2 GENERALISATION ET ESTIMATION DES PERFORMANCES

Résumé

Dans la pratique, l'objectif d'une modélisation statistique n'est pas d'ajuster finement un modèle sur un ensemble d'apprentissage, mais d'obtenir un bon compromis entre performances d'apprentissage et performances de généralisation. Il existe principalement deux manières de résoudre ce dilemme entre le biais et la variance de la famille de fonctions considérée :

- *les méthodes de validation croisée, qui scindent la base de données disponibles de manière à estimer les performances de généralisation du modèle sur des données n'ayant pas servi à l'apprentissage, ce qui permet, a posteriori, d'éliminer les solutions surajustées,*
- *la régularisation qui, par arrêt prématuré de l'apprentissage (early-stopping), ou par ajout d'un terme de pénalisation à la fonction de coût (weight decay), permet de pénaliser a priori les modèles à forte variance.*

En réalité, le surajustement se traduit par une influence trop importante de certains exemples sur l'estimation des coefficients du modèle, qui peut ainsi s'ajuster très précisément à ces exemples. Nous allons étudier dans ce mémoire, par une approche théorique du leave-one-out, une manière de résoudre à la base ce phénomène de surajustement, que la validation croisée classique et la régularisation traitent de manière indirecte.

Un cas de surajustement particulièrement simple à détecter concerne les modèles pour lesquels la matrice jacobienne Z n'est pas de rang plein. Nous nous proposons de le détecter numériquement en vérifiant les propriétés que doivent respecter les termes diagonaux h_{ii} de la matrice de projection $Z\ ({}^tZZ)^{-1}\ {}^tZ$, et nous montrons qu'il correspond à des modèles pour lesquels certains coefficients sont sous-déterminés.

Par ailleurs, nous présentons la notion d'intervalle de confiance sur la sortie du modèle. Nous utilisons dans ce mémoire une expression classique de cet intervalle, fondée sur un développement de Taylor de la sortie du modèle au voisinage de la solution des moindres carrés. Par opposition à la performance de généralisation du modèle, qui peut s'interpréter comme un intervalle de confiance associé à la mesure de la sortie du processus, les intervalles de confiance sur la sortie du modèle dépendent des entrées de ce dernier. Ils permettent ainsi de déterminer les zones de l'espace des entrées où - par manque d'exemples - la confiance sur la prédiction du modèle est trop faible.

En revanche, la détermination d'intervalles de confiance (ou de bornes) sur l'erreur de généralisation empirique, par rapport à l'erreur de généralisation théorique, fait depuis plusieurs années l'objet de recherches sur la théorie de l'apprentissage. Cependant, compte tenu du cadre de cette thèse, ces bornes ne sont pas exploitables pour l'instant.

2.1 Introduction

La propriété d'approximation présentée dans le chapitre précédent soulève, en corollaire, une difficulté bien connue des utilisateurs de réseaux de neurones (ou de tout modèle paramétré) : le dimensionnement des modèles.

Lorsque l'on doit traiter des données bruitées, ce que nous supposerons tout au long de cette thèse, l'objectif est de trouver le modèle optimal, présentant le meilleur compromis possible entre performance d'apprentissage et capacité de généralisation.

Après une introduction à la problématique et une description des principales méthodes utilisées pour y faire face, nous présenterons une discussion sur la nature exacte du phénomène de surajustement.

Nous introduirons ensuite la notion d'intervalle de confiance associé, d'une part, à la sortie d'un modèle, et, d'autre part à l'estimation des performances de généralisation de ce modèle.

2.2 Le dilemme biais / variance

Ce compromis a été formalisé en décomposant la performance moyenne d'un modèle - sur toutes les bases d'apprentissage possibles - en deux parties [Geman 92] : la première, appelée **biais**, rend compte de la différence moyenne entre le modèle et l'espérance mathématique de la grandeur à modéliser ; la seconde, appelée **variance**, reflète l'influence du choix de la base d'apprentissage sur le modèle.

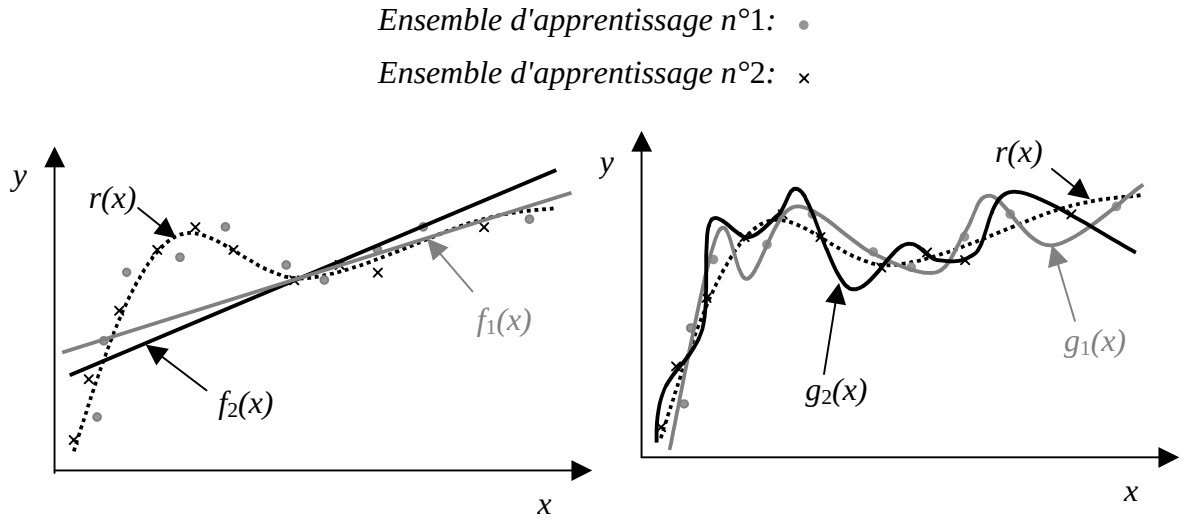

Figure 2.1 : Exemples de familles de fonctions présentant - à gauche - un biais trop élevé - à droite - une variance trop élevée

Sur la figure 2.1, considérons le problème de l'approximation d'une fonction $r(x)$ supposée inconnue, à partir de mesures bruitées de celle-ci. La famille des fonctions affines (notée f) présente un écart important avec le modèle idéal ; cependant cet écart dépend peu de la base d'apprentissage : cette famille de fonction possède donc un biais important et une variance faible. Pour la famille de fonction g, le phénomène inverse se produit : celle-ci est

suffisamment complexe pour s'ajuster au plus près aux points d'apprentissages ; en revanche sa forme varie beaucoup en fonction de ces derniers.

Idéalement, il faut chercher un modèle présentant un biais et une variance aussi faibles que possible. Un bon modèle doit donner ainsi une réponse moyenne satisfaisante tout en dépendant le moins possible des exemples dont on s'est servi pour le concevoir. Dans le cas de l'exemple précédent, la famille de fonctions *h* représentée sur la figure 2.2 réalise très bien ce compromis.

