Notes et commentaires au sujet des conférences de S. Mallat du Collège de France (2018)

L'apprentissage face à la malédiction de la grande dimension

J.E Campagne

Avril 2018, rév. 8 Mars 2020

Table des matières

1 Introduction de la série de conférences **5**

1.1 Traitement du signal 5

1.2 Modélisation/apprentissage non supervisé. 6

1.3 Prédiction/apprentissage supervisé. 7

1.4 Les points communs 7

2 La problématique du sur-apprentissage **9**

2.1 La méthodologie générale 9

2.2 Algorithmes linéaires et à noyaux: Régression/Classification 11

2.3 Biais-Variance 13

3 Malédiction de la grande dimensions (Partie 1) **16**

3.1 Rappels 16

3.2 Cas : L'algorithme des plus proches voisins 17

3.3 Vitesse de décroissance des fluctuations 18

3.4 La régularité (simple) au sens de Lipschitz 19

4 Malédiction de la grande dimensions (Partie 2) **22**

4.1 Les bases orthonormales: linéaire/non-linéaire (introduction) 22

4.2 Débruitage 25

5 Analyse de Fourrier **29**

5.1 Rappels 29

5.2 Analyse Harmonique (Fourier) 30

5.2.1 La convolution: opérateur covariant par translation 30
5.2.2 La base de Fourier 31
5.2.3 Filtrage/Convolution et le tandem approximation-régularité 34
5.2.4 Interpolation et th. d'échantillonage 38
5.2.5 Conventions de la transformée de Fourier (JE) 40

6 Analyse en Ondelettes **41**
6.1 Transformée en ondelettes 43
6.2 Ondelettes et régularité/singularité de fonction 44
6.3 Base orthonormée d'ondelettes 46
6.4 Ondelette versus Filtre 51
6.5 Sparsité ou parcimonie 53
6.6 Quelques commentaires (JE) 54

7 Classification/Régression en grande dimension (Partie I) **70**
7.1 Petit rappel des épisodes précédents 70
7.2 Modèle déterministe vs stochastique: ce n'est pas le probléme! 70
7.2.1 Points de vue bayésien et déterministe 71
7.3 Réduction de dimensionalité: noyau de similarité et hyperplan 74
7.3.1 Trouver l'hyperplan (w, b): la régularisation nécessaire 76
7.3.2 Comment la régularisation permet de stabiliser la réponse? 78
7.3.3 La convexité 80
7.3.4 Le risque en termes des variables duales de w 81

8 Classification/Régression en grande dimension (Partie II) 82

8.1 La régression (modèle à noyau): la balance biais-variance 82

8.2 Le pendant pour la classification 83

8.3 Condition de point selle de Kuhn & Tucker (1950) 85

9 Classification par Support Vector Machine 87

9.1 Le critére de marge 87

9.2 Le meilleur hyperplan possible: pénalisation 89

9.3 La régularisation 91

9.4 Méthode du point selle 92

9.5 Généralisation au cas non-linéaire 93

9.5.1 Noyaux polynomiaux 94

9.5.2 Noyau gaussien 96

9.6 Commentaires de JE. 98

10 La méthode des gradients et une mise en bouche sur les réseaux de neurones 101

10.1 Optimisation par descente de gradient 101

10.1.1 Risque quadratique 102

10.1.2 Batch vs Stochastic gradient 103

10.2 La représentation des données $\phi(x)$ et introduction NN 104

10.2.1 Introduction: que fait 1 neurone et 1 réseau de neurones? 104

10.2.2 Réseau à 1 couche cachée 106

Avertissement: dans la suite vous trouverez mes notes au style libre prises au fil de l'eau et remises en forme avec quelques commentaires (ndje ou bien sections dédiées). Il est clair que des erreurs peuvent s'être glissées et je m'en excuse par avance. Je vous souhaite une bonne lecture.

1. Introduction de la série de conférences

L'objectif de la discipline est l'extraction de la connaissance à partir des données. Voir si il y a des schémas qui se dégagent et que l'on peut mettre sous forme d'algorithmes.

Donc *a priori* les Données ont de l'**Information** (Modèle) qu'il s'agit d'extraire via des **Algorithmes**. Un mot d'ordre: "d'abord comprendre les données!"

C'est une discipline en émergence, il y avait des Statisticiens maintenant ce sont des Informaticiens et des mathématiciens. Les mathématiques ne sont pas uniquement que des statistiques, probabilités, représentions (analyse, Fourier, Ondelettes. . .) et géométries (groupe de symétrie. . .). Coté informatique, il y a l'IA, les bases de données, calculs distribués/parallèles.

Les données de natures différentes: les images, des sons, de la linguistique (texte), mais aussi physique, chimie . . .

1.1 Traitement du signal

C'est l'estimation/approximation dans le but de récupérer du signal $x(u)$ le plus propre possible.

$$x(u) \quad u \in \mathbb{Z}^\ell \text{ (cas discret)}, u \in \mathbb{R}^\ell \text{(cas analogique)}$$

Le cas discret n'est pas le plus simple, en effet dans le cas continue, les notions de régularité sont plus naturelles. Mais on calcule finalement dans le cas discret en pratique. On aura des problèmes du type

$$z = Ax + b$$

Problème inverse (A inversible?)/ comprehensive sensing (de combien de données ai-je besoin?)/compression de données.

Quelle est la dimension? celle de la variable u: d (ex. temps + volume typique en Physique: 4). Donc ici on est en *basse dimension* alors que nous verrons que pour traiter une image par exemple c'est le nombre de pixels qui fixe la dimension du problème (*grande dimension*).

1.2 Modélisation/apprentissage non supervisé.

Estimer un modèle de $x(u)$ revient à trouver sa représentation dans un espace Ω soit **probabiliste** soit **déterministe**. Cela n'a rien à voir avec le fait que le problème sous-jacent soit stochastique ou non. La représentation **probabiliste** peut être plus riche car on peut associer une densité de probabilité $p(x)dx$ que le signal se trouve dans un petit élément de volume de Ω. Dans le cas *déterministe* on peut au mieux dire que la densité de probabilité est uniforme et donc on va chercher des modèles d'entropie maximum.

Donc un modèle probabiliste est un vecteur aléatoire X associé à une densité de probabilité $p(x)$ et on veut construire un estimateur $\tilde{p}(x)$. Pour cela on se sert des données (exemples), $\{x_i\}_{1\leq i\leq n}$.

En 1d, on fait un histogramme simple. Mais la dimension est celle du signal d, c'est typiquement le nombre d'échantillons. Et là il est plutôt de l'ordre de 10^{6-9} si on pense aux pixels d'une image, mais peut être **beaucoup plus grand** encore 10^{23} en mécanique statistique.

Et donc les méthodes simples, intuitives ne marchent plus du tout car en très grande dimension il n'y a quasiment jamais des points $x(u)$ proches les uns des autres ou alors au prix d'avoir des boites dont le volume est commensurable avec tout l'espace. On ne peut faire aucun calcul local en grande dimension et il faut faire des hypothèses assez forte pour estimer $p(x)$.

La représentation probabiliste, on le verra par la suite, ne résout en rien les problèmes de malédiction de la dimensionalité.

Si on a un Modèle on peut faire du Traitement du Signal car on sait « où se trouve le signal » et le codage est plus pertinent (Prédiction). On peut faire de la Synthèse de nouveaux signaux/images. On peut également aborder l'explication du phénomène sous-jacent. Typiquement Physique Statistique.

1.3 Prédiction/apprentissage supervisé.

Estimer la réponse à une question (y) à partir des données x. Si c'est un problème de *classification*, y appartient à un alphabet (index de classe) et il n'y a pas de topologie *a priori* sur le y. L'enjeu est de définir des distances entre les classes. Si c'est un problème de *régression* $y \in \mathbb{R}$ ou $\mathbb{R}^c$ que l'on peut voir comme c réels. Si la régression semble plus difficile car c'est comme si on avait une infinité de classes, il n'en est rien et la difficulté est équivalente. En classification on calcule des frontières dont la dimension est juste $d-1$ donc quand d est grand ça ne fait pas de différence de trouver une frontière ou bien une représentation de y.

Si réponse unique alors $y = f(x)$ (c'est le cas en très grande dimension, il n'y a pas deux échantillons identiques) et l'enjeu est d'estimer f

$$y = f(x) \to \tilde{y} = \tilde{f}(x)$$

avec $\tilde{f}$ est une approximation de f dont le nombre de variables est d donc très grand.

Mais en apprentissage supervisé (classification ou régression) on a des exemples $\{x_i, y_i\}_{1 \leq i \leq n}$. On sait a priori que $y\_i = f(x\_i)$ donc on estime $\tilde{y}_i = \tilde{f}(x_i)$ et l'on voudrait que $\tilde{y}_i \sim y_i$. Dans le cas de l'apprentissage « non supervisé » on ne dispose pas de $p(x_i)$. Mais pour autant le cas « supervisé » a sa propre difficulté car la diversité des fonctions f est beaucoup plus grande que celle des densités de probabilités $p(x)$.

Les problèmes attaqués: reconnaissance de la parole, vision, médical, physique, sociologique, neurophysiologie. Existe-t'il une classe d'équivalence entre tous ces problèmes ? Si c'est le cas les algorithmes développés dans un domaine peuvent être recyclés, et l'on peut également développer des algorithmes génériques.

1.4 Les points communs

A priori il y a des modèles. L'enjeu de la **régularité** de x est fondamentale pour comprendre la représentation. Mais la régularité n'est pas facile: il y a des singularités et pas partout (voir une image). Donc en compression, traitement du signal, il faut comprendre la régularité pour échantillonner astucieusement (régulier: cas linéaire; singulier cas non linéaire). Quelle est la complexité du modèle?

Dans le cas $p(x)$, on est en grande dimension. On va aborder le problème par des notions d'invariance (translation/rotation d'image) donc des processus stationnaires avec des proba qui ne changent pas. L'Entropie va mesurer la complexité et il y a les 2 notions d'entropie qui va nous intéresser: l'entropie de Kolmogorov/déterministe et l'entropie de Shannon/probabiliste.

En prédiction, la réponse à la question $f(x_i)$ est l'interpolation $f(x)$. Le choix de la fonction d'approximation vient de la régularité qui définit l'ensemble/classe de modèle $\mathcal{H}$. Or, les mathématiques en traitement du signal (basse dimension) sont bien comprises (espace Sobolev. . .) par contre ici en grande dimension, on manque d'outils actuellement. Régularité et parcimonie sont des notions duales l'une de l'autre.

Outil	Traitement Signal	Modélisation	Prédiction (App. Supervisé)
Modèle	Régularité de $x(u)$	Régularité de $p(x)$	Régularité de $f(x)$, $f \in \mathcal{H}$
Complexité	Entropie (Kolmogorov/Shannon) déterministe/probabiliste	Invariance/Entropie	
Représentation $\Phi(x)$	Base/Dictionnaire redondant, Parcimonieuse		
	$x = \sum_m \alpha_m g_m$ avec $\Phi(x) = \{\alpha_m\}$ ex. $\mathcal{D} = \{g_m\}$ (Fourier/Ondelettes)	$\Phi(x)$ moments généralisés (probabiliste): famille d'espérance ex. $\{E(\phi_k(x)\}_k$ avec $\phi_k(x) = x^2$ et des polynômes ou pas en général	$\Phi(x)$; pattern discriminant sur y. Changement de représentation pour faire apparaitre un problème linéaire. Notion Invariant.

Outil	Traitement Signal	Modélisation	Prédiction (App. Supervisé)
informatique	ici un outil	Notion de mémoire « associative », réalité virtuelle, info. graphique (synthèse)	l'I. A, GPU et Python mais en fait pas très facile de maitriser ce qu'on fait

Si le Traitement du Signal s'est développé depuis les années 30 avec un domaine industriel fort avec de très beaux problèmes inverses, le cadre des math est bien compris: Analyse Harmonique (Fourier), Stat. et Optimisation.

Modélisation: TS + probabilité et concentration/déviation liée à l'Entropie dues à la grande dimension (loi des grands nombres), th. des groupes mais pas tellement. Les maths depuis 70 bien que l'entropie antérieure.

Prédiction: TS + Modélisation; th des groupes plus présente et géométrie pour trouver des distances (classification)

L'informatique (I.A+ Python) pas très difficile d'accès mais la basse dimension du Traitement du Signal est mal connue par les acteurs qui débutent en Machine Learning. Les Réseaux de Neurones profonds, on ne comprend pas bien donc il faut commencer par la basse dimension. Autre aspect à maitriser c'est le duel biais/variance et *la malédiction de la grande dimension*.

2. La problématique du sur-apprentissage

2.1 La méthodologie générale

La première partie du cours rappelle les ingrédients pour faire les challenges proposés.

Avant tout il faut **se familiariser avec les données**: comprendre leur statistique, voir s'il faut les débruiter avant traitement...

Séparer en (3/4, 1/4) les échantillons pour l'entrainement et les tests internes.

Ensuite, ne pas tomber dans l'idée d'utiliser des Réseaux de Neurones (notés par la suite N.N ou MLP) et pire des Deap NN (DNN). Si le nombres d'échantillons d'apprentissage n'est pas très grand, commencer par : des classificateurs/regresseurs linéaire, à noyaux, des Decisions Trees, du Boosting (Xgboost sur des arbres, Gradient Decision Tree). Pour des cas à grandes dimensions (ex. images. . .) alors les NN et DNN seront plus adaptés.

En apprentissage supervisé: y est la réponse à une question posée sur des données $f(x)$

$$x \to \tilde{y} = \tilde{f}(x)$$

avec $\tilde{f}$ sélectionnée parmi une classe de fonction $\mathcal{H}$. Q: Comment va-t'elle être sélectionnée? R: c'est à travers l'apprentissage sur des données dont on connait la réponse:

$$\{x_i, y_i\}_{i \leq n}$$

et on veut que l'estimation de la réponse $\tilde{f}$ soit proche de la réponse vraie, donc

$$\tilde{y}_i = \tilde{f}(x_i) \simeq f(x_i) = y_i.$$

On donne une fonction de risque qui mesure l'erreur $r(y, \tilde{y})$ qui se différentie si on est en « régression » ou « classification ». Typiquement

Régression	Classification
$y \in \mathbb{R}$	$y \in \mathcal{A}$
$(y - \tilde{y})^2$	1 si $y \neq \tilde{y}$, 0 sinon

mais il y a d'autres fonctions selon le cas de figure.

Donc on estime sur les échantillons d'entrainement un risque empirique calculé à partir de l'algorithme après apprentissage:

$$\tilde{R}_e(\tilde{f}) = \frac{1}{n} \sum_{i=1}^{n} r(y_i, \tilde{f}(x_i))$$

et on veut le minimiser pour sélectionner $\tilde{f}$. Mais ce qui nous intéresse vraiment c'est l'erreur de généralisation que l'on veut minimiser:

$$R(h) = E_{X,Y}(r(Y, h(X))))$$

Dans un challenge on donne $\{x_i, y_i\}_{i \leq n}$, la fonction de risque empirique $\tilde{R}_e$, et des exemples de tests $\{x_i^t\}_{i \leq n_t}$ pour lesquels le site de soumission calcule le score:

$$\tilde{R}_e^t(\tilde{f}) = \frac{1}{n_t} \sum_i r(y_i^t, \tilde{f}(x_i^t))$$

avec y_i^t caché. On ne peut faire que **2 soumissions par jour**.

La différence entre $\tilde{R}_e(\tilde{f})$ et $\tilde{R}_e^t(\tilde{f})$ est que le premier risque est « contaminé » par les échantillons d'apprentissage, alors que le second ne l'est pas, il est une meilleure estimation du risque de généralisation $R(h)$. Ceci étant dit si on fait beaucoup de soumissions sur le site pour améliorer son algorithme, à la fin le score $\tilde{R}_e^t(\tilde{f})$ est contaminé également car l'algorithme se sera adapté aux échantillons de test.

Donc le protocole prévoit en **Juin (1er)** un premier passage en revu de tous les algorithmes soumis sur une série de nouveaux échantillons que personnes n'aura vu auparavant, et en **Décembre** un second et dernier benchmarking sera fait pour clore les challenges.

2.2 Algorithmes linéaires et à noyaux: Régression/Classification

$$x = (x_1, \ldots, x_d)$$

avec d très grand, donc très peu de chance que 2 échantillons soient proches l'un de l'autre. Donc au lieu de trouver par exemple une frontière complexe pour séparer 2 classes, on va essayer d'adapter une représentation $x \to \phi(x)$, pour obtenir

$$\phi(x) = (\phi_1, \ldots, \phi_m) \in \mathbb{R}^m$$

de nouvelle dimension m, et on espère que la séparation entre les 2 classes soit un **hyper-**

plan. On a fait un aplatissement de la frontière.

Mettons que l'on connaisse la paramètrisation de $\phi(x)$, le classificateur linéaire est donné par le signe de la distance d'un échantillon par rapport à l'hyperplan de normale w de m-dimension:

$$\tilde{y} = \text{sgn}(\langle \phi(x), w \rangle - b) = \text{sgn}\left[\sum_{k=1}^{m} \phi_k w_k - b\right]$$

On a sélectionné un algorithme $\mathcal{A}$ qui a pour paramétres (m+1): w, b.

Il faut trouver (w, b) qui minimise le risque empirique de classification binaire puisque on teste ± 1. Dans le cas d'une régression, y est continue

$$y = \langle \phi(x), w \rangle - b$$

et le risque empirique est la différence quadratique $(y - (\langle \phi(x), w \rangle - b))^2$.

Donc l'enjeu en fait est dans le choix de la représentation $\phi(x)$. Les algorithmes de Boosting et de Decision Trees peuvent se voir dans ce cadre linéaire.

L'axe de projection est le critère discriminant entre les 2 classes par combinaison linéaire. Les attributs ϕ_k sont de faibles discriminants mais l'agrégation de tous ces attributs peuvent très bien séparer les échantillons si leur nombre est suffisamment grand (c-a-d la dimension m grande). Dans le cas de Decision Tree et Gradient Decision Trees en fait on va faire également une agrégation (vote) de discriminants faibles.

Les réseaux de neurones en fait vont à la fois apprendre (w, b) mais aussi les ϕ_k, donc la représentation est aussi acquise. Mais même là il faut de l'information *a priori* (en NN/MLP c'est son architecture qui est l'*a priori*).

Donc avec l'apprentissage on a minimiser le risque empirique:

$$\tilde{f} = \underset{h \in \mathcal{H}}{\text{argmin}}\ \tilde{R}_e(h)$$

mais on veut minimiser le risque de généralisation:

$$f_a = \underset{h \in \mathcal{H}}{\text{argmin}}\ R(h)$$

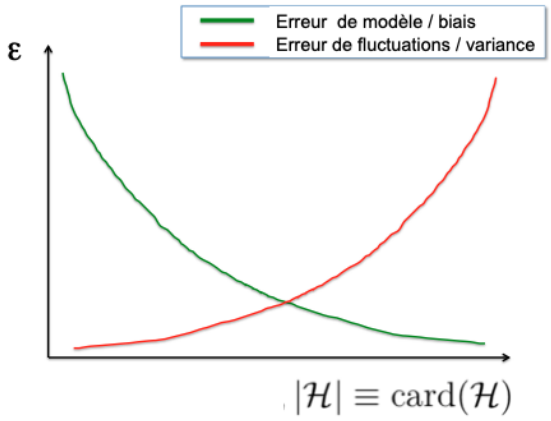

FIGURE 1 – Erreur de Biais-Variance en fonction de la taille de l'ensemble $\mathcal{H}$.

et la différence entre $\tilde{R}_e(h)$ et $R(h)$ est le sujet principal du sur-apprentissage qui survient rapidement quand on se conforme trop aux données.

2.3 Biais-Variance

Donc que vaut $R(\tilde{f})$ c'est-à-dire le vrai risque évalué avec mon estimateur entrainé. On démontre que si on appelle f_I l'algorithme « idéale »:

$$\boxed{R(f_I) \leq R(\tilde{f}) \leq R(f_I) + 2\underset{h\in\mathcal{H}}{\text{Max}}\ |R(h) - R(\tilde{h})|}$$

(la démonstration est dans les notes de cours associées)

Ce que ça dit (surtout la deuxième inégalité) c'est que **l'erreur de généralisation** $R(\tilde{f})$ est bornée par un **terme de biais** car la *classe* $\mathcal{H}$ *n'est pas parfaite* même si on pourrait idéalement obtenir le minimum de risque

$$R(f_I) = \underset{h\in\mathcal{H}}{\text{argmin}}\ R(h) = \underset{h\in\mathcal{H}}{\text{argmin}}\ E_{X,Y}(r(Y, h(X)))$$

et un **terme de fluctuations** lié à la *variabilité de* h *dans* $\mathcal{H}$. Schématiquement on a

$$Erreur \leq Biais + Variance$$

mais on ne peut réduire les 2 termes en même temps. Si la classe $\mathcal{H}$ est grande alors le biais diminue, mais les fluctuations augmentent (Voir une illustration sur la figure 1).

Le travail est qu'en même de trouver les classes de modèles dont les tailles sont petites (cf. $card(\mathcal{H})$ petit) pour contrôler les fluctuations (**entropie**) et bonnes pour bien approximer le problème et garder le biais le plus faible.

Le théorème "Probably approximately correct " (**PAC**) stipule que l'on veut des résultats *uniformes* qq soit la distribution de (X, Y) car non connue *a priori*, et que pour n tendant vers ∞, le terme de fluctuations tende vers 0.

— **Théorème PAC**: Si

$$R(h) \in [0,1](\text{borné}), |\mathcal{H}| \equiv \text{card}(\mathcal{H}) < \infty$$

alors

$$\mathbb{P}\left(\underset{h\in\mathcal{H}}{Max}|R(h) - R(\tilde{h})| \leq \varepsilon\right) \geq 1 - \delta$$

et pour cela on va avoir besoin d'un nombre d'échantillons tel que

$$n \geq \frac{\log(|\mathcal{H}|) + \log(2/\delta)}{2\varepsilon^2}$$

C'est à dire que si on veut une petite erreur ε avec une probabilité assez grande (c-a-d un petit δ) et bien il faut aussi le faire avec un ensemble $\mathcal{H}$ de petite dimension pour ne pas exploser le nombre d'exemples que l'on doit avoir. Mais ne veut-on pas le beurre, l'argent du beurre et le sourire de la crémière ? Une autre façon de voir cette inégalité, si n fixé:

$$\boxed{\varepsilon^2 \geq \frac{\log(|\mathcal{H}|) + \log(2/\delta)}{2n} = \frac{\log(|\mathcal{H}|)}{2n} + terme\ de\ condition}$$

Donc ce que je peux contrôler est grosso modo la taille de $\mathcal{H}$ soit la dimension de l'espace des paramètres.

A quoi correspondent ces 2 termes? En régression linéaire

$$h(x) = \sum_{k=1}^{m} \phi_k w_k - b$$

et donc les paramètres $(\{w_k\}_{k\leq m}, b)$ sont dans $\mathbb{R}^{m+1}$ et donc peuvent prendre un nombre

infini de valeurs a priori. Mais si on suppose $\phi_k < C$, et si on fait des petits changements, la quantité $|R(h) - R(\tilde{h})|$ ne varie pas beaucoup. On peut donc **quantifier les paramètres** tels que

$$w_k = p_k \Delta$$

(idem pour b) avec les p_k ayant N valeurs possibles. Donc $N^{m+1} = |\mathcal{H}|$ et

$$\frac{\log(|\mathcal{H}|)}{2n} = \frac{(m+1)\log N}{2n} \sim \left(\frac{m}{n}\right)$$

Donc pour pouvoir contraindre m paramètres il faut au moins autant d'échantillons et si on ne veut pas trop de fluctuations il faut vraiment que $n \gg m$. Paradoxalement (*a priori*) pour les NN/MLP le nombre de paramètres est nettement plus important que le nombre d'échantillons de training. Donc attention en NN il y a des non-linéarités contractantes.

Autre interprétation, $\log|\mathcal{H}|$ est lié à l'entropie: c'est le nombre de bits pour qualifier $\mathcal{H}$. Plus généralement c'est la notion de complexité.

La démonstration de ce théorème est dans les notes de cours. Il y a un lemme (*inégalité de Hoeffding*) qui contrôle **les grandes déviations**.

Soient $\{Z_i\}$, n variables i.i.d / $\forall i \leq n$, $Z_i \in [a, b]$ and $E(Z_i) = \mu$, si $\bar{\mu} = 1/n \sum Z_i$ alors $\forall \varepsilon > 0$ on a

$$\mathbb{P}\left(|\mu - \bar{\mu}| \geq \varepsilon\right) \leq 2e^{-\frac{2n\varepsilon^2}{(b-a)^2}}$$

(nb: l'estimateur est non biaisée $E(\bar{\mu}) = \mu$). L'hypothèse « iid » (**indépendantes et de même loi**) est forte et les erreurs d'apprentissage/généralisation viennent du biais (= non indépendance) entre les échantillons. C'est un travail très important des personnes qui fournissent les challenges.

Dans la démonstration du théorème PAC, ce qu'il apparait c'est que la borne supérieure est obtenue par la majoration de la probabilité de l'union de deux ensembles, par une somme de probabilité. Ceci pourra être raffiné dans la suite.

3. Malédiction de la grande dimensions (Partie 1)

3.1 Rappels

Les algorithmes $\tilde{f}$ que l'on sélectionne sont ceux qui minimisent un risque empirique

$$\tilde{f} = \underset{h\in\mathcal{H}}{\operatorname{argmin}}\ \tilde{R}_e(h) = \underset{h\in\mathcal{H}}{\operatorname{argmin}} \left\{ \frac{1}{n} \sum_i r(h(x_i), y_i) \right\}$$

et on voudrait que ce risque soit un bon estimateur du risque moyen (de généralisation) tel que

$$f_I = \underset{h\in\mathcal{H}}{\operatorname{argmin}}\ E_{X,Y}[r(h(x), Y)]$$

On a établi que le risque empirique du modèle entrainé est borné

$$R(f_I) \leq \tilde{R}(\tilde{f}) \leq R(f_I) + 2\underset{h\in\mathcal{H}}{Max}|R(h) - R(\tilde{h})|$$

avec une borne supérieure donnée par 2 termes: le **biais** (« le modèle est-il ad equa pour répondre à la question du problème? ») et la **variance** ou les fluctuations (« ma classe de modèle est-elle trop grande? »).

De plus on a vu que si le risque est borné disons [0,1] alors le terme de fluctuations n'est pas trop grand, c'est-à-dire

$$\mathbb{P}\left(\underset{h\in\mathcal{H}}{Max}|R(h) - R(\tilde{h})| \leq \varepsilon \right) \geq 1 - \delta$$

que si on a

$$n \geq \frac{\log(|\mathcal{H}|) + \log(2/\delta)}{2\varepsilon^2}$$

ou à n fixé on déduit l'erreur de fluctuations

$$\varepsilon^2 \geq \frac{\log(|\mathcal{H}|) + \log(2/\delta)}{2n}$$

Implicitement la base des échantillons d'entrainement sont un bon reflet des probabilités sous-jacentes (X, Y).

