**THESE DE DOCTORAT DE L'ECOLE POLYTECHNIQUE**

Spécialité:

**Mathématiques Appliquées**

présentée par

**Alessandro FADDA**

pour obtenir le titre de **DOCTEUR DE L'ECOLE POLYTECHNIQUE**

Sujet de la thèse

# Etude de problèmes inverses par algorithmes d'évolution et réseaux de neurones

soutenue le 15 juin 1998

devant le jury composé de MM

Guy Chavent, président et rapporteur
Jean-Marc Alliot, rapporteur
François James, examinateur
Jean-Claude Nédélec, examinateur
Marc Schoenauer, directeur de thèse
Patrick Valentin, examinateur

## Remerciements

Je tiens tout d'abord à dire merci à Marc Schoenauer, mon directeur de thèse pour la confiance qu'il m'a accordé ainsi que l'ensemble des conseils qu'il m'a donné pendant toute la durée de ce travail.

Je voudrais ensuite remercier François James pour son aide en ce qui concerne la partie chromatographie.

Je remercie également l'ensemble des membres du jury pour leur disponibilité.

Je terminerais en saluant tous les chercheurs du CMAP, les trois secrétaires : Geo, Jeanne et Lilliane ainsi que tous les thésards à qui je souhaite bonne continuation.

*Forza Inter !*

# Table des matières

# Introduction

Cette thèse est consacrée à l'étude des problèmes inverses, c'est-à-dire à l'identification de fonctions qui participent à un processus dont on connaît uniquement l'état initial et l'état final. En général, il est possible d'avoir certaines informations sur cette fonction, mais elles ne sont pas suffisantes pour utiliser des moyens d'approximation classiques.
Les problèmes que nous allons étudier sont d'une part l'identification de la fonction isotherme en chromatographie et d'autre part un problème de robotique mobile, qui consiste à trouver une trajectoire réalisable par un véhicule pour se garer, à partir d'un point quelconque, vers un autre point.
Pour ces deux applications, nous utiliserons les algorithmes d'évolution qui sont des algorithmes d'optimisations stochastiques d'ordre 0 inspirés de l'évolution darwinienne. En d'autres termes, ce sont le ou les structures les mieux adaptées à un certain environnement qui survivront et réussiront à se reproduire. L'évolution a lieu à travers une succession de générations pendant lesquelles on fait évoluer une population de points de l'espace de recherche (individus), tout en appliquant des opérateurs de sélection, de mutation et de croisement afin de donner naissance à des individus toujours plus performants. Le but de ces algorithmes est de minimiser (ou maximiser) la fonction de performance ou fonction de fitness, fonction qui va de l'espace de recherche vers $\mathbb{R}$ et qui traduit la perfomance de chaque individu constituant la population. Par rapport aux méthodes déterministes qui sont basées sur l'existence de dérivées, les algorithmes d'évolution ne demandent que la connaissance de la valeur de la fitness de chaque individu de la population. De plus, ils permettent de trouver un optimum global, ce qui n'est pas le cas des méthodes classiques dont le résultat est lié au choix des conditions initiales et qui donc ne s'appliquent que localement.

La première application consiste à identifier la fonction isotherme en chromatographie. La chromatographie est un processus physique qui a pour but la séparation des divers composants constituant un mélange chimique. Le principe consiste à injecter le mélange dans un tube de grande longueur et de faible diamètre empli d'un solide poreux que l'on appelle colonne. Chaque composant entraîné par un fluide vecteur, va passer d'une phase mobile correspondant au déplacement des composants à une phase stationnaire pendant laquelle il est retenu par le milieu poreux et dont la durée varie en fonction de chaque corps constituant le mélange. Les composants vont donc se propager le long de la colonne à des vitesses différentes, ce qui entraînera leur separation. En sortie de colonne, on constatera une variation de concentration à la sortie de chacun des composants chimiques et qu'on appelle chromatogramme.
La fonction isotherme, qui intervient dans le modèle mathématique de propagation, est une fonction de $\mathbb{R}^M$ vers $\mathbb{R}^M$ ($M$ est le nombre de corps constituant le mélange) qui traduit l'état

d'équilibre isotherme (c'est-à-dire à température constante) entre la phase mobile et la phase stationnaire.
Connaissant les conditions expérimentales (durée et vitesse d'injection du mélange, état initial de la colonne, quantité injectée dans la colonne) et la fonction isotherme, le modèle de propagation nous permet d'obtenir les chromatogrammes (problème direct). Le problème inverse consiste alors, à partir des conditions expérimentales et des chromatogrammes expérimentaux, à retrouver la fonction isotherme. Pour cela, on utilisera plusieurs méthodes évolutionnaires. La performance d'un individu sera alors le résultat de la comparaison entre un chromatogramme expérimental et le chromatogramme obtenu en résolvant le problème direct. Ce problème direct sera quand à lui résolu grâce aux conditions expérimentales et en utilisant l'individu comme fonction isotherme. Plusieurs modèles d'isotherme seront utilisés comme les réseaux de neurones, les fractions rationnelles couplées ou non avec les réseaux de neurones ou encore les modèles d'isotherme existants comme les isothermes de Langmuir ou de Moreau-Valentin.

La deuxième application est un problème test de contrôle non-linéaire. A partir d'une position quelconque et en contrôlant uniquement le braquage des roues, un réseau de neurones doit être capable de garer un véhicule vers un endroit quelconque. Pour cela, à chaque pas de temps de notre simulation, le réseau devra donner l'angle de braquage des roues connaissant la position du véhicule ainsi que son orientation, ce qui permettra de calculer la nouvelle position de la voiture et par la suite de calculer la trajectoire qu'effectue un réseau "au volant" d'un véhicule.
L'algorithme d'évolution fera donc évoluer une population de réseaux, son but étant d'optimiser l'ensemble des paramètres de celui-ci. On obtiendra par ailleurs la performance de chaque réseau en calculant la distance minimum entre la trajectoire qu'il effectue et le but fixé.

Pour résoudre ces deux problèmes, nous avons mis au point un algorithme d'évolution inspiré de l'algorithme GNARL de la programmation évolutionnaire avec cependant des différences importantes liées à l'utilisation de nouveaux opérateurs de mutations, de sélection et surtout une stratégie d'application des mutations qui consiste à adapter pour chaque réseau la mutation qui sera la plus efficace à un moment donné de l'évolution.

Les deux premiers chapitres de cette thèse présentent les réseaux de neurones et les algorithmes d'évolution et, en particulier, l'algorithme que nous avons utilisé ainsi que son adapatation pour l'optimisation paramétrique. Ensuite, nous présenterons dans les chapitres trois et quatre l'ensemble des résultats obtenus. Nous terminerons cette thèse par un résultat théorique sur l'adaptation d'une méthode de Newton et par une méthode de contination pour le calcul d'un point fixe d'un réseau de neurones récurrent.

# Chapitre 1

# Les réseaux de neurones

## 1.1 Introduction

Les réseaux de neurones mathématiques sont une modélisation simpliste des réseaux de neurones biologiques. Le but d'une telle modélisation est de retrouver certaines propriétés intéressantes du cerveau humain comme par exemple la capacité d'apprentissage, de mémorisation ou celle de traiter des informations floues, bruitées ou incomplètes ainsi que son caractère hautement parallèle (tous les neurones agissent simultanément). Du point de vue mathématique, un réseau de neurones est une application de $\mathbb{R}^n$ vers $\mathbb{R}^m$ que l'on peut représenter par un graphe orienté, les sommets étant les neurones et les arêtes les connexions entre neurones. Le réglage des différents paramètres d'un réseau afin d'obtenir un ensemble de réponses en fonction d'un ensemble d'entrées constitue la notion d'apprentissage. Si un réseau a été correctement entraîné, il donnera une réponse correcte non seulement sur les cas qu'il a appris mais aussi pour d'autres cas ne faisant pas partie de son entraînement : c'est la capacité de généralisation. Les réseaux de neurones peuvent ainsi être utilisés entre autres pour des problèmes de mémoires associatives, de reconnaissance de caractères, de prédictions de séries temporelles ou encore les problèmes d'identifications de fonctions et les problèmes inverses qui font l'objet de cette thèse.
Après avoir présenté les réseaux de neurones, nous étudierons un cas particulier de calcul qui est l'adaptation d'une méthode de Newton pour le calcul d'une suite de points fixes d'un réseau de neurones récurrent. La méthode d'apprentissage que nous avons utilisée fera l'objet entre autres du chapitre suivant.

## 1.2 Les neurones artificiels

Un neurone artificiel [MP43] est une unité de calcul formée de plusieurs connexions rentrantes constituant l'entrée du neurone et d'une connexion sortante constituant la réponse du neurone à une certaine stimulation. A chaque connexion rentrante est associée un poids (appartenant à $\mathbb{R}$ ) qui représente la mémoire locale du neurone. De manière générale, la réponse du neurone est calculée en appliquant une fonction dite de transfert à la somme des valeurs d'entrée multipliée par les poids des connexions (voir figue 1.1).

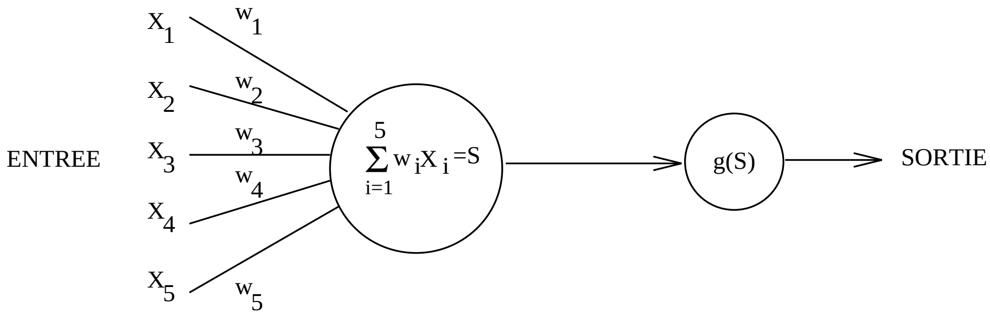



Figure 1.1: Un neurone artificiel

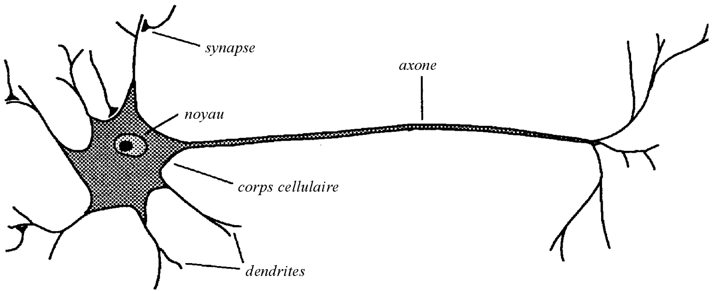



Figure 1.2: Schéma d'un neurone

Comme l'indique la figure 1.1, la réponse $\{O_i\}_{i=1}^N$ d'un neurone pour une certaine stimulation ou entrée $\{X_i\}_{i=1}^N$ est $O_i = g(\sum_{i=1}^N w_i W_i)$ avec $\{w_i\}_{i=1}^N$ les poids des connexions et $g$ est la fonction de transfert. En général, on utilisera une fonction de transfert dite sigmoïde $g_\beta(x) = \frac{1}{1+e^{-\beta x}}$, avec $\beta \in \mathbb{R}$ (voir la figure 1.3) ce qui implique que l'activité d'un réseau appartient à l'intervalle $[-1, +1]$. .

## 1.3 Les réseaux de neurones

La réponse d'un neurone peut constituer l'entrée d'autres neurones. Des neurones ainsi liés forment un réseau de neurones qui comporte une entrée formée de un ou plusieurs neurones (neurones d'entrées) et une sortie formée aussi de un ou plusieurs neurones (neurones de sorties) et dont l'état est la réponse du réseau à une stimulation externe. Les neurones n'appartenant à aucune de ces deux catégories sont appelés neurones cachés ou intermédiaires. On peut associer au réseau un graphe orienté dont les noeuds sont les neurones et les arêtes les connexions entre neurones. Lorsque le graphe des connexions correspondant ne comporte aucun cycle on parle de réseau *feed–forward* sinon on parle de réseau *récurrent*. Les figures 1.4 et 1.5 montrent un exemple de ces deux types de réseaux.
L'ajustement de l'ensemble des paramètres d'un réseau que sont les poids des connexions, la fonction de transfert ou la topologie du réseau, afin d'obtenir certaines sorties en fonction de certaines entrées constitue la notion d'apprentissage. Il existe plusieurs types d'algorithmes effectuant

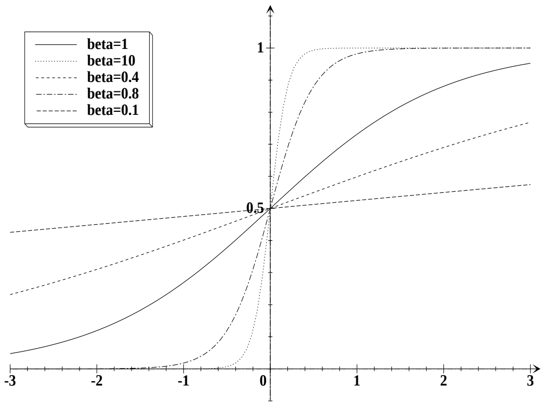



Figure 1.3: Graphe de $g_\beta$ pour plusieurs valeurs de $\beta$

cette tâche, comme l'algorithme de retro–propagation ou encore les algorithmes d'évolution que nous présenterons au chapitre suivant. L'algorithme de rétro–propagation [RHW86, LC87] est une méthode de descente de gradient stochastique et permet de résoudre de nombreux problèmes d'apprentissage. Les limites principales de cette méthode sont qu'elle dépend fortement de l'initialisation du réseau comme c'est le cas pour toutes les méthodes de gradient et elle agit à topologie du réseau fixé ce qui peut parfois être pénalisant. Par ailleurs, il faut connaître un ensemble de réponses du réseau (par rapport à certaines entrées) pour pouvoir ajuster l'ensemble de ses poids. Cela implique que cette méthode ne peut être appliqué pour résoudre certains problèmes comme les problèmes inverses où la fonction à identifier intervient dans un processus dont on connaît uniquement le résultat. Une alternative consiste alors à utiliser un algorithme évolutionnaire comme cela sera le cas pour l'identification de la fonction isotherme en chromatographie présentée au chapitre 3. Il existe cependant certains problèmes inverses où il est possible d'adapter la méthode de retro-propagation, comme le problème de contrôle étudié au chapitre 4 ([NW90]).

## 1.4 Les réseaux de neurones multi–couches

Comme leur nom l'indique ces réseaux sont organisés en couches de neurones, la première constituant l'entrée du réseau et la dernière la sortie. Chaque neurone d'une certaine couche “reçoit” des connexions de la couche précédente et “envoie” des connexions vers la couche suivante. Ce type de réseau est un cas particulier de réseau feed–forward pour lequel la seule condition est qu'il ne contienne aucun cycle dans son graphe des connexions. Cependant en ajoutant des neurones identités (sortie = entrée) on peut obtenir un réseau multi–couches à partir d'un réseau feed–forward.

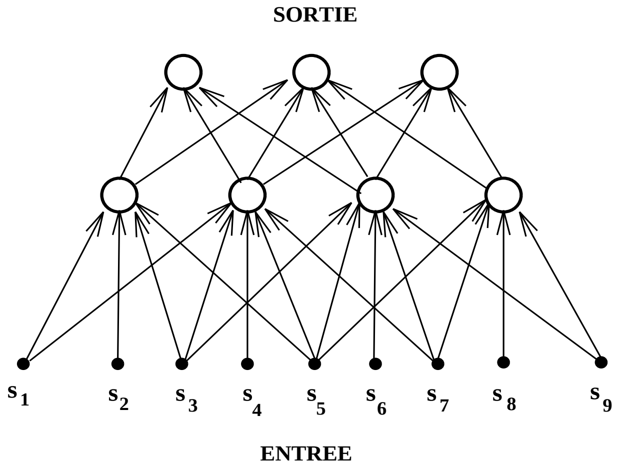



Figure 1.4: Réseau de neurones à deux couches comportant 9 entrées et 2 sorties

Le calcul de la réponse se fait couche par couche en ce sens que chaque neurone d'une certaine couche aura pour entrée la réponse d'un neurone de la couche précédente. Ainsi, connaissant les valeurs d'entrées du réseau, on calcule les réponses de tous les neurones de la première couche ce qui va permettre le calcul des couches supérieures et donc le calcul de la réponse du réseau. Si $\{s_i\}_{i=1}^N$ est l'entrée du réseau alors la réponse $\{x_i^k\}$ d'un neurone $n_i$ de la couche $k$ sera :

$$\begin{cases} x_i^k = g(\sum_j w_{ij} x_j^{k-1}) \\ x_i^0 = s_i \end{cases}$$

avec $w_{ij}$ le poids de la connexion entre le neurone $n_j$ et le neurone $n_i$ (connexion de $n_j$ vers $n_i$).

## 1.5 Capacité d'approximation des réseaux de neurones en couches

Si un réseau de neurones multi-couches possède $k$ (resp. $l$) neurones d'entrées (resp. sortie), alors on peut considérer ce réseau comme une application de $\mathbb{R}^k$ vers $\mathbb{R}^l$.

Citons tout d'abord le théorème de Kolmogorov [HN89] qui dit que toute fonction continue de $\mathbb{R}^k$ vers $\mathbb{R}^l$ peut s'écrire comme un réseau de neurones à deux couches qui utilise $l+1$ fonctions de transfert différentes. Ce théorème ne donne cependant pas d'indications sur ces fonctions de transfert.

**Théorème 1** *Soit une fonction $f : [0,1]^k \to \mathbb{R}^l$ continue. Alors on peut trouver un réseau de neurones à deux couches comportant $k$ entrées, $2k+1$ neurones dans la première couche et $l$ neurones de sortie et qui est égal à $f$.*

L'état des neurones de la première couche est calculée de la manière suivante :

$$z_i = \sum_{j=1}^{k} \lambda^i \Psi(x_j + i\epsilon) + i$$

où $\lambda \in RR$ et la fonction $\Psi$ sont indépendant de $f$ (mais dépendent de $k$) et la constante $\epsilon$ est un nombre rationnel positif.

La sortie du réseau est

$$y_j = \sum_{i=1}^{2k+1} g_j(z_i)$$

où les $g_j, j = 1, ..., l$ sont des fonctions réelles continues dépendant de $f$ et $\epsilon$.

Le théorème suivant dit que les réseaux de neurones multi-couches sont capables d'approcher toutes les fonctions en ce sens que l'espace des fonctions défini par un réseau est dense dans l'espace $L^p(\mu)$ [H90] ou [W90]:
Soit $\mathcal{R}_k(\Psi)$ l'ensemble des fonctions calculables par un réseau à deux couches utilisant la fonction de transfert $\Psi$ et comportant $k$ entrées, 1 sortie et un nombre quelconque de neurones cachés.

**Théorème 2** *Si $\Psi$ est bornée et non constante, alors $\mathcal{R}_k(\Psi)$ est dense dans l'espace $L^p(\mu)$ ($1 \leq p < \infty$) pour toute mesure finie $\mu$ de $\mathbb{R}^k$ i.e. $\forall f \in L^p(\mu)$ et $\forall \varepsilon > 0$, $\exists g \in \mathcal{R}_k(\Psi)$ tel que*

$$\left[ \int_{\mathbb{R}^k} |f(x) - g(x)|^p d\mu(x) \right]^{1/p} < \varepsilon$$

Le cas correspondant à plusieurs couches et plusieurs unités de sorties se déduit grâce aux corollaires 2.6 et 2.7 dans [HSW89]. Par ailleurs, si on rajoute l'hypothèse $\Psi$ continue, on a alors le théorème suivant [H90]:

**Théorème 3** *Si $\Psi$ est continue, bornée et non constante, alors $\mathcal{R}_k(\Psi)$ est dense dans $C(X)$ espace des fonctions continues sur $X \subset \mathbb{R}^k$ pour tout compact $X$ de $\mathbb{R}^k$.*

On trouvera la démonstration de ces théorèmes dans [HSW89] et [H90].

## 1.6 Les réseaux de neurones récurrents

Les réseaux de neurones récurrents possèdent au moins un cycle dans leur graphe de connexions.

Par rapport aux réseaux feed–forward, les réseaux de neurones récurrents (qui en sont un sur-ensemble) ont un éventail de comportement plus large qui les rend peut-être mieux adaptés pour certains problèmes.
On peut utiliser les réseaux de neurones récurrents de deux manières : dynamiquement pour l'approximation de fonctions dépendant du temps et la prédiction de séries temporelles; ou considérer la réponse du réseau comme un état d'équilibre atteint après stabilisation (étude asymptotique). Etant donné que les fonctions que nous voulons identifier ne dépendent pas du temps c'est la deuxième approche qui a été choisie. Le calcul de la réponse du réseau n'est alors pas triviale : l'étude d'un réseau de neurones récurrent dans un tel contexte est alors plus complexe que dans le cas d'un réseau feed–forward et relève alors de l'étude des systèmes dynamiques (existence de point(s) fixe(s), étude de stabilité). Cela entraîne une augmentation de la complexité des calculs qui peut s'avérer fortement pénalisant sur certains problèmes.
Le paragraphe suivant expose une méthode générale de calcul de l'attracteur lorsqu'il existe ainsi qu'une étude sur l'adaptation d'une méthode de Newton sur un cas particulier de calcul.

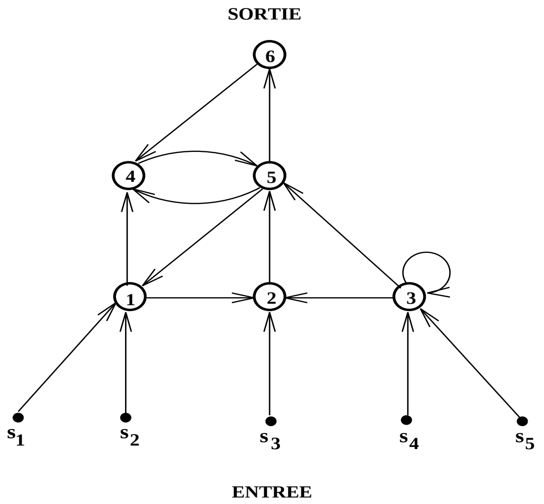



Figure 1.5: Réseau de neurones récurrent

## 1.7 Calcul d'un point fixe d'un réseau de neurone récurrent

L' évolution en temps d'un réseau de neurones récurrent de taille $N$ peut être modélisée par l'équation différentielle

$$\frac{dy}{dt} = -y + \sigma(W, y, S) \tag{1.1}$$

où $y = \{y_i\}_{i=1}^N$ est le vecteur des activations, $W = \{w_{ij}\}_{i,j=1,\dots,N}$ est la matrice des poids du réseau, $\sigma$ est la fonction sigmoïde dont le ou les paramètres peuvent dépendre de chaque neurone et $S = \{s_i\}_{i=1}^N$ avec $s_i = 0$ si le neurone $i$ n'est pas un neurone d'entrée.

Cette équation peut être discrétisée par un schéma d'Euler, ce qui donne

$$\begin{cases} y^{n+1} = (1 - \Delta t)y^n + \Delta t\sigma(W, y^n, S) \\ y^0 = 0 \end{cases} \tag{1.2}$$

- La méthode couramment utilisée pour le calcul d'un point fixe est la méthode des approximations successives : en notant $y_i^n$ l'activation du neurone $i$ à l'étape n et en supposant tous les neurones inactifs à l'étape 0 ($y_i^0 = 0, \quad \forall i = 1, .., N$), elle consiste à calculer les $y_i^n$ avec

$$\begin{cases} y_i^{n+1} = (1 - \Delta t)y_i^n + \Delta t\sigma(W, y^n, S) \quad \forall i = 1, .., N \\ y_i^0 = 0 \quad \forall i = 1, .., N \end{cases} \tag{1.3}$$

On définit a sortie du réseau a la précision $P$ par

$$\left|1 - \frac{y_i^N}{y_i^{N-1}}\right| \leq P, \quad \forall i = 1, ..., N \tag{1.4}$$

La valeur de $P$ dépend de la précision recherchée et on peut considérer que seuls les neurones de sortie ont besoin de vérifier (1.4).
Si on écrit (1.3) sous la forme $y^{n+1} = f(y^n)$ et si la matrice jacobienne de $f$ possède uniquement des valeurs propres de modules inférieur à 1 quelque soit les états $y^n$ alors on convergera vers un point d'équilibre unique pour une entrée $S$ fixée.

- Une autre méthode, présentée au chapitre 5, donne un résultat théorique sur une méthode pour calculer un point fixe en utilisant uniquement un algorithme de Newton.

## 1.8 Cas particulier: calcul d'une suite de points fixes

Nous nous intéressons au problème suivant qui consiste à calculer un nombre assez élevé de points fixes lorsque les valeurs d'entrées successives sont proches (comme pour le problème de la chromatographie présenté au chapitre 3). Afin de réduire les temps de calculs (par une diminution du nombre d'itérations), nous ne ré-initialisons pas les valeurs des activations des neurones à zéro pour le calcul suivant. Par exemple, si on doit calculer $P$ points fixes correspondant à $P$ valeurs d'entrées, on calcule le premier point fixe normalement. Les autres sont calculés en ne ré-initialisant pas les valeurs des activations des neurones (sauf les neurones d'entrées). Cependant, si la norme de la matrice jacobienne est plus grande que 1 alors le résultat qu'on obtient va dépendre de l'état initial.
Pour effecuer ce type de calcul on peut alors soit utiliser la méthode des itérations successives soit utiliser un algorithme de Newton qui convergera plus rapidement (en nombre d'itérations).

### 1.8.1 Utilisation de la méthode de Newton

On sait que la méthode de Newton construit une suite de points $\{x_i\}_{i\geq 1}$ tels que

$$\begin{aligned} x_n &= x_{n-1} - Df(x_{n-1})^{-1} f(x_{n-1}) \\ &= N_f(x_{n-1}) \\ &= N_f^n(x_0) \end{aligned}$$

la suite de Newton $x_0, x_1, x_2, ...$ est définie et converge vers $\zeta$ avec $f(\zeta) = 0$ pourve que $x_0$ soit "assez proche" de $\zeta$ (voir chapite 5).

L'adaptation d'une telle méthode pour les réseaux de neurones récurrents est simple :

si l'on pose

$$\Phi(y) = y - \sigma(W, y, S)$$

alors un point fixe $y^* = \{y_i^*\}_{i=1}^N$ vérifie $\Phi(y^*) = 0$ et il suffit d'appliquer l'algorithme de Newton à la fonction $\Phi$ après avoir calculé $x_0$ par une méthode classique.
Pour savoir si ce point fixe est stable, il faut que les valeurs propres de la matrice jacobienne de la fonction $I - \Delta t \Phi$ au point $y^*$ soient toutes à l'intérieur du cercle unité.

### 1.8.2 Etude numérique

Nous présentons dans ce paragraphe une étude numérique comparative entre la méthode de Newton exposée ci-dessus et trois autres méthodes obtenues en utilisant trois valeurs différentes de $\Delta t$ dans 1.2 ($\Delta t = 0.5, 0.75, 1$) que l'on appellera respectivement *Euler 0.5*, *Euler 0.75* et *Euler 1*.

Avant de présenter les résultats nous faisons quelques précisions et remarques.

- Nous effectuons le test pour des réseaux comportant une entrée une sortie et **h** neurones cachés.

- On définit la densité de connexion **d** qui indique indépendamment du nombre de neurones le nombre total de connexions d'un réseaux. Cette échelle varie de 1 à 10, 1 signifiant que chaque neurone est connecté à au moins un autre neurone et 10 signifiant que chaque neurone est connecté à (**h** + le nombre de neurones de sortie) autres neurones (certains pouvant être les mêmes).

- Nous appellons parcours une suite de points appartenant à [0,1] et constituant l'entrée d'un réseau quelconque et PAS la différence en valeur absolue entre deux valeurs consécutives du parcours. On considérera quatre type de parcours : PAS$\in [10^{-5}, 10^{-4}]$, PAS $\in [10^{-4}, 10^{-3}]$, PAS$\in [10^{-3}, 10^{-2}]$ et PAS $\in [10^{-2}, 10^{-1}]$.

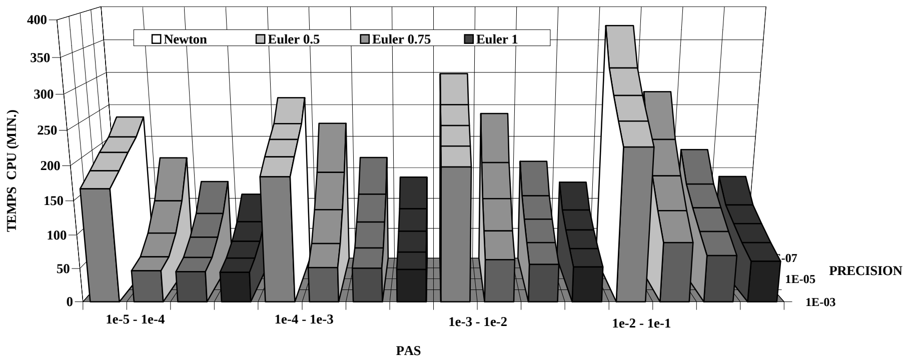
- La PRECISION du calcul est la valeur de $P$ dans 1.4. On considérera cinq valeurs de P : $10^{-7}$, $10^{-6}$, $10^{-5}$, $10^{-4}$ et $10^{-3}$.

- On calcule les points fixes d'un certains nombre de réseaux sur chaque parcours pour chaque valeur de PRECISION et pour les quatre méthodes en fixant **h** et **d**.

- On appelle pourcentage de réussite PER d'une méthode le pourcentage de réseaux pour lesquels on a réussi à calculer tous les points fixes d'un parcours lorsque la PRECISION,**h** et **d** sont fixés.

- Le temps total de calcul $T$ d'une méthode (la PRECISION, **h** et **d** sont fixés) est le temps CPU nécessaire pour calculer l'ensemble des points fixes d'un parcours pour tous les réseaux divisé par PER.

Nous avons effectué plusieurs calculs pour des valeurs de **h** appartenant à l'intervalle $[1, 5]$, $[6, 10]$ et $[11, 20]$ pour des valeurs de **d** appartenant à l'intervalle $[1, 3]$, $[4, 7]$ et $[8, 10]$ et sur des parcours de 1000 points. Le nombre de réseaux testés s'élève en tout à 12000 pour $\mathbf{h} \in [1, 20]$ et $\mathbf{d} \in [1, 10]$. Nous présentons dans les figures 1.6 et 1.7 l'ensemble des résultats significatifs.



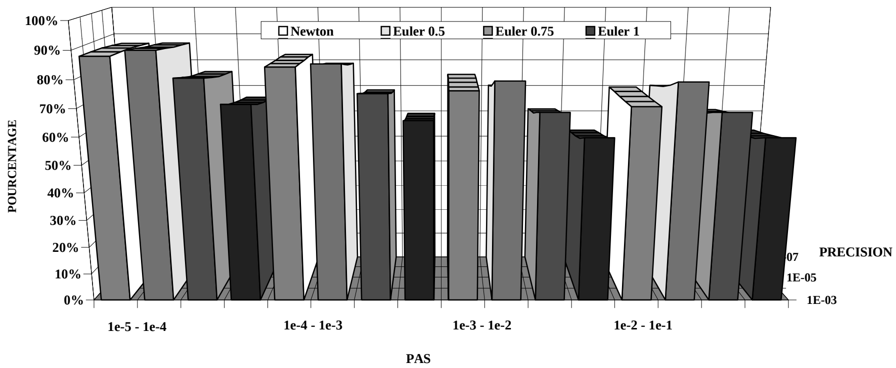
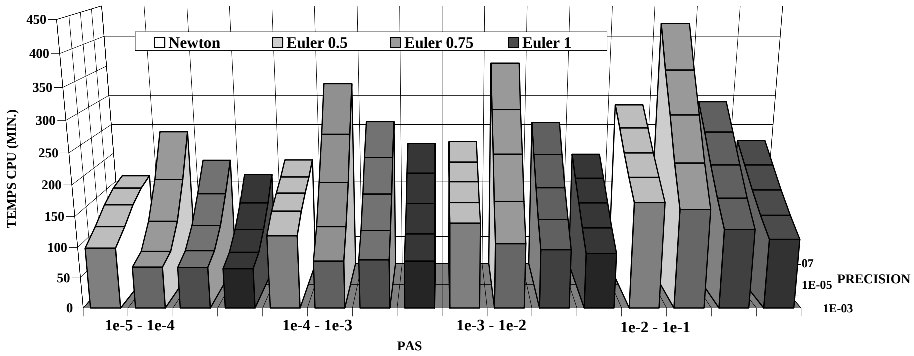




Figure 1.6: *Temps total de calcul $T$ (en haut) et pourcentage de réussite PER (en bas) pour les quatre méthodes pour 12000 réseaux vérifiant* $\mathbf{h} \in [1, 20]$ *et* $\mathbf{d} \in [1, 10]$.



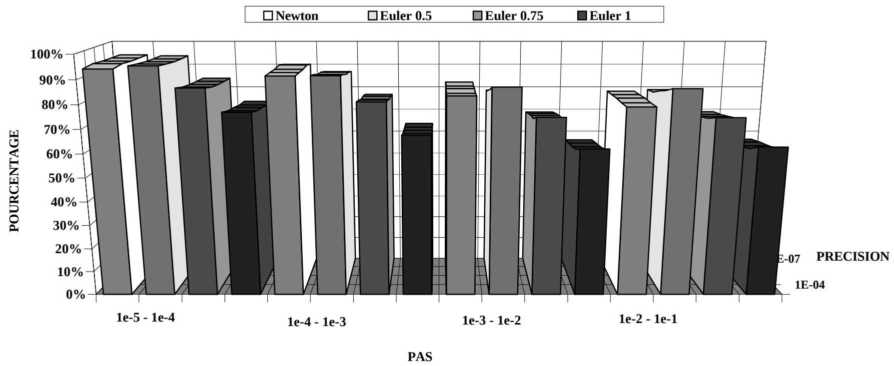




Figure 1.7: *Temps total de calcul $T$ (en haut) et pourcentage de réussite PER (en bas) pour les 4 méthodes pour 1000 réseaux vérifiant* $\mathbf{h} \in [1, 5]$ *et* $\mathbf{d} \in [8, 10]$.

Comme on peut le constater, la méthode de Newton est en générale beaucoup plus coûteuse que les trois autres méthodes. En effet, de nombreuses conditions doivent être réunies pour seulement obtenir des résultats comparables avec les méthodes plus classiques : le nombre de neurones cachés doit être assez petit (inférieur à 5); la PRECISION doit être de $10^{-7}$ ou $10^{-6}$ et la densité assez élevée. Toutes ces conditions impliquent un nombre d'itérations élevés pour les autres méthodes et un coût relativement faible pour une itération de la méthode de Newton. Cependant, le fait que le nombre d'itérations soit souvent largement inférieure aux méthodes classiques ne compense que rarement le coût beaucoup plus plus élevé d'une itération de Newton. Dans toute la suite, lorsque l'on devra calculer la sortie d'un réseau de neurones récurrent, on utilisera la méthode *Euler 1*. De plus, lorsque plusieurs points fixes devront être calculés (voir le paragraphe 1.8), on ne ré-initialisera pas chaque neurone. On a vu que cette méthode pouvait entraîner une différence de résultat pour une même entrée. Cependant, on espère que de tels réseaux seront éliminer par la sélection (voir chapitre suivant).

# Chapitre 2

# Les algorithmes d'évolution

## 2.1 Introduction

Les algorithmes d'évolution sont des algorithmes d'optimisations stochastiques inspirés de l'évolution darwinienne en ce sens que ce sont le ou les individus les mieux adaptés à un certain environnement qui survivront et réussiront à se reproduire.
Du point de vue mathématique, c'est la fonction de performance ou fonction de fitness qui traduit le degré d'adaptation d'un individu (ou point de l'espace de recherche). Il existe plusieurs catégories d'algorithmes d'évolution; cependant leur but est le même, à savoir trouver un individu qui minimise cette fonction. Pour cela, on fait évoluer une population de points de l'espace de recherche pendant un certain nombre de générations tout en appliquant des opérateurs de sélection, de mutation et de croisement afin de donner naissance à des individus toujours plus performants et aussi d'éliminer les plus mauvais.
Un des avantages des algorithmes d'évolution est que ce sont des méthodes d'ordre 0 : contrairement aux méthodes déterministes de type gradient par exemple, qui sont basées sur l'existence de dérivées, ils n'ont pas besoin de connaître des informations autres que la valeur de la fitness de chaque point de la population, ce qui leur permet donc de traiter un plus grand nombre de problèmes. Par ailleurs, leur succès n'est pas lié à l'initialisation de la population, ce qui n'est pas le cas des méthodes déterministes pour lesquels le choix du point de départ est primordial. Autrement dit, les algorithmes d'évolution s'appliquent globalement par opposition aux méthodes déterministes qui ne s'appliquent que localement. Cependant, ces algorithmes sont souvent assez coûteux en temps de calcul. De plus, leur nature stochastique implique qu'il faut lancer plusieur calculs afin d'arriver à un minimum satisfaisant et il existe peu de résultats théoriques expoitables sur la convergence de ces algorithmes.
Dans ce chapitre nous parlerons essentiellement des deux types d'algorithmes que nous avons utilisé : la programmation evolutionnaire (Evolutionary Programming ou EP) et les stratégies d'évolution (Evolution Strategies ou ES). Nous présentons dans les paragraphes suivant ces deux types d'algorithmes ainsi que notre algorithme inspiré de l'algorithme GNARL [A94, A93].

## 2.2 Les algorithmes d'évolution

Le but des algorithmes d'évolution est de minimiser la fonction de *performance* ou de *fitness*, fonction qui va de l'espace de recherche vers $\mathbb{R}$ . Pour cela, on considère une population de taille

$S$ de points de l'espace de recherche (encore appelés *individus* et on évalue leur fitness. A chaque génération on effectue les tâches suivantes :

- On sélectionne un ensemble de points appelés aussi individus en fonction de leur fitness afin d'avantager les meilleurs. Cet ensemble d'individus est copié dans la population intermédiaire.
- On applique à cette population les opérateurs génétiques de mutation et de croisement qui ont pour but l'exploration de l'espace de recherche afin de trouver si possible de meilleurs individus. Le but du croisement est de recombiner plusieurs points de l'espace de recherche afin d'en former d'autres (en général deux individus donnent naissance à deux autres). La mutation est un déplacement aléatoire dans l'espace de recherche.
- On évalue la performance de ces nouveaux individus.
- On choisit parmis tous les individus ceux qui feront partie de la nouvelle population qui sera présentée à la prochaine génération.

On fait ainsi évoluer la population au fil des générations jusqu'à ce que l'on considère qu'un individu vérifie plus ou moins bien un ou plusieurs critère.

Dans cette thèse les résultats presentés ont été trouvés en appliquant deux types d'algorithmes d'évolution : les algorithmes de stratégie d'évolution (Evolution strategies ou ES) et la programmation évolutionaire (Evolutionary programming, EP) dont l'algorithme GNARL fait partie et qui a inspiré notre algorithme.

### 2.2.1 Les stratégies d'évolution

Les stratégies d'évolution qui ont été initialement crées par I. Rechenberg [Rec73] et Schwefel [Sch81] traitent les problèmes d'optimisation paramétrique. Plus récemment Bäck [Bä95] y a apporté une contribution entre autres par des nouvelles stratégies de sélection ainsi que différents types de mutations.
Le schéma de base de ces algorithmes consiste à appliquer sur chaque individus (qui sont des vecteurs appartenant à $\mathbb{R}^N$) d'une population de taille $\mu$ une mutation gaussienne : ajout d'une variable aléatoire gaussienne centrée en zéro et de variance $\sigma$ (ajustée au cours de l'évolution). Ensuite, les $\mu$ meilleurs parmis les parents et ces nouveaux individus sont choisis pour faire partie de la population qui constituera la population de la génération suivante. D'autres stratégies plus récemment développées consistent à créer à partir de $\mu$ parents $\lambda$ enfants (chaque parent créant $\lambda/\mu$ enfants avec $\lambda > \mu$) et de garder pour la génération suivante les meilleurs $\mu$ individus parmi les $\lambda$ crées (stratégie $(\mu,\lambda)$-ES) ou bien parmis les $\lambda + \mu$ (stratégie $(\mu + \lambda)$-ES). On remarquera que le fait de choisir les meilleurs $\mu$ individus parmis les $\lambda$ enfants rend la stratégie $(\mu,\lambda)$-ES non élitiste, c'est-a-dire que le meilleur individu d'une génération n'apparaît plus a la génération suivante et donc la fitness peut augmenter, ce qui n'est pas le cas pour la stratégie $(\mu + \lambda)$-ES. Par ailleurs, on peut utiliser la mutation adaptative qui considère que la variance $\sigma$ de la variable aléatoire gaussienne fait partie des paramètres de chaque individu et est donc aussi ajustée au cours de l'évolution.

- Un premier choix consiste à fixer définitivement au début de l'évolution la variance $\sigma$. Si on note $X \in \mathbb{R}^N$ un individu la mutation aura pour effet de créer l'individus $\tilde{X}$ avec

$$\begin{cases} \tilde{X} = X + N(0, \sigma) \\ \sigma \text{ fixé} \end{cases}$$

  et $N(\mu, \sigma)$ est la variable aléatoire gaussienne de variance $\sigma$.

- La règle dite des 1/5 [Rec73] consiste à augmenter multiplicativement la valeur de la variance $\sigma$ si au moins 1/5 des mutations produisent de meilleurs individus ou bien à diminuer cette valeur dans le cas contraire.

- Un premier type de mutation adaptative considère la variance $\sigma$ comme faisant partie de l'individu et est donc un paramètre qu'il faut régler.

$$\begin{cases} \tilde{\sigma} = \sigma \cdot e^{\tau N(0,1)} \\ \tilde{X} = X + N(0, \tilde{\sigma}) \end{cases}$$

  et $\tau$ est un paramètre que l'on fixe en fonction de N (dimension de l'espace de recherche) au début de l'évolution.

- Le même raisonement a été fait pour ce deuxième type de mutation adaptative mais ici chacune des variable $X_i$ de $X (= \{X_i\}_{i=1}^N)$ possède son propre $\sigma_i$.

$$\begin{cases} \tilde{\sigma_i} = \sigma_i \cdot e^{\tau N(0,1) + \tau' N(0,1)} & \forall i = 1, ..., N \\ \tilde{X_i} = X_i + N(0, \tilde{\sigma_i}) \quad \forall i = 1, ..., N \end{cases}$$

  et $\tau$ et $\tau'$ sont des paramètres que l'on fixe en fonction de N au début de l'évolution.

- Le cas le plus général de la mutation adaptative consiste à utiliser une matrice de covariance symétrique ce qui conduit à régler $\frac{N(N+1)}{2}$ paramètres supplémentaires.

### 2.2.2 La programmation évolutionnaire

Les algorithmes de programmation évolutionnaire ont été développés par L. J. Fogel [FOW66] en Californie pendant les années 60. Ces algorithmes se distinguent des autres types d'algorithmes d'évolution par le fait qu'ils font généralement évoluer des populations d'individus ayant une structure quelconque par opposition aux algorithmes génétiques ([Hol75] et [Gol89]) qui travaillent essentiellement sur des représentations binaires de ces structures. Ainsi, on peut dire que les algorithmes EP agissent sur l'espace des phénotypes alors que les autres agissent sur l'espace des génotypes. Cela entraîne que l'opérateur de croisement est jugé inefficace car la structure qui évolue est considérée comme un tout et non pas comme un ensemble de sous-structures qu'il est possible d'échanger entre individus ([Fog95]). De manière générale, chaque individu d'une population de taille S subit une mutation dont la sévérité est liée à sa fitness. Ensuite, les S meilleurs parmis les parents et les enfants sont choisis pour faire partie de la population à la génération suivante. D'autres mécanismes de sélection peuvent être utilisés sur les 2S individus comme la sélection par tournoi mais la sélection reste toujours élististe.

Dans cette thèse, nous avons utilisé un algorithme d'évolution dérivé de l'algorithme GNARL [A94] pour faire évoluer des réseaux de neurones récurrents ou feed-forward. Après avoir rappelé les caractéristiques principales de cet algorithme, nous présenterons en détail les modifications que nous y avons apporté.

### 2.2.3 L'algorithme GNARL

Cet algorithme qui est décrit dans [A94] et [A93] a pour but de déterminer la structure ainsi que tous les poids d'un réseau de neurones récurrents afin de résoudre une certaine tâche. Le principe de cet algorithme est le suivant :
Une population de $N$ réseaux évolue pendant un certain nombre de générations. A chaque génération, les $N/2$ meilleurs réseaux subissent une mutation (dont la sévérité dépend de sa fitness) alors que les autres sont retirés. La population de la génération suivante est constituée de ces $N/2$ réseaux ayant subit une mutation ainsi que de leurs parents. Deux catégories de mutations sont utilisées dans cet algorithme : les mutations architecturales qui modifient la structure du réseau et les mutations paramétriques qui modifie les poids des connexions.
La différence principale entre l'algorithme GNARL et celui utilisé ici réside principalement dans l'application des différentes mutations disponibles. En effet, en ce qui concerne notre algorithme (que nous appellerons algorithme A), on utilise une stratégie d'application des mutations. De plus on introduit de nouvelles mutations dont les effets seront discutés plus loin dans ce chapitre.

Notre algorithme effectue les tâches suivantes :

- Création de la population initiale ;
- Evaluation de la performance (ou fitness) de chaque individu ;
- Création de la population intermédiaire ou sélection;
- Application des opérateurs de mutations sur la population intermédiaire.

Tout comme l'algorithme GNARL ainsi que tous les autres algorithmes de programmation évolutionnaire, nous n'avons pas introduit un opérateur de croisement. En effet, il nous semble impossible de trouver un tel opérateur pour un réseau de neurones récurrent ou feed-forward qui soit sémantiquement significatif, c'est-à-dire qui ait la valeur d'un croisement et non pas d'une mutation comme cela serait le cas si on échangeait une partie d'un réseau. Un croisement a un sens si il agit sur l'espace des génotypes et non pas sur celui des phénotypes car un point de ce dernier espace constitue un tout alors qu'un point appartenant à l'espace des génotypes est formés à partir d'un ensemble de sous–structures de bases permettant l'utilisation efficace de l'opérateur de croisement.

## 2.3 L'algorithme A

### 2.3.1 Création de la population initiale

La population initiale est créée aléatoirement en respectant les possibles contraintes pouvant exister et communes à tous les individus. Ainsi, le nombre de neurones d'entrée et le nombre de neurones de sortie sont fixés à l'avance et ne varient pas au cours de l'évolution. Par ailleurs, le nombre de neurones cachés est choisi aléatoirement et uniformément sur un intervalle défini auparavant par l'utilisateur. Il en est de même en ce qui concerne le nombre de connexions totales de chaque réseau : la densité de connexions indique, indépendamment du nombre de neurones, le nombre total de connexions d'un réseau. Cette échelle varie de 1 à 10, 1 signifiant que chaque neurone est connecté à au moins un autre neurone et 10 signifiant que chaque neurone est connecté à (nombre de neurones cachés + nombre de neurones de sortie) autres neurones (certains pouvant être les mêmes). La densité pour chaque réseau est tirée aléatoirement et uniformément sur tout intervalle inclus dans l'intervalle $[1, 10]$. Enfin, la valeur des poids des connexions est tiré aléatoirement et uniformément sur l'intervalle $[-1, 1]$.

### 2.3.2 Evaluation de la performance de chaque individu

L'algorithme que nous utilisons a pour but de minimiser la fonction de fitness, ainsi le meilleur individu d'une population est celui qui a la fitness la plus petite tandis que la fitness la plus elevée sera en fait la fitness du plus mauvais individu.

### 2.3.3 Application de l'opérateur de sélection

Connaissant la fitness de chaque individu, on veut avantager ceux dont la performance est supérieure à la moyenne (par rapport au reste de la population). Pour cela, on calcule pour chaque individu le rapport

$$T(i) = \frac{f_{max} - f_i}{f_{max} - \bar{f}} \quad i = 1, ..., S$$

ou $S$ est la taille de la population,
$f_{max}$ est la performance du plus mauvais individu,
$f_i$ est la fitness de l'individu $i$,
$\bar{f}$ est la fitness moyenne sur toute la population.

L'individu $i$ sera copié $E[T(i)]$ fois (E[x] est la partie entière de x) dans la population intermédiaire et sa probabilité pour qu'il soit copié une fois de plus est $T(i) - E[T(i)]$ tout en maintenant constante la taille de la population (le plus mauvais ne figure jamais dans la population intermédiaire).
On remarque que la sélection est moins déterministe que dans l'algorithme GNARL pour lequel les $S/2$ meilleurs individus subissent une mutation alors que les autres sont retirés.

#### 2.3.3.1 Application des opérateurs de mutations

La population intermédiaire étant créée, on va maintenant pouvoir lui appliquer les opérateurs de mutations. On dresse tout d'abord une liste de ces opérateurs.

**Mutation des poids des connexions**

Cette mutation ajoute un bruit gaussien au poids d'une connexion. L'importance de la perturbation est liée à la fitness du réseau auquel la connexion appartient.

Si $w$ est le poids d'une connexion alors sa nouvelle valeur $\tilde{w}$ est:

$$\begin{cases} \tilde{w} = w + N(0,T) \\ \text{avec } T = U(0,1)\frac{f}{f_{max}} \end{cases} \tag{2.1}$$

et $N(\mu, \sigma)$ est la variable aléatoire gaussienne de variance $\sigma$,
$U(0,1)$ est la variable aléatoire uniforme sur $[0,1]$,
$f$ est la fitness du réseau,
$f_{max}$ est la fitness maximum sur toute la population.

Dans l'algorithme qui a été utilisé, on a considéré les mutations suivantes:
- mutation de tous les poids des connexions d'un réseau ;
- mutations de tous les poids des connexions rentrantes d'un neurone (choisi aléatoirement) ;
- mutations de tous les poids des connexions sortantes d'un neurone (choisi aléatoirement).

On note respectivement ces mutations $\mathbf{W}$, $\mathbf{W_{in}}$ et $\mathbf{W_{out}}$.

### Mutation de la fonction de transfert

La fonction de transfert d'un neurone peut contenir un paramètre comme cela est le cas pour la fonction sigmoïde standard : $g_\beta(x) = \frac{1}{1+\exp(-\beta x)}$. On peut alors définir une nouvelle mutation analogue aux mutations des poids des connexions en précisant que le facteur $\beta$ dépend de chaque neurone et que lors de la création de la population initiale, le $\beta$ de chaque neurone sera tiré aléatoirement et uniformément sur un intervalle défini par l'utilsateur.

On définit alors une nouvelle mutation analogue aux mutations des poids des connexions. La nouvelle valeur $\tilde{\beta}$ est:

$$\begin{cases} \tilde{\beta} = \beta + N(0,T) \\ \text{avec } T = U(0,1)\frac{f}{f_{max}} \end{cases} \tag{2.2}$$

Cet opérateur a pour effet de multiplier tous les poids des connexions rentrantes d'un neurone par une même constante. Le paragraphe 2.6 étudie l'influence de cette mutation (qui ne fait pas partie des mutations utilisées dans l'algorithme GNARL) sur un problème d'identification d'une fonction.

### Mutations architecturales

Ces mutations sont les mêmes que dans [A93]:

- Mutation sur une connexion (la mutation 'ajouter une connexion' est notée $\mathbf{C}+$ et la mutation 'enlever une connexion' est notée $\mathbf{C}-$) ;

- Mutation sur un neurone (la mutation 'ajouter un neurone' est notée $\mathbf{U}+$ et la mutation 'enlever un neurone' est notée $\mathbf{U}-$).