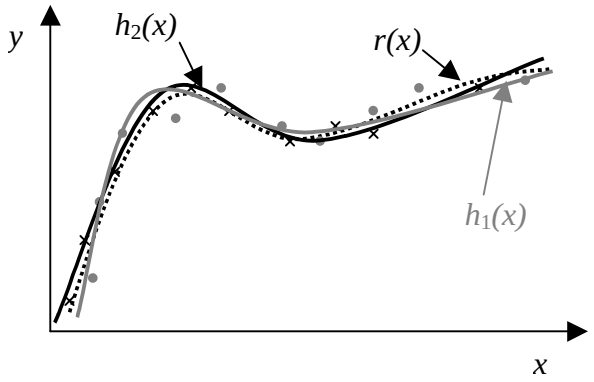

Figure 2.2 : Exemple de famille de fonctions réalisant un bon compromis entre biais et variance

Ce compromis peut certes s'obtenir en augmentant la taille de la base d'apprentissage, mais ce n'est malheureusement pas toujours possible. Dans la pratique, il existe principalement deux façons d'éviter le surajustement (cf. [Gallinari 97]), lorsqu'on dispose d'une base de donnée limitée :

- a posteriori, c'est-à-dire après apprentissage : le surajustement se détecte alors sur la base d'une estimation des performances de généralisation du modèle. La principale méthode utilisée dans le domaine des réseaux de neurones[2] est la validation croisée (voir [Stone 74]), fondée sur un ré-échantillonnage de la base de données,
- a priori, c'est-à-dire en cours d'apprentissage (voire avant celui-ci) : il s'agit des techniques de régularisation, qui visent à pénaliser l'obtention de modèles surajustés, mais qui ne dispensent pas de l'étape d'estimation des performances du modèle.

Dans ce chapitre comme dans les suivants, la question de la sélection de modèle n'est abordée que dans le sens de la sélection de l'architecture optimale, c'est-à-dire de la complexité de la famille de fonctions utilisée. Le problème de sélection des entrées n'est pas traité ici ; le lecteur intéressé pourra pour cela se référer à [Stoppiglia 97].

On suppose généralement que le processus à modéliser comporte plusieurs entrées non bruitées et une sortie bruitée.

---

[2] D'autres méthodes, moins utilisées, existent pour comparer différents modèles entre eux, par exemples les tests d'hypothèses ou les critères d'information (voir [Anders 99]).

Enfin, il est également important de noter que la question de la sélection de l'architecture optimale est étroitement liée à celle de l'estimation des performances de généralisation du modèle : idéalement, ces deux étapes devraient être effectuées simultanément, de manière à comparer des architectures entre elles sur la base de l'estimation des performances de généralisation.

En revanche, il peut être utile - dans certains cas - de séparer la sélection de la taille optimale du modèle, de l'apprentissage du modèle final. Pour ce faire, les notions de bases d'apprentissage et de validation concernent l'étape de sélection de la taille optimale. Dans un second temps, ces deux bases sont regroupées en une seule base d'apprentissage servant à concevoir le modèle final. Dans tous les cas, il est utile de disposer d'une base de test indépendante dont on se sert à la fin pour vérifier la validité et estimer les performances du modèle.

2.3 La validation croisée

Cette méthode repose sur une estimation des performances à partir d'exemples n'ayant pas servi à la conception du modèle. Pour ce faire, on scinde la base d'apprentissage en *D* parties de taille (approximativement) égale. On réalise alors *D* apprentissages du modèle, en laissant à chaque fois une des parties de côté pour le valider (cf. figure 2.3, tirée de [Bishop 97]). La performance du modèle s'obtient à partir des erreurs de validation constatées après les D apprentissages.

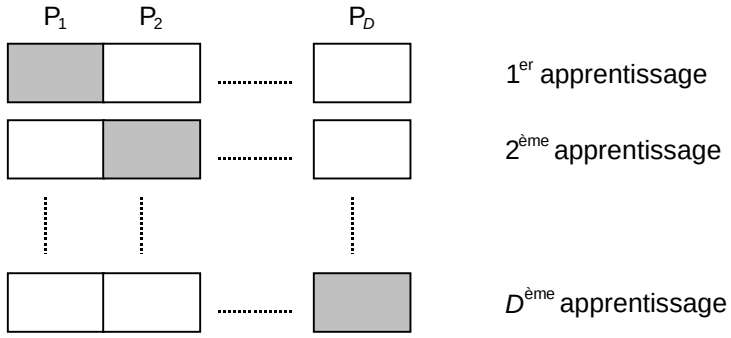

Figure 2.3 : Principe de la validation croisée ; les parties grisées sont utilisées pour la validation et les autres pour l'apprentissage

En utilisant la fonction de coût des moindres carrés, on procède généralement comme suit :

- pour chaque partie laissée de côté, on calcule l'erreur quadratique moyenne de validation (*EQMV*),
- à la fin, la performance de généralisation du modèle - appelée "score de validation croisée" - est estimée en réalisant la moyenne quadratique des *D* erreurs (*EQMV*) précédentes.

Dans le contexte de réseaux de neurones, la recherche de l'architecture optimale s'effectue souvent en partant d'un modèle linéaire et en augmentant progressivement le nombre de

neurones cachés. Le modèle optimal est alors défini comme étant celui qui présente le meilleur score de validation croisée.

La limite naturelle de la validation croisée correspond au cas où *D* est égal au nombre d'exemples dans la base d'apprentissage. Cette méthode est connue sous le nom de "leave-one-out" (voir [Plutowski 94]) car chaque apprentissage n'est validé que sur un seul exemple.

Les difficultés de cette méthode sont de deux ordres :

- le temps de calcul nécessaire, qui - pour une même base d'apprentissage est d'autant plus grand que *D* est élevé (il est donc maximum dans le cas du leave-one-out),
- des performances contrastées en termes de taille de l'architecture sélectionnée et d'estimation des performances. À ce niveau, deux cas sont à distinguer :
  - *le nombre d'exemples est grand au regard de la complexité de la fonction à approcher* (nombre d'entrées, non-linéarité) : dans ce cas, le phénomène de surajustement est difficile à mettre en évidence. La méthode donne certes de bons résultats - même avec un petit nombre de partitions - mais sans grand mérite car il y a peu de risque de surajustement.
  - *le nombre d'exemples est petit au regard de la complexité de la fonction à approcher* : on est obligé d'augmenter le nombre de partitions de façon à garder un nombre suffisant d'exemples pour réaliser l'apprentissage des *D* modèles. Les résultats montrent alors une tendance à la surestimation de la taille des modèles nécessaires et à la sous-estimation des scores de validation croisée. Ceci traduit un phénomène mis en évidence par [Breiman 96] : une petite modification des données d'apprentissage peut entraîner de grandes différences dans les modèles sélectionnés. Autrement dit, si l'on raisonne en termes de fonction de coût, les exemples dont on se sert pour estimer les paramètres d'un modèle peuvent grandement influencer les minima vers lesquels convergent les différents apprentissages. On parle alors d'instabilité vis-à-vis des données d'apprentissage : les *EQMV* calculées à partir des différentes partitions ne peuvent donc pas raisonnablement êtres moyennées pour estimer la performance de généralisation du modèle.