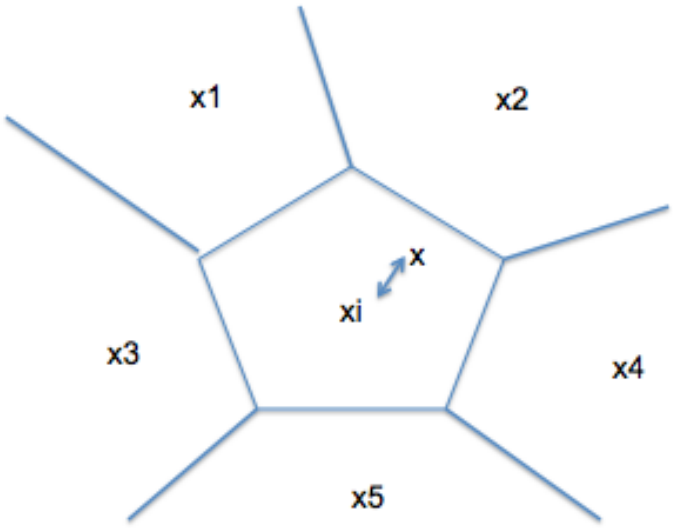

FIGURE 2 – Cellules engendrées par l'algorithme des plus proches voisins.

3.2 Cas : L'algorithme des plus proches voisins

Mettons, un algorithme raisonnable *a priori* est de dire que mon approximation $\tilde{y}$ de la réponse y à partir de x satisfait

$$\tilde{f}(x) = \tilde{y} = y_i \quad si \quad ||x - x_i|| \leq ||x - x_j|| \; pour \; (j \neq i)$$

Cela dessine des cellules à l'intérieure desquelles x et associé à x_i (figure 2) dont on connait la réponse y_i.

Mais pourquoi ça ne marche pas ? En fait si j'ai m paramètres on a vu que l'erreur est dominée par le rapport m/n avec n le nombre d'échantillons. Or, ici le nombre de paramètres m est essentiellement le nombre de points $m \sim n$, et donc ε ne sera jamais petit. On ne contrôle pas l'erreur de généralisation. Il faudra(it) le régulariser pour faire chuter ce nombre de paramètres libres [1].

1. NDJE: l'algorithme K-means est un cas où le nombre de seeds choisi *a priori* est petit par rapport à n

3.3 Vitesse de décroissance des fluctuations

La façon de contrôler l'erreur de généralisation c'est de contrôler la vitesse de décroissance des erreurs de fluctuations (variance): trouver la bonne classe $\mathcal{H}$ et le nombre d'échantillons qui ne doit pas exploser.

Imaginons par exemple que le risque de l'algorithme idéal f_I décroisse avec la taille de la classe comme

$$R(f_I) \leq C(\log|\mathcal{H}|)^{-\alpha}$$

alors

$$\mathbb{P}(R(\tilde{f}) < 3\varepsilon) \geq 1 - 2e^{-(C/\varepsilon)^{1/\alpha}} \quad si \quad n \geq \frac{C^{1/\alpha}}{\varepsilon^{2+1/\alpha}}$$

Si on est capable donc de borné le risque alors avec une probabilité quasi égale à 1, l'erreur va être très petite si n est grand $\sim (1/\varepsilon)^{2+1/\alpha}$. **Par contre α va gouverner la cinétique, et en grande dimension la décroissance naturelle est très lente et donc n devrait être absolument énorme.**

On va chercher des théorèmes de ce type dans la suite du cours qui nous guide sur comment obtenir l'erreur minimale. La démonstration part du théorème PAC en fixant $\log|\mathcal{H}| = \log(2/\delta) = n\varepsilon^2$. Donc, le problème est de trouver des classes d'estimateurs dont la décroissance est la plus rapide possible, où donc le α est le plus grand. Dans ce cas la partie statistique est sous contrôle. Mais c'est pas simple ! On va supposer y est unique alors $y = f(x)$ avec f inconnue, et donc le risque donné par le fait que l'on se place dans la classe $\mathcal{H}$ est

$$R_I = \min_{h\in\mathcal{H}} E_X[r(h(X), f(X))]$$

C'est donc un **problème d'approximation** de la fonction f (inconnue) par une fonction $h \in \mathcal{H}$; mais on ne connait pas la distribution de probabilité de X. Une des façon de se protéger de cette méconnaissance de la statistique des données et de dire la moyenne est plus petite que la valeur maximale. Donc, avec un risque quadratique (type régression)

$$R_I \leq \min_{h\in\mathcal{H}} \sup_{x\in\Omega} |h(X) - f(X)|^2 = \min_{h\in\mathcal{H}} ||h - f||_\infty$$

où Ω est le support des données (on ne connait pas la densité de probabilité). On veut

contrôler $||h - f||_\infty$ avec f inconnue. Il nous faut des *a priori* sur la fonction. **Donc, non seulement il faut des échantillons des données et il faut des formes de modèles sur les données.** On cherche donc à contrôler la quantité

$$\max_{f \in \mathcal{C}} R(f) = \max_{f \in \mathcal{C}} \min_{h \in \mathcal{H}} ||h - f||_\infty$$

Le théorème précédent nous dit que si

$$\max_{f \in \mathcal{C}} R(f) \leq C(\log |\mathcal{H}|)^{-\alpha}$$

alors on a gagné.

Mais quelle type d'information *a priori* peut-on avoir? C'est la **régularité** des fonctions. Pensons à un problème d'approximation à 1 dimension, si la fonction sous-jacente varie lentement on a pas besoin de beaucoup d'échantillons, a contrario si la fonction est très irrégulière le nombre d"échantillons peut être grand ou bien ils seront localisés à des endroits de singularité.

3.4 La régularité (simple) au sens de Lipschitz

Le problème $x \in \mathbb{R}^d$ avec d très grand.

— **Définition**: f est **localement Lipschitz** en x si

$$\exists\ C_x / \ \forall x' \in \mathbb{R}^d \quad |f(x) - f(x')| \leq C_x ||x - x'||$$

(nb. $||x||^2 = \sum |x_i|^2$)

On dit que f est **uniformément Lipschitz** si elle est en tout point et toutes les C_x sont plus petites que C. Si f a une dérivée bornée alors elle est Lipschitz.

Si f est Lipschitz sur $\mathbb{R}$ alors elle est dérivable presque partout. Donc on a des points où la dérivée n'est pas déterminée. En dimension d cela se généralise avec les dérivées partielles. **On peut considérer que la notion de fonction Lipschitzienne est « équivalente » à une fonction à dérivée bornée.**

— **Définition**: f est localement Lipschitz-α si

$$\exists\ C_x, p_x(x') \text{ de degré q} < \alpha / \ \forall x' \in \mathbb{R}^d \quad |f(x) - p_x(x')| \leq C_x ||x - x'||^\alpha$$

Typiquement on regarde le résidus du **polynôme de Taylor** en x de $f(x)$. Si $C_x < C$ alors on dit que la fonction est Lipschitz-α uniforme.

Donc, mettons que l'on a une classe de fonctions Lipschitz

$$\mathcal{C} = \Big\{ f : \mathbb{R}^d \to \mathbb{R} \ / \ \text{f est uniforme Lipschitz} \Big\}$$

De combien d'échantillons a-t'on besoin? On a vu qu'il fallait contrôler la vitesse de décroissance de

$$\max_{f \in \mathcal{C}} R(f) \leq C(\log |\mathcal{H}|)^{-\alpha}$$

Reprenons l'algorithme des **plus proches voisins**. On sait que la fonction étant Lipschitz autour des points échantillons $\{x_i\}$ alors

$$|f(x) - f(x_i)| \leq C_x ||x - x_i||$$

c'est-à-dire qu'on a une bonne approximation de $f(x)$ en disant qu'elle est égale à $f(x_i)$. On abouti alors à une **approximation par morceaux** (figure 3). On généralise en 2D... Prendre ce classificateur n'est pas une bonne idée pour les fluctuations mais pour la modélisation/approximation voyons ce qu'il se passe du point de vue de la décroissance.

Si f est uniformément Lipschitz-α alors $C > 0$

$$||f - \tilde{f}||_\infty \leq C\varepsilon \quad \text{avec} \quad \varepsilon = \max_x \min_i |x - x_i|$$

L'erreur entre la fonction et son approximation par morceaux est gouvernée par la distance maximale entre un point quelconque x et son plus proche voisin de l'échantillonnage. Le cas le pire c'est le cas où x est dans un trou ! Donc, il faut s'assurer que pour tout x la distance maximale par rapport à un échantillon ne soit pas trop grande. De combien d'échantillons n faut-il pour obtenir un ε?

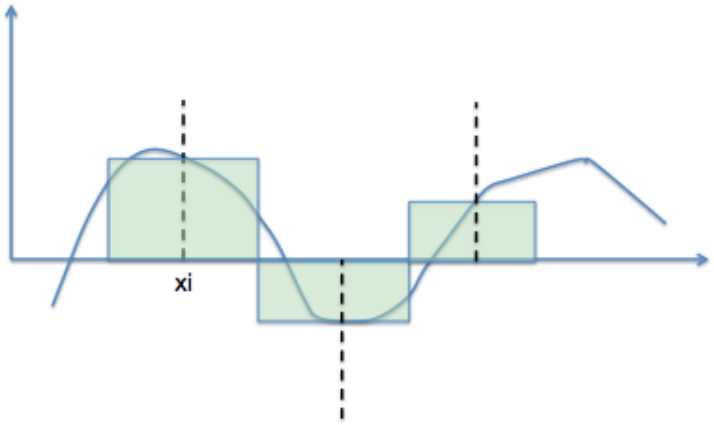

FIGURE 3 – Approximation de $f(x)$ par une fonction constante par morceaux.

Imaginons que $x \in \Omega$ avec $\Omega \in [0,1]^d$, les boules de rayon ε centrées sur les échantillons couvrent tout Ω, c'est-à-dire

$$\Omega \subset \bigcup_{i=1}^{n} \mathcal{B}_\varepsilon(x_i)$$

C'est un problème de codage: le log du nombre de boules donne le nombre de bits d'information.

— **Propriété**: la distribution optimale des boules dans $[0,1]^d$ satisfait

$$\frac{\sqrt{d}n^{-1/d}}{2} \geq \varepsilon \geq \frac{\sqrt{d}n^{-1/d}}{2}\sqrt{\frac{2}{\pi e}}\left[1 + O\left(\frac{d}{\log d}\right)\right]$$

et donc en prenant la seconde inégalité on a quelque chose comme

$$\boxed{n \gtrsim \varepsilon^{-d}\left[\frac{d}{2\pi e}\right]^{d/2}}$$

Mais on a une "double peine": le terme ε^{-d} qui explose dès lors que l'on veut contraindre l'erreur ε, et le terme constant qui varie selon $d^{d/2}$. **Donc, très rapidement même en basse dimension le nombre d'échantillons devient gigantesque si on suppose que la fonction est Lipschitz-α.** En terme de décroissance de l'erreur d'approximation on peut mettre le

résultat sous la forme

$$||f - \tilde{f}||_\infty \leq C\varepsilon \leq C\frac{\sqrt{d}n^{-1/d}}{2}$$

donc la décroissance est extrêmement lente en $n^{-1/d}$, et on a qq chose comme

$$R(f_I) \leq C\frac{\sqrt{d}}{2}(\log|\mathcal{H}|)^{-1/d}$$

c'est la malédiction de la grande dimension car l'espace est essentiellement « vide » en distance euclidienne!

Comment y échapper ? Trouver la bonne régularité des fonctions parmi une grande variabilité. Il faut regarder la nature des données et revenir à du traitement du signal avant de faire du ML.

On a x, y et f dans $y = f(x)$.La régularité de f pour $x \in \Omega$: c'est de l'apprentissage supervisé. Pour Ω, on a dit que son volume est 1 mais si mes données sont dans un volume/hypersurface de dimension beaucoup plus petite (c'est de la **feature selection**)! Quelle est la **régularité de** $x(u)$. Ex. à l'intérieure d'un image il y a des (ir)régularités qu'il faut capturer. On va commencer par de la **basse dimension**:

— faire la différence entre le cas **linéaire et le non-linéaire**; et non-linéaire ne veut pas dire compliqué, mais on ne fait du non-linéaire que si on en a besoin!
— introduire **la parcimonie** (**sparse vector**) qui est une notion duale de **régularité**.

4. Malédiction de la grande dimensions (Partie 2)

Usage du **Traitement du signal** (étude de représentation) pour faire de la réduction de dimensionalité (linéaire) et parcimonie (non-linéaire).

4.1 Les bases orthonormales: linéaire/non-linéaire (introduction)

Dans une base ortho-normale $\mathcal{B} = \{g_m\}_{1\leq m\leq d}$, pour représenter un signal $x(u)$ en dimension d:

$$x = \sum_m \langle x, g_m\rangle g_m$$

On change la représentation x_i que sont les composantes de x initiales, en introduisant les produits scalaires de x dans la nouvelle base. La question est de savoir si dans cette nouvelle base on peut ne garder que quelques coefficients tout en gardant une bonne approximation du signal. Dans le cas linéaire on va pouvoir les ordonner. Soit

$$x_M = \sum_{m=1}^{M} \langle x, g_m \rangle g_m$$

l'erreur faite est

$$\varepsilon_M = ||x - x_M||^2 = \sum_{k=M+1}^{d} |\langle x, g_m \rangle|^2$$

Comment varie la décroissance des coefficients en fonction de M?

Si

$$|\langle x, g_k \rangle| \leq Ck^{-\alpha} \quad (\alpha > 1/2)$$

alors

$$\varepsilon_M < \frac{C^2}{1-2\alpha} M^{1-2\alpha}$$

(on démontre par majoration d'une somme par l'intégrale associée et on pousse la borne supérieure à l'infinie)

Donc si les coefficients ont une décroissance rapide (α grand) alors l'erreur est d'autant meilleure. On vient de faire un raisonnement similaire que celui effectué avec l'algorithme d'apprentissage mais au lieu de travailler sur la fonction f, ici on travaille sur la représentation des données. La question devient quelle est la meilleure base qui garantie la plus grande décroissance?

Quand on n'a pas de connaissance *a priori* sur la représentation du signal dans **le cas linéaire**, on montre (montrera) que **la meilleure base est celle de Karhunen-Loève** ou en **composantes principales** (opérateur de covariance diagonalisé) (voir figure 4). Cependant dés qu'on a des informations sur les données on peut aller plus loin. Par exemple, si on a une **invariance par translation** (image, son) (pas de zéro absolue) alors la meilleure base est celle de **Fourier**. Mais par le théorème de Shannon il y a une correspondance entre Fourrier et un échantillonnage uniforme ce qui convient bien que si les données non pas de singularité. A ce moment là la base de Fourier n'est pas optimale car on sent bien qu'il

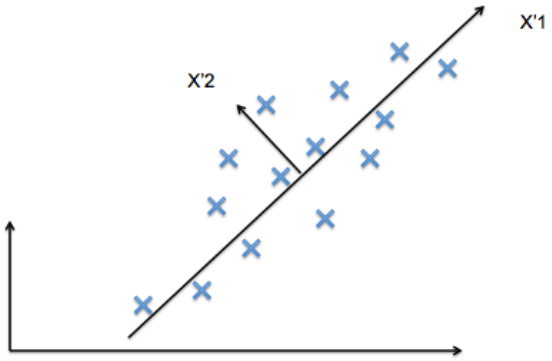

FIGURE 4 – Composantes principales d'un nuage d'échantillons.

faut faire un échantillonnage plus fin prés des singularités (il faudra être **adaptatif**). C'est un processus non-linéaire car les échantillonnages adaptatifs sur f et g ne conviennent pas à celui de $f+g$.

Donc en non-linéaire peut-on faire mieux? Reprenons la décomposition de x dans la nouvelle base orthonormale:

$$x = \sum_k \langle x, g_k \rangle g_k$$

et maintenant, on définit l'approximation en prenant M coefficients mais pas forcément les M premiers (car ça va dépendre du signal x) Donc

$$x_M = \sum_{k \in I_M} \langle x, g_k \rangle g_k$$

et

$$\varepsilon_M = ||x - x_M||^2 = \sum_{k \notin I_M} |\langle x, g_k \rangle|^2$$

Et on sélectionne les indices pour lesquels les coefficients sont les M plus grands:

$$I_M = \{k / |\langle x, g_k \rangle| \geq T_M\}$$

et **le seuil T_M (facile à mettre en oeuvre) va définir une approximation qui va s'adapter au signal ou dit autrement à la régularité de la fonction.** Ça se passe comme cela dans les bases d'**Ondelettes**. Notons qu'un neurone implémente ce seuillage de coefficients non-

linéaires.

4.2 Débruitage

Si on note x le signal de base et B le bruit, ce qu'on récupère en fait c'est un signal bruité Z tel que (gde lettre = aléatoire)

$$Z(u) = x(u) + B(u)$$

On va construire un **modèle déterministe** de x et un **modèle aléatoire** de B. En effet il est assez simple d'avoir des représentations aléatoires du bruit (tout en faisant qq hypothèses néanmoins) par contre par nature le signal est plus compliqué et contrairement à l'idée reçue un modèle « déterministe » est plus simple qu'un modèle stochastique. En effet, pour un modèle déterministe, on dit que $x \in \Omega$ (ex. niveau de gris d'une image) par contre le modèle stochastique (probabiliste) complète en disant qu'il y a des régions de Ω plus ou moins peuplées (nb. pour un bruit on imagine des gaussiennes assez souvent donc représentation simple).

Donc éliminer le bruit revient à trouver un opérateur L tel que mon estimateur de X soit

$$\tilde{X}(u) = LZ(u)$$

et on veut minimiser (mean square error: MSE)

$$R = \underset{B}{E}||x - LZ||^2$$

Deux approches:

— pour modéliser le **signal**: un modèle **linéaire** signifie que x est concentré sur un **hyperplan** (H)

— pour le bruit: un modèle gaussien (bruit blanc) qui peuple donc tout Ω et donc il « suffit » d'éliminer toutes les composantes du bruit en dehors de H par une projection (voir figure 5) [2].

Soit donc un modèle gaussien du bruit (**Bruit Blanc Gaussien ou bbg**):

2. Attention: classiquement en stat. on fait un modèle gaussien de x, et on abouti à ce que le meilleur estimateur est bien l'estimateur linéaire. Mais ici on prend un modèle déterministe de x, donc moins contraint, et l'on va voir que le non-linéaire peut être très efficace dans certaines circonstances.

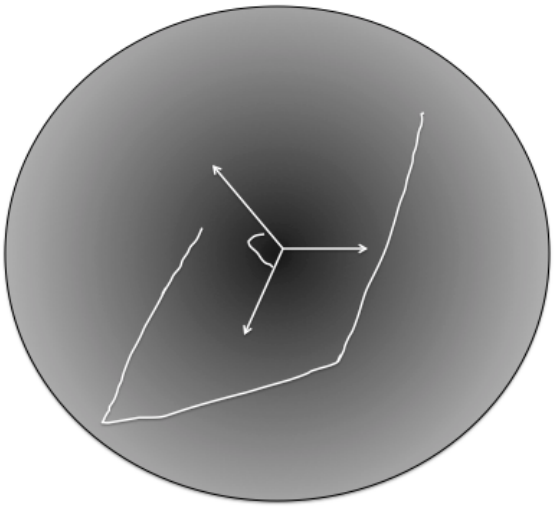

FIGURE 5 – Le signal bruité $Z(u)$ est composé d'un signal clair $x(u)$ évoluant dans un hyperplan, et d'un bruit $B(u)$. Il faut pouvoir réaliser une projection de Z sur l'hyperplan pour retrouver $x(u)$.

— **Moyenne nulle**: $E[B(u)] = 0 \ \forall u$ (ou on peut avoir enlevé la moyenne)
— **Composantes non-corrélées** $E[B(u)B(u')^*] = 0 \, \forall u \neq u'$, et $u = u'$, alors $E[|B(u)|^2] = \sigma^2$.

Donc la densité de probabilité d'obtenir un bruit b est

$$p(b) = c \ \exp\left(-\frac{1}{2} b^T \mathbf{\Sigma}^{-1} b\right) = c \ e^{-|b|^2/(2\sigma^2)}$$

avec $\mathbf{\Sigma} = \sigma^2 I$ la matrice de covariance.

Un bruit blanc reste blanc dans toutes les bases (il est isotrope), mettons une base $\{g_k\}_k$ de Ω

$$\langle Z, g_k \rangle = \langle x, g_k \rangle + \langle B, g_k \rangle$$

Si B est b.b.g

$$E(\langle B, g_k \rangle, \langle B, g'_k \rangle^*) = \sigma^2 \delta_{k,k'}$$

Cela se démontre en développant le produit scalaires

$$\langle B, g_k \rangle = \sum_u B(u) g_k^*(u)$$

$B(u)$ étant des variables aléatoires et $g_k^*(u)$ des constantes. Donc

$$E\left[\sum_{u,u'} B(u)B^*(u')g_k^*(u)g_{k'}(u')\right] = \sum_{u,u'} E[B(u)B^*(u')]g_k^*(u)g_{k'}(u')$$
$$= \sigma^2 g_k^*(u)g_{k'}(u') = \sigma^2\delta_{k,k'}$$

La seule difficulté est de savoir qu'est-ce qui est aléatoire et ce qu'il ne l'est pas.

Donc, prenons l'équivalent de l'opérateur linéaire de projection pour lequel on ne garde que M coefficients, c'est-à-dire que notre approximation:

$$\tilde{X} = Proj_{V_M} Z$$

et on veut minimiser

$$R = \underset{B}{E}||Proj_{V_M} Z - x||^2$$

Simplement

$$||x - P_{V_M} Z||^2 = ||(x - P_{V_M} x) - P_{V_M} B||^2 = ||(x - P_{V_M} x)||^2 + ||P_{V_M} B||^2$$

car $x - P_{V_M} x$ est un vecteur orthogonal à V_M alors que $P_{V_M} B$ est un vecteur de V_M par définition. Si on prend maintenant, l'espérance **par rapport au bruit** alors le premier terme est une constante, et reste à calculer le second $\underset{B}{E}[||P_{V_M} B||^2]$. Or

$$P_{V_M} B = \sum_{k=1}^{M} \langle B, g_k\rangle g_k$$

donc

$$\underset{B}{E}[||P_{V_M} B||^2] = \sum_{k=1}^{M} \underset{B}{E}[|\langle B, g_k\rangle|^2] = M\sigma^2$$

Le résultat donne que le risque se compose de 2 termes: une erreur d'approximation et un terme de fluctuations:

$$\boxed{R = ||x - Proj_{V_M} x||^2 + M\sigma^2}$$

Comment peut-on avoir l'optimum? si M = d, le premier terme est nul mais le second

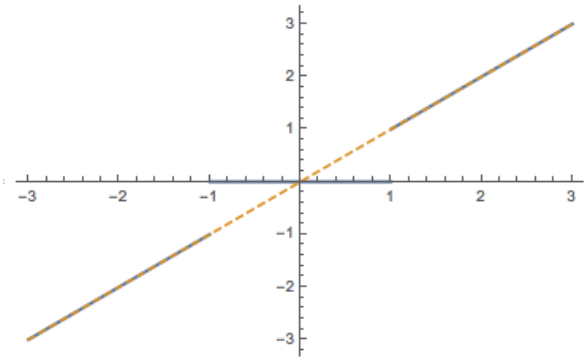

FIGURE 6 – Fonction de seuillage $\rho_\alpha(x)$, ici avec $\alpha = 1$.

est maximal et vice-versa. **Donc on va choisir le M qui réalise une balance entre les 2 termes**, et c'est la **décroissance du biais** qui va faire que M sera plus ou moins petit entre 2 modèles linéaires.

Mais si le signal a des discontinuités, on doit s'adapter au signal avec des algorithmes non-linéaires. Donc comme dans le cas d'approximation de fonction, on va bien choisir l'espace V_M en seuillant les coefficients.

$$LZ = \sum_{k \in I_M} \langle Z, g_k \rangle g_k = \sum_{k \in I_M} \rho_T(\langle Z, g_k \rangle) g_k$$

avec le seuil T tel que

$$T = \underset{1 \leq k \leq d}{max} |\langle B, g_k \rangle| \approx \sigma \sqrt{2 \log d}$$

et la fonction de seuillage définie par (figure 6):

$$\rho_\alpha(x) = \begin{cases} 0 \ si \ |x| < \alpha \\ x \ si \ |x| \geq \alpha \end{cases}$$

A quelle condition ce seuillage est effectif? Ça ira si la représentation (base) est la plus parcimonieuse possible (bien sûre si on connaissait x, on aurait un seul coefficient non nul, cependant on ne connait pas x !). **On veut que l'énergie du signal soit concentrée sur le petit nombre de coefficients, c'est donc un problème d'approximation du signal.** Pour un problème de compression, c'est la même chose: M est le nombre de bits, et on

veut l'erreur la plus petite possible.

5. Analyse de Fourrier

5.1 Rappels

On était parti sur **l'apprentissage supervisé** où la réponse à la question est unique à savoir $y = f(x)$, et on entraîne un algorithme h avec n échantillons étiquetés $\{x_i, y_i\}$. En fait on cherche à approximer f qui appartient à une famille de fonctions de régularité donnée ($f \in \mathcal{C}$) par un algorithme/fonction dans la classe $\mathcal{H}$. **Si on ne connait rien sur la distribution de probabilité** (X, Y) **on a vu que si on veut contrôler l'erreur (pessimiste)**

$$\sup_{f \in \mathcal{C}} \min_{h \in \mathcal{H}} ||f - h||_\infty = \sup_{x \in \mathbb{R}^d} |f(x - h(x))|$$

abouti à la **malédiction de la dimension** car $n \sim \varepsilon^{-d}$ avec potentiellement d très grand. Il faut donc **injecter du savoir sur les données**: traitement du signal et apprentissage non supervisé. Les données appartiennent à un sous-ensemble de Ω de dimension beaucoup plus petite. On a 2 grandes classes de façon de représenter des données: **cas linéaire et le cas non-linéaire**. Si on décompose x selon une **base orthonormale** de Ω, alors

$$x(u) = \sum_{k=1}^{d} \langle x, g_k \rangle g_k(u)$$

et:

— **dans le cas linéaire**: on tronque la somme au M-premiers coefficients (ex. les basses fréquences en Fourier); ainsi une approximation $\tilde{x}_M$ est un **projection** de x sur un ensemble de dimension M et l'erreur induite n'est autre que

$$\varepsilon_M = \sum_{k=M+1}^{d} |\langle x, g_k \rangle|^2$$

et la vitesse de décroissance des coefficients va conditionner la valeur de M. L'échantillonnage de f (cf. les positions des x_i qui donnent $f(x_i)$) sont régulièrement espacés **indépendamment de la fonction** f. Donc si la fonction sous-jacente est régulière

tout va bien, si par contre elle présente des singularités alors l'échantillonnage introduit des erreurs de biais.

— **dans le cas non-linéaire**: en fixant un seuil T_M qui réduit à M le nombre de coefficients comme dans le cas linéaire, **il s'adapte au signal** car dans ce cas l'échantillonnage se **concentre aux singularités de** f et réduit alors le terme de biais.

5.2 Analyse Harmonique (Fourier)

On va revisiter la transformation Fourier car elle se retrouve partout, certes en traitement du signal associée à la réduction de dimension, mais aussi quand sur une variété à décrire on veut représenter des données par des fonctions oscillantes. Ici on la reprend à 1 dimension et on va aborder plusieurs thèmes, entres autres **la régularité associée à la parcimonie**.

5.2.1 La convolution: opérateur covariant par translation

La TF est profondément liée à l'invariance par translation: quand on veut décomposer un opérateur (ex. le Laplacien) qui respecte l'invariance par translation, on va le diagonaliser sur la base de Fourier. Quels sont les opérateurs qui commutent avec la translation (covariants par translation) ?

$$L(x_\tau(u)) = L[x(u-\tau)] = (Lx)(u-\tau)$$

Dans le cas discret, on a

$$\begin{aligned} Lx(u) &= L[\sum_v x(v)\delta(u-v)] = \sum_v x(v)L[\delta(u-v)] \\ &= \sum_v x(v)h(u-v) = \sum_v x(u-v)h(v) = (x*h)(u) = (h*x)(u) \end{aligned}$$

avec h est la **réponse impulsionnelle** de L, c-a-d $L\delta = h$ avec δ l'opérateur de Dirac. La première égalité est la décomposition de $x(u)$ avec δ, la seconde est due à la linéarité de l'opérateur L, la troisième c'est la covariance de L vis-à-vis de la translation, enfin la dernière s'obtient par changement de variable. La notation « $*$ » représente la **convolution**.