### 2.3.4 Classement des mutations

On considère que l'on traite un réseau constitué de $r$ sous–réseaux $r_i$ $(i = 1, ..., r)$ (voir figure 2.1). Le cas d'un réseau unique se déduit en considérant qu'il est constitué d'un seul sous–réseau. On divise l'ensemble des mutations en deux classes : les mutations paramétriques et les mutations architecturales. De plus, on considère que deux mutations sont différentes même si elle sont identiques mais qu'elles n'agissent pas sur le même sous–réseau.

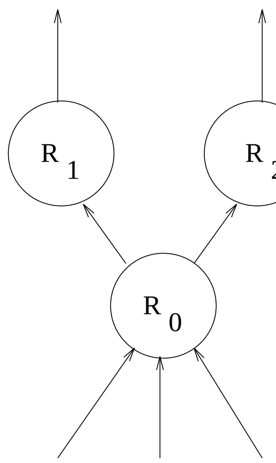



Figure 2.1: *Exemple d'un réseau comportant 3 entrées et 2 sorties et constitué de 3 sous–réseaux : le sous–réseaux $R_0$ a 3 entrées et 2 sorties et les sous–réseaux $R_1$ et $R_2$ ont une entrée et une sortie.*

#### Les mutations paramétriques

Ce type de mutation agit sur l'ensemble des poids de toutes les connexions ainsi que sur la fonction de transfert. On appelle $\mathbf{p}_i$ l'ensemble des mutations paramétriques agissant sur le sous–réseau $r_i$, $\mathbf{p}$ l'ensemble des mutations paramétriques agissant sur les connexions entre sous–réseau et $\mathcal{P}$ l'ensemble de toutes les mutations paramétriques.

$$\begin{aligned}
&\mathbf{p}_i = \{\mathbf{W}^i, \beta^i, \mathbf{W}^i_{in}, \mathbf{W}^i_{out}\} \\
&\mathbf{p} = \{\mathbf{W}, \mathbf{W}_{in}, \mathbf{W}_{out}\} \\
&\mathcal{P} = \bigcup_{i=1}^{r} \mathbf{p}_i \cup \mathbf{p}
\end{aligned}$$

#### Les mutations architecturales

Ces mutations n'agissent que sur la structure des sous–réseaux car la structure du réseau ne variera pas au cours de l'évolution mais sera fixée et identique pour tous les réseaux. On appelle $\mathbf{a}_i$ l'ensemble des mutations architecturales agissant sur le sous–réseau $r_i$ et $\mathcal{A}$ l'ensemble de toutes les mutations architecturales.

$$\begin{aligned}
&\mathbf{a}_i = \{\mathbf{C}^i_+, \mathbf{C}^i_-, \mathbf{U}^i_+, \mathbf{U}^i_-\} \\
&\mathcal{A} = \bigcup_{i=1}^{r} \mathbf{a}_i
\end{aligned}$$

### 2.3.5 Stratégie d'application des mutations

Ce paragraphe présente la stratégie d'application de chaque mutation tout au long de l'évolution. L'application d'une telle stratégie a été motivée par la volonté d'appliquer les différentes mutations de manière non déterministes et ainsi de pouvoir adapter le type de mutation pour chaque réseau au cours de lévolution. L'idée principale réside tout d'abord dans le fait que lorsque l'on change la topologie d'un réseau, il faut lui laisser un certain nombre de génération pour adapter l'ensemble de ses poids. Plus généralement, on laisse a chaque mutation paramétrique un nombre de génération (fixé au début de l'évolution) pour augmenter la performance d'un réseau. Si cette performance n'augmente pas pendant ce nombre de génération, on change de type de mutation.
Avant de présenter l'organigramme de cette stratégie (figure 2.2), nous faisons quelques remarques importantes :

- Cette stratégie s'applique de manière indépendante à chaque réseau.
- A la génération zéro, tous les réseaux commencent avec la mutation $\mathbf{W}^i$ avec $i$ choisi aléatoirement.
- On considère que le gain en fitness d'un réseau est suffisant si:

  $$Fitness(t) - Fitness(t-B) > \frac{Fitness(t)}{100}$$

  où $Fitness(t)$ indique la fitness du réseau à la génération t, $B$ est le nombre de générations de base qui est fixé au début de l'évolution et qui est identique pour chaque réseau.
- Lorsqu'on choisit une nouvelle mutation, c'est selon une certaine probabilité qui a été fixée au début de l'évolution pour chaque type de mutation.

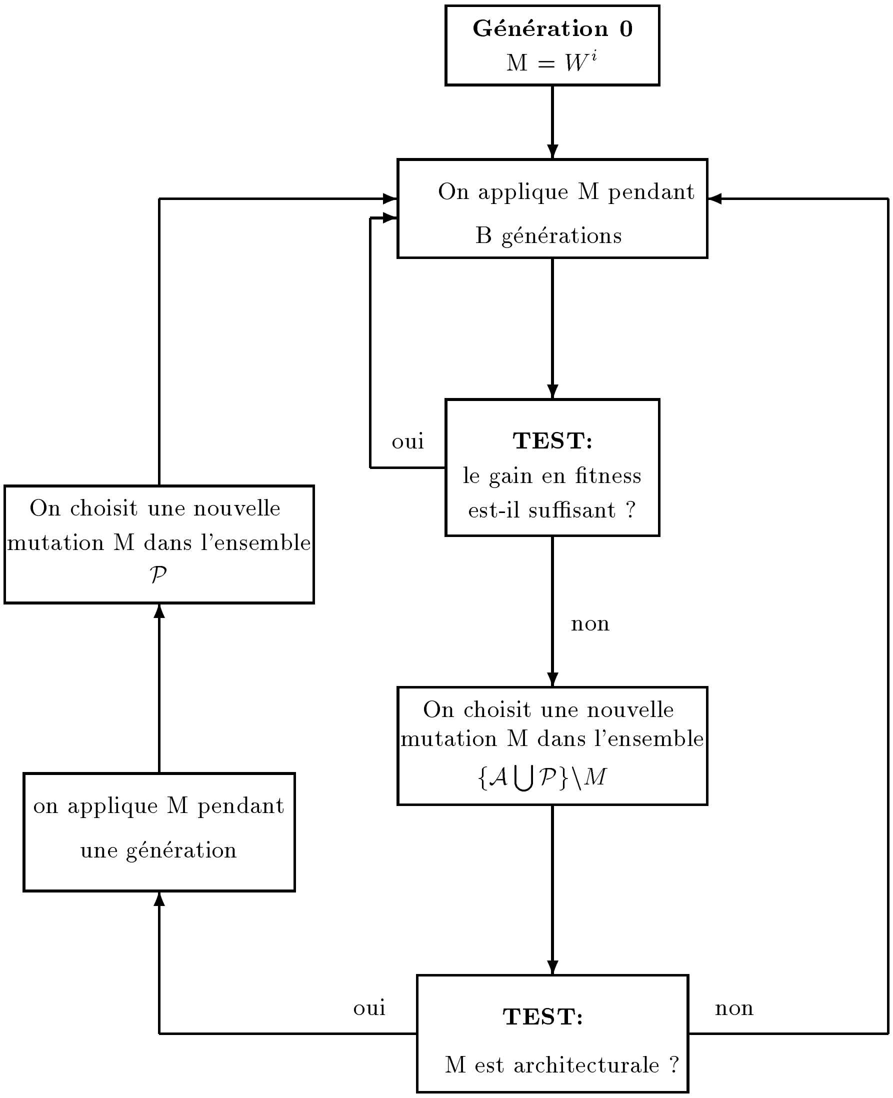



Figure 2.2: *Organigramme de la stratégie d'application de l'ensemble des mutations pour chaque réseau.*

## 2.4 Adaptation de l'algorithme A pour l'optimisation paramétrique

Nous nous situons dans le cadre de l'optimisation paramétrique où l'espace de recherche est $\mathbb{R}^N$ (N étant le nombre de paramètres à identifier). Nous travaillons directement sur l'espace de recherche sans faire de projection sur l'espace $\{0,1\}^N$.
Mis à part les étapes de création de la population initiale et d'application des opérateurs de mutations, l'algorithme est identique à celui utilisé pour l'évolution des réseaux de neurones.
La création de la population initiale consiste ici à tirer aléatoirement et uniformément une série de vecteurs appartenant à $\mathbb{R}^N$ ou à un sous–espace de $\mathbb{R}^N$. L'application des opérateurs de mutations est quant à lui remplacé par l'application des opérateurs de mutations et de croisements.

### 2.4.1 Les mutations

Si l'on note l'individu $x$ avec $x = \{x_i\}_{i=1}^n$ alors cette mutation ajoute un bruit gaussien aux $x_i$. La nouvelle valeur de $x_i$ sera

$$\left\{ \begin{array}{l} \tilde{x_i} = x_i + N(0,T) \\ \text{avec } T = U(0,1)\frac{f}{f_{max}} \end{array} \right. \tag{2.3}$$

et $N(\mu,\sigma)$ est la variable aléatoire gaussienne de variance $\sigma$,
$U(0,1)$ est la variable aléatoire uniforme sur $[0,1]$,
$f$ est la fitness de l'individu $x$,
$f_{max}$ est la fitness maximum sur toute la population.

Dans l'algorithme qui a été utilisé, on a considéré les mutations suivantes:
- mutation de tous les $x_i$, $i = 1,...,n$ (notée $M_1$);
- mutation d'un seul $x_i$ avec $i$ tiré aléatoirement (notée $M_2$).

### 2.4.2 Croisement entre deux individus

Nous utilisons le même type de croisements introduit par [Mic92] et [Rad91].

- Un premier type de croisement entre deux individus $X = \{x_1,...,x_N\}$ et $Y = \{y_1,...,y_N\}$ consiste à choisir aléatoirement un point de croisement ($\in [1,N]$) et à échanger les fragments se trouvant à droite de ce point.

$$\left. \begin{array}{l} (x_1,x_2,...,x_{N-1},x_N) \\ (y_1,y_2,...,y_{N-1},y_N) \end{array} \right\} \rightarrow \left\{ \begin{array}{l} (x_1,x_2,...,x_l,y_{l+1},...,y_{N-1},y_N) \\ (y_1,y_2,...,y_l,x_{l+1},...,x_{N-1},x_N) \end{array} \right.$$

  Ce croisement est noté $Cr_1$ ;

- Pour le croisement barycentrique on tire $\alpha$ uniformément sur [0,1] et on effectue l'opération suivante :

$$\left.\begin{array}{c}(x_1, x_2, ..., x_{N-1}, x_N)\\(y_1, y_2, ..., y_{N-1}, y_N)\end{array}\right\} \rightarrow \left\{\begin{array}{c}(\alpha x_1 + (1-\alpha) y_1, ..., x_N + (1-\alpha) y_N)\\((1-\alpha) x_1 + \alpha y_1, ..., (1-\alpha) x_N + \alpha y_N)\end{array}\right.$$

Ce croisement est noté $Cr_2$.

### 2.4.3 Stratégie d'application des mutations et des croisements

Cet algorithme étant une adaptation de l'algorithme A à l'optimisation paramétrique, nous avons donc juste adapté la stratégie en fonction des nouveaux opérateurs génétiques.
Comme dans le cas des réseaux de neurones, nous faisons quelques remarques sur la stratégie d'application dont l'organigramme sera présenté dans la figure 2.3.

- Cette stratégie s'applique de manière indépendante pour chaque individu.
- A la génération zéro, chaque individu commence avec une mutation choisie aléatoirement.
- On considère que le gain en fitness est suffisant si:

  $$Fitness(t) - Fitness(t-B) > \frac{Fitness(t)}{100}$$

  où $Fitness(t)$ indique la fitness du réseau à la génération t, $B$ est le nombre de générations de base qui est fixé au début de l'évolution et qui est identique pour chaque réseau.
- Lorsqu'on choisit une nouvelle mutation, c'est selon une certaine probabilité qui a été fixée au début de l'évolution pour chaque type de mutation.
- Lorsque l'opérateur génétique est un croisement, il faut trouver un autre individu dans la population dont l'opérateur courant est identique. Si cet autre réseau n'existe pas, on passe à l'étape suivante sans appliquer le croisement.

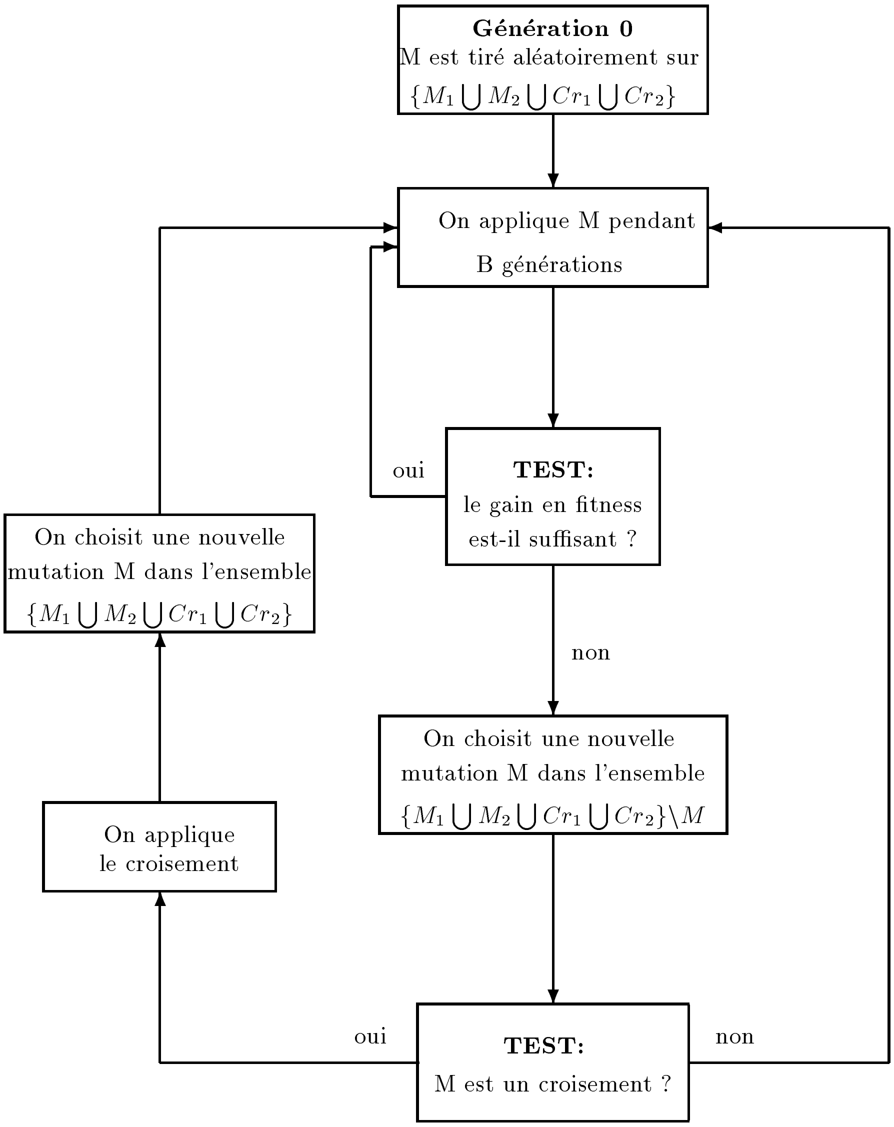



Figure 2.3: *Organigramme de la stratégie d'application de l'ensemble des mutations pour chaque réseau.*

## 2.5 Influence du nombre de générations de base sur l'évolution

Nous rappellons que le nombre de générations de base (noté B) désigne le nombre de générations laissées à une mutation paramétrique pour améliorer la performance de l'individu sur lequel on applique cette mutation. Le but de ce paragraphe est de déterminer si il existe des valeurs de B pour lesquels les résultats sont meilleurs ou bien si au contraire ce paramètre n'influe pas significativement sur l'évolution. Pour cela, nous présentons plusieurs séries de 30 calculs effectués avec des valeurs différentes de B et pour le problème d'identification d'une fonction $f$ de $\mathbb{R}$ dans $\mathbb{R}$ avec $f(x) = \frac{15x}{1+15x}$ en utilisant 11 points test répartis sur l'intervalle [0,1].
La table 2.5 donne la moyenne sur les 30 et les 25 meilleurs calculs des meilleurs fitness obtenus pour plusieurs valeurs du nombre de générations de base (le nombre de générations maximale est de 7500) et la table 2.5 présente la répartition des 30 réseaux en fonction de leur fitness également pour plusieurs valeurs de B. Etant donné que les courbes d'évolutions ne présentent pas des caractéristiques sensiblement différentes, nous ne les présenterons pas. Les différentes

| Générations de base | 1 | 3 | 5 | 7 | 10 | 15 | 17 | 20 | 25 | 30 | 50 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 30 calculs | 5.02 | 3.98 | 3.90 | 4.56 | 5.14 | 4.21 | 4.56 | 3.65 | **2.83** | 5.64 | 5.73 |
| 25 meilleurs calculs | 3.85 | 3.32 | 3.45 | 3.6 | 3.77 | 3.08 | 3.04 | 2.72 | **2.20** | 3.03 | 3.69 |
| Meilleur fitness | 0.81 | 0.49 | 0.71 | 0.77 | 1.19 | 0.72 | 0.62 | 0.78 | **0.38** | 0.78 | 0.84 |

Table 2.1: *Valeur de la moyenne sur les 30 calculs et les 25 meilleurs calculs des meilleurs fitness ainsi que la valeur de la meilleur fitness (sur les 30 calculs) pour plusieurs valeurs du nombre de générations de base (les valeurs des fitness ont été multipliées par 100). Le nombre de générations maximale est de 7500*

| B | Fitness ×100 | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | <1 | 1-2 | 2-3 | 3-4 | 4-5 | 5-6 | 6-7 | 7-8 | 8-9 | 9-10 | >10 |
| 1 | 1 | 3 | 4 | 6 | 6 | 2 | 4 | 0 | 2 | 0 | 2 |
| 3 | 2 | 5 | 2 | 6 | 5 | 5 | 2 | 2 | 1 | 0 | 0 |
| 5 | 1 | 2 | 6 | 7 | 6 | 5 | 3 | 0 | 0 | 0 | 0 |
| 7 | 1 | 7 | 2 | 4 | 1 | 9 | 2 | 1 | 1 | 0 | 2 |
| 10 | 0 | 2 | 9 | 5 | 2 | 6 | 3 | 0 | 0 | 0 | 3 |
| 15 | 2 | 6 | 3 | 7 | 3 | 4 | 1 | 0 | 2 | 1 | 1 |
| 17 | 2 | 5 | 6 | 8 | 1 | 2 | 2 | 1 | 0 | 0 | 3 |
| 20 | 1 | 5 | 6 | 12 | 2 | 3 | 0 | 0 | 0 | 0 | 1 |
| 25 | 6 | 6 | 5 | 5 | 5 | 2 | 0 | 0 | 0 | 1 | 0 |
| 30 | 3 | 4 | 5 | 7 | 4 | 1 | 0 | 2 | 0 | 1 | 3 |
| 50 | 2 | 2 | 5 | 7 | 4 | 0 | 5 | 0 | 0 | 1 | 4 |

Table 2.2: *Nombre de réseaux en fonction de la fitness pour plusieurs valeurs du nombre de générations de base B.*

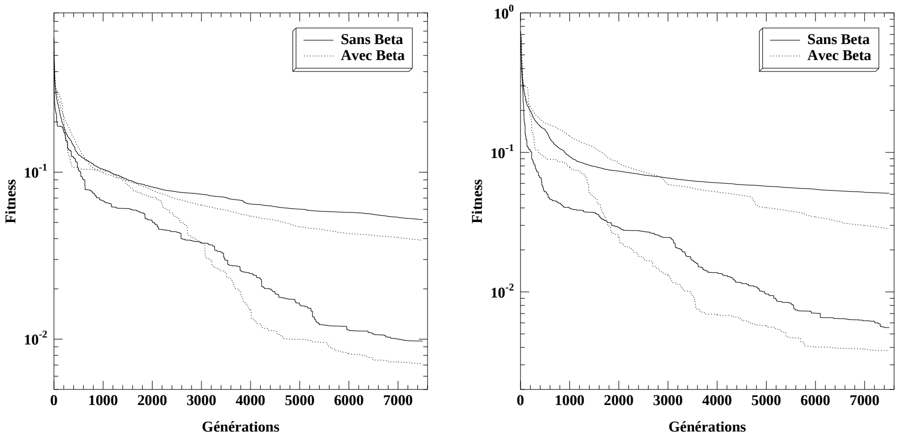
tables nous permettent de montrer que pour B supérieur à 15 et en particulier pour B=25 les résultats obtenus sont meilleurs si on raisonne aussi bien en terme de nombre de réseaux performants qu'en performance du meilleur individus. Cependant, il semble étrange de constater des différences aussi prononcée lorsque le nombre de générations de base varie de 20 à 25 ou de 25 à 30 par exemple. On peut alors penser que ce résultat est lié au nombre total de générations et/ou au problème d'identification.
Dans les deux applications que nous présenterons aux chapitres 3 et 4, nous avons choisi de fixer le nombre de générations de base à 25.

## 2.6 Influence de la mutation $\beta$ sur l'évolution

Dans ce paragraphe, nous nous proposons de démontrer l'efficacité de la mutation $\beta$ sur l'évolution d'une population de réseau de neurones.
Nous avons donc étudié le problème qui consiste à identifier la fonction $f$ de $\mathbb{R}$ dans $\mathbb{R}$ avec $f(x) = \frac{15x}{1+15x}$ en utilisant 11 points test répartis sur l'intervalle [0,1] (même problème que celui du paragraphe précédent).
Pour se rendre compte de l'influence de la mutation $\beta$, nous avons lancé 30 calculs en utilisant cette mutation, 30 calculs sans l'utiliser et pour deux valeurs de B ce qui représente 4 types de calculs.
La table 2.3 indique l'ensemble des données utilisées par l'algorithme et la figure 2.4 donne la moyenne sur les 30 calculs de l'évolution de la fitness minimum ainsi que les meilleurs évolutions pour les deux types de calculs.

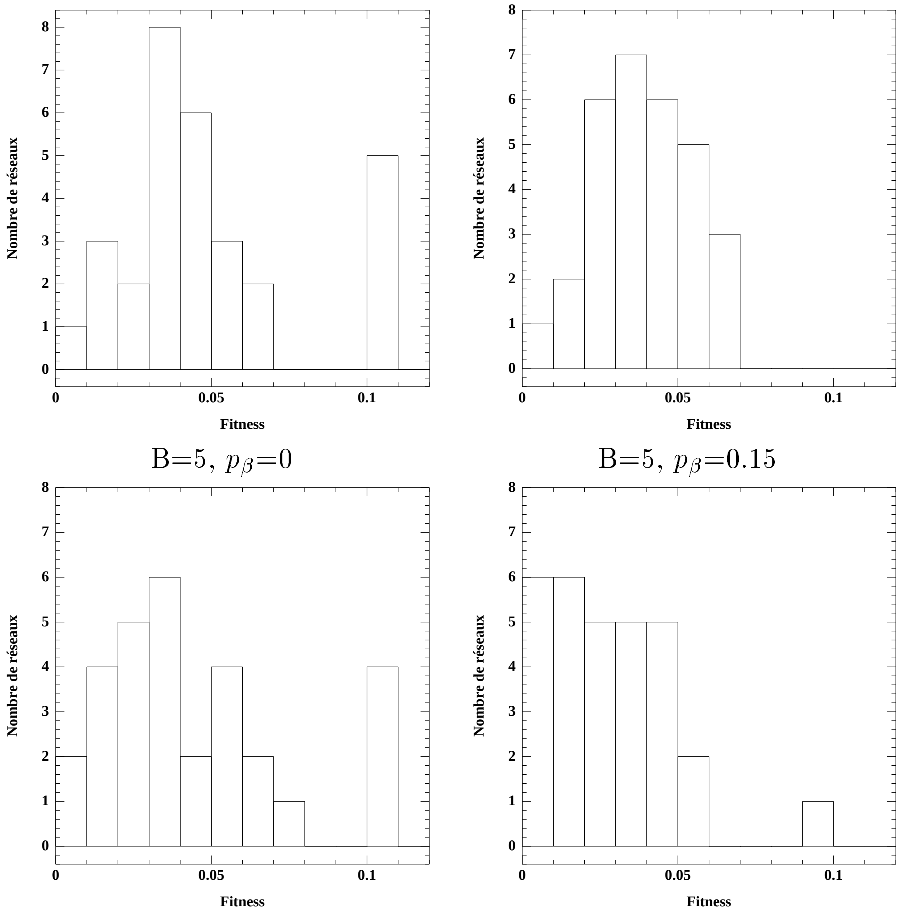
| Taille de la population: 40 | |
|---|---|
| Nombre maximal de générations: 7500 | |
| Générations de base : 5 et 25 | |
| Mutations paramétriques | Mutations architecturales |
| $p\mathbf{W} =$ 0.45 et 0.30 | $p\mathbf{C}_+=$ 0.15 |
| $p\beta =$ 0 et 0.15 | $p\mathbf{C}_-=$ 0.15 |
| $p\mathbf{W_{in}} =$ 0 | $p\mathbf{U}_+=$ 0.125 |
| $p\mathbf{W_{out}} =$ 0 | $p\mathbf{U}_-=$ 0.125 |

Table 2.3: *Ensemble des données utilisées par l'algorithme d'évolution pour les 120 calculs.*




B=5 B=25

Figure 2.4: *Moyenne sur les 30 calculs de l'évolution de la fitness minimum et meilleurs évolutions pour B=5 et B=25.*




B=5, $p_\beta$=0 B=5, $p_\beta$=0.15

B=25, $p_\beta$=0 B=25, $p_\beta$=0.15

Figure 2.5: *Nombre de réseaux en fonction de la fitness pour les* 60 *calculs avec B=5 (en haut) et B=25 (en bas). A gauche, sans utiliser la mutation $\beta$ et à droite en utilisant $\beta$ avec une probabilité de 0.15 (ex: pour B=5 et $p_\beta$=0.15 il y a 7 réseaux sur 30 dont la fitness est comprise entre 0.03 et 0.04).*

Tout d'abord, on constate clairement que l'utilisation de la mutation $\beta$ donne de meilleurs résultats comme le montre la figure 2.4. Plus précisément, cette mutation permet à un moment de l'évolution de contribuer à diminuer la fitness puisque l'on remarque que le nombre de réseaux qui ont une fitness supérieure à 0.1 est de 9 si on compte les deux cas de figure où on n'utilise pas la mutation $\beta$ ($B = 5$ et $B = 25$) alors qu'il est nul lorsqu'on utilise cette mutation (figure 2.5). Cependant, c'est la seule différence que l'on constate pour le cas où $B = 5$ alors que les résultats sont nettement meilleurs comme le montre la répartition du nombre de réseaux "vers les fitness basses" lorsque $B = 25$ et $p_\beta = 0.15$.
Ainsi, dans les deux applications que nous présenterons dans les chapitres 3 et 4, nous avons fixé la probabilité $p_\beta$ à 0.15 et $B$ à 25.

# Chapitre 3

# Identification de la fonction isotherme en chromatographie

## 3.1 La chromatographie préparative

La chromatographie est un processus physique qui a pour but la séparation des divers composants constituant un mélange chimique et qui est fondée sur l'affinité différente de chaque composant pour un milieu donné.
Le principe est le suivant : on injecte le mélange dans un tube de grande longueur et de faible diamètre empli d'un solide poreux que l'on appelle colonne. Chaque composant va passer d'une phase mobile (déplacement des composants) à une phase stationnaire dont la durée varie en fonction de chaque corps constituant le mélange. La phase mobile (gazeuse ou liquide) est assurée par un fluide vecteur injecté en tête de colonne et dont le débit est connu. La phase stationnaire est la phase pendant laquelle un composant va être retenu par le milieu poreux. Il y aura *absorption* si ce composant se fixe sur ce milieu solide (on parlera alors d'une phase stationnaire solide). Une autre possibilté est l'*adsoption* de ce composant par un liquide fixé sur cette structure poreuse (on parlera alors d'une phase stationnaire liquide). Par la suite, on utilisera le terme d'adsorption aussi bien dans un cas que dans l'autre. La durée de la phase stationnaire dépend du composant lui même ainsi que des autres composants en présence. Les composants vont donc se propager le long de la colonne à des vitesses différentes si bien que chacun d'entre eux en sortira à un moment différent : ils seront séparés. La variation de concentration en sortie de colonne au cours du temps s'appelle le chromatogramme.
Les modèles de chromatographie que nous utiliserons sont des modèles adaptés pour la chromatographie préparative qui est le nom que l'on donne à la chromatographie lorsqu'elle est utilisée en temps que procédé industriel de séparation. De par sa nature la propagation dans ce type de processus est non linéaire et la diffusion peut être négligée.
Chaque combinaison entre phase mobile et phase stationnaire donne lieu à un modèle différent (chromatographie liquide-solide encore appelée chromatographie en phase liquide par exemple). Les modèles étudiés sont valables pour la chromatographie en phase liquide ou gazeuse.
Les équations du modèle de propagation sont volontairement simplifiées afin d'avoir une vue d'ensemble du problème et pour ne pas alourdir inutilement les notations. Cependant tous les résultats ont été obtenus en utilisant un modèle plus réaliste nécessaire à l'utilisation de vraies données expérimentales (vrais chromatogrammes). Dans ce chapitre, on s'intéresse au problème

inverse qui consiste à identifier la fonction isotherme, que l'on appelera simplement isotherme par la suite. Cette fonction non linéaire de $\mathbb{R}^M$ vers $\mathbb{R}^M$, où $M$ est le nombre de corps constituant le mélange, traduit l'état d'équilibre isotherme (c'est-à-dire à température constante) entre la phase mobile et la phase stationnaire et intervient dans le modèle mathématique de propagation.
Connaissant les conditions expérimentales (durée et vitesse d'injection du melange, état initial de la colonne, quantité injectée dans la colonne) et cette fonction, le modèle nous permet d'obtenir les chromatogrammes (problème direct). Le problème inverse consiste alors, à partir des conditions expérimentales et des chromatogrammes, à retrouver la fonction isotherme. Pour cela nous emploierons quatre méthodes évolutionnaires que nous avons en partie présentées au chapitre 2.

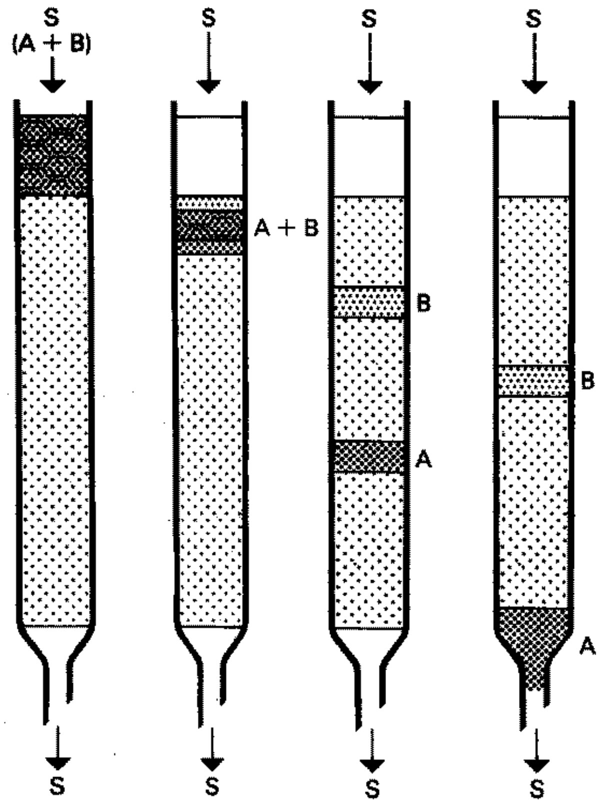



Figure 3.1: *Principe de la chromatographie : la colonne est vide, et au temps $t = 0$, on injecte le mélange constitué de $M$ composants (ici $M = 2$) pendant une durée $t_{inj}$ (on dit qu'on injecte un "créneau"). Les composants vont ensuite se propager à des vitesses différentes et ainsi être séparés lorsqu'ils sortiront de la colonne.*

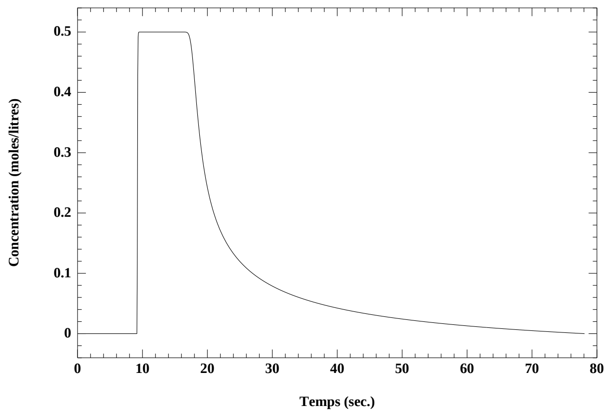



Figure 3.2: *Chromatogramme obtenu pour une injection en tête de colonne au temps $t = 0$ d'un mélange (dans ce cas il y a un seul composant chimique) ayant une concentration de 0.5 mol/l pendant 15 secondes à une vitesse de 12.3 cm/s. La colonne mesure 25 cm.*

## 3.2 Le problème direct

On étudiera uniquement des modèles de propagation en chromatographie préparative. Pour la modélisation les hypothèses suivantes ont été faites :

- la colonne est considérée comme monodimensionelle (de longueur $L$) ;
- la température $T$ est constante ;
- la diffusion axiale est négligée ;
- la vitesse de transfert entre les phases stationnaires et mobiles est infinie et il n'y a pas de réaction chimique ;
- le processus chromatographique est quasi-statique (succession continue d'états d'équilibres thermodynamiques).

L'écriture des équations de conservation de la masse nous mène alors à un modèle de propagation qui entre dans le cadre de la théorie des systèmes hyperboliques non linéaires à une dimension d'espace du premier ordre et qui s'écrit de manière très générale [VG76, J90]

$$\begin{cases} \partial_z w + \partial_t f(w) = 0, \quad 0 \leq z \leq L, t > 0 \\ w(0,t) = w^{inj}(t), \quad w(z,0) = w^{init}(z) \end{cases} \tag{3.1}$$

où $(z,t)$ est dans le domaine $[0,L] \times [0,+\infty[$, $w(z,t) = (w_1(z,t),...,w_M(z,t))^T$ est le vecteur des quantités des corps chimiques se propageant dans la colonne ($w(L,t)$ exprime le chromatogramme) et $f = (f_1,...,f_M)^T$ le flux du système (3.1) qui est une application régulière de $\mathbb{R}^M$ vers $\mathbb{R}^M$. Le flux se décompose en une partie linéaire et une partie non linéaire appelée

isotherme et qui exprime l'état d'équilibre isotherme entre la phase mobile et le phase stationnaire.
Usuellement, les systemes de lois de conservation s'écrivent avec l'espace et le temps inversés par rapport à (3.1) mais l'étude mathématique reste la même. La donnée initiale usuelle est dans notre cas la donnée d'injection ($w(0,t)$) et l'état initial de la colonne ($w(z,0)$) est équivalent à une condition aux limites.

De plus, le système (3.1) est un système de $M$ équations aux dérivées partielles à $M$ inconnues. Pour des modèles d'isothermes simples, on arrive à démontrer l'existence et l'unicité d'une solution pour les systèmes du type (3.1) ainsi que la convergence de certains schémas numériques [J90]. Pour des modèles d'isothermes plus compliqués, ce n'est pas le cas, mais on arrive cependant à avoir une validation numérique pour des schémas de Godunov ou de Van-Leer.

Connaissant les conditions expérimentales et la fonction isotherme, le problème direct consiste alors à résoudre le problème (3.1). La solution peut ensuite être comparée avec des données réelles (résultats d'une vraie simulation) afin de valider le modèle.

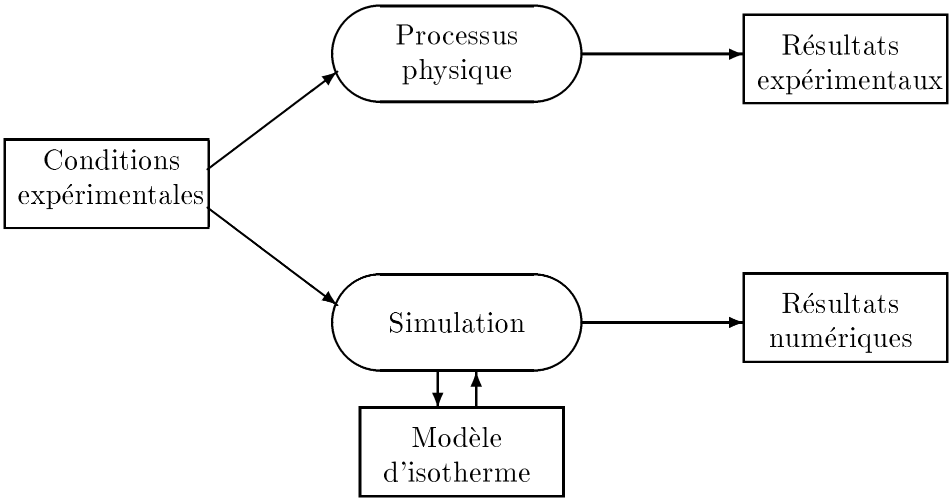



Figure 3.3: *Le problème direct : il peut être soit simulé (on a alors besoin d'un modèle d'isotherme) soit réalisé expérimentalement.*

Comme on l'a vu, pour résoudre ce problème, il faut utiliser un schéma numérique et un modèle d'isotherme.

### 3.2.1 Méthode numérique de résolution

Afin de résoudre numériquement l'équation (3.1), on utilise le procédé de discrétisation de Godunov ou schéma de Godunov [G59].
Les valeurs propres du jacobien du flux $f$ étant positives ou nulles, cela nous permet de déduire une forme très simple pour ce schéma :

$$\left\{ \begin{array}{l} w_j^{n+1} = w_j^n - \lambda(f(w_j^n) - f(w_{j-1}^n)) \\ w_j^0 = w_j^{inj}, \quad w_0^n = w_n^{init} \end{array} \right. \tag{3.2}$$

où $\lambda = \frac{\Delta z}{\Delta t}$ est constante, $\Delta z$ et $\Delta t$ étant respectivement le pas d'espace et le pas de temps. $w_j^n$ est une approximation de la solution $w$ au point $(n\Delta z, j\Delta t)$ et $f = (f_1, ..., f_M)^T$ le flux du système (3.1) qui se décompose en une partie linéaire et une partie non linéaire appelée isotherme.
Comme on peut le constater, la complexité d'un tel schéma est en $Z \times R$ avec $L = Z\Delta z$ et $T = R\Delta t$ ($Z$ est le nombre de pas d'espace et $R$ le nombre de pas de temps). Dans la pratique elle est souvent inférieure et dépend ainsi des conditions expérimentales. En effet, on connaît à chaque pas de temps et pour chaque corps l'intervalle ($\in [0, L]$) où la concentration est non nulle.

### 3.2.2 Les modèles d'isothermes

La fonction isotherme traduit l'état d'équilibre isotherme (c'est-à-dire á température constante) entre la phase mobile et la phase stationnaire et exprime la quantité (en nombre de moles) d'une certaine espèce dans l'une des deux phases. Selon le type de chromatographie les contraintes sur chaque phase et donc les relations d'équilibre seront différentes.
On considère que les interactions ont uniquement lieu en phase stationnaire.

#### L'isotherme de Langmuir

Ce modèle d'isotherme largement utilisé [L16] prend en compte uniquement les interactions entre les solutés (composants chimiques constituant le mélange) et l'adsorbant (liquide ou solide). On a uniquement besoin des $M$ énergies d'interaction entre solutés et adsorbant pour exprimer l'isotherme. L'énergie d'interaction $E_m$ entre une molécule du composant $m$ et l'adsorbant permet de calculer le coefficient de Langmuir $K_m$.
A titre d'exemple, l'expression de l'isotherme pour un équilibre gaz-solide s'écrit

$$h_m(w) = \mathcal{N}^* \frac{w_m}{1 + \sum\limits_{m=1}^{M} w_m}, \quad m = 1, ..., M$$

où $\mathcal{N}^*$ désigne la quantité totale de matière en phase stationnaire. On considérera un système simplifié de la chromatographie en posant $\mathcal{N}^* = 1$. Cependant, si l'on devait résoudre un problème avec des données réelles, il faudrait identifier ce paramètre.
L'expression de l'isotherme peut s'écrire aussi en fonction de la concentration $c_m$ du corps $m$

$$h_m(w) = \mathcal{N}^* \frac{K_m c_m}{1 + \sum\limits_{m=1}^{M} K_m c_m}, \quad m = 1, ..., M$$

#### L'isotherme de Moreau-Valentin

Ce modèle d'isotherme est une généralisation de l'isotherme de Langmuir. On considère qu'il y a non seulement des interactions entre les solutés et l'adsorbant (liquide ou solide) mais aussi entre les molécules en présence.
La formulation analytique de cet isotherme est assez compliquée et dépend évidemment des deux phases. C'est pourquoi on ne donnera ici qu'une formulation très générale de cet isotherme (pour

plus de precision on renvoie à [JV91]).
Pour ce modèle il faut préciser le degré de l'isotherme. Plus le degré est élevé, plus on tient compte des diverses possibilités d'interaction entre les solutés. Ainsi, pour un problème à deux corps et en considérant un isotherme de degré 4, on considérera entre autres l'interaction entre trois molécules du corps 1 et une molécule du corps 2 (énergie d'interaction $E_{31}$) alors que ce n'est pas le cas pour un isotherme de degré 2.
De manière générale, si l'on a $M$ composants dans le mélange et si l'on considère un isotherme de degré $q$, on aura besoin de $\frac{(q+1)(q+2)\ldots(q+M)}{M!} - 1$ énergies d'interaction pour exprimer l'isotherme. Parmi ces paramètres, il y a les $M$ coefficients de Langmuir.
L'expression de l'isotherme de degré $q$ pour un équilibre gaz-solide s'écrit de manière très générale

$$h_m(w) = \mathcal{N}^* \frac{w_m \dfrac{\partial P}{\partial w_m}}{qP}$$

avec $P$ polynôme de degré $q$ en $w_1, \ldots, w_M$, fonction de divers paramètres dont tous les coefficients énergétiques.

On retrouve l'expression d'un isotherme de Langmuir en posant $q = 1$ ou bien en considérant que toutes les énergies d'interaction entres les solutés en phase adsorbée sont nulles.

## 3.3 Le problème inverse

La résolution du problème inverse a pour but l'identification de la fonction isotherme. Cette fonction n'est jamais précisément connue et le fait que seules quelques valeurs particulières peuvent etre calculées rend impossible l'utilisation de méthodes d'approximation classique. Connaissant les conditions expérimentales et les résultats expérimentaux, on veut trouver la fonction isotherme. Plusieurs approches pour résoudre ce problème vont maintenant être présentées. La première a été réalisée par Sepulveda [S94, JS94], tandis que les quatre autres sont l'objet de ce chapitre.

### 3.3.1 La méthode du gradient

Le but de cette méthode ([S94]) est d'identifier les coefficients des isothermes présentés ci-dessus connaissant le chromatogramme expérimental et les conditions initiales. On se ramène alors alors à un problème de minimisation sous contraintes d'une fonction coût définie dans l'espace des paramètres caractérisant le modèle d'isotherme choisi. Pour résoudre ce problème, on définit un Lagrangien associé et on résout sous certaines hypothèses de regularité le système adjoint, ce qui permet alors d'appliquer une méthode de gradient conjugué.
Cette méthode a montré qu'elle était capable de résoudre des problèmes à deux corps en chromatographie liquide-solide avec un isotherme de Moreau-Valentin de degré 2 (5 coefficients à identifier) à partir de vrais chromatogrammes expérimentaux. Les inconvénients de cette méthode sont principalement l'importance du choix du point de départ (pour le gradient conjugué) et du type d'isotherme que l'on va utiliser pour l'identification ce qui implique qu'il faut connaître le plus grand nombre d'informations possibles.

### 3.3.2 Les méthodes évolutionnaires

Nous allons maintenant présenter plusieurs méthodes utilisant les algorithmes d'évolution. Par rapport à la méthode du gradient, l'avantage de ces algorithmes est que leur réussite n'est pas liée à l'initialisation de la population.

#### 3.3.2.1 La fonction de perfomance

Dans ces méthodes la performance d'un individu sera le résultat de la comparaison entre un chromatogramme expérimental et le chromatogramme obtenu en résolvant le problème direct. Ce problème direct sera résolu grâce aux conditions expérimentales et en utilisant l'individu comme isotherme (voir la *Figure 3.4*).

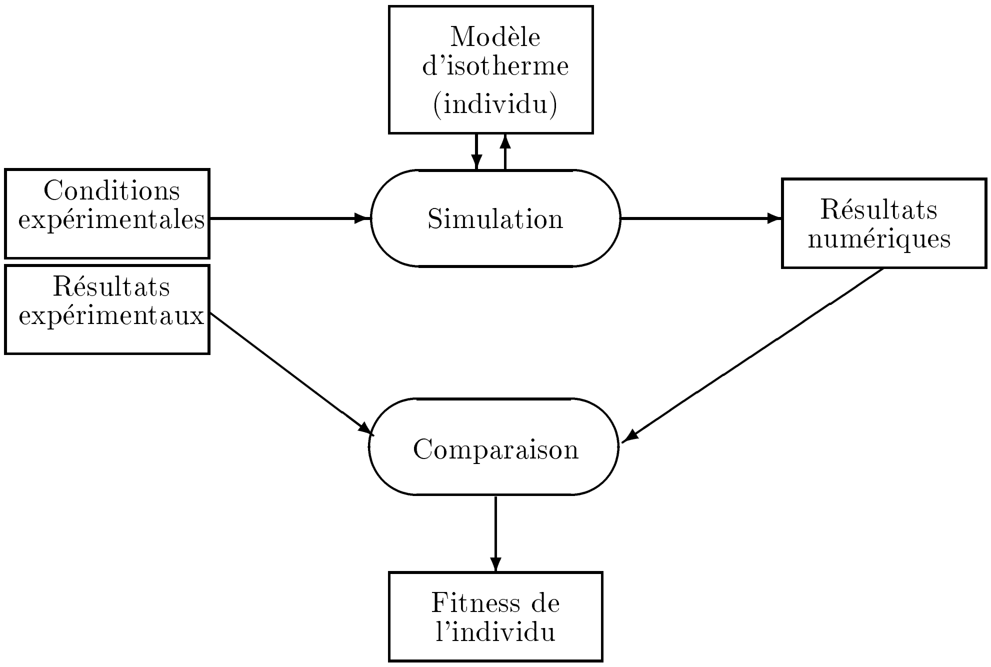



Figure 3.4: *Calcul de la performance d'un individu pour le problème inverse.*

La résolution du problème direct en utilisant l'individu I revient donc à résoudre numériquement le problème

$$\begin{cases} \partial_z w + \partial_t(w + I(w)) = 0, \quad 0 \le z \le L, t > 0 \\ w(0,t) = w^{inj}(t), \quad w(z,0) = w^{init}(z) \end{cases} \tag{3.3}$$

où $(z,t) \in [0,L] \times [0,+\infty[$, $w(z,t) = (w_1(z,t),...,w_M(z,t))^T$ est le vecteur des quantités des corps chimiques se propageant dans la colonne.
La performance $f$ d'un individu est donc donnée par :

$$f = \sum_{i=1}^{M} \|W_i(t) - w_i(L,t)\|_{L^2} \tag{3.4}$$

où $W(t) = \{W_i(t)\}_{i=1}^M$ est le chromatogramme expérimental et $w(z,t) = \{w_i(z,t)\}_{i=1}^M$ est la solution de (3.3).

On verra par la suite qu'il sera nécessaire et/ou plus efficace d'utiliser plusieurs conditions expérimentales (et donc plusieurs chromatogrammes expérimentaux) au cours de l'évolution (on dira qu'on utilise plusieurs points test). La performance d'un individu sera alors

$$f = \frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{M}\frac{1}{p_i^n}\|W_i^n(t) - w_i^n(L,t)\|_{L^2} \tag{3.5}$$

où $N$ est le nombre de conditions expérimentales , $W^n(t) = \{W_i^n(t)\}_{i=1}^M$ est le n-ième chromatogramme expérimental, $w^n(z,t) = \{w_i^n(z,t)\}_{i=1}^M$ est la solution du problème (3.3) correspondant à la n-ième condition expérimentale et $p_i^n$ représente la quantité du corps $i$ injecté lors de l'expérience $n$. Ce dernier coefficient permet d'avoir une représentation égale de chaque experience au niveau de la fitness.
Les paragraphes suivants présentent l'ensemble des méthodes évolutionnaires que nous allons utiliser dans ce chapitre.

#### 3.3.2.2 Les réseaux de neurones comme modèle d'isotherme

Les réseaux de neurones peuvent constituer un bon modèle d'isotherme car, comme on l'a vu au chapite 1, ils sont capables d'approcher toutes les fonctions continues. Par rapport au modèle de Langmuir ou Moreau-Valentin, ce modèle présente a priori l'avantage d'être plus général et donc il n'est pas nécessaire de choisir un type d'isotherme avant une identification comme pour la méthode du gradient.
Nous avons tout d'abord utilisé des réseaux feed-forward mais, lorsque les résultats n'étaient pas jugés suffisamment bons, nous avons utilisé les réseaux récurrents qui en sont un sur-ensemble. Comme on l'a vu au chapitre consacré aux réseaux de neurones (chapitre 1), les réseaux récurrents peuvent être utilisés de deux manières : soit dynamiquement pour les phénomènes dependant du temps ou bien statiquement et dans ce cas la sortie est un état d'equilibre (lorsqu'il existe) atteint après stabilisation. Etant donné que la fonction isotherme ne dépend pas du temps c'est la deuxième utilisation qui a été choisie.
Par ailleurs, pour optimiser à la fois les poids des connexions et la structure du réseau, nous avons utilisé l'algorithme évolutionnaire inspiré de l'algorithme GNARL [A94] qui a été présenté au chapitre 2.

#### 3.3.2.3 Les fractions rationnelles comme modèle d'isotherme

Cette méthode utilise les fractions rationnelles comme modèle d'isotherme. Connaissant l'ensemble des points de contrôle répartis uniformément sur un domaine $D \subset \mathbb{R}^M$ ainsi que leurs valeurs, on peut calculer par interpolation par fraction rationnelle la valeur d'un point quelconque appartenant à $D$. On considérera pour l'instant connu le nombre de points de contrôle afin de vérifier la validité de cette approche mais, si l'on veut appliquer cette méthode pour un problème concret, il faudra faire ce choix qui nécessitera alors des informations supplémentaires (comme pour le choix de l'isotherme dans la méthode du gradient).
Partant de ce modèle, on peut alors utiliser un réseau de neurones pour identifier grâce à notre algorithme les valeurs de ces points, auquel cas on se ramène au cas ci-dessus, sauf les autres

valeurs de la fonction isotherme demandées par le schéma de Godunov qui seront calculées par interpolation et non pas uniquement par le réseau (si l'on observe le schéma aux différences finies utilisé pour la résolution du problème direct, on s'aperçoit qu'en un certain nombre de points qu'on appelera domaine d'identification, il faut calculer la valeur de la fonction isotherme).
Cette approche qui sera présentée au paragraphe 3.5, est motivée par deux raisons : la première est que l'on peut s'attendre à une baisse des temps de calcul qui sont assez élevés en utilisant la méthode énoncée au paragraphe précédent. La deuxième raison est qu'il est sans doute plus facile à un réseau d'identifier les valeurs des points de contrôle plutôt que d'identifier toutes les valeurs des points du domaine d'identification.
Dans ce cas, on passe d'un individu à l'isotherme de la manière suivante :

**Individu** = Réseau de neurones
↓
Calcul des valeurs des points de contrôle
↓
Interpolation par fraction rationnelle
↓
**Isotherme**

Une autre approche consiste non plus à identifier les valeurs par un réseau de neurones mais directement par un algorithme d'évolution qui fera évoluer une population de vecteurs réels de taille $p$ ($p$ points de contrôle).