La littérature conseille généralement d'utiliser $D = 10$. Cependant, ne sachant pas a priori s'il dispose de "peu" ou de "beaucoup" d'exemples (au sens défini ci-dessus), le concepteur essaiera souvent différentes valeurs de *D*. Si l'on se rappelle qu'à partir d'une base d'apprentissage, il est recommandé de procéder à plusieurs initialisations des poids de façon à diminuer le risque de minima locaux, on arrive très vite à un nombre d'apprentissages très élevé. En soi, ceci n'est pas grave si les résultats de ces différents essais sont cohérents. Dans le cas contraire, le découragement peut rapidement intervenir…

Dans les chapitre 3 à 5, nous verrons comment remédier à ces difficultés par une approche originale du leave-one-out.

2.4 La régularisation

Ce terme désigne une série de techniques visant à privilégier les modèles les plus réguliers possible. Ceci part d'un constat simple : le surajustement se traduit le plus souvent par un modèle dont la sortie possède, aux endroits où celui-ci s'est ajusté au bruit, une courbure élevée.

La régularisation peut être effectuée de deux manières :

- en arrêtant l'apprentissage prématurément (early stopping), c'est-à-dire avant que ne soit atteint un minimum de la fonction de coût,
- en ajoutant un terme de pénalisation à la fonction de coût (par exemple par la méthode de "weight decay", présentée plus loin dans le paragraphe 2.4.2).

2.4.1 Early stopping

Cette méthode consiste à arrêter l'apprentissage avant qu'il ne commence à s'ajuster au bruit contenu dans les points d'apprentissage, même si un minimum de la fonction de coût n'est pas atteint. Pour cela, on part :

- d'une architecture surdimensionnée, c'est-à-dire susceptible de conduire à un surajustement si on laissait l'apprentissage se poursuivre jusqu'à convergence,
- d'une partition des exemples disponibles en une base d'apprentissage et une base de validation.

On réalise alors l'apprentissage jusqu'au moment où les performances sur l'ensemble de validation atteignent un minimum. Ceci est représenté schématiquement sur la figure 2.4.

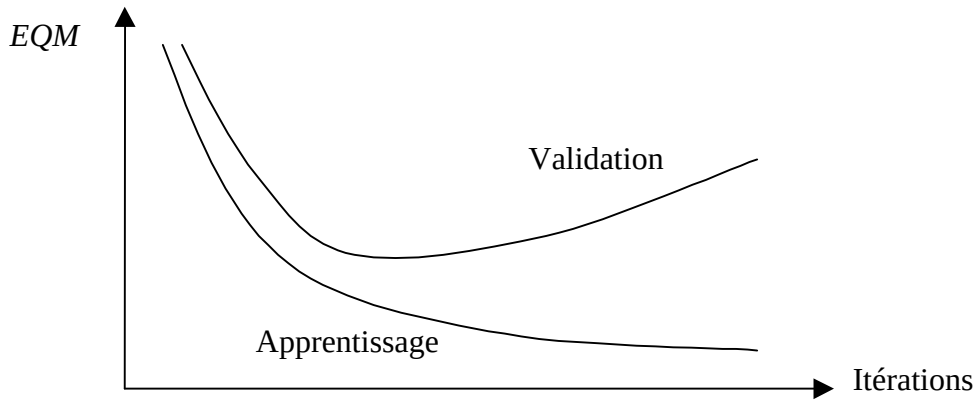

Figure 2.4 : Évolution typique des performances d'apprentissage et de validation

Les reproches que l'on peut faire à cette méthode, qui pourtant donne parfois de bons résultats, concernent à la fois sa mise en œuvre et sa "philosophie".

De même que pour la validation croisée, la meilleure façon de répartir les exemples disponibles en base d'apprentissage / de validation n'est pas définie, mais doit être étudiée au cas par cas, ce qui entraîne des problèmes de reproductibilité des résultats obtenus. Par ailleurs, il n'est plus question ici de procéder d'abord à la recherche de la taille d'architecture

optimale puis de regrouper les bases d'apprentissage et de validation pour avoir une meilleure estimation des paramètres du modèle.

Enfin, cette méthode - dans son principe même - va à l'encontre de la propriété de parcimonie des réseaux de neurones, puisqu'elle utilise des modèles surdimensionnés. Dans la littérature, il n'est pas rare de rencontrer des applications comportant beaucoup plus de paramètres ajustables que d'exemples (voir par exemple [Nelson 91]) ! Pour le concepteur qui cherche à utiliser un minimum de coefficients de manière à avoir un maximum de confiance sur leur estimation, une telle approche ne peut qu'éveiller des soupçons, mais elle est parfois incontournable, par exemple dans le domaine du traitement d'images.

En résumé, la régularisation de la fonction par early stopping est encore utilisée, car elle est rapide et simple à mettre en œuvre.

Dans [Sjöberg 92], les auteurs montrent que l'arrêt prématuré de l'apprentissage revient à utiliser un terme de pénalisation dans la fonction de coût, ce qui justifie la classification de l'early stopping parmi les méthodes de régularisation.

2.4.2 Pénalisation de la fonction de coût (weight decay)

La deuxième façon d'influer sur la régularité du modèle consiste à introduire des contraintes dans la fonction de coût à minimiser. Nous ne détaillerons ici que la technique la plus connue (le *weight decay*), qui consiste à ajouter à la fonction de coût un terme proportionnel à la norme du vecteur des paramètres ajustables[3].

L'idée est ici de privilégier les modèles possédant de petits coefficients (en valeur absolue). La relation entre la taille des coefficients et la régularité de la fonction se comprend aisément en considérant la fonction réalisée par un neurone à une entrée :

$$y = \tanh(w_0 + w_1 x)$$

Le paramètre w_0 ainsi que la plage de variation de la variable x étant fixés, cette fonction est d'autant plus régulière que la valeur absolue du paramètre w_1 est petite :

- si w_1 est très faible, la fonction sera quasiment linéaire,
- si w_1 est très grand, la fonction s'apparentera à un échelon.

On considère donc un terme $\Omega = \sum_i w_i^2$, pour lequel la somme s'effectue sur tous les coefficients du réseau, y compris les biais. Ce terme est le même que celui rencontré dans la méthode dite de *ridge regression* (voir [Saporta 90]), utilisée dans le cadre de régressions linéaires par rapport aux paramètres. Dans les deux cas, l'objectif est de limiter la variance d'un modèle en utilisant un estimateur biaisé.

---

[3] Une autre méthode consiste à pénaliser directement les fonctions à forte courbure par l'ajout - à la fonction de coût - de la norme du vecteur dérivée seconde de la sortie du modèle. Le lecteur intéressé par l'application de cette méthode aux réseaux de neurones pourra consulter l'article [Bishop 93].

Toute la difficulté du weight decay réside dans le dosage optimal entre la fonction de coût initiale et le terme de régularisation. Pour cela, on écrit la nouvelle fonction de coût sous la forme :

$$J^* = J + \nu \Omega \qquad (2.1)$$

Si l'on choisit ν trop grand, le modèle risque d'avoir un biais élevé. Inversement, si ν est trop petit, l'effet du terme de régularisation est trop faible, ce qui se traduit par une variance élevée. La grandeur ν devient donc en fait un paramètre, à estimer au même titre que les poids du réseau : elle est souvent désignée sous le nom d'hyperparamètre.

