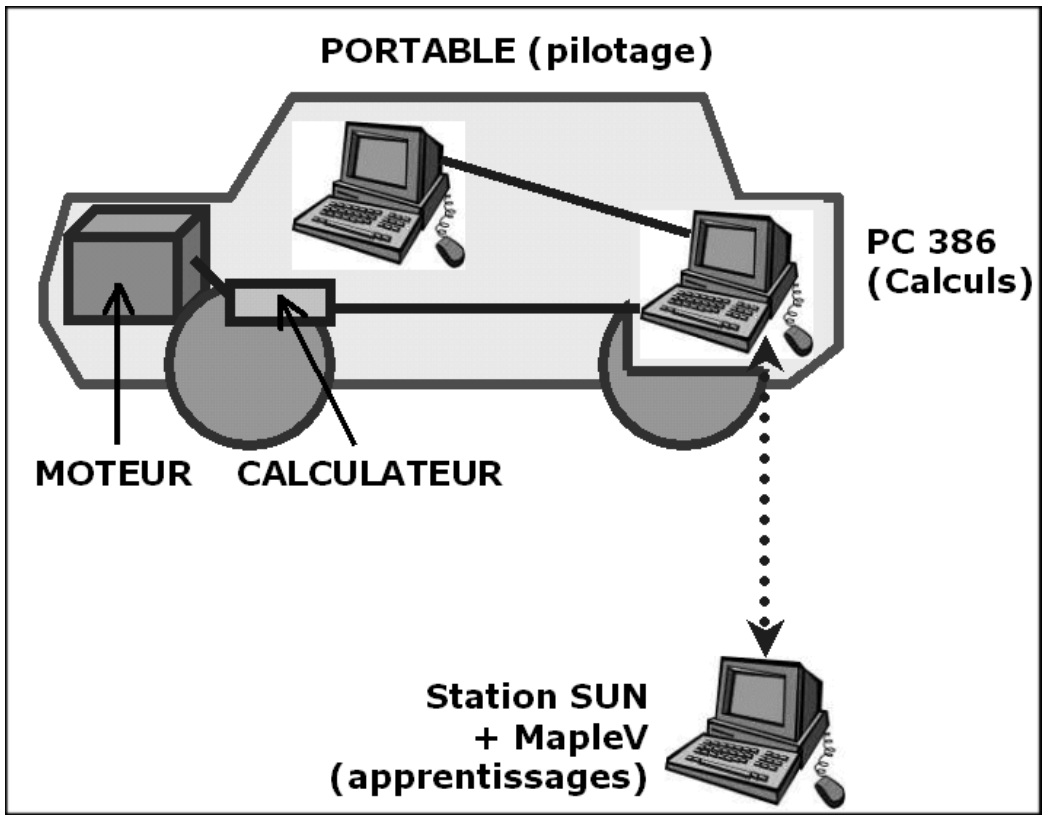


FIG. 9.13 – Installation embarquée - débarquée utilisée pour l'apprentissage du contrôle sur un véhicule de série

Chapitre 10

Mise en œuvre du contrôle par PAP sur un moteur

10.1 Modèle disponible

Ce chapitre expose les réalisations effectuées en pratique dans le cadre de ma thèse sur un véhicule prototype à la Direction de la Recherche de RENAULT.

Pour des raisons de confidentialité, certains résultats ne seront exposés que qualitativement.

Le moteur du prototype était un moteur de série sur un véhicule de série. La modélisation utilisée était celle mise au point par les ingénieurs de RENAULT dans le cadre de l'utilisation d'une "commande en couple". Il s'agissait de ramener le système dynamique que constitue le moteur au cours des combustions dans un espace de représentation le plus simple possible au moyen d'une expertise physique.

En effet, selon les experts RENAULT l'expression de l'état du moteur et des commandes qu'on lui applique en "log de couple" a un comportement "presque linéaire" au cours du temps.

Il va sans dire qu'une des complexités réside dans le fait que les modèles disponibles ne sont que des approximations des phénomènes de combustion se produisant réellement. Ceci a notamment motivé le fait que les experts aient souvent recours à un espace d'état plus grand que celui suffisant parfois d'un point de vue théorique. En augmentant ainsi le nombre de variables disponibles pour un feedback (par exemple des dérivées ou intégrales partielles de la trajectoire), ils espèrent capter la réalité des phénomènes avec plus de

précision.

10.1.1 Comportement qualitatif du moteur

Un moteur à essence fournit de l'énergie via la transformation (mécanique) en rotation d'un mouvement de translation d'un piston entraîné par l'explosion d'un mélange d'essence et d'air. Ce mouvement de rotation rencontre une résistance due à l'inertie de l'axe qu'il entraîne. Les variations d'inertie de cet axe peuvent être dues aux mouvements du véhicule (montées ou descentes de côtes par exemple) ou à des sollicitations de composants auxiliaires (comme la climatisation ou l'assistance à la conduite).

C'est la proportion et les volumes d'essence et d'air injectés dans les cylindres qui pilote l'énergie dégagée par les explosions. L'objectif d'une "**stratégie de contrôle**" est donc de déterminer dans quelles proportions et dans quels volumes il est nécessaire d'injecter air et essence dans le moteur pour qu'il fournisse l'énergie correspondant à ses sollicitations.

L'explosion de l'essence et de l'air injectés dans le cylindre fournissent certes de l'énergie au véhicule, mais est aussi responsable des polluants émis par celui-ci. La législation très stricte sur ce point demande aux constructeurs d'élaborer des stratégies les moins polluantes possibles. En outre, un des objectifs des constructeurs est de consommer le moins possible de carburant pour produire une quantité d'énergie donnée.

Les différents moyens de piloter la combustion dans un moteur essence sont donc :

- **la richesse du mélange** air - essence (notée $\mathfrak{R}$ dans la suite), qui correspond à un coefficient multiplicatif par rapport aux proportions stochiométriques théoriques du mélange air - essence. Le prototype considéré est à richesse 1. Il existe néanmoins des stratégies consistant à faire varier la richesse pendant l'injection (donc en partie pendant l'explosion) dans le cadre d'une injection directe[1] dans le cylindre.

[1]Pour les moteurs à essence, le mélange d'air et d'essence se fait traditionnellement dans une pré chambre, l'injection se faisant en amont du cylindre. Contrairement aux moteurs diesel où l'injection du carburant (qui est déjà mélangé) se fait directement dans le cylindre, nécessitant une pression d'injection au moins aussi forte que la pression dans le cylindre, la plupart du temps très élevée. C'est cette injection directe qui permet de mettre au point des stratégie d'injection "stratifiée" (i.e. à richesse variable au cours d'une même injection), globalement pauvre, mais localement suffisamment riche pour produire l'énergie requise. Il existe néanmoins certains véhicules à injection directe essence, qui permettent

Le fait d'être à richesse 1 implique principalement qu'il est interdit de faire varier le volume d'air sans faire varier le volume d'essence afin de respecter les proportions stochiométriques. Le volume d'essence est donc directement asservi au volume d'air, qui devient la principale variable de contrôle.

- le **volume d'air injecté**, commandé par l'accélérateur et une vanne additionnelle (on comprend ici pourquoi les constructeurs vont vers une commande d'accélération "by wire" —i.e. sans transmission mécanique entre la pédale d'accélération et l'arrivée d'air— : cela leur permet de déconnecter au maximum la demande du conducteur —qui veut accélérer— et l'effet immédiat —une forte injection d'air et d'essence—),
- comme le mélange n'explose pas immédiatement lorsqu'il est soumis à une forte pression (contrairement au cas des moteurs diesels), il est nécessaire d'initier l'explosion par un allumage commandé par des "bougies" qui sont dans les cylindres et permettent d'y provoquer une étincelle. A priori, le meilleur moment pour allumer la bougie est lorsque le piston est à la verticale (position du "point mort haut" : PMH). En pratique, si l'explosion a lieu exactement à ce moment, le piston va partir parfaitement à la verticale, alors que la mécanique destinée à transformer son mouvement de translation de bas en haut exige qu'il se déplace légèrement latéralement : il y aura alors cassure du vilebrequin et du moteur en général. Un paramètre, appelé **avance à l'allumage**, permet de fixer le décalage (angulaire) entre la position de PMH et l'instant d'allumage de la bougie.
  Ce paramètre a une influence sur la combustion dans le cylindre : elle en modifie le rendement.

l'élaboration de stratégies à injection stratifiée dans le cadre des moteurs essence.

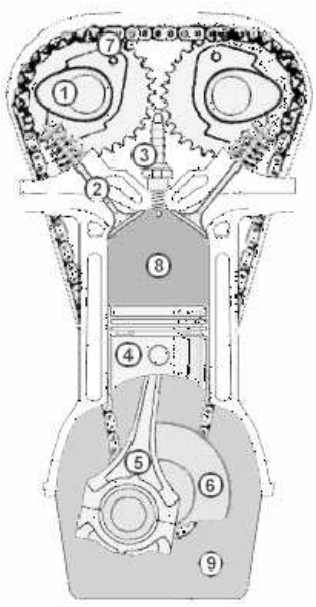


FIG. 10.1 – Diagramme simplifié d'un moteur (ici moteur à deux temps)

1. la came, montée sur l'arbre à cames, actionne la soupape au moment voulu, grâce à son profil excentrique. La soupape est maintenue en position fermée par un ressort de rappel et ne s'ouvre donc que si la came appuie dessus.
2. la soupape joue le rôle d'un clapet commandé qui va laisser passer ou non les gaz à l'admission et à l'échappement.
3. la bougie excitée par l'allumage, produit une étincelle au moment opportun qui enflamme les gaz dans le cylindre.
4. le piston est l'élément qui permet de comprimer les gaz dans le cylindre et de recueillir l'énergie de l'explosion.
5. la bielle relie le piston au vilebrequin et permet ainsi la transformation d'un cycle linéaire (le mouvement du piston) en un cycle rotatif (le mouvement du vilebrequin).
6. le vilebrequin transmet le mouvement entre le piston et le reste du moteur et relie aussi les mouvements différents cylindres.
7. la distribution est un élément composé d'une courroie, d'une chaîne ou d'un jeu de pignons qui relie le ou les arbres à came(s) au vilebrequin, synchronisant ainsi le tout.
8. la chambre de combustion est l'espace formé par le piston rendu en haut et la culasse, partie fixe qui contient les soupapes, la bougie et le ou les arbres à came(s).
9. l'huile sert à lubrifier tout ce qui bouge et diminue ainsi les pertes toujours trop importantes par friction.

10.1.2 Modèle en couple

Dans le cadre de la représentation utilisée en pratique, on considère que l'état du couple moteur se décompose en :

- une composante "**couple moyen intégral**" qui dépend de la résistance opposée au moteur, notée C^+_{MI},
- une composante "**couple du à l'arrivée d'air**", notée $C_{\mathrm{air.mot}}$,
- et une composante "**couple lent**" d'évolution assez lente, notée C_{lent}.

Ces variables d'état étant physiquement majoritairement liées par des relations multiplicatives, c'est l'évolution de leurs logarithmes qui est considérée par les experts. Les variables d'état retenues par sont donc :

$$(X_1, X_2, X_3) = (\log(C_{\mathrm{air.mot}}), \log(C^+_{\mathrm{MI}}), \log(C_{\mathrm{lent}}))$$

Les variables de contrôle correspondantes sont :

- une **commande de couple d'air**, notée $C_{\mathrm{air.cmd}}$, que l'on peut faire correspondre directement à un débit d'air que l'on contrôle grâce à l'accélérateur couplé à une vanne d'arrivée d'air additionnel nommée "papillon",
- un **rendement de l'avance à l'allumage**, noté η_{av}, qui correspond à l'efficacité de l'allumage de la bougie en avance de phase par rapport à la position "point mort haut" du piston.

Les commandes considérées sont aussi les logarithmes de ces variables :

$$(U_1, U_2) = (\log(C_{\mathrm{air.cmd}}), \log(\eta_{\mathrm{av}}))$$

Le modèle de comportement du moteur est quant à lui (σ est l'opérateur d'avance) :

$$(10.1)\quad \begin{cases} \sigma X_1 &= \log((1+\varphi_2)(1-\varphi_1)\cdot\exp(U_1) + \varphi_2\cdot\varphi_1\cdot\exp(X_1)) \\ \sigma X_2 &= X_1 + \log(\eta_{\mathfrak{R}}) + \log(U_2) \\ \sigma X_3 &= X_1 + \log(\eta_{\mathfrak{R}}) + \log(\eta^{(0)}_{\mathrm{av}}) \end{cases}$$

Les paramètres de ce modèle sont φ_1 et φ_2 qui prennent les valeurs suivantes :

$$(10.2)\quad \begin{cases} \varphi_1 &= 1 \\ \varphi_2 &= \gamma\dfrac{V_{mot}\eta_0}{N_{cyl}V_{col}} \end{cases}$$

où η_0 est une constante qui correspond à un rendement du remplissage qui dépend du régime et de la pression dans le collecteur (en effet, ces deux

variables engendrent un phénomène "d'aspiration" dans le cylindre à l'ouverture et des soupapes —à l'admission comme à l'échappement—). Les experts considèrent que pendant un petit intervalle de temps cette valable est pratiquement constante.

D'autre part la variable $\eta_{\mathfrak{R}}$ correspond aux variations du rendement de la combustion suivant la richesse. On se considère à richesse constante (i.e. $\mathfrak{R} \equiv 1$) : elle est donc constante. En pratique les conditions ne sont jamais exactement stochiométriques ; en effet le volume d'essence injecté se colle en partie aux parois (phénomène de "mouillage de parois" bien connu des experts qui est en partie contourné en modélisant les variations de la fine couche d'essence collée aux parois de la pré chambre et du conduit d'admission en fonction des volumes d'essence injectés, du régime et de la pression dans le collecteur).

Constantes utilisées dans le cadre de la modélisation.

variable	valeur	commentaire
R	8.314	Constante des gaz parfaits (J/K/mol)
Γ	1.4	C_p/C_v pour l'air assimilé à un gaz diatomique
M_a	0.0288	Masse molaire de l'air (kg/mol)
T^0	273	0 degrés celsius en Kelvin
V_{col}	4.14	Volume du collecteur d'admission (l)
V_{mot}	1998	Cylindrée Moteur (cm^3)
J_{mot}	0.117	Inertie moteur en ($kg.m^2$)
N_{cyl}	4	Nombre de cylindres
N_{tps}	4	Nombre de temps moteur par cycle
$d\alpha/dt$	$60N_{\text{tps}}/(2N_{\text{cyl}})$	Facteur de conversion angle vers temps
K_{inj}	133	Débit statique de l'injecteur (g/min)
T_{inj}	418	Temps mort de l'injecteur (ms)

Ce modèle a été retranscrit sous MapleV afin d'être utilisé de façon exhaustive par la suite.