**Individu** = Vecteur réel de taille $p$
↓
Interpolation par fraction rationnelle
↓
**Isotherme**

Ces deux approches sont présentées respectivement aux paragraphes 3.6 et 3.7.

#### 3.3.2.4 Identification des coefficients des isothermes par algorithme d'évolution

Cette méthode consiste à utiliser les modèles d'isothermes existants (Moreau-Valentin ou Langmuir) et à identifier les coefficients par un algorithme d'évolution qui fera évoluer une population de vecteurs réels. Cette approche fera l'objet du paragraphe 3.8.
Du point de vue de l'optimisation paramétrique, cette méthode est plus simple que la méthode précédente car, pour un même problème, il faut identifier un moins grand nombre de coefficients. Ainsi, sur un problème à un corps, les paramètres d'un isotherme de Moreau-Valentin de degré 4 sont au nombre de 5 alors qu'il faut 9 points de contrôle pour représenter ce même isotherme par une fraction rationnelle. Cependant cette dernière méthode offre l'avantage de ne pas être liée à un modèle d'isotherme et peut donc être plus facilement généralisée.

## 3.4 Introduction aux résultats

Nous présentons dans les paragraphes qui suivent, l'ensemble des résultats obtenus en utilisant quatre méthodes d'identification d'isothermes en chromatographie liquide-solide :

- Identification en utilisant les réseaux de neurones comme modèle d'isotherme (méthode *Réseau*: paragraphe 3.5).

- Identification en utilisant les fractions rationnelles comme modèle d'isotherme et en identifiant les valeurs des points de contrôle par un réseau de neurones (méthode *Réseau+Interpolation*: paragraphe 3.6).

- Identification en utilisant les fractions rationnelles comme modèle d'isotherme et en identifiant les valeurs des points de contrôle par un algorithme de stratégie d'évolution (méthode *ES+Interpolation*: paragraphe 3.7).

- Identification par algorithme génétique des coefficients des modèles d'isotherme existants (paragraphe 3.8.1).

Dans chaque cas les chromatogrammes à identifier ont été obtenus en résolvant un problème direct avec un isotherme connu et certaines conditions expérimentales. Pour l'ensemble de ces calculs on a souvent fixé le nombre de pas d'espace $Z$ à 50 pour ne pas avoir des temps de calcul trop elevés (ce qui a quand même été le cas). De plus, étant donné la nature stochastique des algorithmes d'évolution, on lance plusieurs fois le même calcul en utilisant les mêmes paramètres.

Les calculs ont été effectués sur deux types de machines dont les performances sont assez proches : stations HP-PA 8000 et Pentium Pro à 200 Mhz.
Par la suite, on présentera des graphiques d'erreur relative d'une courbe par rapport à une autre. Si cette erreur concerne des chromatogrammes, elle aura été calculée en divisant la différence des deux courbes en chaque point par la valeur maximale de la courbe de référence (chromatogramme expérimental) sinon, elle aura été calculée en divisant la différence des deux courbes en chaque point par la valeur de la courbe de référence sur ce point (isotherme à identifier).

On utilisera les notations suivantes:

| Symbole | Unités | Signification |
|---|---|---|
| $L$ | cm | Longueur de la colonne. |
| $T$ | Kelvin | Température. |
| $t_{inj}$ | s | Durée de l'injection (largeur du créneau). |
| $u_0$ | cm/s | Vitesse d'injection. |
| $\varepsilon$ | | Porosité ou fraction de vide. |
| $c_{inj}$ | mol/l | Concentration d'injection (quantité injectée dans la colonne). |
| $c_{init}$ | mol/l | Concentration initiale. |
| $c^i_{inj}$ | mol/l | Concentration d'injection pour le point test $i$. |
| $c^i_{init}$ | mol/l | Concentration initiale pour le point test $i$. |
| $E_i$, $E_{ij}$, $E_{ijk}$ | cal/mol | Coefficients énergétiques intervenant dans l'isotherme. |
| $K_i$ | l/mol | Coefficients de Langmuir. |

Table 3.1: *Liste des notations utilisées.*

## 3.5 Les réseaux de neurones comme modèle d'isotherme (méthode *Réseau*)

On présente tout d'abord quelques données utilisées pour l'initialisation de la population des réseaux de neurones.

- Pour un problème à $n$ corps, l'isotherme est une fonction de $\mathbb{R}^n$ vers $\mathbb{R}^n$, donc les réseaux utilisés comporteront $n$ entrées et $n$ sorties.
- Le nombre de neurones cachés ou intermédiaires (noté $\mathbf{h}$) sera tiré aléatoirement et uniformément pour chaque réseau dans l'intervalle $[1, 5]$.
- La fonction de transfert qu'on utilisera sera toujours la fonction sigmoïde $\sigma_\beta(x) = \frac{1}{1+e^{-\beta x}}$. On définit alors un intervalle de départ pour $\beta$ qui sera toujours $[-10, 10]$.
- On définit la densité de connexion qui indique indépendamment du nombre de neurones le nombre total de connexions d'un réseau. Cette echelle varie de 1 à 10, 1 signifiant que chaque neurone est connecté à au moins un autre neurone et 10 signifiant que chaque neurone est connecté à ($\mathbf{h}$ + le nombre de neurones de sortie) autres neurones (certains pouvant être les mêmes). La densité pour chaque réseau sera toujours tirée aléatoirement et uniformément dans l'intervalle $[1, 10]$.
- Lorsque les réseaux seront récurrents, on utilisera toujours la méthode des itérations successives avec $\Delta t = 1$ (méthode *Euler 1* du chapitre 1) et une précision du calcul $P = 0.001$ (voir chapitre 1). De plus, on ne ré-initialisera pas les neurones (voir chapitre 1) ce qui pourra entraîner une différence de résultats pour une meme entrée. Cependant, on espère que de tels réseaux seront éliminés par la sélection.
- Le nombre de générations de base (qui est un paramètre de notre algorithme (chapitre 2, page 32)) sera toujours égal à 25.
- Lorsque la fitness n'augmente plus depuis 200 générations, on arrête le calcul.

Pour avoir une idée de la complexité des calculs, nous donnerons pour chaque résultat le nombre total de fois où le programme a fait appel à un réseau de neurones. Ce nombre est égal à Nombre de calculs $\times$ Nombre de générations $\times$ Taille de la population $\times$ Complexité du schéma . Cette formule est valable uniquement lorsque l'on prend en compte une seule condition expérimentale (et donc un seul chromatogramme) dans la fonction de fitness (voir (3.4)). Soulignons que le nombre de générations n'est pas forcément égal au nombre maximal de générations car on arrête le calcul lorsque la fitness n'augmente plus pendant 200 générations. Dans le cas où l'on utilise plusieurs conditions expérimentales (voir (3.5)), on remplace la complexité du schéma par la somme des complexités des différents schémas, car la complexité dépend des conditions expérimentales. En réalité, la complexité du schéma multipliée par la taille de la population n'est pas exactement égale au nombre d'appels total aux réseaux pour une génération. Sans entrer dans les détails, cela est dû à l'optimisation de la routine de simulation qui implique que la complexité va dépendre de la fonction isotherme qu'on utilise et donc, comme chaque réseau est différent, on aura une complexité différente (mais du même ordre de grandeur) pour chaque individu. Cette complexité tendra cependant au fil de l'évolution vers la complexité du schéma lorsqu'on utilise l'isotherme à identifier.

### 3.5.1 Chromatographie liquide-solide à un corps

#### 3.5.1.1 Isotherme de Langmuir

Ce problème consiste à identifier une fonction de $\mathbb{R}$ dans $\mathbb{R}$. On commence par l'identification de l'isotherme de Langmuir ($K = 15$).

Les résultats représentent le meilleur des 5 calculs effectués, chaque calcul faisant évoluer 50 réseaux de neurones feed-forward pendant 2000 générations. Le temps total de calcul est d'environ 5 heures CPU . La complexité du schéma est d'environ 1800 et au total le programme a fait appel environ 700 millions de fois aux réseaux de neurones.

| Caractéristiques de la colonne | Conditions expérimentales | | Isotherme à identifier |
|---|---|---|---|
| | Conditions d'injection | Conditions initiales | |
| $L = 25$ cm<br>$T = 300$ K<br>$\varepsilon = 0.5$ | $t_{inj} = 35$ s<br>$u_0 = 12.3$ cm/s<br>$c_{inj} = 0.015$ mol/l | $c_{init} = 0$ mol/l | Langmuir $K_1 = 15$ |

Table 3.2: *Chromatographie liquide-solide : ensemble des données utilisées dans la simulation.*

| Taille de la population: 50 | |
|---|---|
| Nombre maximal de générations: 2000 | |
| Mutations paramétriques | Mutations architecturales |
| $p\mathbf{W} =$ 0.35 | $p\mathbf{C}_+ =$ 0.15 |
| $p\beta =$ 0.15 | $p\mathbf{C}_- =$ 0.15 |
| $p\mathbf{W_{in}} =$ 0 | $p\mathbf{U}_+ =$ 0.1 |
| $p\mathbf{W_{out}} =$ 0 | $p\mathbf{U}_- =$ 0.1 |

Table 3.3: *Ensemble des données utilisées par l'algorithme d'évolution dans la simulation.*

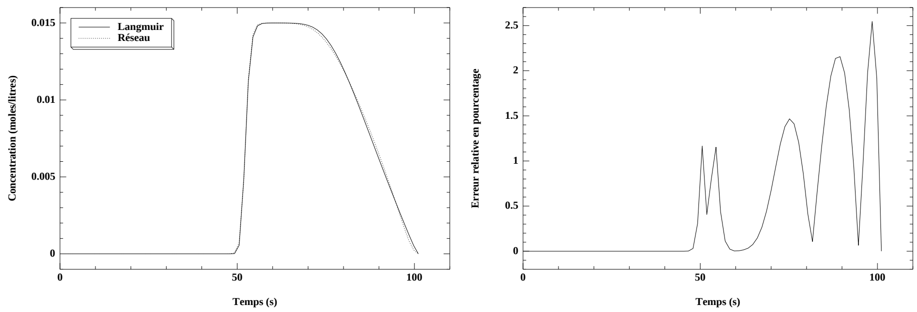



Figure 3.5: *Identification de l'isotherme de Langmuir ($K = 15$) avec 50 pas d'espace : chromatogrammes experimental et identifié à gauche et erreur relative à droite.*

0

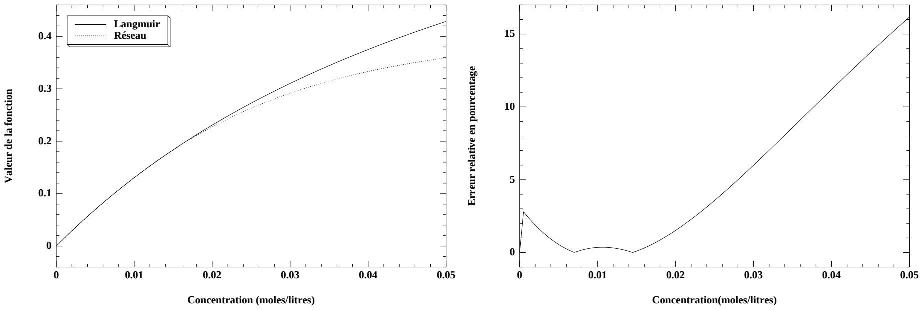



Figure 3.6: *Identification de l'isotherme de Langmuir ($K = 15$) : valeurs des deux fonctions isothermes (à gauche) et différence relative entre les deux fonctions isothermes (à droite).*

On remarque que l'isotherme identifié n'est pas valable sur tout l'intervalle. Si l'on observe le schéma aux différences finies (ici le schéma de Godunov), on s'aperçoit qu'en un certain nombre de points (complexité du schéma) il faut calculer la valeur de la fonction isotherme et que tous ces points appartiennent à l'intervalle défini par la condition initiale et la condition d'injection, ce qui explique qu'en dehors de cet intervalle (qu'on appellera domaine d'identification) la fonction isotherme ne sera pas identifiée.
En réalité, l'ensemble des points constituant ce domaine dépend de la fonction isotherme comme l'indique le schéma de Godunov : pour calculer $w_j^{n+1}$ il faut connaître $f(w_j^n)$ et $f(w_{j-1}^n)$ ($f$ étant le flux constitué d'une partie linéaire et de la fonction isotherme). Ainsi, comme on travaille sur une population de réseaux de neurones, chaque réseau aura son propre domaine d'identification. Cependant, même au début de l'évolution, on ne constate aucune différence majeure sur les bornes du domaine. Par contre, l'ensemble des points est différent (en nombre et en position) mais tend rapidement au cours de l'évolution vers le domaine d'identification de l'isotherme à identifier.

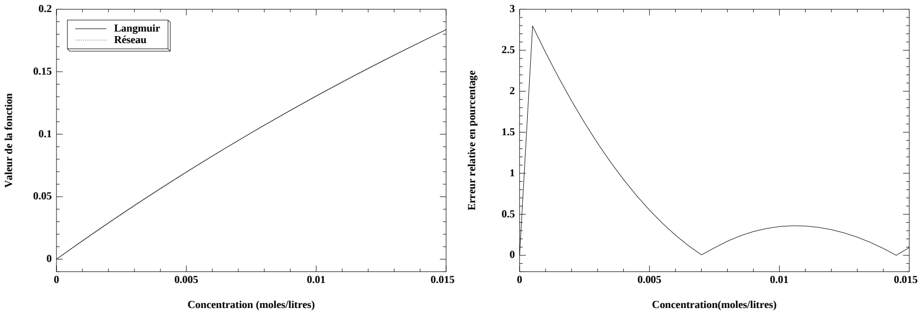



Figure 3.7: *Identification de l'isotherme de Langmuir ($K = 15$) avec 50 pas d'espace sur le domaine* $[0, 0.015]$*: valeurs des deux fonctions isothermes (à gauche) et différence relative entre les deux fonctions isothermes (à droite) sur le domaine d'identification.*

On peut agrandir le domaine d'identification en changeant la concentration d'injection.

#### 3.5.1.2 Isotherme de Langmuir sur un grand domaine

Ce résultat représente le meilleur des 10 calculs effectués, chaque calcul faisant évoluer 50 réseaux de neurones feed-forward pendant 2000 générations. Le temps total de calcul est d'environ 6 heures CPU ce qui est supérieur à l'identification précédente; en effet, la complexité du schéma est d'environ 2200 et au total le nombre d'appels aux réseaux s'élève environ à 1.5 milliards.

| Caractéristiques de la colonne | Conditions expérimentales | | Isotherme à identifier |
|---|---|---|---|
| | Conditions d'injection | Conditions initiales | |
| $L = 25$ cm<br>$T = 300$ K<br>$\varepsilon = 0.5$ | $t_{inj} = 15$ s<br>$u_0 = 12.3$ cm/s<br>$c_{inj} = 0.1$ mol/l | $c_{init} = 0$ mol/l | Langmuir $K_1 = 15$ |

Table 3.4: *Chromatographie liquide-solide : ensemble des données utilisées dans la simulation.*

| Taille de la population: 50 | |
|---|---|
| Nombre maximal de générations: 2000 | |
| Mutations paramétriques | Mutations architecturales |
| $p\mathbf{W} =$ 0.35 | $p\mathbf{C}_+ =$ 0.15 |
| $p\beta =$ 0.15 | $p\mathbf{C}_- =$ 0.15 |
| $p\mathbf{W_{in}} =$ 0 | $p\mathbf{U}_+ =$ 0.1 |
| $p\mathbf{W_{out}} =$ 0 | $p\mathbf{U}_- =$ 0.1 |

Table 3.5: *Ensemble des données utilisées par l'algorithme d'évolution dans la simulation.*

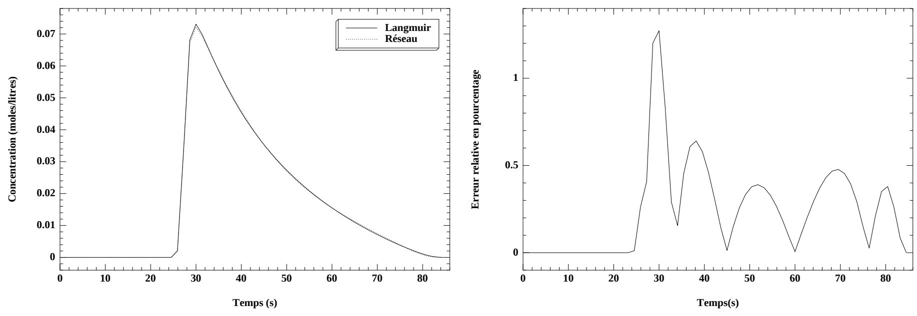



Figure 3.8: *Identification de l'isotherme de Langmuir ($K = 15$) avec 50 pas d'espace : chromatogrammes expérimental et identifié à gauche et erreur relative à droite.*

On remarque que contrairement aux tous premiers résultats obtenus sur ce même problème ([FS95], voir Annexe), on obtient un domaine de validité beaucoup plus grand (environ 20 fois) et il n'a pas été nécessaire de prendre en compte d'autres points test (on avait dû utiliser 9 points test pour le meilleur résultat). Tout d'abord, les différences de résultats s'expliquent par le fait que le nombre de générations est beaucoup plus elevé (2000 contre 150) dans les résultats

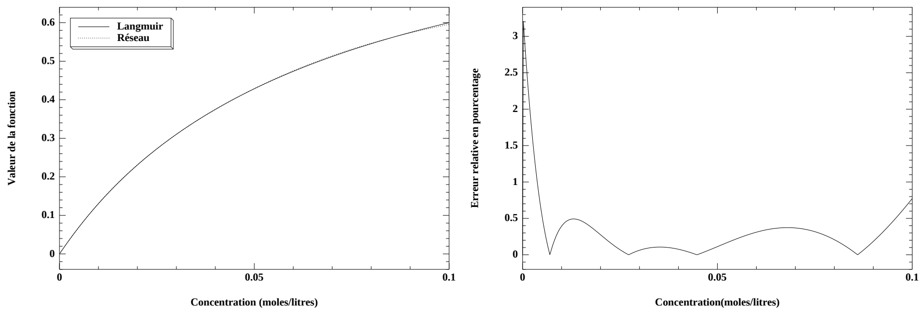



Figure 3.9: *Identification de l'isotherme de Langmuir ($K = 15$) avec 50 pas d'espace sur le domaine* $[0, 0.1]$*: valeurs des deux fonctions isothermes (à gauche) et différence entre les deux fonctions isothermes (à droite).*

de cette thèse. Ensuite, plusieurs modifications ont été depuis apportées à notre algorithme, ce qui n'est sans doute pas négligeable. Par ailleurs, la différence au niveau du nombre de points test est due au fait que le nombre de pas d'espace était beaucoup plus petit (environ 15) et donc le nombre de points du domaine d'identification était aussi faible, ce qui était alors compensé par un plus grand nombre de points test.

#### 3.5.1.3 Isotherme de Moreau-Valentin de degré 4

Si pour l'identification d'un isotherme de Langmuir le choix des réseaux de neurones feed-forward en tant que modèle d'isotherme est suffisant, il semble que, pour des modèles d'isothermes plus compliqués tels que l'isotherme de Moreau-Valentin, il est nécessaire d'utiliser les réseaux de neurones récurrents.
Pour aboutir à ce résultat, nous avons effectué 80 calculs dont 40 en utilisant les réseaux feed-forward et 40 en utilisant les réseaux récurrents (chaque calcul faisant évoluer 40 individus pendant 2000 générations). Nous présentons ci-dessous le meilleur résultat qui a été obtenu avec un réseau récurrent. On étudiera en détail dans le paragraphe suivant les résultats de cette comparaison entre les deux types de réseaux ainsi que les conclusions qu'on peut en tirer.

| Caractéristiques de la colonne | Conditions expérimentales | | Isotherme à identifier |
|---|---|---|---|
| | Conditions d'injection | Conditions initiales | |
| $L = 25$ cm<br>$T = 300$ K<br>$\varepsilon = 0.5$ | $t_{inj} = 35$ s<br>$u_0 = 12.3$ cm/s<br>$c_{inj} = 0.015$ mol/l | $c_{init} = 0$ mol/l | Moreau de degré 4<br>$K_1 = 15$<br>$E_2 =$ -1000<br>$E_3 = 1000$<br>$E_4 =$ -1000 |

Table 3.6: *Chromatographie liquide-solide : ensemble des données utilisées dans la simulation.*

| Taille de la population: 40 | |
|---|---|
| Nombre maximal de générations: 2000 | |
| Mutations paramétriques | Mutations architecturales |
| $p\mathbf{W} =$ 0.35 | $p\mathbf{C}_+ =$ 0.15 |
| $p\beta =$ 0.15 | $p\mathbf{C}_- =$ 0.15 |
| $p\mathbf{W_{in}} =$ 0 | $p\mathbf{U}_+ =$ 0.1 |
| $p\mathbf{W_{out}} =$ 0 | $p\mathbf{U}_- =$ 0.1 |

Table 3.7: *Ensemble des données utilisées par l'algorithme d'évolution dans la simulation.*

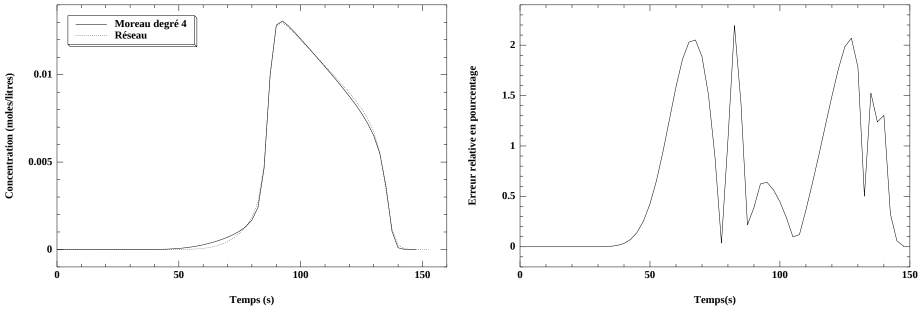



Figure 3.10: *Identification de l'isotherme de Moreau de degré 4 ($K = 15, E_2 = -1000, E_3 = 1000, E_4 = -1000$) avec 50 pas d'espace : chromatogrammes expérimental et identifié à gauche et erreur relative à droite.*

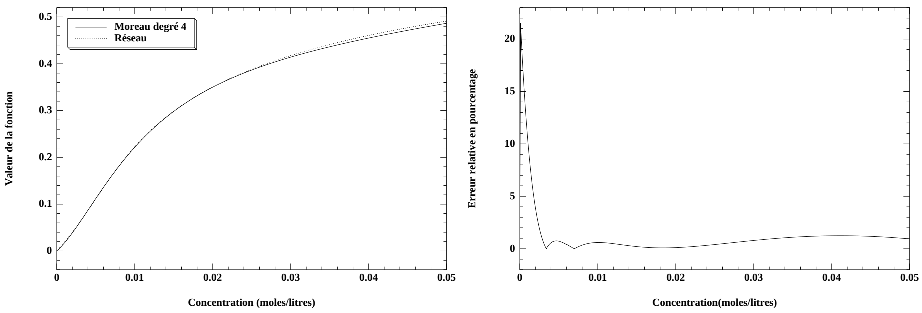



Figure 3.11: *Identification de l'isotherme de Moreau de degré 4 sur le domaine* $[0, 0.015]$*: valeurs des deux fonctions isothermes (à gauche) et différence relative entre les deux fonctions isothermes (à droite) sur l'intervalle* $[0, 0.05]$.

On remarque que même hors du domaine d'identification (voir figure 3.12), l'isotherme de Moreau-Valentin est bien identifé. Cependant, si l'on veut obtenir de manière sûre un domaine de validité plus grand, il faut changer les conditions expérimentales.

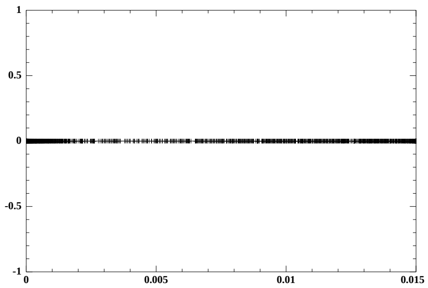



Figure 3.12: *Domaine d'identification.*

#### 3.5.1.4 Isotherme de Moreau-Valentin sur un grand domaine

Jusqu'à présent, nous avons pris en compte un seul point test pour calculer la performance d'un réseau. Cependant, pour ce problème, le manque de résultats nous a conduit à utiliser 2 points test, c'est-à-dire que l'on a pris en compte deux chromatogrammes dans le calcul de la performance (voir (3.5) dans le paragraphe 3.3.2).
Ce résultat représente le meilleur des 10 calculs effectués, chaque calcul faisant évoluer 50 réseaux de neurones récurrents pendant 2000 générations. Le temps total de calcul est d'environ 55 heures CPU. Cela s'explique tout d'abord par une complexité trés elevée : on a deux points test ce qui donne une somme des complexités d'environ 4000 (2200+1800) et au total le programme a fait appel plus de 3 milliards de fois aux réseaux de neurones. Par ailleurs l'appel à un réseau récurrent prend plus de temps que l'appel à un réseau feed-forward.

| Caractéristiques de la colonne | Conditions expérimentales | | Isotherme à identifier |
|---|---|---|---|
| | Conditions d'injection | Conditions initiales | |
| $L$ = 25 cm<br>$T$ = 300 K<br>$\varepsilon$ = 0.5 | $t_{inj}$ = 15 s<br>$u_0$ = 12.3 cm/s<br>$c^0_{inj}$ = 0.1 mol/l<br>$c^1_{inj}$ = 0.015 mol/l | $c^0_{init}$ = 0 mol/l<br>$c^1_{init}$ = 0 mol/l | Moreau de degré 4<br>$K_1$ = 15<br>$E_2$ = -1000<br>$E_3$ = 1000<br>$E_4$ = -1000 |

Table 3.8: *Chromatographie liquide-solide : ensemble des données utilisées dans la simulation.*

Le choix du deuxième point test n'est évidemment pas hasardeux mais résulte du raisonnement suivant. Nous avons tout d'abord lancé ces mêmes calculs mais en utilisant uniquement le premier point test (voir table 3.8), obtenant des résultats assez moyens sur le domaine [0,0.1]. A partir du meilleur isotherme obtenu, nous avons simulé d'autres propagations avec d'autres conditions expérimentales, ce qui nous a permis de mettre en évidence une zone commune où cet isotherme n'était pas correctement identifié en comparant les chromatogrammes obtenus avec les chromatogrammes corrects (obtenus en simulant avec les mêmes conditions expérimentales et l'isotherme que l'on doit identifier). Il est important de souligner que ce raisonnement se fait au niveau des chromatogrammes et non pas au niveau des isothermes qui ne sont pas connus a priori. Par ailleurs, comme le montrent les figures 3.15, les domaines d'identification pour un et

| Taille de la population: 50 | |
|---|---|
| Nombre maximal de générations: 2000 | |
| Mutations paramétriques | Mutations architecturales |
| $p\mathbf{W} =$ 0.35 | $p\mathbf{C}_{+}=$ 0.15 |
| $p\beta =$ 0.15 | $p\mathbf{C}_{-}=$ 0.15 |
| $p\mathbf{W_{in}} =$ 0 | $p\mathbf{U}_{+}=$ 0.1 |
| $p\mathbf{W_{out}} =$ 0 | $p\mathbf{U}_{-}=$ 0.1 |

Table 3.9: *Ensemble des données utilisées par l'algorithme d'évolution dans la simulation.*

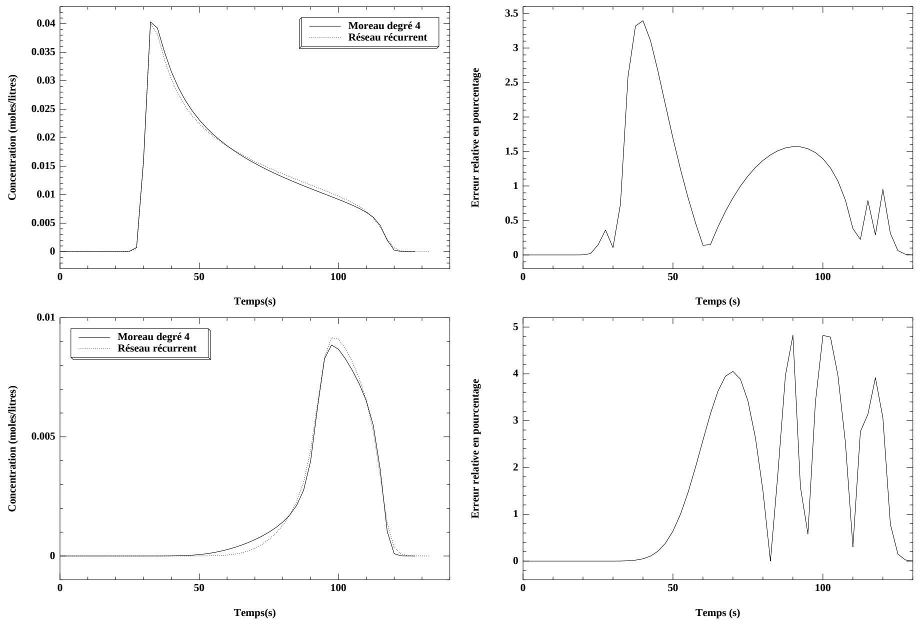



Figure 3.13: *Identification de l'isotherme de Moreau de degré 4 avec 50 pas d'espace : chromatogramme expérimental et chromatogramme identifié pour chaque point test à gauche et erreur relative à droite.*

deux points test ne sont pas vraiment différents. Cependant la différence de résultats s'explique par le fait que, pour un point test, l'algorithme d'évolution arrive à identifier plus facilement un isotherme dont la forme générale est correcte sauf sur la zone citée plus haut plutôt qu'un isotherme correct sur cette zone et moins bon ailleurs. Le fait de rajouter un point test oblige en quelque sorte l'algorithme à trouver un réseau qui identifie correctement l'isotherme sur tout le domaine. En fait, si l'on veut identifier l'isotherme sur le domaine [0,0.1], il semble qu'un troisième point test soit nécessaire à moins de changer le premier comme on aurait pu le deviner en comparant les domaines d'identification pour un et deux points. En effet, les autres calculs qui ont donné de bons résultats ainsi que les meilleurs identifications pour les autres méthodes évolutionnaires (voir paragraphes 3.6.1.2 et 3.7.1.1) présentent le défaut de ne pas identifier trés bien l'isotherme au voisinage de 0.1.

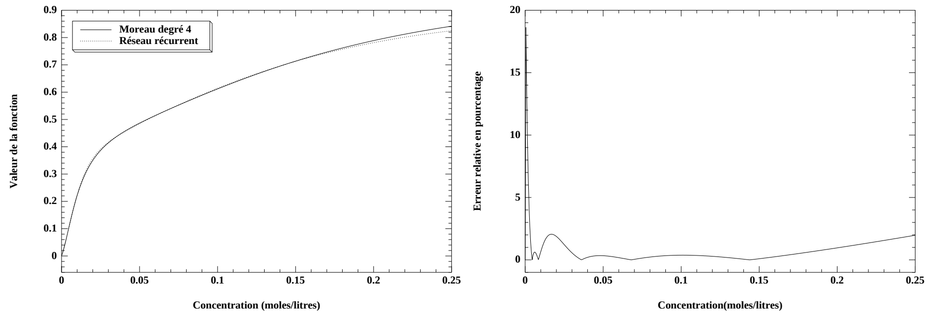



Figure 3.14: *Identification de l'isotherme de Moreau de degré 4 sur le domaine* $[0, 0.1]$*: valeurs des deux fonctions isothermes (à gauche) et différence relative entre les deux fonctions isothermes (à droite) sur l'intervalle* $[0, 0.25]$.

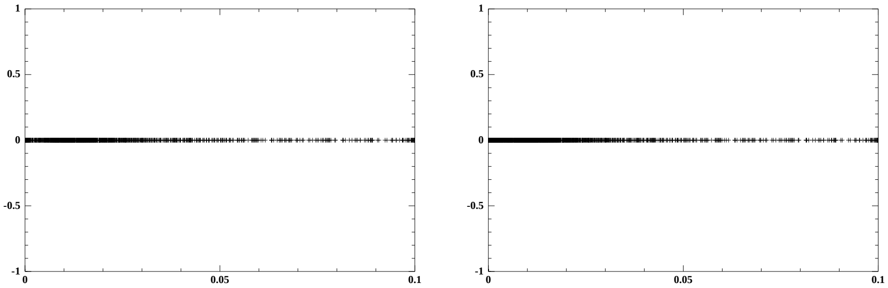


Figure 3.15: *Domaine d'identification pour 1 point test (à gauche) et 2 points test (à droite).*

#### 3.5.1.5 Comparaison entre les réseaux récurrents et les réseaux feed-forward

Cette comparaison a été faite pour une identification d'un isotherme de Moreau-Valentin de degré 4. Pour chaque type de réseau, on a effectué 40 calculs. Le meilleur résultat est un réseau récurrent. Tout d'abord on montre les meilleurs résultats obtenus dans chaque cas.

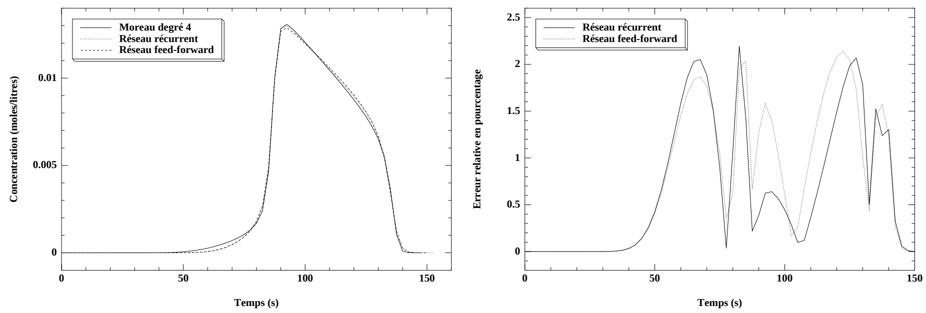



Figure 3.16: *Identification de l'isotherme de Moreau-Valentin : chromatogramme expérimental et chromatogramme identifié pour les deux types de réseaux.*

Bien que cela ne se voit pas sur les Figures ci-dessus, le meilleur résultat est un réseau récurrent avec une fitness de 0.0082 contre 0.009 pour le réseau feed-forward (plus la fitness

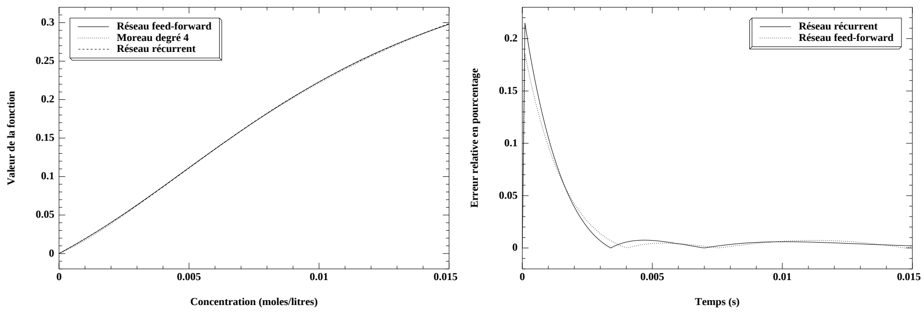



Figure 3.17: *Identification de l'isotherme de Moreau-Valentin sur le domaine* $[0, 0.015]$*: valeurs des trois fonctions isothermes à gauche et différence à droite.*

est petite plus l'identification est précise). On peut donc dire que, pour les meilleurs résultats obtenus, il n'y a aucune différence entre les deux cas de figure. Cependant les résultats ci-dessous vont nous permettre de conclure à une supériorité des réseaux récurrents.

La figure (3.18) nous donne la répartition de la fitness dans les deux cas et pour les 40 calculs.

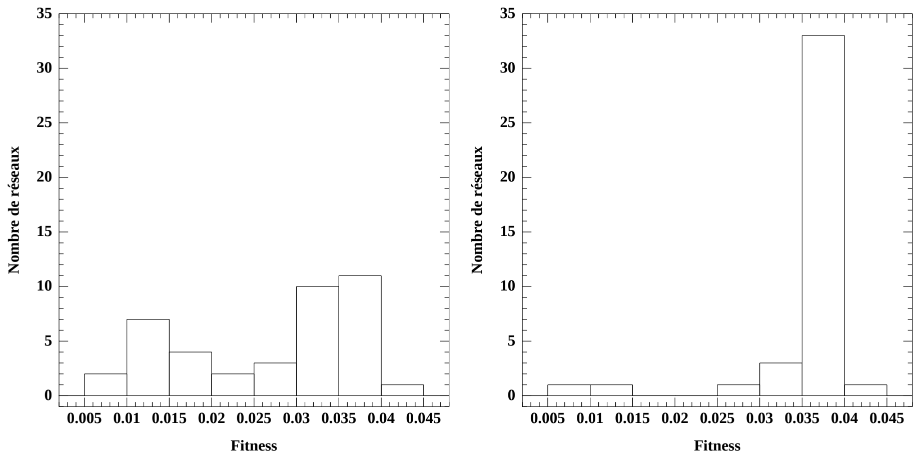



Figure 3.18: *Nombre de réseaux en fonction de la fitness pour les* 80 *calculs. A gauche, les réseaux récurrents, à droite les réseaux multi-couches (exemple: il y a* 7 *réseaux récurrents dont la fitness est comprise entre* 0.01 *et* 0.015*).*

La figure (3.19) représente les courbes d'évolution pour les meilleures évolutions dans les deux cas ainsi que la moyenne sur 40 calculs de l'évolution de la fitness minimum dans chaque cas.

Si l'on considère qu'une fitness inférieure à 0.015 donne un résultat assez bon, on obtiendra donc sur 80 calculs (40 avec des réseaux récurrents et 40 avec des réseaux feed-forward) 9 réseaux récurrents corrects dont 2 très bons (performance inferieure a 0.001) et seulement 2 réseaux feed-forward corrects dont un très bon. De plus, au vu de la figure (3.18), il semble que pour ce

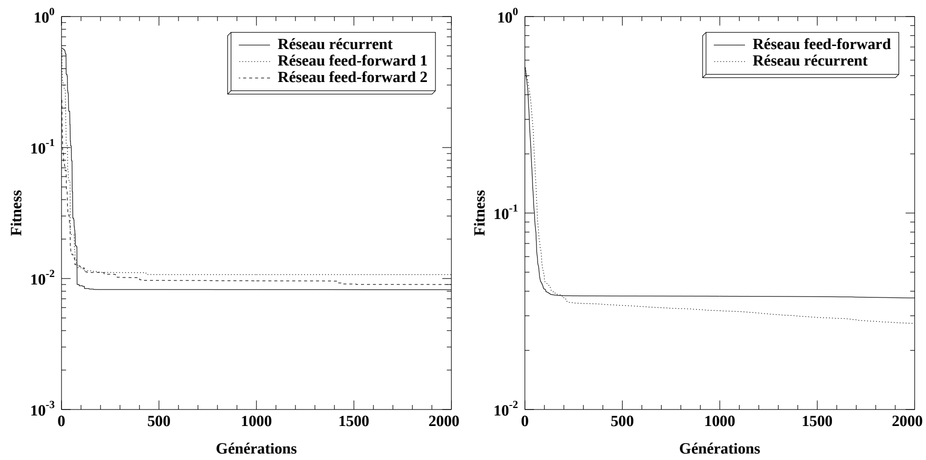



Figure 3.19: *Meilleures évolutions sur les* 40 *calculs (à gauche) et moyenne sur les* 40 *calculs de l'évolution de la fitness minimum (à droite).*

problème les réseaux de neurones feed-forward soient limités : environ 95% des réseaux obtenus sont mauvais contre environ 60% pour les réseaux récurrents. S'ajoute à cela qu'en moyenne on aura toujours un meilleur résultat en utilisant les réseaux récurrents. On peut donc conclure que, pour l'identification d'un isotherme de Moreau de degré assez élevé, il est préférable d'utiliser un réseau récurrent comme modèle. Cependant cette conclusion doit être nuancée par le fait qu'en moyenne le temps CPU nécessaire à 2 calculs utilisant les réseaux récurrents est égal au temps CPU nécessaire pour effectuer 9 calculs en utilisant les réseaux feed-forward.

#### 3.5.1.6 Conclusion intermédiaire

Les résultats présentés sur les différents problèmes d'identification à un corps sont excellents si l'on reste dans le cadre du problème où l'on doit identifier un isotherme avec un nombre de pas d'espace égal à 50, ce qui n'est pas suffisant pour pouvoir traiter des données réelles. Cependant, on voit que même avec 50 pas d'espace les calculs deviennent trop coûteux. De plus, le choix des conditions expérimentales dont dépend le domaine d'identification est important, d'une part car il détermine le domaine de validité de l'isotherme identifié, d'autre part lors de l'évolution (ajout d'un deuxième point test pour l'identification d'un isotherme de Moreau-Valentin de degré 4 sur un grand domaine). Par ailleurs, on a vu qu'il était préférable d'utiliser les réseaux récurrents comme modèle lorsque le degré de l'isotherme à identifier devient élevé, bien que le coût en temps de calcul soit alors environ quatre fois plus important que pour un réseau feed–forward.

### 3.5.2 Chromatographie liquide-solide à deux corps

Ce problème consiste à identifier une fonction de $\mathbb{R}^2$ dans $\mathbb{R}^2$. Plutôt que de choisir systéma tiquement un réseau feed-forward à deux entrées et deux sorties, on a aussi utilisé un réseau constitué de deux sous-réseaux feed-forward comportant chacun deux entrées et une sortie. Pour chaque cas nous avons effectué 25 calculs. Nous présentons dans ce paragraphe le meilleur résultat obtenu sur ces 50 calculs pour un réseau avec une structure comportant 2 sous-réseaux. On étudiera dans le paragraphe suivant les résultats de cette comparaison entre les deux types de réseaux.

On va voir que, pour un problème à deux corps (et plus), le domaine d'identification représente un problème dû au fait que la répartition des points à l'intérieur de ce domaine n'est pas uniforme et donc certains points vont avoir un poids plus important que d'autres.

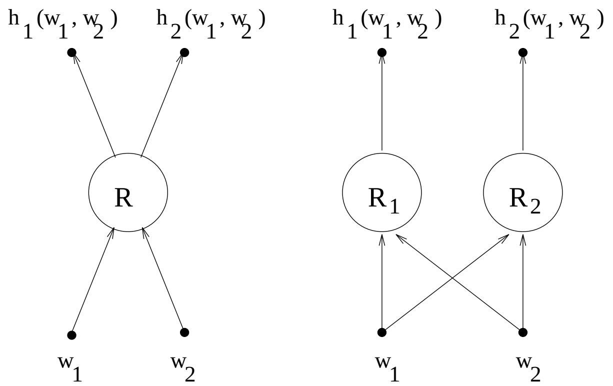



Figure 3.20: *Deux structures possibles pour le problème à deux corps.* $h = (h_1, h_2)$ *est la fonction isotherme.*

#### 3.5.2.1 Isotherme de Langmuir en utilisant un point test

Ce résultat représente le meilleur des 25 calculs effectués, chaque calcul faisant évoluer 35 individus pendant 3000 générations. Le temps total de calcul est d'environ 48 heures CPU. La complexité du schéma est d'environ 1300 et au total le programme a fait appel plus de 5 milliards de fois aux réseaux de neurones (on a multiplié la complexité par 2 car on a 2 sous-réseaux).

| Caractéristiques de la colonne | Conditions expérimentales | | Isotherme à identifier |
|---|---|---|---|
| | Conditions d'injection | Conditions initiales | |
| $L = 25$ cm | $t_{inj} = 10$ s | | Langmuir |
| $T = 300$ K | $u_0 = 12.3$ cm/s | | $K_1 = 15$ |
| $\varepsilon = 0.5$ | $c_{inj} = (0.01,\ 0.0001)$ mol/l | $c_{init} =(0.0001,\ 0.09)$ mol/l | $K_2 = 25$ |

Table 3.10: *Chromatographie liquide-solide à deux corps : ensemble des données utilisées dans la simulation.*

| Taille de la population: 50 | |
|---|---|
| Nombre maximal de générations: 3000 | |
| Mutations paramétriques | Mutations architecturales |
| $p\mathbf{W} =$ 0.35 | $p\mathbf{C}_{+}=$ 0.11 |
| $p\beta =$ 0.15 | $p\mathbf{C}_{-}=$ 0.11 |
| $p\mathbf{W_{in}} =$ 0 | $p\mathbf{U}_{+}=$ 0.095 |
| $p\mathbf{W_{out}} =$ 0 | $p\mathbf{U}_{-}=$ 0.095 |
| $p\mathbf{GW} =$ 0 | |
| $p\mathbf{GW_{in}}=$ 0 | |
| $p\mathbf{GW_{out}}=$ 0 | |

Table 3.11: *Ensemble des données utilisées par l'algorithme d'évolution dans la simulation.*

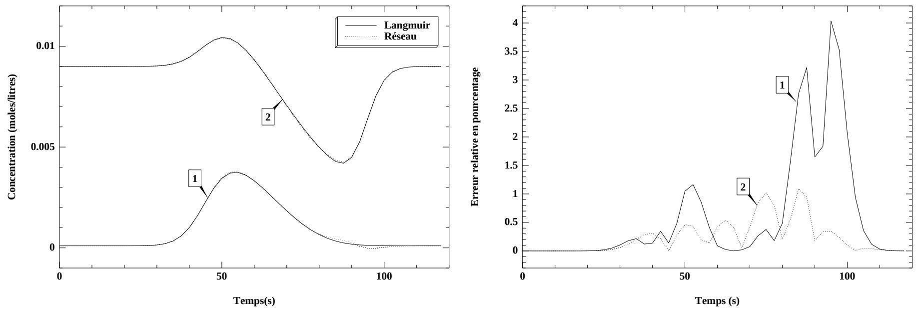



Figure 3.21: *Identification d'un isotherme de Langmuir ($K_1 = 15$ et $K_2 = 25$) avec 50 pas d'espace en utilisant 1 point test : chromatogramme expérimental et chromatogramme identifié à gauche et difference relative à droite.*

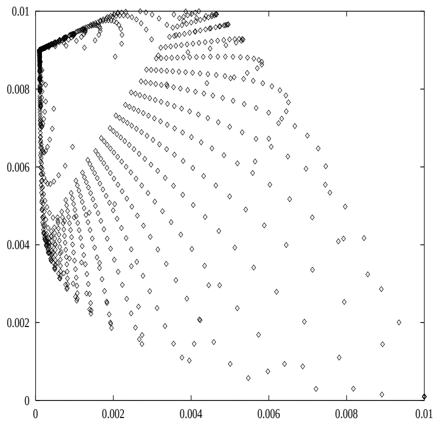


Figure 3.22: *Domaine d'identification avec un point test (en abscisses la concentration du corps 1 et en ordonnées celle du corps 2).*

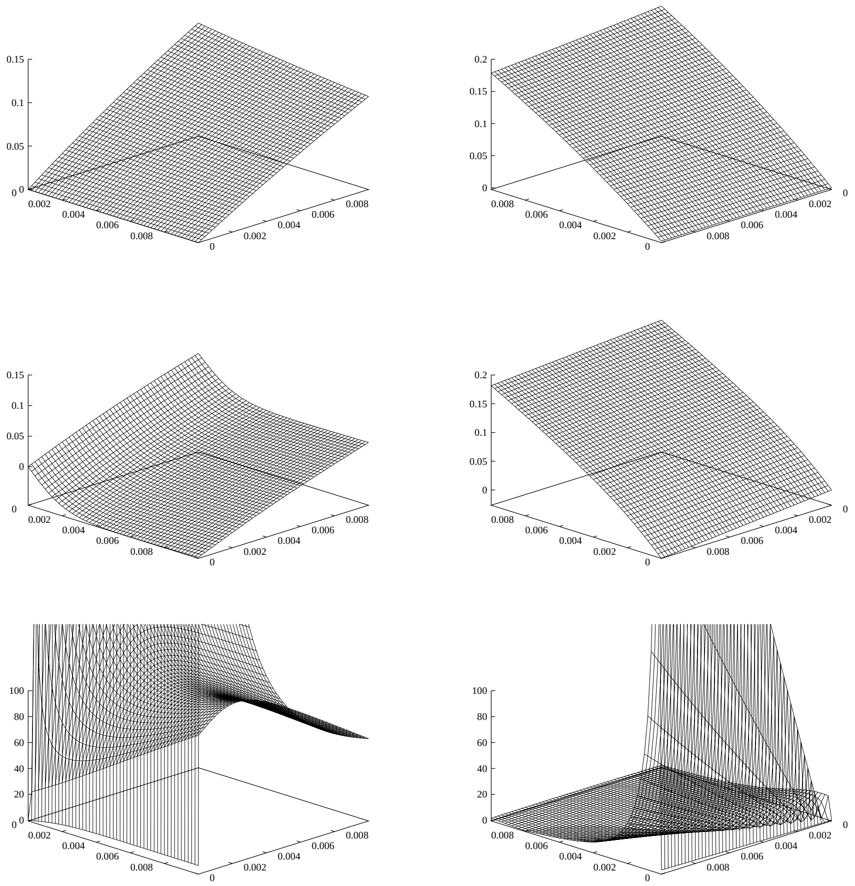


Figure 3.23: *Valeurs des deux fonctions isothermes expérimentales (en haut), valeurs des deux fonctions isothermes identifiées (au milieu) et différence relative en pourcentage (en bas).*

On remarque que, même si le chromatogramme est très bien identifié, l'isotherme ne l'est pas. C'est pour cela que l'on a effectué le même calcul mais en rajoutant un autre point test, non seulement pour agrandir le domaine d'identification mais aussi pour le rendre plus uniforme.

