ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DE LA
MAÎTRISE EN GÉNIE ÉLECTRIQUE
M.Ing.

PAR
VINCENT BERTHIAUME

ANALYSE DE CARACTÈRES MANUSCRITS PAR LA TRANSFORMÉE EN ONDELETTES

MONTRÉAL, LE 12 SEPTEMBRE 2002

CE PROJET A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

M. Christian Gargour, directeur de projet
Département de génie électrique à l'École de technologie supérieure

M. Mohamed Cheriet, codirecteur
Département de génie de la production automatisée à l'École de technologie supérieure

Mme Rita Noumeir, présidente du jury
Département de génie électrique à l'École de technologie supérieure

M. Tien Bui, professeur
Department of computer science à l'université Concordia

IL A FAIT L'OBJET D'UNE PRÉSENTATION DEVANT JURY ET PUBLIC

LE 12 MARS 2002

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

ANALYSE DE CARACTÈRES MANUSCRITS

PAR LA TRANSFORMÉE EN ONDELETTES

Vincent Berthiaume

SOMMAIRE

L'extraction des caractéristiques est l'une des composantes principales de tout système de reconnaissance de formes. Ce mémoire porte sur l'extraction et l'utilisation des caractéristiques issues d'une transformée en ondelette appliquées à la reconnaissance de caractères. L'objectif poursuivi consiste à proposer et à étudier de nouvelles variantes de méthodes de traitement de ce type de caractéristiques.

Dans ces méthodes, nous avons fait usage de la transformée de Fourier suivie de la transformée en ondelettes. Nous avons extrait les caractéristiques les plus discriminantes et les avons validées à l'aide de deux types de classifieurs : 1NN et SVM avec validation. Des résultats intéressants ont été obtenus, indiquant que les variantes considérées de ces familles de caractéristiques sont prometteuses.

HANDWRITTEN CHARACTER ANALYSIS

BY WAVELET TRANSFORM

Vincent Berthiaume

ABSTRACT

Feature extraction is one of the principal componants of any pattern recognition system. This thesis focuses on extraction and use of features coming from a wavelet transform applied on character recognition. The followed objective consists of proposing and studying new variants of processing methods of this feature type.

In these methods, we have used Fourier transform followed by wavelet transform. We have extracted the most discriminant features and have validated them by the help of two classifier types : 1NN and SVM with validation. Interesting results have been obtained, indicating that considered variants of these feature families are promising.

REMERCIEMENTS

La réalisation de ce mémoire a été rendue possible grâce à plusieurs personnes que je dois remercier. Ainsi, je dois d'abord remercier spécialement les directeurs de ce mémoire, Messieurs Christian Gargour et Mohamed Cheriet, qui m'ont proposé le sujet très intéressant de ce mémoire et m'ont soutenu pendant la durée de celui-ci. Lors de cette durée, ils ont proposé leurs idées et ont été ouverts aux miennes.

Je dois ensuite remercier Madame Rita Noumeir et Monsieur Tien D. Bui pour avoir fait partie du jury. Ils ont du lire la première version aride de ce mémoire et m'ont bien conseillé pour ma présentation.

Je dois ensuite remercier mes parents, qui m'ont soutenu financièrement et m'ont encouragé pour les études de maîtrise.

Finalement, je dois remercier mon collègue Nedjem-Eddine Ayat, qui m'a permis d'enrichir ce mémoire en produisant d'autres résultats expérimentaux à partir de mes données.

TABLES DES MATIÈRES

Page

LISTE DES ABRÉVIATIONS ET DES SIGLES

BOC	base orthonormale complète
CPO	coefficient de paquets d'ondelette
DF	descripteur de Fourier
DFO	descripteur de Fourier-ondelettes
DO	descripteur d'ondelette
DKL	divergence de Kullback-Leibler
i.e.	c'est-à-dire
IMO	invariant de moments d'ondelettes
IMPO	invariant de moments de paquets d'ondelettes
LDB	*local discriminant basis*
MPO	moments de paquets d'ondelettes
RC	représentation complète
ROC	représentation orthonormale complète
SDS	sélection directe séquentielle
SI	sélection individuelle
TF	transformée de Fourier
TO	transformée en ondelettes
§	section
δ	fonction d'impulsion
$[\{..\}]$	vecteur composé des éléments de $\{..\}$
$\lvert\{..\}\rvert$	taille de $\{..\}$
$\lvert\cdot\rvert$	module de $\cdot$
$\lVert\cdot\rVert$	norme de la fonction $\cdot$
$L^2(\{..\})$	ensemble des fonctions de carrés sommables sur $\{..\}$
$\mid$	tel que
$\{..\}\times\{\cdots\}$	produit cartésien de $\{..\}$ avec $\{..\}$
$\{..\}\setminus\{\cdots\}$	$\{..\}$ sans les éléments de $\{..\}$
$.\equiv\cdot$	. défini comme $\cdot$
$.\overset{(...)}{=}\cdot$	équations (..) impliquent que $.=\cdot$