Les figures 10.2 et 10.3 montrent une évolution des variables (X_1, X_2, X_3) et (U_1, U_2) pour un point de départ et une des stratégies de contrôle utilisée par les experts RENAULT. Les figures 10.4 et 10.5 montrent une évolution des variables (X_1, X_2, X_3) et (U_1, U_2) pour un point de départ et une perturbation aléatoire de la stratégie de contrôle précédente.

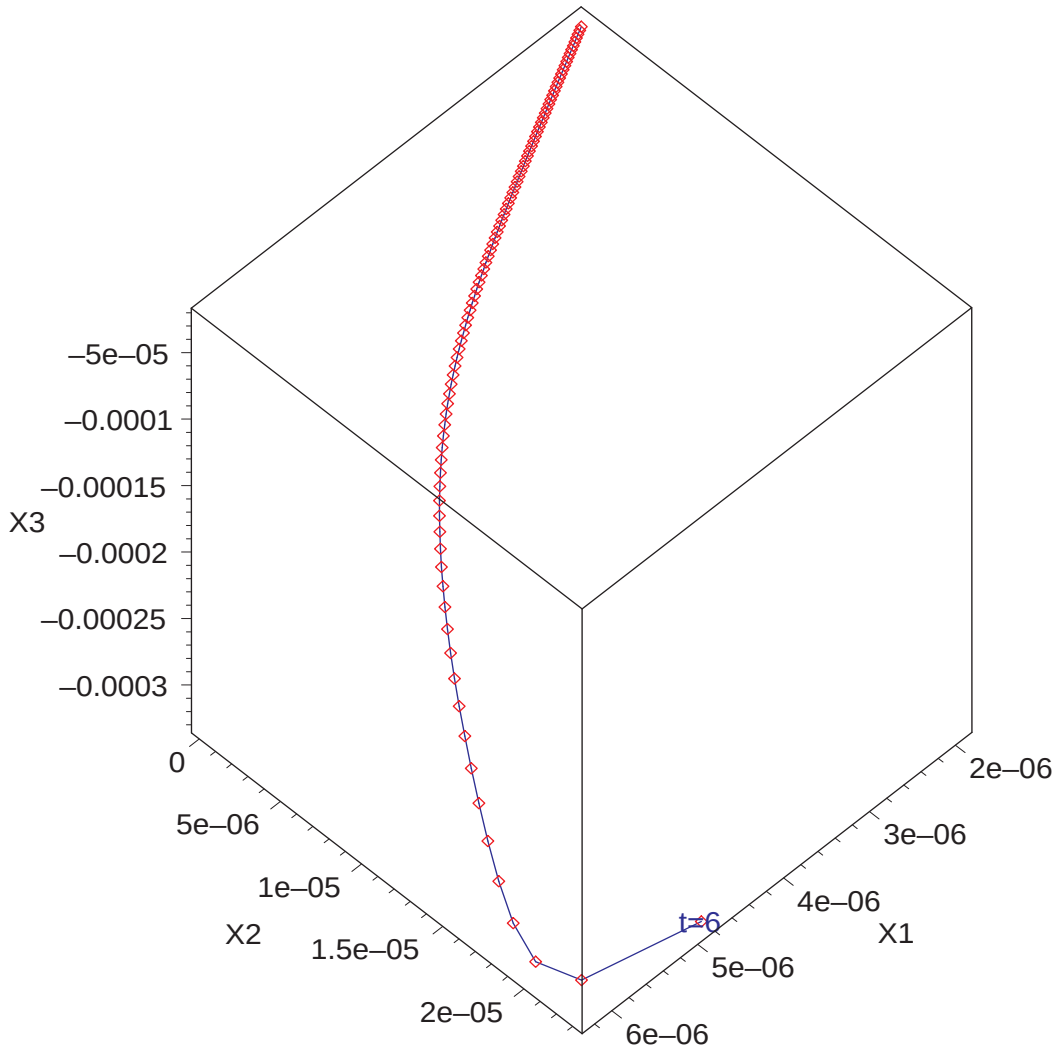


FIG. 10.2 – Evolution des variables d'état X_1, X_2 et X_3 pour une stratégie de contrôle des experts à partir de la 6ème itération.

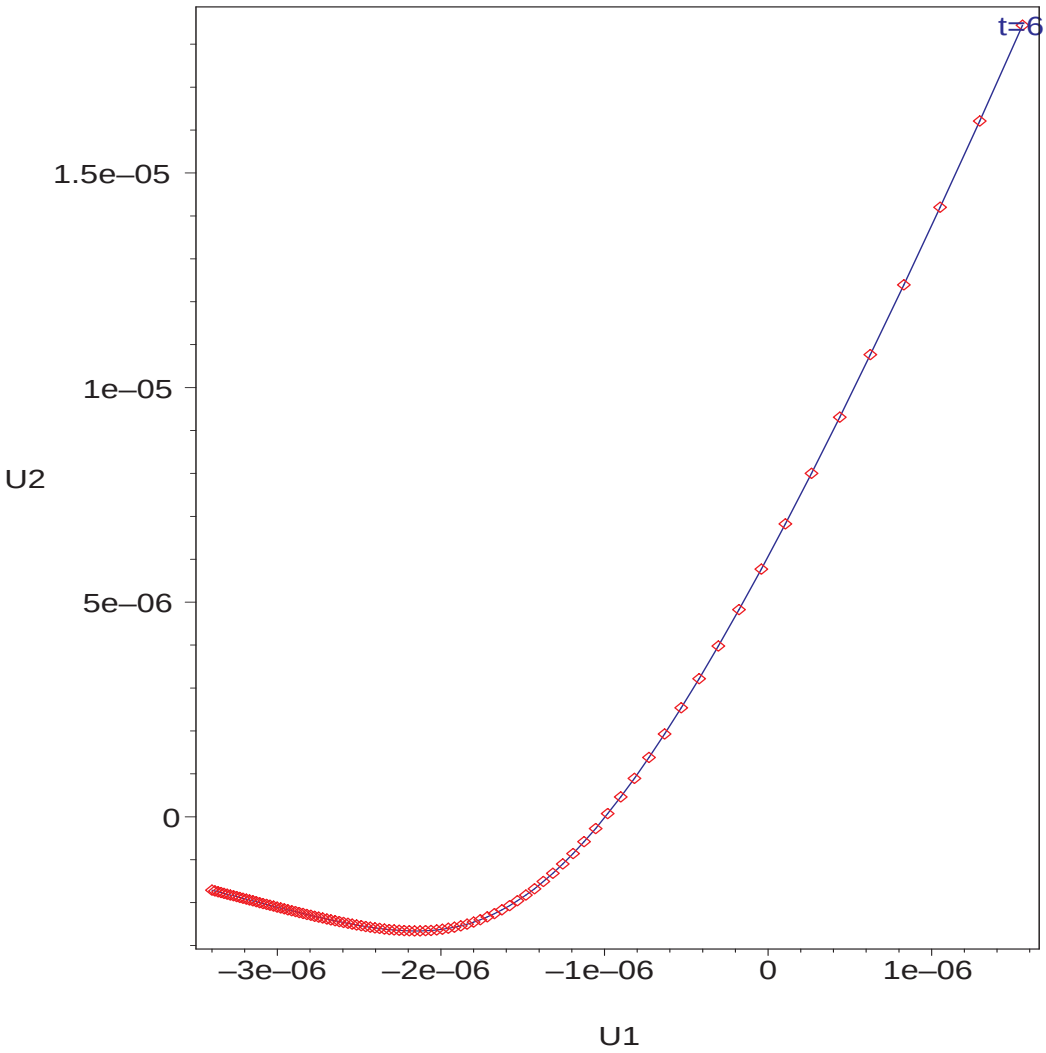


FIG. 10.3 – Evolution des variables de contrôle U_1 et U_2 pour une stratégie de contrôle des experts à partir de la 6ème itération.

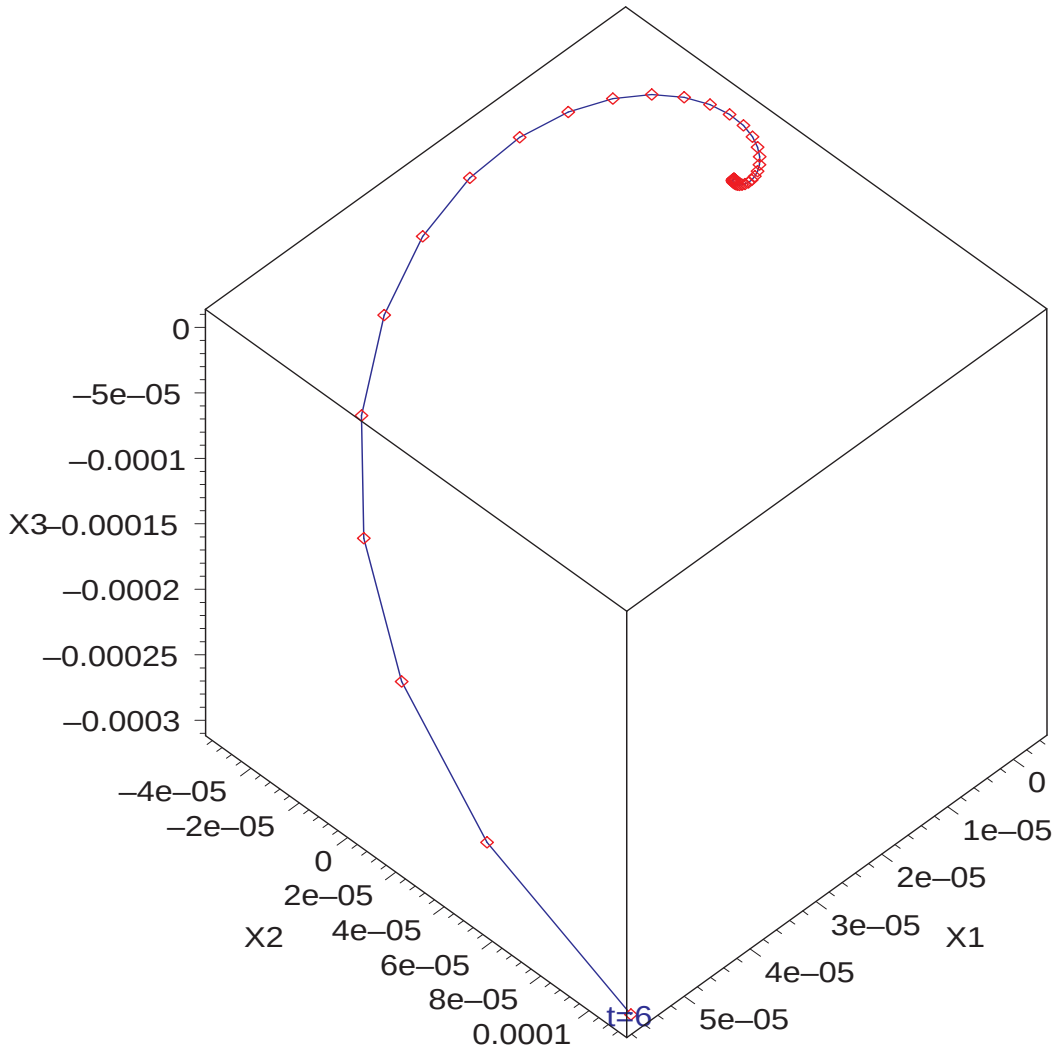


FIG. 10.4 – Evolution des variables d'état X_1, X_2 et X_3 pour une stratégie de contrôle légèrement aléatoire à partir de la 6ème itération.

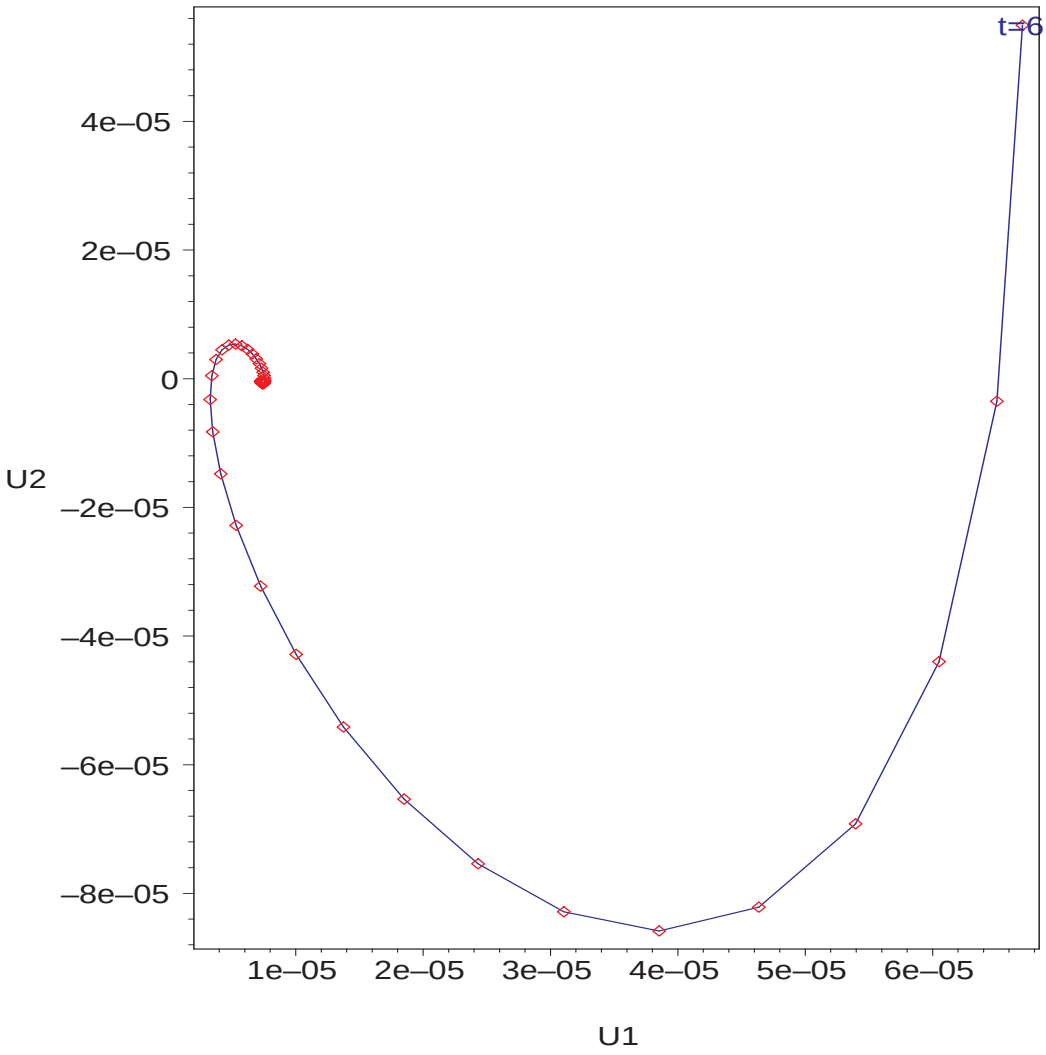


FIG. 10.5 – Evolution des variables de contrôle U_1 et U_2 pour une stratégie de contrôle légèrement aléatoire à partir de la 6ème itération.