#### 3.5.2.2 Isotherme de Langmuir en utilisant deux points test

Les résultats des figures (3.24) et (3.25) représentent le meilleur des 10 calculs effectués, chaque calcul faisant évoluer 35 individus pendant 6000 générations. Le temps total de calcul est d'environ 80 heures CPU. La complexité du schéma est d'environ 2700 et au total le programme a fait appel environ 8 milliards de fois aux réseaux de neurones (on a multiplié la complexité par 2 car on a 2 sous-réseaux).

| Caractéristiques de la colonne | Conditions expérimentales | | Isotherme à identifier |
|---|---|---|---|
| | Conditions d'injection | Conditions initiales | |
| $L = 25$ cm | $t_{inj} = 10$ s | | Langmuir |
| $T = 300$ K | $u_0 = 12.3$ cm/s | | $K_1 = 15$ |
| $\varepsilon = 0.5$ | $c^0_{inj} = (0.01,\ 0.0001)$ mol/l | $c^0_{init} = (0.0001,\ 0.009)$ mol/l | $K_2 = 25$ |
| | $c^1_{inj} = (0.0001,\ 0.009)$ mol/l | $c^1_{init} = (0.01,\ 0.0001)$ mol/l | |

Table 3.12: *Chromatographie liquide-solide : ensemble des données utilisées dans la simulation.*

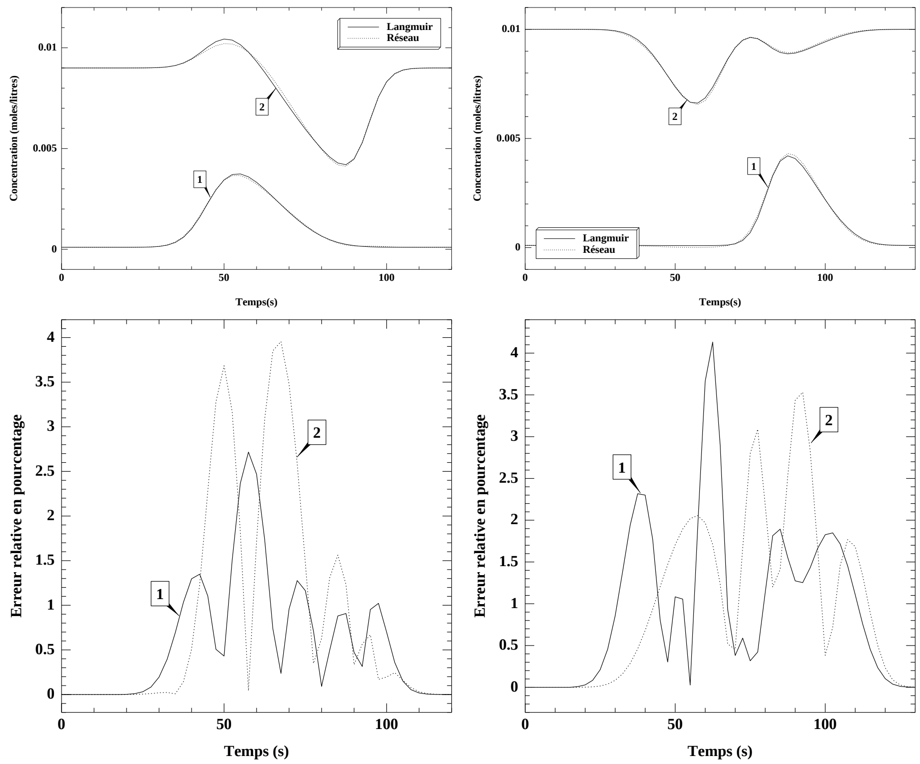



Figure 3.24: *Identification d'un isotherme de Langmuir ($K_1 = 15$ et $K_2 = 25$) avec 50 pas d'espace en utilisant 2 points test : chromatogramme expérimental et chromatogramme identifié pour chaque point test en haut et erreur relative en pourcentage bas.*

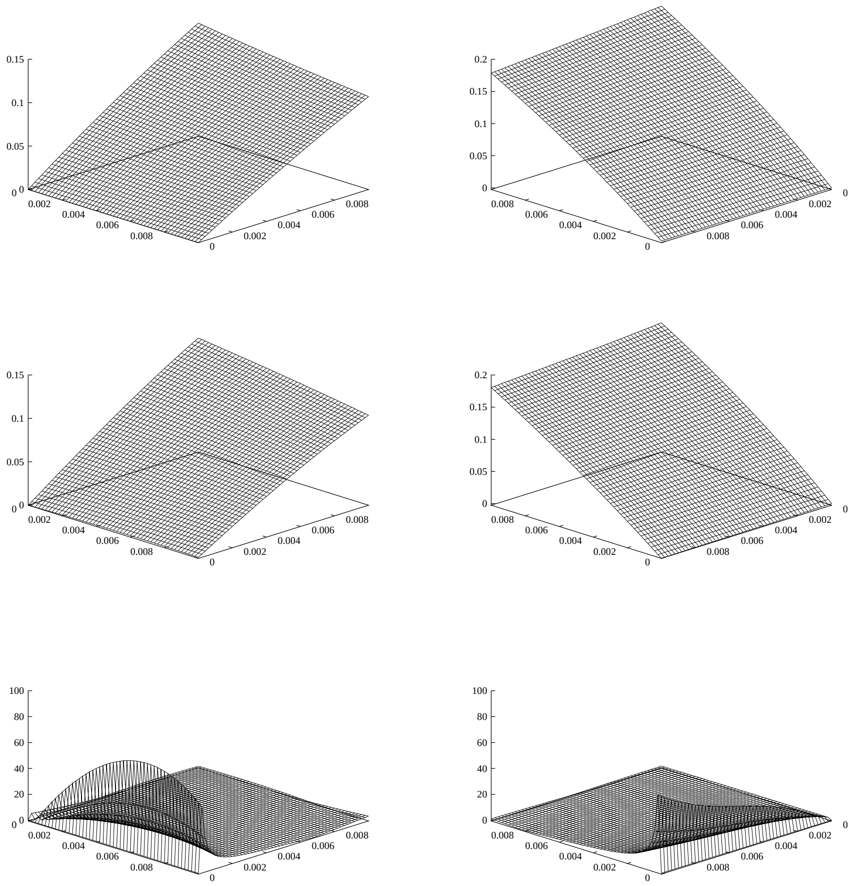



Figure 3.25: *Valeurs des deux fonctions isothermes expérimentales (en haut), valeurs des deux fonctions isothermes identifiées (au milieu) et différence relative (en bas).*

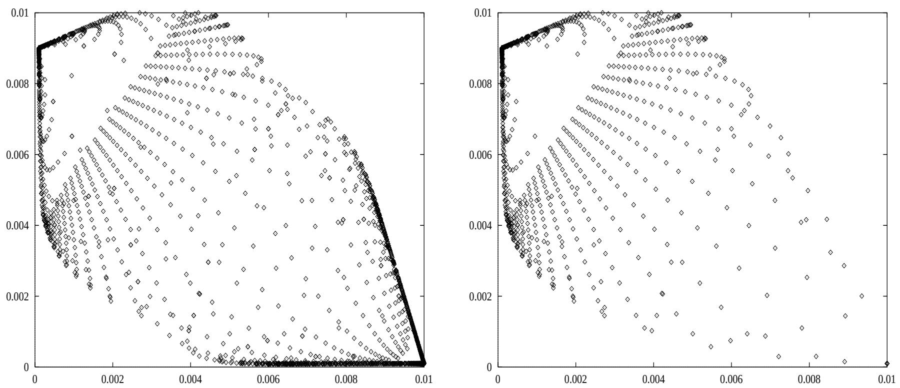



Figure 3.26: *Domaine d'identification avec 2 points test à gauche et celui avec 1 point test à droite (en abscisses la concentration du corps 1 et en ordonnées celle du corps 2).*

Un deuxième type d'identification pour un problème à deux corps sur un domaine d'identification beaucoup plus grand ([0,0.5]×[0,0.5]) a été testé sans cependant obtenir des résultats suffisamment bons pour être présentés.

#### 3.5.2.3 Comparaison entre les structures à un réseau et les structures à deux réseaux

Nous rappellons que cette comparaison a été faite pour une identification d'un isotherme de Langmuir ($K_1 = 15$ et $K_2 = 25$). Pour chaque type de réseau on a effectué 25 calculs. Le meilleur résultat est un réseau ayant une structure à deux sous-réseaux. Tout d'abord nous présentons les meilleurs résultats obtenus dans chaque cas.

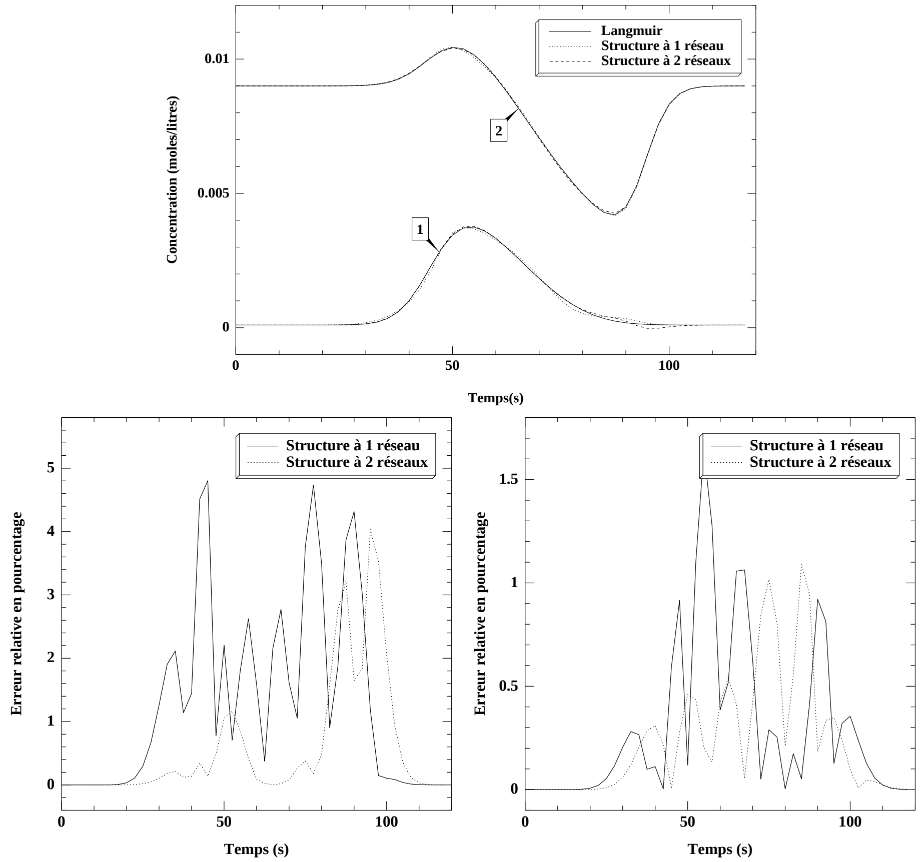



Figure 3.27: *Identification de l'isotherme de Langmuir ($K_1 = 15$ et $K_2 = 25$) : chromatogramme expérimental et chromatogramme identifié pour les deux types de réseau en haut et erreur relative pour les deux types de réseau (à gauche chromatogramme 1 et à droite chromatogramme 2).*

La figure (3.28) représente les courbes des meilleurs évolutions dans les deux cas ainsi que la moyenne sur 25 calculs de l'évolution de la fitness minimum dans chaque cas.

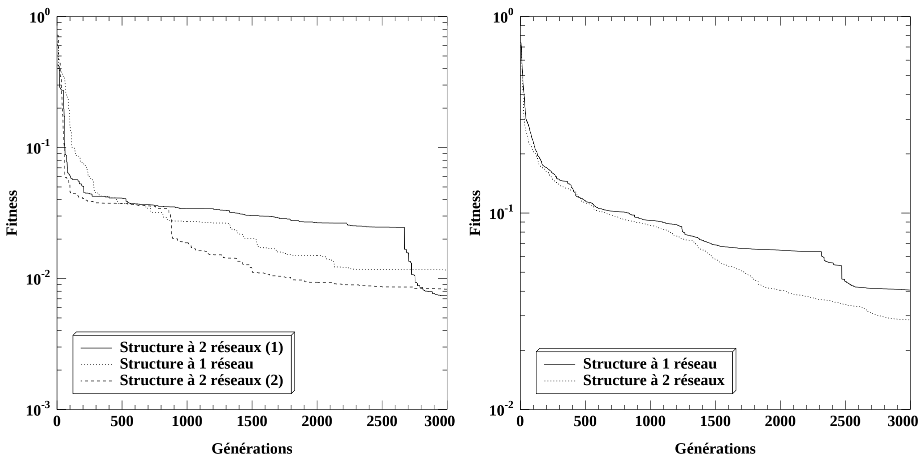



Figure 3.28: *Meilleurs évolutions sur les* 25 *calculs (à gauche) et moyenne sur les* 25 *calculs de l'évolution de la fitness minimum (à droite).*

La conclusion que nous pouvons faire suite à cette étude comparative est que l'on obtient clairement de meilleurs résultats lorsque le réseau possède une structure comportant deux sous-réseaux comme le montrent aussi bien les meilleurs résultats obtenus que les courbes d'évolution.

### 3.5.3 Conclusion

Au vu des résultats obtenus pour les problèmes à un et deux corps, on peut globalement conclure à la fiabilité de l'approche par les réseaux de neurones. Cependant, plusieurs remarques doivent etre faites : tout d'abord, les résultats pour le problème à deux corps n'ont pu être établis que pour un isotherme de Langmuir et sur un domaine de validité assez petit, contrairement aux résultats présentés pour un corps où l'on a pu identifier un isotherme de Moreau-Valentin de degré 4 sur un domaine assez grand. Les temps de calcul deviennent par ailleurs assez élevés même si le nombre de pas d'espace est relativement petit, ce qui ne laisse pas entrevoir pour l'instant une application sur des données réelles ; de plus, on a vu que le choix des conditions expérimentales est important, car d'une part ces conditions déterminent le domaine de validité de l'isotherme et d'autre part elles influent sur la qualité des résultats. Par ailleurs, on a vu que les réseaux récurrents sont plus performants que les réseaux feed-forward lorsque le degré de l'isotherme augmente avec cependant un coût en temps de calcul trés supérieur.
Dans le but d'obtenir de meilleurs résultats et d'abaisser les temps de calcul, nous avons couplé le modèle des réseaux de neurones avec une méthode d'interpolation classique : nous faisons toujours évoluer une population de réseaux de neurones, mais ces derniers ne serviront plus qu'au calcul des valeurs des points de contrôle. Ensuite, par interpolation, on pourra alors obtenir l'isotherme. L'idée de base de cette méthode est qu'il est sans doute plus facile pour un réseau d'identifier quelques valeurs d'une fonction plutôt que toute la fonction sur un certain intervalle.

## 3.6 Identification des valeurs des points de contrôle par un réseau de neurones (méthode *Réseau+Interpolation*)

Cette méthode utilise les fractions rationnelles comme modèle d'isotherme et un réseau de neurones est chargé d'identifier les valeurs des points de contrôle qui sont répartis uniformément sur le domaine d'identification.
Les identifications qui ont été faites avec cette méthode sont en partie les mêmes que celles effectuées avec la méthode qui utilise les réseaux de neurones comme modèle d'isotherme. Cependant, nous n'avons pas traité les deux problèmes d'identification des isothermes de Langmuir car ils ne présentent aucune difficulté lorsque le nombre de pas d'espace vaut 50. Le paragraphe suivant nous montre que même avec un nombre de pas d'espace élevé, on obtient d'excellents résultats.
Nous précisons que l'algorithme utilisé est celui de Bulirsch et Stoer [NRC].

### 3.6.1 Chromatographie liquide-solide à un corps

#### 3.6.1.1 Isotherme de Langmuir

Pour obtenir ce résultat, un seul calcul a été effectué. Ce calcul a fait évoluer 50 réseaux de neurones feed-forward pendant 2000 générations. Le temps total de calcul est d'environ 2 jours CPU sur un Pentium II à 300 Mhz.

| Caractéristiques de la colonne | Conditions expérimentales | | Isotherme à identifier |
|---|---|---|---|
| | Conditions d'injection | Conditions initiales | |
| $L = 25$ cm<br>$T = 300$ K<br>$\varepsilon = 0.5$ | $t_{inj} = 15$ s<br>$u_0 = 12.3$ cm/s<br>$c_{inj} = 0.5$ mol/l | $c_{init} = 0$ mol/l | Langmuir $K_1 = 15$ |

Table 3.13: *Chromatographie liquide-solide : ensemble des données utilisées dans la simulation.*

| Taille de la population: 50 | |
|---|---|
| Nombre maximal de générations: 2000 | |
| Mutations paramétriques | Mutations architecturales |
| $p\mathbf{W} =$ 0.35 | $p\mathbf{C}_+ =$ 0.15 |
| $p\beta =$ 0.15 | $p\mathbf{C}_- =$ 0.15 |
| $p\mathbf{W_{in}} =$ 0 | $p\mathbf{U}_+ =$ 0.1 |
| $p\mathbf{W_{out}} =$ 0 | $p\mathbf{U}_- =$ 0.1 |

Table 3.14: *Ensemble des données utilisées par l'algorithme d'évolution dans la simulation.*

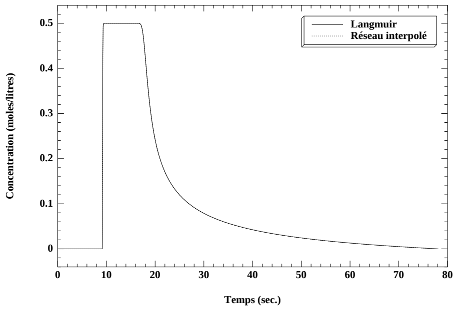



Figure 3.29: *Identification de l'isotherme de Langmuir ($K = 15$) avec 1000 pas d'espace : chromatogrammes expérimental et identifié. L'erreur relative en pourcentage est inférieure à* $10^{-6}$.

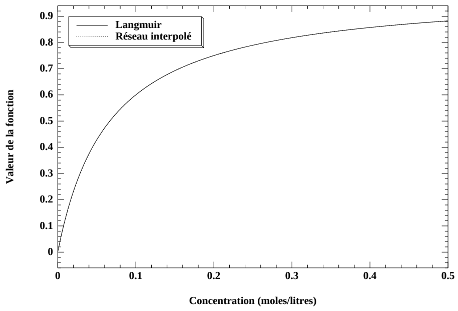



Figure 3.30: *Identification de l'isotherme de Langmuir ($K = 15$) avec 1000 pas d'espace sur le domaine* $[0, 0.5]$*: valeurs des deux fonctions isothermes. L'erreur relative en pourcentage est inférieure à* $10^{-9}$.

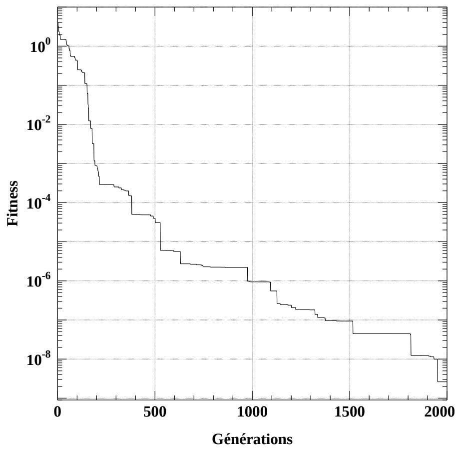



Figure 3.31: *Evolution de la fitness minimum.*

Comme le montrent les différentes figures, l'isotherme de Langmuir est parfaitement identifié lorsque le nombre de pas d'espace est élevé. De plus, les résultats obtenus au bout de 250 générations (moins de 7 heures CPU) sont déjà très bons : la fitness est de 0.000288, alors que pour la plupart des résultats obtenus sur ce même problème avec d'autres méthodes, celle-ci est au mieux 10 fois supérieure.

#### 3.6.1.2 Première identification d'un isotherme de Moreau-Valentin de degré 4

Nous avons refait exactement le même calcul que dans le paragraphe 3.5.1.4. Pour l'interpolation nous utilisons 9 points de contrôle répartis uniformément sur l'intervalle $[0, 0.1]$. Ce résultat représente le meilleur des 20 calculs effectués, chaque calcul faisant évoluer 50 réseaux de neurones récurrents pendant 2000 générations. Le temps total de calcul est d'environ 57 heures CPU et le nombre d'appels aux réseaux récurrents s'élève au total à 1 800 000.

| Caractéristiques de la colonne | Conditions expérimentales | | Isotherme à identifier |
|---|---|---|---|
| | Conditions d'injection | Conditions initiales | |
| $L = 25$ cm<br>$T = 300$ K<br>$\varepsilon = 0.5$ | $t_{inj} = 15$ s<br>$u_0 = 12.3$ cm/s<br>$c^0_{inj} = 0.1$ mol/l<br>$c^1_{inj} = 0.015$ mol/l | $c^0_{init} = 0$ mol/l<br>$c^1_{init} = 0$ mol/l | Moreau de degré 4<br>$K_1 = 15$<br>$E_2 = -1000$<br>$E_3 = 1000$<br>$E_4 = -1000$ |

Table 3.15: *Chromatographie liquide-solide : ensemble des données utilisées dans la simulation.*

| Taille de la population: 50 | |
|---|---|
| Nombre maximal de générations: 2000 | |
| Mutations paramétriques | Mutations architecturales |
| $p\mathbf{W} =$ 0.35 | $p\mathbf{C}_+ =$ 0.15 |
| $p\beta =$ 0.15 | $p\mathbf{C}_- =$ 0.15 |
| $p\mathbf{W_{in}} =$ 0 | $p\mathbf{U}_+ =$ 0.1 |
| $p\mathbf{W_{out}} =$ 0 | $p\mathbf{U}_- =$ 0.1 |

Table 3.16: *Ensemble des données utilisées par l'algorithme d'évolution dans la simulation.*

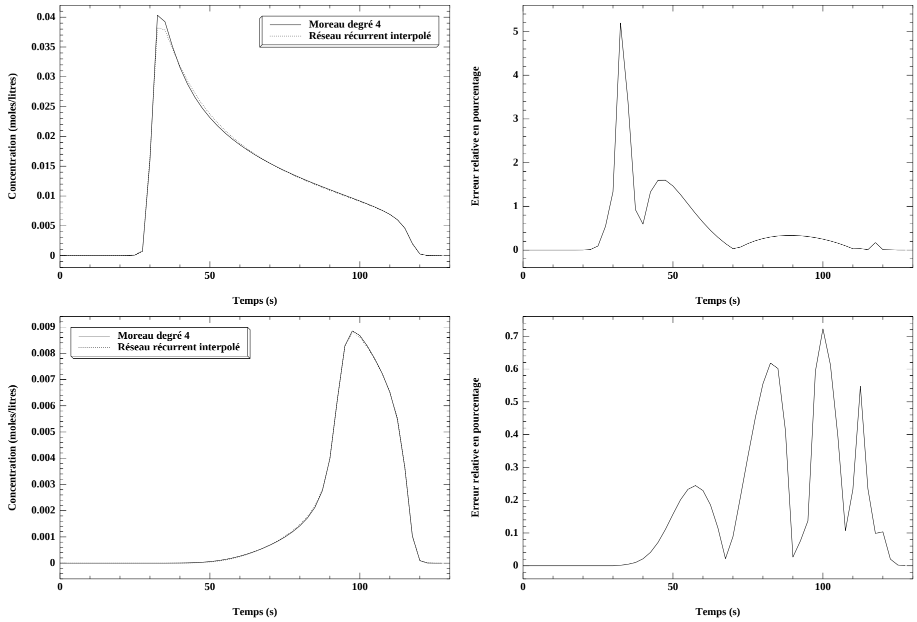



Figure 3.32: *Identification de l'isotherme de Moreau de degré 4 avec 50 pas d'espace : chromatogramme expérimental et chromatogramme identifié pour chaque point test en haut et erreur relative en bas.*

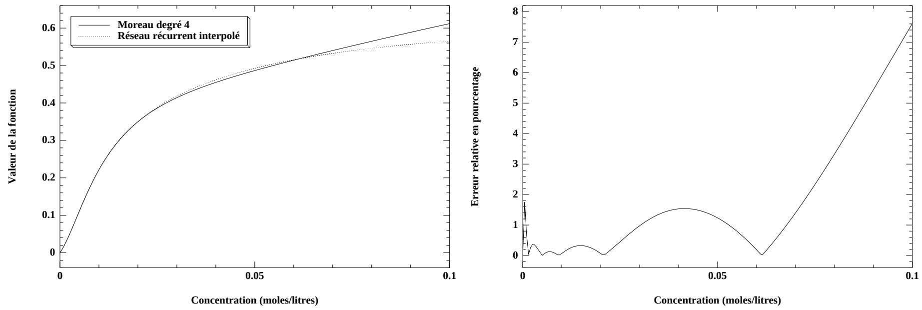



Figure 3.33: *Identification de l'isotherme de Moreau de degré 4 sur le domaine* $[0, 0.1]$*: valeurs des deux fonctions isothermes (à gauche) et différence relative entre les deux fonctions isothermes (à droite).*

Comme le montre la figure 3.33, l'isotherme n'est pas très bien identifié au voisinage du point 0.1 bien que la fitness soit meilleure que pour le résultat obtenu en utilisant un réseau de neurones comme modèle d'isotherme (fitness de 0.0137 et 0.0057). De plus, comme les autres bons résultats obtenus présentent le même défaut, cela nous laisse penser que si l'on veut identifier l'isotherme sur le domaine [0,0.1], il faut alors utiliser d'autres conditions expérimentales (une condition d'injection plus grande que 0.1 pour le premier point test) ou bien rajouter un autre point test. Nous avons choisi de rajouter un troisième point test en refaisant le même raisonnement que dans la section 3.5.1.4.

#### 3.6.1.3 Deuxième identification d'un isotherme de Moreau-Valentin de degré 4

Ce résultat représente le meilleur des 40 calculs effectués, chaque calcul faisant évoluer 50 réseaux de neurones récurrents pendant 7500 générations. Le temps total de calcul est d'environ 10 jours CPU.

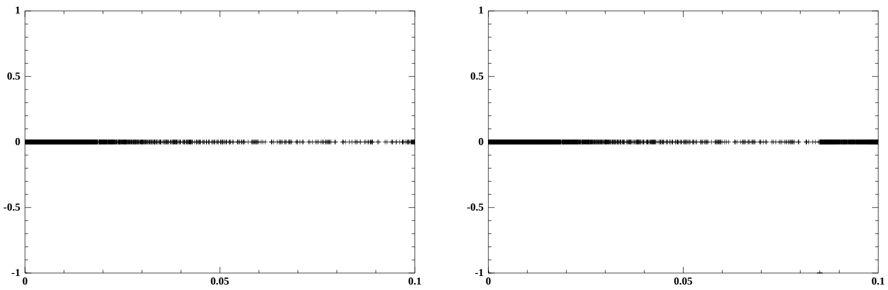



Figure 3.34: *Domaine d'identification pour 2 points test (à gauche) et 3 points test (à droite).*

| Caractéristiques de la colonne | Conditions expérimentales | | Isotherme à identifier |
|---|---|---|---|
| | Conditions d'injection | Conditions initiales | |
| $L = 25$ cm<br>$T = 300$ K<br>$\varepsilon = 0.5$ | $t_{inj} = 15$ s<br>$u_0 = 12.3$ cm/s<br>$c^0_{inj} = 0.1$ mol/l<br>$c^1_{inj} = 0.015$ mol/l<br>$c^1_{inj} = 0.085$ mol/l | $c^0_{init} = 0$ mol/l<br>$c^1_{init} = 0$ mol/l<br>$c^1_{init} = 0.1$ mol/l | Moreau de degré 4<br>$K_1 = 15$<br>$E_2 =$ -1000<br>$E_3 = 1000$<br>$E_4 =$ -1000 |

Table 3.17: *Chromatographie liquide-solide : ensemble des données utilisées dans la simulation.*

| Taille de la population: 50 | |
|---|---|
| Nombre maximal de générations: 7500 | |
| Mutations paramétriques | Mutations architecturales |
| $p\mathbf{W} =$ 0.35 | $p\mathbf{C}_+ =$ 0.15 |
| $p\beta =$ 0.15 | $p\mathbf{C}_- =$ 0.15 |
| $p\mathbf{W_{in}} =$ 0 | $p\mathbf{U}_+ =$ 0.1 |
| $p\mathbf{W_{out}} =$ 0 | $p\mathbf{U}_- =$ 0.1 |

Table 3.18: *Ensemble des données utilisées par l'algorithme d'évolution dans la simulation.*

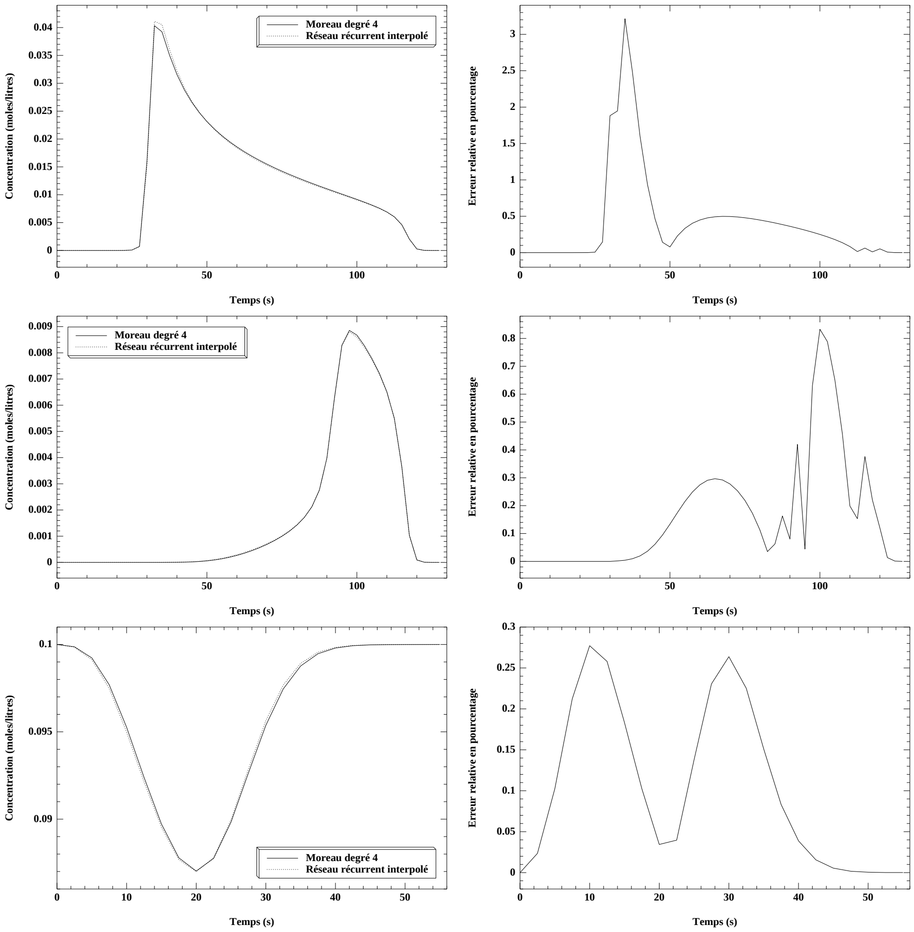



Figure 3.35: *Identification de l'isotherme de Moreau de degré 4 avec 50 pas d'espace : chromatogramme expérimental et chromatogramme identifié pour chaque point test à gauche et erreur relative à droite.*

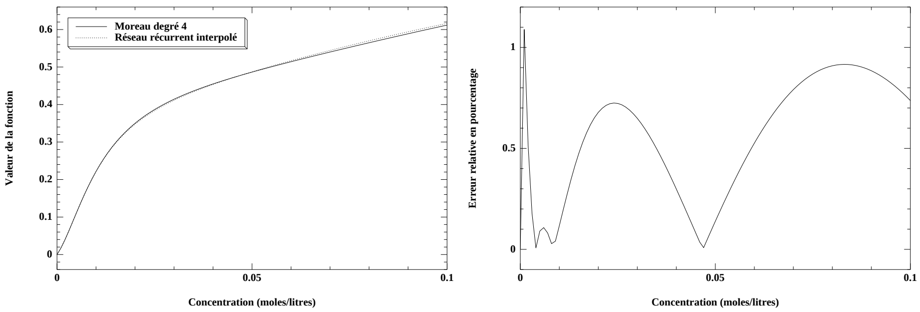



Figure 3.36: *Identification de l'isotherme de Moreau de degré 4 sur le domaine* $[0, 0.1]$*: valeurs des deux fonctions isothermes (à gauche) et différence relative entre les deux fonctions isothermes (à droite).*

Comme le montre les différentes figures, l'isotherme est trés bien identifié sur le domaine

$[0, 0.1]$ (moins de 1% d'erreur). On remarque par ailleurs que la comparaison avec le calcul précédent souligne encore l'influence des conditions expérimentales sur la qualité des résultats. On peut néamoins déjà conclure que pour un problème à un corps cette méthode donne de trés bons résulats, avec cependant les mêmes remarques que pour la méthode qui utilise uniquement les réseaux de neurones, c'est-à-dire des temps de calcul qui restent élevés pour un nombre de pas d'espace assez petit. On peut de plus commencer à établir une comparaison avec cette dernière méthode aussi bien en ce qui concerne les temps de calcul (qui sont moins élevés pour la méthode par interpolation) que sur les résultats obtenus. En effet, sur le problème d'identification de l'isotherme de Moreau-Valentin qui utilise deux points test, les temps de calculs sont environ deux fois moins élevés (20 calculs effectués en 57 heures contre 10 calculs effectués en 55 heures pour 9 points de contrôle et en utilisant des réseaux récurrents). De plus, l'identification d'un isotherme de Langmuir avec 1000 pas d'espace montre la supériorité des résultats par rapport à l'autre méthode. Cependant, il faut souligner que l'on a supposé connu le nombre de points de contrôle, ce qui suppose le connaissance d'informations supplémentaires sur l'isotherme à identifier.

### 3.6.2 Chromatographie liquide-solide à deux corps

Les résultats des figures (3.38) et (3.39) représentent le meilleur des 40 calculs effectués, chaque calcul faisant évoluer 40 réseaux de neurones feed-forward (constitués de deux sous-réseaux) pendant 4000 générations et en utilisant deux points test. Pour l'interpolation, nous utilisons 9 points de contrôle répartis uniformément sur le domaine $[0, 0.5] \times [0, 0.5]$. Le temps total de calcul est d'environ 6 jours CPU.

| Caractéristiques de la colonne | Conditions expérimentales | | Isotherme à identifier |
|---|---|---|---|
| | Conditions d'injection | Conditions initiales | |
| $L = 25$ cm<br>$T = 300$ K<br>$\varepsilon = 0.5$ | $t_{inj} = 10$ s<br>$u_0 = 50$ cm/s<br>$c^0_{inj} = (0.5,\ 0.0005)$ mol/l<br>$c^1_{inj} = (0.5,\ 0.5)$ mol/l | $c^0_{init} = (0,\ 0.5)$ mol/l<br>$c^1_{init} = (0,\ 0)$ mol/l | Langmuir<br>$K_1 = 15$<br>$K_2 = 25$ |

Table 3.19: *Chromatographie liquide-solide : ensemble des données utilisées dans la simulation.*

| Taille de la population: 50 | |
|---|---|
| Nombre maximal de générations: 4000 | |
| Mutations paramétriques | Mutations architecturales |
| $p\mathbf{W} =$ 0.35 | $p\mathbf{C}_{+}=$ 0.11 |
| $p\beta =$ 0.15 | $p\mathbf{C}_{-}=$ 0.11 |
| $p\mathbf{W_{in}} =$ 0 | $p\mathbf{U}_{+}=$ 0.095 |
| $p\mathbf{W_{out}} =$ 0 | $p\mathbf{U}_{-}=$ 0.095 |
| $p\mathbf{GW} =$ 0 | |
| $p\mathbf{GW_{in}}=$ 0 | |
| $p\mathbf{GW_{out}}=$ 0 | |

Table 3.20: *Ensemble des données utilisées par l'algorithme d'évolution dans la simulation.*

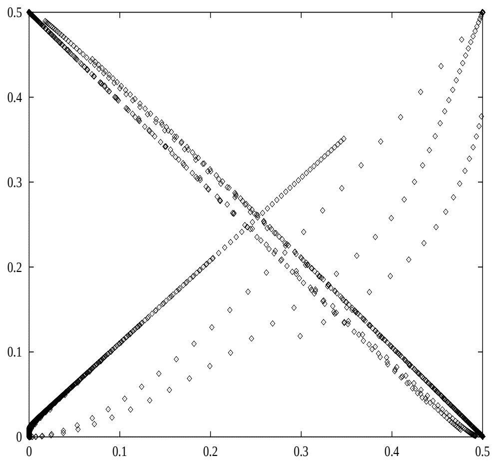



Figure 3.37: *Domaine d'identification*

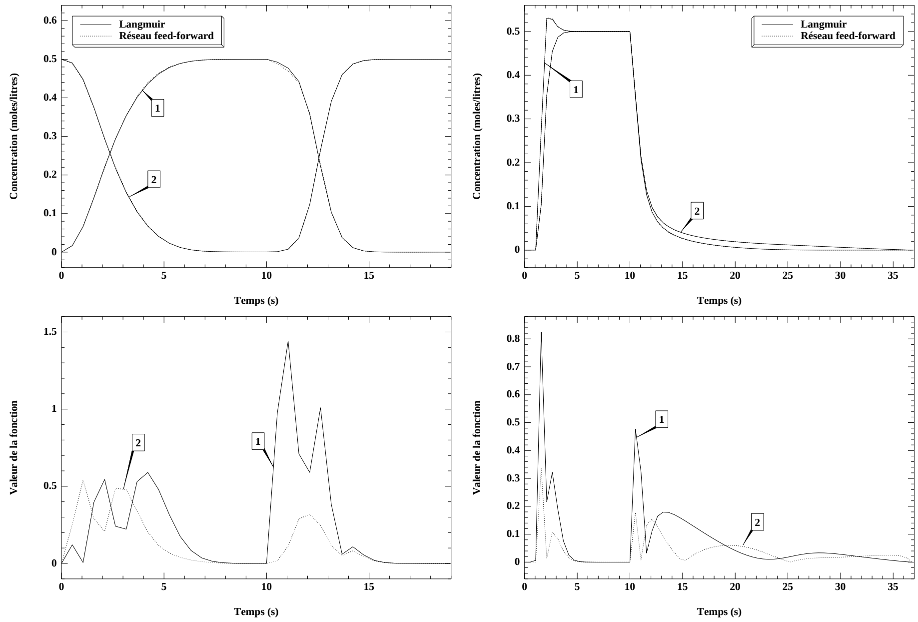



Figure 3.38: *Identification d'un isotherme de Langmuir ($K_1 = 15$ et $K_2 = 25$) avec 50 pas d'espace en utilisant 2 points test : chromatogramme expérimental et chromatogramme identifié pour chaque point test en haut et erreur relative en pourcentage en bas.*

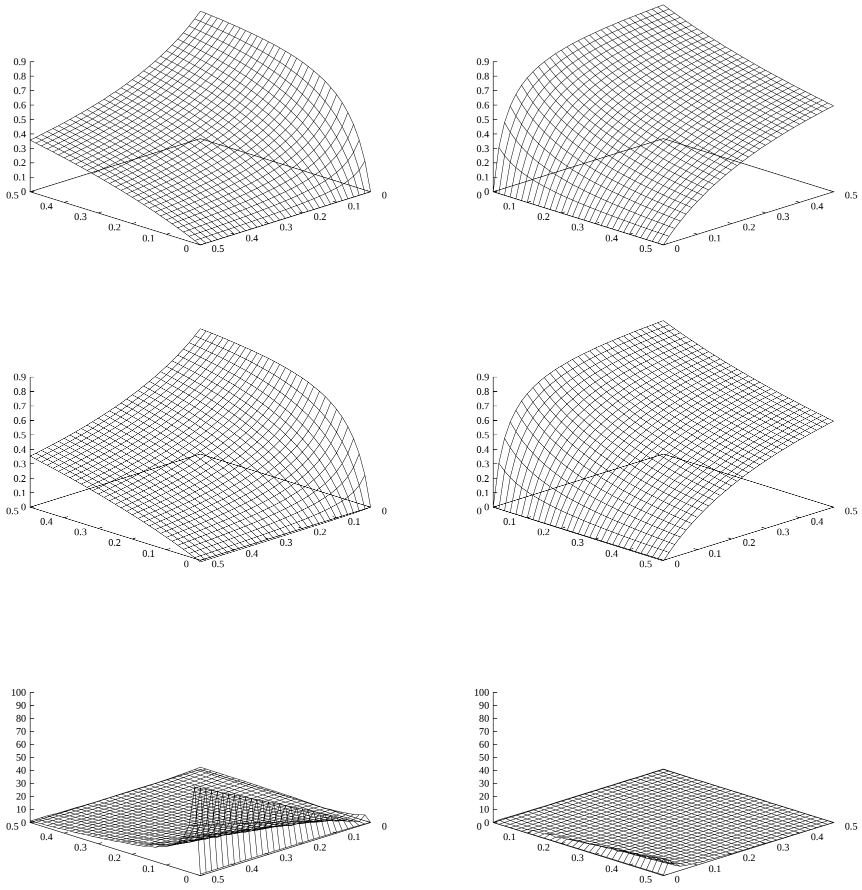


Figure 3.39: *Valeurs des deux fonctions isothermes expérimentales (en haut), valeurs des deux fonctions isothermes identifiées (au milieu) et différence relative en pourcentage (en bas).*

Les figures 3.38 et 3.39 montrent d'excellents résultats pour cette identification dont le domaine de validité est de plus largement supérieur par rapport aux résultats obtenus jusqu'à présent. On constate cependant que c'est le seul trés bon résultat obtenu sur les 40 calculs effectués, ce qui nous montre que l'on commence à atteindre les limites de cette méthode. Par ailleurs, si on effectuait ce même calcul avec la méthode qui utilise uniquement les réseaux de neurones, on constaterait que les temps de calculs seraient environ deux fois plus élevés. En effet, le résultat à deux corps présenté pour cette dernière méthode a une complexité quatre fois inférieur par rapport à la complexité du problème de ce paragraphe et un temps de calcul environ deux fois inférieur.

### 3.6.3 Conclusion

L'ensemble des résultats obtenus aussi bien pour un que pour deux corps nous permet de conclure à la supériorité de cette méthode par rapport à celle qui utilise uniquement les réseaux de neurones, bien qu'elle requiert la connaissance du nombre de points de contrôle. Tout d'abord cette différence se situe au niveau des temps de calcul qui peuvent être comme on l'a vu deux

fois moins élevés aussi bien sur des problèmes à un ou deux corps.
On peut continuer le même raisonement qui nous a conduit à faire uniquement calculer par le réseau de neurones les valeurs des points de contrôle et à ensuite interpoler pour obtenir la fonction isotherme. Cependant on n'utilisera plus un réseau de neurones mais on identifiera directement les valeurs des points de contrôle par un algorithme de stratégie d'évolution (algorithme ES).

## 3.7 Identification des valeurs des points de contrôle par stratégies d'évolution (méthode *ES+Interpolation*)

Comme pour le paragraphe précédent, cette méthode utilise les fractions rationnelles comme modèle d'isotherme mais nous utiliserons un algorithme de stratégie d'évolution $(\lambda + \mu)$-ES pour identifier les valeurs des points de contrôle.
Les paramètres de l'algorithme que nous avons utilisés sont les suivant:

- Nous utilisons une stratégie (7+100)-ES : 7 parents génèrent 100 enfants et la population de la génération suivante sera constituée des 7 meilleurs parmi ces 107 individus.
- La probabilité de croisement est nulle et donc celle de mutation vaut 1.
- Nous considérons que la variance $\sigma$ utilisée lors de la mutation varie au cours de l'évolution. Ainsi si l'on note $X = \{X_i\}_{i=1}^{K}$ un individu avec $K$ le nombre de points de contrôle, alors, après mutation, on aura un nouvel individu $\tilde{X}$ vérifiant :

$$\left\{ \begin{array}{l} \tilde{\sigma} = \sigma \cdot e^{N(0,1)} \\ \tilde{X} = X + N(0, \tilde{\sigma}) \end{array} \right.$$

  La valeur de $\sigma$ au début de l'évolution est fixée à 0.2.
- Le nombre maximal de générations est égal à 1000. Dans tous les résultats qui suivent nous avons constaté que ce nombre a toujours été suffisant, la plupart des calculs s'arrêtant même avant (un calcul s'arrête lorsque la fitness ne diminue plus pendant 200 générations).

### 3.7.1 Chromatographie liquide-solide à un corps

#### 3.7.1.1 Première identification d'un isotherme de Moreau-Valentin de degré 4

Nous avons refait exactement le même calcul que dans les paragraphe 3.5.1.4 et 3.6.1.2. Pour l'interpolation nous utilisons 9 points de contrôle répartis uniformément sur l'intervalle $[0, 0.1]$. Ce résultat représente le meilleur des 20 calculs effectués, chaque calcul faisant évoluer 50 réseaux de neurones récurrents pendant 2000 générations. Le temps total de calcul est d'environ 36 heures CPU.

| Caractéristiques de la colonne | Conditions expérimentales | | Isotherme à identifier |
|---|---|---|---|
| | Conditions d'injection | Conditions initiales | |
| $L = 25$ cm | $t_{inj} = 15$ s | | Moreau de degré 4 |
| $T = 300$ K | $u_0 = 12.3$ cm/s | | $K_1 = 15$ |
| $\varepsilon = 0.5$ | $c^0_{inj} = 0.1$ mol/l | $c^0_{init} = 0$ mol/l | $E_2 =$ -1000 |
| | $c^1_{inj} = 0.015$ mol/l | $c^1_{init} = 0$ mol/l | $E_3 = 1000$ |
| | | | $E_4 =$ -1000 |

Table 3.21: *Chromatographie liquide-solide : ensemble des données utilisées dans la simulation.*

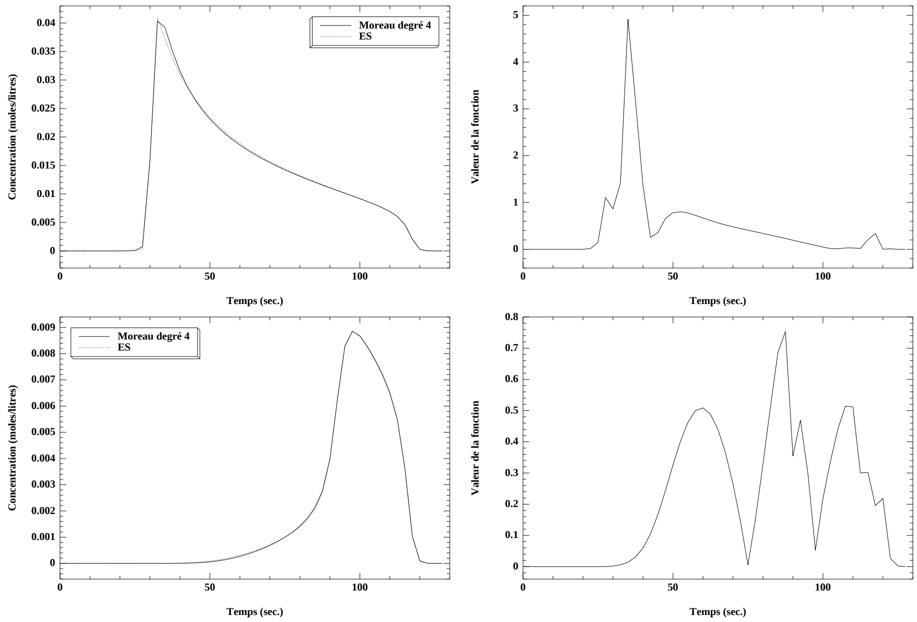



Figure 3.40: *Identification de l'isotherme de Moreau de degré 4 avec 50 pas d'espace : chromatogramme expérimental et chromatogramme identifié pour chaque point test en haut et erreur relative en bas.*

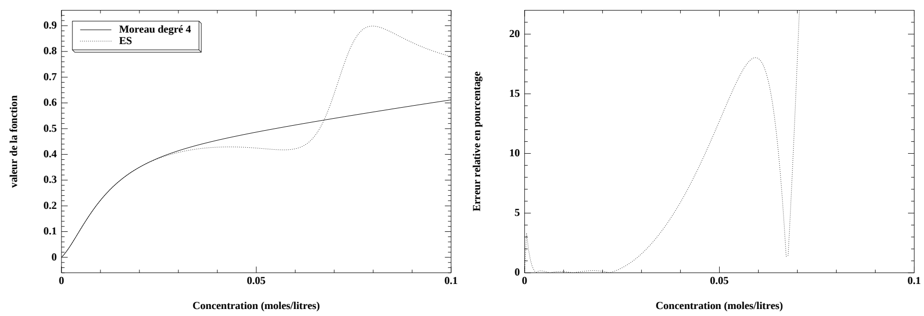



Figure 3.41: *Identification de l'isotherme de Moreau de degré 4 sur le domaine* $[0, 0.1]$*: valeurs des deux fonctions isothermes (à gauche) et différence relative entre les deux fonctions isothermes (à droite).*

Comme le montre la figure 3.41, l'isotherme est très mal identifié sur une grande partie du domaine d'identification. De plus, comme les autres bons résultats obtenus présentent les mêmes caractéristiques, on peut conclure que pour identifier l'isotherme sur le domaine [0,0.1] il faut choisir d'autres conditions expérimentales ou bien rajouter un troisième point test. Nous avons donc lancé 40 calculs en utilisant 3 points test comme au paragraphe 3.6.1.3. Cependant, aucun résultat suffisamment acceptable pour être présenté n'a été trouvé.