La seule véritable réponse apportée à l'estimation de cet hyperparamètre et de son interprétation est une approche bayesienne des réseaux de neurones décrite dans [MacKay 92] et d'autres publications du même auteur. Sans détailler les fondements théoriques ni la manière de mettre en œuvre les techniques bayesiennes, les avantages de cette approche sont :

- l'utilisation de tous les exemples d'apprentissage pour sélectionner et estimer les paramètres du modèle,
- l'accès immédiat aux intervalles de confiance.

Il s'agit cependant d'une technique assez compliquée à mettre en œuvre, et peu utilisée dans des applications pratiques.

2.5 Le surajustement

2.5.1 Discussion : qu'est-ce que le surajustement ?

Après cette revue des outils les plus fréquemment utilisés dans le domaine des réseaux de neurones pour sélectionner le meilleur modèle tout en évitant les solutions surajustées, les questions suivantes se posent :

- quelle est l'origine exacte du surajustement ?
- quelle démarche faudrait-il mettre en œuvre pour régler ce problème à la base ?

Le surajustement caractérise une fonction dont la complexité - c'est-à-dire le nombre et la nature des degrés de libertés - est telle qu'elle est capable de s'ajuster exactement aux exemples d'apprentissage, même si ceux-ci sont entachés de bruit. Ce phénomène est donc - à l'origine - un phénomène local : dans certains domaines des entrées, la fonction utilise localement certains de ses degrés de liberté de manière à passer précisément par certains exemples.

Cette définition du surajustement suppose que tous les exemples ont la même importance et que l'on recherche effectivement une solution dont la réponse est - en moyenne - satisfaisante. Généralement, cette hypothèse est formalisée dans la fonction de coût choisie.

Ainsi, pour éviter le surajustement, il faudrait limiter l'influence de chaque exemple sur l'estimation des poids du modèle. Les valeurs de ceux-ci doivent être déterminées par

l'ensemble de la base d'apprentissage, et non par un exemple particulier (on retrouve ici le dilemme biais/ variance).

Dans cette optique, les méthodes précédentes ne s'attaquent pas directement au fond du problème :

- la validation croisée et l'early stopping sont des méthodes palliatives, qui n'empêchent pas le surajustement, mais permettent de le détecter : on se sert d'une base de validation pour détecter les zones de forte courbure de la sortie du modèle, entraînées par l'ajustement trop précis de la sortie du modèle aux exemples d'apprentissage,
- le weight decay est une heuristique : l'équivalence entre le fait que les poids du réseau soient "grands" et le surajustement n'a - à notre connaissance - jamais été démontrée ni dans un sens ni dans l'autre. Certes, elle se comprend intuitivement et [Bartlett 97] a montré que, pour avoir une bonne capacité de généralisation, la taille des poids est plus importante que la taille du réseau. Cependant, ceci pourrait très bien se révéler inexact dans tel ou tel cas particulier : cela dépend de la forme réelle de la fonction à approcher.

L'objectif que l'on a cherché à atteindre dans ce travail est le suivant : trouver un moyen de résoudre à la base le problème du surajustement. Pour cela, après quelques éléments sur la détection du surajustement, nous présenterons une étude détaillée de la théorie et de la mise en œuvre du leave-one-out.

2.5.2 Détection du surajustement

La détection du surajustement se fonde sur la détermination du rang de la matrice Z. Cependant, il est difficile de vérifier - numériquement - si une matrice est de rang plein. Nous proposons donc de considérer la matrice $H = Z\ ({}^tZ\ Z)^{-1}\ {}^tZ$, qui est la matrice de projection orthogonale sur le sous-espace des solutions : si Z est de rang plein, et dans ce cas seulement, H doit vérifier les propriétés suivantes :

$$\forall i \in [1, ..., N]\ \ 0 \leq h_{ii} \leq 1 \text{ où } h_{ii} \text{ est le } i^{\text{ème}} \text{ terme diagonal de } H \tag{2.2}$$

$$\text{trace}\ (H) = \sum_{i}^{N} h_{ii} = \text{rang}\ (Z) \tag{2.3}$$

De plus, dans le cas où la matrice Z possède une colonne dont tous les termes sont égaux, par exemple égaux à 1, c'est-à-dire si le modèle comporte un terme ajustable constant (appelé "biais" dans le contexte des réseaux de neurones) :

$$\forall i \in [1, ..., N]\ \ \frac{1}{N} \leq h_{ii} \tag{2.4}$$

Théoriquement, les termes $\{h_{ii}\}_{i\ =\ 1,\ ...,\ N}$ ne sont définis que dans le cas où ${}^tZ\ Z$ est inversible, c'est-à-dire dans le cas où Z est de rang plein. Dans le cas contraire, le sous-espace de R^N défini par les vecteurs colonnes de Z (sous-espace appelé "sous-espace des solutions") n'est pas, au point correspondant à la solution des moindres carrés, de dimension q.

Ceci est caractéristique du surajustement : en effet, cela signifie que le modèle dispose, localement, de plus de paramètres ajustables que d'exemples[4]. Par analogie avec la résolution d'un système linéaire, le modèle dispose, localement, de plus d'inconnues que d'équations, ce qui conduit à une indétermination.

Contrairement à ce que l'on pourrait supposer, le surajustement ne traduit donc pas réellement la présence de paramètres "inutiles", mais plutôt la "sous-détermination" de certains paramètres.

De plus, il a été montré [Saarinen 93] qu'une déficience dans le rang de la matrice jacobienne a un effet négatif sur l'efficacité des algorithmes du second ordre. Récemment, [Zhou 98] a suggéré une procédure dans laquelle le rang de la matrice jacobienne est aussi réduit que possible, et le réseau est élagué lors de l'apprentissage : les résultats obtenus ainsi présentent une légère amélioration (en termes d'erreur quadratique moyenne sur un ensemble de test), par rapport à des modèles dont les paramètres sont estimés par des algorithmes conventionnels du second ordre. Nous proposons d'utiliser le même principe à un niveau différent.

D'après les relations (2.2), (2.3) et (2.4), trois cas sont à considérer :

1. (au moins) une des valeurs singulières de la matrice Z est égale à zéro : alors rang (Z) < q et les termes diagonaux $\{h_{ii}\}_{i = 1, ..., N}$ ne peuvent être calculés. Il y a donc au moins un des paramètres du modèle qui est sous-déterminé, ce qui se traduit par un surajustement.
2. aucune des valeurs singulières de la matrice Z n'est égale à zéro, mais les valeurs calculées $\{h_{ii}\}_{i = 1, ..., N}$ ne satisfont pas les relations (2.2) à (2.4) : comme dans le premier cas, Z n'est pas de rang plein. L'absence de valeurs singulières nulles résulte alors de problèmes de précision des calculs.
3. toutes les valeurs singulières sont non nulles et les $\{h_{ii}\}_{i = 1, ..., N}$ calculés satisfont les trois conditions (2.2) à (2.4).