Dans le cas continu, on procède de la même manière avec la masse/distribution de Dirac:

$$\int_{-\infty}^{+\infty} x(u)\delta(u)du = u(0)$$

pour des fonctions $x(u)$ ayant des propriétés de régularité. Voici donc un résumé:

Analogique	Discret
$x(u), u \in \mathbb{R}$	$u \in \mathbb{Z}$
$x(u) = \int_{-\infty}^{+\infty} x(v)\delta(u-v)dv$	$x(u) = \sum_{v=-\infty}^{+\infty} x(v)\delta(u-v)$
$Lx(u) = \int x(v)h(u-v)dv = (x*h)(u)$	$Lx(u) = \sum_v x(v)h(u-v) = (x*h)(u)$

L'opérateur linéaire qui commute avec la translation (soit continue soit pas sauts entiers) est la convolution. Et cela se généralise à n'importe quelle translation. Peut-on diagonaliser ces opérateurs de convolution et dans quelle base?

5.2.2 La base de Fourier

Soit la fonction $e_\omega(u) = e^{i\omega u}$ c'est un vecteur propre de la convolution

$$(e_\omega * h)(u) = \sum_v e^{i\omega(u-v)}h(v) = e^{i\omega u}\sum_v e^{-i\omega v}h(v) = e_\omega(u)\hat{h}(\omega)$$

avec

$$\hat{h}(\omega) = \sum_v e^{-i\omega v}h(v)$$

la **Transfornée de Fourier** (TF) de h. En continue on définit d'une manière similaire [3]

$$\hat{h}(\omega) = \int dv\ e^{-i\omega v}h(v)$$

On définit la **Base de Fourier** selon le cas « analogique » ou « discret » de la façon suivante, tout en gardant à l'esprit que l'on pense en « continue » (ex. les notions de régularité) et on

3. NDJE: il y a une constante de normalisation qui dépend de convention.

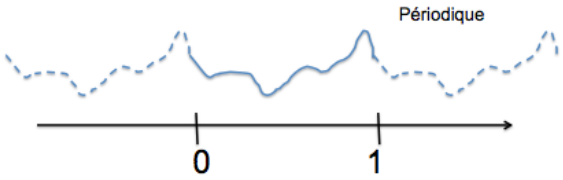

FIGURE 7 – "Périodisation" d'une fonction définie sur l'intervalle $[0,1]$.

calcule en « discret » (cf. échantillonnage et qd d augmente ça converge vers l'analogique), d'où l'intérêt de voir les 2 formulations.

— **Analogique**:

On considère les fonctions $L^2[0,1]$: ensemble de fnt de carré sommable, cf. d'énergie finie sur le support $[0,1]$ (ce n'est pas restrictif et la notion de support compact est adaptée au cas pratique). On impose une périodicité comme sur la figure 7: La base orthonormale de Fourier est alors définie part

$$\boxed{g_k(u) = e^{i2\pi\ ku};\ k \in \mathbb{Z}, u \in [0,1]}$$

— **Discret**:

Dans ce cas on prend d échantillons et on impose aussi une périodicité sur $\mathbb{Z}$ en dehors de $0, \ldots, d-1$. La base de Fourier s'écrit

$$\boxed{g_k(u) = e^{i2\pi k\ u/d}; 0 \leq k < d; 0 \leq u < d}$$

Maintenant dans les 2 cas de figures x se projeté dans la base g_k avec le produit scalaire associé. Exemple en continue (analogique):

$$x(u) = \sum_k \frac{\langle x, g_k \rangle}{||g_k||^2} g_k(u); \quad ||x||^2 = \sum_k |\langle x, g_k \rangle|^2$$

Le coefficient de Fourier est égal à:

$$\langle x, g_k \rangle = \int_0^1 x(u) e^{-i2\pi\ ku} = \hat{x}(\omega = 2\pi k) \equiv \hat{x}(k)$$

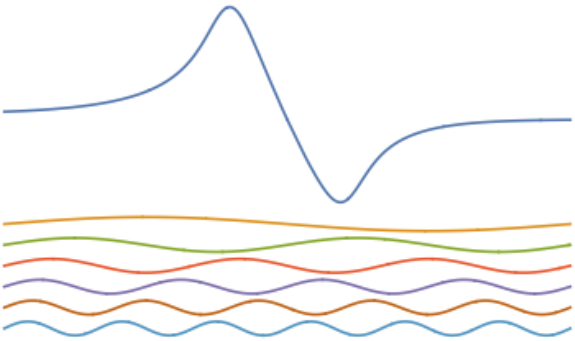

FIGURE 8 – Décomposition d'une fonction en sinusoïdes.

Donc en analogique (à support compact) la reconstruction/synthèse du signal s'écrit

$$x(u) = \sum_{\omega = 2\pi k \in \mathbb{Z}} \hat{x}(\omega)\ e^{i\omega u}$$

Dans le cas discret avec la normalisation $||g_k(u)||^2 = d$, et il vient

$$x(u) = \frac{1}{d} \sum_{k=1}^{d} \hat{x}\left(\frac{2\pi k}{d}\right) e^{i2\pi \frac{ku}{d}}$$

Notons dans le cas discret, qu'il nous faut d coefficients à calculer de Fourier pour d composantes de x donc d^2 opérations sont nécessaires. Fort heureusement on a trouver **des algorithmes (FFT) nécessitants que** $d \log d$ **opérations**. C'est un point fondamental et qui se retrouve dans la science des données qui lie math et algorithme.

La décomposition en sinusoïdes de n'importe quelle fonction à support compact (d'énergie finie) n'est absolument pas intuitif. Par exemple il faut imaginer décomposer la fonction bleue de la figure 8 en pondérant des sinusoïdes dont les 6 premières sont dessinées. La variabilité très rapide locale fait intervenir une sinusoïde à grande fréquence, mais ailleurs les oscillations rapides doivent être compenser exactement pour laisser place à une fonction à variations douces représentée par des sinusoïdes de basses fréquences. Si on voit cela sur les maths, intuitivement c'est plus difficile à première vue.

5.2.3 Filtrage/Convolution et le tandem approximation-régularité

Les coefficients $\hat{x}(\omega)$ représentent les différents « poids » de pondérations pour reconstruire le signal, et donc **la décroissance des coefficients de Fourier est liée à la régularité de la fonction**. On veut ne garder que M coefficients et avoir une petite erreur d'approximation. Avec $\int |x(u)|du < \infty$ idem pour h, alors

$$\boxed{z(u) = (x * h)(u) \Leftrightarrow \hat{z}(\omega) = \hat{x}(\omega)\hat{h}(\omega)}$$

On se rappelle que la convolution $(x * h)$ n'est autre que l'application de l'opérateur L linéaire covariant par translation, c'est-à-dire $z = Lx$. L'équivalence ci-dessus se démontre facilement en décomposant x sur la base de Fourier et par l'application de L sur les vecteur de la base.

Filtrage et Convolution sont des notions identiques et le jargon change selon les domaines. Par exemple, on veut enlever les hautes fréquences pour lisser un signal. Soit le **filtre passe bas**

$$\hat{h}_M(\omega) = \begin{cases} 1 & -\frac{M}{2} \leq \omega \leq \frac{M}{2} \\ 0 & ailleurs \end{cases}$$

dans l'espace réel il va correspondre à une convolution avec $h_M(u)$ qui est un **sinus cardinal** (périodisé sur $[0, 1]$).

Donc, dans le cadre d'approximation de fonction, on se rappelle que l'on veut projeter le signal x sur une base orthonormale (ici la base de Fourier) et ne garder que les M-premiers coefficients (ici filtrer les hautes fréquences). On définit ainsi le projecteur sur l'espace V_M (sous-espace de Ω) à l'aide de la convolution par le filtre passe bas:

$$Proj_{V_M} x = x * h_M$$

On sait alors que si la décroissance des coefficients de Fourier de x est suffisamment rapide alors la convolution par h_M sera une bonne approximation. Et cette décroissance est liée à la régularité de x. Qu'en est-il exactement? La dérivée de $x(u)$, notée $x'(u)$ commute avec la translation, ainsi il est facile de montrer (intégration par partie) que

$$\widehat{x'}(\omega) = i\omega\ \hat{x}(\omega)$$

et que pour la k-ieme dérivée:

$$\widehat{x^{(k)}}(\omega) = (i\omega)^k \ \hat{x}(\omega)$$

Une **notion de régularité** peut être introduite par le fait que les **dérivées de la fonction n'explosent pas**, c'est-à-dire que

$$||x^{(q)}||^2 = \int_0^1 \left|\frac{d^q x}{du^q}\right|^2 du < \infty$$

Cela se traduit donc (Plancherel) dans le domaine de Fourier par le fait que

$$\sum_{\omega=2\pi k\in\mathbb{Z}} |\omega|^{2q}|\hat{x}(\omega)|^2 < \infty$$

Pour que cela soit vrai il faut imposer une décroissance de $\hat{x}(\omega)$ en fonction de ω ([4]):

$$|\hat{x}(\omega)| = o(|\omega|^{-q}) = o(|k|^{-q});\ \omega = 2\pi k$$

et q par extension peut être un nombre $\alpha \in \mathbb{R}$. Ce sont des fonctions dans des espaces de Sobolev dérivables α-fois.

Donc la décroissance des coefficients de Fourier est liée à la notion de régularité à travers la notion de dérivabilité au sens de Sobolev.

Du théorème d'approximation, on se souvient que si les coefficients dans une base ($\alpha > 1/2$) satisfont

$$|\langle x, g_k\rangle| = O(k^{-\alpha}) \text{ alors } \varepsilon_M^2 = ||x - x_M||^2 = O(M^{1-2\alpha})$$

Cela se traduit (en continue) $g_k = e^{i2\pi ku}$ par le fait que les coefficients de Fourier d'une fonction α fois dérivable au sens de Sobolev,

$$|\langle x, g_k\rangle| = |\hat{x}(\omega)| = o(|k|^{-\alpha})$$

et donc l'erreur est contrainte par la relation ci-dessus et même le grand « O » peut être remplacer par un petit ¨o¨ ce qui est un peu plus fort. Donc, dés qu'on a une régularité, on

4. le petit "o(a)" veut dire négligeable devant « a »

a une approximation possible et vice-versa. Par contre, dés que l'on a une discontinuité, la régularité de Soboloev n'est pas suffisante, le filtrage dans la base de Fourier introduit des erreurs (biais) et il faudra changer de type de régularité.

Théorème d'échantillonnage: Soit l'ensemble des fonctions dont les coefficients de Fourier à hautes fréquences sont nuls:

$$V_M = \left\{ x/\text{support de } \hat{h}(\omega) \in [-M/2, M/2] \right\}$$

La projection de x sur V_M est telle que

$$x_M = Proj_{V_M} x = x * h_M; \quad \hat{h}(\omega) = \mathbb{I}_{[-M/2,M/2]}(\omega)$$

Si on suppose que $x \in L^2[\mathbb{R}]$ (c'est juste pour simplifier l'expression de h, sinon il faut périodiser le sinus cardinal), alors

$$h_M(u) = \frac{\sin \pi x/T}{\pi x/T}; \quad \text{avec} \quad T = 2\pi/M$$

Soit $h_M(u - nT) = h_{M,nT}(u)$ avec $n \in \mathbb{Z}$, alors ses coefficients de Fourier sont

$$\widehat{h_{M,nT}}(\omega) = \hat{h}_M(\omega) e^{-i2\pi \ n \ (\omega/M)}$$

avec $\hat{h}_M(\omega) = 1$ sur le support de ω est $[-M/2, M/2]$; donc on a une nouvelle base orthonormale (voir formule de Poisson pour démontrer que tte fonction de V_M se décompose selon une somme de $h_{M,nT}(u)$)

$$\{h_M(u - nT)\}_{n \in \mathbb{Z}}$$

Donc $\forall x \in V_M$ on obtient la décomposition

$$\boxed{\begin{aligned} x(u) &= \sum_{n \in \mathbb{Z}} c_n h_M(u - nT) \\ &= \sum_{n \in \mathbb{Z}} x(nT) h_M(u - nT) \end{aligned}}$$

la seconde égalité venant de ce que $h_M(mT - nT) = \delta_{m,n}$. **Ainsi à partir des échantillons** $x(nT)$ **on peut reconstruire le signal** $x(u)$ **à l'aide des interpolations données par** $h_M(u - nT)$.

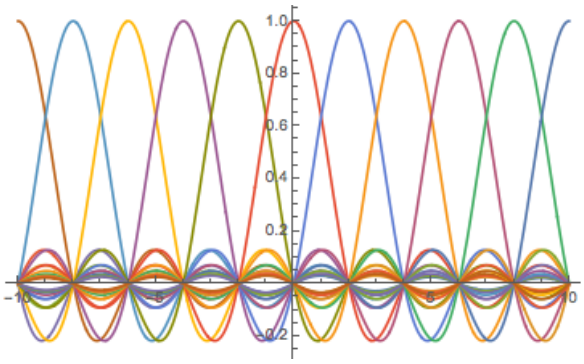

FIGURE 9 – Séries de sinus cardinal $h_{M,nT}(x)$, $T = 2\pi/M$.

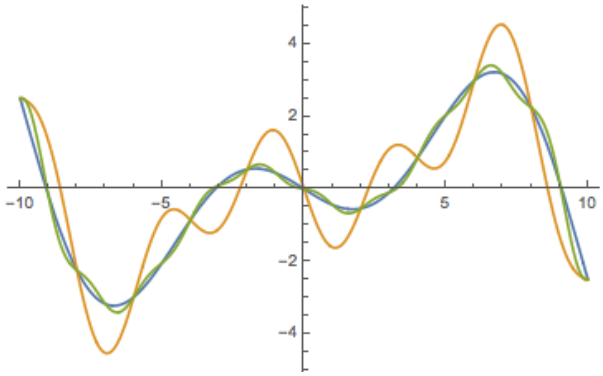

FIGURE 10 – Approximation de la fonction $x(u)$ (bleu) par deux fonctions issues de l'échantillonnage avec $T = 2$ en jaune et $T = 1$ en vert.

Par exemple, avec $T = 2$ et $n = -10, -9, \ldots, 9, 10$ les $h_M(u-nT)$ se présentent comme un sinus cardinal translaté (voir figure 9). Plus T diminue plus les sinus cardinaux sont resserrés est l'approximation de $x(u)$ est d'autant meilleure, comme on peut le constater sur l'exemple de la figure 10, où on a $T = 2$ en jaune et $T = 1$ en vert, alors que $x(u)$ est en bleu. Donc, pour des fonctions régulières uniformes pas de problème, par contre **dés qu'il y a une discontinuité, on ne peut bénéficier (en termes de nombre d'échantillons) de la continuité en dehors de celle-ci**. C'est LA grosse limite de l'analyse (régularité) de Fourier. On verra d'autre type de régularités qui s'adaptent au signal.

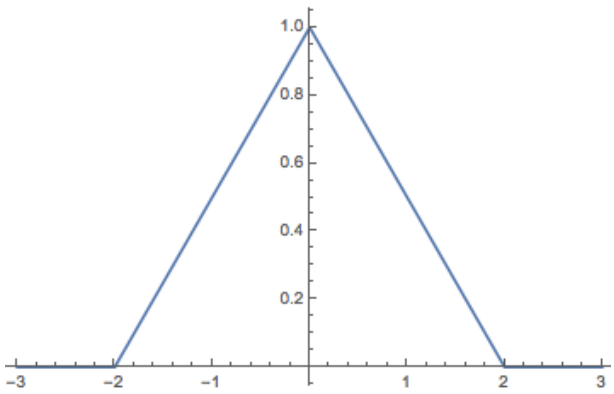

FIGURE 11 – Fonction triangle.

5.2.4 Interpolation et th. d'échantillonage

Le théorème d'échantillonnage est plus général, il faut le penser en termes de base d'interpolation associée. Soit $h(t) = 1$ sur le support $[-T/2, T/2]$ et prenons l'espace vectoriel engendré par la collection des translatées de h:

$$V_T = Vect\left\{h(t - nT)\right\}_{n\in\mathbb{Z}}$$

Le Th d'échantillonnage associé à h nous dit alors:

$$x \in V_T \Leftrightarrow x(t) = \sum_n x(nT)h(t - nT)$$

On voit donc que V_T est l'espace des **fonctions constantes par morceaux** de taille T.

Mais on peut faire mieux! Si on prend la fonction "triangle" $h(t)$ de la figure 11 avec un support $[-T, T]$ et la valeur en 0 égale à 1, alors V_T est l'espace des fonctions **splines linéaires** (ou affine par morceaux) qui donnent des approximations comme sur la figure 12. On peut calculer le filtre associé, ce qui donne la fonction sur la figure 13. Donc, on peut obtenir des approximations fines de fonctions régulières, mais c'est plus général que cela car on va vouloir s'adapter à la régularité de la fonction (ou à ses discontinuités) et donc vouloir obtenir un **théorème d'échantillonnage dans le cas non-linéaire: c'est l'analyse en Ondelettes.**

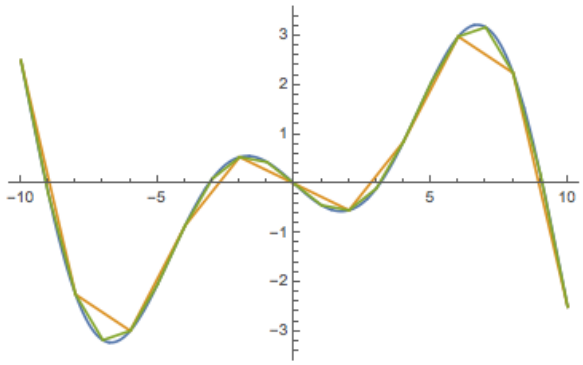

FIGURE 12 – Approximation d'une fonction $f(x)$ (bleu) par des fonctions affines par morceaux issues du théorème d'échantillonnage avec le fonction triangle.

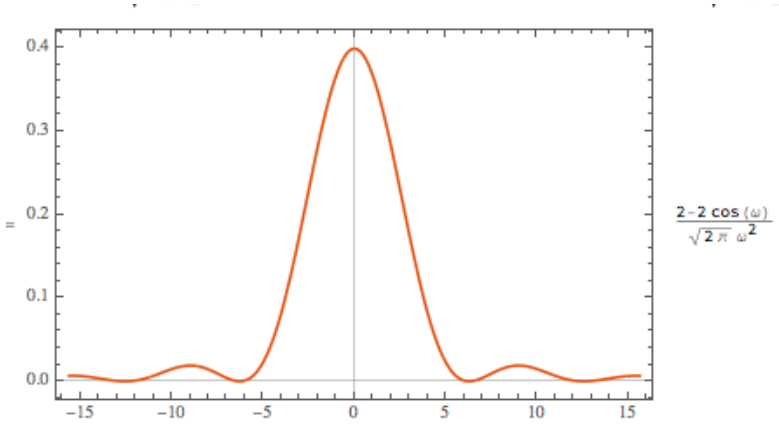

FIGURE 13 – Filtre associé à la fonction triangle.

TF	$\hat{f}(\omega) = \frac{1}{\sqrt{m}} \int f(x) e^{iq\omega x} dx$
TF Inverse	$f(x) = \frac{\sqrt{m}}{2\pi} \int \hat{f}(\omega) e^{-iq\omega x} dx$
Translation	$\widehat{f(x-h)}(\omega) = e^{iqwh} \hat{f}(\omega)$
Dilatation	$\widehat{\frac{1}{s} f(\frac{x}{s})}(\omega) = \hat{f}(s\omega)$
Moyenne	$\hat{f}(0) = \frac{1}{\sqrt{m}} \int f(x) \mathrm{d}x$
Plancherel	$\int f(x) g^*(x) \mathrm{d}x = \frac{m}{2\pi} \int \hat{f}(\omega) \hat{g}^*(\omega) \mathrm{d}\omega$
Parseval	$\int \lvert f(x) \rvert^2 \mathrm{d}x = \frac{m}{2\pi} \int \lvert \hat{f}(\omega) \rvert^2 \mathrm{d}\omega$
Dérivée	$\widehat{f^{(p)}(x)}(\omega) = (-iq\omega)^p \hat{f}(\omega)$
Convolution	$\widehat{f * g} = \sqrt{m} \hat{f}(\omega) \hat{g}(\omega)$
Multiplication par une phase	$\widehat{e^{i\xi x} f(x)}(\omega) = \hat{f}(\xi/q + \omega)$

TABLE 4 – Exemples des propriétés de la transformation de Fourier exprimées suivant une formulation générique en 1D. S. Mallat utilise $(m = 1, q = -1)$ qui correspond à la convention également $(a = 1, b = -1)$ de Mathematica; mais dans la littérature on peut trouver des formulations avec $m = 1, 2\pi$ et $q = \pm 1, \pm 2\pi$.

5.2.5 Conventions de la transformée de Fourier (JE)

Avant de conclure ce chapitre, abordons un point pratique à savoir les conventions d'écritures. En effet, à la fois dans la littérature, mais aussi dans l'usage de librairie informatique, il est toujours bon de savoir quelle est la convention utilisée. Dans le tableau 4 la TF dépend de 2 paramètres (m, q) qui influent sur l'écriture des formules.

6. Analyse en Ondelettes

On se rappelle que dans le cadre de l'apprentissage, on veut faire de la réduction de dimensionnalité, donc comprendre les formes de régularité du signal. Dans le cas de Fourier en tronquant la représentation aux M-premiers coefficients, on introduit implicitement une régularité uniforme (locale). Ceci ne convient pas dès que le signal est régulier « par morceaux » avec aux jointures des singularités.

C'est dans ce cadre d'extension que sont introduites les **Ondelettes** dont le **support est localisé à la fois en temps et en fréquence**. Dennis Gabor (Prix Nodel pour l'holographie, 1971) introduit la représentation d'états cohérents en Mécanique Quantique en accord avec la relation d'incertitude $\Delta x \Delta p \geq h$. Puis D. Gabor est passé au traitement du son où la localité temps-fréquence est bien associée à la notion de note de musique (spectrogramme de Gabor). Ce type de représentation se généralise dans plusieurs domaines.

L'idée de base de l'ondelette est que c'est une sorte de « sinusoïde localisée » notée ψ. La notion de localité se rend par la condition

$$\boxed{\int_{-\infty}^{\infty} \psi(u)du = 0}$$

et pour s'adapter aux fonctions plus ou moins régulières, on opère 2 transformations:

— **une translation** ($b \in \mathbb{R}$)

— **et une dilatation** ($s \in \mathbb{R}^{+*}$)

que l'on applique à l'ondelette de base $\psi(u)$ pour donner la famille

$$\boxed{\psi_{s,b}(u) = \frac{1}{\sqrt{s}}\psi\left(\frac{u-b}{s}\right)}$$

Sur la figure 14 est montré l'effet du scaling sur une ondelette: en haut: $s = 1$, au milieu: $s = 2$ (basse fréquence), en bas: $s = 1/2$ (haute fréquence) [5].

5. NDJE: la représentation au tableau que fait S. Mallat est un peu trompeuse, on a l'impression qu'il dessine $\phi(x)$ que l'on appelle la « scaling function » des ondelettes centrée sur $x = 0$, et non $\psi(x)$.

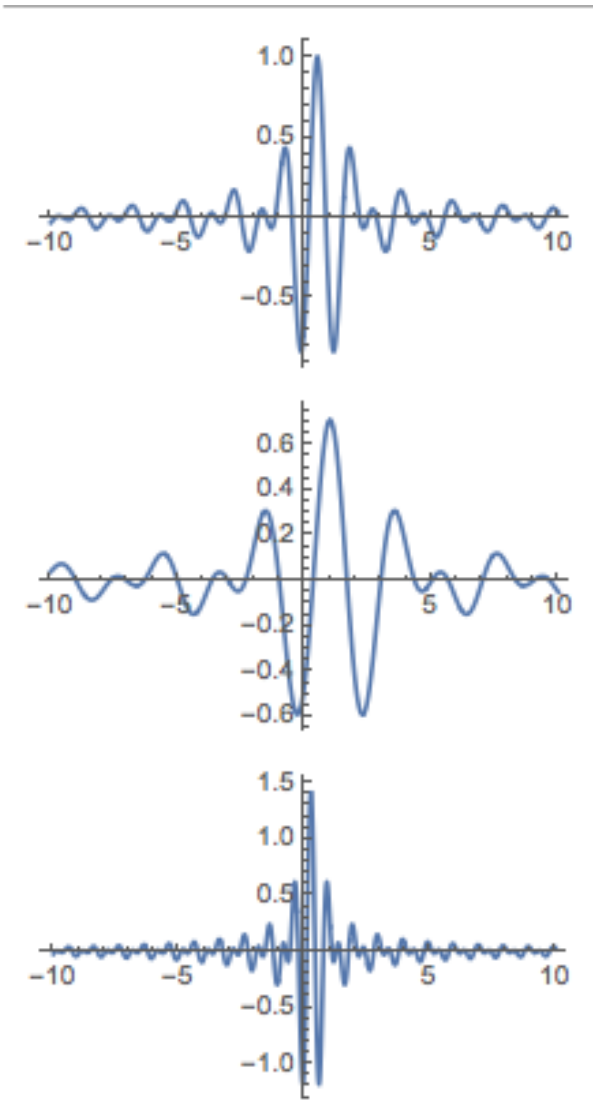

FIGURE 14 – (haut): ondelette de base; (milieu): $s = 2$ basse fréquence; (bas): $s = 1/2$ haute fréquence.

6.1 Transformée en ondelettes

Soit un signal $x(u)$, sa transformée en Ondelettes s'écrit comme

$$Wx(s,u) = \langle x, \psi_{s,u} \rangle = \int dv\ x(v) \frac{1}{\sqrt{s}} \psi^* \left(\frac{v-u}{s} \right) \equiv \int x(v) \bar{\psi}_s(u-v) dv$$

avec 1) on considère que ψ **est réelle** et 2) on définit $\bar{\psi}$ selon

$$\boxed{\bar{\psi}_s(u) \equiv \frac{1}{\sqrt{s}} \psi \left(-\frac{u}{s} \right)}$$

Donc, on voit que la **Transformé en Ondelettes est une convolution**:

$$\boxed{Wx(s,u) = (x * \bar{\psi}_s)(u)}$$

c'est donc un **filtrage** par $\bar{\psi}_s$.

La transformée de Fourier de l'ondelette est donnée par

$$\hat{\psi}(\omega) = \int \psi(t) e^{-i\omega t} dt$$

Comme par définition $\hat{\psi}(0) = 0$ et l'ondelette étant réelle alors

$$\hat{\bar{\psi}}_s(\omega) = \sqrt{s} \hat{\psi}^*(s\omega)$$

Le support en fréquence est de type « passe-bande » comme le montre le spectrogramme $|\hat{\psi}(\omega)|$ de l'ondelette sur la figure 15 (étant réelle le spectrogramme est symétrique par rapport à 0). Et si $s > 1$ le spectrogramme se déplace à *basses fréquences*, et à l'inverse si $s < 1$ il se déplace à *hautes fréquences* en cohérence avec le scaling de l'ondelette en fonction de s.