10.1.3 Contrôle LQI

Le type de contrôle mis en place par les experts avant le début de ce travail était un contrôle LQI (pour Linear Quadratic Integral). Il s'agit d'un contrôle optimal (minimisant l'intégrale au cours du temps d'un coût quadratique) "*à la*" PI (Proportionnel Intégral).

Le contrôleur PAP a utilisé les variables disponibles dans ce cadre, un vecteur d'état de dimension huit constitué à chaque instant de (les variables étoilées sont des consignes qui varient lentement) :

$$X = \left(X_1(t), X_2(t), X_3(t), \int_{\tau \leq t} X_2(\tau) d\tau, \int_{\tau \leq t} X_3(\tau) d\tau, X_2^*(t), X_3^*(t), X_1^*(t) \right)$$

Pour considérer ce vecteur d'état, il est nécessaire de construire un sur-système de (10.1) en introduisant la dynamique linéaire de $Y_i = \int_{\tau \leq t} X_i(\tau) d\tau$ ainsi :

$$\sigma\, Y_i = Y_i + X_i\, \delta_t$$

et celle, triviale, des X_i^* qui est constante à petite échelle de temps :

$$\sigma\, X_i^* = X_i^*$$

Il est donc assez simple d'écrire le sur système correspondant à la stratégie de régulation LQI.

Le contrôle sera émulé par un PAP noté Ψ_W :

$$U = \Psi_W(X)$$

10.1.4 Version linéarisée du modèle

Les experts RENAULT utilisent une version linéarisée du modèle (10.1) autour du ralenti qui leur permet de calculer (en choisissant des matrices de coût ad hoc) une matrice de Riccati pour effectuer un contrôle en boucle.

Ce contrôleur a été exposé plus haut en section 2.3.2.

Si on prend en compte les non linéarités du système (10.1) et qu'on utilise la matrice de contrôle K retenue par les experts RENAULT : on obtient un contrôle instable, comme le montre la figure 10.6 obtenue à partir de simulations de contrôle du modèle non linéaire. On voit que les trajectoires divergent en moyenne (ainsi que en min et en max, si tant est que cela ait un sens) à partir de la 50ème combustion au moins.

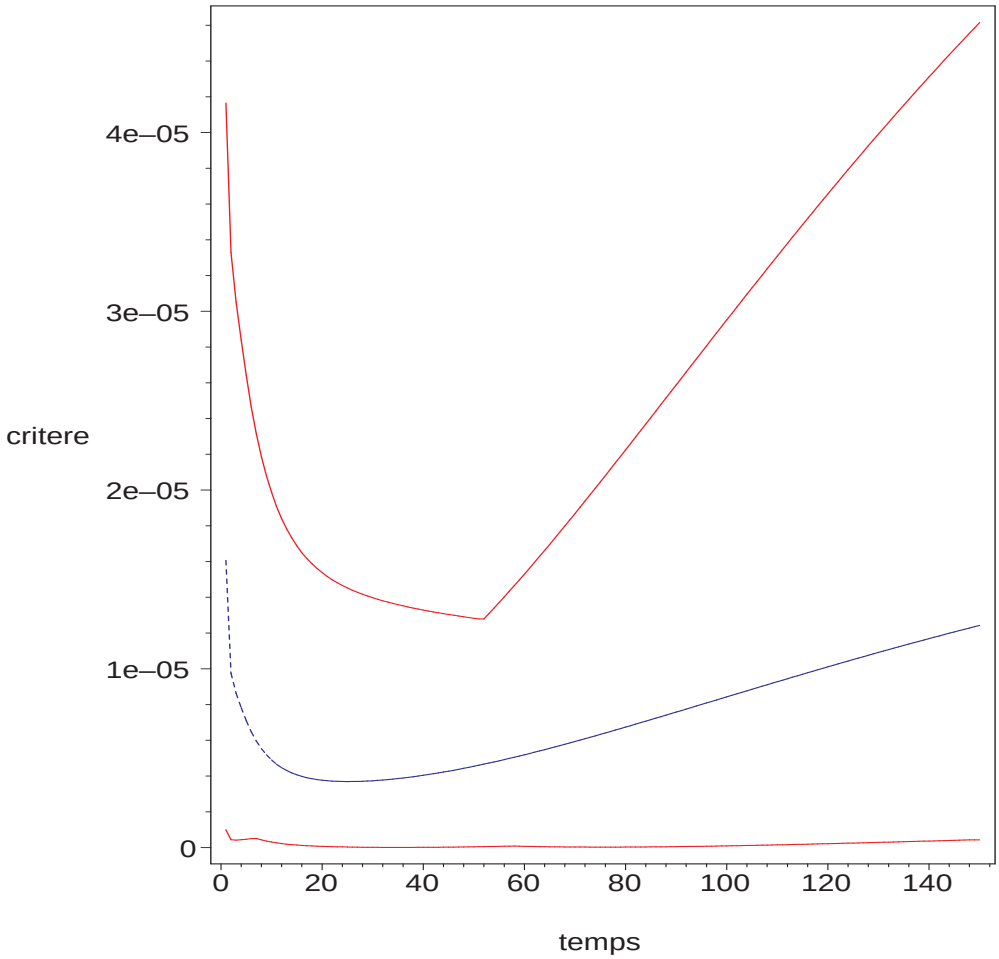


FIG. 10.6 – Cette illustration montre l'évolution de la moyenne, du minimum et du maximum du critère à chaque instant (150 instants en abscisse) calculés sur 150 trajectoires.

Cela confirme la conviction des experts selon laquelle la contrôle autour du ralenti est très délicat, même sans sollicitations énergétiques externes (comme la climatisation).

Le recours à un autre type de contrôleur intéressait donc les experts de RENAULT. C'est dans cette situation qu'un véhicule prototype a été assemblé autour du projet de contrôle par réseaux de neurones : le véhicule a été équipé (en effet certains aménagements sont nécessaires afin de transformer un véhicule de série en prototype), et le système de commande en couple a été mise en place.

Prototype. Les commandes d'injection et d'avance à l'allumage étaient directement transmises au calculateur par un ordinateur de type PC positionné dans le coffre du véhicule, qui obtenait à chaque combustion les informations nécessaires aux calculs des consignes à appliquer. Un système "de sécurité" appliquait automatiquement les commandes de la combustion précédente dans le cas où le transfert d'information n'était pas assez rapide (à 2000 tours moteur par minute, les calculs doivent être effectués à plus de 30 Hz.

D'autre part, le système mis en place permettait de piloter le système à partir d'un PC portable au niveau du siège du conducteur, et donc de basculer en mode "classique", "LQI", ou "réseau de neurones" et de commander la sauvegarde des données sur disquette (dans le coffre du véhicule). Il y avait en fait plusieurs "mode réseau de neurones" correspondants à des jeux de poids différents, et à des fonctions d'activation affines par morceaux ou sigmoïdes.

10.2 Initialisation du PAP

Les résultats obtenus avec le contrôle de Riccati sur système linéarisé impliquait qu'il convenait de considérer plusieurs points de fonctionnement pour initialiser le PAP.

J'ai retenu trois points de fonctionnement (leur description exacte est confidentielle) :

- un au niveau d'une situation de ralenti classique,
- un autre juste avant un calage (faible régime et moyenne charge),
- un au dessus du ralenti (régime plus fort).

Le package MapleV réalisé m'a permis :

- d'obtenir des développements limités du système (10.1) en tout point de l'espace d'état,

– de calculer les contrôles locaux en résolvant les équations de Riccati associées (le package faisait alors appel aux librairies MATLAB de résolution de ce type d'équations),
– d'initialiser un PAP à partir des matrice locales obtenues.

Plusieurs PAP ont été initialisés pour des triplets de points de fonctionnements légèrement différents.

10.3 Apprentissage "en ligne" des PAP

10.3.1 Première version "hors ligne"

Les PAP initialisés ont été entraînés sur le simulateur moteur développé sous MapleV dans le cadre de la thèse, puis utilisés sur le véhicule. Un écueil est apparu à cet instant : plus le PAP était adapté au simulateur (et plus il le contrôlait bien) et moins il était adapté au moteur réel.

Cela impliquait que le modèle moteur fourni par les experts était éloigné du comportement réel de moteur. Cela peut avoir deux raisons :
– d'une part le modèle détaillé plus haut fait de nombreuses hypothèses (sans doute trop réductrices) sur le déroulement de la combustion,
– d'autre part les premiers essais réalisés avec le contrôleur linéaire, puis avec le PAP mais en intégrant une erreur de calcul, ont sans doute fatigué le moteur en l'éloignant du comportement d'un moteur neuf, sans doute plus proche du modèle.

Toujours est-il qu'à l'issue de ces premiers essais, je décidais de transférer au maximum l'apprentissage sur le véhicule.

10.3.2 Apprentissage en ligne

Pour l'apprentissage, un jeu de poids donné est utilisé plusieurs fois pour contrôler le véhicule, le comportement de toutes les variables en jeu est enregistré. Le tout est ensuite rejoué sur une station sun en utilisant MapleV afin de rétropropager les erreurs. On voit bien que ce mécanisme mélange une rétropropagation selon un modèle vraisemblablement distant de la réalité avec l'utilisation des activations et des erreurs réellement enregistrées sur véhicule (figure 10.7).

Cet **apprentissage "mixte"** me fut inspiré par les résultats de Johansson et Rantzer [Johansson & Rantzer, 1997] [Rantzer & Johansson, 1997] [Johansson & Rantzer, 1996], ainsi que par un article de Leung, Lam et Tam [Leung *et al.* , 1998] et par mes propres travaux autour de la preuve possible de résultats de stabilité sur le

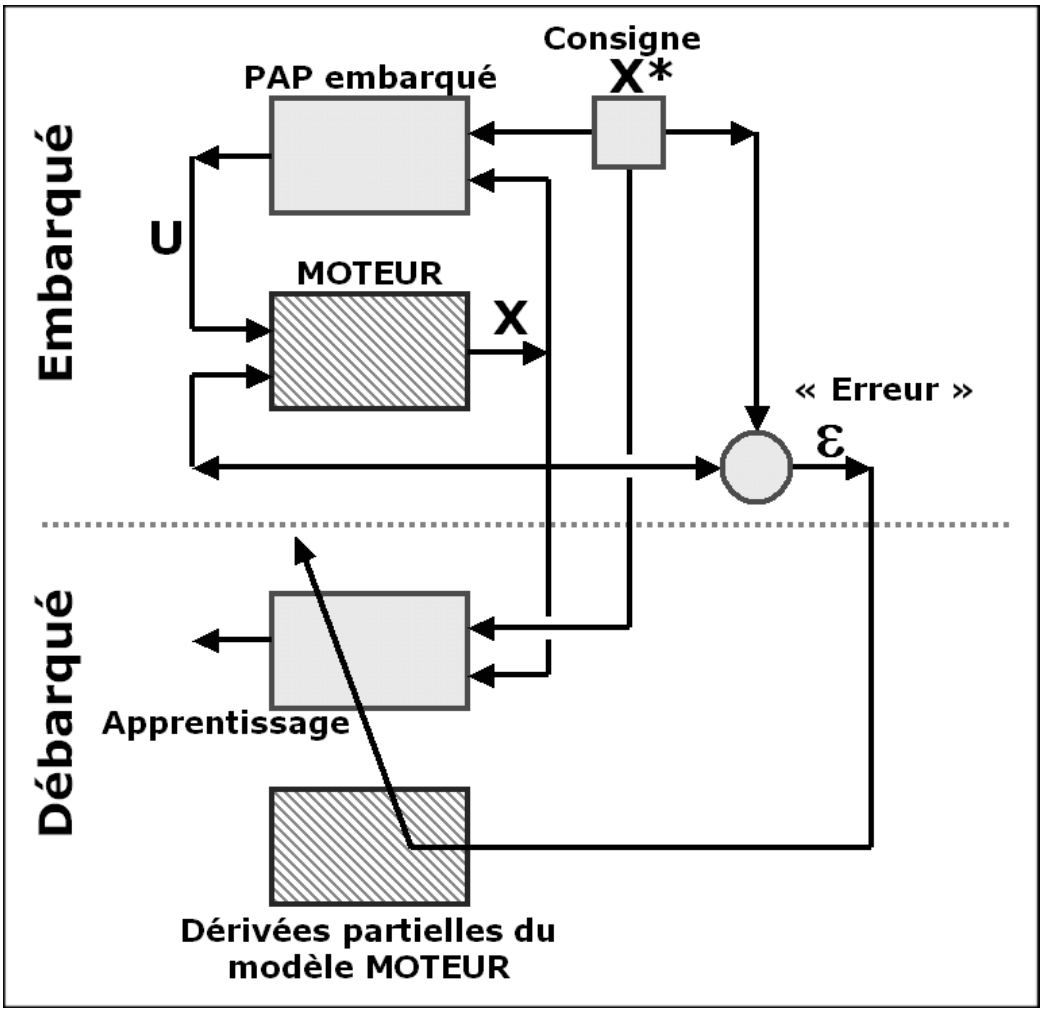


FIG. 10.7 – Diagramme du principe d'apprentissage finalement utilisé

contrôle par PAP.

En effet, il faut noter que les PAP arrivaient assez facilement à contrôler le simulateur et que les ingénieurs contrôlent ce genre de moteur depuis longtemps avec des stratégies assez frustres (même si la consommation et la pollution ne sont pas toujours optimisées) ; ma conviction fût donc que le système réel possède certains attracteurs robustes autour du ralenti.

Et si cette attraction est suffisamment forte, un léger écart avec le modèle peut être compensée (c'est ce qu'on peut tirer par exemple d'une re lecture de [Leung *et al.* , 1998]) pourvu qu'on rétro propage l'erreur réelle en utilisant les activations réelles du PAP au même instant.

En effet, si on écrit le gradient du modèle comme une légère perturbation du gradient du moteur, alors si la dynamique de l'évolution des poids du PAP comporte un attracteur suffisamment puissant, il y aura tout de même convergence [Freidlin & Ventcel, 1998].