Par abus de langage, nous désignerons les minima pour lesquels la matrice jacobienne est de rang plein sous le terme "*minima de rang plein*". Dans le cas contraire, nous parlerons de "*surajustement avec déficience du rang*".

Un calcul des $\{h_{ii}\}_{i = 1, ..., N}$ fondé sur une décomposition de Z en valeurs singulières est présenté en Annexe 1.

Pratiquement, lors d'essais sur différents problèmes, nous avons effectivement constaté les phénomènes suivants :

- lorsque les $\{h_{ii}\}_{i = 1, ..., N}$ calculés ne satisfont pas aux relations (2.2) à (2.4), il y a au moins un neurone caché dont les poids sont très grands et ne contribuent donc pas à la constitution d'un modèle dont la sortie est régulière,

---

4 Prenons par exemple le cas de deux coefficients ne servant qu'à ajuster la valeur de la sortie d'un exemple particulier de la base d'apprentissage : les deux colonnes correspondantes de la matrice Z sont composées de zéros, sauf pour la ligne correspondant à l'exemple en question, ce qui explique la déficience globale du rang de Z.

- lorsque l'erreur résiduelle sur un exemple *i* est très petite (de plusieurs ordres de grandeurs par rapport à l'écart-type du bruit), le h_{ii} correspondant est systématiquement supérieur à 1, c'est-à-dire incompatible avec la relation (2.2).

Ceci confirme que les modèles pour lesquels les relations (2.2) à (2.4) ne sont pas satisfaites sont des modèles surajustés. Dans la pratique, pour une architecture donnée, on réalise plusieurs apprentissages, en partant à chaque fois d'initialisations aléatoires des poids, et l'on garde le modèle le plus satisfaisant pour le problème posé. Par conséquent, les conditions (2.2) à (2.4) peuvent être utilisées pour éliminer les modèles qui sont certainement surajustés. La figure 2.5 montre un exemple de modèle ne satisfaisant pas aux relations (2.2) à (2.4) : les h_{ii} correspondant aux trois exemples désignés sont supérieurs à 1. Ceci correspond effectivement à un surajustement du modèle par rapport aux exemples d'apprentissage.

Tout cela repose sur l'hypothèse selon laquelle la déficience du rang de *Z* n'est pas due à une redondance de coefficients structurelle, due à l'architecture du modèle utilisé (voir Annexe 2).

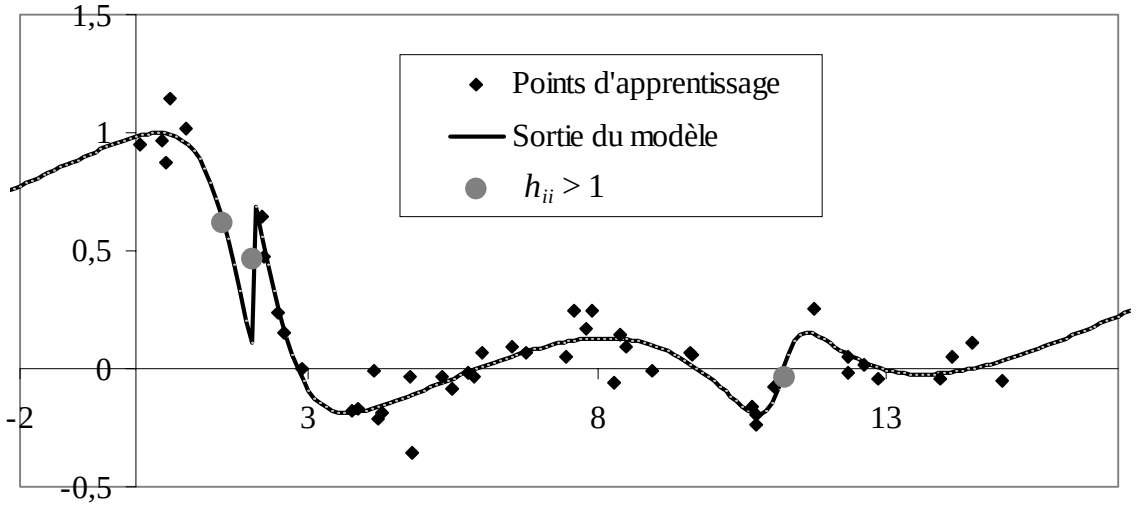

Figure 2.5 : Exemple de modèle présentant trois h_{ii} calculés supérieurs à 1

Nous verrons dans les chapitres 3 et 4 qu'une déficience dans le rang de *Z* est une condition suffisante mais non nécessaire pour attester de la présence d'un surajustement : nous montrerons ainsi que les $\{h_{ii}\}_{i = 1, \ldots, N}$ peuvent être utilisés de manière plus subtile.

2.6 Les intervalles de confiance

2.6.1 Introduction

La notion d'intervalle de confiance est étroitement liée à celle de variable aléatoire : établir un intervalle de confiance - au seuil de confiance 1 - α - pour une valeur aléatoire *Y*, c'est trouver un intervalle qui, avec une probabilité 1 - α, contient la valeur de l'espérance mathématique de *Y*.

Il existe différentes manières d'estimer des intervalles de confiance pour la sortie d'un modèle non linéaire : [Tibshirani 96]. Il s'agit essentiellement des méthodes analytiques décrites dans

[Seber 89], des approches par ré-échantillonnage, type "boostrapping" [Efron 93] et des méthodes basées sur l'approche bayesienne et le weight decay [De Vaux 98]. Nous avons choisi, pour une raison sur laquelle nous reviendrons plus tard, d'utiliser la première de ces approches.

Il a été montré que, dans l'hypothèse d'un bruit de mesure gaussien, et si le modèle hypothèse est vrai (c'est-à-dire si la famille de fonctions considérée contient la fonction de régression inconnue), alors un intervalle de confiance approché pour l'espérance mathématique de la sortie $E(Y_p \mid \boldsymbol{x})$, avec un niveau de confiance 1-α, est donné par :

$$E\left(Y_p \middle| \boldsymbol{x}\right) \in f\left(\boldsymbol{x}, \boldsymbol{\theta}_{LS}\right) \pm t_{\alpha}^{N-q}\, s \sqrt{{}^t\mathbf{z}\left({}^tZZ\right)^{-1}\mathbf{z}}\ , \tag{2.5}$$

expression dans laquelle :

- $\boldsymbol{\theta}_{LS}$ est la solution des moindres carrés obtenue par apprentissage,
- t_{α}^{N-q} désigne la valeur prise par une variable de Student à N - q degrés de liberté avec un niveau de confiance 1-α,
- s est une estimation de l'écart-type du bruit de mesure, estimation sur laquelle nous reviendrons dans le chapitre 4, paragraphe 4.4.

Cet intervalle de confiance suppose en outre que la matrice ${}^tZ\,Z$ est inversible, c'est-à-dire que la matrice jacobienne Z est de rang plein. On ne peut donc pas calculer d'intervalles de confiance sur la sortie de modèles qui présentent un surajustement avec déficience de rang, argument supplémentaire pour ne pas considérer ces modèles.