### 3.7.2 Chromatographie liquide-solide à deux corps

Les résultats des figures (3.42) et (3.43) représentent le meilleur des 40 calculs effectués. Le temps total de calcul est d'environ 100 heures CPU.

| Caractéristiques de la colonne | Conditions expérimentales | | Isotherme à identifier |
|---|---|---|---|
| | Conditions d'injection | Conditions initiales | |
| $L = 25$ cm<br>$T = 300$ K<br>$\varepsilon = 0.5$ | $t_{inj} = 10$ s<br>$u_0 = 50$ cm/s<br>$c^0_{inj} = (0.5,\ 0.0005)$ mol/l<br>$c^1_{inj} = (0.5,\ 0.5)$ mol/l | <br><br>$c^0_{init} = (0,\ 0.5)$ mol/l<br>$c^1_{init} = (0,\ 0)$ mol/l | Langmuir<br>$K_1 = 15$<br>$K_2 = 25$ |

Table 3.22: *Chromatographie liquide-solide : ensemble des données utilisées dans la simulation.*

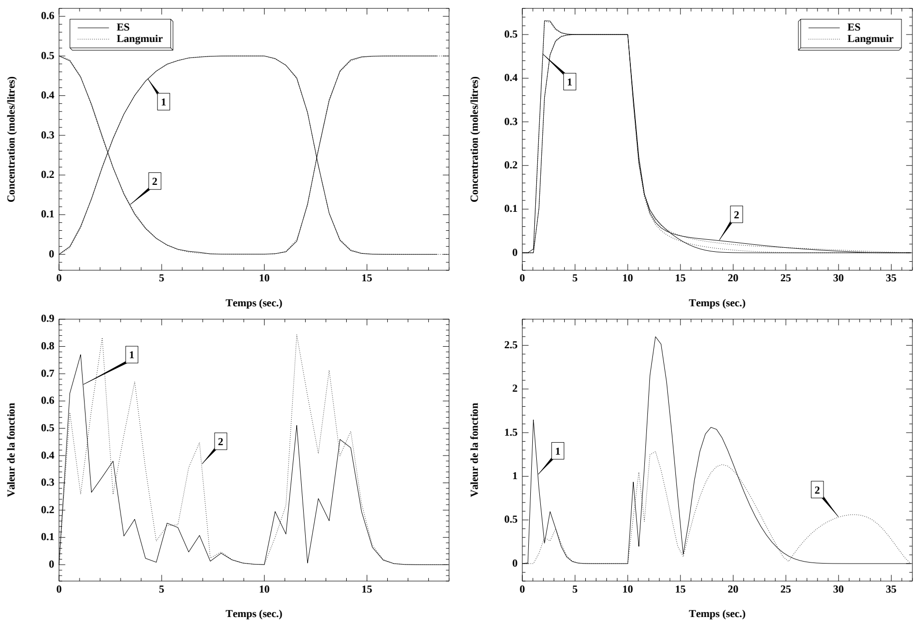



Figure 3.42: *Identification d'un isotherme de Langmuir ($K_1 = 15$ et $K_2 = 25$) avec 50 pas d'espace en utilisant 2 points test : chromatogramme expérimental et chromatogramme identifié pour chaque point test en haut et erreur relative en bas.*

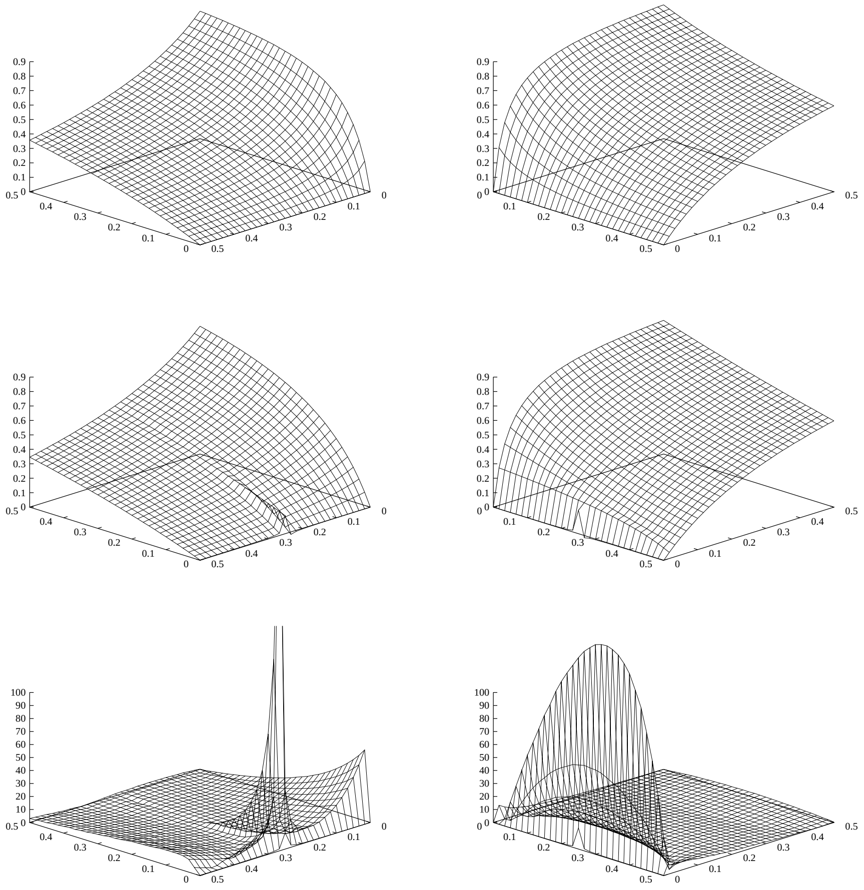


Figure 3.43: *Valeurs des deux fonctions isothermes expérimentales (en haut), valeurs des deux fonctions isothermes identifiées (au milieu) et différence relative (en bas).*

Les résultats sont ici trés bons malgré certaines valeurs erronées de la fonction isotherme. De plus, le fait que les autres 39 résultats sont trés mauvais nous indique que les limites de cette méthode telle qu'elle est actuellement sont atteintes.

### 3.7.3 Conclusion

Les résultats présentés pour un et deux corps nous montrent que cette méthode est moins efficace que celle qui utilise un réseau de neurones pour calculer les valeurs des points de contrôle. On peut penser que cela est dû au fait qu'il n'y a aucune corrélation entre ces différentes valeurs, ce qui favorise alors, lorsque le nombre de pas d'espace n'est pas élevé, l'apparition de fonctions chahutées voire discontinues mais ayant une trés bonne fitness pendant l'évolution.

## 3.8 Identification des coefficients des isothermes par stratégies d'évolution

Les modèles d'isothermes que nous utilisons dans ce paragraphe sont ceux de Moreau-Valentin. Le problème est donc d'identifier les coefficients de ces isothermes connaissant les chromatogrammes observés et les conditions initiales. Pour cela nous utilisons l'adaptation de notre algorithme à l'optimisation paramétrique présentée au chapitre 2.
Nous présentons dans les paragraphes suivants des résultats pour des problèmes d'identification à deux et trois corps d'isothermes de Moreau-Valentin de degré 2, ce qui revient à identifier respectivement cinq et neuf coefficients.

Nous faisons tout d'abord quelques remarques concernant l'algorithme ainsi que la simulation:

- Les simulations ont été effectuées avec 100 et 200 pas d'espace et non pas 50 comme pour les précédents résultats car les calculs sont beaucoup plus rapides.
- Les paramètres de l'algorithme (taille de la population, valeurs des probabilités de croisements et de mutations, ...) ont été fixés sans motivation particulière.
- Le nombre de générations de base est de 25.

Dans les deux paragraphes suivants, nous présentons les résultats obtenus pour des problèmes d'identification à deux et trois corps et lorsque le nombre de pas d'espace vaut 100 et 200. On terminera ensuite ce paragraphe par des commentaires concernant ces résultats ainsi que la conclusion sur cette approche.

### 3.8.1 Chromatographie liquide-solide à deux corps

#### 3.8.1.1 Un premier résultat

Ce résultat représente le meilleur des 5 calculs effectués, chaque calcul faisant évoluer 500 individus pendant 200 générations. Le nombre de pas d'espace est de 100. Le temps total de calcul est de 15 heures CPU.

| Caractéristiques de la colonne | Conditions expérimentales | |
|---|---|---|
| | Conditions d'injection | Conditions initiales |
| $L = 25$ cm | $t_{inj} = 35$ s | |
| $T = 300$ K | $u_0 = 12.3$ cm/s | |
| $\varepsilon = 0.5$ | $c^0_{inj} = (0.05,\ 0.05)$ mol/l | $c^0_{init} = (0,\ 0)$ mol/l |
| | $c^1_{inj} = (0.04,\ 0.04)$ mol/l | $c^1_{init} = (0,\ 0.04)$ mol/l |

Table 3.23: *Chromatographie liquide-solide à deux corps : ensemble des données utilisées dans la simulation.*

| Taille de la population: 500 | |
|---|---|
| Nombre maximal de générations: 200 | |
| Mutations | Croisements |
| $p\mathbf{W} =$ 0.35 | $p\mathbf{C}_+ =$ 0.11 |
| $p\beta =$ 0.15 | $p\mathbf{C}_- =$ 0.11 |

Table 3.24: *Ensemble des données utilisées par l'algorithme d'évolution dans la simulation.*

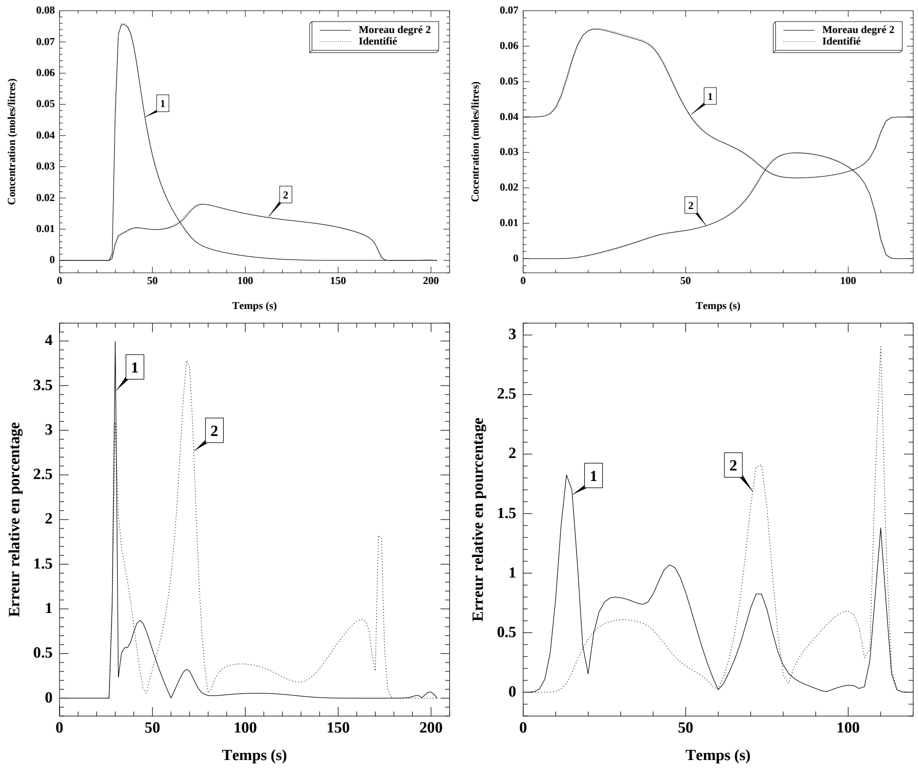



Figure 3.44: *Identification d'un isotherme de Moreau-Valentin de degré* 2 *avec* 100 *pas d'espace en utilisant* 2 *points test : chromatogramme expérimental et chromatogramme identifié pour chaque point test (en haut) et différence relative pour chaque chromatogramme (en bas).*

| Nom des coefficients | $K_1$ | $K_2$ | $E_{20}$ | $E_{11}$ | $E_{02}$ |
|---|---|---|---|---|---|
| Coefficients expérimentaux | 25 | 40 | -1000 | 1000 | 1000 |
| Coefficients identifiés | 25.55 | 39.23 | -967.12 | 959.49 | 959.85 |
| Erreur relative en % | 2.2 | 1.92 | 3.29 | 4.05 | 4.01 |

Table 3.25: *Résultats d'identification des coefficients énergétiques.*

#### 3.8.1.2 Un deuxième résultat

Ce résultat représente le meilleur des 7 calculs effectués, chaque calcul faisant évoluer 500 individus pendant 200 générations. Le nombre de pas d'espace est de 200. Le temps total de calcul est d'environ 80 heures CPU.

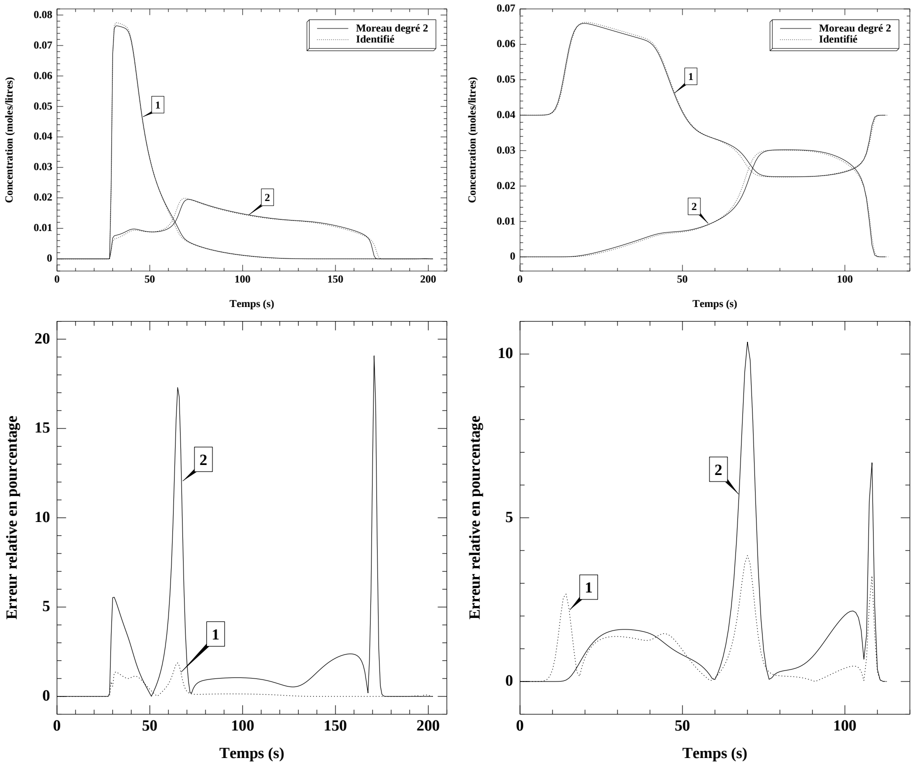



Figure 3.45: *Identification d'un isotherme de Moreau-Valentin de degré* 2 *avec* 200 *pas d'espace en utilisant* 2 *points test : chromatogramme expérimental et chromatogramme identifié pour chaque point test (en haut) et différence relative pour chaque chromatogramme (en bas).*

| Nom des coefficients | $K_1$ | $K_2$ | $E_{20}$ | $E_{11}$ | $E_{02}$ |
|---|---|---|---|---|---|
| Coefficients expérimentaux | 25 | 40 | -1000 | 1000 | 1000 |
| Coefficients identifiés | 27.08 | 39.06 | -892.23 | 967.78 | 967.85 |
| Erreur relative en % | 8.32 | 2.35 | 10.77 | 3.22 | 3.15 |

Table 3.26: *Résultats d'identification des coefficients énergétiques.*

### 3.8.2 Chromatographie liquide-solide à trois corps

#### 3.8.2.1 Un premier résultat

Ce résultat représente le meilleur des 6 calculs effectués, chaque calcul faisant évoluer 500 individus pendant 300 générations. Le nombre de pas d'espace est de 100. Le temps total de calcul est d'environ 30 heures CPU.

| Caractéristiques de la colonne | Conditions expérimentales | |
|---|---|---|
| | Conditions d'injection | Conditions initiales |
| $L = 25$ cm | $t_{inj} = 35$ s | |
| $T = 300$ K | $u_0 = 12.3$ cm/s | |
| $\varepsilon = 0.5$ | $c^0_{inj} = (0.05,\ 0.05,\ 0.05)$ mol/l | $c^0_{init} = (0,\ 0,\ 0)$ mol/l |
| | $c^1_{inj} = (0.04,\ 0.04,\ 0.015)$ mol/l | $c^1_{init} = (0,\ 0.04,\ 0.04)$ mol/l |

Table 3.27: *Chromatographie liquide-solide à trois corps : ensemble des données utilisées dans la simulation.*

| Taille de la population: 500 | |
|---|---|
| Nombre maximal de générations: 300 | |
| Mutations | Croisements |
| $pM_1 =$ 0.4 | $pCr_1=$ 0.1 |
| $pM_2 =$ 0.4 | $pCr_2=$ 0.1 |

Table 3.28: *Ensemble des données utilisées par l'algorithme d'évolution dans la simulation.*

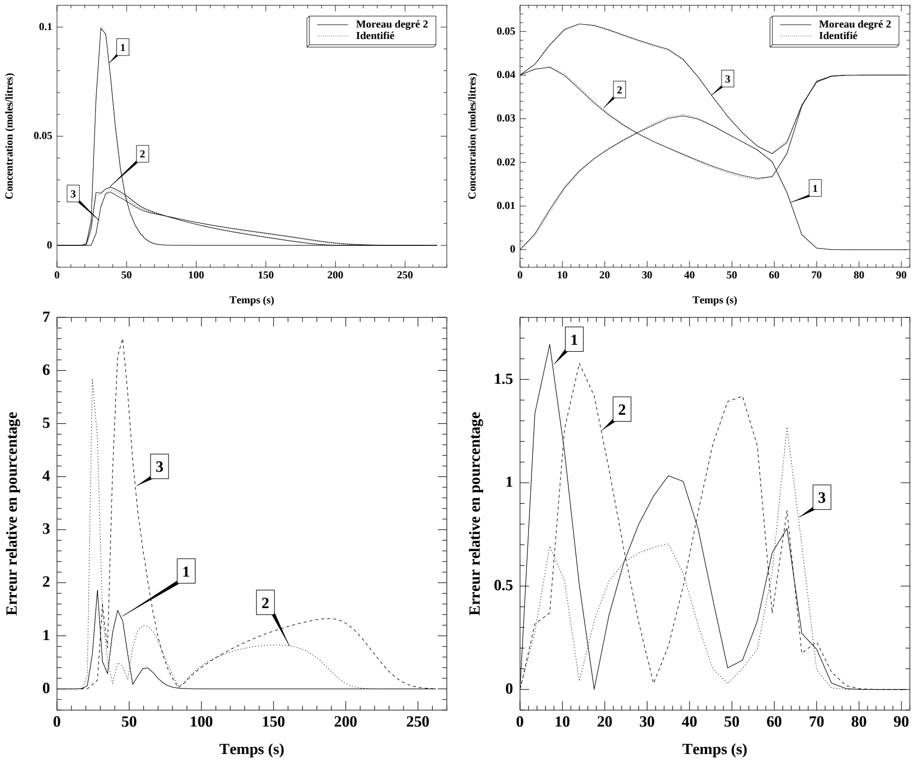



Figure 3.46: *Identification d'un isotherme de Moreau-Valentin de degré* 2 *avec* 100 *pas d'espace en utilisant* 2 *points test : chromatogramme expérimental et chromatogramme identifié pour chaque point test (en haut) et difference relative pour chaque chromatogramme (en bas).*

| Nom des coefficients | $K_1$ | $K_2$ | $K_3$ | $E_{200}$ | $E_{110}$ | $E_{020}$ | $E_{101}$ | $E_{011}$ | $E_{002}$ |
|---|---|---|---|---|---|---|---|---|---|
| Coefficients expérimentaux | 15 | 25 | 40 | -1000 | 1000 | 1000 | 1000 | -1000 | 1000 |
| Coefficients identifiés | 14.7 | 26.6 | 37.2 | -989.8 | 886.2 | 882.3 | 636.3 | -989.1 | 899.5 |
| Erreur relative en % | 2 | 6.4 | 7 | 1.02 | 11.38 | 11.77 | 36.37 | 1.09 | 10.05 |

Table 3.29: *Résultats d'identification des coefficients énergétiques.*

#### 3.8.2.2 Un deuxième résultat

Ce résultat représente le meilleur des 6 calculs effectués, chaque calcul faisant évoluer 500 individus pendant 400 générations. Le nombre de pas d'espace est de 200. Le temps total de calcul est d'environ 7 jours CPU.

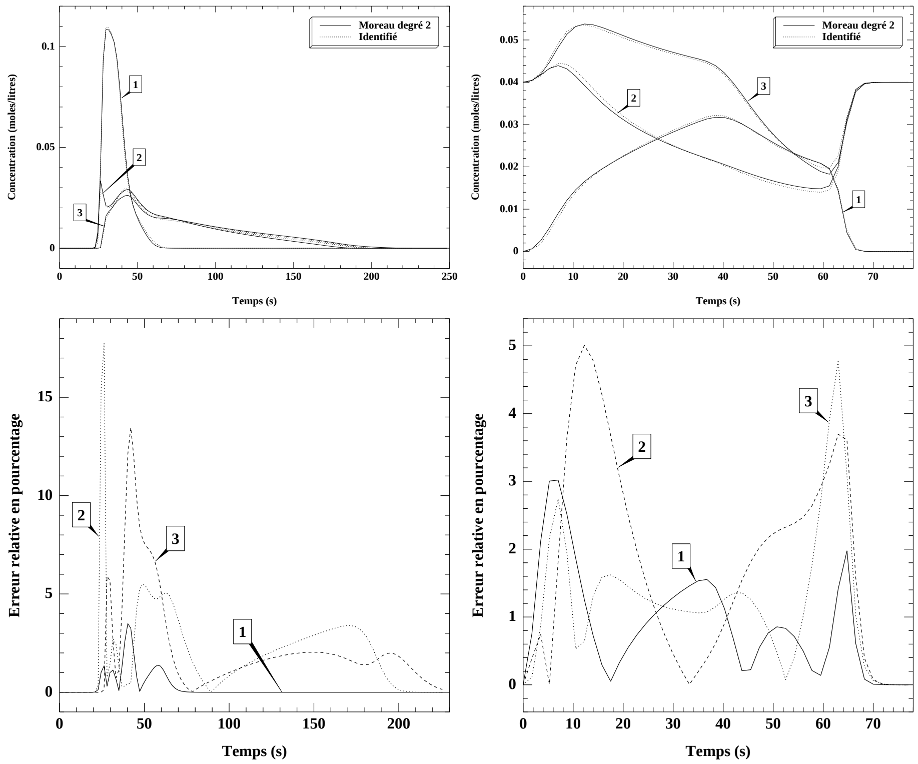



Figure 3.47: *Identification d'un isotherme de Moreau-Valentin de degré* 2 *avec* 200 *pas d'espace en utilisant* 2 *points test : chromatogramme expérimental et chromatogramme identifié pour chaque point test (en haut) et diffèrence relative pour chaque chromatogramme (en bas).*

| Nom des coefficients | $K_1$ | $K_2$ | $K_3$ | $E_{200}$ | $E_{110}$ | $E_{020}$ | $E_{101}$ | $E_{011}$ | $E_{002}$ |
|---|---|---|---|---|---|---|---|---|---|
| Coefficients expérimentaux | 15 | 25 | 40 | -1000 | 1000 | 1000 | 1000 | -1000 | 1000 |
| Coefficients identifiés | 15.2 | 31 | 33.9 | -973.7 | 973.5 | 887.7 | 403.1 | -980.9 | 568.6 |
| Erreur relative en % | 1.33 | 24 | 15 | 2.6 | 2.6 | 11.23 | 56.69 | 1.91 | 43.14 |

Table 3.30: *Résultats d'identification des coefficients énergétiques.*

### 3.8.3 Conclusion

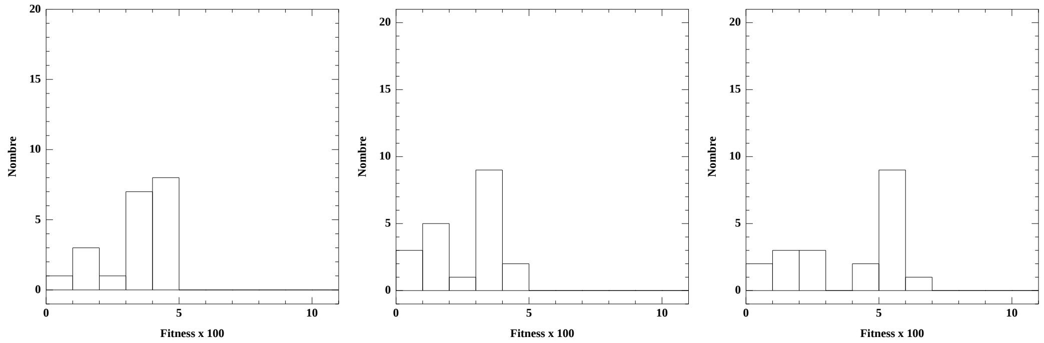
Les excellents résultats obtenus avec cette méthode nous montre que cette approche est fiable bien qu'elle suppose la connaissance d'un modèle d'isotherme. De plus, les temps de calcul deviennent rapidement élevés ce qui nous empêche d'effectuer des identifications avec un nombre de pas d'espace assez grand (le temps de calcul est multiplié par 4 environ lorsque l'on double le nombre de pas d'espace). Par ailleurs, on constate que l'erreur relative sur certains coefficients est trés importante. Cela est peut etre dû au fait que certain d'entre eux influent moins que d'autres sur le critère de performance (la fonction de fitness), ce qui pourrait nous amener à considérer d'autres critères comme les moments d'ordre $p$ (voir [S94]).

## 3.9 Comparaison entre toutes les méthodes

La première remarque que nous pouvons faire est que les résultats obtenus en utilisant les méthodes évolutionnaires sont moins bons que les résultats obtenus grâce à la méthode du gradient qui demande cependant la connaissance a priori d'un modèle analytique, ce qui n'est pas le cas pour la plupart des méthodes évolutionnaires que nous avons présentées. Les derniers résultats sur la méthode du gradient concernent à notre connaissance un problème à deux corps basé sur des données réelles et simulé avec un nombre de pas d'espace d'environ 2000 alors que les meilleurs résultats que nous avons obtenus concernent des problèmes à un deux ou trois corps avec au mieux 200 pas d'espace ou encore un problème à un corps avec 1000 pas d'espace.
La principale limite des méthodes évolutionnaires réside en l'état actuel principalement dans le temps de calcul. Ainsi, si l'on voulait identifier un isotherme de Moreau-Valentin de degré 2 avec une simulation à 2000 pas d'espace, il aurait fallu multiplier les temps de calcul de la simulation à 200 pas d'espace par un ordre de grandeur de 100, ce qui n'est pas réalisable actuellement.
Parmi les méthodes évolutionnaires, la méthode d'identification des coefficients des isothermes donne les meilleurs résultats (mais demande comme pour la méthode du gradient la connaissance du modèle de l'isotherme) et ne peut être comparée avec les autres. Ainsi, dans ce paragraphe, nous comparerons les trois autres méthodes évolutionnaires : *Réseau*, *Réseau+Interpolation* et *ES+Interpolation*.

### 3.9.1 Comparaison entre les trois méthodes évolutionnaires

Nous comparons tout d'abord les résultats obtenus pour le problème à un corps (présenté aux paragraphes 3.6.1.2, 3.5.1.4 et 3.7.1.1) d'identification de l'isotherme de Moreau-Valentin de degré 4 en utilisant deux points test.
Les figures 3.48 représentent la répartition en fonction de leur fitness des meilleurs individus trouvés sur un ensemble de calculs et pour chaque méthode.
Les figures 3.49 donnent pour les meilleurs individus trouvés (sur 20 calculs) et pour chaque méthode l'erreur relative par rapport à l'isotherme à identifier.




*Réseau* *Réseau+Interpolation* *ES+Interpolation*

Figure 3.48: *Répartition pour les trois méthodes sur un problème à un corps d'identification d'un isotherme de Moreau-Valentin de degré 4 en utilisant deux points test. Il y a 20 calculs pour chaque méthode.*

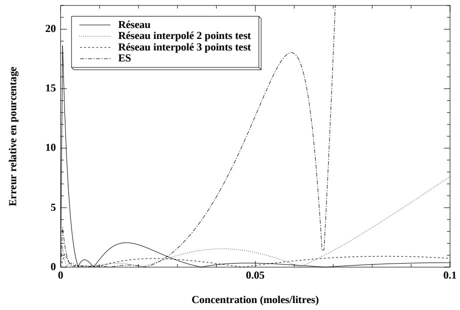
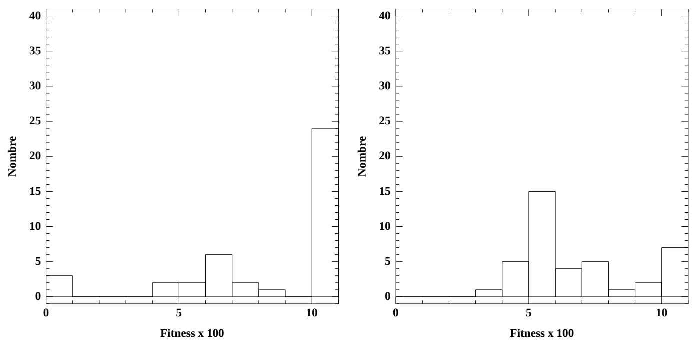


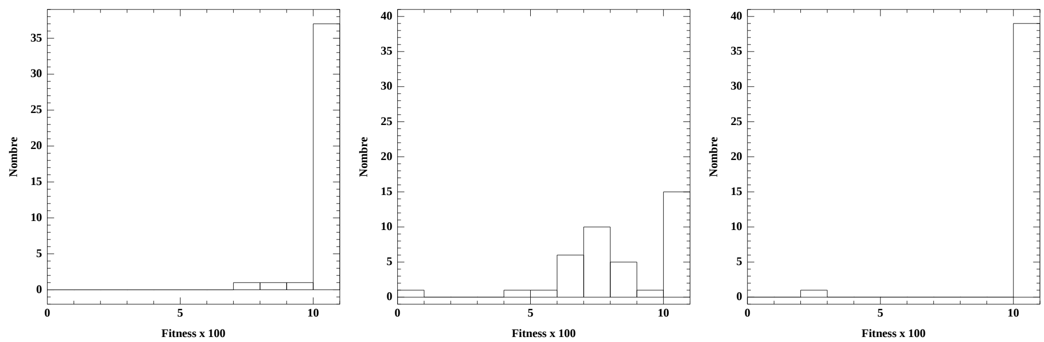

Figure 3.49: *Meilleures erreurs relatives (sur 20 calculs) par rapport à l'isotherme de Moreau-Valentin de degré 4 pour les trois méthodes.*

Bien que la fitness pour la méthode *Réseau* soit plus élevée que pour les deux autres méthodes (0.0137 contre 0.0051 pour la méthode *ES+Interpolation* avec deux points test et 0.0057 pour la méthode *Réseau+Interpolation* avec 2 points test et 0.0036 avec 3 points test), on constate que l'isotherme est mieux identifié sauf au voisinage de zéro. De plus, pour la méthode *ES+Interpolation*, on obtient la meilleur fitness pour deux points test alors que l'isotherme est très mal identifié sur une grande partie de l'intervalle [0,0.1]. On peut expliquer ces résultats d'une part par le fait qu'une mauvaise identification au voisinage de zéro est pénalisante. D'autre part, le fait qu'à fitness pratiquement égale (ES et réseau interpolé), on obtienne des identifications très différentes, est dû au nombre de pas d'espace trop petit qui "permet" à de telles solutions d'exister. Ainsi, si l'on teste le même individu avec 100 pas d'espace, on obtient une mauvaise identification d'un des deux chromatogrammes.
Les figures 3.50 donnent les répartitions pour les deux méthodes d'interpolation pour trois points test, ce qui nous montre la supériorité de la méthode *Réseau+Interpolation* sur la méthode *ES+interpolation* pour laquelle on n'obtient aucun résultat acceptable.




Réseau interpolé ES

Figure 3.50: *Répartition pour les deux méthodes par interpolation sur un problème à un corps d'identification d'un isotherme de Moreau-Valentin en utilisant trois points test. Il y a 40 calculs pour chaque méthode.*




*Réseau* *Réseau+Interpolation* *ES+Interpolation*

Figure 3.51: *Répartition pour les trois méthodes sur un problème à deux corps d'identification d'un isotherme de Langmuir en utilisant deux points test. Il y a 40 calculs pour chaque méthode.*

Pour un problème à deux corps, on obtient clairement un meilleur résultat avec la méthode *Réseau+Interpolation* (voir les figures 3.51 et 3.52). Ce résultat qui est par ailleurs très bon est le seul réellement acceptable sur les 40 calculs lancés. On peut faire la même remarque pour la méthode *ES+Interpolation* avec cependant un moins bon résultat comme le montre la figure 3.52. On note de plus que la méthode *Réseau* ne donne aucun résultat acceptable.

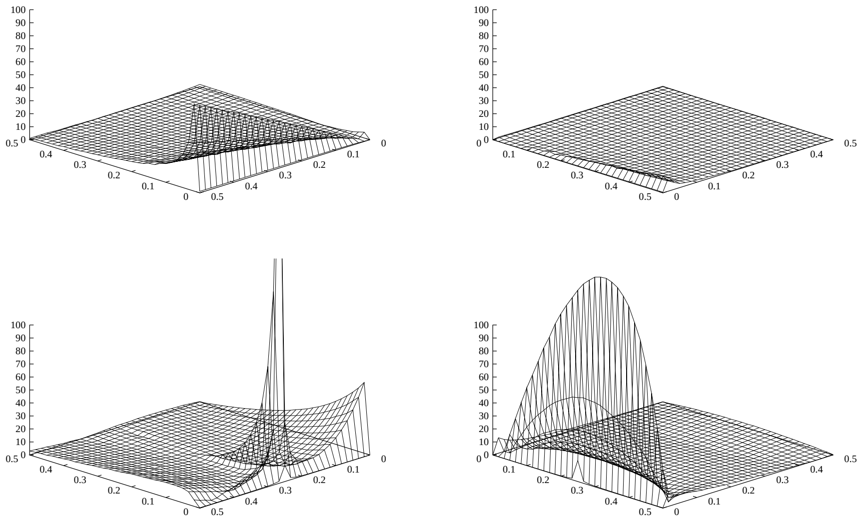


Figure 3.52: *Erreur relative en pourcentage entre l'isotherme de Langmuir et l'isotherme identifié pour la méthode* Réseau *(en haut) et la méthode* ES+Interpolation *(en bas).*

Dans ce qui suit, nous utilisons le meilleur réseau trouvé avec la méthode *Réseau* en lui faisant cependant calculer uniquement les valeurs des points de contrôle. On obtient ensuite la fonction isotherme en effectuant une interpolation. Le réseau que nous utilisons a été obtenu sur le problème à un corps d'identification d'un isotherme de Moreau-Valentin de degré 4.

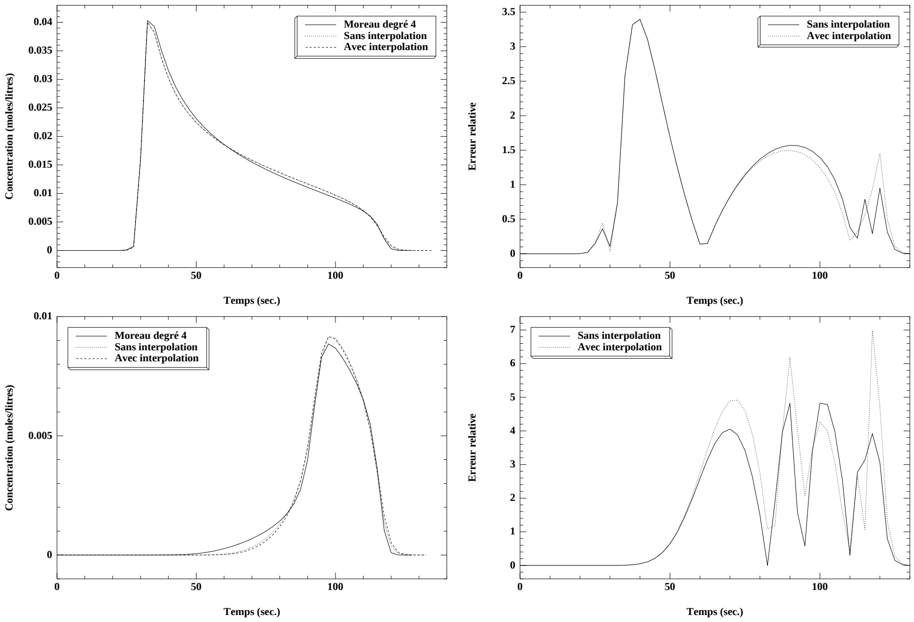



Figure 3.53: *Identification de l'isotherme de Moreau de degré 4 avec 50 pas d'espace : chromatogramme expérimental et chromatogrammes identifiés pour chaque point test et pour les deux méthodes à gauche et erreurs relatives pour les deux méthodes à droite.*

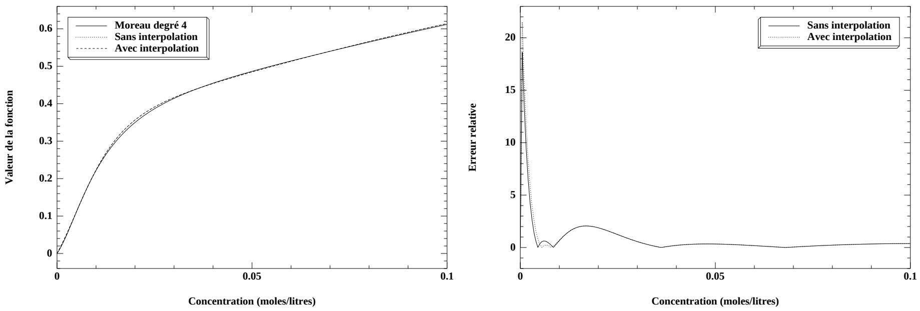



Figure 3.54: *Identification de l'isotherme de Moreau de degré 4 sur le domaine* $[0, 0.1]$*: valeurs des deux fonctions isothermes (à gauche) et différence relative entre les deux fonctions isothermes (à droite) pour les deux méthodes.*

Réciproquement, les résultats suivants montrent comment se comporte un réseau obtenu par la méthode *Réseau+Interpolation* lorsqu'il doit calculer tous les points du domaine d'identification et non plus uniquement les valeurs des points de contrôle. Pour cela, nous avons utilisé le meilleur réseau trouvé (pour le même problème) par la méthode *Réseau+Interpolation*.

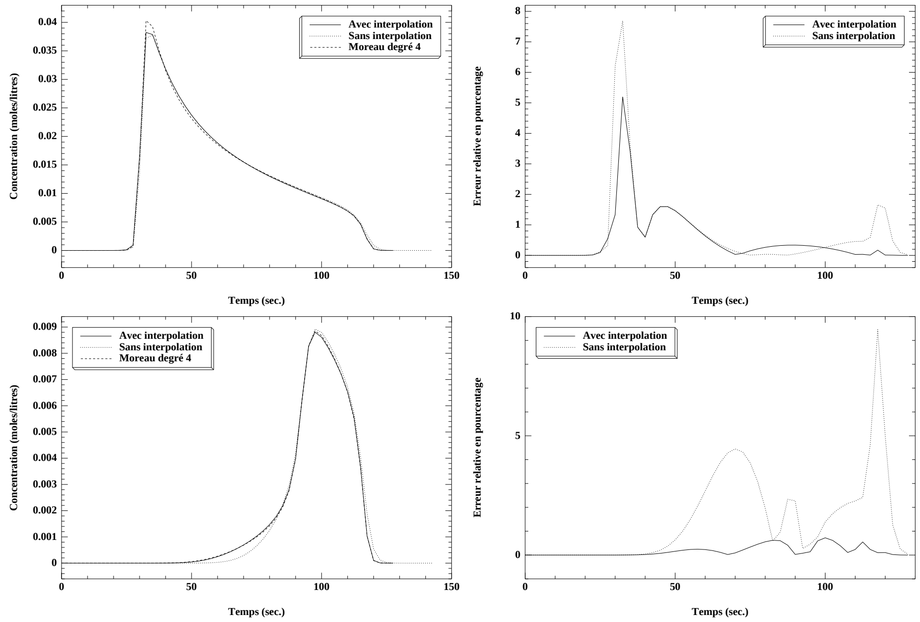



Figure 3.55: *Identification de l'isotherme de Moreau de degré 4 avec 50 pas d'espace : chromatogramme expérimental et chromatogramme identifié pour chaque point test en haut et erreur relative en bas.*

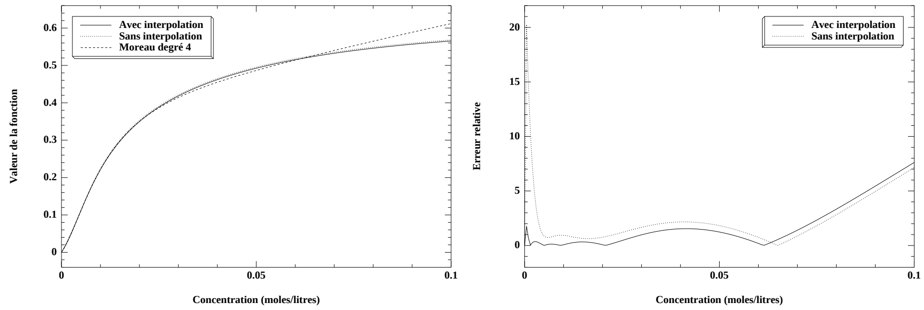



Figure 3.56: *Identification de l'isotherme de Moreau de degré 4 sur le domaine* $[0, 0.1]$*: valeurs des deux fonctions isothermes (à gauche) et différence relative entre les deux fonctions isothermes (à droite).*

Comme on pouvait logiquement le penser, le réseau qui identifie corectement l'ensemble des points du domaine d'identification identifie donc correctement les valeurs des points de contrôle alors qu'un réseau entraîné à identifier les valeurs des points de contrôle n'identifie pas aussi bien les points du domaine d'identification.
Pour deux corps, nous ne présentons pas de figure mais on arrive également à la même conclusion.

## 3.10 Conclusion

Parmi les méthodes évolutionnaires, c' est la méthode de calcul des coefficients des isothermes par stratégie d'évolution qui donne de meilleurs résultats. Cependant, comme la méthode de gradient, elle demande la connaissance a priori d'un modèle analytique, ce qui n'est pas le cas pour les autres méthodes évolutionnaires que nous avons présentées. En général, ces méthodes sont trés coûteuses en temps de calcul et ne permettent donc pas de traiter pour l'instant des problèmes avec des données réelles, qui nécessitent de travailler avec un nombre de pas d'espace élevé.

Pour les trois méthodes qui utilisent les fraction rationnelles et les réseaux de neurones comme modèle d'isotherme, on a vu que le choix des conditions expérimentales (vitesse d'injection, durée d'injection, conditions initiales, conditions d'injection) dont dépend le domaine d'identification est très important si l'on veut identifier correctement un isotherme sur un certain domaine. La méthode *Réseau+Interpolation* donne des résultats encourageants comme le montrent les résulats pour un et deux corps mais semble cependant avoir atteint en l'état actuel sa limite comme le montre le (seul) très bon résultat obtenu sur les 40 calculs lancés pour un problème à deux corps. Cette méthode est néanmoins sans aucun doute meilleure que celle qui utilise uniquement les réseaux de neurones (méthode *Réseau*). En effet, elle est d'une part beaucoup plus rapide et d'autre part on a vu qu'il était plus facile pour un réseau d'identifier correctement plusieurs valeurs de points de contrôle plutôt que d'identifier toutes les valeurs du domaine d'identification. Par ailleurs, la méthode *ES+Interpolation* présente à notre avis le défaut de ne pas imposer une assez grande cohérence entre ces différentes valeurs, ce qui expliquerait que la méthode qui calcule ces valeurs par réseaux soit meilleure.

# Chapitre 4

# Un problème de contrôle non-linéaire

## 4.1 Introduction

Dans ce chapitre, nous étudions un problème test de contrôle non-linéaire par des réseaux de neurones : à partir d'une position quelconque et en contrôlant uniquement le braquage des roues, un réseau de neurones doit être capable de garer la voiture de manière précise. Le réseau connaîtra uniquement la position $(x, y)$ du milieu de l'essieu avant, ainsi que l'angle $\theta$ formé par l'axe de la voiture et l'axe des abscisses et devra alors être capable de donner l'angle $\phi$ qui est l'angle formé par les roues avant et l'axe du véhicule. Pour trouver un réseau qui effectue cette tâche, nous utiliserons l'algorithme d'évolution présenté au chapitre 2.
Le cadre de ces travaux est le même que dans [NW90] qui traite cependant un problème différent avec une méthode différente : garer un véhicule et sa remorque à partir d'une position quelconque avec un réseau optimisé par une méthode de rétro–propagation en temps. Cette méthode est dérivée de la méthode de rétro–propagation qui ne peut être utilisée ici, car il faudrait des données sur les trajectoires qui ne sont pas connues a priori. Par ailleurs, ce problème est différent du problème de contrôle tel qu'il est concu dans [RFLM93] qui, à partir d'une trajectoire connue, donne la série de commandes nécessaires à un véhicule et $n$ remorques pour suivre cette trajectoire.
Apres avoir présenté le problème ainsi que toutes les données nécessaires à l'utilisation de notre algorithme, nous présenterons plusieurs réseaux de neurones feed-forward capables de garer un véhicule à partir de positions de départ quelconques avec des taux de réussite d'environ 80-90% et en suivant des trajectoires réalistes. Nous étudierons par ailleurs le comportement de ces réseaux pour des points de départ très éloignés du but fixé, l'influence de l'injection de bruit sur leur comportement ainsi que le couplage de plusieurs réseaux.

## 4.2 Les équations du mouvement

Dans notre simulation, nous faisons l'hypothèse que la voiture avance à une vitesse uniforme $u$. Après chaque pas, nous considérons connus la position $(x, y)$ du milieu de l'essieu avant, ainsi que l'angle $\theta$ formé par l'axe de la voiture et l'axe des abscisses.
Avant chaque pas, il faut connaître $\phi$, qui est l'angle formé par les roues avant et l'axe des abscisses.
C'est le réseau de neurones qui va assurer le rôle du conducteur : il recevra comme entrées les

variables $x, y$ et l'angle $\theta$ et devra donner comme réponse l'angle $\phi$. Ainsi, partant d'une position initiale $(X_0, Y_0, \theta_0)$ le réseau va pouvoir donner l'angle $\phi_0$ avec lequel nous pourrons calculer la nouvelle position $(X_1, Y_1, \theta_1)$ (voir la figure 4.1). La voiture décrira alors une trajectoire, son but étant d'arriver en $(0, 0, 0)$.

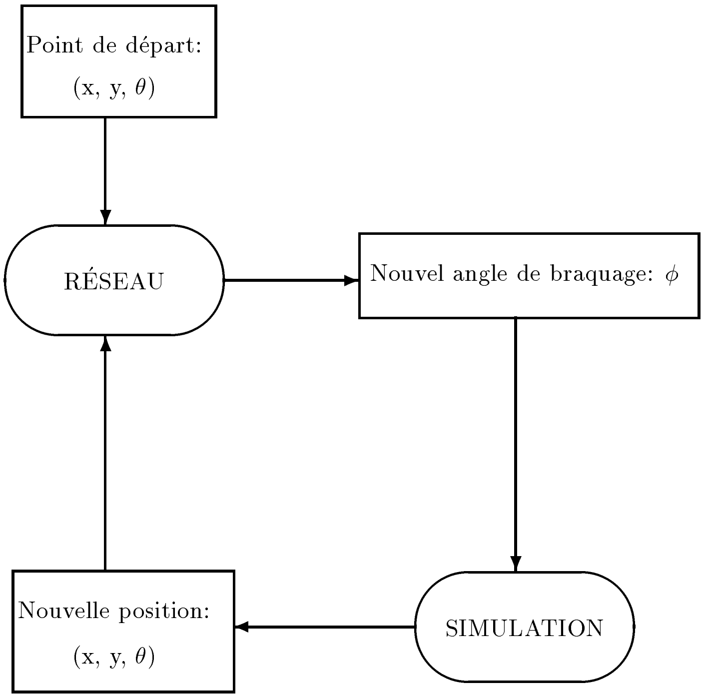



Figure 4.1: *Simulation du problème.*

Résoudre ce problème implique résoudre le problème qui consiste à garer la voiture en un point quelconque. En effet, garer la voiture en $(x_f, y_f, \theta_f)$ revient à garer la voiture en $(0, 0, 0)$ sur le repère orthonormé d'origine $(x_f, y_f)$ et dont l'axe des ordonnées forme un angle de $\theta_f$ avec l'axe des ordonnées du repère initial. Les vrais trajectoires sont ensuite calculées en utilisant les matrices de passage d'un repère à l'autre.

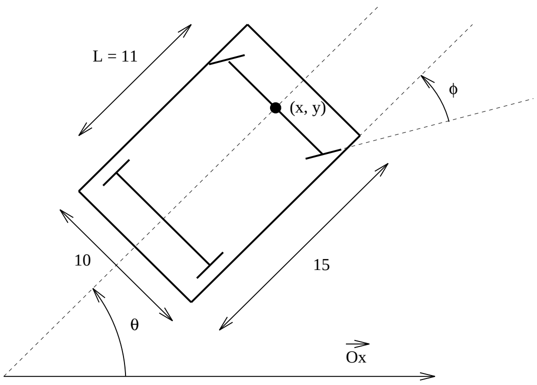



Figure 4.2: Les dimensions de la voiture

La figure 4.2 donne les dimensions de la voiture. C'est la distance $L = 11$ séparant l'essieu avant de l'essieu arrière qui sera employé dans les équations du mouvement.
Dans notre simulation, nous faisons l'hypothèse que la voiture avance avec un pas uniforme $u$ dont la valeur numérique est 3 unités. Par ailleurs, l'angle de braquage maximum $\phi$ est de 0.7 radians (environ 45 degrés).

La figure 4.2 ci-dessous sert à illustrer comment les équations du mouvement ont été déduites. Considérons la position de départ suivante avec A le milieu de l'essieu avant, B le milieu de l'essieu arrière et $\phi$ l'angle de braquage des roues. Si les roues avant avancent d'une petite distance $u$, alors le point $A$ se déplace en $A'$ et le point $B$ se déplace en $B'$. Nous faisons alors les approximations suivantes : $distance(AA') = u$ et $B'$ appartient à la droite AB.

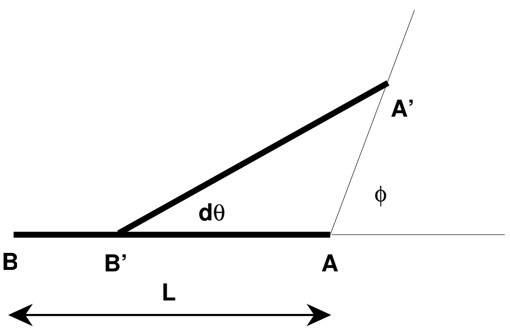



Figure 4.3: Mouvement de la voiture

Ce qui conduit à

$$L \sin(d\theta) = u \sin(\phi)$$

et donc

$$d\theta = \arcsin\left(\frac{u \sin(\phi)}{L}\right)$$

En résolvant le reste du système, on obtient finalement

$$\begin{cases} x' = x + cos(\phi + \theta) \\ y' = x + sin(\phi + \theta) \\ \theta' = \theta + d\theta \end{cases} \tag{4.1}$$

## 4.3 Les données de l'évolution

Dans ce paragraphe, nous présentons toutes les données et tous les paramètres nécessaires à l'utilisation de notre algorithme.

### 4.3.1 La fonction de performance

On considère la trajectoire d'une voiture $x_i = (X_i, Y_i, \theta_i)$ ($\theta_i$ exprimé en radians) , $0 < i \leq N$ (N est le nombre maximal d'itération) $x_0 = (X_0, Y_0, \theta_0)$ donné. La fitness $F$ du réseau qui décrit cette trajectoire est la distance minimum par rapport au point $(0, 0, 0)$ :

$$\begin{aligned} F &= \max(\min_i d(x_i, (0,0,0)) - \tfrac{u}{2}, \quad 0) \\ \text{avec} \quad d(x_i,(0,0,0)) &= \sqrt{X_i^2 + Y_i^2 + M^2 min(\theta_i^2, (\theta_i - 2\pi)^2), (\theta_i + 2\pi)^2)}, \quad M = 50 \end{aligned} \tag{4.2}$$

Lorsqu'il y aura plusieurs trajectoires, la fitness effective F sera la moyenne des fitness $F$ correspondant à chaque trajectoire.
Nous utilisons l'expression $min(\theta_i^2, (\theta_i - 2\pi)^2), (\theta_i + 2\pi)^2)$ afin que la voiture puisse faire au maximum un tour sur elle même (et pas plus) dans les deux sens. De plus on multiplie cette quantité par $M = 50$ pour qu'elle ne soit pas négligeable devant $X_i^2 + Y_i^2$. Par ailleurs on soustrait la quantité $\frac{u}{2}$ de la fitness car, avec un pas de $u$, une précision supérieure à $\frac{u}{2}$ ne signifie rien (voir la figure 4.4 pour des exemples de trajectoires ainsi que les distances minimales obtenus). On remarquera que, comme pour la chromatographie, l'algorithme doit minimiser la fonction de performance.

### 4.3.2 Les paramètres de l'algorithme

Nous faisons dans ce paragraphe quelques remarques et précisions sur l'évolution et l'initialisation de la population.