Pour les ondelettes, le support satisfait une condition d'Heisenberg comme en Mécanique Quantique, c'est-à-dire que

$$Surface \geq Cte = 1/4$$

Ce support se déforme donc selon l'axe des fréquences comme montré schématiquement

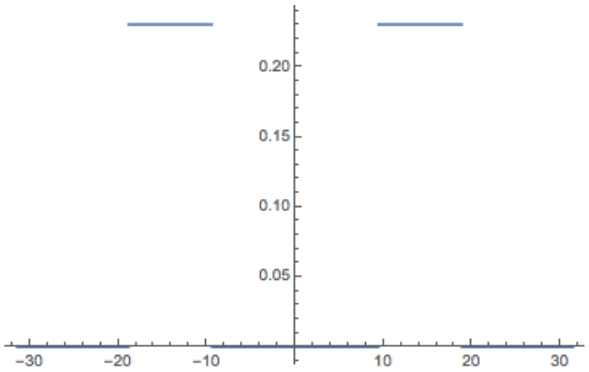

FIGURE 15 – Filtre typique d'une ondelette ψ.

sur la figure 16 quand on dilate l'ondelette; en translation selon l'axe des temps, la forme du support ne change pas.

6.2 Ondelettes et régularité/singularité de fonction

Reprenons la notion de régularité au sens de Lipschitz-α avec ici:

$$|x(v) - p_u(v)| \leq C_u |v - u|^\alpha \; \forall v \in \mathbb{R}$$

$p_u(v)$ est un polynôme au voisinage de u de degré $q = \lfloor \alpha \rfloor$ (entier juste plus petit que $\alpha \geq 0$). Si $\alpha = 0$ la fonction est bornée, si $\alpha \geq 1$ elle est dérivable, et pour $0 < \alpha < 1$ la fonction peut présenter un « pincement ». Tout ceci peut se schématiser comme sur la figure 17.

Prop. Si x est Lipschitz-α en u alors le coefficient d'ondelette (cf. le produit scalaire) satisfait la relation

$$\boxed{|\langle x, \psi_{s,u} \rangle| \leq C_u \; \beta \; s^{\alpha + \frac{1}{2}}}$$

si ψ est une ondelette à moments nuls [6], c'est-à-dire quelle oscille un peu plus pour ne *pas voir une régularité polynomiale*

$$\int \psi(t) t^k dt = 0; \quad k \leq q$$

6. NDJE: les ondelettes de Ingrid Daubechies ont cette propriété

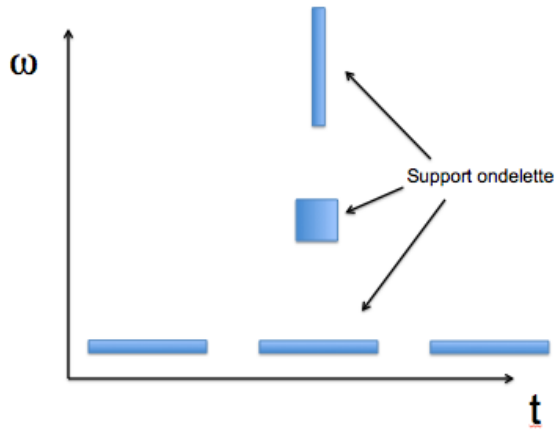

FIGURE 16 – Évolution des supports des ondelettes en fonction de la fréquence et du temps. La surface est satisfait une relation de type "relation d'incertitude d'Heisenberg".

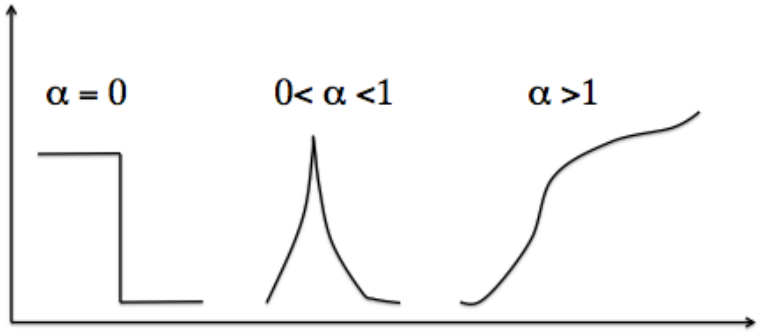

FIGURE 17 – Type de fonction Lipschitz-α.

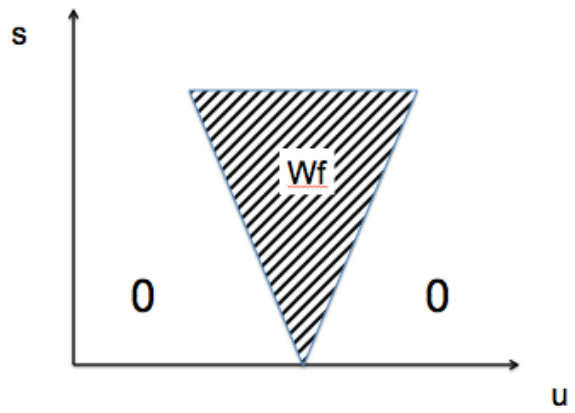

FIGURE 18 – La zone hachurée indique schématiquement l'évolution en fonction de l'échelle s du support des coefficients d'ondelettes non nuls.

(propriété qui se transmet pour les versions dilatées et translatées).

La proposition dit que si s tend vers 0 (petit support) et si α est proche de 1 (fonction dérivable) alors le coefficient d'ondelette tend vers 0. **Donc le nombre de coefficients diminues quand s tend vers 0** comme sur le schéma de la figure 18. Lors de la démonstration, on trouve la constante β qui ne dépend que de l'ondelette

$$\beta = \int |y|^{\alpha} |\psi(y)| dy$$

Donc, la régularité Lipschitz-α de la fonction est « imprimée » dans la décroissance des coefficients d'ondelette en fonction du changement d'échelle selon $s^{\alpha+1/2}$.

6.3 Base orthonormée d'ondelettes

D. Gabor dans les années 40 avait trouvé une famille de fonctions basée sur la forme

$$\psi_{\nu}^{Gabor}(t) \propto e^{-x^2/2} e^{i\nu x}$$

Ce sont des « paquets » d'ondes. Jean Morlet, ingénieur à ELF Aquitaine, à l'origine du vocable « ondelette », utilisait en fait la même famille (partie réelle uniquement) avec $\nu =$

$\sqrt{2/\log 2}$. Mais ces fonctions $\psi_{s,b}$ ne formaient pas une base orthonormale. Ce n'est que dans les années 1985 que **Yves Meyer** démontre l'existence de telles bases orthonormales contre toute attente. Il donne un nouvel élan au traitement du signal et l'usage des ondelettes s'est diversifié dans divers domaines des mathématiques pour classifier des « régularités ».

Pour construire une base orthonormale, dans un premier temps on va discrétiser les échelles. En effet, on a pas besoin de tous les coefficients d'ondelettes comme on peut le voir sur la figure 16 où le support des ondelettes a une étendue en s et donc il y a de la redondance à prendre s continue. **Il suffit de prendre des échelles dyadiques:** 2^j. En traitement du son on prend des valeurs intermédiaires.

Soit donc la famille

$$\boxed{\psi_j(u) = \frac{1}{\sqrt{2^j}}\psi\left(\frac{u}{2^j}\right) \quad \forall j \in \mathbb{Z}}$$

La question devient alors: est-ce que la donnée des coefficients d'ondelettes pour tous les j suffit à reconstruire le signal? Obtenir ces coefficients revient à faire des convolutions

$$Wx_j(u) \equiv \langle x, \psi_j\rangle(u) = (x * \overline{\psi_j})(u)$$

En fait la question revient à se poser la question de savoir si à partir de la transformée de Fourier des coefficients en ondelettes, peut-on reconstruire la transformée de Fourier du signal x ? Notons la propriété:

$$\widehat{\overline{\psi_j}}(\omega) = \sqrt{2^j}\hat{\psi}^*(2^j\omega)$$

Donc

$$\widehat{Wx_j}(\omega) = \hat{x}(\omega)\widehat{\overline{\psi_j}}(\omega) = \sqrt{2^j}\ \hat{x}(\omega)\ \hat{\psi}^*(2^j\omega)$$

Le produit $\hat{x}(\omega)\ \hat{\psi}^*(2^j\omega)$ peut se schématiser pour quelques valeurs de j selon la figure 19. Donc, pour retrouver $\hat{x}(\omega)$, il faut que $\forall\omega$ il y ait une ondelette qui donne $\hat{\psi}^*(2^j\omega)$ non nul. En d'autres termes, il faut un certain recouvrement.

Imposons la condition (Littlewood-Paley) supplémentaire de type « conservation de puissance » (*connu des années 30 avec la transformée de Fourier pour essayer de faire de*

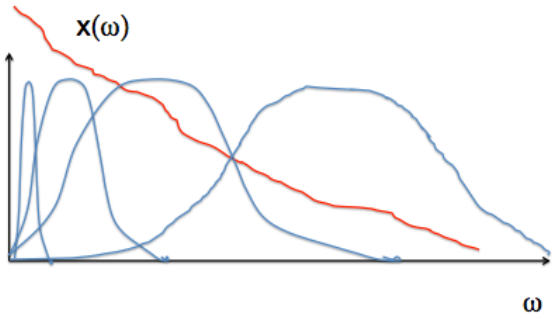

FIGURE 19 – Décroissance typique de la transformée de Fourier $\hat{x}(\omega)$ à comparer aux supports d'ondelettes.

la localisation):

$$\boxed{\forall \omega,\ \sum_{j\in\mathbb{Z}} |\hat{\psi}(2^j\omega)|^2 = 1}$$

Avec cette propriété, on va montrer que $x(u)$ se décompose selon

$$\boxed{x(u) = \sum_{j\in\mathbb{Z}} 2^{-j}(Wx_j * \psi_j)(u) = \sum_{j\in\mathbb{Z}} 2^{-j}[(x * \overline{\psi_j}) * \psi_j](u)}$$

En Fourier cela revient à dire que

$$\begin{aligned}
\hat{x}(\omega) &= \sum_j 2^{-j}\widehat{Wx_j}(\omega)\widehat{\psi_j}(\omega) \\
&= \sum_j 2^{-j}\sqrt{2^j}\ \hat{x}(\omega)\ \hat{\psi}^*(2^j\omega)\sqrt{2^j}\ \hat{\psi}(2^j\omega) \\
&= \hat{x}(\omega)\sum_j |\hat{\psi}(2^j\omega)|^2
\end{aligned}$$

ce qui est vrai d'après la conservation de la « puissance » imposée ci-dessus. On montre aussi que l'énergie de x est conservée

$$||x||^2 = \sum_{j\in\mathbb{Z}} 2^{-j}||Wx_j||^2$$

Maintenant, pour ce qui concerne la translation en u il ne faut pas laisser de trous

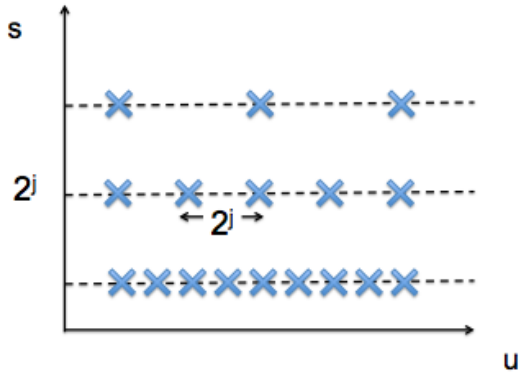

FIGURE 20 – Échantillonnage "optimal" par discrétisation dyadique de l'espace (s, u).

également. Si on se rapporte à la figure 16, on voit que pour les basses fréquences (petites dilatations: 2^{-j} avec j>0) le support est large donc on a pas besoin de faire un sampling fin, *a contrario* à haute fréquence le support s'affine et donc le sampling en u doit être plus fin. On définit alors pour pour $s = 2^j$, un sampling $u_n = n2^j$. Le schéma de la figure 20 résume le sampling dyadique.

La famille complète [7] d'ondelettes s'écrit donc à partir de l'ondelette de base ψ:

$$\boxed{\left\{\psi_{j,n}(u) = \psi_{2^j,2^j n}(u) = \frac{1}{\sqrt{2^j}}\psi\left(\frac{u - 2^j n}{2^j}\right)\right\}_{j,n\in\mathbb{Z}}}$$

Il se trouve que A. Haar en 1910 avait trouvé une base orthonormale de $L^2[0,1]$ (fonctions de carré sommable sur $[0,1]$) définit comme $\psi_{j,n}(u)$ à partir de la fonction ψ de la figure 21 (à laquelle il faut ajouter la fonction unité 1 à la collection de fonctions $\psi_{j,n}$). La transformée de Fourier de ψ est assez oscillante (voir figure 22) à cause des singularités en 0, 1/2 et 1 [8]. Cependant, il est assez simple de montrer que les $\psi_{j,n}^{Haar}$ sont orthogonales car soit les supports sont disjoints, soit le support de l'une est dans une partie constante

7. NDJE: juste une aparté. Il y a potentiellement un problème: quid de composante continue (et très basse fréquence) de x? avec uniquement des $\psi_{j,n}$ il faudrait une infinité d'ondelettes avec d'échelle 2^{-j} pour $j \to \infty$. S. Mallat a introduit la « scaling function » ϕ (qui peut servir à définir aussi ψ) qui n'est autre qu'un filtre passe-bas. Donc, on peut reconstruire un signal avec une série finie dilatée et translatée de ϕ et une série infinie de $\psi_{j,n}$. Voir la section en fin de ce cours.

8. NDJE: l'ondelette de Haar est un cas particulier des ondelettes de Daubechies notée **Db1**

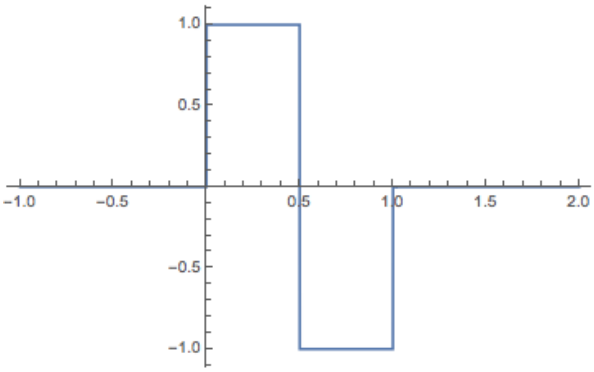

FIGURE 21 – Ondelette $\psi(x)$ de Haar.

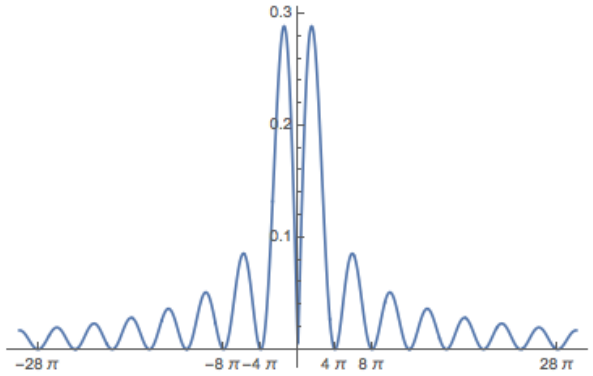

FIGURE 22 – Filtre associé à l'ondelette ψ de Haar.

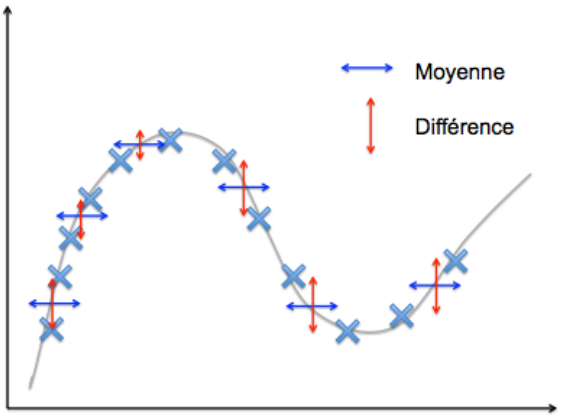

FIGURE 23 – Opération de la transformée en ondelettes effectuée par l'ondelette de Haar.

de l'autre, ce qui donne des produits scalaires trivialement nuls. On peut montrer que pour tout (n, j) tels que $0 \leq 2^j n \leq 1$, c-a-d $j \leq 0$ et $0 \leq n < 2^{-j}$ alors

$$||x(u) - \sum_{j=-J}^{0} \sum_{n=0}^{2^{-j}-1} \langle x, \psi_{j,n} \rangle \psi_{j,n}|| \underset{J \to \infty}{\longrightarrow} 0$$

(nb. x à une constante prés). L'approximation de la fonction $x(u)$ est de type « constante par morceaux » jusqu'à l'échelle $1/2^J$.

On aimerait une approximation meilleure mais tout portait à croire que ce n'était pas possible dans le même temps d'avoir une base orthonormale d'ondelettes. C'est le résultat majeur d'Y. Meyer en 1986 d'avoir prouver l'existence de telles bases avec des régularités bien meilleures.

6.4 Ondelette versus Filtre

Prenons la base de Haar et un signal discrétisé. Le calcul des coefficients d'ondelettes à la taille 2^j dans cette base revient à calculer la moyenne sur 2 intervalles consécutifs de taille $1/2$ plus petite et à en faire la moyenne. Donc, schématiquement à une étape j on procède comme sur la figure 23. A l'étape suivante, on itère avec les positions obtenues.

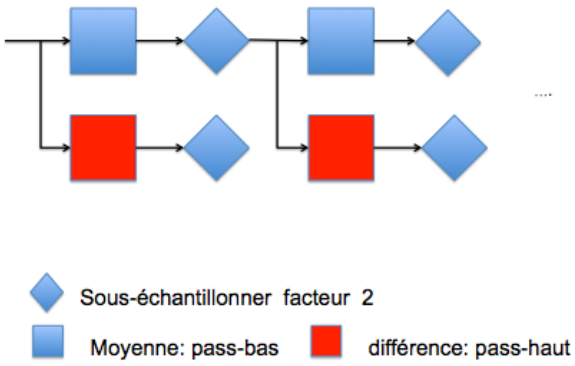

FIGURE 24 – Cascade de filtres passe-bas et passe-haut mis en jeu lors d'une transformée en ondelettes de Haar.

On a donc un **algorithme en cascade** de type:

$$(a, b) \rightarrow (\frac{a+b}{\sqrt{2}}, \frac{a-b}{\sqrt{2}})$$

Cela se met très naturellement sous forme graphique d'une cascade de 2 filtres (figure 24), l'un **passe-bas** (la moyenne dans le cas Haar) et l'autre **passe-haut** (la différence dans le cas de Haar). Et en fait Mallat & Meyer ont montré qu'on pouvait totalement caractériser et construire les ondelettes à partir de la **connaissance de ces 2 types de filtres** et qui plus est que l'on peut construire un algorithme rapide de calcul de la transformée en ondelettes [9].

Maintenant, les **réseaux de neurones** (profonds, DNN) sont en définitives des cascades de filtres et l'apprentissage va consister à **« apprendre » les filtres.** On peut donc se focaliser sur les filtres pour les aspects de mise en œuvre technique, mais pour comprendre le résultat on a envie de connaître le « filtre équivalent » de toute la cascade. **Or, ce filtre équivalent serait l'ondelette**. Donc pour comprendre le réseau il faudrait en quelque sorte reconstruire l'ondelette équivalente. **Mais la grosse différence entre une cascade de filtres et un DNN réside dans les non-linéarités introduites dans les réponses des neurones. Donc cela est beaucoup plus compliqué.**

9. NDJE: voir par exemple http://cas.ensmp.fr/~chaplais/Wavetour_presentation/ondelettes/Biortho_Wave_and_filt.html

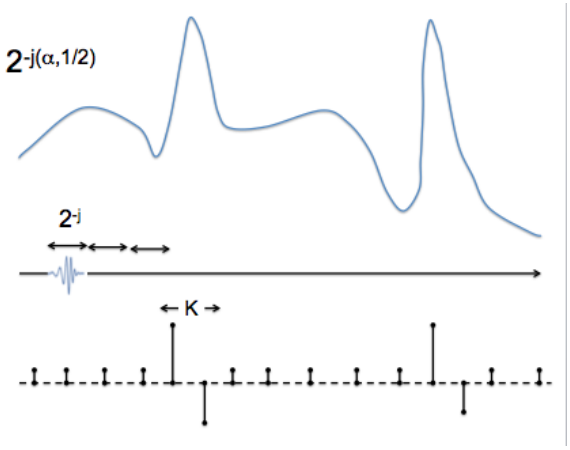

FIGURE 25 – Estimation du lieu des grands coefficients d'ondelettes au voisinage des singularités de la fonction.

6.5 Sparsité ou parcimonie

Soit une fonction Lipschitz-α, on sait que la décroissance des coefficients d'ondelettes va se faire en $2^{-j(\alpha+1/2)}$ ($j > 0$) et les coefficients non nuls (au delà d'un seuil) vont se concentrer au niveau des singularités de la fonction, comme sur le schéma de la figure 25. Le support de l'ondelette est de 2^{-j} (j grand) et elle se déplace de ce pas en translation, donc tant que la fonction est assez uniforme sur le support de l'ondelette, le produit scalaire est nul. Les 2 zones de grandes variations de la fonction (singularité) vont ne concerner qu'un petit nombre de coefficients. A grande échelle (j petit) on se rappelle que l'ondelette à un grand support et que l'échantillonnage et aussi à grand pas, donc on ne garde également que peu d'échantillons.

Quand on ne garde que les échantillons au dessus d'un seuil, l'échantillonnage est adaptatif (donc non-linéaire), et donc le phénomène de Gibbs qui concernait la TF où l'on ne garde que les M-premiers termes, disparait.

Le cours se termine avec quelques illustrations sur des cas 1D et 2D. Avant de conclure ce chapitre, je vais me permettre quelques ajouts.

6.6 Quelques commentaires (JE)

Dans le cours, S. Mallat a introduit l'ondelette ψ à support en fréquence borné. Elle est bien adaptée pour capturer les discontinuités du signal. Discontinuités qui sont plutôt liées à la « haute fréquence » donc associé au filtre « passe-haut ». Cependant, le signal a aussi des parties régulières qui concernent donc sa composante « basse fréquence »; il faudrait donc un filtre « passe-bas » pour la capturer. S. Mallat en a touché un mot à propos de l'algorithme en cascade, en faisant allusion à ces deux filtres « passe-bas » pour les régularités et « passe-haut » pour les discontinuités.

Je me propose de résumer (sans démonstration) quelques propriétés de l'Analyse Multi-Résolution (AMR) mise au point par S. Mallat dans les années 1987-89 [10]. Il est fort possible que dans les cours des années ultérieures, il en fasse lui-même un développement. Voyez donc ce qui suit comme un complément pour les curieux/ses.

6.6.0.1 *La « scaling function » ϕ et son filtre passe-bas*

On va prendre les fonctions de carrée (énergie) sommable sur $\mathbb{R}$ (cf. $L^2(\mathbb{R})$). Considérons une cascade d'ensembles emboités de type ($n \in \mathbb{Z}$):

$$\cdots \subset V_{-n} \subset \cdots \subset V_{-2} \subset V_{-1} \subset V_0 \subset V_1 \subset V_2 \subset V_n \subset \ldots$$

l'intersection des V_i se réduit à la fonction 0 tandis que l'union rempli $L^2(\mathbb{R})$. Si on note que V_j est associé à une échelle d'introspection s_j alors

$$\boxed{V_{j_1} \subset V_{j_2} \Leftrightarrow s_{j_2} < s_{j_1}}$$

(*nb. on peut rencontrer un changement de signe d'indice dans la littérature, tout dépend de la définition de l'opérateur de dilatation dans la suite).*

Le choix « dyadique » pour l'échelle associée à V_j a déjà été motivé et en fait une fonction de V_0 et sa version « jumelle » dans V_j sont reliées par:

$$f(2^j x) \in V_j \Leftrightarrow f(x) \in V_0$$

10. voir la référence Mallat, S.G., 1989,"A theory for multiresolution signal decomposition: the wavelet representation, IEEE Trans. PAMI 11, 674-693.

Il existe une fonction $\phi \in V_0$ dont les translatées par pas entiers couvre l'ensemble V_0, c'est-à-dire que:

$$\forall f \in V_0, f(x) = \sum_{k\in\mathbb{Z}} c_k T_k[\phi](x) = \sum_{k\in\mathbb{Z}} c_k \phi(x-k)$$

et les $\{T_k[\phi], k \in \mathbb{Z}\}$ forment une base orthonormale de V_0. On démontre que l'intégrale de ϕ satisfait la relation

$$\int_{-\infty}^{\infty} \phi(x)dx = 1$$

L'espace V_j est généré par la base orthonormale $\{D_j T_k[\phi], k \in \mathbb{Z}\}$ si D_j représente l'opérateur de dilatation (contraction) par 2^j:

$$amplitude \quad \rightarrow \quad \times 2^{j/2} \tag{1}$$

$$x \quad \rightarrow \quad 2^j x \tag{2}$$

et

$$D_j T_k[f](x) = 2^{j/2} f(2^j x - k).$$

Donc, comme $V_0 \subset V_1$, alors $\phi \in V_0$ se décompose sur la base de V_1, ce qui donne la relation fondamentale (*scaling equation* ou *two-scale equation*):

$$\boxed{\phi(x) = \sum_{k\in\mathbb{Z}} h_k \sqrt{2}\phi(2x-k)}$$

Les coefficients $\{h_k, k \in \mathbb{Z}\}$ forment un filtre passe-bas au sens de Fourier. Il est appelé « filtre d'ondelette » ou *wavelet filter* dans la littérature. Considérons, la fonction de transfert $H(\omega)$ constitué par ce filtre:

$$\boxed{H(\omega) = \sum_{k\in\mathbb{Z}} h_k e^{-ik\omega} \equiv \sqrt{2} m_0(\omega)}$$

alors dans l'esapce de Fourier la scaling equation s'écrit

$$\boxed{\hat{\phi}(\omega) = m_0(\omega/2)\hat{\phi}(\omega/2)}$$

Par itération, **en se donnant la fonction de transfert donc le filtre passe-bas, on peut**

construire la transformée de Fourier de la fonction ϕ:

$$\boxed{\hat{\phi}(\omega) = \prod_{n=1}^{\infty} m_0(2^{-n}\omega)}$$

A propos du filtre discret $\{h_k, k \in \mathbb{Z}\}$, on montre que

$$\sum_{k\in\mathbb{Z}} h_k = \sqrt{2} \quad \Rightarrow \quad m_0(0) = 1 \tag{3}$$

$$\sum_{k\in\mathbb{Z}} h_k h_{k-2\ell} = \delta_\ell \qquad \forall \ell \in \mathbb{Z} \tag{4}$$

La première relation vient de la condition sur l'intégrale de $\phi(x)$, la seconde découle de l'orthonormalité des fonctions $\phi(x)\phi(x-\ell)$ d'une part et $\phi(2x-k)\phi(2x-2\ell-m)$ d'autre part. Cette seconde équation est à l'origine de *l'algorithme à trous* élaboré avec la self-convolution de h.