Nous nous plaçons donc dans une situation où les données $X(t)$ (variables d'état) qui sont mesurées sur le système réel (moteur contrôlé par un PAP). On a donc un système du type :

$$\left\{ \begin{array}{lcll} U(t) & = & \Psi_W(X(t)) & \text{PAP embarqué} \\ X(t+dt) & = & F(X(t),U(t)) & \text{Moteur réel} \end{array} \right. \tag{10.3}$$

et l'apprentissage se fait à partir de ces valeurs enregistrées mais en remplaçant F (le moteur) par son modèle $\hat{F}$. Le moteur intervient principalement par ses dérivées partielles pendant l'apprentissage. On va donc remplacer :

$$\delta F_w(x) = \partial_X F|_x + \partial_U F|_x \cdot \partial_w \Psi_W|_x$$

par :

$$\delta \hat{F}_w(x) = \partial_X \hat{F}\Big|_x + \partial_U \hat{F}\Big|_x \cdot \partial_w \Psi_W|_x$$

L'erreur commise, qui est en $\delta F_w(x) - \delta \hat{F}_w(x)$ va surtout avoir de mauvaises conséquences si les directions des dérivées partielles de F et de $\hat{F}$ sont différentes. Dans le cas où elles sont très proches, on peut espérer réduire les erreurs commises malgré cette substitution.

La mise en place de ce type d'apprentissage permit deux ou trois pas de descente de gradient. Ici un "pas de descente" n consiste en :

1. plusieurs utilisations du PAP avec un jeu de poids $W(n)$ sur le véhicule réel et enregistrement de comportement du PAP (activations) et du moteur,

2. rétropropagation des erreurs commises en utilisant les grandeurs enregistrées et le modèle moteur pour les calculs de gradient,
3. mise à jour des poids : $W(n) \rightarrow W(n+1)$.

La figure 10.8 montre les résultats obtenus : en abscisse figure un numéro d'essai, en ordonnée le critère de performance (sans unité, plus la valeur est faible et plus le contrôle est performant autour de ce point de fonctionnement) : les trois croix sont le minimum, la moyenne et le maximum obtenus. Les essais 1 à 7 sont identiques aux essais 9 à 15. La seule différence est que pour les premiers le contrôleur est un PAP et pour les autres il s'agit du contrôleur de référence des experts. L'essai 8 réalisé avec un PAP n'a pas de correspondance pour le contrôleur des experts car il a été nécessaire d'arrêter le moteur devenu dangereusement instable sous ce contrôle.

Un essai consiste en l'utilisation répétée d'un moteur (plus de 40 fois) suivant un protocole précis en terme de sollicitations en régime - charge. D'une répétition à l'autre, le protocole était qualitativement identique car le contrôle était manuel (allumage ou non de la climatisation à tel instant, accélération à tel autre, etc).

Les moyennes des performances moyennes sont comparables (si on excepte les essais utilisant le contrôleur LQI ayant mis le moteur dans une situation instable), alors que la dispersion est bien plus faible pour le contrôleur PAP.

Il est vraisemblable que si il avait été possible de réaliser plus de pas de rétropropagation, les performances moyennes du PAP auraient encore baissé.

10.4 Vérification de stabilité et de robustesse

Une fois un PAP mis au point, le package MapleV était utilisé dans le cadre d'une :

- Vérification markovienne :
  1. pour déduire des valeurs des poids et du modèle moteur une formulation markovienne d'une étape de contrôle,
  2. pour vérifier que la chaîne de Markov associée convergeait bien vers des états sans risque pour le moteur.
- Vérification de Lyapunov :
  1. pour déduire des valeurs des poids et du modèle moteur les inéquations linéaires correspondant à la définition compatible avec

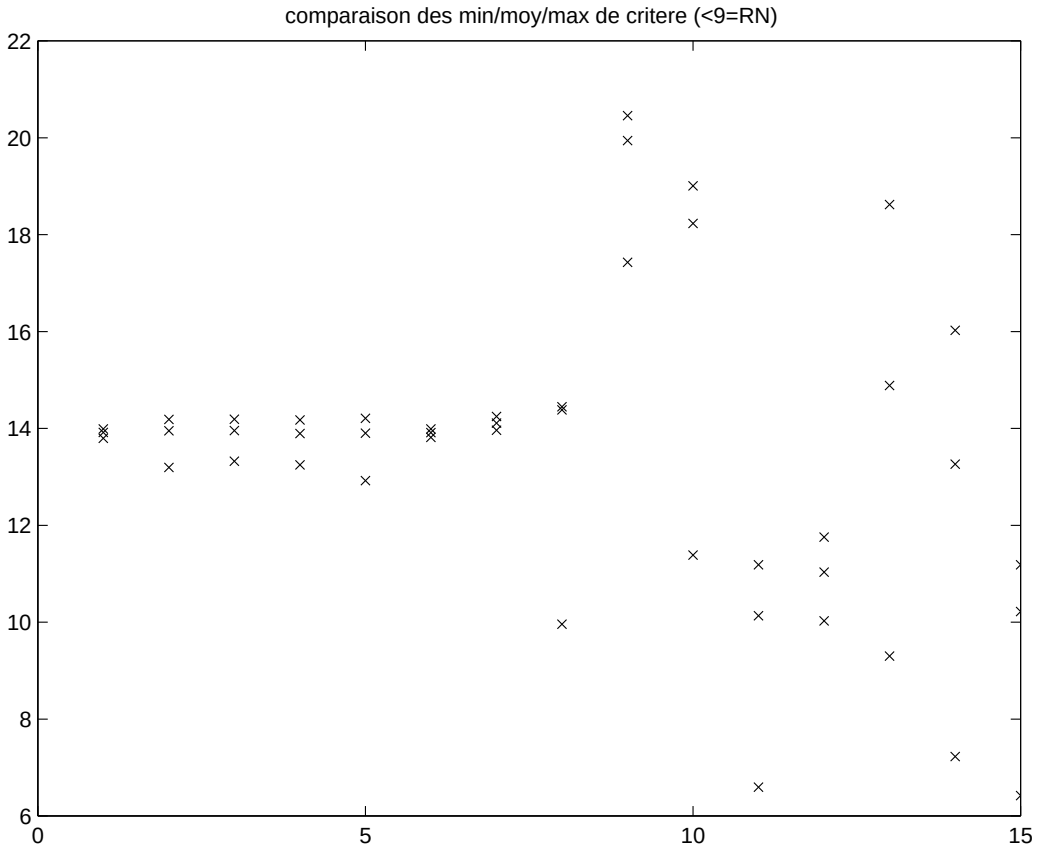


FIG. 10.8 – Performances du contrôle par PAP : comparaison avec des contrôles classiques. En abscisse : un numéro d'essai, en ordonnée le critère de performance (sans unité, plus la valeur est faible et plus le contrôle est performant autour de ce point de fonctionnement) : les trois croix sont le minimum, la moyenne et le maximum obtenu. Les essais 1 à 7 sont identiques aux essais 9 à 15. La seule différence est que pour les premiers le contrôleur est un PAP et pour les autres il s'agit du contrôleur de référence des experts. L'essai 8 réalisé avec un PAP n'a pas de correspondance pour le contrôleur des experts car il a été nécessaire d'arrêter le moteur devenu dangereusement instable sous l'effet de ce contrôle. On remarque que le contrôleur PAP est plus stable et plus performant que le contrôleur des experts.

le chapitre 8.2 des cellules polyhédrales engendrées par le couplage du PAP et du moteur,

2. pour vérifier l'existence d'une matrice de Lyapunov garantissant la stabilité du contrôle de ce couplage.

A chaque itération, seuls les jeux de poids vérifiant les deux points ont été conservés.

Ceci engendrait un pilotage plutôt "manuel" des apprentissages :

1. utilisation de tous les essais (numérotés 1 à 8 sur la figure 10.8) pour passer d'un jeu de poids $W(n)$ à un jeu de poids $W(n,1)$,

2. vérifications (Markov + Lyapunov) à partir d'un jeu de poids $W(n,k)$:
   - si les deux sont faites : utilisation de $W(n,k) : W(n+1) = W(n,k)$
   - sinon : élimination des enregistrements d'un des essais utilisés pour calculer $W(n,k)$ (celui pour lequel la variance des performances était la plus grande) : calcul d'un nouveau jeu de poids $W(n,k+1)$ et retour au point 2.

Dans le cadre pratique du contrôle du prototype, ces étapes n'ont pas été automatisées.

10.5 Bilan

Les résultats ont été considérés comme très satisfaisants par RENAULT. En effet, le contrôle non linéaire d'un moteur de série par un réseau de neurones a été réalisé. Le contrôle obtenu était plus stable et au moins aussi performant que celui des experts. Le caractéristiques principales de cette méthodologie par RENAULT furent :

- **Nécessité de disposer d'un modèle du système à contrôler.**
  Ce point n'est pas un véritable obstacle puisque la faisabilité de l'émulation d'un moteur par un perceptron a été démontrée lors de l'expérience *Mack Trucks* (cf 9.3).
- **Garantie de performances par rapport à l'état de l'art.**
  La correspondance mise au point entre les fonctions continues affines par morceaux et les PAP permet de garantir la possibilité de construire un PAP émulant localement les meilleurs contrôleurs linéaires mis au point par les experts.

- **Facilité à embarquer.**
  Le correspondance entre les PAP et les fonctions affines par morceaux permet de traduire un PAP aux poids fixés en une collections de fonctions affines à appliquer sur des zones définies par des inégalités linéaires. L'industrie automobile dispose déjà dans la plupart de ses calculateurs embarqués de fonctions *hardware* émulant cette fonctionnalité (nommées *cartographies*). Il est donc envisageable d'embarqué un PAP aux poids fixés à faible coût.
- **Garanties de stabilité.**
  Un contrôle par PAP offre des garanties de stabilité non seulement théoriques mais vérifiables empiriquement. Bien sûr, le package MapleV permettant de le vérifier implémenté dans le cadre de ces travaux doit être amélioré (de nombreuses opérations sont pilotées par l'utilisateur, elles peuvent être automatisées).

D'autre part, j'espère que les nombreux échanges lors des revues de projets ou lors de réunions travail (par exemple avec l'équipe mettant au point les contrôleurs des véhicules hybrides explosion - électrique) avec les autres chercheurs de RENAULT ont mis en avant l'avantage de pouvoir se ramener à des chaînes de Markov. En effet cela permet de transformer le contrôle moteur (qui est habituellement considéré comme une "boîte noire") en une machine à états dont il est possible d'étudier l'influence sur la mise au point du véhicule dans sa globalité.

Bibliographie

[Abbas, 1993] Abbas, Hassane. 1993. *Contribution à l'étude de la réduction formelle des systèmes différentiels méromorphes linéaires.* Ph.D. thesis, Institut National Polytechnique de Grenoble.

[Afanasev *et al.* , 1996] Afanasev, V.N., Kolmanovskii, V.B., & Nosov, V.R. 1996. *Mathematical Theory of control system design.* Kluwer.

[Agarual, 1992] Agarual, R. P. 1992. *Difference equations and inequalities.* Marcel Dekkar.

[Agarwal, 1997] Agarwal, M. 1997. A Systematic Classification of Neural-Network-Based Control. *IEEE conf. on control*, pp75–93.

[Airy, 1840] Airy, G. B. 1840. On the regulator of the clock-work for effecting uniform movement of equatoreals. *Memoirs of the Royal Astronomical Society*, **11**, 249– 267.

[Albertini & Sontag, 1992] Albertini, F., & Sontag, E. D. 1992. *State observability in recurrent neural networks.* Tech. rept. 92-07rev. SYCON.

[Albertini & Sontag, 1994] Albertini, F., & Sontag, E. D. 1994. *Uniqueness of weights for recurrent nets.* Tech. rept. Rutgers university, dept of mathematics.

[Amari., 1998] Amari., Shun-Ichi. 1998. Natural gradient works efficiently in learning. *Neural Computation*, **10**, 251–276.

[Aranda-Bricaire & Moog, 1995] Aranda-Bricaire, E., & Moog, Ch. 1995. A linear algebraic framework for dynamics feedback linearization. *IEEE trans. on Auto. Control*, **40**, 127–132.

[Aranda-bricaire *et al.* , 1994] Aranda-bricaire, E., Moog, C. H., & Pomet, J. B. 1994. *Infinitesimal brunovsky form for nonlinear systems with applications to dynamic linearization.* Tech. rept. INRIA.

[Arimoto, 1966] Arimoto, S. 1966. Linear, stationnary, optimal feedback control systems. *information and control*, **9**, 79–93.

[Arnold, 1974] Arnold, V. 1974. *Equations différentielles ordinaires.* MIR.

[Arnold, 1976] Arnold, V. 1976. *Métodes Mathématiques de la mécanique classique*. MIR.

[Arnold, 1980] Arnold, V. 1980. *Chapitres supplémentaires à la théorie des équations différentielles ordinaires*. MIR. Chap. formes normales.

[Ashhab *et al.* , 1998] Ashhab, M. S., Sefanopoulou, A. G., Cook, J. A., & Levin, M. B. 1998. Camless engine control for a robust unthrottled operation. *SAE technical paper series*, 159–169.

[Atkinson *et al.* , 1998] Atkinson, C, Long, T, & Hanzevack, E.L. 1998. Virtual Sensing : a Neural Network-Based Intelligent Performance and Emissions Prediction System for On-Board Diagnostics and Engine Control. *In :* SAE (ed), *International Congress and Exposition, February 1998, Detroit, MI, USA, Session : Electronic Engine Controls*, vol. PT-73.

[Aubin, 1996] Aubin, Jean-Pierre. 1996. *Neural networks and qualitative physics*. Cambridge University Press.

[Aubin *et al.* , 1980] Aubin, J.P., Bensoussan, A., & Ekeland, Ivar. 1980. *Mathematical technics of optimal control and decision*. Birkauser.

[Azencott & Berthelor, 1965] Azencott, Robert, & Berthelor, P. 1965. *Géométrie algébrique élémentaire*. ENS-IHP.

[Azzo, 1988] Azzo, d'. 1988. *Linear Control System Analysis and Design, Conventional and Modern*. McGraw Hill.

[Bacciotti, 1992] Bacciotti, A. 1992. *Local Stability of non linear control systems*. World Scientific.

[Bahrami, 1993] Bahrami, M. 1993. *RBF networks as direct PID controller of nonlinear plants*. Tech. rept. university of new south wales, school of electrical engineering.

[Balachandran *et al.* , 1996] Balachandran, K., Dauer, J. P., & Balasubramaniam, P. 1996. Controllability of semilinear integrodifferential systems in banach spaces. *journal of mathematical systems, estimation and control*, **6**(4), 1–10.

[Balde & Jouan, 1998] Balde, M., & Jouan, P. 1998. Genericity of observability of control affine systems. *control, optimization and calculus of variations*, **3**, 345–359.

[Barbut, 1970] Barbut, M. 1970. *Ordre et classification : algèbre combinatoire*. Hachette université.

[Bart, 1973] Bart, H. 1973. *Meromorphic operator valued functions*. Mathematisches Centrum.

[Barto & Anandan, 1985] Barto, A. G., & Anandan, P. 1985. Pattern recognizing stochastic learning automata. *IEEE trans on systems, man and cybernetics*, **15**, 360–375.