Nous nous proposons ici de répondre à deux questions fréquemment posées au sujet des intervalles de confiance.

2.6.2 Différence entre performance du modèle et intervalle de confiance ?

Pour répondre convenablement à cette question, il convient tout d'abord de rappeler ce que l'on entend par performance du modèle. La performance (sous-entendu "de généralisation") du modèle est une mesure de la différence entre la réalité et la prédiction du modèle, sur un ensemble représentatif de données. Pour cela, à partir de l'hypothèse (1.4), nous avions supposé que le bruit de mesure était gaussien. En caractérisant le modèle par son écart-type résiduel s, on obtient par conséquent une grandeur directement comparable à l'écart-type du bruit de mesure. Si le modèle hypothèse est vrai, alors la performance de généralisation est égale à l'écart-type du bruit de mesure de la sortie du processus.

Par une démonstration similaire à celle conduisant à la formule (2.5), on peut montrer que, pour tout $\boldsymbol{x}$ fixé, la variable aléatoire $\dfrac{y_p \middle| \boldsymbol{x} - E\left(Y_p \middle| \boldsymbol{x}\right)}{S}$ suit asymptotiquement une loi de Student à N - q degrés de liberté, et ainsi donner un deuxième intervalle de confiance pour $E(Y_p \mid \boldsymbol{x})$, centré cette fois-ci sur la sortie mesurée, avec un niveau de confiance de 1-α :

$$E\left(Y_p \middle| \boldsymbol{x}\right) \in y_p \middle| \boldsymbol{x} \pm t_{\alpha}^{N-q}\, s\,, \tag{2.6}$$

En résumé, la performance d'un modèle est définie par une estimation de l'écart type résiduel : il s'agit d'une grandeur moyenne caractérisant un modèle. En revanche, l'intervalle de confiance sur la prédiction du modèle est une grandeur qu'il faut calculer pour chaque vecteur d'entrée $\boldsymbol{x}$, puisque son expression fait intervenir la grandeur $\boldsymbol{z}(\boldsymbol{x}) = \left.\frac{\partial f(\boldsymbol{x}, \boldsymbol{\theta})}{\partial \boldsymbol{\theta}}\right|_{\boldsymbol{\theta} = \boldsymbol{\theta}_{LS}}$. Il s'agit en fait de la précision avec laquelle le modèle est capable d'ajuster **localement** sa sortie, compte tenu des exemples et des degrés de liberté disponibles.

Dans la représentation traditionnelle du plan (mesure, prédiction), les deux intervalles de confiance doivent donc être distingués et représentés perpendiculairement l'un à l'autre (figure 2.6).

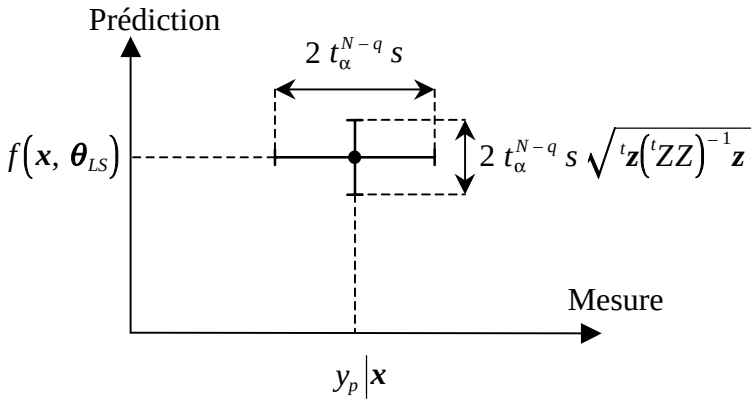

Figure 2.6 : Intervalles de confiance sur la mesure et sur la prédiction

De plus, il existe - pour les exemples d'apprentissage - un lien entre ces deux intervalles de confiance. En effet, nous avons montré dans le paragraphe 2.4.2 que :

$$\sqrt{{}^{t}\boldsymbol{z}^{i}\left({}^{t}ZZ\right)^{-1}\boldsymbol{z}^{i}} = \sqrt{h_{ii}} \leq 1 \qquad (2.7)$$

En résumé :

- la performance du modèle permet de définir un intervalle de confiance sur la mesure, intervalle qui est le même quel que soit le vecteur d'entrée $\boldsymbol{x}$,
- l'intervalle de confiance sur la prédiction du modèle doit être calculé pour chaque vecteur d'entrée $\boldsymbol{x}$. Il est égal à l'intervalle de confiance sur la mesure multiplié par un facteur qui, pour tous les exemples d'apprentissage, est inférieur ou égal à 1.

Considérons par exemple le modèle de la figure 2.7. Il s'agit de modéliser un processus à une entrée à partir des exemples représentés. La sortie du modèle est représentée par une ligne continue ; l'intervalle de confiance à 95 % sur la sortie du modèle par une zone grisée.

Le tracé (mesure, prédiction), pour les points d'apprentissage de l'exemple précédent, est représenté sur la figure 2.8. L'intervalle de confiance sur la mesure, qui est le même pour tous les points, n'y a été représenté qu'une seule fois afin de pas nuire à la lisibilité du graphique.