A propos de la fonction de transfert m_0, la relation d'orthogonalité des fonctions $\phi(x)$ et $\phi(x-k)$ implique que

$$\sum_{\ell\in\mathbb{Z}} |\hat{\phi}(\omega + 2\pi\ell)|^2 = 1 \tag{5}$$

$$\Rightarrow |m_0(\omega)|^2 + |m_0(\omega+\pi)|^2 = 1 \tag{6}$$

6.6.0.2 *La « wavelet function » ψ et son filtre passe-haut: relation avec ϕ*

Dans le cours, S. Mallat montre qu'à partir d'une ondelette ψ on peut construire une base orthonormale de $L^2(\mathbb{R})$ par dilatation et translation dyadiques (*nb: attention ici le j a le signe opposé de celui du cours*):

$$\Big\{\psi_{j,k}(x) = 2^{j/2}\psi(2^j x - k) = D_j T_k[\psi](x), j, k \in \mathbb{Z}\Big\}$$

A $j = j_0$ fixé, la famille $\{D_{j_0} T_k[\psi](x), k \in \mathbb{Z}\}$ est **une base orthonormale de l'espace $W_{j_0} = V_{j_0+1} \ominus V_{j_0}$; c'est-à-dire que V_{j_0+1} est la somme orthogonale de V_{j_0} et de W_{j_0}.**

Dans le cas $j_0 = 0$, l'ondelette ψ étant membre de V_1, on a une scaling relation du

même type que pour ϕ, à savoir:

$$\boxed{\psi(x) = \sum_{k\in\mathbb{Z}} g_k \sqrt{2}\phi(2x-k)}$$

avec les coefficients $\{g_k, k \in \mathbb{Z}\}$ définissant le filtre discret associé à ψ. Comme pour m_0 on définit la fonction de transfert m_1 telle que

$$\boxed{m_1(\omega) = \frac{1}{\sqrt{2}} G(\omega) = \frac{1}{\sqrt{2}} \sum_{k\in\mathbb{Z}} g_k e^{-i\omega k}}$$

On démontre comme pour ϕ que la transformée de Fourier de ψ est soumise à la relation

$$\boxed{\hat{\psi}(\omega) = m_1(\omega/2)\hat{\phi}(\omega/2)}$$

Les deux fonctions de transfert m_1 et m_0 sont reliées par la relation:

$$m_1(\omega)\overline{m_0(\omega)} + m_1(\omega+\pi)\overline{m_0(\omega+\pi)} = 0$$

En fait il existe une fonction 2π-périodique $\lambda(x)$ telle que $\lambda(\omega) = -\lambda(\omega+\pi)$ et telle que

$$m_1(\omega) = \lambda(\omega)\overline{m_0(\omega+\pi)}$$

Si de plus on choisit $|\lambda(\omega)|^2 = 1$ on obtient une base orthogonale de $\psi_{j,k}$. Les choix typiques sont $\lambda(\omega) = -e^{-i\omega},\ e^{-i\omega},\ e^{i\omega}$, d'une manière standard on prend

$$\boxed{m_1(\omega) = -e^{-i\omega}\overline{m_0(\omega+\pi)}}$$

qui finit de fixer la fonction de transfert m_1 (et donc ψ) dés que l'on définit m_0. Notons que cette définition implique immédiatement que

$$|m_0(\omega)|^2 + |m_1(\omega)|^2 = 1$$

donc l'énergie est conservée dans la décomposition; et les coefficients g_k satisfont la relation

$$\boxed{g_k = (-1)^k h_{1-k}}$$

ce qui fait de g_k **un filtre passe-haut et les deux filtres** h **et** g **sont des filtres miroirs en quadrature.** *Notons que si on prend* $\lambda(\omega) = e^{i\omega}$ *alors* $g_k = (-1)^{k-1}h_{-1-k}$*; et que l'on peut toujours ajouter un shift constant sur les indices de* h *et* g *qui ne font faire qu'une translation de position des fonctions* ϕ *et* ψ*.* **Donc, finalement la donnée du filtre passe-bas** h **permet de construire:**

— $m_0(\omega)$ **puis** $\hat{\phi}(\omega)$ **puis** $\phi(t)$**;**

— **le filtre passe-haut** g **qui permet de définir** $m_1(\omega)$ **et enfin** $\hat{\psi}(\omega)$ **et** $\psi(t)$**.**

6.6.0.3 *Les ondelettes de I. Daubechies*

Dans le cours, S. Mallat introduit les ondelettes de Haar, puis donne des arguments pour utiliser des ondelettes avec les N premiers moments nuls en liaison avec la décroissance rapide des coefficients d'ondelettes d'un signal x Lipschitz-α.

I. Daubechies (1992) invente un type d'ondelette ψ dont les N premiers moments sont nuls et avec la propriété

$$|\phi(x)| \leq C_2(1+|x|)^{-\alpha}; \quad \alpha > N$$

alors on montre que

$$m_0(\omega) = \left(\frac{1+e^{-i\omega}}{2}\right)^N \times \mathcal{P}(\omega)$$

avec $\mathcal{P}(\omega)$ un polynôme trigonométrique. Les relations entre m_1 et m_0 données dans le paragraphe précédent donnent

$$|m_0(\omega)|^2 + |m_0(\omega+\pi)|^2 = 1$$

Si on pose

$$|\mathcal{P}(\omega)|^2 \equiv P_0(y); \quad y \equiv \sin^2\frac{\omega}{2}$$

avec P_0 un polynôme, alors

$$(1-y)^N P_0(y) + y^N P_0(1-y) = 1$$

Comme y^N et $(1-y)^N$ sont premiers entres eux, en utilisant Bezout, on montre que

$$P_0(y) = \sum_{k=0}^{N-1} \begin{pmatrix} N+k-1 \\ k \end{pmatrix} y^k$$

Ensuite, on utilise un théorème de Fejér-Riesz pour passer de P_0 à $\mathcal{P}$. En quelques mots, on peut reformuler P_0 en terme d'un polynôme en $\cos\omega$, puis en $z = e^{i\omega}$ puisque $\cos\omega = (z + 1/z)/2$:

$$P_0(\cos\omega) = \frac{a_0}{2} + \sum_{k=1}^{N-1} a_k \cos^k \omega = \sum_{j=-(N-1)}^{N-1} c_j z^j = P_0(z)$$

On montre alors que

$$P_0(z) = c \prod_{j=1} r(z - \alpha_j)(1/z - \alpha_j^*)$$

avec $c > 0$, et $|\alpha_j| \geq 1$, ce qui conduit à

$$\mathcal{P}(\omega) = \sqrt{c} \prod_{j=1}^{r} (e^{i\omega} - \alpha_j)$$

Finalement

$$\boxed{m_0(\omega) = \sqrt{c} \left(\frac{1 + e^{-i\omega}}{2} \right)^N \prod_{j=1}^{r} (e^{i\omega} - \alpha_j)}$$

Prenons un exemple $N = 2$ qui concerne l'ondelette notée **Db2** (ou DAUB4) dans la littérature. Donc,

$$\begin{aligned} P_0(y) &= 1 + 2y \\ &= 2 - \cos\omega \\ &= \frac{1}{2z}(-z^2 + 4z - 1) \\ &= \frac{2 - \sqrt{3}}{2}(z - (2 + \sqrt{3}))(1/z - (2 + \sqrt{3})) \end{aligned}$$

ce qui conduit à

$$\mathcal{P}(\omega) = \frac{\sqrt{2 - \sqrt{3}}}{\sqrt{2}}(e^{i\omega} - (2 + \sqrt{3}))$$

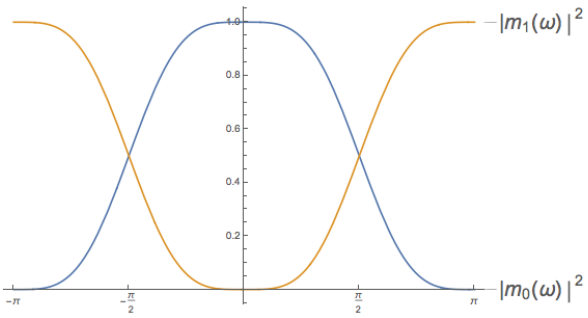

FIGURE 26 – Filtres passe-haut (m_1) de ψ et passe-bas (m_0) de ϕ pour l'ondelette de I. Daubechies "Db2".

Comme $\sqrt{2-\sqrt{3}} = (\sqrt{3}-1)/\sqrt{2}$, finalement m_0 est donné par

$$\boxed{m_0(\omega) = \frac{1}{8}\left(1+\sqrt{3}+(3+\sqrt{3})e^{-i\omega}+(3-\sqrt{3})e^{-i2\omega}+(1-\sqrt{3})e^{-i3\omega}\right)}$$

Les graphes de $|m_0(\omega)|^2$ (passe-bas) et $|m_1(\omega)|^2$ (passe-haut) correspondants sont donnés sur la figure 26. A partir du développement de m_0, on extrait les éléments du filtre discret passe-bas h_k et ceux du filtre passe-haut g_k :

$$\begin{array}{rcrcrcl} h_0 & = & -g_1 & = & \frac{1+\sqrt{3}}{4\sqrt{2}} & \approx & 0.482963 \\ h_1 & = & g_0 & = & \frac{3+\sqrt{3}}{4\sqrt{2}} & \approx & 0.836516 \\ h_2 & = & -g_{-1} & = & \frac{3-\sqrt{3}}{4\sqrt{2}} & \approx & 0.224144 \\ h_3 & = & g_{-2} & = & \frac{1-\sqrt{3}}{4\sqrt{2}} & \approx & -0.12941 \end{array}$$

D'une manière générale, l'odelette d'ordre N à des filtres associés de longueur $2N$.

Pour construire une représentation de $\phi(x)$ dont le support est $[0, 2N-1]$ (pour Db2, $N=2$), Daubechies et Lagarias ont développé un algorithme simple qui consiste en:

- définir 2 matrices $(2N-1)\times(2N-1)$ selon $T_0 = \sqrt{2}h_{2i-j-1}; 1 \leq i,j \leq 2N-1$ et $T_1 = \sqrt{2}h_{2i-j}; 1 \leq i,j \leq 2N-1$;
- pour $x \in]0,1]$ donner la liste des digits en base 2 de x tronquée à m digits : ex. pour $m=8$, $0.05_{10} = 0.00001100_2$ donne $\{0,0,0,0,1,1,0,0\} = d_{1\leq j\leq 8}$

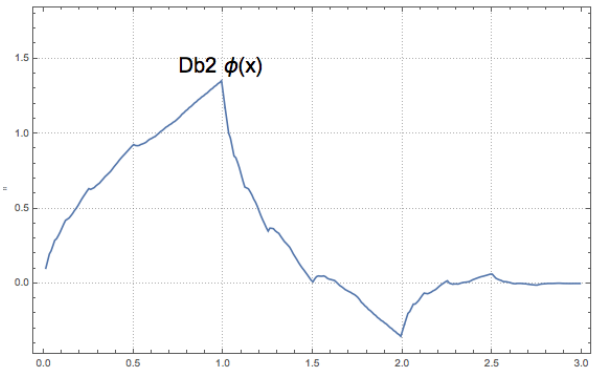

FIGURE 27 – Ondelette $\phi(x)$ Db2.

— Pour chaque d_j associer T_{d_j} et construire le produit de matrice

$$T_m(x) = \prod_{1 \leq j \leq m} T_{d_j}$$

Cette matrice converge vers une matrice $(2N-1)\times(2N-1)$ dont chaque colonne converge vers le vecteur:

$$T_m(x) \underset{m\to\infty}{\longrightarrow} \begin{pmatrix} \phi(x) \\ \phi(x+1) \\ \cdots \\ \phi(x+2N-2) \end{pmatrix}$$

Dans le cas de Db2 on obtient par cette procédure $\phi(x), \phi(x+1)$ et $\phi(x+2)$ par moyenne par ligne de $T_m(x)$. Le graphe de $\phi(x)$ ainsi obtenu est donné sur la figure 27. Les petits structures ne sont pas dues à des artefacts d'approximations numériques. Pour $\psi(x)$, en se servant de la relation

$$\psi(x) = \sum_{k\in\mathbb{Z}} g_k \sqrt{2}\phi(2x-k)$$

et le calcul de $\phi(x)$ avec les coefficients du filtre passe-haut donnés ci-dessus on obtient le graphe de la figure 28 [11].

Avant de finir cette section, mentionnons que la solution de l'équation de Bezout sur

11. nb. Il existe un autre algorithme pour calculer directement $\psi(x)$.

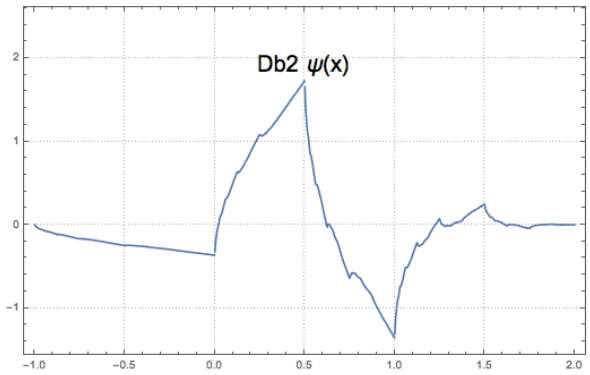

FIGURE 28 – Ondelette $\psi(x)$ Db2.

P_0 choisie plus haut peut être étendue si le degré de P_0 n'est pas restreint à $N-1$. Dans ce cas, on peut prendre $P(x) = P_0(x) + y^N R(1/2-y)$ avec R un polynôme impair préservant la positivité de $P(x)$. Ainsi, la famille d'ondelettes de Daubechies vue dans cette section peuvent être étendue donnant: les symmlets, les ondelettes complexes Daubechies, les coiflets...

Maintenant examinons comment obtenir les coefficients d'ondelettes d'un signal discrétisé x_i et sa synthèse.

6.6.0.4 *DWT (discret wavelet transform) et son inverse IDWT (version)*

En se rappelant la décomposition

$$L^2(\mathbb{R}) = \bigoplus_{j=-\infty}^{\infty} W_j = V_{j_0} \oplus \bigoplus_{j\geq j_0}^{\infty} W_j$$

, ceci peut être matérialisé par la figure des « poupées gigognes » (29). S. Mallat a dans son cours élaboré la première décomposition pour mettre en évidence les discontinuités d'une fonction; nous avons plutôt considéré la seconde version où j_0 est la plus fine échelle d'introspection.

Si on prend un signal f échantillonné à petite l'échelle d'introspection J, alors $\tilde{f}_J \in V_J$ (nb. $J \gg 0$) et qu'on considère que l'échelle la plus grossière est j_0 alors comme

$$V_J = V_{J-1} \oplus W_{J-1} = V_{j_0} \oplus \bigoplus_{j\geq j_0}^{J-1} W_j$$

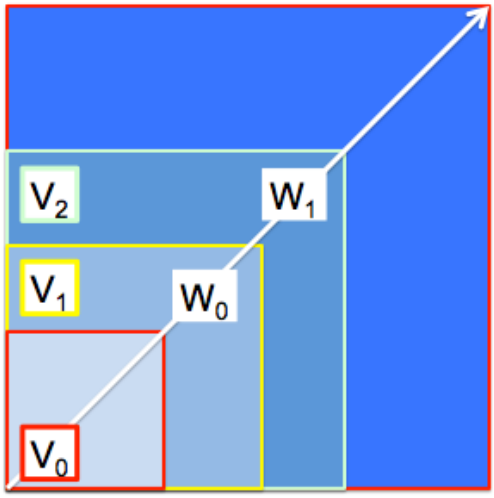

FIGURE 29 – Ensembles emboités d'une analyse multi-résolution.

alors

$$\tilde{f}_J(x) = \overbrace{\sum_{k=-\infty}^{\infty} c_{j_0,k} 2^{j_0/2} \phi(2^{j_0}x - k)}^{V_{j_0}} + \sum_{j \geq j_0}^{J-1} \left\{ \overbrace{\sum_{k=-\infty}^{\infty} d_{j,k} 2^{j/2} \psi(2^j x - k)}^{W_j} \right\}$$

$$= \sum_k c_{j_0,k} \phi_{j_0,k}(x) + \sum_{j \geq j_0}^{J-1} \sum_k d_{j,k} \psi_{j,k}(x)$$

Par application de l'orthogonalité des $\phi_{j,k}$ et $\psi_{j',k'}$, il vient

$$c_{j_0,k} = \langle \tilde{f}_J, \phi_{j_0,k} \rangle = \sum_\ell h_{\ell - 2k} \langle \tilde{f}_J, \phi_{j_0+1,\ell} \rangle$$

Avec les équations définissants les filtres passe-bas (h_k) et passe-haut (g_k) faisant intervenir l'équation de scaling de ϕ et l'équivalente pour ψ, on peut démontrer facilement que

$$\phi_{j,\ell} = \sum_k h_{k-2\ell} \phi_{j+1,k} \tag{7}$$

$$\psi_{j,\ell} = \sum_k g_{k-2\ell} \phi_{j+1,k} \tag{8}$$

$$\tag{9}$$

Tout ceci donne la relation de récurrence (en « anonymisant » j_0)

$$\boxed{c_{j-1,k} = \sum_{\ell} h_{\ell-2k} c_{j,\ell}}$$

De même la relation $d_{j-1,k} = \langle \tilde{f}_J, \psi_{j-1,k} \rangle$ donne une relation de récurrence

$$\boxed{d_{j-1,k} = \sum_{\ell} g_{\ell-2k} c_{j,\ell}}$$

L'algorithme **DWT** (*discret wavelet transform*) peut se résumer comme sur la figure 30.

L'algorithme inverse (**IDWT**) consiste à utiliser $c_{j_0,.}, d_{j_0+1,.}, \ldots, d_{j,.}, \ldots, d_{J-1,.}$ pour synthétiser $c_{J,.}$. Localement, on aimerait à partir de $(c_{j-1,.}, d_{j-1,.})$ retrouver $c_{j,.}$. Or, le développement de V_J aurait tout aussi bien se faire selon le découpage:

$$V_J = V_{J-1} \oplus W_{J-1} = V_{j_0-1} \oplus \bigoplus_{j \geq j_0-1}^{J-1} W_j$$

si on pousse d'un cran l'échelle d'introspection. Ainsi

$$\tilde{f}_J(x) = \sum_k c_{j_0-1,k} \phi_{j_0-1,k}(x) + \sum_{j \geq j_0-1}^{J-1} \sum_k d_{j,k} \psi_{j,k}(x)$$

Par identification de la partie commune et la partie différente, puis en identifiant les coefficients de $\phi_{j_0,k}$ et finalement anonymisant j_0 on trouve la relation de récurrence

$$\boxed{c_{j,k} = \sum_{\ell} c_{j-1,\ell} h_{k-2\ell} + \sum_{\ell} d_{j-1,\ell} g_{k-2\ell}}$$

En itérant cette relation on s'aperçoit que reconstruire c_J n'a pas besoin des $c_{j,.}$ intermédiaires sauf bien sûre $c_{j_0,.}$. Le schéma de l'IDWT est donc celui donné sur la figure 31. Ceci dit, si on passe par une phase de DWT avant de procéder à une nouvelle IDWT par exemple après un « nettoyage » des coefficients de détails, alors on dispose des $c_{j,.}$ intermédiaires.

La mise en œuvre concrète présente quelques subtilités que je vais effleurer ici pour la phase DWT. En principe comme montré ci-dessus, on commence par initialiser $c_{J,k} = x_k$

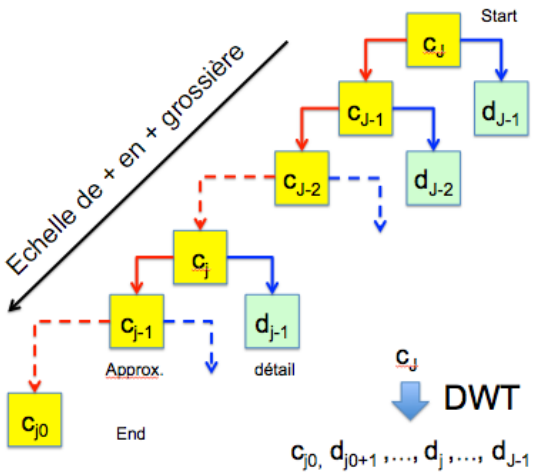

FIGURE 30 – Schéma d'une décomposition en ondelettes (Discret Wavelet Transform).

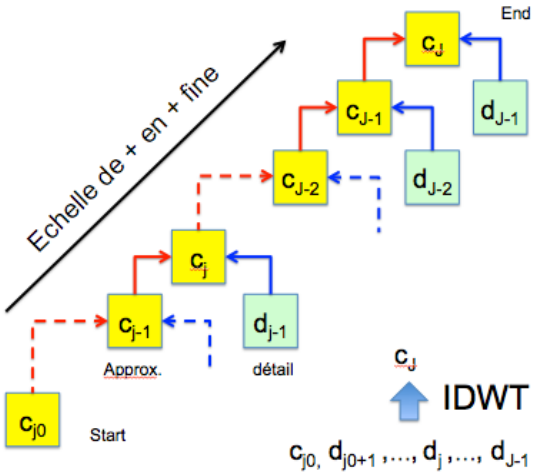

FIGURE 31 – Reconstruction d'un signal à l'aide des coefficients d'ondelettes (Inverse Discret Wavelet Transform).

pour $k = 0, \ldots, N_s - 1$ si on dispose de N_s échantillons, puis on calcule progressivement $(c_{J-1,.}, d_{J-1,.})$, $(c_{J-2,.}, d_{J-2,.})$ etc. Mais on convient assez naturellement que cela s'entendrait mieux si on initialise un $c_{0,k} = x_k$ pour procéder par la suite au calcul de couple $(c_{1,.}, d_{1,.})$, $(c_{2,.}, d_{2,.})$ etc. Ceci n'est en fait qu'un simple jeu d'écriture des indices j. Donc la récurrence sur les $c_{j,k}$ devient alors:

$$c_{j+1,k} = \sum_{\ell} h_{\ell-2k} c_{j,\ell}; \quad j = 0, \ldots$$

Ensuite, si on considère les ondelettes de Daubechies d'ordre N alors le filtre h est de longueur $2N$: $\{h_0, h_1, \ldots, h_{2N-1}\}$. Ainsi la somme sur ℓ est contrainte par $0 \leq \ell - 2k \leq 2N - 1$, ce qui par changement d'indice se reporte sur les éléments concernés du vecteurs $c_{j,.}$ selon

$$c_{j+1,k} = \sum_{\ell=0}^{2N-1} h_\ell \; c_{j,\ell+2k}$$

Ce type de relation est typique d'une multiplication du vecteur $c_{j,.}$ par une matrice circulante à décalage de 2 entre chaque ligne dont la première est $\{h_0, h_1, \ldots, h_{2N-1}, 0, \ldots, 0\}$ de longueur N_s.

La question est de connaître la dimension du vecteur $c_{j+1,.}$ connaissant celui du vecteur $c_{j,.}$. En fait cette question est reliée aux effets (ou gestion) des bords. Si le signal à une longueur en puissance de 2, cf. $N_s = 2^s$, une possibilité est de construire des matrices successives de dimension $2^{s-n} \times 2^{s-n+1}$ avec une diminution d'un facteur 2 du nombre de coefficients par itération et une hypothèse de signal périodique. *Mathematica* et *pyWavelets (pywt)* ont une autre stratégie avec une relation de récurrence de la taille du nombre de coefficients selon avec $w_0 = N_s$ et $FL = 2N$ (filter length):

$$w_j = \left\lceil \frac{1}{2}(w_{j-1} + FL - 2) \right\rceil \quad (Mathematica) \tag{10}$$

$$\textit{ou bien} \quad w_j = \left\lfloor \frac{1}{2}(w_{j-1} + FL - 1) \right\rfloor \quad (pywt) \tag{11}$$

($\lceil x \rceil$: *ceil(x)*; $\lfloor x \rfloor$: *floor(x)*). Notons que les 2 relations sont équivalentes.

Enfin, il y a la profondeur d'introspection (raffinement) maximale admissible, c'est-à-dire la valeur maximale de j dans la relation ci-dessus de récurrence sur $c_{j,.}$ (où $j = 0$ est

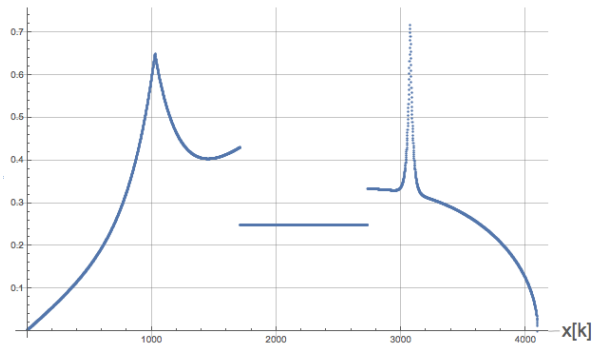

FIGURE 32 – Fonction test échantillonnée (4096) présentant quelques discontinuités de différents types.

l'état du signal en entrée). Il y a des définitions différentes selon les librairies:

$$j_{max} = \left\lfloor \log_2(N_s) + \frac{1}{2} \right\rfloor = s \quad (Mathematica) \tag{12}$$

$$j_{max} = \left\lfloor \log_2\left(\frac{N_s}{FL-1}\right) \right\rfloor < s \quad (pywt) \tag{13}$$

(ex. pour *pywt* avec les ondelettes Db2 on a $j_{max} = s - 1$). La définition prise par *pywt* considère que le vecteur $c_{j_{max},.}$ doit le moins possible être affecté par les effets de bord.

6.6.0.5 *Seuillage: différence entre FFT et DWT, phénomène de Gibbs*

Prenons l'exemple d'un signal comme celui de la figure 32 numérisé avec $4096 = 2^{12}$ échantillons. La décomposition à l'aide d'ondelettes Db2 où on a limité à 6 le niveau d'introspection ou raffinement (le max est à 12) est donnée sur la figure 33. Sur chaque ligne sont portés les coefficients $d_{j,k}$ des « détails » haute-fréquence et sur la dernière les coefficients $c_{6,k}$ d'approximation basse fréquence. Le nombre total de coefficients de détails est de 4109, et 66 coefficients d'approximation. Dans le cours, S. Mallat propose de ne garder que M coefficients de **détails** les plus significatifs: prenons $M = 100$. Il y a donc 100+66=166 coefficients d'ondelettes gardés. D'une manière similaire en faisant une FFT, on peut garder les M premiers coefficients basse-fréquence.

Primo, S. Mallat mentionne le phénomène de Gibbs mis en évidence en 1848 par H. Wilbraham mathématicien anglais, mais popularisé par en 1898 A. Michelson (de Michelson & Morlet) qui pensait à un artefact de ses instruments et que finalement J.W

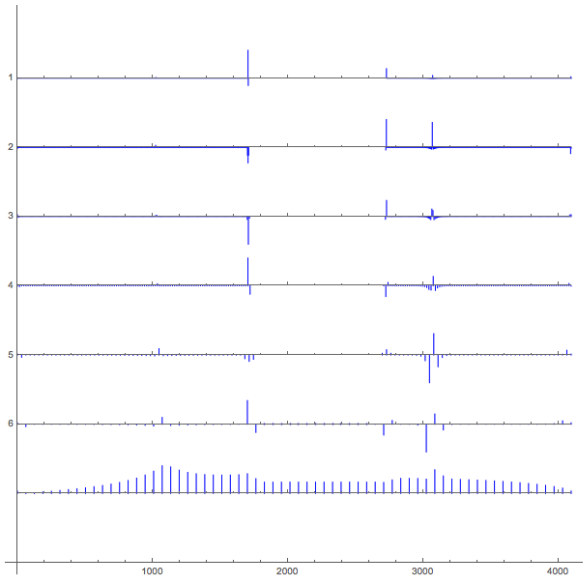

FIGURE 33 – Coefficients d'ondelettes ψ à différentes échelles de détails. L'échantillonnage du bas correspond à l'approximation basse fréquence obtenu par l'odelette ϕ.

Gibbs expliqua comme phénomène d'origine mathématique lié au défaut d'approximation de la transformation de Fourier au voisinage de discontinuité. Ce phénomène se voit bien dans notre cas de « seuillage » comme sur la figure 34. Avec le même nombre de coefficients la reconstruction en ondelettes est bien plus fidèle (voir figure 35). Ce qui montre l'intérêt d'un algorithme adaptatif (non-linéaire selon la notion de S. Mallat). Bien entendu si on ne disposait que de la FFT il y aurait moyen de s'en tirer en filtrant (ce qui reviendrait à adoucir les irrégularités).