[Bateman, 1945] Bateman, H. 1945. The control of an elastic fluid. *Bulletin of the AMS*, **51**.

[Bellman, 1961] Bellman, R. 1961. *Adaptive control processes : a guided tour*. Princeton university press.

[Benaïm & Hirsch, 1999] Benaïm, Michel, & Hirsch, Morris W. 1999. Stochastic approximation algorithms with constant step size whose average is cooperative. *Ann. Appl. Probab.*, **9**(1), 216–241.

[Benveniste *et al.* , 1987] Benveniste, A., Metivier, M., & P., Priouret. 1987. *Algorithmes adaptatifs et approximations stochastiques*. Masson.

[Bianco & Sadek, 1998] Bianco, S. A., & Sadek, I. S. 1998. Optimal control of a class of time-delayed distributed systems by orthogonal functions. *J. Franklin Inst.*, **335B**, 1477–1942.

[Bissell, 1989] Bissell, C.C. 1989. Stodola, Hurwitz and the genesis of the stability criterion. *Internat. J. Control*, **50**, 2313–2332.

[Bittani *et al.* , 1989] Bittani, S., Laub, A. J., & Willems, J. C. 1989. *The Riccati equation*. Springer Verlag. Chap. Count Riccati and the early days of the Riccati equation, by Bittani, S., pages 1–10.

[Blanchini, 1995] Blanchini, F. 1995. Nonquadratic Lyapunov functions for robust control. *Automatica*, **31**(3), 451–461.

[Blanchini & Miani, 1994] Blanchini, F., & Miani, S. 1994. *Robust control via variable structure*. Springer. Chap. Piecewise-linear functions in robust control, pages 213–243.

[Blanchini & Miani, 1996] Blanchini, F., & Miani, S. 1996. A new class of universal Lyapunov functions for the control of uncertain linear systems. *Pages 1027–1032 of : Proc of the 30th conf on Decision and Control*.

[Blumer, 1989] Blumer, A. 1989. Learnability and the VC dimension. *Journal of the ACM*, **36**, 929–965.

[Bonnard, 1981] Bonnard, B. 1981. Contrôlabilité des systèmes non linéaires. *C. R. Acad. Sc. Paris*, **292**, 535–537.

[Bourbaki, 1958] Bourbaki, N. 1958. *Eléments de mathématiques, Algèbre, Chap 3, Algèbre multilinéaire*. Hermann.

[Bourbaki, 1967] Bourbaki, N. 1967. *Variétés différentielles et analytiques, fascicule de résultats 1-7*. Hermann.

[Bourbaki, 1971] Bourbaki, N. 1971. *Variétés différentielles et analytiques, fascicule de résultats 8-15*. Hermann.

[Boyd *et al.* , 1993] Boyd, S, Balakrishnan, V., Feron, E., & El Ghaoui, L. 1993. Control system analysis and synthesis via linear matrix inequalities. *Pages 2147–2154 of : ACC Proc.*

[Boyd *et al.* , 1994] Boyd, S, Balakrishnan, V., Feron, E., & El Ghaoui, L". 1994. *Linear matrix inequalities in system and control theory*. Society for Industrial and Applied Mathematics. SIAM.

[Brjuno, 1989] Brjuno, A.D. 1989. *Local Methods in Nonlinear Differential Equations*. Springer-Verlag.

[Brjuno & Jeltsch, 1995] Brjuno, A.D., & Jeltsch, R. 1995. *Stability Theory*. Verlag.

[Brockett, 1976] Brockett, R.W. 1976. Nonlinear systems and differential geometry. *IEEE Proceedings*, **64**, 61–72.

[Brunovsky, 1970] Brunovsky, P. 1970. A classification of linear controllable systems. *Kibernetika (Praha)*, **6**, 173–188.

[Brunovsky, 1975] Brunovsky, P. 1975. Contrôlabilité bang-bang, contrôlabilité différentiable. *Annali di Matematica Pura ed Applicata*, 93–119.

[Buck, 50, Nov, 1943] Buck, R.C. 50, Nov, 1943. Partition of Space. *American Mathematical Monthly*, 541–544.

[Byrnes, 1998] Byrnes, C. I. 1998. On the Riccati partial differential equation for nonlinear Bloza and Lagrange problems. *J. of Mathematical Systems, estimation and control*, **8**(1), 1–54.

[Byrnes *et al.* , 1997] Byrnes, C. I., Delli Priscoli, F., Isidori, A., & Kang, W. 1997. Structurally stable output regulation of nonlinear systems. *automatica*, **33**(3), 369–385.

[Byrnes, 1983] Byrnes, C.I. 1983. Control theory, inverse spectral problem and real algebraic geometry. *Proceedings of the conference held at Michigan Technology*, Birkhäuser.

[Byrnes, 1997] Byrnes, C.I. 1997. *Output regulation of uncertain non linear system*. Brikhauser.

[Califano *et al.* , 1999] Califano, C., Monaco, S., & Normand-Cyrot, D. 1999. On the problem of feedback linearization. *Systems et control letters*, **36**, 61–67.

[Calise & Rydyk, 1998] Calise, A. J., & Rydyk, R. T. 1998. Nonlinear adaptive flight control using neural networks. *IEEE control systems*, 14–24.

[Cartan, 1914] Cartan, E. 1914. Sur l'équivalence absolue de certains systèmes d'équations différentielles et sur certaines familles de courbes. *Bullentin de la Société Mathématique de France*, **42**, 12–48.

[Cartan, 1930] Cartan, E. 1930. Sur un problème d'équivalence et la théorie des espaces métriques généralisés. *Mathematica*, **4**, 114–136.

[Cartan, 1951] Cartan, E. 1951. *La théorie des groupes finis et continus et la géométrie différentielle*. Gauthier-Villars.

[Cartan, 1958] Cartan, E. 1958. *Leçon sur les invariants intégraux*. Hermann.

[Cartan, 1977] Cartan, H. 1977. *Cours de Calcul différentiel*. Hermann.

[Casas *et al.* , 1998] Casas, E., Troltzsch, F., & Unger, A. 1998. Second order suffisent optimality conditions for a class of elliptic control problems. *In : Contrôle et équations aux dérivées partielles*, vol. 4.

[Celikovsky, 1995] Celikovsky, S. 1995. Topological equivalences and topological linearization of controlled dynamical systems. *Kybernetika*, **31**, 141–150.

[Charlet & Lévine, 1989] Charlet, B., & Lévine, J. 1989. On Dynamic Feedback Linearization. *Systems and Control Letters*, **13**, 143–151.

[Charlet *et al.* , 1989] Charlet, B., Lévine, J., & Marino, R. 1989. On dynamic feedback linearization. *systems and control letter*, **13**, 143–151.

[Chaudhuri, 1995] Chaudhuri, T. R. 1995. *From conventional control to intelligent neurocontrol methods - a survey of the literature*. Tech. rept. 95/170C. Macquarie university, dept of computing.

[Chen, 1993] Chen, C. 1993. Self-organizing Neural Control System Design for Dynamic Processes. *Int. J. Control*, **24**(8), 1487–1507.

[Chen, 1995] Chen, F. 1995. Adaptative Control of a Class of Nonlinear Discerte-Time System Using Neural Networks. *IEEE trans. Automatic Control*, **40**(5), 791–801.

[Cheremensky & Fomin, 1996] Cheremensky, A., & Fomin, V. 1996. *Operator approach to linear control theory*. Kluwer.

[Chung, 1967] Chung, K. L. 1967. *Markov chains with stationnary transition probabilites*. Springer-Verlag.

[Clark, 1985] Clark, C.W. 1985. *Theory and applications of nonlinear control systems*. Elsevier. Chap. Optimal control theory and renewable ressource management, pages 60–90.

[Clarke *et al.* , 1999] Clarke, F. H., Ledyaev, Yu. S., Sontag, E. D., & I., Subbotin A. 1999. Asymptotic controllability implies feedback stabilization. *IEEE trans. on automatic control*.

[Clerget, 1988] Clerget, M. 1988. Training a 3-node neural network if NP complet. *ACM*, Proceedings of the first ACM workshop on the computational learning theory.

[Coron, 1994] Coron, Jean-Michel. 1994. On the stabilization by output feedback law of controllable and observable systems. *Math. Control Signals Systems*, **7**, 187–216.

[Crevier, 1993] Crevier, D. 1993. *A la recherche de l'intelligence artificielle.* flammarion.

[Cypkin, 1962] Cypkin, J. Z. 1962. *Theorie des systèmes asservis par plus ou moins.* Dunod.

[Dayan, 1995] Dayan, P. 1995. *Feudal Q-learning.* Tech. rept. University of tonronto, Dept of computer science.

[Dies, 1983] Dies, J.E. 1983. *Chaînes de Markov sur les permutations.* Lecture Notes in Mathematics. Springer-Verlag.

[Dreiseitl, 1992] Dreiseitl, S. 1992. *Accelerating the backpropagagtion algorithm by local methods.* Tech. rept. Research institute for symbolic computation, Johannes Kepler universitat.

[Dreisietl, nov 1992] Dreisietl, S. nov 1992. *Accelerating the Backpropagation Algorithm by local methods.* Tech. rept. J Kepler Universitat, Linz, Austria.

[Dubreil, 1964] Dubreil, P. 1964. *Leçon d'algèbre moderne.* Collections universitaires de Mathématiques.

[Eccles, 1989-1992] Eccles. 1989-1992. *Évolution du cerveau et création de la conscience.* Champs/ Flammarion.

[Edelsbrunner, 1987] Edelsbrunner, H. 1987. *Algorithms in Combinatorial Geometry.* Springer-Verlag. Chap. Fundamental Concepts in Combinatorial Geometry.

[Ekeland, 1968] Ekeland. 1968. Aspect concret du problème de poursuite. *In : séminaire Lions-Schwartz.*

[Engelbrecht, 1988] Engelbrecht, J.K. 1988. *Nonlinear evolution equations.* Wiley.

[Fang & Kincaid, 1998] Fang, Y., & Kincaid, T. G. 1998. Global properties for a class of dynamical neural circuits. *J. Franklin inst.*, **335B**(1), 163–177.

[Faure, 1986] Faure, P. 1986. *Eléments d'Automatique.* Dunod.

[Faure & Clerget, 1979] Faure, P., & Clerget, M. 1979. *Opérateurs Rationnels Positifs, Application à l'hyperstabilité et aux Processus Aléatoires.* Dunod.

[Favard, 1957] Favard, J. 1957. *Cours de géométrie différentielle locale.* Gauthier-Villars.

[Feldbaum, 1973] Feldbaum, A. 1973. *Principes théoriques des systèmes asservis optimaux.* MIR.

[Felgenhauer, 1996] Felgenhauer, U. 1996. *On optimal criteria for control problems. part I : theory.* Tech. rept. BTU Cottbus.

[Fitt, 1972] Fitt, J.M. 1972. On the observability of non linear systems. *Verlag*, **4**, 129–156.

[Fitt, 1987] Fitt, J.M. 1987. Feedback linearization of discrete time systems. *Systems Control Letters*, **9**, 411–416.

[Fliess *et al.* , 1992] Fliess, M., Lievine, J., & Martin, Ph. 1992. Sur les systèmes non linéaires différentiellement plats. *CR Acad. Sci. Paris*, 619–624.

[Fliess *et al.* , 1995] Fliess, M., Lévine, J., Martin, P., Ollivier, F., & Rouchon, P. 1995. *Flatness and dynamic feedback linearizability : two approaches.* Tech. rept. CNRS-supélec.

[Fomin, 1991] Fomin, V. N. 1991. *Discrete linear control systems.* Kluwer.

[Fréchet, 1938] Fréchet, M. 1938. *Méthode des fonctions arbitraires. Théorie des événements en chaîne dans le cas d'un nombre fini d'états possibles.* Gauthier-Villard.

[Freeman *et al.* , 1998] Freeman, R. A., Krstic, M., & Kokotovic, P. V. 1998. Robustness of adaptive nonlinear control to bounded uncertainties. *Automatica*, **34**(10), 1227–1230.

[Fregene & Kennedy, 1999] Fregene, K, & Kennedy, D. 1999. Control of a high-order power system by neural adaptive feedback linearization. *In : Proceedings ISIC/ISAS.*

[Freidlin & Ventcel, 1998] Freidlin, M, & Ventcel, A. 1998. *Random perturbations of dynamical systems.* Springer-Verlag.

[Frieze, 1989] Frieze, A. 1989. *Random Graphs.* Jhon Wiley.

[Fu & Miller, 1997] Fu, J. H., & Miller, D. F. 1997. On the matrix method for generalized Poincaré normal form reduction. *random and computational dynamics*, **5**(2-3), 125–144.

[Fujisawa & Kuh, 1972] Fujisawa, T., & Kuh, E. S. 1972. Piecewise linear theory of nonlinear networks. *SIAM J. of Appl Math*, **22**, 307–328.

[Gardner, 1982] Gardner, R. B. 1982. Differencial geometric methods interfacing control theory. *In : Differential geometric control theory.*

[Garofalo *et al.* , 1993] Garofalo, F., Celentano, G., & Glienemo, L. 1993. Stability robustness of interval matrices via Lyapunov quadratic functions. *IEEE trans on automatic control*, **38**(2).

[Gille *et al.* , 1958] Gille, J. C., Declaulne, P., & Pélegrin, M. 1958. *Systèmes asservis non linéaires*. Dunod.

[Girosi, 1995] Girosi, Frederico. 1995. Approximation error bounds that use VC-bounds. *In : Proceedings ICANN*.

[Goh, 1992] Goh, C. J. 1992. The neural network approach to several nonlinear control problems. *Pages 2749–2761 of : World congress of nonlinear analysts*, vol. I-IV.

[Gorinevsky & Feldkamp, 1996] Gorinevsky, D., & Feldkamp, L. A. 1996. RBF network feedforward compensation of load disturbance in idle speed control. *IEEE control systems*, december.

[Goursat, 1905] Goursat, E. 1905. *Cours d'analyse mathématique*. Gauthier-Villars.

[Goursat, 1922] Goursat, E. 1922. *Leçons sur le problème de Pfaff*. Hermann.

[Goutte & Ledoux, 1996] Goutte, C., & Ledoux, C. 1996. *Overview of connectionnist control using MLP*. Tech. rept. LAFORIA, paris VI.

[Grossberg, 1988] Grossberg, S. 1988. Nonlinear neural networks : principles, mechanisms, and architectures. *Neural Networks*, **1**, 17–61.

[Gullapalli, 1991] Gullapalli, V. 1991. *Learning to control dynamic systems via associative reinforcement learning*. Tech. rept. university of massachusetts, computer and infomation sciences department.