- Au cours de l'évolution, nous adoptons le raisonnement suivant : si la performance n'est pas assez petite au bout de 500 générations alors on recommence un autre calcul. Le seuil a été fixé après observation de plusieurs calculs.
- Nous ne fixons pas un nombre maximal de générations mais si la fitness n'augmente pas pendant 500 générations alors on arrête le calcul.
- La taille de la population est de 30 réseaux de neurones feed-forward (formés d'une seule sous-structure) comportant 3 entrées $(X, Y, \theta)$ et 1 sortie $(\phi)$.
- Les différentes probabilités de mutations utilisées par l'algorithme d'évolution sont :

| Mutations paramétriques | Mutations architecturales |
|---|---|
| $p\mathbf{W} = \;\; 0.4$ | $p\mathbf{C}_+ = \;\; 0.15$ |
| $p\beta = \;\; 0.1$ | $p\mathbf{C}_- = \;\; 0.15$ |
| $p\mathbf{W_{in}} = \;\; 0$ | $p\mathbf{U}_+ = \;\; 0.1$ |
| $p\mathbf{W_{out}} = \;\; 0$ | $p\mathbf{U}_- = \;\; 0.1$ |

Table 4.1: *Ensemble des probabilités de mutations utilisées par l'algorithme d'évolution.*

- Le nombre de générations de base est fixé à 25.
- Le nombre de neurones cachés ou intermédiaire (noté **h**) est tiré aléatoirement et uniformément pour chaque réseau sur l'intervalle $[1, 3]$.
- La fonction de transfert qu'on utilise est $\sigma_\beta(x) = \frac{1}{1+e^{-\beta x}}$. On définit alors un intervalle de départ pour $\beta$ qui est $[-10, 10]$.
- La densité de connexions indique indépendamment du nombre de neurones le nombre total de connexions d'un réseau. Cette échelle varie de 1 à 10, 1 signifiant que chaque neurone est connecté à au moins un autre neurone et 10 signifiant que chaque neurone est connecté à (**h** + le nombre de neurones de sortie) autres neurones (certains pouvant être les mêmes). La densité pour chaque réseau est tirée aléatoirement et uniformément sur l'intervalle $[1, 10]$.
- Nous ne mettons pas de facteur de pénalisation pour les trajectoires trop longues mais nous limitons le nombre maximal d'itération $N$. Pour cela, on estime la plus longue trajectoire optimale parmi les points test (en unités de distance). On considère ensuite qu'une trajectoire reste acceptable si elle ne dépasse pas deux fois cette longueur (on pourrait mettre un critère plus sevère). On divise alors cette estimation par la vitesse $u$ pour obtenir un nombre d'itérations.
  Nous avons choisi de ne pas rajouter une pénalité pour les trajectoires trop longues, car il faudrait connaître la longueur de la trajectoire optimale ou bien pénaliser les réseaux indépendamment de la position de départ (on pénalise les réseaux qui dépassent 100 itérations par exemple) ce qui peut soit induire l'agorithme en erreur, soit être inutile lorsque le réseau est entraîné avec plusieurs points test, ce qui sera toujours notre cas.

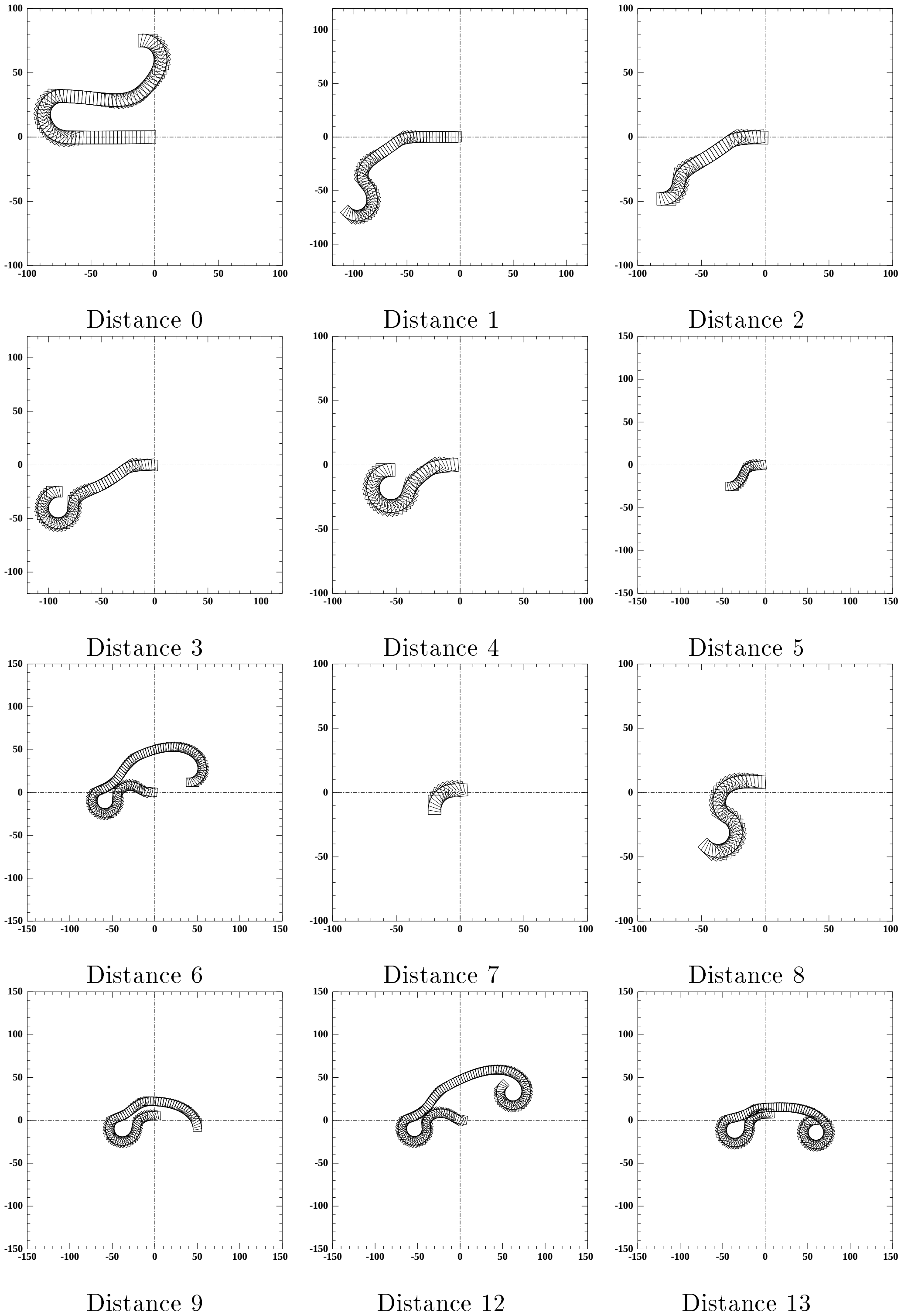



Figure 4.4: *Quelques trajectoires. Distance x signifie que la distance minimum par rapport au point* $(0,0,0)$ *pour cette trajectoire est* $x$.

## 4.4 Les résultats

### 4.4.1 Un premier résultat

Le réseau de neurones pour lequel nous allons donner un premier résultat est un réseau de neurones feed-forward qui a été entraîné avec 1 point test : (50,50,0).
Ce réseau (de fitness nulle) a été obtenu en 129 générations (environ 40 secondes CPU sur un Pentium 90) et a 2 neurones intermédiaires et 3 connexions.

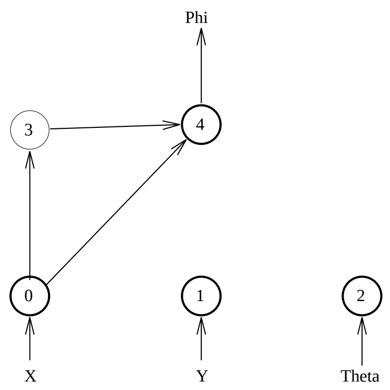



Figure 4.5: Diagramme du réseau

On considère que le calcul des trajectoires pour les points de départ $(X, Y, \theta)$ avec $Y < 0$ est identique au calcul de la trajectoire pour les points de départ $(X, -Y, -\theta)$ et en considérant que la réponse du réseau est $-\phi$ au lieu de $\phi$. Cette méthode n'entraîne pas une symétrie parfaite pour toutes les trajectoires car le réseau n'est pas une fonction périodique.

Nous allons présenter plusieurs types de graphiques relatifs au domaine [-100, +100] × [-100, +100] × $[-\pi, +\pi]$.

- Les graphiques que nous allons présenter dans la figure 4.7 doivent être interprétés de la manière suivante :
  Chaque petit graphique décrit l'espace $[-100, +100] \times [-100, +100]$ des coordonnées initiales (en $X$ et $Y$) et correspond à un certain $\theta$ qui est l'orientation initiale de la voiture (il y a 8 orientations initiales allant de $-\pi$ à $\frac{3\pi}{4}$).
  A chaque coordonnée initiale est associée une couleur allant du blanc au noir en passant par des niveaux de gris. Plus la couleur est claire, plus la voiture se gare bien à partir de cette position. Le blanc signifiant que la voiture est parfaitement garée (distance 0) tandis que le noir signifie que la voiture est soit loin du but soit assez proche (quelques unités) mais mal orientée (distance 10 et +).

- Les graphiques de la figure 4.8 sont du même type que ci-dessus mais traitent de la longueur des trajectoires.

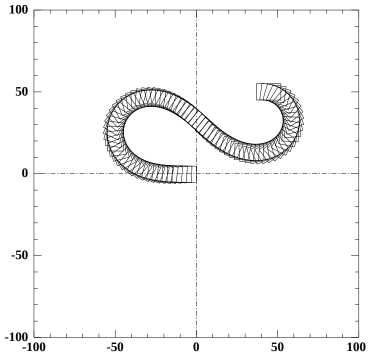



Figure 4.6: *Trajectoire obtenue pour le point test (50,50,0).*

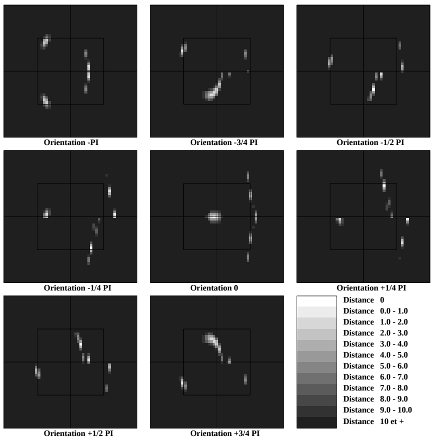



Figure 4.7: *Distance en fonction de la position initiale de la voiture. Orientation i PI correspond à une orientation initiale de* $i\pi$*. Chaque figure représente l'espace* $[-100, +100] \times [-100, +100]$*. Les coordonnées d'un point sont ses coordonnées de départ et sa couleur indique la distance minimum obtenue par rapport au point* $(0, 0, 0)$*.*

- La figure 4.6 montre la trajectoire obtenue pour le point test (50,50,0).
- La figure 4.9 donne la répartition en fonction de la distance minimum de 20808 trajectoires issues de 20808 points de départ (51 pour X et Y et 8 pour $\theta$) repartis sur le domaine [-100, +100] $\times$ [-100, +100] $\times$ $[-\pi, +\pi]$.

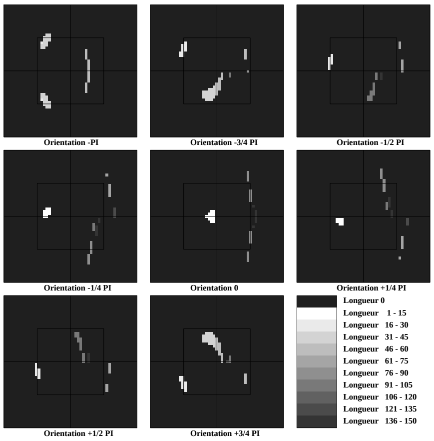
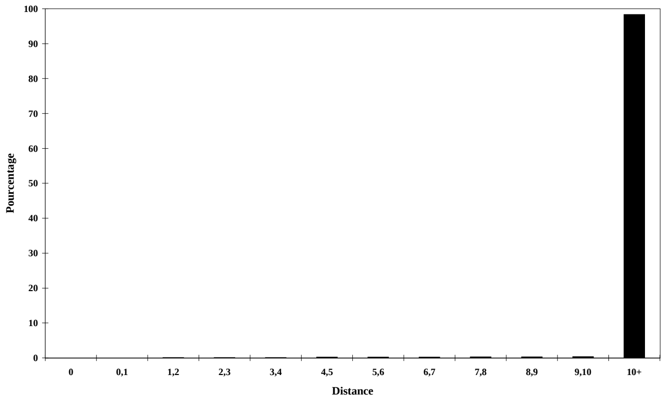



Figure 4.8: *Nombre d'itérations pour arriver au but avec un maximum de 160 itérations. Orientation i PI correspond à une orientation initiale de $i\pi$. Chaque figure représente l'espace $[-100, 100] \times [-100, 100]$. Les coordonnées d'un point sont ses coordonnées de départ et sa couleur indique le nombre d'itérations nécessaires pour arriver au point $(0, 0, 0)$. Les zones noires correspondent aux points d'où la voiture n'arrive pas à se garer.*




| 0 | 0-1 | 1-2 | 2-3 | 3-4 | 4-5 | 5-6 | 6-7 | 7-8 | 8-9 | 9-10 | 10+ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.024 | 0.02 | 0.077 | 0.067 | 0.096 | 0.19 | 0.18 | 0.18 | 0.26 | 0.27 | 0.32 | 98.32 |

Figure 4.9: *Répartition (en pourcentage) des distances minimales par rapport à (0,0,0) sur un échantillon de 20808 trajectoires issues de 20808 points de départ ($51 \times 51 \times 8$) répartis uniformément sur le domaine $[-100, +100] \times [-100, +100] \times [-\pi, \pi]$. Il y a 12 catégories de répartition : 0, 0-1, 1-2, 2-3, 3-4, 4-5, 5-6, 6-7, 7-8, 8-9, 9-10 et 10+.*

Comme le montre les figures 4.7 et 4.9, l'entraînement avec un seul point test n'est pas suffisant si l'on veut résoudre le problème sur un domaine assez grand.
Dans le paragraphe suivant nous avons effectué les calculs en partant avec cinq point test et en ajoutant un point test lorsque la fonction de performance devient nulle.

### 4.4.2 Ajout de points test pendant l'évolution

Notre but étant d'obtenir un réseau qui soit capable de se garer à partir de toutes les positions, nous rajoutons un nouveau point test lorsque la fitness est nulle. On espère ainsi réduire les zones d'où la voiture ne se gare pas. Pour cela, on calcule la performance de tous les points appartenant à un espace défini à l'avance (espace qui englobe l'ensemble des points test de départ). Par exemple si on travaille sur l'espace des points de départ $[-100, +100] \times [-100, +100] \times [-\pi, +\pi]$ alors le programme parcourra $51 \times 51 \times 8 = 20808$ points et le point test que nous ajouterons sera celui qui aura la plus mauvaise fitness. On recommence ensuite ce processus jusqu'à ce que la fitness n'augmente plus. Dans le paragraphe qui suit, nous présentons deux séries de résultats pour lesquels on a rajouté quatre et trois points test alors qu'au départ il y en avait cinq et six respectivement.

### 4.4.3 Un deuxième résultat

Le réseau de neurones pour lequel nous allons donner un deuxième résultat est un réseau de neurones feed-forward qui a été entraîné avec 5 points test de base ((-100,0,0), (-100,100,0), (0,100,0), (100,100,0), (100,0,0)) auxquels sont venus ensuite s'ajouter 4 points test : (80,0,$\frac{\pi}{4}$), (60,0,$\frac{\pi}{4}$), (40,0,$\frac{\pi}{4}$), (20,0,$\frac{\pi}{4}$). Le nombre maximal d'itérations a été fixé à 125.
Avant d'obtenir ce résultat, l'algorithme a échoué cinq fois à 500 générations. Le seuil de performance pour continuer l'évolution après ces 500 générations a été fixé à 15 (c'est-à-dire que en moyenne chaque trajectoire devaient s'approcher à une distance d'au moins 15). Le temps de calcul pour la sixième tentative est de 2 heures CPU sur un Pentium 90 pour atteindre une fitness nulle pour 9 points test (voir table 4.2). Le temps total de calcul est de 158 minutes (38 minutes pour les cinq premières tentatives).
Les cinq réseaux issus de cette évolution sont notés 5a,6a, 7a,8a et 9a et correspondent respectivement aux réseaux entraînés avec 5,6,7,8 et 9 points test. Chacun de ces réseaux a une fitness nulle par rapport à son nombre de point test.

| Générations | Fitness atteinte | Nombre de points test | Temps CPU en minutes | Nom du réseau | Taille du réseau | Nombre de connexions |
|---|---|---|---|---|---|---|
| 3261 | 0 | 5 | 62 | 5a | 7 | 21 |
| 3608 (+347) | 0 | 6 | 72 | 6a | 12 | 41 |
| 3988 (+380) | 0 | 7 | 85 | 7a | 9 | 27 |
| 4452 (+464) | 0 | 8 | 103 | 8a | 15 | 50 |
| 4793 (+341) | 0 | 9 | 120 | 9a | 12 | 43 |

Table 4.2: *Résumé de l'évolution, temps de calcul obtenus sur un Pentium 90 ainsi que la taille (nombre de neurones cachés) et le nombre de connexions des réseaux.*

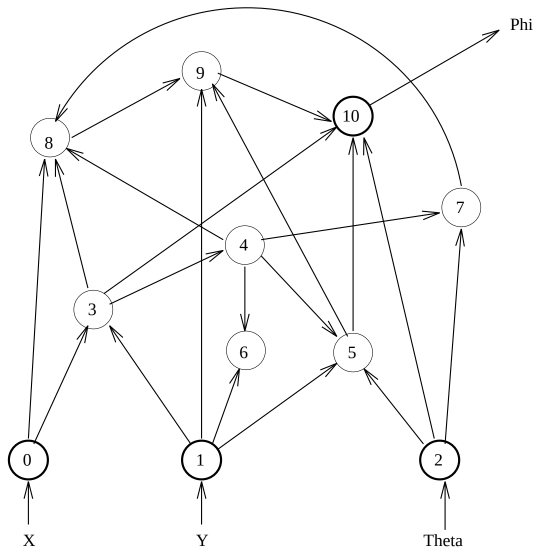
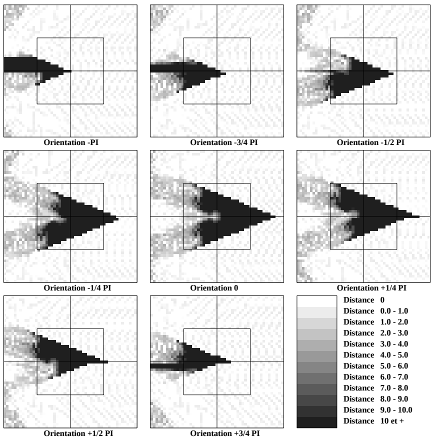
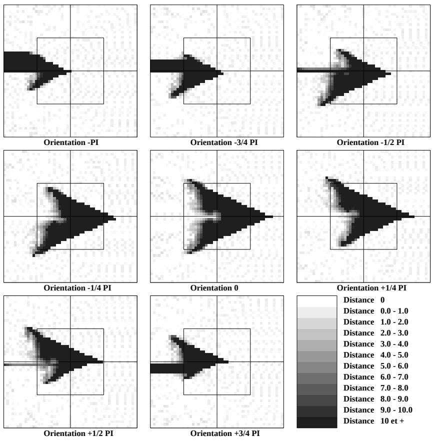


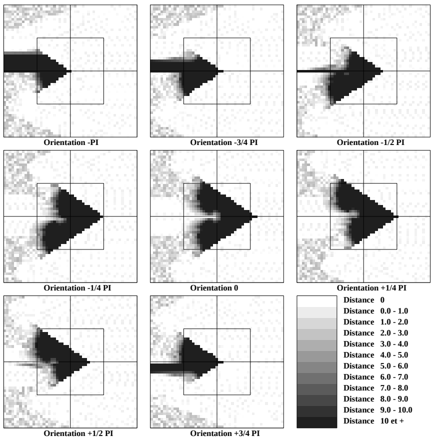

Figure 4.10: Le réseau 5a



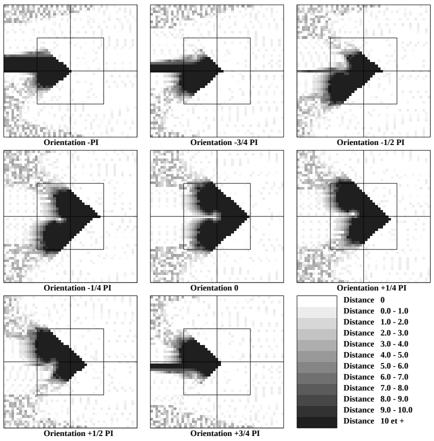

5a



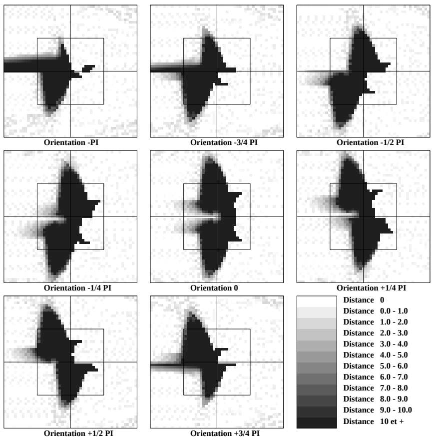

6a




7a




8a




9a

Figure 4.11: *Distance en fonction de la position initiale de la voiture pour les réseaux 5a, 5b, 5c et 5d. Orientation i PI correspond à une orientation initiale de i*$\pi$*. Chaque figure représente l'espace* $[-100, +100] \times [-100, +100]$*. Les coordonnées d'un point sont ses coordonnées de départ et sa couleur indique la distance minimum obtenue par rapport au point* $(0, 0, 0)$.

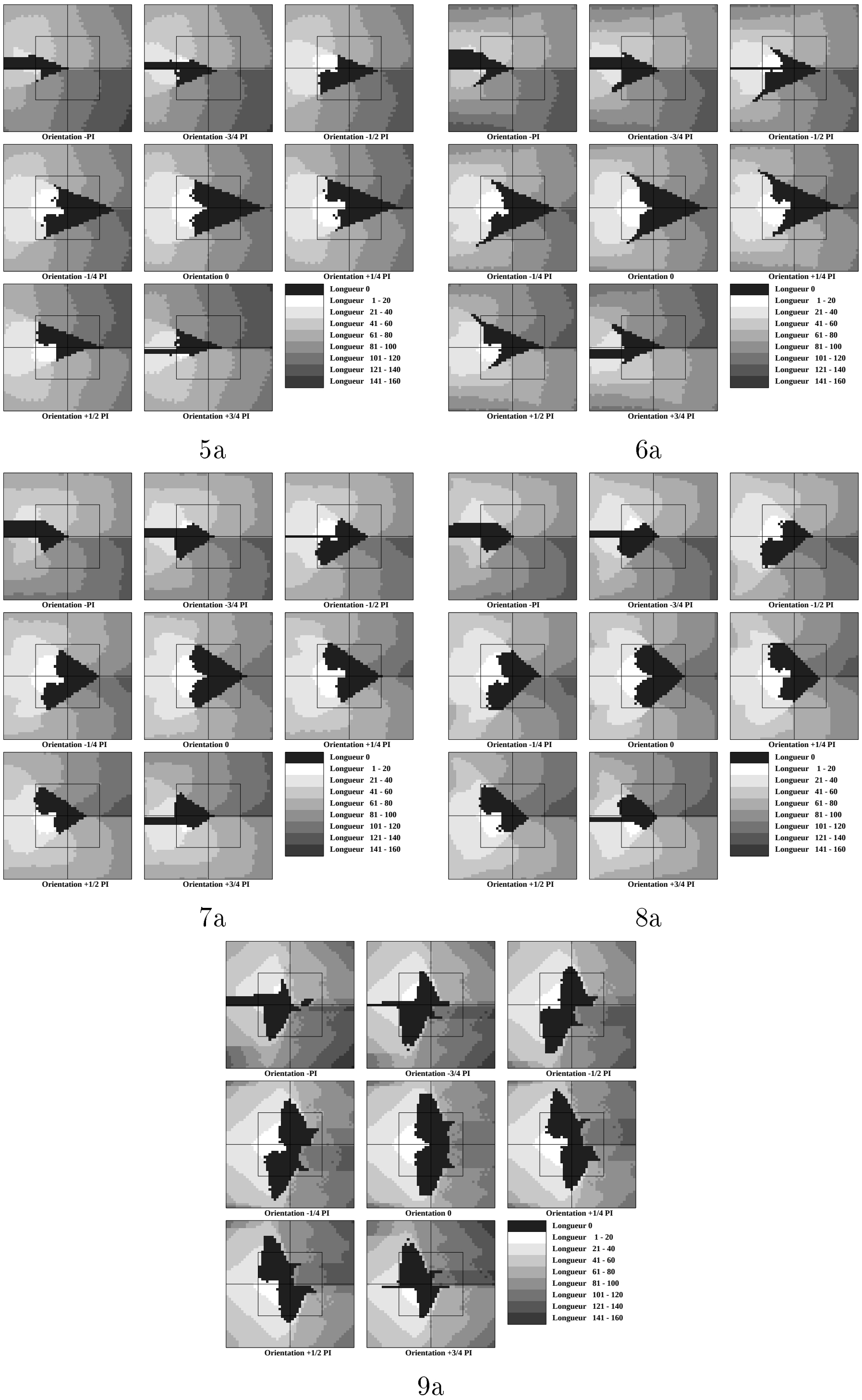
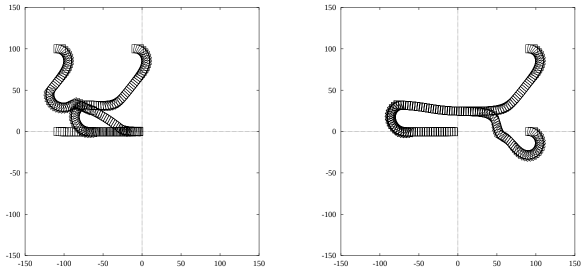
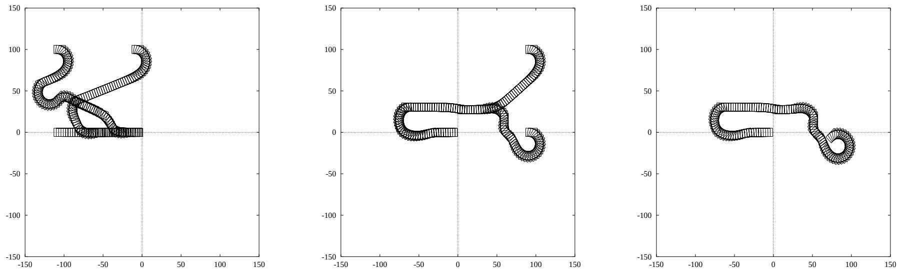


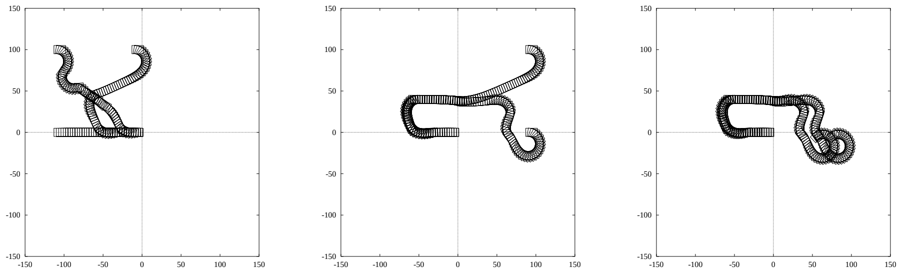

Figure 4.12: *Nombre d'itérations pour arriver au but avec un maximum de 160 itérations pour les réseaux 5a, 5b, 5c et 5d. Orientation i PI correspond à une orientation initiale de $i\pi$. Chaque figure représente l'espace $[-100, 100] \times [-100, 100]$. Les coordonnées d'un point sont ses coordonnées de départ et sa couleur indique le nombre d'itérations nécessaires pour arriver au point $(0, 0, 0)$. Les zones noires correspondent aux points d'où la voiture n'arrive pas à se garer.*



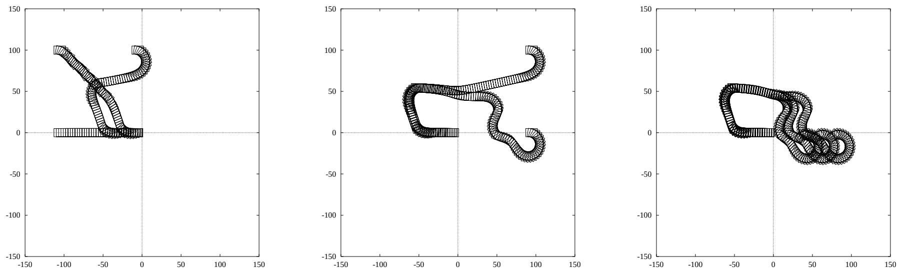
5 points test



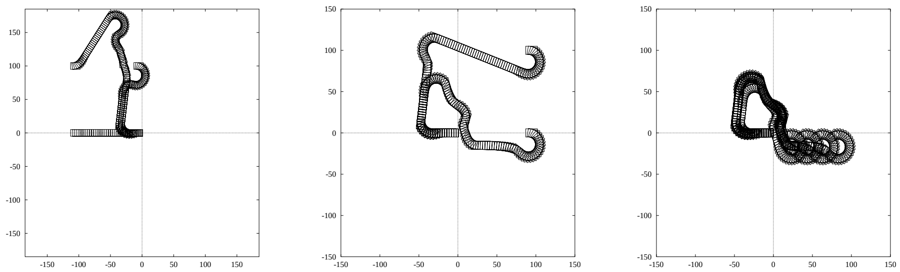
6 points test



7 points test

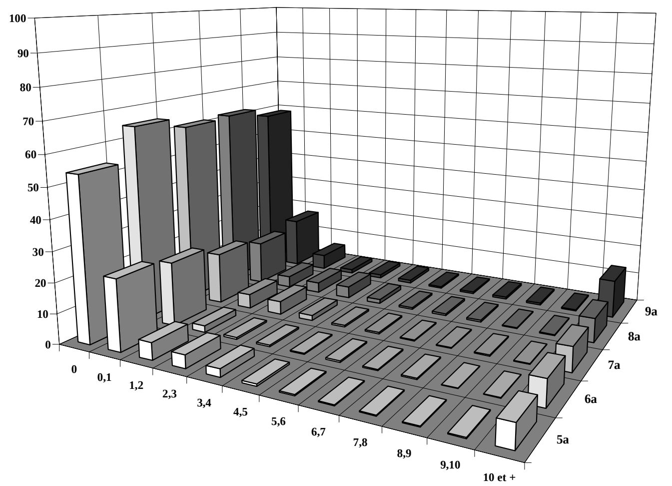


8 points test



9 points test

Figure 4.13: *Ensemble des trajectoires obtenues pour l'ensemble des points test et pour chaque réseau. A gauche les trajectoires des trois premiers points test , au milieu celles des points test 3 et 4 et à droite celles des autres points test.*




| Nom | 0 | 0-1 | 1-2 | 2-3 | 3-4 | 4-5 | 5-6 | 6-7 | 7-8 | 8-9 | 9-10 | 10+ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5a | 54.08 | 23.49 | 5.68 | 4.30 | 2.61 | 0.74 | 0.42 | 0.31 | 0.31 | 0.33 | 0.36 | 7.37 |
| 6a | 64.97 | 21.33 | 2.18 | 0.63 | 0.53 | 0.35 | 0.48 | 0.38 | 0.35 | 0.17 | 0.2 | 8.41 |
| 7a | 61.22 | 17.34 | 4.82 | 4.46 | 1.74 | 0.56 | 0.52 | 0.33 | 0.3 | 0.4 | 0.24 | 8.04 |
| 8a | 61.91 | 14.77 | 3.9 | 3.5 | 3.97 | 1.44 | 0.63 | 0.63 | 0.6 | 0.33 | 0.28 | 8 |
| 9a | 58.51 | 17.38 | 5.2 | 1.33 | 1.15 | 1.01 | 0.51 | 0.55 | 0.7 | 0.61 | 0.37 | 12.64 |

Figure 4.14: *Répartition (en pourcentage) des distances minimales par rapport à (0,0,0) sur un échantillon de 20808 trajectoires issues de 20808 points de départ (*$51 \times 51 \times 8$*) répartis uniformément sur le domaine* $[-100, +100] \times [-100, +100] \times [-\pi, \pi]$ *pour les réseaux 5a,6a,7a,8a et 9a. Il y a 12 catégories de répartition : 0, 0-1, 1-2, 2-3, 3-4, 4-5, 5-6, 6-7, 7-8, 8-9, 9-10 et 10+.*

A partir de cinq points d'entraînement on arrive à obtenir une très bonne généralisation sur tout le domaine [-100, +100] $\times$ [-100, +100] $\times[-\pi, \pi]$ et en fait les zones noires sont les seules zones sur lesquels les réseaux ne se garent pas (voir paragraphe 4.4.6).
L'ajout de points test ne change pas spectaculairement les performances : le pourcentage de distance inférieure à 4 reste identique et le taux d'échec des réseaux (pourcentage de distance 10 et +) ne varie presque pas. On constate même une augmentation pour le réseau 9a (voir la figure 4.14). qui commence d'ailleurs à présenter quelques défauts comme certaines trajectoires trop longues (voir figure 4.13) qui sont peut-être dûs à une trop grande spécialisation. De plus, pour ce réseau, on voit bien que l'ajout du neuvième point test a eu pour effet de modifier la zone où la voiture ne se gare pas et non pas d'en diminuer la taille.
Cependant, pour les réseaux 6a,7a et 8a, on constate quand même des améliorations comme un plus grand pourcentage de réussite (pourcentage de distance 0). On remarque aussi que dans l'ensemble les trajectoires sont très correctes (voir le réseau 8a sur la figure 4.13).
Etant donné que les résultats sont assez mauvais sur le domaine inclus dans [-50, 50] $\times$ [-50,

50] pour tous les $\theta$, nous avons effectué un autre calcul avec des points test appartenant à ce domaine.

### 4.4.4 Un troisième résultat

Nous avons effectué un autre calcul avec les points test de base suivant : (-50,0,0), (-50,50,0), (0,50,0), (0,25,0), (50,50,0), (50,25,0) auxquels sont venus ensuite s'ajouter 3 points test : $(0,0,\frac{3\pi}{4})$, $(50,13.333,-\frac{3\pi}{4})$ et $(-46.666,0,-\pi)$. Le nombre maximal d'itérations a été fixé á 100. Les graphiques qui suivent sont du même type que ceux du paragraphe précédent sauf qu'ils représentent les résultats pour les quatre réseaux obtenus pour 6,7,8 et 9 points test (on note ces réseaux 6b,7b,8b et 9b).
Avant d'obtenir ce résultat, l'algorithme a échoué quatorze fois à 500 générations. Le seuil de performance pour continuer l'évolution après ces 500 générations a été fixé á 15. Le temps de calcul pour la quatorzième tentative est de 514 minutes CPU sur un Pentium 90 pour atteindre une fitness d'environ 0.1 pour 9 points test (tous les points test arrivent á une distance 0 sauf le point test 8 qui ne s'approche qu'á une distance très proche de 1). Le temps total de calcul est de 610 minutes en comptant les 96 minutes pour les quatorze premières tentatives (voir table 4.3).

| Générations | Fitness atteinte | Nombre de points test | Temps CPU (en minutes) | Nom du réseau | Taille du réseau | Nombre de connexions |
|---|---|---|---|---|---|---|
| 2229 | 0 | 6 | 44 | 6b | 9 | 29 |
| 2634 (+405) | 0 | 7 | 56 | 7b | 8 | 25 |
| 4577 (+1943) | 0 | 8 | 140 | 8b | 15 | 45 |
| 11752 (+7175) | 0.25 | 9 | 514 | 9b | 13 | 44 |

Table 4.3: *Résumé de l'évolution, temps de calcul obtenus sur un Pentium 90 ainsi que la taille (nombre de neurones cachés) et le nombre de connexions des réseaux.*

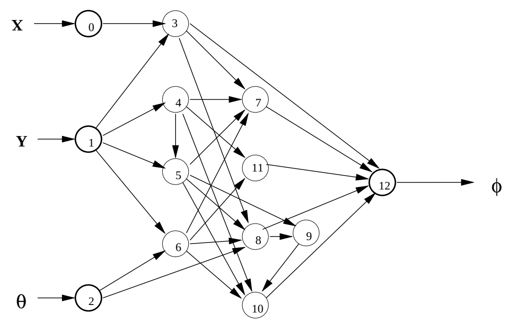



Figure 4.15: Le réseau 6b

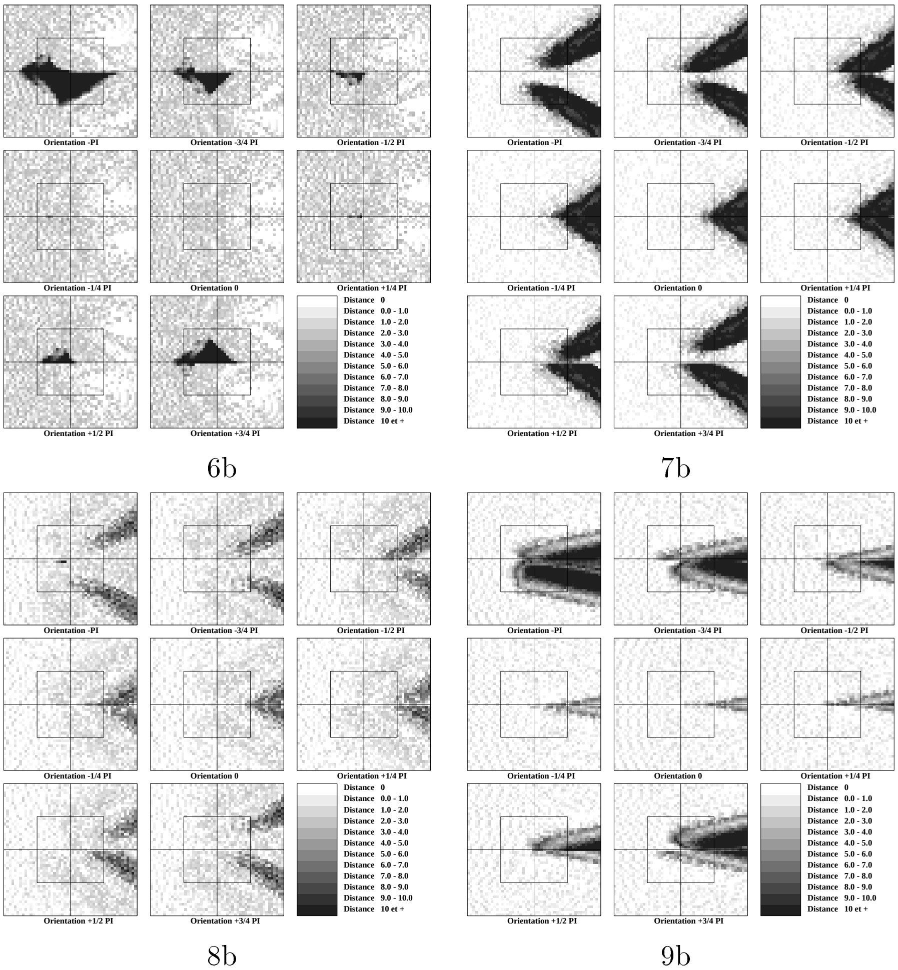
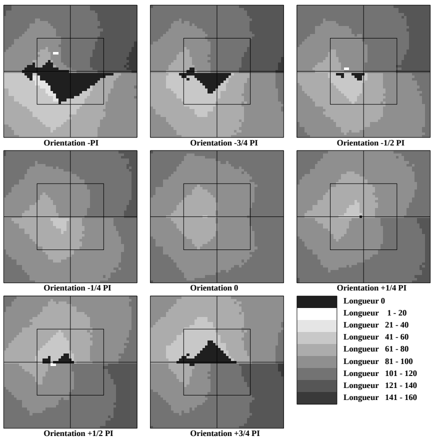
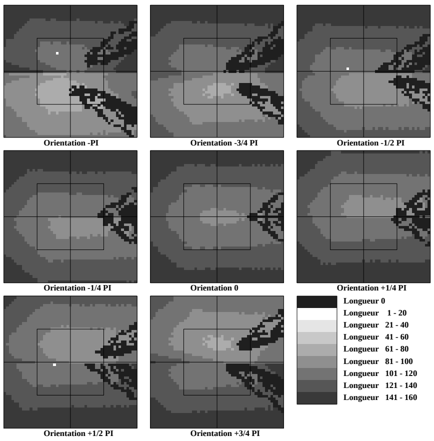


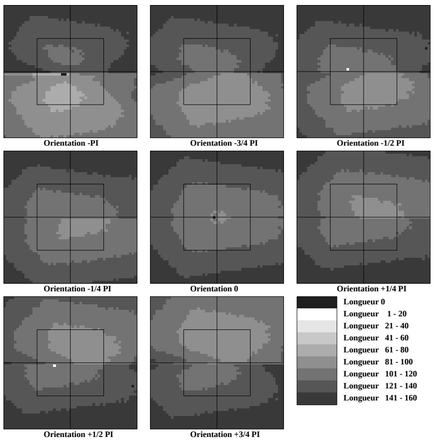

Figure 4.16: *Distance en fonction de la position initiale de la voiture pour les réseaux 6b, 7b, 8b et 9b. Orientation i PI correspond à une orientation initiale de* $i\pi$*. Chaque figure représente l'espace* $[-100, +100] \times [-100, +100]$*. Les coordonnées d'un point sont ses coordonnées de départ et sa couleur indique la distance minimum obtenue par rapport au point* $(0, 0, 0)$.



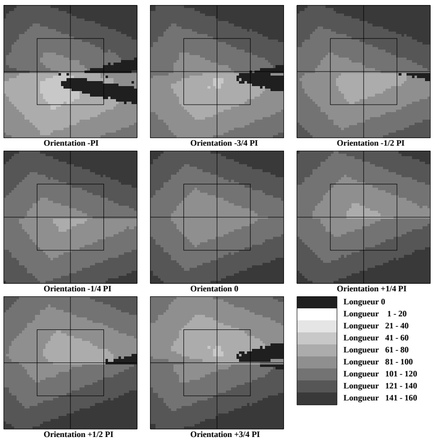

6b




7b




8b




9b

Figure 4.17: *Nombre d'itérations pour arriver au but avec un maximum de 160 itérations pour les réseaux 6b, 7b, 8b et 9b. Orientation i PI correspond à une orientation initiale de* $i\pi$*. Chaque figure représente l'espace* $[-100, 100] \times [-100, 100]$*. Les coordonnées d'un point sont ses coordonnées de départ et sa couleur indique le nombre d'itérations nécessaires pour arriver au point* $(0, 0, 0)$*. Les zones noires correspondent aux points d'où la voiture n'arrive pas à se garer.*

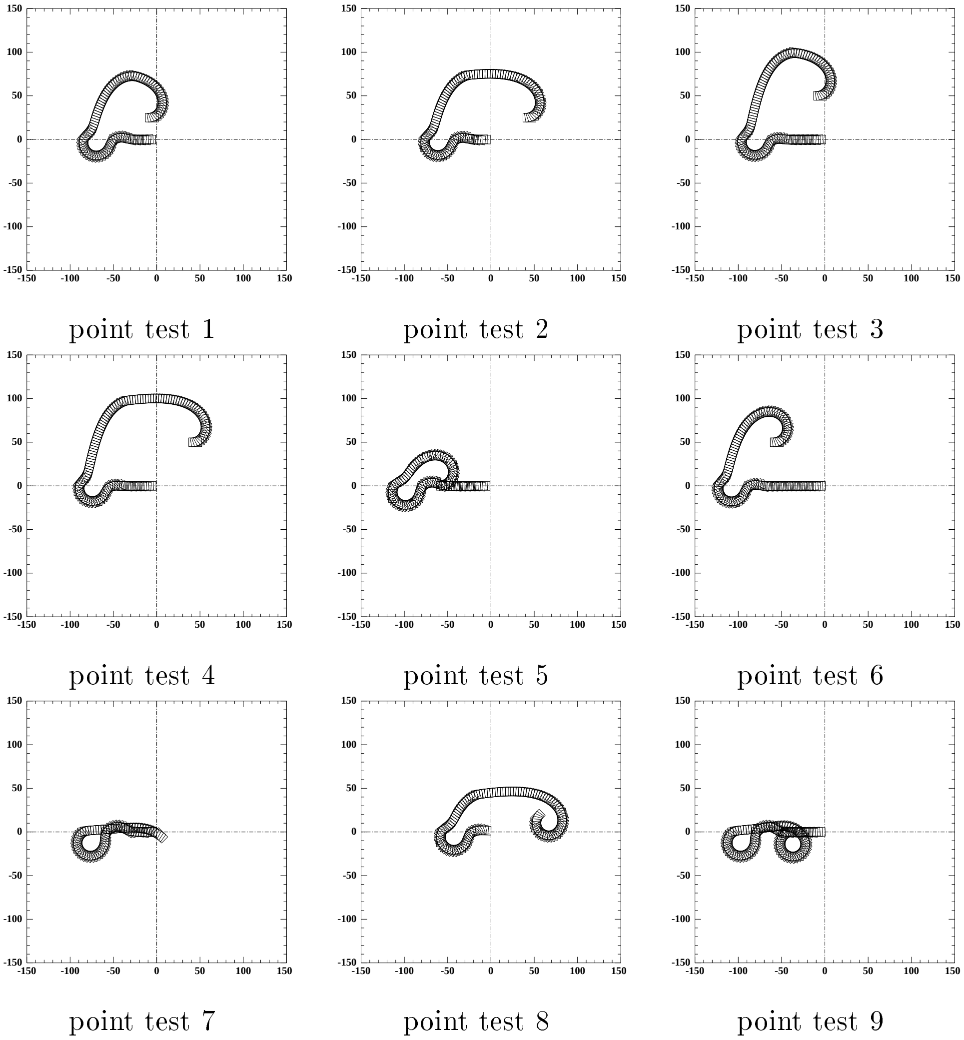
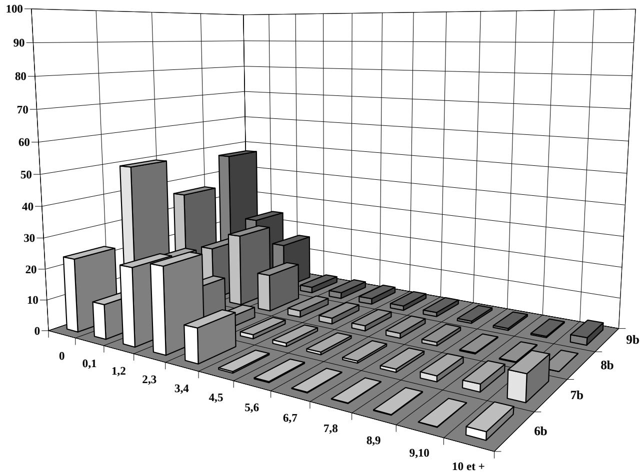



Figure 4.18: *Trajectoires obtenus pour l'ensemble des points test pour le réseaux 6b. A gauche les trajectoires des trois premiers points test , au milieu celles des points test 3 et 4 et à droite celles des autres points test. Les trajectoires pour les réseaux 7b 8b et 9b sont comparables à celles-ci.*




| Nom | 0 | 0-1 | 1-2 | 2-3 | 3-4 | 4-5 | 5-6 | 6-7 | 7-8 | 8-9 | 9-10 | 10+ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 6b | 23.53 | 11.05 | 24.55 | 26.76 | 10.63 | 0.47 | 0.3 | 0.17 | 0.15 | 0.16 | 0.12 | 2.1 |
| 7b | 49.29 | 19.38 | 11.48 | 3.40 | 1.31 | 1.08 | 0.81 | 0.69 | 0.9 | 1.73 | 2.17 | 7.73 |
| 8b | 36.16 | 18.21 | 24.27 | 12.37 | 2.17 | 1.93 | 1.62 | 1.58 | 1.1 | 0.32 | 0.2 | 0.02 |
| 9b | 47.03 | 23.90 | 15.94 | 2.11 | 2.04 | 1.97 | 1.54 | 1.38 | 0.72 | 0.60 | 0.39 | 2.35 |

Figure 4.19: *Répartition (en pourcentage) de la distance minimum par rapport à (0,0,0) d'un échantillon de 20808 trajectoires issues de 20808 points de départ (*$51 \times 51 \times 8$*) repartis uniformément sur le domaine* $[-100,+100]\times[-100,+100]\times[-\pi,\pi]$ *pour les réseaux 6b,7b,8b et 9b. Il y a 12 catégories de répartition : 0, 0-1, 1-2, 2-3, 3-4, 4-5, 5-6, 6-7, 7-8, 8-9, 9-10 et 10+.*

L'ajout de points test pour ce deuxième résultat a apporté la même amélioration que pour le premier résultat : un taux de réussite très supérieur par rapport à 6b bien que le pourcentage de distances inférieures à 4 soit meilleur pour 6b (95%) alors qu'il est respectivement d'environ 85%, 92% et 88% pour 7b, 8b et 9b. On note cependant, comme le montrent les graphiques 7b et 8b de la figure 4.16, que le passage de sept à huit points test (ajout du point $(50,13.333,-\frac{3\pi}{4})$) a eu exactement l'effet recherché : une forte atténuation et même une suppression des zones noires (zones d'où le véhicule ne se gare pas), le taux d'échec passant de 7,73% à 0,02%. Par ailleurs, le fait que les taux d'échecs sont très faibles est sans doute lié au fait que toutes les trajectoires sont de même nature. Ainsi, toutes les trajectoires issue des points de départ pour lesquels le véhicule se présente "face" au but ne sont pas acceptables bien qu'elles arrivent parfaitement au point (0,0,0). Pour résoudre ce problème, on peut alors utiliser un autre réseau sur ces zones comme le montre le paragraphe suivant.

### 4.4.5 Utilisation de plusieurs réseaux

Comme on peut le constater sur les différents résultats, on voit que les réseaux obtenus après avoir ajouté un point test généralisent moins les réseaux déjà existants qu'ils les complètent, en ce sens que les zones de mauvaise performance ne sont pas les mêmes. Ainsi, nous avons couplé plusieurs réseaux en choisissant à chaque fois celui qui se garera le mieux pour un point de départ donné (s'il y en a plusieurs, on choisit celui qui a la trajectoire la plus courte). Nous précisons que le réseau choisi effectuera entiérement le parcours et que cette stratégie est appliquée en temps réel.
Nous présentons trois résultats :

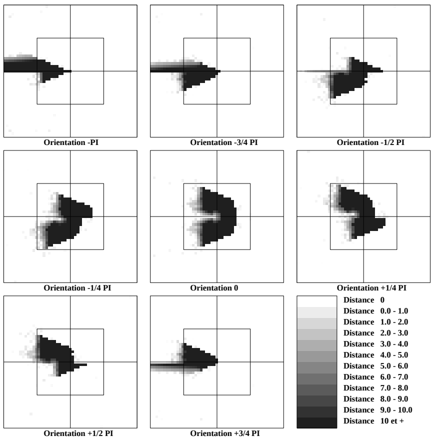
- Couplage des 5 réseaux obtenus pour le premier résultat.
- Couplage des 4 réseaux obtenus pour le deuxième résultat.
- Couplage de ces 9 réseaux.

| 5 réseaux | 1a | 1b | 1c | 1d | 1e |
|---|---|---|---|---|---|
| % d'utilisation | 8.95 | 11.64 | 46.05 | 15.79 | 17.55 |

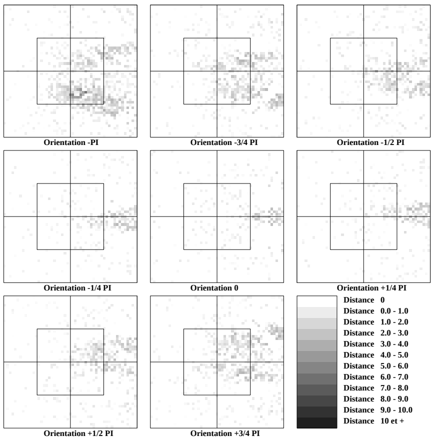
| 4 réseaux | 2a | 2b | 2c | 2d |
|---|---|---|---|---|
| % d'utilisation | 27.62 | 20.81 | 14.25 | 37.32 |

| 9 réseaux | 1a | 1b | 1c | 1d | 1e | 2a | 2b | 2c | 2d |
|---|---|---|---|---|---|---|---|---|---|
| % d'utilisation | 7.02 | 8.43 | 37.44 | 12.86 | 13.33 | 8.29 | 3.51 | 2.23 | 6.89 |
| % d'utilisation | 79.08 | | | | | 20.92 | | | |

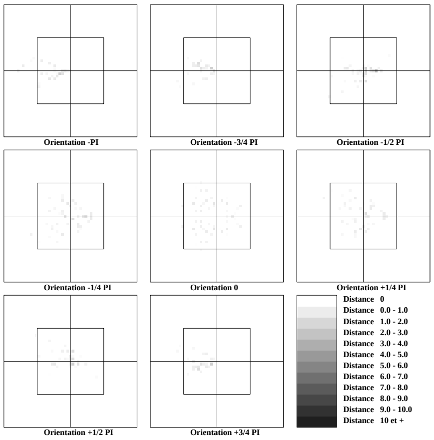
Table 4.4: *Répartition (en pourcentage) de l'utilisation de chaque réseau sur un échantillon de 20808 trajectoires issues de 20808 points de départ* ($51 \times 51 \times 8$) *répartis uniformément sur le domaine* $[-100, +100] \times [-100, +100] \times [-\pi, \pi]$.