Notons que la littérature ainsi que les librairies d'ondelettes fourmillent de procédures de seuillage adaptatif pour enlever le bruit qui vient se rajouter au signal, autres que ne garder que les M coefficients les plus significatifs. Mais on va s'arrêter là pour ce chapitre.

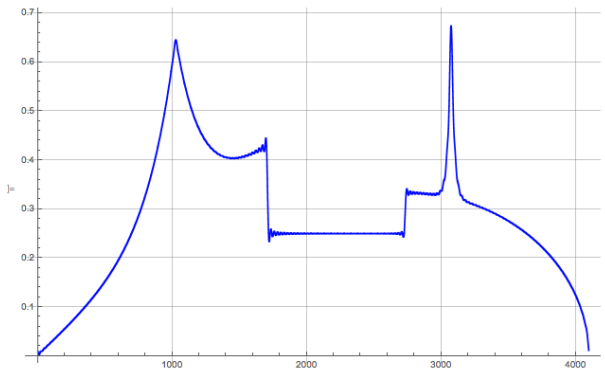

FIGURE 34 – Phénomène de Gibbs au voisinage des singularités du signal par troncature à basse fréquence de la transformée de Fourier.

FIGURE 35 – Reconstruction du signal par IDWT en ne gardant que les coefficients de détails les plus significatifs, ajoutés à ceux de l'approximation basse fréquence, tout en gardant le même nombre total de coefficients que la FFT inverse de la figure **??**.

7. Classification/Régression en grande dimension (Partie I)

7.1 Petit rappel des épisodes précédents

On se rappelle que la problématique centrale est la **« malédiction de la dimension »** quand on veut par exemple estimer/approximer des fonctions en très grande dimension. Pour attaquer le problème, on a vu qu'il fallait s'attacher à **réduire cette dimension** du problème en trouvant des formes de **régularités**. On est retourné en basse dimension, pour appréhender le sujet de la régularité (Lipschitzienne), pour explorer la **Transformée de Fourier** (*cas linéaire*) à la **Transformation en Ondelettes** (*cas non-linéaire*) et plus généralement les **représentations parcimonieuses**. Ces outils vont nous permettre de construire des représentations dans le **cadre de la classification et la régression**.

On va reprendre ces notions en liaison avec **l'algorithmique** et notamment dans ce cours les techniques de **classification/régression linéaire** avec des représentations comme les **classificateurs à noyaux**.

7.2 Modèle déterministe vs stochastique: ce n'est pas le probléme!

Quand on aborde les problèmes de classification/régression on peut avoir deux points de vue:

	Déterministe	Stochastique
Modèle	$y = f(x)$	y fonction aléatoire de x
Question	on veut estimer f	trouver le y le plus probable (Bayes)

Mais trancher entre les deux points de vue (« l'un est meilleur que l'autre ») n'est pas le problème! La véritable question est : comment capturer la régularité du problème pour pouvoir déterminer les paramètres du modèle avec le moins d'échantillons possibles.

7.2.1 Points de vue bayésien et déterministe

Examinons par exemple **le point de vue bayesien** dans la cadre de l'apprentissage supervisé en dimension d ($x \in \mathbb{R}^d$) dans lequel on dispose de n échantillons $\{x_i, y_i\}_{i \leq n}$. On note $\tilde{y}$ **l'estimateur de** y tel que

$$\tilde{y} = \tilde{f}(x)$$

Pour se faire on se donne un **coût ou risque** (le vocable qui sera pris par la suite) à se tromper en estimant y par $\tilde{y}$: $r(y, \tilde{y})$. On cherche $\tilde{f}$ qui **minimise le risque en moyenne** à savoir

$$\tilde{f} = \underset{f \in \mathcal{H}}{\operatorname{argmin}} \; E_{X,Y}[r(Y, f(X))]$$

avec (X, Y) les distributions de probabilité qui « génèrent » les x et y. Rappelons que les fonctions de risque « classiques » sont :

— **Classification** où $y \in \{-1, 1\}$ (généralisable à K classes), on utilise un risque binaire définit par

$$r(y, \tilde{y}) = \begin{cases} 0 & si \; y = \tilde{y} \\ 1 & si \; y \neq \tilde{y} \end{cases}$$

— **Régression** où $y \in \mathbb{R}$, on prend par exemple un risque quadratique

$$r(y, \tilde{y}) = (y - \tilde{y})^2$$

Dans cette approche probabiliste, il faut calculer la moyenne du risque, ici pour la régression

$$\begin{aligned} E_{X,Y}[r(Y, f(X))] &= \int \int p(x, y) r(y, f(x)) dx dy \\ &= \int_{\mathbb{R}^d} p(x) \int_{\mathbb{R}} p(y|x) r(y, f(x)) dy dx \end{aligned}$$

En classification, on remplacerait $\int dy$ par une somme discrète sur les classes $\sum_{y=1}^{K}$, et donc

$$\begin{aligned} E_{X,Y}[r(Y, f(X))] &= \int_{\mathbb{R}^d} p(x) \sum_{y=1}^{K} p(x, y) r(y, f(x)) dx \\ &= \int_{\mathbb{R}^d} p(x) \sum_{y=1}^{K} p(x, y) \times \mathbb{I}(y \neq f(x)) dx \end{aligned}$$

Donc à x fixé, on veut minimiser

$$\sum_{y=1}^{K} p(x, y) \times \mathbb{I}(y \neq f(x))$$

c'est-à-dire que lorsque $p(y, x)$ est grand on veut $f(x) \approx y$, ce qui s'écrit en d'autres termes :

$$\boxed{p(\tilde{f}(x)|x) = \max_{f}(p(f(x)|x))}$$

On reconnait le **classificateur bayesien** qui sélectionne la classe la plus probable sachant x. Est-ce la fin de l'histoire puisqu'on a une formule/un schéma ?

En fait, il y a d'une manière sous-jacente une hypothèse de **régularité sur** $p(y|x)$ qui peut se formuler « trivialement »: *quand x bouge peu alors la probabilité $p(y|x)$ ne change pas trop*. Ca veut simplement dire que « proche » d'un échantillon d'entrainement x_i on considère que $p(y|x_i) = y_i = Cte$. C'est **l'algorithme des plus proches voisins** que nous avons rencontré lors du cours du 31/1/18 (Partie 1). Mais **l'espace entre les échantillons en grande dimension est énorme** si on veut quelques échantillons pour pouvoir estimer raisonnablement $p(y|x)$. Ou bien autrement dit, il faut une très grande densité d'échantillons n pour considérer que l'erreur d'étiquetage ε soit faible. On a vu que

$$n \approx \varepsilon^{-d}$$

Pour le **point de vue déterministe**, f est une unique fonction qu'il s'agit de déterminer pour laquelle $y = f(x)$. En fait, on peut définir une probabilité conditionnelle

$$p(y|x) = \delta(y - f(x))$$

Le fait que f soit unique n'est pas surprenant, d'autant moins d'ailleurs à cause de la grande dimensionalité d du problème, car on a beaucoup d'informations pour distinguer par exemple des images de chat, chien, voiture... ou bien d'échantillons sonores pour reconnaitre qui parle (bien entendu c'est l'idée...). Donc par l'identification de $p(y|x)$ on se ramène à la discussion précédente.

Prenons le cas de la régression avec y une variable continue

$$E_{X,Y}[r(Y, f(X))] = \int_{\mathbb{R}^d} p(x) \int_{\mathbb{R}} p(y|x)(y - f(x))^2 dy \; dx$$

En fait on veut minimiser à x fixé, $(y - f(x))^2$ sachant x. Donc, la solution n'est autre que

$$\boxed{\tilde{f}(x) = E[Y/X] = \int_{\mathbb{R}} y \; p(y|x) \; dy}$$

C'est un résultat particulier du résultat général

$$\min_{\mu} E[(Y - \mu)^2] \Leftrightarrow \mu = E[Y]$$

Le problème est qu'en grande dimension pour estimer $E[Y/X]$, autour de x fixé: soit la boule pour récolter des échantillons est énorme et alors il est vraisemblable que $p(y|x) = Cte$ n'est pas correcte, soit il faut une grande densité d'échantillons pour que le rayon de la boule soit petit mais alors $n \approx \varepsilon^{-d}$. On retombe toujours sur le même problème de la malédiction de la dimension qui fait qu'ici ces espérances conditionnelles sont très difficiles à calculer en grande dimension (voir les MCMC et autres techniques quasi-Monte-Carlo d'intégration).

Donc que cela soit d'un point de vue déterministe ou d'un point de vue probabiliste qui donne à première vue un schéma pour résoudre le problème, la malédiction de la dimensionalité reste entière.

La solution (peut-être provisoire) est de contourner le problème et **d'imposer une forte régularité** soit sur $f(y)$ (point de vue déterministe) soit sur $p(y|x)$ (point de vue probabiliste). On ne suppose plus uniquement que la fonction est localement régulière de type Lipschitz ou Lipschitz-α, mais on va réduire drastiquement la dimensionalité d du problème et le réduire à l'extrême à **1 seule variable**. En généralisant cette idée (trés brutale) de passer de d variables à 1 variable, cela va donner la quasi totalité des

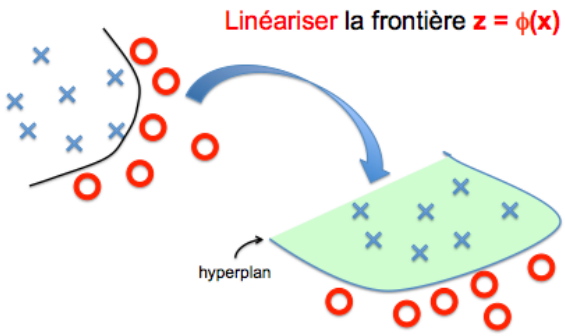

FIGURE 36 – Linéarisation du problème par changement de représentation $z = \phi(x)$. On cherche alors l'hyperplan qui sépare les 2 classes d'échantillons.

algorithmes d'apprentissage autres que NN (MLP).

7.3 Réduction de dimensionalité: noyau de similarité et hyperplan

Par exemple, on se place dans le cadre d'une **classification binaire** $y = \pm 1$. Le principe est le suivant, pour séparer les 2 classes en dimension d, on utilise une transformation

$$z = \phi(x)$$

de telle façon que la frontière entre les 2 classes soit un **hyperplan: ainsi on a linéarisé la frontière.** La figure 36 schématise le procéder.

Dans le nouvel espace, pour classer un échantillon, il suffit de connaître sa position par rapport à l'hyperplan, c'est-à-dire connaître le signe de la distance entre l'échantillon et l'hyperplan. Cela se traduit par la définition d'un estimateur de y, noté $\tilde{y}$, comme suit

$$\boxed{\tilde{y} = \text{sgn}\left\{\langle w, \phi(x)\rangle + b\right\}}$$

avec bien entendu w la normale à l'hyperplan et $\langle w, \phi(x)\rangle$ la projection vectorielle de $\phi(x)$ sur la normale, et b la position du plan affine. **Il faut estimer les paramètres** (w, b)**: les composantes du vecteurs** w **et le scalaire** b**, soit** $d + 1$ **variables.**

On introduit la notion de **similarité** pour caractériser si 2 échantillons sont « proches »

l'un de l'autre comme pour le cas de l'algorithme des plus proches voisins qui en basse dimension est naturel (cf. on sait qu'en grande dimension ça ne marche pas). Donc prenons la distance euclidienne dans le nouvel espace.

$$||\phi(x) - \phi(x')||^2 = ||\phi(x)||^2 + ||\phi(x')||^2 - 2\langle\phi(x), \phi(x')\rangle$$

Donc 2 échantillons sont simillaires si leur distance est proche de 0, c'est-à-dire que leur produit scalaire est grand. Donc étudier la similarité dans le nouvel espace revient à étudier la fonction (**noyau**)

$$\boxed{k(x, x') = \langle\phi(x), \phi(x')\rangle}$$

Et finalement on a le choix (et équivalence) dans cet exemple de linéarisation: **soit définir la fonction ϕ du changement de variable, soit définir directement le noyau de similarité entre 2 points dans le nouvel espace.** Et le problème va donc de trouver le « bon » changement de variables pour linéariser la frontière, ou le « bon » noyau qui permet d'agréger au mieux les échantillons en clusters.

Un autre façon de voir ce problème est de développer le produit scalaire:

$$\phi(x) = (\varphi_k(x))_{k \leq d'} \rightarrow \langle w, \phi(x)\rangle = \sum_k w_k \varphi_k(x)$$

On réalise alors une moyenne pondérée (w est normé à 1) entre d' **« patterns » ou « structures »** $\varphi(x)$, ou encore **on réalise un « vote »** entre les structure, et w est le **vecteur discriminant** [12].

Maintenant qu'on a linéarisé la frontière à chercher pour séparer les 2 classes, est-on tiré d'affaire par rapport à la malédiction de la dimension? Il faut répondre à deux types de questions (pas forcément dans l'ordre):

— Q1: comment optimiser $(w, b) \in \mathbb{R}^{d+1}$?

— Q2 : comment optimiser le changement de variables ou **représentation** $\phi(x)$?

Q2 est la plus critique, et pour y répondre on envisage deux stratégies:

— **soit on fixe *a priori*** la représentation $\phi(x)$, que l'on peut remplacer par le choix du **noyau de similarité** $k(x_1, x_2)$;

12. NDJE: S. Mallat montre sur un exemple simple que compter le nombre de pixels « rouges » dans une image où il y a soit des voitures soit des camions de pompiers, peut être une structure/pattern discriminant assez simple pour reconnaitre des camions de pompiers.

— **soit l'algorithme est capable d'apprendre par lui-même la représentation** car on a pas suffisamment d'éléments/informations pour la fixer *a priori*. C'est le domaine des **réseaux de neurones (NN/MLP)**.

7.3.1 Trouver l'hyperplan (w, b): la régularisation nécessaire

On oublie pour l'heure $\phi(x)$, dans le nouvel espace on le note x qui est un vecteur en dimension d' mais on la note d pour simplifier, et on cherche donc un classifieur sous la forme

$$\operatorname{sgn}\left(\sum_k w_k x_k + b\right)$$

L'espace des classifieurs $\mathcal{H}$ est définit par l'ensemble des $\tilde{f}$ dépendant de (w, b)

$$\mathcal{H} = \left\{(w, b) \in \mathbb{R}^{d+1}\right\}$$

Or on a vu dans le cours du 24/1/18 que *l'erreur de fluctuations* liée à la taille de $\mathcal{H}$ est donnée par

$$\varepsilon^2 \sim \frac{\log |\mathcal{H}|}{n}$$

c'est-à-dire le rapport entre la taille de $\mathcal{H}$ et le nombre d'exemples/échantillons. Ici même si $\mathbb{R}^{d+1}$ est infini, on peut le quantifier mais quand bien même il faudrait un **nombre infini de « bins »**, ce qui entraîne un ensemble infini des « possibles » qui entraîne l'effet d'**overfitting**. Donc, il faut **imposer une condition pour diminuer la dimensionalité** du $\mathcal{H}$ à considérer et réduire l'overfit.

Notation: si $x' = (x_1, \ldots, x_d, b)$ et $w' = (w_1, \ldots, w_d, 1)$ alors $\sum_k w_k x_k + b = \sum_\ell w'_\ell x'_\ell$. Par la suite on oublie les $'$.

Soit le nouveau **risque empirique régularisé** (attention x_i ici signifie le i-éme échantillon et non la i-éme composante):

$$\boxed{\tilde{R}_r(h) = \frac{1}{n}\sum_{i=1}^{n} r(y_i, h(x_i)) + \lambda ||w||^2} \qquad \text{avec } h(x_i) = \operatorname{sgn}\langle w, x_i\rangle$$

Par le nouveau terme, on réduit $\mathcal{H}$ à un **ensemble compact**. D'une manière générale les notions **d'overfitting, de stabilité de la prédiction, et de convexité de $\mathcal{H}$ sont étroitement**

liées.

Examinons un instant le cas *sans régularisation* pour un problème de régression (on verra la classification après qui ajoute une complication). On a un risque empirique $\tilde{R}(h)$ tel que

$$\tilde{R}(h) = \tilde{R}(w) = \frac{1}{n}\sum_{i=1}^{n}(y_i - h(x_i))^2$$

La minimisation par rapport aux composantes de w donne une équation liénaire

$$\frac{1}{n}\sum_i \langle w, x_i\rangle x_i = \frac{1}{n}\sum_i y_i x_i \equiv c$$

(nb: attention encore ici x_i est un vecteur en dimension $d+1$ tout en sachant que le d et la dimension de l'espace après le changement de représentation). Si on définit la matrice $\mathbf{A}$ par

$$\mathbf{A} = \frac{1}{n}\sum_i x_i x_i^T$$

alors on a à résoudre un système linéaire de type

$$\mathbf{A}w = c$$

dont la solution si $\mathbf{A}$ est inversible devient:

$$w = \mathbf{A}^{-1}c$$

Mais, il y a 2 « mais »:

- en général $\mathbf{A}$ est non-inversible: ça peut très bien se gérer avec la notion de pseudo-inverse de Moore-Penrose;
- $\mathbf{A}^{-1}$ est **instable** (branche des « problèmes inverses »), et une erreur sur c (changer la réponse y_i pour un x_i) entraine une amplification des erreurs. Ceci engendre **naturellement de l'overfit**.

Voyons un peu de quoi il retourne. La définition de $\mathbf{A}$ dit quelle est symétrique, elle est donc diagonalisable selon la forme:

$$\mathbf{A} = \mathbf{ODO}^T$$

avec $\mathbf{O}$ une matrice orthogonale, et $\mathbf{D}$ la matrice diagonale des valeurs propres de $\mathbf{A}$.

Si l'espace engendré par les n échantillons (V) est de dimension d alors $\mathbf{A}$ est inversible ($\mathbf{D}$ n'a pas de valeur nulle). Si l'espace engendré est de dimension plus petite, alors il faut construire la pseudo-inverse notée $\mathbf{A}^+$. L'ensemble $\mathbb{R}^d$ est séparé en 2 sous-ensembles vectoriels V et $V^\perp$ définis par

$$\begin{cases} \forall x \in V^\perp & \mathbf{A}^+ x = 0 \\ \forall x \in V & x = \mathbf{A}w \Rightarrow \mathbf{A}^+ x = w \end{cases}$$

Notons que l'on peut utiliser la décomposition en valeur singulière (SVD) pour définir la pseudo-inverse. La solution du problème linéaire

$$\mathbf{A}w = c$$

se compose d'une solution générale à l'équation homogène $\mathbf{A}w = 0$ et d'une solution particulière donnée par l'application de la pseudo-inverse si (et seulement si) $\mathbf{A}\mathbf{A}^+c = c$, donc d'une manière générale:

$$w_{sol} = \mathbf{A}^+ c + \mathbf{P_0} w_0$$

avec $\mathbf{P_0} = \mathbf{I} - \mathbf{A}^+\mathbf{A}$ le projecteur orthogonal sur le noyau de $\mathbf{A}$; $\mathbf{A}\mathbf{P_0} = 0$.

Si de plus, on contraint la norme L2 de w (cf. $||w||^2$) a être minimale, alors cela réduit w_{sol} à la solution pseudo-inverse:

$$\boxed{\mathbf{A}w = c \text{ et } ||w||^2 \text{ minimale} \Rightarrow w = \mathbf{A}^+ c}$$

D'où la présence dans l'expression du risque du terme $||w||^2$. **Cette opération réduit la dimensionalité du problème et donc aide à combattre l'overfitting.**

7.3.2 Comment la régularisation permet de stabiliser la réponse?

Nous avons trouver un moyen de calculer les paramètres de l'hyperplan (w, b) qui sépare les 2 populations d'échantillons étiquetés $\{x_i, y_i\}$. Mais est-ce que cela répond au problème de **l'erreur de généralisation**, c'est-à-dire quand on prend des **échantillons de test**. Donc, à la suite de l'entraînement, pour tout x on dispose d'un estimateur de la

réponse $\tilde{f}(x)$ (*rappel: implicitement b est contenu dans x*)

$$\tilde{f}(x) = \langle \tilde{w}, x \rangle$$

Selon le type de x, on obtient des risques empiriques différents:

$$\begin{cases} x_i \in \text{"Training set"} & \to \tilde{R}(\tilde{f}) \\ x_i \in \text{"Test set"} & \to R^{test}(\tilde{f}) \end{cases}$$

Imaginons que le « Test set » soit identique au « Training » set à l'exception d'un seul échantillon $(x_k, \bar{y}_k)$. Si l'estimateur $\tilde{f}$ est stable, alors un petit changement de ce type n'a pas d'influence sur le risque, on n'a pas d'overfit. En fait, on se pose la question de savoir ce qu'aurait donner le calcul de w à la suite du changement de la valeur de c (notée $\bar{c}$) quand on change y_k en $\bar{y}_k$? Il vient $\mathbf{A}\bar{w} = \bar{c}$ et

$$||w - \bar{w}|| = ||\mathbf{A}^{-1}(c - \bar{c})|| \leq ||\mathbf{A}^{-1}|| \times ||c - \bar{c}||$$

En principe $||c - \bar{c}||$ est petit car on a fait un petit changement sur 1 seul échantillon. Par contre

$$||\mathbf{A}^{-1}|| = \max(\sigma_\ell^{-1})_\ell$$

Donc si la matrice A a une valeur propre très petite, il va y avoir une instabilité. Et pour se prémunir de petite valeur propre dans les « problèmes inverses », on a recours à la régularisation ! C'est un sujet découvert dans plusieurs domaines dans les années 1940 et connu sous le vocable de régularisation de Tikhonov–Miller mentionné par S. Mallat.

Comment cette régularisation opère une stabilisation en pratique? Rappelons que le risque empirique est égal à

$$\tilde{R}_r(h) = \frac{1}{n} \sum_{i=1}^{n} (\langle \tilde{w}, x_i \rangle - y_i)^2 + \lambda w.w^T$$

Le gradient par rapport à w donne l'équation régularisée suivante:

$$(\mathbf{A} + \lambda \mathbf{I})w = c$$

Or la matrice $\mathbf{A} + \lambda \mathbf{I}$ est toujours inversible car en utilisant la décomposition de $\mathbf{A}$, on se

rend compte que la nouvelle matrice diagonale est donnée par:

$$\mathbf{D}_r = \mathbf{D} + \lambda \mathbf{I}$$

et donc les nouvelles valeurs propres sont toujours $\sigma'_\ell \geq \lambda$. Ainsi, quand on se pose la question de l'effet d'un petit changement comme précédemment on obtient l'inégalité

$$||w - \bar{w}|| \leq ||(\mathbf{A} + \lambda \mathbf{I})^{-1}|| \times ||c - \bar{c}|| = \frac{1}{\lambda}||c - \bar{c}||$$

donc si $||c - \bar{\{c\}}|| \approx \varepsilon$ alors $||w - \bar{w}|| \approx \varepsilon$. **On a bien supprimer/diminuer l'overfit.**

7.3.3 La convexité

Une autre façon de voir les choses, le problème (Pb1) revient à faire une minimisation

$$\min \tilde{R}_r(w) = \min \left(\frac{1}{n} \sum_{i=1}^{n} (\langle w, x_i \rangle - y_i)^2 + \lambda ||w||_2^2 \right)$$

qui est équivalent au problème (Pb2) de minimisation sous contrainte, à savoir

$$\min \tilde{R}(w) = \min \left(\frac{1}{n} \sum_{i=1}^{n} (\langle w, x_i \rangle - y_i)^2 \right) \quad \text{avec} \quad ||w||_2^2 < \beta$$

Or $\tilde{R}(w)$ est une **fonction convexe** dont on veut trouver le minimum **avec une contrainte**. On peut interpréter ce problème comme celui d'un lagrangien où il va correspondre un λ (*multiplicateur de Lagrange*) qui correspond au β qui borne la norme L2 de w. Et dans ce cadre, la résolution du problème Pb2, revient à imposer que l'espace de w ($\mathcal{H}_\beta$) est de plus petite dimension que $\mathcal{H}$ sans contrainte. **Encore une fois on combat la dimension en réduisant l'espace des fonctions où l'on cherche la solution.**

On peut imposer différents types de régularisation mais on retient essentiellement la norme L2 et la norme L1:

- **Norme L2**: $||w||_2^2 = \sum_k |w_k|^2 < \beta$, plus facile à manipuler;
- **Norme L1**: $||w||_1 = \sum_k |w_k| < \beta$ qui favorise un w « sparse » (creux) avec beaucoup de 0 (représentation parcimonieuse) et quelques composantes de grandes valeurs et sert pour la « feature selection ».

7.3.4 Le risque en termes des variables duales de w

On se rend compte assez vite que $w \in V$ l'espace engendré par les échantillons d'entrainement x_i. En effet, on peut linéairement décomposer w selon une composante V (w_1) et une composante $V^\perp$ (w_2), donc le risque empirique régularisé

$$\begin{aligned}\min \tilde{R}_r(w) &= \min \left(\frac{1}{n} \sum_{i=1}^{n} (\langle w, x_i \rangle - y_i)^2 + \lambda ||w||_2^2 \right) \\ &= \min \left(\frac{1}{n} \sum_{i=1}^{n} (\langle w_1, x_i \rangle - y_i)^2 + \lambda (||w_1||^2 + ||w_2||^2) \right)\end{aligned}$$

Le terme de régularisation va faire tendre la composante w_2 vers 0 ce qui contraint $w \in V$ et mécaniquement cela a pour conséquence que

$$\boxed{w = \sum_{i=1}^{n} \alpha_i x_i}$$

(il s'agit du *representer theorem* qui a des applications pratiques quand $n < d$). Les α_i sont les **variables duales** de w. Le risque empirique régulariser à minimiser a donc l'expression

$$\boxed{\tilde{R}_r(\alpha) = \frac{1}{n} \sum_{i=1}^{n} (\sum_{k=1}^{n} \alpha_k \langle x_k, x_i \rangle - y_i)^2 + \lambda \sum_{k,k'} \alpha_k \alpha_{k'} \langle x_k, x_{k'} \rangle}$$

$\tilde{R}_r(\alpha)$ est donc une **forme quadratique convexe** en α_k avec des coefficients qui sont les produits scalaires $\langle x_k, x_i \rangle$. Mais souvenons nous qu'en fait, il s'agit des produits scalaires *après changement de représentation* donc il faut plutôt les voir comme $\langle \phi(x_k), \phi(x_i) \rangle$. **En définitive, il n'est pas nécessaire de connaitre ϕ mais plutôt les valeurs du noyau de similarité $k(x_k, x_i)$.**

On a $\approx n^2$ coefficients (pdts scalaires) indépendamment de la dimension de $\phi(x)$ ([13]). Donc si je dois les calculer à partir de $\phi(x)$ ca peut être prohibitif si la dimension de $\phi(x)$ est grande. Surtout pour de très bonnes raisons on a envie d'avoir une dimension de $\phi(x)$ très grande (ex. plusieurs milliers). En effet, en très grande dimension, on trouve toujours

13. ndje imaginer que les composantes de $\phi(x)$ soient tous les monomes d'un certain degré obtenus à partir des composantes de x. Ex. $(x_1, x_2, x_3, x_1x_2, x_1x_3, x_2x_3, x_1^2x_2, x_1^2x_3, x_2^2x_1, x_2^2x_3, x_3^2x_1, x_3^2x_2, \dots)$

une feature (au sens large) qui est discriminante et qui arrive à séparer les 2 classes. Ce fût l'espoir des années 2000 car à l'époque on s'est dit (en caricaturant): « on a qu'à se placer en très grande dimension de structure/features! ». En plus si on calcule les produits scalaires directement par une formule analytique du noyau, on a pas d'impact sur le temps de calcul.