[Hace *et al.* , 1995] Hace, A., Safaric, R., & Jezernik, K. 1995. *Artificial neural network control for manipulators and Lyapunov theory*. Tech. rept. university of maribor, faculty of electrical engineering and computer sciences.

[Halanay, 1994] Halanay, A. 1994. *Time-varying discrete linear systems input-output operator*. Birhaüser-Verlag.

[Hayman, 1964] Hayman, W.K. 1964. *Meromorphic functions*. Clarendon Press, Oxford.

[Hayman, 1971] Hayman, W.K. 1971. *Analytic and algebraic independance of meomorphic functions*. Springer-Verlag.

[Heermann, 1992] Heermann, P. D. 1992. *Neural network techniques for stable learning control of nonlinear systems*. Ph.D. thesis, university of texas at austin.

[Helton & Zhan, 1997] Helton, J. W., & Zhan, W. 1997. Piecewise Riccati equations and the bounded real lemma. *Int journal of robust and nonlinear control*, **7**, 741–757.

[Hennequin & Tortrat, 1965] Hennequin, P. L., & Tortrat, A. 1965. *Théorie des probabilités et quelques applications*. Masson.

[Hermina-Alonso & Pilar-Perez, 1996] Hermina-Alonso, T., & Pilar-Perez, M. 1996. Brunovsky canonical form for linear systems over commutative rings. *Linear Algebra and its applications*, **233**, 131–147.

[Hernadez-lerma, 1996] Hernadez-lerma, O. 1996. *Discrete time markov control process*. Springer.

[Hirsch & Smith, 2003] Hirsch, Morris W., & Smith, Hal L. 2003. Competitive and cooperative systems : mini-review. *Lecture Notes in Control and Inform. Sci.*, **294**, 183–190.

[Hirsh, 1974] Hirsh, M. W. 1974. *Smoothing of piecewise linear manifolds*. Princeton university press.

[Horng, 1998] Horng, J. H. 1998. Output tracking of a class of unknown nonlinear discrete-time systems using neural networks. *J. Franklin Inst.*, **335B**(3), 503–515.

[Huang & Huang, 1996] Huang, Y. P., & Huang, C. C. 1996. The integration and application of fuzzy and grey modelling methods. *fuzzy sets and systems*, **78**(107-119).

[Hunt *et al.* , 1982] Hunt, L. R., Renjeng, S., & Meyer, G. 1982. Design for multi-input nonlinear systems. *In : Differential geometric control theory*.

[Hunt, 1981] Hunt, L.R. 1981. Linear equivalent of nonlinear time varying systems. *International symposium on mathematical theory of networks and systems*, 119–123.

[Hunt & Su, 1983] Hunt, L.R., & Su, R. 1983. Design for multi-input nonlinear systems. *Proceedings of the conference held at Michigan Technology*, 268–297.

[Isidori, 1989] Isidori, A. 1989. *Non linear control systems : an introduction*. Springer-Verlag.

[Isidori, 1995] Isidori, A. 1995. *Nonlinear control systems*. Springer.

[Isidori *et al.* , 1981] Isidori, A., Kerner, A.J., & Gori-Giori, C. 1981. Nonlinear decoupling via feedback : a differential geometrical approach. *IEEE trans. on Automatic Control*, **26**, 331–345.

[Jabri & Flower, 1992] Jabri, M., & Flower, B. 1992. weight perturbation : an optimal architecture and learning technics for analog VLSI feedforward and recurrent multilayered networks. *IEEE trans. Neural Networks*, **3**(1), 154–157.

[Jabubczyk, 1980] Jabubczyk, B. 1980. Existence and uniqueness of realization of nonlinear systems. *SIAM J. Contr. Optimiz.*, **18**, 455–471.

[Jagannarthan, 1996] Jagannarthan, S. 1996. Multilayer Discrete-Time Neural-Net Controller with Guaranteed Performance. *IEEE trans. on Neural Networks*, **7**(1), 107–130.

[Jakubczyk, 1980] Jakubczyk, B. 1980. On linearization of control systems. *Bulletin de l'Académie Polonaise des Sciences, Ser Sci Math*, **XXVIII**(9-10), 517–522.

[Jakubczyk & Respondek, 1980] Jakubczyk, B., & Respondek, W. 1980. On linearization of control systems. *bulletin de l'academie polonaise des sciences*, **28**(9-10).

[Jakubczyk & Sontag, 1990] Jakubczyk, B., & Sontag, E. D. 1990. Controllability of nonlinear discrete time systems : a lie-algebraic approach. *SIAM J. Control and Opt.*, **28**, 1–33.

[Jakubczyk *et al.* , 1984] Jakubczyk, B., Respondek, Witold, & Tchon, K. (eds). 1984. *Geometric theory of nonlinear control systems*. international conference, Bierutowice, Poland.

[Jervis & Fitzgerald, 1993] Jervis, T. T., & Fitzgerald, W. J. 1993. *Optimization schemes for neural networks*. Tech. rept. CUED/F-INFENG/TR 144. Cambridge university engineering dept.

[Jin & Pipe, 1993] Jin, Y., & Pipe, A.G. 1993. Neural Networks for prediction and control of discrete systems. *Parallel Sci. Comput.*, **1**(4), 395–419.

[Johansson & Rantzer, 1996] Johansson, M., & Rantzer, A. 1996. *Computation of piecewise quadratic Lyapunov functions for hybrid systems*. Tech. rept. Lund institute of technology, Dept of automatic control.

[Johansson & Rantzer, 1997] Johansson, M., & Rantzer, A. 1997. On the computation of piecewise quadratic Lyapunov functions. *In : Proc of the 30th Conference on Decision and Control*.

[Jondarr, 1996] Jondarr, Gibb. 1996. *Backpropagation familiy album*. Tech. rept. C/TR96-05. Dpt of computing Macquarie university.

[Judjevic, 1997] Judjevic, V. 1997. *Geometric Control Theory*. Cambridge.

[Kailath, 1981] Kailath, T. 1981. *Lectures on Wiener and Kalman Filtering*. Springer-Verlag.

[Kalman, 1960] Kalman, R. E. 1960. A new approach to linear filtering and prediction problems. *J. of Basic Eng.*, **82**, 35–45.

[Kalman, 1972] Kalman, R.E. 1972. Kronecker invariants and feedback, Ordinary Differential Equations. *Acad. Press*, 459–471.

[Kambhampati *et al.* , 1996] Kambhampati, C., Manchanda, S., Delgado, A., Green, G., & Warwick, K. 1996. The relative Order and Inverse of Recurrent Networks. *Automatica*, **32**(1), 117–123.

[Kawamura & Ikai, 1991] Kawamura, Y, & Ikai, T. 1991. A relation between neural network learning and control theory. *Pages 469–474 of : Recent adv in mathematical theory of systems.*

[Keerthi & Ravindran, 1995] Keerthi, S. S., & Ravindran, B. 1995. *A tutorial survey of reinforcement learning.* Tech. rept. Indian institute of sciences, Dept of computer sciences.

[Khavin, 1997] Khavin, V.P. 1997. *Complex analysis I : entire and meromorphic functions.* Springer.

[Kirk, 1970] Kirk, D. E. 1970. *Optimal control theory.* Prentice-Hall.

[Kitchens, 1998] Kitchens, Buce P. 1998. *Symbolic dynamics.* Springer.

[Krener, 1975] Krener, A.J. 1975. Bilinear and nonlinear différential systems realization. *SIAM J. of Contr*, **13**, 825–834.

[Krylov, 1934] Krylov, N.M. 1934. *Sur quelques développements formels en série dans la mécanique non linéaire.* Kiev.

[Ku, 1995] Ku, C. 1995. Diagonal Recurrent Neural Networks for Dynamic Systems Control. *IEEE trans. on Neural Networks*, **6**(1), 144–155.

[Kuvayev & Sutton, 1997] Kuvayev, D., & Sutton, R. S. 1997. *Model-based reinforcement learning.* Tech. rept. university of massachusetts, Dept of computer science.

[Lafferriere & d., 1993] Lafferriere, G. A., & D., Sontag E. 1993. *Remarks on control Lyapunov functions for discontinuous stabilizing feedback.* Tech. rept. Portland university and Rutgers university, Depts of mathematics.

[Landau, 1981-82-83] Landau, I.D. 1981-82-83. *Outils et modèles mathématiques pour l'automatique, l'analyse de systèmes et le traitement du signal.* Ed CNRS.

[Lang, 1972] Lang, S. 1972. *Introduction to algebraic geometry.* Addison Wesley.

[Lang, 1982] Lang, S. 1982. *Feedbaxk control of linear and non linear systems.* Springer-Verlag.

[LaSalle, 1961] LaSalle, J.P. 1961. *Stability by Liapunov's Direct Method with Applications*. Academic Press.

[LaSalle, 1986] LaSalle, J.P. 1986. *The Stability and Control of Discrete Processes*. Springer-Verlag.

[Ledyaev & Sontag, 1997] Ledyaev, Y. S., & Sontag, E. D. 1997. *A Lyapunov characterisation of robust stabilization*. Tech. rept. Western Michigan university and Rutgers university, Depts of mathematics.

[Lehalle, 1994] Lehalle, Charles-Albert. 1994. *Dynamique de système à mémoire associative : commentaires d'articles de M. Benaim et M. Hirsch ; mémoire de DEA*. M.Phil. thesis, Université de Pierre et Marie Curie.

[Lehalle & Azencott, 1998a] Lehalle, Charles-Albert, & Azencott, Robert. 1998a. BASIS OF NONLINEAR CONTROL WITH PIECEWISE AFFINE NEURAL NETWORKS. *In : Nonlinear theory and its applications*. NOLTA.

[Lehalle & Azencott, 1998b] Lehalle, Charles-Albert, & Azencott, Robert. 1998b. Piecewise Affine Neural Networks and Nonlinear Control. *In : International conference on Artificial Neural Networks (ICANN)*. ICANN.

[Lehalle & Azencott, 1999a] Lehalle, Charles-Albert, & Azencott, Robert. 1999a. How Piecewise Affine Perceptrons can generate stable nonlinear controls. *In : IEEE International Symposium on Intelligent Control Intelligent Systems and Semiotics (ISIC-ISAS-1999), Cambridge, MA*. ISIC/ISAS.

[Lehalle & Azencott, 1999b] Lehalle, Charles-Albert, & Azencott, Robert. 1999b. Piecewise Affine Neural Networks and Nonlinear Control : stability results. *In : International conference on Artificial Neural Networks (ICANN)*. ICANN.

[Leitmann & Pandey, 1990] Leitmann, G., & Pandey, S. 1990. *Modelling, Estimation and control of systems with uncertainty*. Birkhauser. Chap. Aircraft control for flight in an uncairtain environment : takeoff in windshear, pages 283–302.

[Leung *et al.* , 1998] Leung, F. H., Lam, H. K., & Tam, P. K. 1998. Design of fuzzy controllers for uncertain nonlinear systems using stability and robustness analysis. *Systems and control letters*, **35**, 237–243.

[Levada, 1998] Levada, F. Y. 1998. *Control of indefinite nonlinear dynamical systems*. Springer Verlag.

[Levin, 1993] Levin, A. 1993. Control of Nonlinear Dynamical Systems using Neural Networks : Controllability and Stabilization. *IEEE trans. on Neural Networks*, **4**(2), 192–206.

[Levin & Narendra, 1993] Levin, A. U., & Narendra, K. S. 1993. Control of nonlinear dynamical systems using neural networks : controllability and stabilization. *IEEE trans on neural networks*, **4**(2), 192–206.

[Levin & Narendra, 1996] Levin, A.U., & Narendra, K.S. 1996. Control of Nonlinear Dynamical Systems Using Neural Networks - Part II : Observability, Identification, and Control. *IEEE trans. on Neural Networks*, **7**(1), 30–42.

[Liang, 1979/80] Liang, B. 1979/80. A classification of linear control systems. *Linear and multilinear algebra*, 261–264.

[Libre & Ponce, 1996] Libre, J, & Ponce, E. 1996. Global first harmonic bifurcation for odd piecewise linear control problem. *Dynam. Stability Systems*, **11**(1), 49–88.

[Lichnerowicz, 1947] Lichnerowicz, A. 1947. *Algèbre et analyse linéaire*. Masson.

[Lino, 1995] Lino, Li-Zhi. 1995. *The basic structure in discrete time optimal control algorithms*. Tech. rept. Hong Kong Baptist University, Dept of mathematics.

[Lions & Zuazua, 1995] Lions, J. L., & Zuazua, E. 1995. Approximate controllability of a hydro-elastic coupled system. *control, optimization and calculus of variations*, **1**, 1–15.

[Lions, 1968] Lions, J.L. 1968. *Contrôle optimal de systèmes gouvernés par des équations aux dérivées partielles*. Gauthier-Villard.

[Lobry, 1970] Lobry, C. 1970. Contrôlabilité des systèmes non linéaires. *SIAM J. control*, **8**(4), 573–605.

[Lyapunov, 1947] Lyapunov, A.M. 1947. *Problème général de la stabilité du mouvement*. Ann of Math Stud, Princeton UP.

[Magni & Bennani, 1997] Magni, J. F., & Bennani, S. 1997. *Robust flight linear control*. Springer Verlag.

[Magni *et al.* , 1997] Magni, Jean-François, Bennani, Samir, & Terlow, Jan (eds). 1997. *Robust flight control*. Springer-Verlag.

[Malliavin, 1972] Malliavin, P. 1972. *Géométrie Différentielle intrinsèque*. Hermann.

[McCallum, 1992] McCallum, R. A. 1992. *First results with utile distinction memory for reinforcement learning*. Tech. rept. 446. university of Rochester, Computer dept.

[Minsky, 1964] Minsky, M. 1964. *Perceptrons : an Introduction to computational geometry*. The MIT Press.

[Moller, 1994] Moller, J. 1994. *Lectures on Random Voronoïtessellations.* Springer-Verlag.

[Monaco & Normand-Cyrot, 1995] Monaco, S., & Normand-Cyrot, D. 1995. An unified representation for nonlinear discrete-time and sampled dynamics. *journal of mathematical systems, estimation and control*, **5**(1), 1–27.

[Montague *et al.* , 1991] Montague, G. A., Willis, M. J., J., Tham M., & Morris, A. J. 1991. Artificial neural network based control. *Control*, **1**(2), 266–271.

[Namba, 1979] Namba, M. 1979. *Families of meromorphic functions on compact riemann surfaces.* Springer-Verlag.

[Narendra, 1994] Narendra, K. 1994. Adaptative Control of Nonlinear Multivariable Systems using Neural Networks. *Neural Networks*, **7**(5), 737–752.

[Narendra & Mukhopadhyay, 1994] Narendra, K. S., & Mukhopadhyay, S. 1994. Adaptive control of nonlinear multivariable systems using neural networks. *neural networks*, **7**(5), 737–752.