5 réseaux

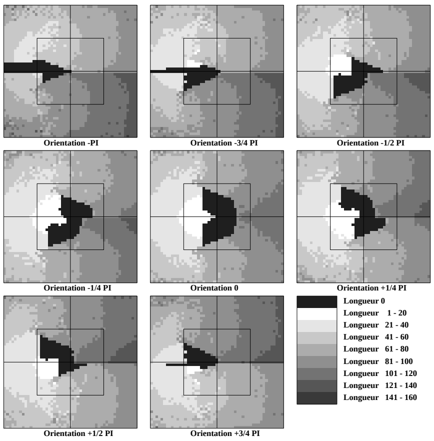



4 réseaux

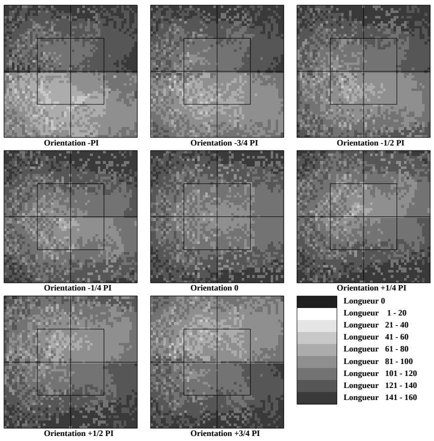



9 réseaux

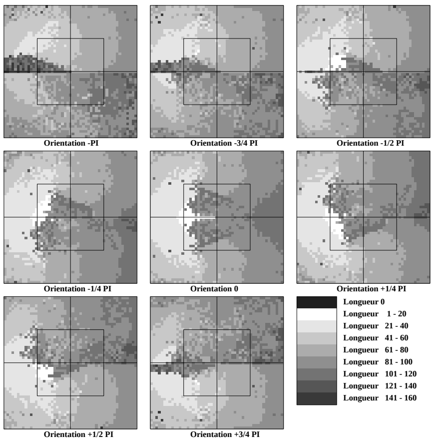
Figure 4.20: *Distance en fonction de la position initiale de la voiture pour les 3 couplages de réseaux. Orientation i PI correspond à une orientation initiale de* $i\pi$*. Chaque figure représente l'espace* $[-100, +100] \times [-100, +100]$*. Les coordonnées d'un point sont ses coordonnées de départ et sa couleur indique la distance minimum obtenue par rapport au point* $(0, 0, 0)$*.*




5 réseaux




4 réseaux




9 réseaux

Figure 4.21: *Nombre d'itérations pour arriver au but avec un maximum de 160 itérations pour les 3 couplages de réseaux. Orientation i PI correspond à une orientation initiale de* $i\pi$. *Chaque figure représente l'espace* $[-100, 100] \times [-100, 100]$. *Les coordonnées d'un point sont ses coordonnées de départ et sa couleur indique le nombre d'itérations nécessaire pour arriver au point* $(0, 0, 0)$. *Les zones noires correspondent aux points d'où la voiture n'arrive pas à se garer.*

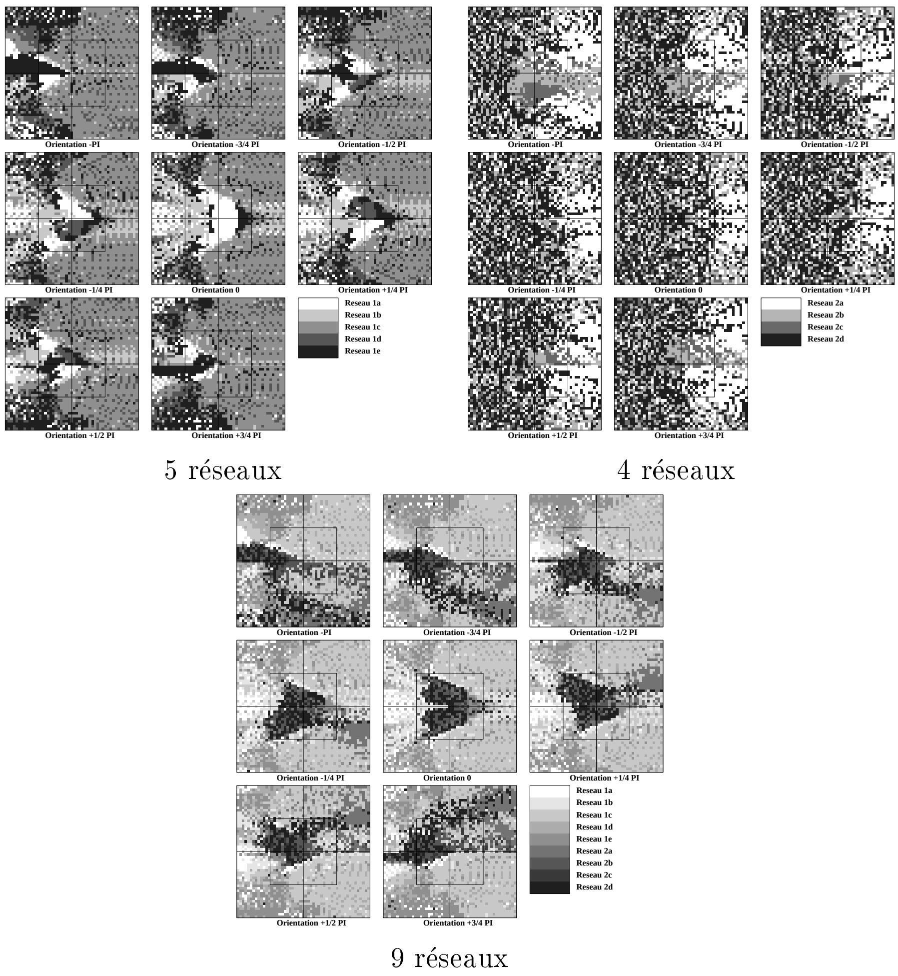
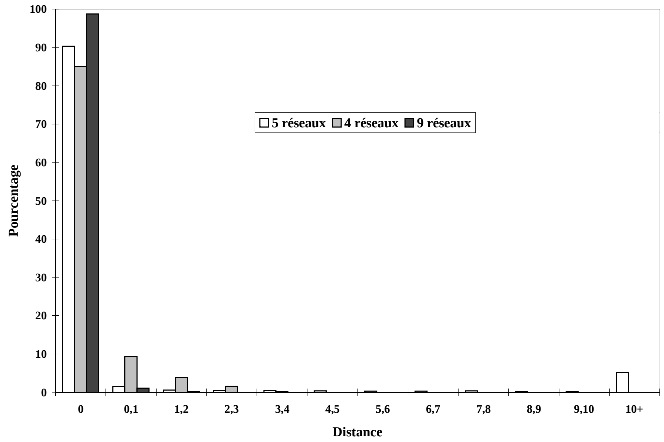



Figure 4.22: *Réseau utilisé en fonction de la position initiale de la voiture pour les 3 couplages de réseaux. Orientation i PI correspond à une orientation initiale de* $i\pi$. *Chaque figure représente l'espace* $[-100,+100]\times[-100,+100]$. *Les coordonnées d'un point sont ses coordonnées de départ et sa couleur indique la distance minimum obtenue par rapport au point* $(0,0,0)$.

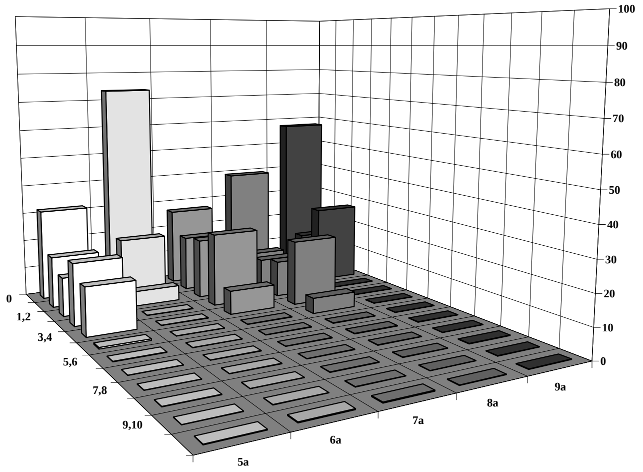



| | 0 | 0-1 | 1-2 | 2-3 | 3-4 | 4-5 | 5-6 | 6-7 | 7-8 | 8-9 | 9-10 | 10+ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5 réseaux | 90.33 | 1.51 | 0.56 | 0.38 | 0.45 | 0.32 | 0.31 | 0.25 | 0.32 | 0.24 | 0.13 | 5.16 |
| 4 réseaux | 85.01 | 9.29 | 3.91 | 1.53 | 0.21 | 0.03 | 0.005 | 0.005 | 0 | 0 | 0 | 0 |
| 9 réseaux | 98.71 | 1.08 | 0.18 | 0.024 | 0.0048 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Figure 4.23: *Répartition (en pourcentage) des distances minimales par rapport à (0,0,0) sur un échantillon de 20808 trajectoires issues de 20808 points de départ* ($51 \times 51 \times 8$) *répartis uniformément sur le domaine* $[-100, +100] \times [-100, +100] \times [-\pi, \pi]$ *et pour les 5 réseaux. Il y a 12 catégories de répartition : 0, 0-1, 1-2, 2-3, 3-4, 4-5, 5-6, 6-7, 7-8, 8-9, 9-10 et 10+.*

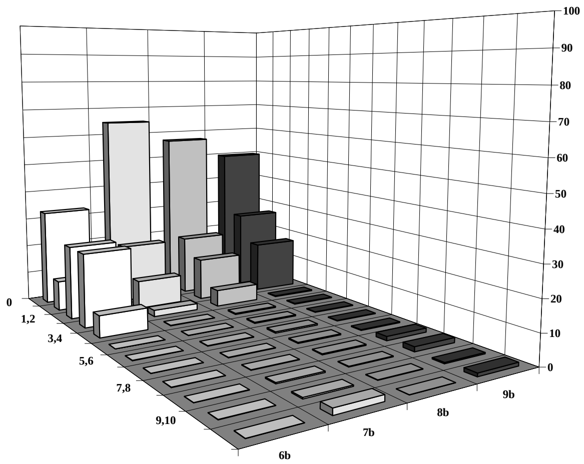
### 4.4.6 Comportement des réseaux sur des distances plus grandes

Nous donnons les graphiques de répartition de distance pour les réseaux 5a, 6a, 7a, 8a, 9a et 6b, 7b, 8b, 9b pour un échantillon de 323208 trajectoires ($201 \times 201 \times 8$) représentatifs du domaine [-1000, +1000] $\times$ [-1000, +1000] $\times$ $[-\pi, \pi]$ (figures 4.4.6 et 4.4.6) ainsi que quelques exemples de trajectoires (figure 4.26). Nous ne présentons pas les graphiques du type 4.11 ou 4.16 car ce sont des fichiers beaucoup trop grands. Cependant, ces graphiques montrent que les seules zones d'où les réseaux n'arrivent pas à se garer sont celles que nous avons déjà mis en évidence dans les figures 4.11 et 4.16.




| Nom | 0 | 0-1 | 1-2 | 2-3 | 3-4 | 4-5 | 5-6 | 6-7 | 7-8 | 8-9 | 9-10 | 10+ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5a | 31.58 | 17.57 | 13.16 | 20.90 | 16.13 | 0.48 | 0.00 | 0.00 | 0.00 | 0.00 | 0.01 | 0.16 |
| 6a | 73.65 | 20.86 | 5.18 | 0.01 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.28 |
| 7a | 26.98 | 19.12 | 20.54 | 24.99 | 8.07 | 0.01 | 0.00 | 0.00 | 0.00 | 0.00 | 0.01 | 0.28 |
| 8a | 39.24 | 10.10 | 10.49 | 12.50 | 22.24 | 5.24 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | 0.16 |
| 9a | 57.56 | 15.04 | 27.18 | 0.02 | 0.01 | 0.02 | 0.01 | 0.01 | 0.01 | 0.01 | 0.00 | 0.14 |

Figure 4.24: *Répartition (en pourcentage) des distances minimum par rapport à (0,0,0) sur un échantillon de 323208 trajectoires issues de 323208 points de départ (*$201 \times 201 \times 8$*) répartis uniformément sur le domaine* $[-1000, +1000] \times [-1000, +1000] \times [-\pi, \pi]$ *pour les réseaux 5a, 6a, 7a, 8a, 9a. Il y a 12 catégories de répartition : 0, 0-1, 1-2, 2-3, 3-4, 4-5, 5-6, 6-7, 7-8, 8-9, 9-10 et 10+.*




| Nom | 0 | 0-1 | 1-2 | 2-3 | 3-4 | 4-5 | 5-6 | 6-7 | 7-8 | 8-9 | 9-10 | 10+ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 6b | 32.86 | 10.24 | 24.78 | 24.78 | 7.27 | 0.04 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.02 |
| 7b | 64.76 | 20.00 | 9.68 | 2.18 | 0.18 | 0.10 | 0.15 | 0.12 | 0.16 | 0.38 | 0.39 | 1.89 |
| 8b | 57.02 | 20.37 | 14.47 | 5.77 | 0.43 | 0.38 | 0.43 | 0.35 | 0.30 | 0.20 | 0.16 | 0.10 |
| 9b | 49.75 | 26.97 | 17.45 | 0.25 | 0.24 | 0.19 | 0.23 | 0.33 | 1.41 | 1.62 | 0.41 | 1.15 |

Figure 4.25: *Répartition (en pourcentage) des distances minimum par rapport à (0,0,0) sur un échantillon de 323208 trajectoires issues de 323208 points de départ* $(201 \times 201 \times 8)$ *répartis uniformément sur le domaine* $[-1000, +1000] \times [-1000, +1000] \times [-\pi, \pi]$ *pour les réseaux 6b,7b,8b et 9b. Il y a 12 catégories de répartition : 0, 0-1, 1-2, 2-3, 3-4, 4-5, 5-6, 6-7, 7-8, 8-9, 9-10 et 10+.*

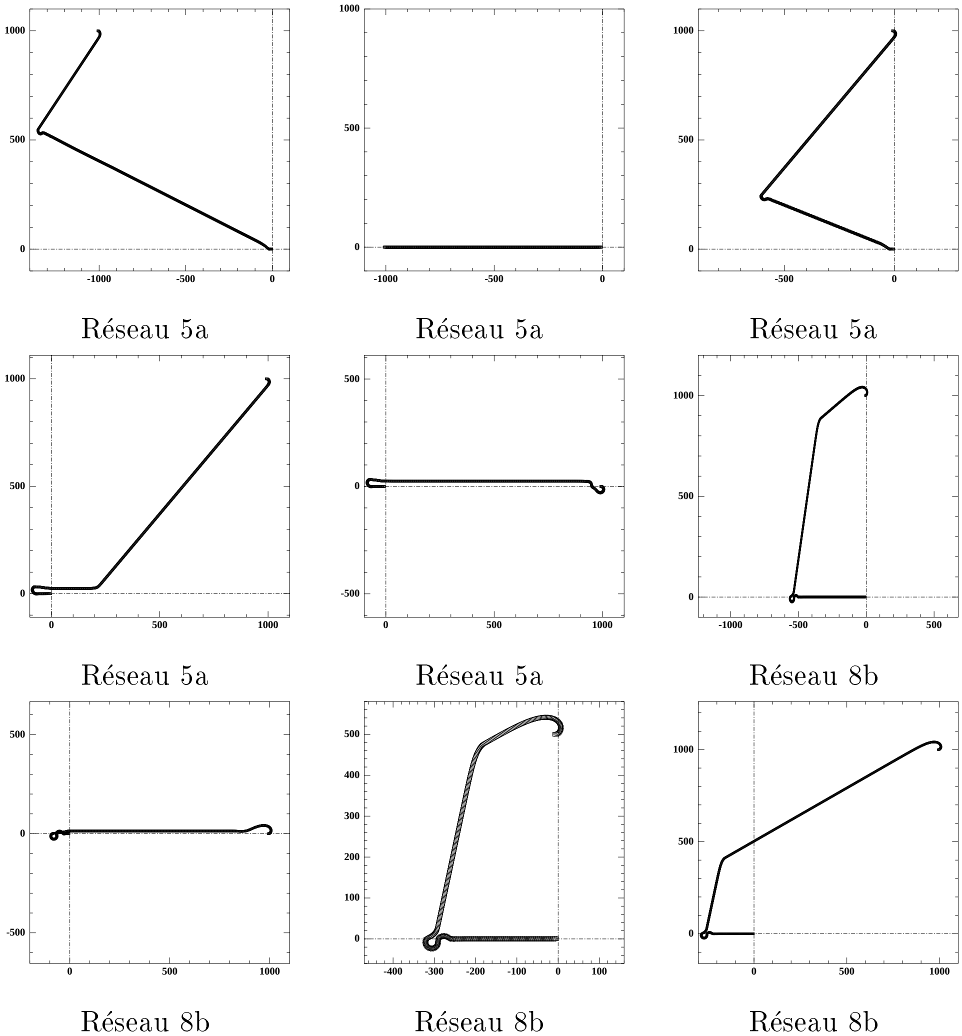
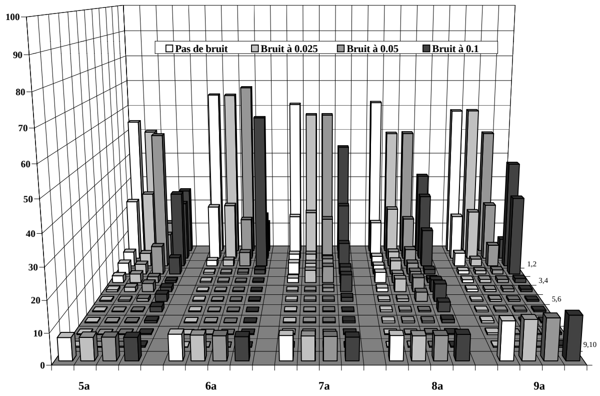



Figure 4.26: *Quelques exemples de trajectoires.*

### 4.4.7 Influence du bruit sur le comportement des réseaux

Nous étudions dans ce paragraphe l'influence d'une perception bruitée de la position de la voiture à chaque itération par le réseau. On considère que l'entrée du réseau est maintenant ($X_i + a\varepsilon_{X_i}$, $Y_i + a\varepsilon_{Y_i}$, $\theta + a\varepsilon_{\theta_i}$) oú $\varepsilon$ est une variable aléatoire uniforme sur [-1,1] et $a$ l'amplitude du bruit. Nous avons effectué les calculs pour des bruits d'amplitude maximale 0.1,0.05 et 0.025 unités. Les graphiques de la figure 4.27 correspondent respectivement aux réseaux issus du premier et du deuxième résultat.



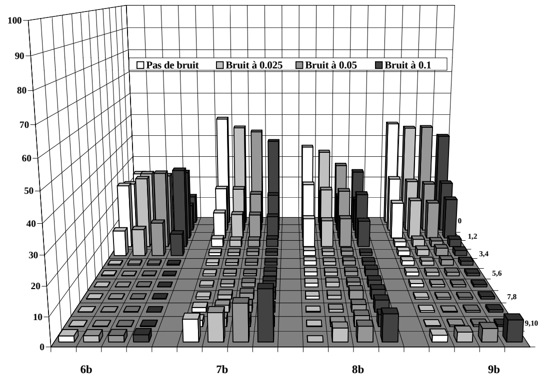




Figure 4.27: *Répartition (en pourcentage) des distances minimum par rapport à (0,0,0) sur un échantillon de 20808 trajectoires issues de 20808 points de départ (*$51 \times 51 \times 8$*) répartis uniformément sur le domaine* $[-100, +100] \times [-100, +100] \times [-\pi, \pi]$ *pour les réseaux 5a,6a,7a,8a et 9a (en haut) et les réseaux 6b,7b,8b et 9b (en bas). Il y a 12 catégories de distance : 0, 0-1, 1-2, 2-3, 3-4, 4-5, 5-6, 6-7, 7-8, 8-9, 9-10 et 10+.*

Comme on peut le constater, l'ajout de bruit entraîne une baisse du taux de réussite (mais le pourcentage de distance inférieur à 3 reste plus ou moins constant) et une très faible augmentation du taux d'échec pour les réseaux 5a, 6a, 7a, 8a ,9a et 6b alors que pour les réseaux 7b, 8b et 9b le taux de réussite baisse d'autant que le taux d'échec augmente. On peut cependant conclure que la plupart des réseaux résistent bien au bruit.

### 4.4.8 Conclusion

Les résultats présentés ci-dessus montrent que l'algorithme a réussi à trouver plusieurs réseaux qui se garaient bien depuis presque tous les points de départ et même depuis tous les points de départ mais avec certaines trajectoires non optimales. Ces résultats ont été obtenus en utilisant l'algorithme présenté au chapitre 2 et en ajoutant un point test appartenant à une zone peu performante à chaque fois que la fonction de fitness devenait nulle. On a ainsi pu rajouter quatre et cinq points test comme nous l'avons montré dans les deux résultats. On constate que cette méthode a plus pour effet d'augmenter le taux de réussite ( pourcentage de points atteignant le point (0,0,0)) que de diminuer le taux d'échec (pourcentage de points qui sont très loin du point (0,0,0)) qui était l'objectif initial. Cependant nous avons mis en évidence un cas où cette méthode a fait diminuer le taux d'échec de 8% à 0,02%.
Afin d'obtenir des réseaux avec des taux de réussite proche de 100% et des taux d'échec nul et aussi pour améliorer la qualité des trajectoires, on a vu que le couplage de plusieurs réseaux était très efficace.
Par ailleurs, nous avons montré que les réseaux trouvés résistaient bien à l'ajout de bruit à chaque pas de notre simulation de trajectoires.

# Chapitre 5

# Adaptation d'une méthode de Newton et d'une méthode de continuation pour le calcul d'un point fixe d'un réseau de neurones récurrent

## 5.1 Introduction

Nous présentons dans ce chapitre l'adaptation de la méthode de Newton pour le calcul d'un point fixe d'un réseau de neurones récurrent. On sait que le problème qui est lié à l'application de l'algorithme de Newton réside dans le choix du point de départ qui doit être "assez proche" de la solution. Du point de vue des réseaux de neurones récurrents, il faut partir d'une initialisation des neurones proche de leur état final. Ainsi, en adaptant un résultat sur la méthode de Newton [SS93] qui consiste à partir d'une solution triviale (obtenue lorsque tous les poids des connexions sont nuls) et par une méthode de continuation (en amenant progressivement les valeurs des poids à leur valeurs réelles), à suivre la solution en appliquant successivement la méthode de Newton.
Les résultats présentés sont valables pour trois types de fonction de transfert: sigmoïde, arctg et les fonctions gaussiennes utilisées dans les réseaux dits *RBF*.
Après avoir rappelé un ensemble de résultats concernant la méthode de Newton, nous présenterons le résultat principal ainsi que sa démonstration.

## 5.2 La méthode de Newton

Soit $E$ et $F$ deux espaces de Banach. On considère une application analytique $f : D_r(x_0) \to F$ avec $x_0 \in E$ et $D_r(x_0) = \{x \in E \quad | \quad \|x - x_0\| \leq r\}$ et on suppose que les quantités suivantes sont bien définies pour $x \in D_r(x_0)$

$$
\begin{aligned}
\beta(f,x) &= \|Df(x)^{-1}f(x)\| \\
\gamma(f,x) &= \sup_{k>1}\left(\frac{\|Df(x)^{-1}D^kf(x)\|}{k!}\right)^{\frac{1}{k-1}} \\
\alpha(f,x) &= \beta(f,x)\gamma(f,x)
\end{aligned}
\tag{5.1}
$$

La méthode de Newton construit une suite de points $\{x_i\}_{i\geq 1} \in D_r(x_0)$ tels que

$$
\begin{aligned}
x_n &= x_{n-1} - Df(x_{n-1})^{-1}f(x_{n-1}) \\
&= N_f(x_{n-1}) \\
&= N_f^n(x_0), \quad x_0 \in D_r(x_0)
\end{aligned}
$$

Par ailleurs, on utilisera les quantités suivantes:

$$\tau(\alpha) = \frac{(1+\alpha) - \sqrt{(1+\alpha)^2 - 8\alpha}}{4} \quad , \quad 0 \leq \alpha \leq 3 - 2\sqrt{2} \sim 0.1715$$

et

$$\alpha_0 = \frac{1}{4}(13 - 3\sqrt{17}) \sim 0.157671$$

**Théorème 4** *Soit $f : D_r(x_0) \to F$ une application analytique, $\beta = \beta(f,x_0)$, $\alpha = \alpha(f,x_0)$, $\gamma = \alpha\beta$ et $r \geq \frac{\tau(\alpha)}{\gamma}$. Si $\alpha \leq \alpha_0$ la suite de Newton $x_1, x_2, \ldots$ est définie, converge vers $\zeta \in D_r(x_0)$ avec $f(\zeta) = 0$ et on a*

$$\|x_{n+1} - x_n\| \quad \leq \left(\frac{1}{2}\right)^{2^n - 1} \|x_1 - x_0\| \quad \forall n \geq 1 \tag{5.2}$$

Si un point $x_0 \in E$ vérifie (5.2) alors $x_0$ est appelé un zéro approché de $f$ et $\zeta$ le (vrai) zéro associé.

**Corollaire 1** *Soit $f : E \to F$ une application analytique, $x_0 \in E$ vérifiant $\alpha = \alpha(f,x_0) \leq \alpha_0$ et $\zeta$ le zéro associé. Alors la suite de Newton $\{x_n\}_{n\geq 1}$ vérifie $\|x_n - \zeta\| \leq \varepsilon$ si $n \geq (\log|\log\frac{\tau(\alpha)}{\varepsilon\gamma}|) + 1$.*

Le résultat suivant va nous permettre de remplacer la condition sur $\alpha = \alpha(f,x_0)$ par une condition moins restrictive en appliquant la méthode de Newton de manière répétitive.

Une homotopie $f_t : E \to F$, $0 \leq t \leq 1$ est une famille d'application analytique. De plus, l'application induite $[0,1] \times E \to F$ est continue et le pas associé est une application continue $[0,1] \to E$, $t \mapsto \zeta_t$ vérifiant $f_t(\zeta_t) = 0$ et $Df_t(\zeta_t) : E \to F$ est un isomorphisme. $\{f_t, \zeta_t\}$ est appelé le couple homotopie-pas.

Soit $T = \{t_0 = 0, t_1, ..., t_i, ..., t_k = 1\}$, $t_i < t_{i+1}$, $|T| = k$ et $x_i = N_{f_{t_i}}(x_{i-1}) \quad i = 1, ..., k$. On dit que la méthode de Newton suit le couple homotopie-pas $\{f_t, \zeta_t\}$ par rapport à $T$ pourvu que les $x_i$ soient bien définis, $\alpha(f_{t_i}, x_i) < \alpha_0$ et $x_i$ est le zéro approché de $f_{t_i}$, $i = 1, ..., |T|$.

**Théorème 5** *(Shub & Smale) Soit $F = \{f_t, \zeta_t\}$ un couple homotopie-pas tel qu'il a été défini ci-dessus. On pose $\Delta = \frac{1}{k}$, $k \in \mathbb{N}^+$, $\bar{\gamma} \in \mathbb{R}^+$ tel que $\beta(f_{t'}, \zeta_t) \leq \frac{\bar{\alpha}}{\bar{\gamma}}$ et $\gamma(f_{t'}, \zeta_t) \leq \bar{\gamma}$ si $|t' - t| \leq \Delta$ ($\bar{u} \simeq 0.02207$ et $\bar{\alpha} \simeq 0.08019$). Soit $\|y^0 - \zeta_0\| \leq \frac{\bar{u}}{\bar{\gamma}}$ alors si $T = \{t_0 = 0, t_1 = \Delta, ..., t_i = i\Delta, ..., t_k = k\Delta = 1\}$ la méthode de Newton suit $F$. On a en fait*

$$\|y^i - \zeta_{t_i}\| \leq \frac{\bar{u}}{\bar{\gamma}} \quad i = 1, ..., k$$

*avec $\zeta_{t_i}$ le vrai zéro de $f_{t_i}$.*

Le nombre d'iterations de Newton nécessaire pour avoir une approximation à $\varepsilon$ près du zéro $\zeta_1$ de la fonction $f_1$ est alors

$$|T| + (\log|\log \frac{\tau(\alpha)}{\varepsilon\gamma}|)$$

avec $\alpha = \alpha(f_1, x_1)$ et $\gamma = \gamma(f_1, x_1)$.

Le paragraphe suivant explique en détails l'application de ce résultat aux réseaux de neurones récurrents.

## 5.3 Adaptation aux réseaux de neurones récurrents

L' évolution d'un réseau de neurones récurrent de taille $N$ peut être modélisée par l'équation différentielle

$$\frac{dy}{dt} = -y + \sigma(W, y, S) \tag{5.3}$$

$y = \{y_i\}_{i=1}^N$ étant le vecteur des activations, $W = \{w_{ij}\}_{i,j=1,...,N}$ la matrice des poids du réseau. $\sigma$ est la fonction de transfert dont le ou les paramètres peuvent dépendre de chaque neurone. $S = \{s_i\}_{i=1}^N$ avec $s_i = 0$ si le neurone $i$ n'est pas un neurone d'entrée.

Cette équation peut être discrétisée par un schéma d'Euler, ce qui donne

$$\begin{cases} y^{n+1} = (1 - \Delta t) y^n + \Delta t \sigma(W, y^n, S) \\ y^0 = 0 \end{cases} \tag{5.4}$$

si l'on pose

$$\Phi(y) = y - \sigma(W, y, S)$$

alors un point fixe $y^* = \{y_i^*\}_{i=1}^N$ vérifie $\Phi(y^*) = 0$.

Dans tout ce qui va suivre les calculs vont être faits pour trois fonctions de transfert différentes que l'on appellera dorénavant $g, h$ et $f$. $\sigma$ désignera au choix $g, h$ ou $f$. De plus et sauf indication contraire on pourra remplacer $g$ par $h$.
On a

$$\sigma(W,y,S) = \{\sigma_i(W,y,S)\}_{i=1}^{N}$$

$$g_i(W,y,S) \equiv g_i(\sum_{j=1}^{N} w_{ij}y_j + s_i) = \frac{1}{1+e^{-\theta_i(\sum_{j=1}^{N} w_{ij}y_j+s_i)}}, \quad \theta_i \in \mathbb{R}$$

$$h_i(W,y,S) \equiv h_i(\sum_{j=1}^{N} w_{ij}y_j + s_i) = \frac{2}{\pi}\arctan\left(\theta_i(\sum_{j=1}^{N} w_{ij}y_j + s_i)\right), \quad \theta_i \in \mathbb{R}$$

$$f_i(W,y,S) \equiv f_i(\sum_{j=1}^{N} (w_{ij}y_j - d_i)^2 + (s_i-d_i)^2) = e^{-\theta_i \sum_{j=1}^{N}(w_{ij}y_j-d_i)^2+(s_i-d_i)^2}, \quad \theta_i > 0, d_i \in \mathbb{R}$$

De plus, on pose

$$\Phi^\sigma(y) = y - \sigma(W,y,S)$$

Lorsqu'il n'y aura pas d'ambiguïté on utilisera la lettre $\Phi$.
Pour appliquer la méthode de Newton, il est nécessaire de calculer les matrices $D\Phi^\sigma(y)$.
Pour la fonction $g$ (et $h$) on a

$$(\Phi(y))_i = y_i - g_i(\sum_{j=1}^{N} w_{ij}y_j + s_i) \quad j = 1,...,N \tag{5.5}$$

et

$$\begin{aligned} D_j(\Phi(y))_i &= \delta_{ij} - \frac{\partial}{\partial y_j}\left(g_i(\sum_{j=1}^{N} w_{ij}y_j + s_i)\right) \quad i,j = 1,...,N \\ &= \delta_{ij} - w_{ij}g_i'(\sum_{j=1}^{N} w_{ij}y_j + s_i) \quad i,j = 1,...,N \end{aligned}$$

on a donc

$$D\Phi^g(y) = I - G(W,y,S)W \quad \text{avec} \quad G(W,y,S) = diag(g_i'(\sum_{j=1}^{N} w_{ij}y_j + s_i))$$
$$D\Phi^h(y) = I - H(W,y,S)W \quad \text{avec} \quad H(W,y,S) = diag(h_i'(\sum_{j=1}^{N} w_{ij}y_j + s_i))$$

Pour la fonction $f$ on a

$$(\Phi^f(y))_i = y_i - f_i(\sum_{j=1}^{N}(w_{ij}y_j - d_i)^2 + (s_i-d_i)^2) \quad i = 1,...,N$$

et donc

$$
\begin{aligned}
Dj(\Phi^f(y))_i &= \delta_{ij} - \frac{\partial}{\partial y_j}\left( f_i(\sum_{j=1}^{N}(w_{ij}y_j - d_i)^2 + (s_i - d_i)^2)\right) \quad i,j = 1,...,N \\
&= \delta_{ij} - 2w_{ij}(w_{ij}y_j - d_i) f_i'(\sum_{j=1}^{N}(w_{ij}y_j - d_i)^2 + (s_i - d_i)^2)) \quad i,j = 1,...,N
\end{aligned}
$$

On a donc

$$
\begin{aligned}
& D\Phi^f(y) = I - F(W,y,S)M(W) \\
& \text{avec } F(W,y,S) = diag(f_i'(\sum_{j=1}^{N}(w_{ij}y_j - d_i)^2 + (s_i - d_i)^2))) \\
& \text{et } M(W) = \{2w_{ij}(w_{ij}y_j - d_i)\}_{i,j=1,\ldots,N}
\end{aligned}
$$

Pour appliquer le théorème (5) on définit les trois homotopies $\Phi_t^g$, $\Phi_t^h$ et $\Phi_t^f$, correspondant respectivement aux fonctions $g$, $h$ et $f$

$$
\begin{array}{rccl}
\Phi_t^\sigma : & [0,1]^N & \to & \mathbb{R}^N \quad 0 \le t \le 1 \\
& y & \mapsto & \Phi_t^\sigma(y) = y - \sigma(tW, y, S)
\end{array}
$$

Les pas associés aux homotopies $\Phi_t^\sigma$ sont les applications

$$
\begin{array}{rccl}
\Phi_t^\sigma : & [0,1] & \to & [0,1]^N \\
& t & \mapsto & \zeta_t^\sigma
\end{array}
$$

vérifiant

$$
\left\{
\begin{array}{l}
\Phi_t^g(\zeta_t^g) = 0 \\
D\Phi_t^g(y) = I - G(tW,y,S)tW \text{ est bien défini}
\end{array}
\right.
$$

$$
\left\{
\begin{array}{l}
\Phi_t^h(\zeta_t^h) = 0 \\
D\Phi_t^h(y) = I - H(tW,y,S)tW \text{ est bien défini}
\end{array}
\right.
$$

$$
\left\{
\begin{array}{l}
\Phi_t^f(\zeta_t^f) = 0 \\
D\Phi_t^f(y) = I - F(tW,y,S)M(tW) \text{ est bien défini}
\end{array}
\right.
$$

### 5.3.1 Résultat principal

Soit $F^\sigma = \{\Phi_t^\sigma, \zeta_t^\sigma\}$ avec $\sigma = \{f, g, h\}$ un couple homotopie-pas tel qu'il a été défini ci-dessus.

Si

$$\|I - D\Phi_t^\sigma(\zeta_t^\sigma)\| \le c_\sigma < 1$$

alors il existe une constante $k_\sigma$ telle que $k_\sigma$ pas de Newton sont suffisants pour suivre $F^\sigma$. En particulier, on a (voir les notations au paragraphe suivant)

$$k_g \geq \frac{a}{2}\frac{N^{1/2}\bar{\theta}\bar{w}_1\|\theta\|_\infty^2\|W\|_2\|W\|_1}{\bar{\alpha}(1-c_g)^2}$$

$$k_h \geq \frac{3}{\pi}\frac{N^{1/2}\bar{\theta}\bar{w}_1\|\theta\|_\infty^2\|W\|_2\|W\|_1}{\bar{\alpha}(1-c_h)^2}$$

$$k_f \geq \frac{\|\theta\|_\infty^{1/2}\|W\|_2}{\sqrt{\log_{10}\left|1-\frac{\bar{\alpha}(1-c_f)^2}{4N\bar{\theta}\bar{w}_2\|\theta\|_\infty(\|W\|_4+\|d\|_\infty\|W\|_2)}\right|}}$$

De plus, le nombre d'itérations de Newton nécessaire pour avoir une approximation à $\varepsilon$ près du zéro $\zeta_1^\sigma$ de la fonction $\Phi^\sigma$ est alors

$$k_\sigma + (\log|\log\frac{\tau(\alpha_\sigma)}{\varepsilon\gamma_\sigma}|)$$

avec $\alpha_\sigma = \alpha(\Phi_1^\sigma, \zeta_1^\sigma)$ et $\gamma_\sigma = \gamma(\Phi_1^\sigma, \zeta_1^\sigma)$.

### Quelques notations

Pour la démonstration, on utilise les notations suivantes:

$$\|W\|_1 = \left(\sum_{i=1}^N\left(\sum_{j=1}^N|w_{ij}|\right)^2\right)^{1/2}$$

$$\|W\|_2 = \left(\max_{1\leq i\leq N}\sum_{j=1}^N|w_{ij}|^2\right)^{1/2}, \quad \bar{w}_1 = \max(1, \|W\|_2)$$

$$\|W\|_4 = \left(\max_{1\leq i\leq N}\sum_{j=1}^N|w_{ij}|^4\right)^{1/2}, \quad \|d\|_\infty = \max_{1\leq i\leq N}|d_i|, \quad \bar{w}_2 = \max(1, \|W\|_4 + \|d\|_\infty\|W\|_2)$$

$$\|\theta\|_\infty = \max_{1\leq i\leq N}|\theta_i|, \quad \bar{\theta} = \max(1, \|\theta\|_\infty)$$

Pour démontrer ce résultat, on a besoin des deux résultats suivants qui seront démontrés en appendice.

## 5.3.2 Résultat 1

On a quelque soit $k > 1$

$$\begin{aligned}
&\sup_{k>1}\left(\tfrac{\|D^k\Phi^g(y)\|}{k!}\right)^{\frac{1}{k-1}} \leq 2aN^{1/2}\bar{\theta}\bar{w}_1\|\theta\|_\infty\|W\|_2\\
&\sup_{k>1}\left(\tfrac{\|D^k\Phi^h(y)\|}{k!}\right)^{\frac{1}{k-1}} \leq \tfrac{3}{2}N^{1/2}\bar{\theta}\bar{w}_1\|\theta\|_\infty\|W\|_2\\
&\sup_{k>1}\left(\tfrac{\|D^k\Phi^f(y)\|}{k!}\right)^{\frac{1}{k-1}} \leq 4N^{1/2}\|\theta\|_\infty\bar{\theta}(\|W\|_4+\|d\|_\infty\|W\|_2)\bar{w}_2
\end{aligned} \tag{5.6}$$

### 5.3.3 Résultat 2

On a

$$\begin{aligned}
\|\Phi^g_{t'}(\zeta_t)\| &\leq \tfrac{1}{4}|t-t'|\|\theta\|_\infty\|W\|_1 \\
\|\Phi^h_{t'}(\zeta_t)\| &\leq \tfrac{2}{\pi}|t-t'|\|\theta\|_\infty\|W\|_1 \\
\|\Phi^f_{t'}(\zeta_t)\| &\leq N^{1/2}(1-e^{-|t-t'|^2\|\theta\|_\infty\|W\|_2^2})
\end{aligned} \tag{5.7}$$

### 5.3.4 Démonstration du résultat principal

En $t_0 = 0$ on a une solution triviale au problème $\Phi^\sigma(y) = 0$, c'est-à-dire $y = \{\sigma_i(s_i)\}_{i=1}^N$. Pour appliquer le théorème 5, il faut ensuite majorer les quantités $\beta(\Phi^\sigma_{t'}, \zeta^\sigma_t)$ et $\gamma(\Phi^\sigma_{t'}, \zeta^\sigma_t)$.

On effectue tout d'abord le calcul pour $g$

- Si $\|G(tW, y, S)tW\| \leq c_g < 1$ alors $D\Phi^g_t(y) = I - G(tW, y, S)tW$ est inversible et on a

$$\|D\Phi^g_t(y)^{-1}\| \leq \frac{1}{1-\|G(tW,y,S)tW\|} \leq \frac{1}{1-c_g}$$

donc en utilisant (5.6) et (5.7) on obtient les majorations suivantes :

$$\begin{aligned}
\gamma(\Phi^g_{t'}, \zeta^g_t) &= \sup_{k>1}\left\|\frac{D\Phi^g_{t'}(\zeta^g_t)^{-1}D^k\Phi^g_{t'}(\zeta^g_t)}{k!}\right\|^{\frac{1}{k-1}} \\
&\leq \sup_{k>1}\|D\Phi^g_{t'}(\zeta^g_t)^{-1}\|^{\frac{1}{k-1}}\sup_{k>1}\left\|\frac{D^k\Phi^g_{t'}(\zeta^g_t)}{k!}\right\|^{\frac{1}{k-1}} \\
&\leq \sup_{k>1}\left(\frac{1}{1-c_g}\right)^{\frac{1}{k-1}}\left(2aN^{1/2}\bar{\theta}\|\theta\|_\infty\bar{w}_1\|W\|_2\right) \\
&\leq \frac{2aN^{1/2}\bar{\theta}\|\theta\|_\infty\bar{w}_1\|W\|_2}{1-c_g} = \bar{\gamma}_g
\end{aligned}$$

et

$$\beta(\Phi^g_{t'}, \zeta^g_t) \leq \frac{\frac{1}{4}|t-t'|\|\theta\|_\infty\|W\|_1}{1-c_g}$$

il suffit alors de chercher $|t'-t|$ tel que $\beta(\Phi^g_{t'}, \zeta^g_t) \leq \dfrac{\bar{\alpha}}{\bar{\gamma}_g}$. On trouve

$$|t'-t| \leq \frac{2\bar{\alpha}(1-c_g)^2}{aN^{1/2}\bar{\theta}\bar{w}_1\|\theta\|^2_\infty\|W\|_2\|W\|_1}$$

On effectue les mêmes calculs pour $h$ et $f$

• Si $\|H(tW,y,S)tW\| \leq c_h < 1$ alors $D\Phi_t^h(y) = I - H(tW,y,S)tW$ est inversible et on a

$$\|D\Phi_t^h(y)^{-1}\| \leq \frac{1}{1-\|H(tW,y,S)tW\|} \leq \frac{1}{1-c_h}$$

donc en utilisant (5.6) et (5.7) on obtient les majorations suivantes :

$$\gamma(\Phi_{t'}^h,\zeta_t^h) \quad \leq \frac{\frac{3}{2}N^{1/2}\bar{\theta}\|\theta\|_\infty\bar{w}_1\|W\|_2}{1-c_h} = \bar{\gamma}_h$$

et

$$\beta(\Phi_{t'}^h,\zeta_t^h) \quad \leq \frac{\frac{2}{\pi}|t-t'|\|\theta\|_\infty\|W\|_1}{1-c_h}$$

il suffit alors de chercher $|t'-t|$ tel que $\beta(\Phi_{t'}^h,\zeta_t^h) \leq \dfrac{\bar{\alpha}}{\bar{\gamma}_h}$. On trouve

$$|t'-t| \leq \frac{\pi}{3}\frac{\bar{\alpha}(1-c_h)^2}{N^{1/2}\bar{\theta}\bar{w}_1\|\theta\|_\infty^2\|W\|_2\|W\|_1}$$

• Si $\|F(tW,y,S)M(tW)\| \leq c_f < 1$ alors $D\Phi_t^f(y) = I - F(tW,y,S)M(tW)$ est inversible et on a

$$\|D\Phi_t^f(y)^{-1}\| \leq \frac{1}{1-\|F(tW,y,S)M(tW)\|} \leq \frac{1}{1-c_f}$$

donc en utilisant (5.6) et (5.7) on obtient les majorations suivantes :

$$\gamma(\Phi_{t'}^f,\zeta_t^f) \quad \leq \frac{4N^{1/2}\|\theta\|_\infty\bar{\theta}(\|W\|_4+\|d\|_\infty\|W\|_2)\bar{w}_2}{1-c_f} = \bar{\gamma}_f$$

et

$$\beta(\Phi_{t'}^f,\zeta_t^f) \quad \leq \frac{N^{1/2}(1-e^{-|t-t'|^2\|\theta\|_\infty\|W\|_2^2})}{1-c_f}$$

il suffit alors de chercher $|t'-t|$ tel que $\beta(\Phi_{t'}^f,\zeta_t^f) \leq \dfrac{\bar{\alpha}}{\bar{\gamma}_f}$. On trouve

$$|t'-t| \leq \frac{1}{\|\theta\|_\infty^{1/2}\|W\|_2}\sqrt{\log_{10}\left|1-\frac{\bar{\alpha}(1-c_f)^2}{4N\bar{\theta}\bar{w}_2\|\theta\|_\infty(\|W\|_4+\|d\|_\infty\|W\|_2)}\right|}$$

### 5.3.5 Stabilité du point fixe $\zeta_1^\sigma$

Pour que $\zeta_1^\sigma$ soit stable, il faut que la matrice jacobienne de la fonction $I - \Delta t\Phi_1^\sigma$ possède des valeurs propres dont le module soit plus petit que 1. Dans le résultat principal (5.3.1) nous avons fait l'hypothèse que $\|I - D\Phi_t^\sigma(\zeta_t^\sigma)\| < 1$; donc si $\lambda$ est valeur propre de $I - D\Phi_1^\sigma(\zeta_1^\sigma)$, alors $1-\Delta t+\Delta t\lambda$ est valeur propre de $I - \Delta t\Phi_1^\sigma(\zeta_1^\sigma)$ dont le module est strictement inférieur à 1 quelque soit $\Delta t$, ce qui montre que $\zeta_1^\sigma$ est stable.

## 5.4 Appendice

Dans ce paragraphe, nous démontrons les résultats 1 et 2 des paragraphes 5.3.2 et 5.3.3. Pour démontrer 5.3.2, nous aurons besoin d'établir les deux résultats suivants :

### 5.4.1 Résultat principal sur la fonction $g$

On a

$$\sup_{k>1}\left(\frac{\sum_{i=1}^{N}|g_i^{(k)}(\sum_{j=1}^{N}w_{ij}y_j+s_i)|^2}{(k!)^2}\right)^{\frac{1}{2(k-1)}} \leq 2aN^{1/2}\|\theta\|_\infty\bar{\theta}, \quad a\simeq 0.09017 \tag{5.8}$$

### 5.4.2 Résultat principal sur la fonction $h$

On a

$$\sup_{k>1}\left(\frac{\sum_{i=1}^{N}|h_i^{(k)}(\sum_{j=1}^{N}w_{ij}y_j+s_i)|^2}{(k!)^2}\right)^{\frac{1}{2(k-1)}} \leq \frac{3}{2}N^{1/2}\bar{\theta}\|\theta\|_\infty \tag{5.9}$$

### 5.4.3 Démonstration du résultat principal sur la fonction $g$

Avant de démontrer ce résultat, on établit tout d'abord trois résultats intermédiaires.

#### 5.4.3.1 Résultat A

Soit $g(x)=\dfrac{1}{1+e^{-\theta x}}$ alors la dérivée k-ième s'écrit :

$$g^{(k)}(x)=\theta^k g(x)\sum_{i=0}^{k-1}\gamma_{i,k}(1-g(x))^{k-i} \quad \forall k\geq 1 \tag{5.10}$$

avec quelque soit $k\geq 1$

$$\begin{cases} \gamma_{0,k}(k+1)=\gamma_{0,k+1} \\ (\gamma_{i,k}-\gamma_{i-1,k})(k-i+1)=\gamma_{i,k+1} \quad i=1,...,k-1 \\ \gamma_{k-1,k}=-\gamma_{k,k+1} \end{cases} \tag{5.11}$$

**Démonstration**

On adopte la notation $g^{(k)} \equiv g^{(k)}(x)$

On démontre le résultat par récurrence :

On a :

$$g' = \theta g(1-g)$$

on suppose

$$g^{(k)} = \theta^k g \sum_{i=0}^{k-1} \gamma_{i,k}(1-g)^{k-i} \quad \forall k \geq 1$$

alors pour $k \geq 0$

$$\begin{aligned}
(g^{(k)})' &= \theta^k[\theta g(1-g) \sum_{i=0}^{k-1} \gamma_{i,k}(1-g)^{k-i} - g \sum_{i=0}^{k-1}(k-i)\gamma_{i,k}(1-g)^{k-i-1}\theta g(1-g)] \\
&= \theta^{k+1} g[\sum_{i=0}^{k-1} \gamma_{i,k}(1-g)^{k-i+1} - \sum_{i=0}^{k-1}(k-i)\gamma_{i,k}(1-g)^{k-i} g] \\
&= \theta^{k+1} g[\sum_{i=0}^{k-1} \gamma_{i,k}(1-g)^{k-i+1} + \sum_{i=0}^{k-1}(k-i)\gamma_{i,k}(1-g)^{k-i}(1-g-1)] \\
&= \theta^{k+1} g[\sum_{i=0}^{k-1} \gamma_{i,k}(1-g)^{k-i+1} + \sum_{i=0}^{k-1}(k-i)\gamma_{i,k}(1-g)^{k-i+1} - \sum_{i=0}^{k-1}(k-i)\gamma_{i,k}(1-g)^{k-i}] \\
&= \theta^{k+1} g[\sum_{i=0}^{k-1} (\gamma_{i,k} + (k-i)\gamma_{i,k})(1-g)^{k-i+1} - \sum_{i=0}^{k-1}(k-i)\gamma_{i,k}(1-g)^{k-i}] \\
&= \theta^{k+1} g[\sum_{i=0}^{k-1} ((k-i+1)\gamma_{i,k})(1-g)^{k-i+1} - \sum_{i=1}^{k}(k-i+1)\gamma_{i-1,k}(1-g)^{k-i+1}] \\
&= \theta^{k+1} g[(k+1)\gamma_{0,k}(1-g)^{k+1} + \sum_{i=1}^{k-1} ((k-i+1)(\gamma_{i,k} - \gamma_{i-1,k})(1-g)^{k-i+1} \\
&\qquad - \gamma_{k-1,k}(1-g)]
\end{aligned}$$

donc en posant (5.11) on obtient

$$g^{(k+1)} = \theta^{k+1} g \sum_{i=0}^{k} \gamma_{i,k+1}(1-g)^{k-i+1} \quad \forall k \geq 1$$

ce qui démontre le résultat.