Mais: **tout va marcher certes sauf que si la dimension de $\phi(x)$ est trop grande, on va tomber dans l'overfitting.** Et donc en pratique on revient à des aspects très terre à terre de savoir comment trouver le bon $\phi(x)$, ou comment trouver les bonnes features.

8. Classification/Régression en grande dimension (Partie II)

8.1 La régression (modèle à noyau): la balance biais-variance

On se souvient que dans le cas de la « malédiction de la dimension », l'erreur de fluctuation et le nombre d'échantillons se comportent comme $n \approx \varepsilon^{-d}$. Le travail sur le changement de représentation $\phi(x)$ (cf. partie I) a été de diminuer drastiquement la dimension de l'espace dans lequel on cherche un estimateur $\tilde{f}$ (dans le cas déterministe où $y = f(x)$), pour ne garder comme paramètres les $d+1$ composantes de $w \in \mathbb{R}^{d+1}$, et donc $n \approx d\varepsilon^{-2}$. Donc il n'y a plus d'explosion quand d est grand. Mais ce n'est pas pour ça qu'on est tiré d'affaires pour autant, il faut aussi trouver ϕ et éviter l'overfitting.

Si on définit $x \in \Omega = \{x \in \mathbb{R}^d / ||x|| \leq 1\}$ (borné) et de même la réponse y est bornée $[-1, 1]$ (en régression), le problème de minimisation $\tilde{R}_r(w)$ et $\tilde{w} = \underset{w}{\operatorname{argmin}}\ \tilde{R}_r(w)$, c'est-à-dire que $\tilde{w}$ est la meilleure direction discriminante (minimise le risque empirique régularisé). Si on définit le « vrai » risque

$$R(w) = E_{X,Y}[r(Y, \langle w, X \rangle)]$$

qu'on a envie de comparer à $E(\tilde{R}_r(\tilde{w}))$, c'est-à-dire

$$E(\tilde{R}_r(\tilde{w})) \leq \min_{w \in \mathcal{H}} R(w) + \varepsilon^2$$

On aimerait que ε^2 diminue rapidement quand la nombre d'échantillons (n) augmente.

Théorème: Si on contraint w à être chercher dans l'ensemble $\mathcal{H}_{B\sqrt{d}} = \{w/||w||^2 < B^2 d\}$ alors

$$\boxed{\varepsilon^2 = 150\frac{B^2 d}{n}, \quad \text{et} \quad \lambda = \frac{\varepsilon}{3dB^2}}$$

Voyons à quoi on s'attend avec ce résultat. D'une part, si on régularise avec une valeur de λ, le résultat sur w est que sa norme est contrainte et cela d'autant plus que λ augmente. Donc, la taille de la classe $\mathcal{H}$ diminue ($\propto B$) quand λ augmente. Ainsi, on comprend la relation $\lambda \leftrightarrow B$. D'autre part, la première relation dit que $\varepsilon^2 \propto$ (taille de $\mathcal{H}$)$/n$; c'est à quoi on s'attendait par la formule générale. Enfin, on s'aperçoit que si on veut ε petit on ne peut choisir λ trop grand car il introduirait un terme de biais trop important.

Ainsi, dés que le nombre d'échantillons n est grand devant la taille d du vecteur ϕ alors on garantie **une erreur de fluctuation ε petite**. Mais maintenant rien ne garantie que **le terme de biais** $\min_{w \in \mathcal{H}_{B\sqrt{d}}} R(w)$ soit petit, c'est-à-dire que je vais approximer les données de training par combinaison linéaire avec $w \in \mathcal{H}_{B\sqrt{d}}$. **C'est la limite d'un tel estimateur où on fait reposer la classification/régression sur 1 seule direction w discriminante.**

8.2 Le pendant pour la classification

Par rapport à la régression vue précédemment, une complication s'ajoute: c'est la **non-convexité du problème** au première abord. En effet, dans le cas de la régression, on a une fonction convexe (car le risque est convexe) à minimiser sous contrainte. Le problème se résout parfaitement avec une descente de gradient. En classification ce n'est pas le cas, il va falloir passer par une phase pour rendre convexe le problème.

En classification en effet on dispose typiquement d'un risque $r(y, \tilde{y})$ avec l'estimateur $\tilde{y}$ d'expression:

$$r(y, \tilde{y}) = \begin{cases} 0 & si\ y = \tilde{y} \\ 1 & si\ y \neq \tilde{y} \end{cases}; \tilde{y} = \text{sgn}(\langle w, x \rangle + b)$$

Si on prend $y_i = 1$, tant que $\tilde{y}_i = -1$ (je suis du mauvais coté de l'hyperplan) alors le risque est 1, et dés que $\tilde{y}_i = 1$ (je suis du bon coté) alors le risque est 0.

Petits rappels sur la convexité:

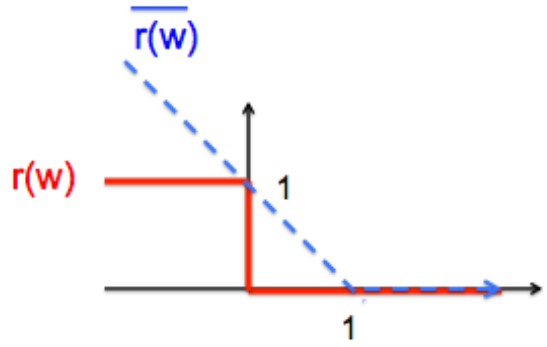

FIGURE 37 – Convexification du risque $r(w)$ par majoration $\bar{r}(w)$.

— Un ensemble Ω est convexe si

$$\forall x, y \in \Omega, \alpha \in [0,1], \alpha x + (1-\alpha)y \in \Omega$$

— une fonction $f(x)$ est convexe sur Ω si

$$\forall x, y \in \Omega, \alpha \in [0,1], f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y)$$

— L'ensemble Ω_c des x tels que $f(x) \leq c$, est un ensemble convexe si f est convexe;
— Enfin, si f a un minimum f_{min} sur Ω convexe, alors

$$\Omega_{min} = \{x / f(x) = f_{min}\}$$

est convexe; et il se réduit à 1 point si f est *strictement convexe*.

Donc **le risque n'est pas convexe** (voir figure 37 la courbe rouge). En effet, en prenant $x_1 = -1$ et $x_2 = 1$ on devrait avoir la propriété

$$\alpha \in [0,1], \ f(\alpha x_1 + (1-\alpha)x_2) \leq \alpha$$

qui est n'est pas satisfaite pour $\alpha = 1/2 - \epsilon$ ($\epsilon \ll 1$) puisque le membre de gauche vaut 1. **Pour le rendre convexe** (en le simplifiant) et garantir un ε de fluctuations petit comme on a vu pour la régression, on utilise un **majorant du risque** r comme celui de la courbe bleue sur la figure (37) (Hinge loss à l'origine des Support Vector Machine de V. N. Vapnik &

A. Chervonenkis (1963)). Ainsi, quand on obtient une borne supérieur du nouveau risque ($\bar{r}$), de facto c'est un majorant de l'ancien risque (r).

8.3 Condition de point selle de Kuhn & Tucker (1950)

Cela concerne la minimisation du risque avec une contrainte sur $||w||^2$. Soit le problème 1

$$\min_{x\in\Omega} f(x) \quad \text{et} \quad C_k(x) \leq 0 \quad (Prob.\ 1)$$

où $C_k(x)$ sont une série de contraintes. Alors considérons le lagrangien ($\lambda = (\lambda_1, \ldots, \lambda_K)$, et $\lambda_k \geq 0$)

$$L(x, \lambda) = f(x) + \sum_{k=1}^{K} \lambda_k C_k(x)$$

Une **condition suffisante** (Prop. 2) s'exprime comme ceci: **s'il existe un point selle** $(\bar{x}, \bar{\lambda})$ (figure 38) dont la définition est

$$\forall (x, \lambda) \in \Omega \times (\mathbb{R}^+)^K \quad L(\bar{x}, \lambda) \leq L(\bar{x}, \bar{\lambda}) \leq L(x, \bar{\lambda}) \quad (Prop.2)$$

alors on a les propriétés suivantes

$$\begin{cases} \bar{x} \text{ est solution de Prob. 1} \\ \text{et } \forall k, \bar{\lambda}_k C_k(\bar{x}) = 0 \end{cases}$$

La seconde propriété signifie que si la contrainte $C_k(\bar{x}) < 0$ alors $\bar{\lambda}_k = 0$, et à la frontière de la contrainte $\bar{\lambda}_k \neq 0$. **Ce résultat est fondamental en optimisation.** La démonstration est la suivante.

Soit donc un point qui est un point selle alors en prenant la première inégalité et en explicitant le lagrangien, il vient

$$f(\bar{x}) + \sum_k \lambda_k C_k(\bar{x}) \leq f(\bar{x}) + \sum_k \bar{\lambda}_k C_k(\bar{x})$$

donc

$$\forall \lambda_k \geq 0, \quad \sum_k (\lambda_k - \bar{\lambda}_k) C_k(\bar{x}) \leq 0$$

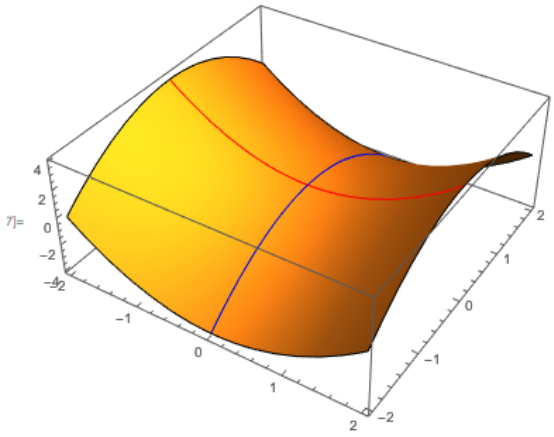

FIGURE 38 – Point selle à la croisée des courbes bleue (max) et rouge (min).

On peut donc prendre pour un k_0 fixé, $\lambda_{k_0} = \bar{\lambda}_{k_0} + 1$ et $\forall k \neq k_0 \ \lambda_k = \bar{\lambda}_k$. Ceci se traduit par

$$C_{k_0}(\bar{x}) \leq 0$$

et ceci est vrai pour tout k_0. Donc les contraintes du Prob. 1 sont satisfaites.

Autre cas particulier, on fixe pour k_0, $\lambda_{k_0} = 0$ et $\forall k \neq k_0 \ \lambda_k = \bar{\lambda}_k$. Ceci se traduit par la relation

$$\bar{\lambda}_{k_0} C_{k_0}(\bar{x}) \geq 0$$

Or, comme on a montré que $\forall k, \ C_k(\bar{x}) \leq 0$, donc comme $\bar{\lambda}_{k_0} \geq 0$ alors $\bar{\lambda}_{k_0} C_{k_0}(\bar{x}) = 0$ et cela $\forall k_0$. Maintenant prenons la seconde inégalité concernant le lagrangien et explicitons le, il vient:

$$f(\bar{x}) + \sum_k \bar{\lambda}_k C_k(\bar{x}) \leq f(x) + \sum_k \bar{\lambda}_k C_k(x)$$

Or $\forall k, \ \bar{\lambda}_k C_k(\bar{x}) = 0$, de plus $C_k(x) \leq 0$ et $\bar{\lambda}_k \geq 0$ donc $\sum_k \bar{\lambda}_k C_k(x) \leq 0$. Finalement, on constate que

$$f(\bar{x}) \leq f(x)$$

donc $\bar{x}$ et bien un minimum et il répond à Prob. 1. ■

Lorsque le problème est **convexe**, c'est-à-dire que $f(x)$ ainsi que les contraintes C_k, de

même que l'ensemble Ω dans lequel on cherche x sont convexes, alors

$$\text{Si } \exists x_0 \in \Omega / C_k(x_0) \leq 0 \Rightarrow \text{alors Prop. 2 est nécessaire.}$$

et on peut exprimer le résultat sous forme de différentielle qui est utile en optimisation convexe.

Ainsi les conditions nécessaires et suffisantes à chaque fois qu'on a un risque convexe avec des contraintes convexes (et une solution dans un espace convexe)

$$\begin{aligned} \partial_x L(\bar{x}, \bar{\lambda}) &= 0 = \partial_x f(\bar{x}) + \sum_k \bar{\lambda}_k \partial_x C_k(\bar{x}) && (14) \\ \partial_\lambda L(\bar{x}, \bar{\lambda}_k) &= C_k(\bar{x}) \leq 0 && (15) \\ \sum_k \bar{\lambda}_k C_k(\bar{x}) &= 0 && (16) \end{aligned}$$

Le but du jeu par la suite est de se retrouver dans ce cadre de convexité fusse-t'il au prix de convexifier (en majorant) le risque par exemple comme dans le cas de la classification.

9. Classification par Support Vector Machine

Il s'agit de recherches qui ont été menées dans les années 90 à 2005. Se sont des algorithmes qui fonctionnent bien (pas forcément ceux qu'on utilise en pratique) mais il y a des concepts intéressants: comment linéariser un problème, comment le rendre convexe et donc le résoudre avec des algorithmes simples et efficaces. De plus, le cadre mathématique permet de contrôler tout jusqu'au risque des estimateurs.

9.1 Le critére de marge

Donc, on se place dans le cas d'une classification de 2 classes $y = \pm 1$ avec $x \in \mathbb{R}^d$. **L'estimateur/classificateur linéaire** est donné par le signe de x par rapport à l'hyperplan (w, b) (figure 39). **Si le problème est séparable** alors l'hyperplan existe, et pour toutes les

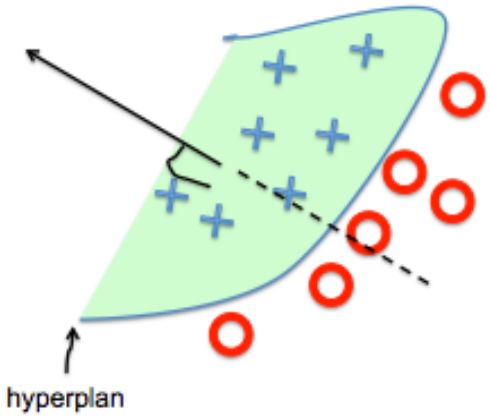

FIGURE 39 – Normal à l'hyperplan séparateur.

données d'entraînement on a

$$\forall (x_i, y_i)_{i \leq n}, \quad y_i \times (\langle w, x_i \rangle + b) \geq 0$$

($= 0$ sur l'hyperplan lui-même). Cependant, l'hyperplan n'est pas unique, comme sur l'exemple de la figure 40. Pour obtenir une solution optimale, on utilise l'idée de **Vapnik**: il faut prendre l'hyperplan qui est le plus robuste vis-à-vis de petits changements des données d'entraînement. C'est-à-dire prendre l'hyperplan qui est le « plus loin » des deux classes: x_i^+ et x_i^- (figure 41).

Notons que sur l'hyperplan $\langle w, x \rangle + b = 0$ et en dehors $\langle w, x \rangle + b = c \neq 0$. De part et d'autre du plan, on peut trouver les points x_1 et x_2 les plus proches de l'hyperplan appartenant aux deux classes, et tel que le l'hyperplan se trouve au milieu (c'est-à-dire que le point $(x_1 + x_2)/2$ est sur l'hyperplan) donc:

$$\begin{cases} \langle w, x_1 \rangle + b &= m||w|| \\ \langle w, x_2 \rangle + b &= -m||w|| \end{cases}$$

Avec $m||w||$ la distance de x_1 (x_2) au plan. On peut rescaler les x tels que la distance

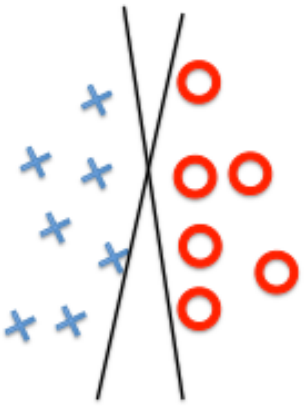

FIGURE 40 – Quelle droite (ou hyperplan en dimension d) choisir?

entre x_1 et x_2 soit égale à 2, donc

$$\boxed{||w|| = \frac{1}{m}}$$

Maximiser la marge « $2m$ » (et donc m) revient à minimiser $||w||$. Sachant que w est une combinaison linéaire des x_i. Le problème se reformule ainsi : trouver (w, b) tel que [14]

$$\boxed{y_i \times (\langle w, x_i \rangle + b) \geq 1; \text{ et } \quad \min(||w||^2)}$$

9.2 Le meilleur hyperplan possible: pénalisation

Le problème est bien du type vu au précédent cours avec une fonction à minimiser, ici la norme de w, et n contraintes $(C_i(w, b))$. Le lagrangien s'écrit alors de la façon suivante:

$$\boxed{L(w, b, \alpha) = \frac{1}{2}||w||^2 + \sum_i \alpha_i (1 - y_i \times (\langle w, x_i \rangle + b))}$$

14. NDJE: La figure 41 le b est inclus dans x comme dans les cours précédents de S. Mallat. La borne « 1 » dans l'inégalité vient de ce que les points sont au delà de la marge de part et d'autre de l'hyperplan et que l'on à renormaliser (w, b) pour faire en sorte que la marge soit $1/||w||$.

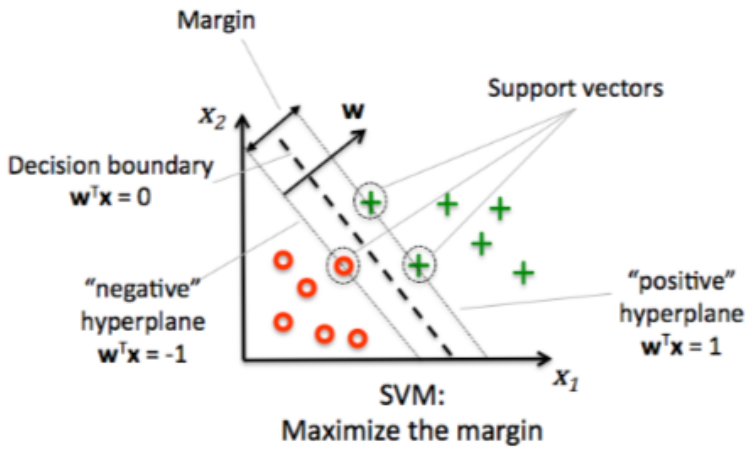

FIGURE 41 – La notion de marge (margin) et de vecteurs de support (support vectors).

Maintenant, la minimisation va bien donner le meilleur hyperplan, encore faut-il qu'il existe. La plus part du temps, il ne va pas exister de classificateur linéaire (ici l'hyperplan) qui sépare parfaitement tous les échantillons d'entraînement [15]. Dans ce cas de figure, en fait on va chercher un hyperplan qui donne l'erreur de classification minimum. Vapnik propose de définir l'erreur comme de combien faut-il déplacer les points/échantillons mal placés pour les ramener au delà de la marge pour les reclassifier correctement. Donc, si on note $\xi_i \geq 0$ la distance pour l'échantillon i qu'il faut pour le déplacer alors,

$$y_i \times (\langle w, x_i \rangle + b) \geq 1 - \xi_i \quad \text{(Soft SVM)}$$

(si $\xi_i = 0$ on parle de « Hard SVM »). Bien entendu, on veut pénaliser ces réarrangements. Donc, on reformule le problème comme ceci:

$$\boxed{\begin{cases} \text{Min}(\frac{1}{2}||w||^2 + C\sum_i \xi_i) \\ y_i \times (\langle w, x_i \rangle + b) \geq 1 - \xi_i \end{cases}}$$

15. NDJE: sauf à se mettre en dimension suffisamment grande.

9.3 La régularisation

On a déjà vu que pour faire en sorte que les fluctuations n'introduisent pas trop d'erreur, il faut pouvoir contraindre la taille de la classe (l'espace) dans laquelle on cherche les paramètres. Comment cela se traduit dans notre problème. Si on fixe (w, c) alors

$$\xi_i \geq 1 - y_i \times (\langle w, x_i \rangle + b)$$

et on veut $\xi_i \geq 0$, donc

$$\xi_i = \max(1 - y_i \times (\langle w, x_i \rangle + b), 0)$$

Donc, la minimisation à réaliser peut s'écrire en posant $\bar{y}_i = \langle w, x_i \rangle + b$ (régression linéaire)

$$\frac{1}{2}||w||^2 + C \sum_i max(1 - y_i \bar{y}_i, 0)$$

Or la fonction

$$r(x, y, w, b) = r(y, \bar{y}) = \max(1 - y\bar{y}, 0)$$

n'est autre qu'une version convexifiée du risque de classification binaire $r(y, \tilde{y})$ avec $\tilde{y} = \text{sgn}(\bar{y})$ vue au cours précédent. Donc le risque global

$$\tilde{R}(w, b) = \sum_{i=1}^{n} r(x_i, y_i, w, b)$$

est bien convexe, et le problème de minimisation se présente comme

$$\text{Min}(\tilde{R}(w, b) + \frac{1}{2}||w||^2)$$

C'est-à-dire la minimisation d'un **risque convexe régularisé par la norme de** $||w||^2$ qui permet de restreindre la taille de la classe des hypothèses sur (w, b) et donc minimise les erreurs de fluctuations. Le problème est bien posé et le calcul se déroule correctement.

9.4 Méthode du point selle

Si on se place dans le cas du Hard SVM ($\xi_i = 0$) alors le lagrangien est donné par

$$L(w, b, \alpha) = \frac{1}{2}||w||^2 + \sum_i \alpha_i (1 - y_i \times (\langle w, x_i \rangle + b))$$

Le point selle est le lieu où l'on cherche un minimum dans l'espace (w, c) et un maximum pour α_i avec $\alpha_i \geq 0$. Les gradients respectifs du lagrangien donnent

$$\begin{cases} \sum_i \alpha_i y_i = 0 & (R1) \\ w = \sum_i (\alpha_i y_i) x_i & (R2) \\ 1 - y_i \times (\langle w, x_i \rangle + b) = 0 \quad (\forall \alpha_i \neq 0) & (R3) \end{cases}$$

La seconde équation montre bien que w est une combinaison linéaire des x_i (cf. $\alpha_i y_i = \pm \alpha_i$). Si $\alpha_i = 0$ alors $1 - y_i \times (\langle w, x_i \rangle + b) < 0$ c'est-à-dire que l'échantillon est au delà de la marge donc la contrainte ne s'applique pas. Donc, le nombre de contraintes « actives » ($\alpha_i > 0$) dépend du nombre de points qui sont sur les 2 hyperplans frontières.

Finalement, l'algorithme va être piloter non pas par le nombre total d'échantillons, mais uniquement par ceux qui sont les plus proches de l'hyperplan pour lesquels $\alpha_i > 0$ et donc

$$\boxed{w = \sum_{i \in \mathcal{I}} (\alpha_i y_i) x_i}$$

D'où le vocable « Support Vectors des échantillons éléments de $\mathcal{I}$ ».

Pour résoudre le problème, on se place dans l'espace dual des multiplicateurs de Lagrange (α_i). Si on tient compte de $R1$ et $R2$, le développement du lagrangien donne

$$L(w, b, \alpha) = -\frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \langle x_i, x_j \rangle + \sum_{i=1}^{n} \alpha_i$$

à minimiser pour les α_i. Ce qui se fait par les techniques de gradient conjugué et gradient stochastique.

Pour le cas de « Soft SVM » on doit minimiser

$$\frac{1}{2}||w||^2 + C\sum_i \xi_i$$

qui se traduit par le fait que les α_i ont une contrainte supplémentaire:

$$0 \leq \alpha_i \leq C$$

9.5 Généralisation au cas non-linéaire

On a vu que l'optimisation de la marge est équivalent à une convexification du risque et une régularisation, et que finalement le problème ne dépend que des produits scalaires $\langle x_i, x_j\rangle$, et on obtient une frontière linéaire (cf. un hyperplan). On sait que pour trouver une frontière non-linéaire, il faut trouver une représentation $z = \phi(x)$ qui linéarise le problème et on a vu que tout revient à faire le changement

$$\langle x_i, x_j\rangle \rightarrow \langle \phi(x)_i, \phi(x)_j\rangle = k(x_i, x_j)$$

avec $k(x, x')$ une mesure de la similarité entre x et x' (noyau). L'optimisation est la même que dans le cas linéaire, et la solution de l'estimateur de y_i est

$$\begin{aligned}\tilde{f}(x) &= \text{sgn}(\langle w, \phi(x)_i\rangle + b) = \text{sgn}(\sum_j \alpha_j y_j \langle \phi(x), \phi(x)_j\rangle + b)\\ &= \text{sgn}(\sum_j \alpha_j y_j k(x_j, x) + b)\end{aligned}$$

Dans les années 90-2000 à souffler un vent d'optimisme. On a compris que non seulement on pouvait trouver une frontière très complexe mais aussi qu'on pouvait caractériser la complexité de la frontière dont la dimension du vecteur $\phi(x)$ peut être très grande. Voyons deux exemples: **les noyaux polynomiaux et les noyaux gaussiens**.

9.5.1 Noyaux polynomiaux

Prenons l'exemple d'un noyau ($x_0 \equiv 1$)

$$\boxed{k(x,x') = (1+\langle x, x'\rangle)^p = \left(\sum_{j=0}^{d} x_j x'_j\right)^p}$$

Que sont les $\Phi(x)$? En fait si $\vec{j} = \{j_1, \ldots, j_p\}$ avec toutes les permutations

$$\Phi(x) = \left\{\varphi_{\vec{j}}(x)\right\}_{\vec{j}=(0,\ldots,d)^p}$$

et

$$k(x,x') = \sum_{\vec{j}=(0,\ldots,d)^p} \prod_{i=1}^{p} x_{j_i} x'_{j_i} = \sum_{\vec{j}=(0,\ldots,d)^p} \prod_{i=1}^{p} x_{j_i} \prod_{i=1}^{p} x'_{j_i} = \langle \Phi(x), \Phi(x')\rangle$$

donc $\dim\Phi(x) = (1+d)^p = d'$, et

$$\boxed{\varphi_{\vec{j}}(x) = \prod_{i=1}^{p} x_{j_i}}$$

c'est-à-dire **tous les monômes de degré** p. L'estimateur $\tilde{f}(x)$ (en régression) s'exprime comme

$$\tilde{f}(x) = \sum_{\ell=1}^{n} \alpha_\ell y_\ell \sum_{\vec{j}=(0,\ldots,d)^p} \prod_{i=1}^{p} x_{j_i}^{(\ell)} \prod_{i=1}^{p} x_{j_i}$$

Donc $\tilde{f}(x)$ est un polynôme arbitraire de degré p. L'astuce est qu'il n' a pas été nécessaire de fitter ces monômes de degré p (on a fitté les variables duales, c'est-à-dire α_i) et donc p peut être très grand et en dimension d, donc $d' = (d+1)^p$ termes! Imaginez avec $d = 10^6$ pour une image et $p = 5\ldots$ Alors qu'en fait toutes les sommes sont bornées à n, et donc le nombre de produits scalaires est $n(n+1)/2$. **Donc la technique semble être très puissante.** Est-ce un miracle, ou bien il y a un truc qui cloche? où est le prix à payer ? Comme on peut l'imaginer, c'est dans les erreurs de fluctuations: **l'overfitting est très vite un problème: on a beaucoup trop de features et peu d'échantillons.** La raison du succès est non seulement que l'algorithme est stable mais aussi qu'on peut **toujours** se placer dans un cas où l'hyperplan répond au problème sur les échantillons de traîning (ajustement en régression, ou séparation en classification). Cela vient de la proposition

suivante:

Proposition : si les $\{x_i\}_{i\leq n}$ sont linéairement indépendants (et $d' \geq n$) alors

$$\forall y_i,\ \exists w/\ \langle w, x_i\rangle = y_i$$

La matrice dont les lignes sont les vecteurs x_i est inversible donc w existe. **Il suffit donc de se placer en suffisamment grande dimension.** On peut même faire un peu mieux. Si on prend un échantillon au hasard (noté x_1) pris comme référence des coordonnées, alors

$$\{x_1 - x_i\}_{2\leq i\leq n}$$

sont linéairement indépendants ($d' \geq n-1$) alors

$$\forall y_i,\ \exists (w,b)/\ \langle w, x_i\rangle + b = y_i$$

En effet, on se ramène au cas du théorème

$$\begin{cases} \langle w, x_i - x_1\rangle = y_i - y_1 \\ \langle w, x_1\rangle + b = y_1 \end{cases}$$

La première égalité indique que w existe, et la seconde fixe b.