[Nguang & Fu, 1998] Nguang, S. K., & Fu, M. 1998. Global quadratic stabilization of a class of nonlinear systems. *international journal of robust and nonlinbear control*, **8**, 483–497.

[Nijmeijer, 1990] Nijmeijer, H. 1990. *Non linear dynamical control system.* Springer-Verlag.

[Nyquist, 1932] Nyquist, H. 1932. Regeneration theory. *Bell Systems Technical Journal*, **11**, 126–147.

[Ollivier, 1992] Ollivier, F. 1992. *Some theoretical problems in effective differential algebra and their relation to control theory.* Tech. rept. Ecole polytechnique.

[Pallu de la Barrière, 1965] Pallu de la Barrière, R. 1965. *Cours d'automatique théorique.* Dunod.

[Papin, 1681] Papin, D. 1681. *Nouvelle Manière pour lever l'eau par la force du fer.* Societé Royale de Londres.

[Pazman, 1993] Pazman, A. 1993. *Nonlinear statistical models.* Kluwer.

[Pearlmutter, 1992] Pearlmutter, B. 1992. Gradient descent : second order momentum and saturating error. *Pages 887–894 of :* Moody (ed), *Advances in neural information processing systems*, vol. 4.

[Poggio & Girosi, 1986] Poggio, Tomaso, & Girosi, Frederico. 1986. *A theory of networks for approximation and learning.* Tech. rept. MIT AI Lab and Center for Bio.

[Polycarpou, 1996] Polycarpou, M. 1996. Stable Adaptative Neural Control Scheme for Nonlinear Systems. *IEEE trans. Automatic Control*, **41**(3), 447–451.

[Polycarpou & Ioannou, 1991] Polycarpou, M. M., & Ioannou, P. A. 1991. *Identification and control of nonlinear systems using neural network models : design and stability analysis*. Tech. rept. 91-09-01. university of california, dept of electrical engineering systems.

[Pomet, 1994] Pomet, J. B. 1994. *A differential geometric setting for dynamic equivalence and dynamic linearization*. Tech. rept. 2312. inria.

[Preparata, 1985] Preparata, P.F. 1985. *Computational Geometry : an Introduction*. Springer-Verlag.

[Proano & Bialasieuries, 1993] Proano, J.C., & Bialasieuries, J.T. 1993. Neurodynamic adaptative control system. *Kibernetica*, **29**(1), 3–47.

[Puskorius & Feldkamp, 1993] Puskorius, G. V., & Feldkamp, L. A. 1993. Neurocontrol of nonlinear dynamical systems with kalman filter trained recurrent networks. *IEEE trns on control*.

[Qiao & Mizumoto, 1996] Qiao, W. Z., & Mizumoto, M. 1996. PID type fuzzy controller and parameters adaptive method. *fuzzy sets and systems*, **78**, 23–35.

[Quam, 1969] Quam, P. 1969. *Introduction à la géométrie des variétés différentielles*. Dunod.

[Ram & Santamaria, 1993] Ram, A., & Santamaria, J. C. 1993. Multistrategy learning in reactive control systems for autonomous robotic navigation. *Informatica*, **17**(4), 347–369.

[Rantzer & Johansson, 1997] Rantzer, A., & Johansson, M. 1997. *Piecewise linear quadratic optimal control*. Tech. rept. ISRN LUTFD2/TFRT - 7569 - SE. Lund institute of technology, Dept of automatic control.

[Rauch, 1996] Rauch, V. 1996. *Stratégie de régulation par commande de couple*. Note technique D.R. Renault.

[Rauch, 1997] Rauch, V. 1997. *Modélisation et commande du couple d'un moteur à essence : préliminaires*. Note technique D.R. Renault.

[Riccati, 1742] Riccati, J. F. 1742. *Animadversations in aequations differentiales secondi gradus*. Vol. 8. Actorum eruditorum quae lipsiae publicantur.

[Robinson, 1999] Robinson, Clark. 1999. *Dynamical systems, second edition*. CRC Press.

[Rockafellar, 1988] Rockafellar, R. T. 1988. *On the essential boundedness of solutions to problems in piecewise linear quadratic optimal control*. Analyse mathematique et applications. Gauthier villars. Pages 437–443.

[Ronco, 1997] Ronco, E. 1997. *Incremental polynomial controller networks : two self organising non linear controllers*. Ph.D. thesis, Glasgaow university, Faculty of mechanical engineering.

[Ronco & Gawthrop, 1999] Ronco, E, & Gawthrop, P. J. abd Hill, D. J. 1999. *A practical continuous-time nonlinear generalized predictive controller*. Tech. rept. EE-99001. Department of mechanical engineering, university of glasgow.

[Ronco & Gawthrop, 1998] Ronco, E., & Gawthrop, P. J. 1998. *Two controller networks automatically constructed through system linearization and learning*. Tech. rept. CSC-98002. Centre for system and control, Department of mechanical engineering, university of glasgow.

[Rouchin, 1995] Rouchin, P. 1995. Necessary condition and genericity of dynamic feedback linearization. *journal of mathematical systems, estimation, and control*, **3**, 345–358.

[Rouchon, 1995] Rouchon, P. 1995. Necessary Condition and Genericity of Dynamic Feedback Linearisation. *J. of Mathematical Systems, Estimation and Control*, **5**(3), 345–358.

[Rumelhart *et al.* , 1986] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. 1986. *Parallel Distributed Processing*. Vol. 1, chap 8. MIT Press, Cambridge. Chap. Learning internal representation by error propagation.

[Sauer, 1971] Sauer, N. 1971. On the uniform convergence of relative frequencies of events to their probability. *Theory of probaility and its application*, **16**, 265–280.

[Sauer, 1972] Sauer, N. 1972. On the density of families of sets. *Journal of combinatorial theory (A)*, **13**, 145–147.

[Sheppard *et al.* , 1994] Sheppard, M., Oswald, A., Valenzuela, C., Sullivan, C., & Sotudeh, R. 1994. *Reinforcement learning in control*. Tech. rept. University of Teesside.

[Sontag, 1981] Sontag, E. D. 1981. Nonlinear regulation : the piecewise linear approach. *IEEE trans. on automatic control*, **26**(2), 346–358.

[Sontag, 1984] Sontag, E. D. 1984. *An algebraic approach to bounded controllability of linear systems*. Tech. rept. Rutgers university, dept of mathematics.

[Sontag, 1992a] Sontag, E. D. 1992a. Feedback stabilization using rwo-hidden-layer nets. *IEEE trans neural networks*, **3**, 981–990.

[Sontag, 1992b] Sontag, E. D. 1992b. *Neural networks for control*. Tech. rept. Rutgers university, dept of mathematics.

[Sontag, 1997] Sontag, E. D. 1997. *A learning result for continuous time recurrent neural networks*. Tech. rept. Rutgers university, dept of mathematics.

[Sontag, 1998a] Sontag, E. D. 1998a. *Recurrent neural networks : some systems-theoretic aspects*. Tech. rept. Rutgers university, dept of mathematics.

[Sontag & Qiao, 1998] Sontag, E. D., & Qiao, Y. 1998. *Further results on controllability of recurrent neural networks*. Tech. rept. Rutgers university, Dept of mathematics.

[Sontag & Sussmann, 1997] Sontag, E. D., & Sussmann, H. 1997. *Complete controllability of continuous-time recurrent neural networks*. Tech. rept. Rutgers university, Dept of mathematics.

[Sontag, 1989a] Sontag, E.D. 1989a. *Controllability and linearized regulation*. Tech. rept. rutgers university.

[Sontag, 1990a] Sontag, E.D. 1990a. *Mathematical Control Theory*. Springer-Verlag.

[Sontag, 1995] Sontag, E.D. 1995. *Feedback stabilization of nonlinear systems*. Tech. rept. Rutgers university.

[Sontag, 1998b] Sontag, E.D. 1998b. *Mathematical Control Theory, second edition*. Springer-Verlag.

[Sontag & Writh, 1997] Sontag, E.D., & Writh, F. R. 1997. *Remarks on universal nonsingular controls for discrete-time systems*. Tech. rept. Elsevier preprint.

[Sontag, 1989b] Sontag, S. D. 1989b. Remarks on the time-optimal control of a class of hamiltonian systems. *Pages 217–221 of : Proc IEEE Conf Decision and Control*.

[Sontag, 1990b] Sontag, S. D. 1990b. Feedback stabilization of nonlinear systems. *Progr. Systems Control theory*, **4**, 61–81.

[Spedicato, 1993] Spedicato, E. E. 1993. *Algorithms for continuous optimization*. Kluwer Acad. Pub.

[Su, 1982] Su, R. 1982. On the linear equivalent of nonlinear systems. *Systems and control letters*, v 2, n 1.

[Sussmann, 1982] Sussmann, H. J. 1982. Lie brackets, real analyticity and geometric control. *In : Differential geometric control theory*.

[Sussmann, 1988] Sussmann, H. J. 1988. *Why real analyticity is important in control theory*. Tech. rept. rutgers university.

[Sussmann, 1989] Sussmann, H. J. 1989. *Optimal control*. Tech. rept. rutgers university, dept of mathematics.

[Sussmann, 1998] Sussmann, H. J. 1998. *Geometry and optimal control*. Tech. rept. rutgers university.

[Sussmann & Brockett, 1982] Sussmann, H.J., & Brockett, R.W. 1982. *Differential Geometric Control Theory*. Birkhauser.

[Szentágithai, 1978] Szentágithai, J. 1978. The neuron network of the cerebral cortex : a functionnal interpretation. *Proc R Soc Lond [Biol]*, **201**, 219–48.

[Szentágithai, 1983] Szentágithai, J. 1983. The modular archtectonic principle of neural centers. *Rev Physiol Biochem Pharmacol*, **98**, 11–61.

[Tan & Yu, 1993] Tan, S., & Yu, Y. 1993. *On-line stable nonlinear modelling by structurally adaptive neural nets*. Tech. rept. National university of singapore, dept of electrical engineering.

[Thrun & Mitchell, 1993] Thrun, S., & Mitchell, T. M. 1993. *Lifelong Robot Learning*. Tech. rept. IAI-TR-93-7. University of Bonn, Institut for informatik.

[Traver *et al.* , 1999] Traver, L., Atkinson, C., & Long, T. 1999. Neural Network-Based Diesel Engine Emissions Prediction Using In-Cylinder Combustion Pressure. *In :* SAE (ed), *International Fuels and Lubricants Meeting Exposition, May 1999, Dearborn, MI, USA, Session : Emission Formation Processes in SI and Diesel Engines*, vol. 1999-01-1532.

[Tsinias, 1995] Tsinias, J. 1995. Smoothly global stabilization by dynamical feedback. *SIAM Journal of Control and optimization*, **33**(4), 1071–1085.

[Tsinias, 1996] Tsinias, J. 1996. Version of Sontag's input to state stability condition and output feedback global stabilization. *journal of mathematical systems, estimation and control*, **6**(1), 1–17.

[Tzirkel-Hancock & Fallside, 1991] Tzirkel-Hancock, E., & Fallside, F. 1991. *A direct control method for a class of nonlinear systems using neural networks*. Tech. rept. cambridge university engineering dept, england.

[Udriste, 1980] Udriste, C. 1980. *Complex analysis proceedings of the summer school*. Springer-Verlag.

[Udriste, 1994] Udriste, C. 1994. *Convex functions and optimization methods on riemannian manifold*. Kluwer Acad. Pub.

[Unar & Murray-smith, 1995] Unar, M. A., & Murray-smith, D. J. 1995. *Artificial neural networks for ship steering control systems*. Tech. rept. centre for system and control, dept of electronics and electrical engineering, university of glasgow.

[Ungerer, 1999] Ungerer, C. 1999. *Identifying time-delays in nonlinear control systems : an application of the adaptive time-delay neural network*. Tech. rept. technische universitat munchen, institut fur informatik.

[Van Der Essen, 1995] Van Der Essen, A. 1995. *Automorphisms of affine spaces*. Kluwer.

[Vanualaila *et al.* , 1998] Vanualaila, J., HA, J. H., & Nakagiri, S. I. 1998. Pathfinding and lyapunov function. *dynamics and stability of systems*, **13**(4).

[Vapnik, 1995] Vapnik, V. 1995. *The Nature of statistical learning theory*. Springer.

[Vayatis & Azencott, 1999] Vayatis, N., & Azencott, R. 1999. Distribution-Dependent Vapnik-Chervonenkis Bounds. *Pages 230–240 of : EuroCOLT 1999*.

[Venugopal & Pandya, 1994] Venugopal, K., & Pandya, A. 1994. A Recurrent Neural Network Controller and Learning Algorithm for the On-Line Learning Control of Autonomous Underwater Vehicules. *Neural Networks*, 833–846.

[Warwick, 1995] Warwick, K. 1995. A Critique of Neural Networks for Discrete-Time Linear Control. *Int. J. Control*, **61**(6), 1253–1264.

[Warwik & Irwin, 1992a] Warwik, K., & Irwin, G.W. 1992a. Neural network for control systems. *Automatica*, **28**(6), 1083–1112.

[Warwik & Irwin, 1992b] Warwik, K., & Irwin, G.W. 1992b. *Neural network for control systems*. IEEE Workshop.

[Weintraub & White, 1997] Weintraub, B., & White, D. 1997. Learning optimal control for unmanned supermaneuverable technology : LOCUST. *In : INNS World congress on neural networks*.

[Wiener, 1948] Wiener, N. 1948. *Cybernetics, or control and communication in the animal and the machine*. Hermann.

[Willems, 1971] Willems, J. C. 1971. Least squares stationnary optimal control and algebraic Riccati equation. *IEEE trans Autom. Control*, 621–634.

[Wonham, 1974] Wonham, W. 1974. Linear and multilinear control, a geometric approach. *Lecture Notes in Economics and Mathematical Systems, Springer-Verlag*, 121–126.

[Wredenhagen & Belanger, 1994] Wredenhagen, G. F., & Belanger, P. R. 1994. Piecewise linear quadratic control for systems with input constraints. *Automatica*, **30**(3), 403–416.

[Yang, 1982] Yang, C.C. 1982. *Factorization theory of meromorphic functions*. ? ? ?

[Zagalak & Lafay, 1993] Zagalak, P., & Lafay, J. 1993. System structure and control 1. *Acad. of Sc. of the Czech Rep.*, 405–515.

[Zames, 1998] Zames, G. 1998. Adaptive control : toward a complexity-based general theory. *Automatica*, **34**(10), 1161–1167.

[Zeller *et al.* , 1997] Zeller, M., Sharma, R., & Schulten, K. 1997. Motion planning of a pneumatic robot using a neural network. *IEEE control systems*.

[Zhang, 1993] Zhang, G.H. 1993. *Theory of entire and meromorphic functions*. AMS.