De plus, on déduit de (5.11) que

$$\begin{cases} \gamma_{0,k} = k! & \forall k \geq 1 \\ \gamma_{k-1,k} = (-1)^{k-1} & \forall k \geq 1 \end{cases} \tag{5.12}$$

#### 5.4.3.2 Résultat B

On considère les $\gamma_{i,k}$ tels qu'ils ont été définis dans (5.11). Alors on a :

$$\left( \frac{\sum_{i=0}^{k-1} |\gamma_{i,k}|^2}{(k!)^2} \right)^{\frac{1}{2(k-1)}} < 2 \quad \forall k > 1 \tag{5.13}$$

**Démonstration**

On pose :

$$S_k = \frac{\sum\limits_{i=0}^{k-1} |\gamma_{i,k}|^2}{(k!)^2} \quad \forall k \geq 1$$

On sait que quelque soit $k \geq 1$ et $i = 1, ..., k-1$

$$\begin{aligned}
& \gamma_{i,k+1} = (\gamma_{i,k} - \gamma_{i-1,k})(k-i+1) \\
\Longrightarrow \quad & \gamma_{i,k+1} \leq (|\gamma_{i,k}| + |\gamma_{i-1,k}|)(k-i+1) \\
\Longrightarrow \quad & \tfrac{1}{2}|\gamma_{i,k+1}|^2 \leq (|\gamma_{i,k}|^2 + |\gamma_{i-1,k}|^2)(k-i+1)^2
\end{aligned}$$

donc, en sommant de $i = 1$ à $k-1$ on obtient quelque soit $k \geq 1$

$$\begin{aligned}
& \tfrac{1}{2}\sum_{i=1}^{k-1} |\gamma_{i,k+1}|^2 \leq \sum_{i=1}^{k-1} |\gamma_{i,k}|^2 (k-i+1)^2 + \sum_{i=1}^{k-1} |\gamma_{i-1,k}|^2 (k-i+1)^2 \\
\Longleftrightarrow \quad & \tfrac{1}{2}(\sum_{i=0}^{k} |\gamma_{i,k+1}|^2 - (\gamma_{0,k+1})^2 - (\gamma_{k,k+1})^2) \leq \sum_{i=0}^{k-1} |\gamma_{i,k}|^2 (k-i+1)^2 \\
& \qquad -(\gamma_{0,k})^2(k+1)^2 + \sum_{i=1}^{k} |\gamma_{i-1,k}|^2 (k-i+1)^2 - (\gamma_{k-1,k})^2 (k+1)^2
\end{aligned}$$

donc en utilisant (5.12) on déduit que quelque soit $k \geq 1$

$$\begin{aligned}
& \tfrac{1}{2}(\sum_{i=0}^{k} |\gamma_{i,k+1}|^2 - ((k+1)!)^2 - 1) \leq \sum_{i=0}^{k-1} |\gamma_{i,k}|^2 (k-i+1)^2 - ((k+1)!)^2 \\
& \qquad + \sum_{i=1}^{k} |\gamma_{i-1,k}|^2 (k-i+1)^2 - 1 \\
\Longleftrightarrow \quad & \tfrac{1}{2}\sum_{i=0}^{k} |\gamma_{i,k+1}|^2 \leq \sum_{i=0}^{k-1} |\gamma_{i,k}|^2 (k-i+1)^2 + \sum_{i=1}^{k} |\gamma_{i-1,k}|^2 (k-i+1)^2 - \tfrac{1}{2}((k+1)!)^2 - \tfrac{1}{2} \\
\Longleftrightarrow \quad & \tfrac{1}{2}\sum_{i=0}^{k} |\gamma_{i,k+1}|^2 \leq \sum_{i=0}^{k-1} |\gamma_{i,k}|^2 (k-i+1)^2 + \sum_{i=0}^{k-1} |\gamma_{i,k}|^2 (k-i)^2 - \tfrac{1}{2}((k+1)!)^2 - \tfrac{1}{2} \\
\Longleftrightarrow \quad & \tfrac{1}{2}\sum_{i=0}^{k} |\gamma_{i,k+1}|^2 \leq 2(k+1)^2 \sum_{i=0}^{k-1} |\gamma_{i,k}|^2 - \tfrac{1}{2}((k+1)!)^2 - \tfrac{1}{2} \\
\Longleftrightarrow \quad & \tfrac{1}{2}\frac{\sum\limits_{i=0}^{k} |\gamma_{i,k+1}|^2}{((k+1)!)^2} \leq \frac{2\sum\limits_{i=0}^{k-1} |\gamma_{i,k}|^2}{(k!)^2} - \frac{1}{2((k+1)!)^2} - \frac{1}{2} \\
\Longleftrightarrow \quad & S_{k+1} < 4S_k - (\frac{1}{((k+1)!)^2} + 1)
\end{aligned}$$

On a donc

$$S_{k+1} < 4S_k \quad \forall k \geq 1$$

Par suite

$$S_{k+1} < 4S_k < 4^2 S_{k-1} < ... < 4^k S_1$$

et comme $S_1 = 1$, on obtient finalement

$$S_k < 4^{k-1} \quad \forall k \geq 1$$

donc

$$S_k^{\frac{1}{2(k-1)}} = \left( \frac{\sum_{i=0}^{k-1} |\gamma_{i,k}|^2}{(k!)^2} \right)^{\frac{1}{2(k-1)}} < 2 \quad \forall k > 1$$

#### 5.4.3.3 Résultat C

On a

$$g^2(x) \sum_{j=0}^{k-1} (1 - g(x))^{2(k-j)} < a = \frac{(1+\sqrt{5})^2}{(3+\sqrt{5})^2(2+\sqrt{5})} \simeq 0.09017 \tag{5.14}$$

**Démonstration**

On adopte la notation $g \equiv g(x)$

$$\begin{aligned} g^2 \sum_{j=0}^{k-1} (1-g)^{2(k-j)} &= g^2 \sum_{j=1}^{k} (1-g)^{2j} \\ &= g^2 \left( \sum_{j=0}^{k} (1-g)^{2j} - 1 \right) \\ &= g^2 \left( \frac{1-(1-g)^{2(k+1)}}{1-(1-g)^2} - 1 \right) \\ &= g(1-g)^2 \left( \frac{1-(1-g)^{2k}}{2-g} \right) \end{aligned}$$

donc

$$g^2 \sum_{j=0}^{k-1} (1-g)^{2(k-j)} < \frac{g(1-g)^2}{2-g} \quad \text{car} \quad 0 < g(x) < 1 \quad \forall x \in \mathbb{R}$$

L'étude de la fonction $x \mapsto \dfrac{g(x)(1-g(x))^2}{2-g(x)}$ permet ensuite de montrer que

$$\frac{g(x)(1-g(x))^2}{2-g(x)} \leq a = \frac{(1+\sqrt{5})^2}{(3+\sqrt{5})^2(2+\sqrt{5})} \simeq 0.09017 \quad \forall x \in \mathbb{R}$$

Les trois résultats établis aux paragraphes 5.4.3.1, 5.4.3.2 et 5.4.3.3 vont nous permettre de démontrer le résultats principal 5.4.1.

**Démonstration du résultat principal 5.4.1**

On utilise la notation $g_i \equiv g_i(\sum_{j=1}^{N} w_{ij} y_j + s_i)$

D'après (5.10) et (5.14) on a

$$\begin{aligned} |g_i^{(k)}| &\leq |\theta_i|^k |g_i| \left(\sum_{j=0}^{k-1} |\gamma_{j,k}|^2\right)^{1/2} \left(\sum_{j=0}^{k-1} (1-g_i)^{2(k-j)}\right)^{1/2} \\ &\leq |\theta_i|^k a^{1/2} \left(\sum_{j=0}^{k-1} |\gamma_{j,k}|^2\right)^{1/2} \end{aligned}$$

donc

$$\begin{aligned} \left(\sum_{i=1}^{N} |g_i^{(k)}|^2\right)^{1/2} &\leq a \left(\sum_{i=1}^{N} |\theta_i|^{2k}\right)^{1/2} \left(\sum_{j=0}^{k-1} |\gamma_{j,k}|^2\right)^{1/2} \\ &\leq a N^{1/2} \max_{1\leq i\leq N} |\theta_i|^k \left(\sum_{j=0}^{k-1} |\gamma_{j,k}|^2\right)^{1/2} \\ &\leq a N^{1/2} \|\theta\|_\infty^k \left(\sum_{j=0}^{k-1} |\gamma_{j,k}|^2\right)^{1/2} \quad \text{avec } \|\theta\|_\infty = \max_{1\leq i\leq N} |\theta_i| \end{aligned}$$

Par suite, en utilisant (5.13) on termine la démonstration.

### 5.4.4 Démonstration du résultat principal sur la fonction $h$

Avant de démontrer ce résultat, on établit tout d'abord deux résultats intermédiaires.

#### 5.4.4.1 Résultat A

Soit $h(x) = \frac{2}{\pi} \arctan(\theta x)$ alors la dérivée k-ième de $h$ s'écrit :

$$\begin{aligned} h^{(k)}(x) &= \frac{2}{\pi} \frac{\theta^k}{(1+\theta^2 x^2)^k (\theta x)^{k(2)}} \sum_{i=0}^{\nu(k)} \gamma_{i,k} (\theta x)^{2i+1} \forall k \geq 1 \\ \text{avec} \quad \nu(k) &= E[\tfrac{k+1}{2}] - 1 \end{aligned} \tag{5.15}$$

avec quelque soit $p \geq 1$

$$\begin{cases} \gamma_{0,2p+1} = \gamma_{0,2p} \\ \gamma_{i,2p+1} = (2i+1)\gamma_{i,2p} + (2i-1-4p)\gamma_{i-1,2p} \quad i = 1, ..., p-1 \\ \gamma_{p,2p+1} = -(2p+1)\gamma_{p-1,2p} \end{cases} \tag{5.16}$$

et quelque soit $p \geq 2$

$$\begin{cases} \gamma_{i,2p} = 2((i+1)\gamma_{i+1,2p-1} + (i+1-2p)\gamma_{i,2p-1}) \quad i = 0, ..., p-2 \\ \gamma_{p-1,2p} = -2p\gamma_{p-1,2p-1} \end{cases} \tag{5.17}$$

**Démonstration**

On démontre le résultat par récurrence

On établit tout d'abord le résultat pour $h^{(2p+1)}$ quelque soit $p \geq 1$
On a :

$$h'(x) = \frac{2}{\pi}\frac{\theta}{1+\theta^2x^2}$$

on suppose

$$h^{(2p)}(x) = \frac{2}{\pi}\frac{\theta^{2p}}{(1+\theta^2x^2)^{2p}}\sum_{i=0}^{p-1}\gamma_{i,2p}(\theta x)^{2i+1} \quad \forall p \geq 1$$

alors pour $p \geq 1$

$$\begin{aligned}
(h^{(2p)})'(x) &= \frac{2}{\pi}\frac{\theta^{2p}}{(1+\theta^2x^2)^{4p}}[(1+\theta^2x^2)^{2p}\sum_{i=0}^{p-1}\gamma_{i,2p}\theta^{2i+1}x^{2i}(2i+1) \\
&\qquad -4p\theta^2x(1+\theta^2x^2)^{2p-1}\sum_{i=0}^{p-1}\gamma_{i,2p}(\theta x)^{2i+1}] \\
&= \frac{2}{\pi}\frac{\theta^{2p}}{(1+\theta^2x^2)^{2p+1}}[(1+\theta^2x^2)\sum_{i=0}^{p-1}\gamma_{i,2p}\theta^{2i+1}x^{2i}(2i+1) - 4p\theta^2x\sum_{i=0}^{p-1}\gamma_{i,2p}(\theta x)^{2i+1}] \\
&= \frac{2}{\pi}\frac{\theta^{2p+1}}{(1+\theta^2x^2)^{2p+1}}[\sum_{i=0}^{p-1}\gamma_{i,2p}(2i+1)(\theta x)^{2i} + \sum_{i=0}^{p-1}(2i+1-4p)\gamma_{i,2p}(\theta x)^{2i+2} \\
&\qquad + \sum_{i=1}^{p-1}(2i-1-4p)\gamma_{i-1,2p}(\theta x)^{2i}] \\
&= \frac{2}{\pi}\frac{\theta^{2p+1}}{(1+\theta^2x^2)^{2p+1}}[\gamma_{0,2p} + \sum_{i=1}^{p-1}((2i+1)\gamma_{i,2p}(\theta x)^{2i+2} + (2i-1-4p)\gamma_{i-1,2p})(\theta x)^{2i} \\
&\qquad -(2p+1)\gamma_{p-1,2p}(\theta x)^{2p}]
\end{aligned}$$

donc en posant (5.15) on obtient

$$h^{(2p+1)}(x) = \frac{2}{\pi}\frac{\theta^{2p+1}}{(1+\theta^2x^2)^{2p+1}}\sum_{i=0}^{p}\gamma_{i,2p+1}(\theta x)^{2i} \quad \forall k \geq 1$$

ce qui démontre le résultat pour le cas impair.

On établit ensuite le résultat pour $h^{(2p)}$ quelque soit $p \geq 2$
On a :

$$h^{(2)}(x) = \frac{2}{\pi}\frac{-2\theta^3x}{(1+\theta^2x^2)^2}$$

on suppose

$$h^{(2p-1)}(x) = \frac{2}{\pi}\frac{\theta^{2p-1}}{(1+\theta^2x^2)^{2p-1}}\sum_{i=0}^{p-1}\gamma_{i,2p}(\theta x)^{2i} \quad \forall p \geq 1$$

alors pour $p \geq 1$

$$
\begin{aligned}
(h^{(2p-1)})' &= \frac{2}{\pi}\frac{\theta^{2p}}{(1+\theta^2x^2)^{2p}}[(1+\theta^2x^2)\sum_{i=1}^{p-1}2i\gamma_{i,2p-1}(\theta x)^{2i-1}-2\theta x(2p-1)\sum_{i=0}^{p-1}\gamma_{i,2p-1}(\theta x)^{2i}] \\
&= \frac{2}{\pi}\frac{\theta^{2p}}{(1+\theta^2x^2)^{2p}}[\sum_{i=1}^{p-1}2i\gamma_{i,2p-1}(\theta x)^{2i-1}+\sum_{i=1}^{p-1}2i\gamma_{i,2p-1}(\theta x)^{2i+1} \\
&\qquad -2(2p-1)\sum_{i=0}^{p-1}\gamma_{i,2p-1}(\theta x)^{2i+1}] \\
&= \frac{2}{\pi}\frac{\theta^{2p}}{(1+\theta^2x^2)^{2p}}[\sum_{i=0}^{p-2}(2i+2)\gamma_{i+1,2p-1}(\theta x)^{2i+1}+\sum_{i=1}^{p-1}2i\gamma_{i,2p-1}(\theta x)^{2i+1} \\
&\qquad -2(2p-1)\sum_{i=0}^{p-1}\gamma_{i,2p-1}(\theta x)^{2i+1}] \\
&= \frac{2}{\pi}\frac{\theta^{2p}}{(1+\theta^2x^2)^{2p}}[(2\gamma_{1,2p-1}-2(2p-1)\gamma_{0,2p-1})\theta x \\
&\qquad +\sum_{i=1}^{p-2}((2i+2)\gamma_{i+1,2p-1}+(2i-4p+2)\gamma_{i,2p-1})(\theta x)^{2i+1} \\
&\qquad +(2(p-1)\gamma_{p-1,2p-1}-2(2p-1)\gamma_{p-1,2p-1})(\theta x)^{2p-1}] \\
&= \frac{2}{\pi}\frac{\theta^{2p}}{(1+\theta^2x^2)^{2p}}[(2\gamma_{1,2p-1}-2(2p-1)\gamma_{0,2p-1})\theta x \\
&\quad +\sum_{i=1}^{p-2}((2i+2)\gamma_{i+1,2p-1}+(2i-4p+2)\gamma_{i,2p-1})(\theta x)^{2i+1}-2p\gamma_{p-1,2p-1})(\theta x)^{2p-1}]
\end{aligned}
$$

donc en posant (5.16) on obtient

$$h^{(2p)}(x)=\frac{2}{\pi}\frac{\theta^{2p}}{(1+\theta^2x^2)^{2p}}\sum_{i=0}^{p-1}\gamma_{i,2p}(\theta x)^{2i+1}\quad \forall p\geq 1$$

ce qui démontre le résultat pour le cas pair.
De plus, comme $\gamma_{p,2p+1}=-(2p+1)\gamma_{p-1,2p}$ et $\gamma_{p-1,2p}=-2p\gamma_{p-1,2p-1}$ on déduit que

$$\left\{\begin{array}{l}\gamma_{p,2p+1}=(2p+1)! \\ \gamma_{p-1,2p}=-(2p)!\end{array}\right. \tag{5.18}$$

#### 5.4.4.2 Résultat B

On considère les $\gamma_{i,k}$ tels qu'ils ont été définis dans (5.16) et (5.17). Alors on a :

$$\left(\frac{\sum_{i=0}^{\nu(k)}|\gamma_{i,k}|}{k!}\right)^{\frac{1}{k-1}}<3\quad \forall k>1 \tag{5.19}$$

**Démonstration**

On pose :

$$S_k = \frac{\sum_{i=0}^{\nu(k)} |\gamma_{i,k}|}{k!} \quad \forall k \geq 1$$

On sait que quelque soit $p \geq 1$ et $i = 1, ..., p-1$

$$\begin{aligned} & \gamma_{i,2p+1} = (2i+1)\gamma_{i,2p} + (2i-1-4p)\gamma_{i-1,2p} \\ \Longrightarrow \quad & \gamma_{i,2p+1} \leq (2i+1)|\gamma_{i,2p}| + |2i-1-4p||\gamma_{i-1,2p}| \end{aligned}$$

donc, en sommant de $i = 1$ à $p-1$ on obtient quelque soit $p \geq 1$

$$\begin{aligned} & \sum_{i=1}^{p-1} |\gamma_{i,2p+1}| \leq \sum_{i=1}^{p-1} (2j+1)|\gamma_{i,2p}| + \sum_{i=1}^{p-1} |2i-1-4p||\gamma_{i-1,2p}| \\ \Longleftrightarrow \quad & \sum_{i=0}^{p} |\gamma_{i,2p+1}| - |\gamma_{0,2p+1}| - |\gamma_{p,2p+1}| \leq \sum_{i=0}^{p-1} (2j+1)|\gamma_{i,2p}| - |\gamma_{0,2p}| \\ & \qquad + \sum_{i=1}^{p} |2i-1-4p||\gamma_{i-1,2p}| - |2p+1||\gamma_{p-1,2p}| \end{aligned}$$

donc, en utilisant (5.18) et le fait que $\gamma_{0,2p+1} = \gamma_{0,2p}$ on a

$$\begin{aligned} & \sum_{i=0}^{p} |\gamma_{i,2p+1}| \leq (2p-1)\sum_{i=1}^{p-1} |\gamma_{i,2p}| + \sum_{i=0}^{p-1} |2i+1-4p||\gamma_{i,2p}| \\ \Longleftrightarrow \quad & \sum_{i=0}^{p} |\gamma_{i,2p+1}| \leq (2p-1)\sum_{i=0}^{p-1} |\gamma_{i,2p}| + (4p-1)\sum_{i=0}^{p-1} |\gamma_{i,2p}| \\ \Longleftrightarrow \quad & \sum_{i=0}^{p} |\gamma_{i,2p+1}| \leq (6p-2)\sum_{i=0}^{p-1} |\gamma_{i,2p}| \\ \Longleftrightarrow \quad & S_{2p+1} = \frac{\sum_{i=0}^{p} |\gamma_{i,2p+1}|}{(2p+1)!} \leq \frac{6p-2}{(2p+1)!}\sum_{i=0}^{p-1} |\gamma_{i,2p}| \end{aligned}$$

de plus comme $S_{2p} = \frac{\sum_{i=0}^{p-1} |\gamma_{i,2p}|}{(2p)!}$ on obtient

$$S_{2p+1} \leq \frac{6p-2}{2p+1} S_{2p} \leq \frac{6p}{2p+1} S_{2p} \quad \forall p \geq 1 \tag{5.20}$$

Par ailleurs, on sait que quelque soit $p \geq 1$ et $i = 1, ..., p-2$

$$\begin{aligned} & \gamma_{i,2p} = 2(i+1)\gamma_{i+1,2p-1} + 2(i+1-2p)\gamma_{i,2p-1} \\ \Longrightarrow \quad & |\gamma_{i,2p}| \leq 2(i+1)|\gamma_{i+1,2p-1}| + 2|2p-i-1||\gamma_{i,2p-1}| \end{aligned}$$

donc, en sommant de $i = 1$ à $p-2$ on obtient quelque soit $p \geq 2$

$$\begin{aligned} & \sum_{i=0}^{p-2} |\gamma_{i,2p}| \leq 2\sum_{i=0}^{p-2} (j+1)|\gamma_{i+1,2p-1}| + 2\sum_{i=0}^{p-2} |2p-i-1||\gamma_{i,2p-1}| \\ \Longleftrightarrow \quad & \sum_{i=0}^{p-2} |\gamma_{i,2p}| \leq 2\sum_{i=0}^{p-1} j|\gamma_{i,2p-1}| + 2\sum_{i=0}^{p-2} |2p-i-1||\gamma_{i,2p-1}| \\ \Longleftrightarrow \quad & \sum_{i=0}^{p-1} |\gamma_{i,2p}| - |\gamma_{p-1,2p}| \leq 2(p-1)\sum_{i=0}^{p-1} |\gamma_{i,2p-1}| + 2\sum_{i=0}^{p-1} |2p-i-1||\gamma_{i,2p-1}| - 2p|\gamma_{p-1,2p-1}| \end{aligned}$$

donc, comme $\gamma_{p-1,2p} = -2p\gamma_{p-1,2p-1}$ on a

$$\begin{aligned}
& \sum_{i=0}^{p-1} |\gamma_{i,2p}| \leq 2(p-1)\sum_{i=0}^{p-1} |\gamma_{i,2p-1}| + 2(2p-1)\sum_{i=0}^{p-1} |\gamma_{i,2p-1}| \\
\Longleftrightarrow \quad & \sum_{i=0}^{p-1} |\gamma_{i,2p}| \leq (6p-4)\sum_{i=0}^{p-1} |\gamma_{i,2p-1}| \\
\Longleftrightarrow \quad & S_{2p} = \frac{\sum_{i=0}^{p-1} |\gamma_{i,2p}|}{(2p)!} \leq (6p-4)\frac{\sum_{i=0}^{p-1} |\gamma_{i,2p-1}|}{(2p)!}
\end{aligned}$$

de plus comme $S_{2p-1} = \dfrac{\sum_{i=0}^{p-1} |\gamma_{i,2p-1}|}{(2p)!}$ on obtient

$$S_{2p} \leq \frac{6p-4}{2p} S_{2p-1} \leq \frac{6p-3}{2p} S_{2p-1} \quad \forall p \geq 2 \tag{5.21}$$

En utilisant (5.20) et (5.21) on obtient

$$\begin{aligned}
& S_k \leq \frac{3(k-1)}{k} S_{k-1} \\
\Longleftrightarrow \quad & S_k \leq \frac{3^2(k-1)(k-2)}{k(k-1)} S_{k-2} \\
\Longleftrightarrow \quad & S_k \leq 3^l \frac{(k-1)!}{(k-l-1)!}\frac{(k-l)!}{k!} S_{k-l} \\
\Longleftrightarrow \quad & S_k \leq 3^{k-2} \frac{(k-1)!}{1!}\frac{2!}{k!} S_2 \\
\Longleftrightarrow \quad & S_k \leq 4\frac{3^{k-2}}{k} = a_k \quad \text{car} \quad S_2 = |\gamma_{0,2}| = 2
\end{aligned}$$

En posant $b_k = a_k^{1/(k-1)}$, on montre que

$$\left(\frac{b_{k+1}}{b_k}\right)^{k(k-1)} = \frac{3}{4}\frac{k^k}{(k+1)^{k-1}} = c_k$$

Par suite, comme $c_k \geq 1$ et $c_2 = 1$, $b_k$ est croissante et $b_2 = 2$, donc $b_k < \lim_{k\to\infty} b_k = 3$.

Les deux résultats établit aux paragraphes 5.4.4.1 et 5.4.4.2 vont nous permettre de démontrer le résultats principal 5.4.2.

### Démonstration du résultats principal 5.4.2

On utilise la notation $h_i^{(k)} \equiv h_i^{(k)}(\sum_{j=1}^{N} w_{ij}y_j + s_i)$

D'après (5.15) on a quelque soit $k \geq 1$

$$\begin{aligned}
|h^{(k)}(x)| \quad & \leq \frac{2}{\pi}\frac{|\theta|^k|\theta x|^{2\nu(k)+1}}{(1+\theta^2x^2)^k|\theta x|^{k(2)}} \sum_{i=0}^{\nu(k)} |\gamma_{i,k}| \\
& \leq \frac{2}{\pi}\frac{|\theta|^k|\theta x|^{k-1}}{(1+\theta^2x^2)^k} \sum_{i=0}^{\nu(k)} |\gamma_{i,k}|
\end{aligned}$$

L'étude de la fonction $r_k(x) = \dfrac{2}{\pi}\dfrac{|\theta|^k|\theta x|^{k-1}}{(1+\theta^2x^2)^k}$ permet d'écrire

$$|r_k(x)| \leq \frac{2}{\pi}|\theta|^k l_k$$
$$\text{avec} \quad l_k = \left(\frac{k-1}{k+1}\right)^{\frac{k-1}{2}}\left(\frac{k+1}{2k}\right)^k$$

donc

$$\begin{aligned}
\left(\sum_{i=1}^{N}|h_i^k|^2\right)^{1/2} &\leq \frac{2}{\pi}\left(\sum_{i=1}^{N}(|\theta_i|^k l_k\sum_{j=0}^{\nu(k)}|\gamma_{j,k}|)^2\right)^{1/2} \\
&\leq \frac{2}{\pi}l_k\left(\sum_{i=1}^{N}(|\theta_i|^{2k}\right)^{1/2}\sum_{j=0}^{\nu(k)}|\gamma_{j,k}| \\
&\leq \frac{2}{\pi}l_k N^{1/2}\|\theta\|_\infty^k\sum_{j=0}^{\nu(k)}|\gamma_{j,k}|
\end{aligned}$$

et

$$\sup_{k>1}\left(\frac{\sum_{i=1}^{N}|h_i^{(k)}|^2}{(k!)^2}\right)^{\frac{1}{2(k-1)}} \leq N^{1/2}\bar{\theta}\|\theta\|_\infty \sup_{k>1}(l_k)^{\frac{1}{k-1}}\sup_{k>1}\left(\frac{\sum_{j=0}^{\nu(k)}|\gamma_{j,k}|}{k!}\right)^{\frac{1}{k-1}}$$

On termine la démonstration avec (5.19) et en démontrant que $\sup_{k>1}(l_k)^{\frac{1}{k-1}} < \frac{1}{2}$ (étude de la fonction $x \mapsto \sqrt{\dfrac{x-1}{x+1}}\left(\dfrac{x+1}{2x}\right)^{\frac{x}{x-1}}$ pour $x \geq 2$).

### 5.4.5 Démonstration du résultat 1 (5.3.2)

$$\|D^k\Phi(y)\| = \sup_{y_{(i)}\neq 0}\frac{\|D^k\Phi(y).y_{(1)}.y_{(2)}.....y_{(k)}\|}{\|y_{(1)}\|.\|y_{(2)}\|...\|y_{(k)}\|} \quad \forall k \geq 1 \tag{5.22}$$

On commence par démontrer la première partie du résultat .Dans cette partie, on peut remplacer $g$ par $h$ et, pour ne surcharger les notations, on utilisera la lettre $\Phi$ pour designer $\Phi^g$.

$$D_j(\Phi(y))_i = \delta_{ij} - w_{ij}g_i' \quad \text{avec} \quad g_i^{(k)} \equiv g_i^{(k)}(\sum_{j=1}^{N}w_{ij}y_j + s_i) \quad \forall k \geq 1$$

donc

$$D_{jk}^2(\Phi(y))_i = -w_{ij}w_{ik}g_i^{(2)}$$

si

$$D^{k-1}_{j_1\dots j_{k-1}}(\Phi(y))_i = -w_{ij_1}w_{ij_2}\dots w_{ij_{k-1}}g_i^{(k-1)}$$

alors

$$\begin{aligned} D^{k}_{j_1\dots j_k}(\Phi(y))_i &= \frac{\partial}{\partial y_{j_k}}\left(D^{k-1}_{j_1\dots j_{k-1}}(\Phi(y))_i\right) \\ &= -w_{ij_1}w_{ij_2}\dots w_{ij_{k-1}}w_{ij_k}g_i^{(k)} \end{aligned}$$

Calculons $(D^k\Phi(y).y_{(1)}.y_{(2)}.\dots.y_{(k)})_i$ pour $i = 1,\dots,N$ et quelque soit $k \geq 1$
On pose $\xi_{(l)} = W.y_{(l)}$ et donc $\xi_{(l),i} = \sum\limits_{j=1}^{N} w_{ij}y_{(l),j}$ alors on montre par récurrence que

$$(D^k\Phi(y).y_{(1)}.y_{(2)}.\dots.y_{(k)})_i = -\prod_{l=1}^{k}\xi_{(l),i}g_i^{(k)} \quad \forall k \geq 1 \tag{5.23}$$

En effet, supposons que

$$(D^{k-1}\Phi(y).y_{(1)}.y_{(2)}.\dots.y_{(k-1)})_i = -\prod_{l=1}^{k-1}\xi_{(l),i}g_i^{(k-1)} \quad \forall k > 1$$

alors quelque soit $k \geq 1$ on a

$$\begin{aligned} (D^k\Phi(y).y_{(1)}.y_{(2)}.\dots.y_{(k)})_i &= \sum_{j=1}^{N}(D^k\Phi(y).y_{(1)}.y_{(2)}.\dots.y_{(k-1)})_{ij}y_{(k),j} \\ &= \sum_{j=1}^{N}\frac{\partial}{\partial y_j}\left(D^{k-1}\Phi(y).y_{(1)}.y_{(2)}.\dots.y_{(k-1)}\right)_i y_{(k),j} \\ &= -\sum_{j=1}^{N}\frac{\partial}{\partial y_j}\left(\prod_{l=1}^{k-1}\xi_{(l),i}g_i^{(k-1)}\right)y_{(k),j} \\ &= -\sum_{j=1}^{N}\prod_{l=1}^{k-1}\xi_{(l),i}\frac{\partial g_i^{(k-1)}}{\partial y_j}y_{(k),j} \\ &= -\sum_{j=1}^{N}\prod_{l=1}^{k-1}\xi_{(l),i}w_{ij}g_i^{(k)}y_{(k),j} \\ &= -\prod_{l=1}^{k-1}\xi_{(l),i}\left(\sum_{j=1}^{N}w_{ij}y_{(k),j}\right)g_i^{(k)} \\ &= -\prod_{l=1}^{k}\xi_{(l),i}g_i^{(k)} \end{aligned}$$

Par ailleurs

$$\prod_{l=1}^{k}|\xi_{(l),i}|^2 \leq \left(\sum_{j=1}^{N}|w_{ij}|^2\right)^k \|y_{(1)}\|^2.\|y_{(2)}\|^2\dots\|y_{(k)}\|^2 \quad \forall k \geq 1 \tag{5.24}$$

On déduit donc de (5.22), (5.23) et (5.24) que quelque soit $k \geq 1$

$$\begin{aligned}
\|D^k\Phi(y)\| &\leq \left(\sum_{i=1}^{N}\left(\sum_{j=1}^{N}|w_{ij}|^2\right)^k |g_i^{(k)}|^2\right)^{1/2} \\
&\leq \left(\max_{1\leq i\leq N}\sum_{j=1}^{N}|w_{ij}|^2\right)^{k/2}\left(\sum_{i=1}^{N}|g_i^{(k)}|^2\right)^{1/2} \\
&\leq \|W\|_2^k\left(\sum_{i=1}^{N}|g_i^{(k)}|^2\right)^{1/2}
\end{aligned}$$

donc

$$\sup_{k>1}\left(\frac{\|D^k\Phi(y)\|}{k!}\right)^{\frac{1}{k-1}} \leq \sup_{k>1}\|W\|_2^{\frac{k}{k-1}}\sup_{k>1}\left(\frac{\sum_{i=1}^{N}|g_i^{(k)}|^2}{(k!)^2}\right)^{\frac{1}{2(k-1)}} \qquad \forall k>1$$

et donc en posant $\bar{w}_1 = \max(1, \|W\|_2)$ et en utilisant les résultats 5.8 et 5.9 on termine la première partie de la démonstration.

Démontrons maintenant la deuxième partie. Cette démonstration étant analogue à la première partie, on ne détaillera pas les calculs.

On pose $a_j = w_{ij}y_j - d_i$ et $f_i^{(k)} \equiv f_i^{(k)}(\sum_{j=1}^{N}|a_j|^2 + |s_i - d_i|^2)$

On a

$$D^k_{j_1\ldots j_k}(\Phi(y))_i = -2^k w_{ij_1}a_{j_1}w_{ij_2}a_{j_2}\ldots w_{ij_k}a_{j_k}f_i^{(k)}$$

Calculons $(D^k\Phi^f(y).y_{(1)}.y_{(2)}.....y_{(k)})_i$ pour $i = 1,\ldots,N$ et quelque soit $k \geq 1$

On pose

$$\begin{aligned}
\mu_{(l),i} &= \sum_{j=1}^{N} w_{ij}a_jy_{(l),j} \\
&= \sum_{j=1}^{N} w_{ij}(w_{ij}y_j - d_i)y_{(l),j}
\end{aligned}$$

alors on montre par récurrence que

$$(D^k\Phi^f(y).y_{(1)}.y_{(2)}.....y_{(k)})_i = -2^k\prod_{l=1}^{k}\mu_{(l),i}f_i^{(k)} \quad \forall k \geq 1 \tag{5.25}$$

Par ailleurs, quelque soit $i = 1,\ldots,N$

$$\begin{aligned}
|\mu_{(l),i}| &\leq \sum_{j=1}^{N}|w_{ij}|^2|y_j||y_{(l),j}| + |d_i|\sum_{j=1}^{N}|w_{ij}||y_{(l),j}| \\
&\leq \left(\sum_{j=1}^{N}|w_{ij}|^4|y_j|^2\right)^{1/2}\|y_{(l)}\| + |d_i|\left(\sum_{j=1}^{N}|w_{ij}|^2\right)^{1/2}\|y_{(l)}\| \\
&\leq (\|W\|_4 + \|d\|_\infty\|W\|_2)\|y_{(l)}\|
\end{aligned}$$

donc

$$\prod_{l=1}^{k} |\mu_{(l),i}|^2 \leq (\|W\|_4 + \|d\|_\infty \|W\|_2)^{2k} \|y_{(1)}\|^2 . \|y_{(2)}\|^2 ... \|y_{(k)}\|^2 \quad \forall k \geq 1 \tag{5.26}$$

On déduit donc de (5.22), (5.25) et (5.26) que quelque soit $k \geq 1$

$$\begin{aligned} \|D^k \Phi^f(y)\| &\leq 2^k (\|W\|_4 + \|d\|_\infty \|W\|_2)^k \left( \sum_{j=1}^{N} |f_i^{(k)}|^2 \right)^{1/2} \\ &\leq 2^k N^{1/2} (\|W\|_4 + \|d\|_\infty \|W\|_2)^k \|\theta\|_\infty^k \end{aligned}$$

donc quelque soit $k > 1$

$$\begin{aligned} \sup_{k>1} \left( \tfrac{\|D^k \Phi^f(y)\|}{k!} \right)^{\frac{1}{k-1}} &\leq 4N^{1/2} \|\theta\|_\infty \bar{\theta} (\|W\|_4 + \|d\|_\infty \|W\|_2) \bar{w}_2 \sup_{k>1} \left( \tfrac{1}{k!} \right)^{\frac{1}{k-1}} \\ &\leq 2N^{1/2} \|\theta\|_\infty \bar{\theta} (\|W\|_4 + \|d\|_\infty \|W\|_2) \bar{w}_2 \end{aligned}$$

### 5.4.6 Démonstration du résultat 2 (5.3.3)

On a

$$\Phi_{t'}^g(\zeta_t) = \zeta_t - g(t' W \zeta_t + S)$$

Ou encore

$$(\Phi_{t'}^g(\zeta_t))_i = \zeta_{t,i} - g_i(t' \sum_{j=1}^{N} w_{ij} \zeta_{t,j} + s_i) \quad i = 1, ..., N$$

On pose $\alpha_i = \sum\limits_{j=1}^{N} w_{ij} \zeta_{t,j}$

En utilisant la formule de Taylor avec reste intégral, on obtient

$$\begin{aligned} g_i(t' \alpha_i + s_i) &= g_i(t \alpha_i + s_i + (t' - t) \alpha_i) \\ &= g_i(t \alpha_i + s_i) + (t' - t) \alpha_i \int_0^1 g_i'(t \alpha_i + s_i + x(t' - t) \alpha_i) dx \end{aligned}$$

donc

$$\begin{aligned} (\Phi_{t'}^g(\zeta_t))_i &= \zeta_{t,i} - g_i(t \alpha_i + s_i) - (t' - t) \alpha_i \int_0^1 g_i'(t \alpha_i + s_i + x(t' - t) \alpha_i) dx \\ &= (\Phi_t(\zeta_t))_i - (t' - t) \alpha_i \int_0^1 g_i'(t \alpha_i + s_i + x(t' - t) \alpha_i) dx \\ &= -(t' - t) \alpha_i \int_0^1 g_i'(t \alpha_i + s_i + x(t' - t) \alpha_i) dx \end{aligned}$$

De plus, comme $|g_i'(x)| \leq \dfrac{\theta_i}{4} \quad \forall x \in \mathbb{R}$ on a

$$|(\Phi_{t'}^g(\zeta_t))_i| \leq \frac{1}{4} |t - t'| |\alpha_i| |\theta_i|$$

Par suite

$$\begin{aligned}
\|\Phi^g_{t'}(\zeta_t)\| &= \left(\sum_{i=1}^N |(\Phi^g_{t'}(\zeta_t))_i|^2\right)^{1/2} \\
&\leq \tfrac{1}{4}|t-t'| \left(\sum_{i=1}^N |\alpha_i|^2|\theta_i|^2\right)^{1/2} \\
&\leq \tfrac{1}{4}|t-t'| \left(\max_{1\leq i\leq N} |\theta_i|^2 \sum_{i=1}^N |\alpha_i|^2\right)^{1/2} \\
&\leq \tfrac{1}{4}|t-t'|\|\theta\|_\infty \left(\sum_{i=1}^N |\alpha_i|^2\right)^{1/2} \\
&\leq \tfrac{1}{4}|t-t'|\|\theta\|_\infty \left(\sum_{i=1}^N (\sum_{j=1}^N |w_{ij}|)^2\right)^{1/2} \quad \text{car } |\alpha_i| \leq \sum_{j=1}^N |w_{ij}| \\
&\leq \tfrac{1}{4}|t-t'|\|\theta\|_\infty \|W\|_1
\end{aligned}$$

Le calcul de $\|\Phi^h_{t'}(\zeta_t)\|$ est identique mis à part le fait que $|h'_i(x)| \leq \frac{2}{\pi}|\theta_i|$.

Calculons $\|\Phi^f_{t'}(\zeta_t)\|$.
On a

$$(\Phi^f_{t'}(\zeta_t))_i = \zeta_{t,i} - f_i(\sum_{j=1}^N |t'w_{ij}\zeta_{t,j} - d_i|^2 + |s_i - d_i|^2) \quad i = 1, ..., N$$

$$\begin{aligned}
\sum_{j=1}^N |t'w_{ij}\zeta_{t,j} - d_i|^2 + |s_i - d_i|^2 &= \sum_{j=1}^N |(t'-t+t)w_{ij}\zeta_{t,j} - d_i|^2 + |s_i - d_i|^2 \\
&\leq \sum_{j=1}^N |(t'-t)w_{ij}\zeta_{t,j}|^2 + \sum_{j=1}^N |tw_{ij}\zeta_{t,j} - d_i|^2 + |s_i - d_i|^2
\end{aligned}$$

donc, comme $f_i$ est décroissante

$$\begin{aligned}
\zeta_{t,i} - f_i(\sum_{j=1}^N |t'w_{ij}\zeta_{t,j} - d_i|^2 + |s_i - d_i|^2) \leq \zeta_{t,i} - f_i(&|t'-t|^2 \sum_{j=1}^N |w_{ij}\zeta_{t,j}|^2 + |s_i - d_i|^2 \\
&+ \sum_{j=1}^N |tw_{ij}\zeta_{t,j} - d_i|^2)
\end{aligned}$$

On pose $a_i = \sum_{j=1}^N |w_{ij}\zeta_{t,j}|^2$, $b_i = \sum_{j=1}^N |tw_{ij}\zeta_{t,j} - d_i|^2 + |s_i - d_i|^2$ et $h = |t' - t|$
On a alors quelque soit $i = 1, ..., N$

$$\begin{aligned}
(\Phi^f_{t'}(\zeta_t))_i &\leq \zeta_{t,i} - f_i(b_i + h^2 a_i) \\
&\leq \zeta_{t,i} - e^{-\theta_i(b_i + h^2 a_i)} \\
&\leq \zeta_{t,i} - e^{-\theta_i b_i} - e^{-\theta_i(b_i + h^2 a_i)} + e^{-\theta_i b_i} \\
&\leq e^{-\theta_i b_i}(1 - e^{-\theta_i h^2 a_i}) \\
&\leq 1 - e^{-\theta_i h^2 a_i}
\end{aligned}$$

Par ailleurs

$$a_i = \sum_{j=1}^{N} |w_{ij}\zeta_{t,j}|^2 \leq \sum_{j=1}^{N} |w_{ij}|^2 \leq \max_{1\leq i\leq N} \sum_{j=1}^{N} |w_{ij}|^2 \leq \|W\|_2^2$$

donc

$$\begin{array}{ll} & -\theta_i h^2 a_i \geq -h^2\|\theta\|_\infty \|W\|_2^2 \\ \Longleftrightarrow & e^{-\theta_i h^2 a_i} \geq e^{-h^2\|\theta\|_\infty \|W\|_2^2} \\ \Longleftrightarrow & \left(1 - e^{-\theta_i h^2 a_i}\right)^2 \leq \left(1 - e^{-h^2\|\theta\|_\infty \|W\|_2^2}\right)^2 \\ \Longleftrightarrow & \left(\sum_{j=1}^{N} (1 - e^{-\theta_i h^2 a_i})^2\right)^{1/2} \leq N^{1/2} |1 - e^{-h^2\|\theta\|_\infty \|W\|_2^2}| \end{array}$$

ce qui termine la démonstration.

# Conclusion générale

Les différents résultats obtenus sur les deux problèmes que nous avons traités dans cette thèse nous permettent de conclure à la fiabilité de l'approche par les réseaux de neurones, surtout en ce qui concerne le problème de robotique mobile. En effet, l'algorithme que nous avons utilisé nous a permis d'obtenir un ensemble de réseaux capables de se garer à partir d'un pourcentage élevé de positions de départ, tout en résistant bien à l'injection de bruit. L'étape suivante pour obtenir de meilleurs résultats consiste principalement à intégrer directement la longueur des trajectoires dans la performance d'un réseau afin d'obtenir un ensemble de trajectoires plus réalistes. De plus, il faudrait pouvoir démontrer qu'un réseau est fiable sur un certain domaine, c'est-à-dire qu'il n'existe pas un point de départ sur ce domaine tel que le réseau ne pourra pas garer le véhicule à partir de ce point. Un autre point important réside dans l'introduction d'obstacles.

Pour l'identification de la fonction isotherme en chromatographie, on a vu qu'une autre méthode qui utilise les fractions rationnelles et les réseaux de neurones donne de meilleurs résultats que la méthode qui utilise uniquement les réseaux. Cependant, plusieurs remarques doivent être faites : tout d'abord, ces résultats, bien qu'encourageants, exigent un temps de calcul beaucoup trop élevé, ce qui ne nous permet pas d'entrevoir réellement la limite de cette méthode; ensuite, il semble que cette méthode soit liée à la méthode qui utilise uniquement les réseaux de neurones (bien qu'elle soit plus performante) en ce sens que si la méthode par réseau est limitée alors l'autre méthode l'est aussi. Ainsi, l'amélioration des résultats pour les méthodes qui utilisent les réseaux de neurones passe par des temps de calcul beaucoup moins élevés et une meilleur connaissance théorique des réseaux, ce qui permettrait de diminuer significativement l'espace de recherche.

Les autres méthodes évolutionnaires présentées sur ce même problème ont aussi des temps de calcul assez élevés. Cependant, c'est actuellement la seule limitation de la méthode qui identifie les coefficients des isothermes pour laquelle on a par ailleurs obtenu les meilleurs résultats. Cependant, pour cette approche, il faut souligner qu'elle demande (comme pour la méthode de gradient) la connaissance a priori d'un modèle analytique. En ce qui concerne la méthode qui identifie les valeurs des points de contrôle d'une fraction rationnelle par stratégie d'évolution, il semble que la trop grande liberté ou le manque de “cohérence” qui existe entre ces valeurs limite l'efficacité de cette méthode.

Il serait par ailleurs intéressant d'ajouter certains critères dans la fonction de performance comme le critère de la distance des moments ou les moments d'ordre $p$ afin de constater si les résultats sont meilleurs et aussi si ils sont obtenus plus facilement.

Nous avons d'autre part constaté certaines différences entre les réseaux feed-forward et les réseaux récurrents. Tout d'abord, lorsque la fonction à identifier était moins “régulière” qu'un isotherme de Langmuir comme par exemple un isotherme de Moreau-Valentin de degré 4, nous avons obtenu en moyenne des meilleurs résultats en utilisant une population de réseaux de neurones récurrents: sur $2 \times 40$ évolutions, le nombre de réseaux ayant une très bonne performance est plus élevé lorsqu'on utilise les réseaux récurrents et le nombre de mauvais réseaux est beaucoup plus élevé lorsqu'on utilise les réseaux feed-forward. Cependant, les meilleurs réseaux obtenus dans les deux cas ont une fitness presque égale et l'emploi des réseaux récurrents est beaucoup plus coûteux en temps CPU.

# Bibliographie


[A93] Angeline P. J. (1993). Evolutionary Algorithms and Emergent Intelligence. PhD thesis, Ohio State University, Colombus.

[A94] Angeline P. J., Saunders G. M. and Pollack J. B. (1994). An Evolutionary Algorithm that Constructs Recurrent Neural Networks, *IEEE Transactions on Neural Networks.*

[Bä95] T. Bäck. *Evolutionary Algorithms in theory and practice.* New-York: Oxford University Press, 1995.

[FS95] A. Fadda et M. Schoenauer. Evolutionary chromatographic law identification by recurrent neural nets. *Proceedings on the 4th Annual Conference on Evolutionary Programming*, page 219-235. MIT Press, mars 1995.

[FOW66] Fogel L. J., Owens A. J. and Walsh M. J. (1966). *Artificial Intelligence through Simulated Evolution*, John Wiley, New York.

[Fog95] D. B. Fogel. *Evolutionary Computation. Toward a New Philosophy of Machine Intelligence.* IEEE Press, Piscataway, NJ, 1995.

[G59] Godunov S. K. (1959). A Finite Difference Method for Numerical Computation of Discontinuous Solutions of the Equations of Flow Dynamics, *Math USSR Sb.*, **47**, pp 271-290.

[Gol89] D. E. Goldber, *Genetic Algorithms in search, optimisation and machine learning*, Addison Wesley, 1989.

[HN89] R. Hecht-Nielsen. *Neuro-computing.* Addison Wesley, 1989.

[Hol75] J. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, Ann Harbor, 1975

[HSW89] Hornik, K., Stinchcombe, M., White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, **2**, 359-366.

[H90] Hornik, K. (1990). Approximation capabilities of multilayer feedforward networks. *Neural Networks*, **4**, 251-257.

[J90] James F. (1990). Sur la modélisation mathématiques des équilibres diphasiques et des colonnes de chromatographie. PhD thesis, École Polytechnique.

[JS94] James F. and Sepulveda M. (1994). Parameter identification for a hyperbolic system arising in chemical engineering, *Inverse Problems*, vol 10 n° 6 pp 1299-1314.

[JV91] James F., Valentin P. (1991). An analytical model of multicomponent isothermal adsorption equilibrium, Adsorption Processes for Gas Separation, CNRS-NSF Gif 91, in F. Meunier & M.D. Levan Eds, *Récents progrès en Génie des Procédés*, **5**, n° 17, pp 49-56.

[L16] Langmuir I. (1916). *Jour. Am. Chem. Soc.*, **38**, p 2221.

[LC87] Le Cun Y. (1987). *Modèles connexionnistes de l'apprentissage.* PhD thesis; Université Pierre et Marie Curie, Paris.

[MP43] McCulloch, W.S. et W. Pitts (1943). A Logical Calculus of Ideas Immanent in Nerveuse Activity. *Bulletin of Mathematical Biophisics* **5**, 115-133.

[Mic92] Z. Michalewicz, Genetic Algorithms+Data Structures=Evolution Programs, Springer Verlag 1992.

[NRC] William H. Press, Saul A. Teukosky, William T. Vetterling et Brian P. Flannery. Numerical Recipices in C, Second edition **3**, pp 111-113.

[NW90] D. Nguyen et B. Widrow. *Neural Network for Control*, chapitre The Truck backer-uppe: an example of self-learning dans Neural Networks, pages 287-300. MIT Pres, Cambridge, MA, 1994.

[Rad91] N. J. Radcliffe, Equivalence Class Analysis of Genetic Algorithms, in *Complex Systems* **5**, pp 183-205, 1991.

[Rec73] I. Rechenberg. *Evolution stategie: Optimierung Technisher Systeme nach Principien des Biologischen Evolution.* Fromman-Holzboog Verlag, Stuttgart, 1973.

[RFLM93] P. Rouchon, M. Fliess, J. Lévine, et P. Martin. Flatness and motion planing: the car with n trailers. In *Proc. ECC'93, Groningen*, pages 1518-1522, 1993.

[RHW86] Rumelhart, D.E. G.E. Hinton, et R.J. Williams (1986). Learning Internal representations by Error Back Propagation. Dans *Parrallel Distributed Processing*, vol. 1, chap 8.

[S94] Sepulveda M. (1993). Identification de Paramètres pour un système hyperbolique. Application a l'estimation des isothermes en chromatographie. PhD thesis, École Polytechnique, Palaiseau.

[SS93] Shub M. and Smale S. (1993). Complexity of Bezout's theorem, *Journal of the American Mathematical Society*, vol 10 n° 2 pp 459-500.

[Sch81] H.-P. Schwefel *Numerical Optimisation of Computer Models.* John Wiley & Sons, New-York, 1981. 1995 - $2^{nd}$ edition.

[VG76] Valentin P. and Guiochon G. (1976). Propagation of Finite Concentration in Gas Chromatography, *Separation Science*, **10**, pp 245-305.

[W90] White H. (1990). Connectionist Nonparametric Regression: Multilayer Feedforward Networks Can Learn Arbitrary Mapings. *Neural Networks* **3** pp 535-549.

# Annexe