Donc en fait on peut toujours trouver des $\phi(x)$ de dimension suffisante pour trouver un hyperplan qui convient. En revenant dans l'espace d'origine des x la frontière devient non-linéaire. Mais, l'erreur de fluctuations est

$$\varepsilon^2 \approx \frac{\log|\mathcal{H}|}{n} = \frac{d'}{n} = \frac{(1+d)^p}{n}$$

d'où une catastrophe, car l'erreur est potentiellement énorme. Pour la contrôler il faut absolument que $d' \ll n$. Donc soit dés le départ d est petit, soit il faut faire de la « feature reduction », quitte à ce que par la suite on augmente légèrement la dimension de d à d'.

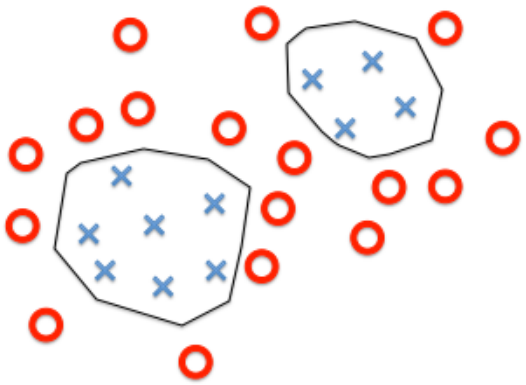

FIGURE 42 – Clusters isolés non convexes, frontière non-linéaire et non convexe.

9.5.2 Noyau gaussien

Imaginons que nous ayons un problème du type de la figure 42. La frontière est clairement non convexe et ne peut être générée par un noyau polynomiale. Le noyau que nous allons envisager est définit par la Radial Basis Functions (**RBF**)

$$\boxed{k(x_1, x_2) = e^{-\frac{1}{2\sigma^2}||x_1 - x_2||^2}}$$

Quel est le $\Phi(x)$ dans ce cas? On montre qu'une solution s'écrit comme

$$\varphi_{\vec{k}}(x) = e^{-\frac{1}{2\sigma^2}||x||^2} \prod_{\vec{j}=(0,\dots,d)^k} x_{j_i}$$

mais il y a une infinité de k, donc $\Phi(x)$ **est de dimension infinie.** Or, on remarque que ce noyau se calcule très simplement, alors que si on passait par l'étape des $\varphi_{\vec{k}}(x)$ il aurait fallu une infinité d'opérations!

Si $||x_1 - x_2|| \ll \sigma$ alors $k(x_1, x_2) \approx 1 - ||x_1 - x_2||^2/(2\sigma^2)$, donc le noyau se comporte comme un distance euclidienne locale et on va pouvoir déterminer un classificateur (régression) linéaire locale (cf. la figure 43). De proche en proche on va donc définir une frontière qui va pouvoir être de n'importe quelle structure. Mais quel est le prix à payer ? Toujours la balance biais-variance. Si σ est trop petit, il n'y a pas d'échantillons (**rappel:**

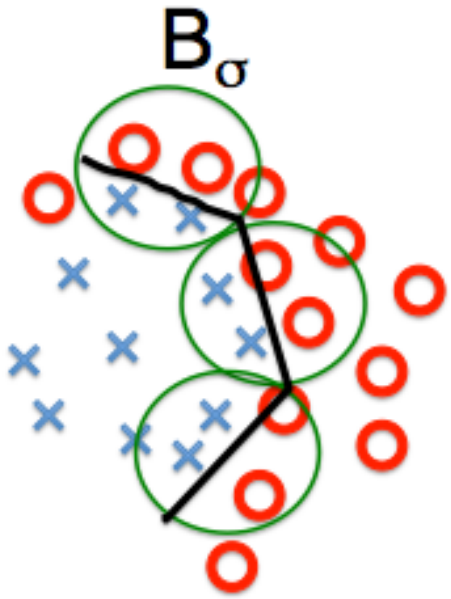

FIGURE 43 – Par application progressive d'un filtre de taille σ on linéarise localement, et in fine on obtient une frontière complexe.

on est en grande dimension, le dessin en 2D est tout à fait trompeur). Donc σ doit être suffisamment grand pour que la volume de la boule soit du même ordre de grandeur que la taille de l'espace, le classificateur gaussien devient un classificateur linéaire ! et alors on ne peut rendre compte de détails. **On détermine σ par cross-validation** en pratique et on peut obtenir non pas des classificateurs linéaires mais un peu courbe localement. Mais il n'y a pas de miracle.

9.6 Commentaires de JE.

Avant de clore ce cours, je voulais revenir sur la technique SVM. On peut l'aborder d'une manière totalement différente. Soit donc des $\{x_i, y_i\}_{i\leq n}$ avec $y_i \in \{0, 1\}$, l'idée est un modèle probabiliste paramétré linéaire (j'essaye de garder des notations en lien avec le cours de S. Mallat):

$$h_w(x) = P(y = 1|x; w) = 1 - P(y = 0|x; w)$$

Donc la probabilité que y soit 0 ou 1 est donnée par

$$P(y|x; w) = h_w(x)^y(1 - h_w(x))^{1-y}$$

On forme alors le likelihood suivant en supposant que tous les échantillons sont indépendants

$$\mathcal{L}(w) = \prod_{i=1}^{n} P(y_i|x_i; w)$$

dont on cherche un maximum pour trouver la valeur de w. Pour utiliser les algorithmes de minimisation on utilise plutôt $-\log \mathcal{L}(w)$ que l'on normalise par le nombre d'échantillons n. Ainsi, on forme la fonction

$$J(w) = -\frac{1}{n}\sum_{i=1}^{n} \{y_i \log(h_w(x_i)) + (1 - y_i) \log(1 - h_w(x_i))\}$$

Il faut maintenant donner une expression (modèle) pour $h_w(x)$ (c'est-à-dire $P(y = 1|x; w)$). En principe en lien avec le cours de S. Mallat on chercherait quelque chose comme

$$h_w(x) = \frac{1}{2}(1 + \mathrm{sgn}(\langle x, w\rangle))$$

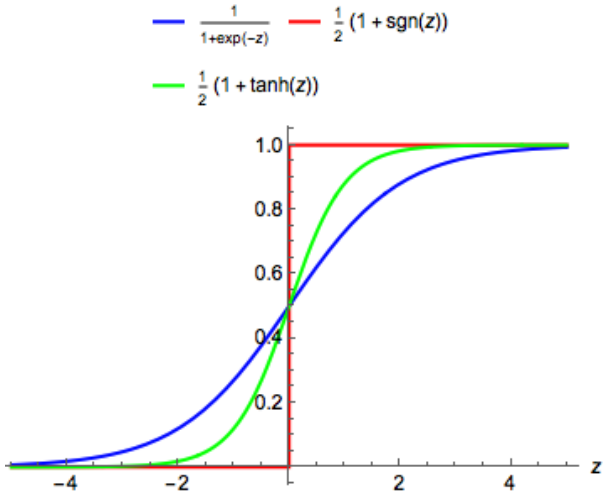

FIGURE 44 – Plusieurs fonctions $h_w(z)$ pour réaliser la minimisation du likelihood.

(nb. où le b est intégré dans le produit scalaire) Mais cette expression ne peut convenir comme argument du log, c'est la raison pour laquelle D. Cox (1958) introduit la *fonction logistique* (figure 44) dans ce type de problème

$$h_w(x) = \frac{1}{1 + e^{-\langle x,w\rangle}}$$

Mais on aurait tout aussi bien pu introduire la *fonction* "tanh"

$$h_w(x) = \frac{1}{2}(1 + \tanh(\langle x, w\rangle))$$

Pour finir sur la **régression logistique**, on ajoute une régularisation pour corriger l'overfitting via l'introduction d'une pénalité de type norme $L2$, c'est-à-dire que $J(w)$ devient

$$\boxed{J_r(w) = -\frac{1}{n}\sum_{i=1}^{n}\{y_i \log(h_w(x_i)) + (1-y_i)\log(1-h_w(x_i))\} + \lambda\sum_{j=1}^{d}|w_i|^2}$$

Pour l'étape « non-linéaire » à noyau, je faisais remarquer que l'expression précédente pouvait ce généraliser selon (dans la littérature λ est sorti des $\{\}$ et on remplace $1/(2\lambda) =$

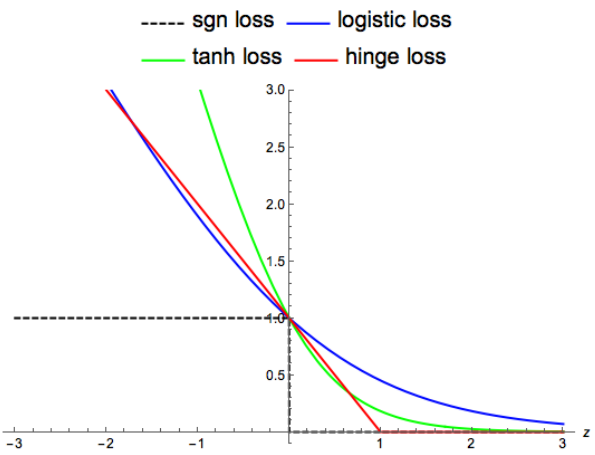

FIGURE 45 – Fonctions de risque/coût candidates.

C):

$$\boxed{J_r(w) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{n}\{y_i \text{cost}_1(\langle x_i, w\rangle) + (1-y_i)\text{cost}_0(\langle x_i, w\rangle)\}}$$

Les fonctions $\text{cost}_1(z)$ pour l'expression logistique et tanh sont montrées sur le graphe 45 en bleu et vert respectivement. Mais prenons le cas de la courbe rouge qui exprime un **effet de seuil**, dans le cours de S. Mallat on parle de « hinge loss » associée à la marge étant donné l'interprétation géométrique qu'il en donne

$$\text{cost}_1^{hinge}(\langle x_i, w\rangle) = \begin{cases} 1 - \langle x_i, w\rangle & \langle x_i, w\rangle < 1 \\ 0 & \langle x_i, w\rangle \geq 1 \end{cases}$$

C'est trois fonctions sont des majorants de la fonction « signe » présentée en noir sur la figure 45 comme dans le cours de S.Mallat. Elles permettent de convexifier le problème [16].

Dans le cas de la minimisation de $J_r(w)$, pour interpréter l'origine du choix cost_1^{hinge}, je prenais le cas où $C \gg 1$. Le problème devient alors une minimisation

$$\min_w \left(\frac{1}{2}||w||^2\right)$$

16. nb. on a rescalé les fonctions de coût logistic/tanh par $1/\log(2)$ pour ce faire, cela ne contribue pas à la minimisation.

sous **les contraintes** (qui annulent le terme en C) suivantes

$$\begin{cases} \langle x_i, w\rangle \geq 1 & \text{si } y_i = 1 \\ \langle x_i, w\rangle \leq -1 & \text{si } y_i = 0 \end{cases}$$

Ce problème est en tout identique à celui mentionné par S. Mallat adapté pour $y_i = \{0, 1\}$. **Donc, la minimisation de $J_r(w)$ donne le même résultat que SVM dans le cas $C \gg 1$.**

10. La méthode des gradients et une mise en bouche sur les réseaux de neurones

10.1 Optimisation par descente de gradient

Les algorithmes de minimisation ne sont pas simplement des outils, ils sont des conditions de possibilité. Possibilité de faire de l'apprentissage sous entendu. On a donc le problème suivant

$$\min_{w\in\mathcal{H}} R(w)$$

et pour ce faire on calcule les gradients:

$$\vec{\nabla} R(w) = \left(\frac{\partial R(w)}{\partial w_k}\right)_{k\leq d} \equiv \partial_w R(w)$$

et par itération en partant d'une valeur initiale $w = w_0$, on obtient

$$\boxed{w_{k+1} = w_k - \eta \vec{\nabla} R(w_k)}$$

En effet, la dérivée de $R(w)$ par rapport à n'importe quelle direction $\vec{n}$ (unitaire) est donnée par

$$\frac{\partial R(w)}{\partial \vec{n}} = \vec{\nabla} R(w).\vec{n}$$

Donc la direction du gradient maximise la dérivée: c'est la plus grande **montée** locale. L'existence des gradients en tout point est donnée par la régularité (Lipschitz au moins) de $R(w)$ et le fait qu'il y a ou pas de minimum local est lié à la convexité de $R(w)$. Et la

vitesse de convergence est donnée par le Hessien (dérivée seconde).

10.1.1 Risque quadratique

$$R(w) = \frac{1}{2}\langle \mathbf{A}w, w\rangle + \langle b, w\rangle$$

avec $\mathbf{A}$ symétrique définie positive (*cf. typiquement une matrice de covariance*). Donc

$$\partial_w R(w) = \mathbf{A}w + b$$

Une condition nécessaire pour que w^* soit la solution, c'est que

$$\partial_w R(w^*) = 0 \Rightarrow w^* = -\mathbf{A}^{-1}b$$

Ainsi la vitesse de convergence peut être étudiée en regardant l'écart à chaque étape entre w_k et w^*. On a

$$w_{k+1} - w^* = (\mathbf{I} - \eta\mathbf{A})(w_k - w^*)$$

Ainsi en norme

$$||w_{k+1} - w^*|| \leq ||(\mathbf{I} - \eta\mathbf{A})|| \times ||w_k - w^*||$$

Si on note $||(\mathbf{I} - \eta\mathbf{A})|| = \rho$ alors

$$||w_{k+1} - w^*|| \leq \rho^k ||w_0 - w^*||$$

La convergence est assurée si $\rho < 1$ et on veut la plus petite possible en jouant sur η. Or,

$$||(\mathbf{I} - \eta\mathbf{A})|| = \max(|1 - \eta\sigma_i|) = \max(|1 - \eta\sigma_{min}|, |1 - \eta\sigma_{max}|)$$

Ainsi en prenant

$$\eta = \frac{2}{\sigma_{max}^2 + \sigma_{min}^2}$$

on a le meilleur ρ

$$\rho = \frac{\sigma_{max}^2 - \sigma_{min}^2}{\sigma_{max}^2 + \sigma_{min}^2}$$

On voit que si on conditionne la matrice A tel que $\sigma_{max} \approx \sigma_{min}$ on obtient une

convergence rapide. Toute cette discussion ce généralise en prenant **A** comme étant la matrice Hessienne (des dérivées secondes) du risque (on rend quadratique le problème localement).

10.1.2 Batch vs Stochastic gradient

Dans le cas d'apprentissage à n échantillons, le risque empirique est calculé comme moyenne de risque sur l'ensemble des échantillons et n peut être très grand. Or à chaque étape w_{k+1} nécessite le calcul

$$\partial_w \tilde{R}(w) \propto \sum_{i=1}^{n} \partial_w r(y_i, x_i; w)$$

donc n opérations. On parle de **Batch de n éléments**. Pour le **gradient stochastique**, la stratégie prise est de calculer un estimateur de la direction de stepping calculé avec **1 élément**:

$$w_{k+1} = w_k - \eta \partial_w \tilde{r}(y_i, x_i; w_k)$$

Or, on veut minimiser le vrai risque (*indépendamment de la technique choisie*), à savoir

$$\min_w R(w) = \min_w E_{Y,X}(r(Y, X, w))$$

ce qui est garantie aussi par les 2 méthodes.

Quel est l'avantage? Si on a des redondances dans les échantillons, la méthode du Batch n'est pas efficace puisqu'on effectue des calculs superflus. Dans le cas Stochastique, les échantillons redondants vont être utilisés à différentes étapes (k) de l'algorithme. Par contre le G. Stochastique est très bruité contrairement au G. Batch, cela se traduit par une vitesse de convergence différentes. Dans un cas fortement convexe (cf. le Hessien est bien conditionné):

$$E(\tilde{R}(w_k) - \min_w \tilde{R}(w)) \approx \begin{cases} O(1/k) & \text{(G. stochastique)} \\ O(\rho^k) & \text{(G. batch)} \end{cases}$$

Donc la méthode « Gradient stochastique » est plus rapide à petit nombre d'itérations, mais le « Gradient batch » converge plus vite vers la solution à grands nombres

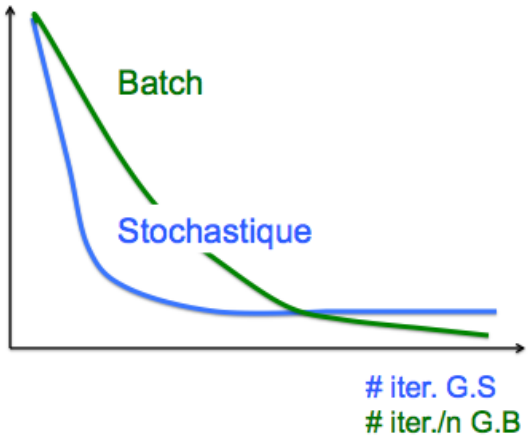

FIGURE 46 – Schéma qui indique la vitesse de convergence de la descente du gradient stochastique et du gradient par batch.

d'itérations. Schématiquement cela se traduit par la figure 46. Des méthodes essayent de combiner les avantages des deux méthodes. Voir par exemple les travaux de F. Bach et al. (SAG method) [17]

10.2 La représentation des données $\phi(x)$ et introduction NN

10.2.1 Introduction: que fait 1 neurone et 1 réseau de neurones?

Jusqu'à présent la **représentation** $\phi(x)$ est **séparée** de l'opération de **classification/régression**. On fait un choix *a priori* par exemple du type de noyau (ex. polynômial, gaussien) et de ses paramètres internes (ex. le degré p, la largeur σ), puis dans le cas d'une classification, on donne un estimateur linéaire:

$$\tilde{h}(x) = \langle w, \phi(x) \rangle + b$$

La question en suspend est : comment choisir $\phi(x)$**?** De deux choses l'une, soit on a des renseignements sur les données et on peut donc faire un « choix éclairé »; soit on

17. voir https://www.di.ens.fr/%7Efbach/Defazio_NIPS2014.pdf.

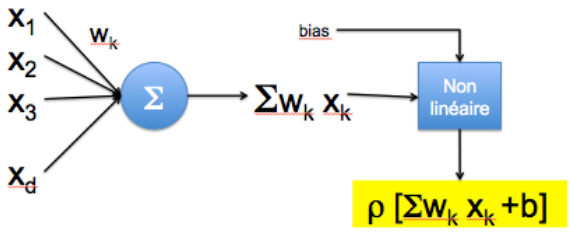

FIGURE 47 – Opérations effectuées par 1 neurone.

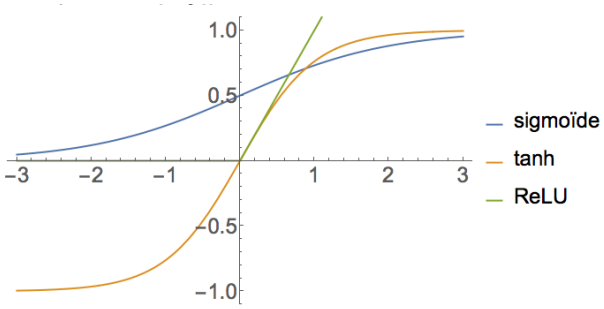

FIGURE 48 – Fonctions d'activation (ρ) d'un neurone.

fait apprendre à l'algorithme la bonne représentation. **Les réseaux de neurones (noté par la suite soit NN: Neural Net, ou MLP: Multi Layer Perceptron) apprennent en même temps la représentation ET le classificateur.** La représentation est une **cascade de classificateurs linéaires** que l'on passe dans des **filtres non-linéaires** (figure 47). Pour ce qui concerne la fonction $\rho(x)$ non-linéaire plusieurs choix ont été/sont utilisés: trois typiques (avec ReLU: rectified linear unit $= max(0, x)$) sont présentés sur la figure 48. Le résultat d'un classificateur linéaire (w, b) sépare en deux classes, et la fonction ρ dans le cas (sigmoïde/tanh) va réaffirmer le choix de classe avec cependant une zone de transition « floue »; dans du ReLU en dessous de l'hyperplan on a une classification et au dessus une régression linéaire. Dans les deux cas le neurone opère une séparation de l'espace.

Un MLP (figure 49) construit en cascade une représentation $\phi(x)$ et une régression linéaire finale avec éventuellement une classification. Si x_0 est l'entrée, avec 2 couches cachées et 1 régression finale et on différentie les opérations non-linéaires à chaque étape

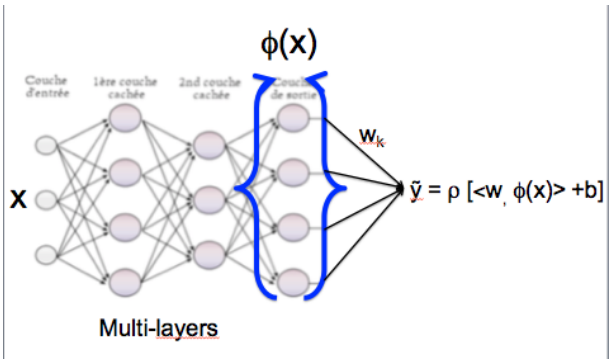

FIGURE 49 – Le réseau produit à la fin la représentation $\phi(x)$ ainsi que la classification finale.

d'activation des neurones:

$$\tilde{y} = \rho_3 \left[W_3 \rho_2 \left[W_2 \rho_1 [W_1 x_0 + b_1] + b_2 \right] + b_3 \right]$$

L'enjeu est donc de faire **une optimisation de tous les neurones**, donc d'apprendre la représentation $\phi(x)$ tout en opérant la régression finale. On va minimiser un risque comme on l'a fait jusqu'à présent, mais maintenant le nombre de paramètres va être très grand. Comment comprendre les architectures des MLP. C'est ce qu'on fait Y. LeCun, G. Hinton, Y. Bengio et d'autres en utilisant des convolutions, des informations sur les entrées pour réduire les connexions.

10.2.2 Réseau à 1 couche cachée

Dans le cas de 1 couche cachée on a un résultat important. Donc, soit le réseau de la figure 50.

$$\tilde{h}(x) = \tilde{y} = \rho \left[\sum_{k=1}^{M} \alpha_k \ \rho \left[\sum_{m=1}^{d} w_{k,m} x_m + b_k \right] + b \right] \approx f(x)$$

Petit rappel: quand on fait une classification on définit une classe d'approximation

$$\mathcal{H} = \{ h_w / \text{paramètres de l'algo(w)} \}$$

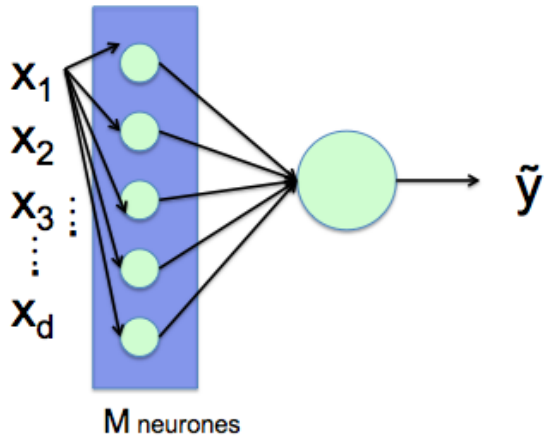

FIGURE 50 – Réseau à 1 couche cachée de M neurones.

et on fait une minimisation d'un risque empirique éventuellement régularisé

$$\min_{h\in\mathcal{H}} \tilde{R}(h_w)$$

Mais $\tilde{y} = \tilde{h}(x)$ est-il/elle proche de $y = f(x)$. La classe de $\mathcal{H}$ doit être assez grande pour pouvoir fitter n'importe quelle fonction mais bien entendu pas trop grande pour éviter l'overfit.

Dans le cas de 1 seule couche cachée a-t'on suffisamment de paramètres? En fait on peut se demander quelle est la famille de fonctions engendrées par ce type d'architecture. Quelle est la régularité de $f(x)$? Dans ce cas la dernière non-liéarité ne change pas la classe (régularité) et donc on peut se focaliser sur la représentation $\phi(x)$ telle que

$$\phi(x) = \sum_{k=1}^{M} \alpha_k \ \rho\left[\sum_{m=1}^{d} w_{k,m}x_m + b_k\right] = \sum_{k} \alpha_k \ \rho\left[\langle w_k, x\rangle + b_k\right]$$

En d'autres termes si on laisse libres (α_k, w_k, b_k) quel espace génère-t'on ? La réponse est simple et spectaculaire : **toute fonction continue peut être approximée avec un NN à 1 couche cachée!** [18].

18. nb, S. Mallat parle d'un NN 2 couches

Théorème (1989-93): Si $\rho \in C(\mathbb{R})$ (continue) et ce n'est **pas un polynôme**, alors si $f \in C(\mathbb{R}^d)$, f est approximable avec n'importe quelle précision par un NN-1 couche cachée (NN-1hl) à savoir que

$$\forall \varepsilon > 0, \forall K(\text{compact}) \subset \mathbb{R}^d \Rightarrow \exists \phi \ (\text{NN} - 1\text{hl}) \ / \ \max_{x \in K} |f(x) - \phi(x)| < \varepsilon$$

La démonstration sera faite l'an prochain. Mais ce résultat même s'il est spectaculaire, ne va pas résoudre notre problème, car imaginer que **la taille de la couche cachée M explose en dimension d** alors on aura pas gagner: **l'erreur de fluctuations explose** comme la taille de $\mathcal{H}$.

Prenons un exemple pour comprendre pourquoi le théorème n'apporte pas de solution générale à la maitrise de la dimensionalité. Donc soit $\rho(x) = e^{ix}$ (ou $\cos(x)$) c'est-à-dire le noyau des séries de Fourier.

$$\phi(x) = \sum_{k=1}^{M} \alpha_k e^{i(\langle w_k, x \rangle + b_k)} = \sum_{k=1}^{M} \alpha_k e^{ib_k} e^{i\langle w_k, x \rangle}$$

Si on se place sur le compact $x \in [0,1]^d$ alors on retrouve une décomposition en série de Fourier que l'on a vue

$$\phi(x) = \sum_{w_k \in \mathbb{Z}^d} \hat{\phi}(w_k) e^{i2\pi \langle w_k, x \rangle}$$

Si la régularité de la fonction est Lipschitz, on garantie une décroissance des coefficients de Fourier et on peut couper la série

$$0 \leq |w_k| \leq C$$

Mais le nombre de w_k croit selon C^d, **on retombe dans le piège de la malédiction de la dimension**.

Ce résultat se généralise à n'importe quelle fonction continue ρ (sauf polynôme), et on montre que le nombre de paramètres (de neurones) explose. Donc le NN à 1 couche cachée approxime certes n'importe quelle fonction continue mais au prix de l'éventuelle explosion du nombre de paramètres en grande dimension d.

L'enjeu des architectures est donc de résoudre le problème de la dimensionalité tout en pouvant approximer une large classe de fonctions avec une régularité cachée que le réseau

de neurones sait capturer. Le pourquoi ça marche n'est pas encore compris totalement.