Annexes

Développements informatiques

[3/package]Package MapleV

Une package Maple a été développé pour étudier le découpage engendré par un PAP. Il a été utilisé pour initialiser le PAP émulant un contrôle non linéaire sur le véhicule Renault et pour générer les illustrations de ce mémoire.

Ses fonctions principales sont : le découpage d'une cellule polyhédrale en sous-cellules compte-tenu des matrices et vecteurs de poids d'un PAP ; l'affichage de ces cellules (en dimension 2) ; la simulation des trajectoires d'un PAP en boucle fermée ; la mise au point d'un contrôleur LQI de type PAP à partir d'un système dynamique formel.

Voici les principales fonctionnalités de ce package utilisées dans cette partie :

dansAide
`dansAide(fct, explications) : assigne explications comme aide a fct`

"vide"
`vide(lpts) : renvoie [] si les points de la liste lpts sont tous identiques, lpts sinon. pour l'instant inutilise`

"PPForDraw"
`PPForDraw(prpt) : renvoie une suite de segments permettant de dessiner la partition prpt. utilise les variables globales pNM`

"Arrow"
`Arrow(point, vect, t1, t2, t3) : renvoie une courbe partant de point de direction vect, de t2 epaisseur de corps, t2 epaisseur de tete et la tete occupant t3 de la longueur`

"addLine"
addLine(W1, b1, X1, X2, X3) : renvoie W1 et b1 augmentes de la ligne qui emule une bande passant par (X1,X2) et X3

"drawParts"
drawParts() : renvoie une representation graphique des cellules cells qui est une liste de listes de sommets

"interHL"
interHL(lpts, line) : renvoie deux ensembles de coordonnees correspondants a la liste des points lpts coupee en deux par la ligne line

"concatInters"
concatInters(liste) : prépare le résultat de inetCells pour l'afficahge

"coordsToPointsB"
coordsToPointsB(listedecoords, str) : renvoie une liste de points correspondants aux coordonnees de la liste en les nommants a partir de str, cf coordsToPoints

"linizeM"
linizeM(matrix) : divise chaque élément de la matrice par la somme de sa ligne

"lineSum"
lineSum(M, l) : renvoie la summe de la ligne l de M

"coordsToPoints"
coordsToPoints(listedecoords) : renvoie une liste de points correspondants aux coordonnees de la liste, cf coordsToPointsB

"naturalIndex"
naturalIndex(cellCode) : retourne le numéro d'une cellule

"makeGrid"
makeGrid([xMin, xMax], nbX, [yMin, yMax], nbY) : retourne une liste de points répartis suivant ces paramètres

"numero"
numero(cell, cdx, W1, b1, W2, b2) : attribue un numero a la cellule cell, d'apres l'index cdx (il doit être défini par correspIdx)

"sats"
sats(W1,B1,W2,B2,cell) : numérotation vectorielle (codage) d'une cellule

"countPoints"
countPoint(grid, nbCells) : renvoie un tableau de répartition des points dans les cellules au départ (lignes) et à la fin (clonnes). repart[i,j] est le numbre de points qui sont passés de la cellule i à la cellule j. Nécessite de nombreuses variables globales (W1, b1, W2, b2, et setCorrespIdx appelé)

"hhullb"
hhullb(listedepoints, str) : retourne la liste de segments coorespondants au polyhedre en les nommant a partir de strN, cf hhull

"countPointsIte"
countPointIte(grid, base) : renvoie un tableau de répartition des points dans les cellules au départ (lignes) et à la fin (clonnes). repart[i,j] est le numbre de points qui sont passés de la cellule i à la cellule j. Nécessite de nombreuses variables globales (W1, b1, W2, b2, et setCorrespIdx appelé)

"cellVsLine"
cellVsLine(C, L) : intersection d'une cellule et d'un segment

"trajectoires"
trajectoires(xyi, xyf, W1, b1, W2, b2, nbe, rg) : renvoie une liste de trajectoires de duree nbe partant d'un quadrillage de xyi a xyf de rg2 points

"maxAbs_xy"
maxAbsxy(liste de couples [x, y] : renvoie la max en v.a. de tout le monde

"PAP_1"
PAP1([x,y], W1, b1) : action de la première couche

"isIntoCell"
isIntoCell([x, y], cell) : retourne t or f suivant que le point [x,y] est ou non dans la cellule cell

"sat2num"
sat2num(cellCode) : retourne un numéro d'ordre (à la Gödel) correspondant

"trajectoire"
trajectoire(xy, W1, b1, W2, b2, nbe) : renvoie une liste des nbe positions consecutives partants de xy

"cellCode"
cellCode(W1, b1, W2, b2, [x, y]) : renvoie le numéro multidim (compartible sats) de la cellule qui contient le point

"sat"
sat(W1,b1,cell) : retourne le vecteur de saturation de la cellule cell de la partition primaire. utilise isob.

"triangleArea"
triangleArea(liste) : surface d'un triangle, exception=-1

"initWS"
initWS(nbhide) : genere les poids (variables globales : W1, b1, W2, b2).

"primPartL"
primPartL(W1, b1) : genere les lignes de la partition primaire

"enleveDoublets"
enleveDoublets(liste) : explicite

"makeRel"
makeRel(W1, b1, W2, b2, xyi, xyf, n) : trace l'ensemble des fleches qui correspondent a un quadrillage de n

"isob"
isob(cellule) : retourne l'isobarycentre des points dont les coordonnees sont les liste cell

"inter2cells"
inter2cells(C1, C2) : renvoie la liste des points d'intersection entre les deux cellules

"polyArea"
polyArea(liste) : surface d'un polygone

"MX2"
MX2(W1, b1, W2, b2, cell) : renvoie la matrice equivalente a Psi avant saturation sur cell. utilise satMX.

"localizePoint"
localizePoint([x,y]) : retourne le numéro d'orde de la cellule correspondante (sous réserve que les poids soient W1, b1, W2, b2 et que setCorrespIdx() ait été appelé)

"PAP_2"
PAP2([x,y], W1, b1) : après passage de l'activation

"decoupe"
decoupe(cells, W1, b1, W2, b2) : renvoie la liste de cellules decoupees par le PAP

"firstTraj"
firstTraj(tarj) : premiers éléments des trajectoires

"PAP_3"
PAP3([x, y], W1, b1, W2, b2) : première étape de l'action de la couche 2

"PAP"
PAP(xy, W1, b1, W2, b2) : renvoie une liste qui contient l'image de xy par [W1, b1, W2, b2]

"lastTraj"
lastTraj(tarj) : derniers éléments des trajectoires

"list2mat"
list2mat(L) : renvoie L sous forme de matrice

"primaryPart"
primaryPart(hypercube, W1, b1) : construction de la partition initiale sur l'hypercube demande, elle est retournee sous la forme d'une liste de listes de coordonnees ordonnes. utilise primaryPt(HC, W, B, nbe)

"Phi"
Phi(X) : activation affine par morceaux, vectorielle

"diffpl"
diffpl(lpts, pt) : renvoie vrai si lpts est vide ou que le dernier point de lpts est different de pt, vrai sinon

"setCorrespIdx"
setCorrespIdx(list : :Xpart) : stoque dans une variable globale une indexation des cellules

"getLimit"
getLimit(transitionMatrix) : renvoie la distrib limite qui correspond à la matrice de transition en question

"ligne"
ligne(X1, X2, X3) : renvoie la ligne de [W1, b1] qui emule une bordure passant par (X1,X2) pour -1 et X3 pour 1

"aides"
aides() : retourne toutes les aides disponibles

"phi"
phi(x) : activation affine par morceaux, scalaire

"intersecte"
intersecte([[x1, y1], [x2, y2]], [[X1, Y1], [X2, Y2]]) : intersection des deux segments. Pas d'intersection renvoie false. if(type(false, numeric)) pour le code d'erreur

"hhull"
`hhull(listedepoints) : retourne la liste de segments coorespondants au polyhedre, cf hhullb`

"intersSL"
`intersSL(ln,sg) : retourne le point d'intersection entre une ligne ln et un segment sg, si ils sont paralleles ou que le segment est un point retourne [] et si l'intersection n'est pas a l'interieur du segment, retourne 0`

"secondaryPart"
`secondaryPart(Hc, W1, b1, W2, b2) : renvoie la partition terminale du PAP sur l'hypercube Hc defini par la liste de ses sommets (ordonnee).`

"satMX"
`satMX(W1, b1, cell) : renvoie W1' et b1', les versions saturees de W1 et b1 sur cell, cf MX2`

"aideSur"
`aideSur(fonction) : renvoie l aide sur la fonction`

"filterPartG"
`filterPartG(part) : renvoie une representation graphique de la partition part`

Articles

Les aspects les plus saillants de ces travaux ont été exposés lors de quatre conférences internationales, et ont donc été publiés dans les actes de ces congrès.

Les deux premiers exposés touchent au mode d'initialisation des PAP particulièrement adapté au contrôle. Celui à destination d'ICANN met en avant les aspects théoriques et celui à destination de NOLTA les aspects appliqués.

ICANN'98 : Piecewise Affine Neural Networks and Nonlinear Control. Charles-Albert Lehalle et Robert Azencott.

NOLTA'98 : BASIS OF NONLINEAR CONTROL WITH PIECEWISE AFFINE NEURAL NETWORKS. Charles-Albert Lehalle et Robert Azencott.

ABSTRACT.

Linear control of systems governed by an equation such as $x_{n+1} = Ax_n + Bu_n$ (where x and u belong to finite dimensional vector spaces) is widely known [AZZO, 1988]. Control design is more complex when dealing with nonlinear systems (for instance [BONNARD, 1981] or [BYRNES, 1983]). A well-known method consists in linearizing the system at positions that occur the most often, then solving this serie of linear control problems, and finally patching such local controls together [JAKUBCZYK, 1980].

The piecewise affine neural networks are identical to feed-forward neural networks except that their activation function is continuous piecewise affine rather than sigmoidal. These neural networks can implement quite generic continuous piecewise affine functions on polyhedral cells. Therefore they stand between a collection of local linear controls and a nonlinear control deduced straight from the nonlinear system (this is not always possible, especially when the exact dynamic of the model is unknown [HUNT & SU, 1983] and [SUSSMANN & BROCKETT, 1982]).

On the other hand, since the learning phase of a neural network by gradient retropropagation in a closed loop is difficult [Jagannarthan, 1996], a neural network accurate initialization based on linearizations of the system to be controlled is a real benefit. Besides, piecewise affine neural networks can be considered as approximations of neural networks with sigmoidal activation ; this allow some results to be extended to more standard perceptrons.

The aim of this paper is to exhibit the main properties of this new kind of neural networks and to show how the training of these networks can be initialized from linear functions, especially in a control environment

First, the action of a piecewise affine neural network is specified by a partition of its input space such that on each polyhedral cell generated by this partition, the network action is affine. Then the slopes and the constants of these affine pieces are characterized as functions of the network weights. The number of cells are determined as a function of the number of hidden neurons. Then a basic result is that in a given hypercube any continuous piecewise affine function can be emulated by a piecewise affine neural network.

The next section presents a methodology to initialize piecewise affine neural networks for controlling nonlinear systems and shows an illustration.

Les deux exposés suivants mettent en avant les résultats de stabilité sur le contrôle engendré par un PAP. L'exposé à ICANN est centré sur les aspects théoriques alors que celui à ISIC met en avant les implications opérationnelles de ces résultats.

ICANN'99 : Piecewise Affine Neural Networks and Nonlinear Control : stability results. Charles-Albert Lehalle et Robert Azencott.

ISIC/ISAS'99 : How Piecewise Affine Perceptrons can generate stable nonlinear controls. Charles-Albert Lehalle et Robert Azencott.

Abstract.

Design of a nonlinear control is a difficult task, especially when an exact model of the system to control in not known ([Hunt & Su, 1983] or [Sussmann & Brockett, 1982]). The possibilities of tuning artificial neural networks to control have been explored for instance by S. Jagannarthan [Jagannarthan, 1996] and E.D. Sontag [Sontag, 1992a].

The purpose of this paper is to expose stability properties of artificial neural networks tuned to control. The neural networks used are Piecewise Affine Perceptrons (PAP), a subclass of perceptrons. It has been shown in

[Lehalle & Azencott, 1998b] that they have properties that can be used to initialize them to control a given nonlinear system. Besides they have the same useful properties as classical perceptrons (see [Lehalle & Azencott, 1998a]) : the universal approximation property and the generalization property.

The stability results given here are obtained by constructing piecewise quadratic Lyapunov functions. The basis of this kind of functions can be found for example in [Sontag, 1981] and [Johansson & Rantzer, 1997] and references therein.

This paper will at first establish a result about PAP that will be use to adapt a result about stability of piecewise affine continuous-time systems, then a similar result will be set about discrete-time ones, after that a methodology to tune PAP for control of nonlinear systems will be exposed and finaly this will be illustrated by an example : the control of an engine combustion model by a PAP.

Brevets

Ces travaux ont donné lieux à deux dépots de brevets et plusieurs plis d'huissiers par la Direction de la Recherche de Renault.

Procédé d'initialisation d'un réseau de neurones.
Brevet numéro 99 03930
Inventeurs : Charles-Albert Lehalle et Julien Piana.
L'invention concerne un procédé d'initialisation d'un réseau de neurones destiné à émuler une fonction non linéaire, tel qu'il est réalisé par une boîte d'initialisation recevant en entrée les données disponibles sur la fonction, soit un échantillon d'entrées et un échantillon de sorties, et effectuant :

- *la sélection automatique du nombre de zones linéaires nécessaires, compte-tenu des informations disponibles sur la fonction non-linéaire, pour approximer linéairement ladite fonction,*
- *les calcule automatique des formes linéaires sur chacune des zones de la fonction, soit leur pente,*
- *le conception automatisée du réseau de neurones qui associe à chaque zone de la fonction la forme linéaire correspondante,*
- *le calcul des valeurs initiales des paramètres du réseau neuronal, soit les poids et les biais de la couche de neurones cachés et de la couche de neurones de sortie.*

Extension internationale du brevet précédent.
Numéro international WO 00/58909

Système de contrôle du moteur d'un véhicule par réseaux de neurones.
Brevet numéro 97 15802
Inventeurs : Charles-Albert Lehalle et Julien Piana.
L'invention concerne un système de contrôle embarqué du fonctionnement d'un moteur thermique de véhicule comprenant un calculateur qui dérive les

commandes des actionneurs physiques du moteur pour régler certaines variables des fonctionnement autour de consignes et qui comprend :

- *un modèle au moins partiel du moteur*
- *un contrôleur constitué d'un réseau de neurones non linéaire, d'une part recevant des valeurs de consigne, des paramètres d'état du moteur, et des paramètres extérieurs, et d'autre part relié au modèle du moteur pour son adaptation aux évolutions du moteur*
- *un comparateur entre le fonctionnement réel du moteur et les consignes*