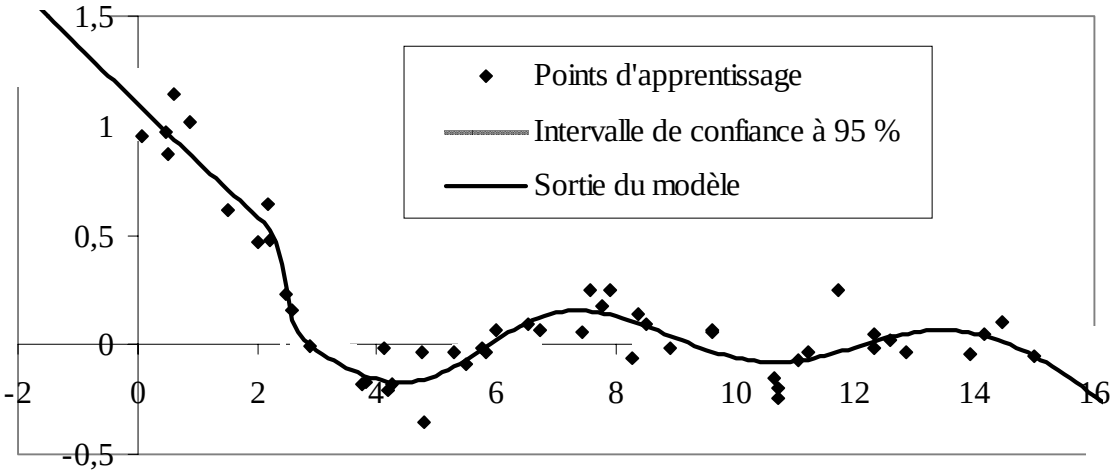

Figure 2.7 : Exemple de modélisation avec intervalle de confiance associé

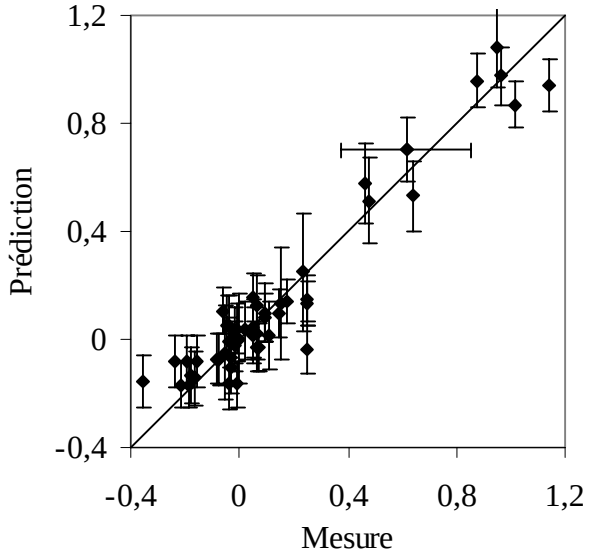

Figure 2.8 : Intervalles de confiance sur la mesure et sur la prédiction pour le modèle de la figure 2.7

Enfin, pour être plus complet, précisons qu'il est possible de combiner les deux expressions (2.5) et (2.6), en supposant qu'elles sont statistiquement indépendantes, afin d'obtenir un intervalle de confiance approché sur la réalisation de la variable aléatoire $Y_p \mid \boldsymbol{x}$, toujours en supposant le modèle hypothèse vrai. À un niveau de confiance 1 - α, l'expression approchée de cet intervalle vaut (voir [Seber 89], page 193) :

$$Y_p \left| \boldsymbol{x} \in \; f\left(\boldsymbol{x}, \boldsymbol{\theta}_{LS}\right) \pm t_{\alpha}^{N-q}\, s \sqrt{1 + {}^{t}\mathbf{z}\left({}^{t}ZZ\right)^{-1}\mathbf{z}} \right. \tag{2.7}$$

Cependant - dans la pratique - sachant que ces deux notions ne sont pas indépendantes, il est plus prudent de considérer séparément l'incertitude sur la sortie du modèle et l'incertitude sur la mesure de la sortie du processus.

2.6.3 Comment interpréter les intervalles de confiance ?

De manière générale, les intervalles de confiance sur la sortie d'un modèle permettent de statuer sur la validité d'une prédiction. Il est néanmoins utile de les interpréter avec plus de précision.

Comme nous venons de le montrer, l'intervalle de confiance est une grandeur locale, à calculer pour chaque vecteur $\boldsymbol{x}$ de l'espace des entrées. Cet intervalle correspond à la précision avec laquelle on est capable d'ajuster localement le modèle. Elle doit donc nous permettre de détecter les zones de l'espace des entrées où il n'y a pas assez d'exemples d'apprentissage.

Il ne s'agit pas pour autant (ou pas seulement) d'une mesure de la densité d'exemples au voisinage d'un vecteur d'entrée $\boldsymbol{x}$. Pour s'en convaincre, examinons l'exemple de la figure 2.9. Il s'agit d'une régression linéaire dont les paramètres sont estimés à partir d'un ensemble d'exemples répartis en deux groupe disjoints.

Dans le cas d'un modèle linéaire, il est facile de vérifier que ${}^t\boldsymbol{z}\ ({}^tZZ)^{-1}\ \boldsymbol{z}$ est une fonction quadratique des entrées du modèle. Ce comportement quadratique se retrouve effectivement sur la figure 2.9 et l'on constate que les intervalles de confiance sont plus faibles dans l'intervalle des entrées [3 ; 7], où il n'y a pas de points, qu'aux endroits où l'on dispose d'exemples. Ceci est tout à fait normal si la fonction de régression du processus est effectivement linéaire. Dans le cas contraire, par exemple si la fonction de régression présentait une bosse au milieu, il faudrait que l'on dispose d'exemples supplémentaires, de façon à justifier l'utilisation d'un modèle plus compliqué qu'un modèle linéaire.

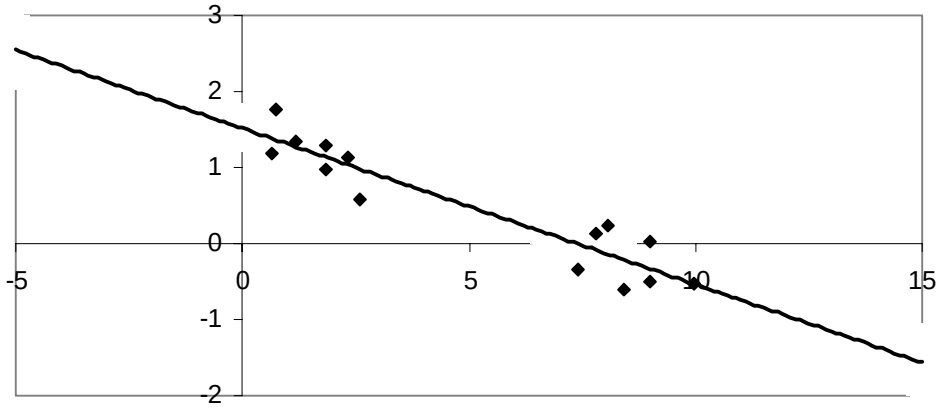

Figure 2.9 : Intervalles de confiance sur une régression linéaire estimée à partir d'un ensemble d'exemples répartis en deux sous-groupes disjoints

Considérons maintenant le cas de la figure 2.10, pour lequel l'intervalle où il n'y a pas de points est plus petit que précédemment, mais contient manifestement une non-linéarité de la fonction de régression. Le modèle considéré est un réseau de neurones à un neurone caché avec terme direct.

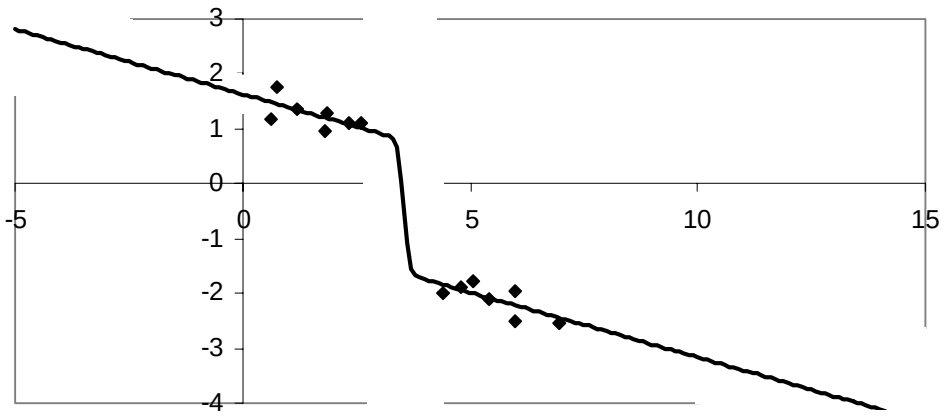

Figure 2.10 : Intervalles de confiance traduisant l'incertitude sur le positionnement d'une non-linéarité

Sur cet exemple, l'incertitude sur le positionnement de la non-linéarité - compte tenu des points disponibles - se traduit par des intervalles de confiance localement très élevés. Il suffit de rajouter quelques points (cf. figure 2.11) pour que cette incertitude soit levée.

Ici encore, le modèle de la figure 2.11 ainsi que les intervalles de confiance associés, ne sont exacts que si le modèle-hypothèse à 1 neurone caché et terme direct est vrai. Cependant, compte tenu des deux exemples ajoutés, il n'y a aucune raison de soupçonner, de la part du processus, un comportement autre que celui modélisé.

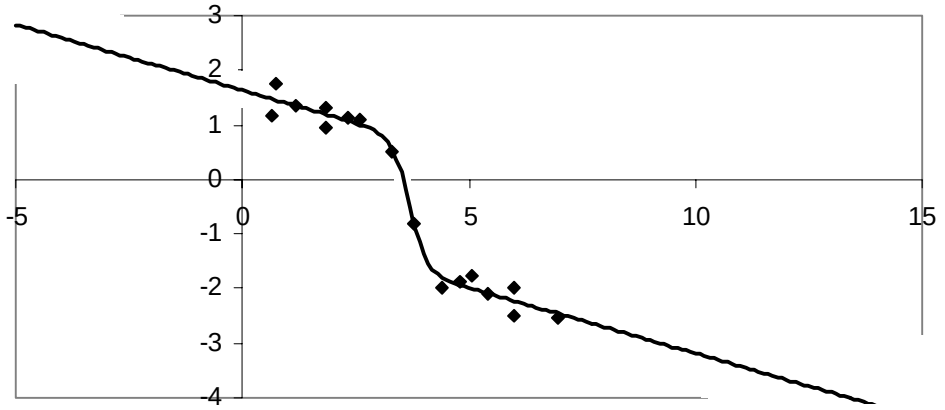

Figure 2.11 : Intervalles de confiance après ajout de deux exemples dans la zone d'incertitude

En résumé, les intervalles de confiance sont bien plus qu'une simple mesure de la densité d'exemples au voisinage d'un point de l'espace des entrées car ils tiennent également compte de la non-linéarité locale du modèle. En fait, les intervalles de confiance traduisent l'incertitude sur le positionnement d'une courbe moyenne (la sortie du modèle), compte tenu des exemples disponibles et de la complexité de la famille de fonctions utilisée. Ils s'avèrent donc constituer une aide indispensable. On trouve, dans la plupart des ouvrages sur la régression non linéaire ([Seber 89] ou [Bates 88]), diverses expressions des intervalles de confiance pour des modèles linéaires et non linéaires.

Il convient cependant de les utiliser avec précaution car des intervalles de confiance restreints ne garantissent pas forcément l'exactitude du modèle : ils peuvent provenir d'un manque d'exemples dans une région où l'interpolation la plus simple ne s'avère pas exacte. Pas plus que les autres méthodes de modélisation, les réseaux de neurones ne sont capables d'extrapoler des comportements non décelables sur les exemples d'apprentissage.

Nous reviendrons sur leur utilisation dans le cadre de la sélection de modèles au chapitre 4.

2.7 Bornes sur les performances de généralisation

L'obtention de bornes sur l'erreur de généralisation d'un modèle est un sujet ouvert qu'étudient beaucoup de théoriciens.

Le problème peut être posé de la manière suivante :

Étant donné un modèle ou encore une famille de fonctions, un algorithme d'apprentissage pour cette famille, et une mesure empirique fondée sur une base de N exemples, est-il possible, pour toute fonction résultant de l'algorithme d'apprentissage, de borner la différence entre la performance réelle de généralisation et sa mesure empirique ?

Les bornes obtenues sont valides avec une certaine probabilité sur l'ensemble d'apprentissage et permettent d'obtenir un intervalle de confiance sur l'estimation des performances de généralisation du modèle. Cet intervalle est alors valable quelle que soit la probabilité portant sur les données du problème et concerne toutes les fonctions du modèle. Les théorèmes concernant ces bornes sont donc des théorèmes de convergence uniforme en probabilité et c'est pour cette dernière raison qu'ils sont souvent cités comme étant des résultats au pire cas : ils sont valables pour toutes les fonctions du modèle et pas seulement pour la fonction obtenue par l'algorithme d'apprentissage.

Les bornes décroissent généralement avec le nombre d'exemples N et conduisent alors à calculer le nombre minimal d'exemples à fournir pour être sûr à 1 - α % que notre estimation de la performance de généralisation ne diffère de la véritable performance de généralisation que d'une quantité ε choisie préalablement.

Tous ces travaux se fondent de près ou de loin sur l'approche de Vapnik et Chervonenkis dont une présentation peut se trouver dans [Bottou 97], ou directement dans [Vapnik 82]. La notion de dimension de Vapnik-Chervonenkis (ou VC-dimension), qui caractérise la capacité d'un modèle (voir par exemple [Euvrard 93]), est centrale.

Dans la pratique, ces bornes sont extrêmement difficiles à obtenir et nécessitent un grand nombre de données (typiquement 100000) pour être intéressantes. Calculer la VC-dimension ou d'autres dimensions combinatoires pour utiliser les théorèmes de convergence uniforme est fastidieux et, souvent, seule une estimation assez lâche de ces dimensions est disponible. À l'heure actuelle, ces approches théoriques ne fournissent pas de résultats pratiques. Leur intérêt est essentiellement de donner des indications sur comment choisir un modèle par rapport à un autre en exhibant des caractéristiques utiles pour le contrôle de la performance de généralisation.

Concernant l'estimation des performances de généralisation par leave-one-out, il est nécessaire (voir [Kearns 97]), pour obtenir de telles bornes, que l'algorithme utilisé ait une certaine stabilité, c'est-à-dire que la solution obtenue en supprimant un exemple de la base d'apprentissage ne soit pas trop éloignée de la solution obtenue sur tous les exemples. Cette notion de stabilité a été formalisée de deux manières différentes par [Devroye 79] et [Kearns 97], dans les deux cas sous la forme "stable à δ près dans 1- β % des cas".

En particulier, [Kearns 97] montre que, sous réserve de stabilité et par rapport à la vraie performance de généralisation, l'estimation obtenue par leave-one-out n'est jamais pire que l'estimation fournie par l'erreur empirique d'apprentissage.

En fait, toute la difficulté réside dans la détermination des propriétés de stabilité de tel ou tel algorithme. Dans le cas d'une minimisation de l'erreur quadratique, l'ajout d'un terme de pénalisation de la fonction de coût (weight decay) semble être une possibilité pour obtenir une certaine forme de stabilité (voir [Bartlett 97]), mais ceci reste à préciser.

Dans les chapitres suivants, nous ne considérerons pas cette question des bornes, dont l'utilisation pratique n'est pas envisageable pour l'instant. Tout au long de ce mémoire, nous utiliserons néanmoins l'erreur quadratique moyenne obtenue par leave-one-out pour estimer les performances de généralisation d'un modèle. Nous montrerons pourquoi, dans certains cas, cette estimation n'est pas raisonnable, ce qui nous amènera à définir une manière plus fiable de procéder.