LISTE DES FIGURES

Page

INTRODUCTION

De nos jours, les ordinateurs exécutent certaines tâches qui l'étaient autrefois par l'être humain, ce qui permet d'économiser des coûts de main-d'œuvre. Les paramètres d'une tâche à exécuter à tout instant donné peuvent devoir dépendre du contexte à cet instant. Pour fixer correctement ces paramètres, l'ordinateur doit donc connaître l'information sur le contexte. Cette information peut être envoyée par une personne via le clavier de l'ordinateur. Maheureusement, personne n'a accès à un ordinateur en tout lieu et à tout moment et tout le monde n'est pas aussi rapide au clavier qu'à la main pour écrire.

Pour contourner ce problème, l'information peut être écrite sur papier puis au moment où cela est possible, transférées sur ordinateur. Ce transfert peut être effectué par une personne via le clavier ou par *un système de reconnaissance de caractères*. Un système de reconnaissance de formes a pour fonction d'assigner à toute forme sa *classe* d'appartenance.

Toute information consomme de la mémoire. Si l'information sur les images de caractères doit être conservée en mémoire, un avantage de reconnaître les caractères est que l'on peut ne conserver que la fraction de l'information portant sur leur classe d'appartenance et ainsi économiser de la mémoire. Si en plus, cette information doit être lue et/ou traitée plusieurs fois, un autre avantage est l'économie de temps de traitement. Ainsi, lorsque l'information à traiter est présente en grande quantité, son transfert par système de reconnaissance optique est plus rentable que par une personne via le clavier.

Dans un système de reconnaissance *optique* de formes (figure 1), un capteur d'images échantillonne l'image qu'il reçoit d'une scène, puis un ordinateur localise des formes dans cette image, extrait de l'information sur ces formes et leur assigne leurs classes à partir de cette information.

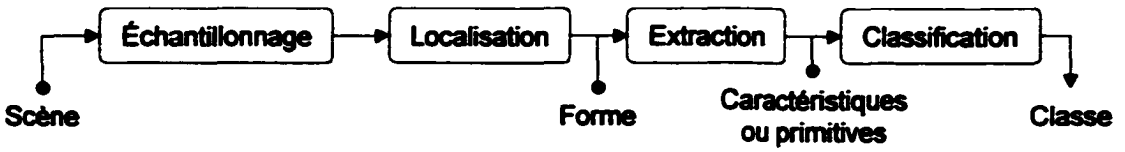


Figure 1 Structure d'un système de reconnaissance de formes

Les systèmes de reconnaissance de formes sont catégorisés selon l'approche qu'ils utilisent : approche structurelle, statistique, appartenance floue et appariement de masques. Dans ce mémoire, seule l'approche statistique est considérée. Avec l'approche structurelle, les éléments d'information extraits des formes sont appelés *primitives*. Avec l'approche statistique et l'appartenance floue, ces éléments sont appelés *caractéristiques*.

Dans l'image d'une forme, la seule information pertinante est celle portant sur sa classe, que nous appelerons information *discriminante*. Le but des caractéristiques est de ne contenir que l'information discriminante, ce qui est pratiquement improbable. Elles contiennent aussi de l'information non discriminante, et une fraction de celle-ci peut nuire à la reconnaissance, car elle porte sur des paramètres aléatoires. Dans le cas où la forme est un caractère manuscrit, ces paramètres sont les suivants : largeur du tracé, bruit (du au support d'écriture et à la numérisation), paramètres locaux (qui dépendent du scripteur qui écrit et du moment où il écrit : prononciation des coins, longueur des branches, des dépassements, présence de boucles ou de trous dans le tracé) et globaux (position, taille, inclinaison et orientation des caractères).

En reconnaissance de caractères, il existe à ce jour plusieurs familles courantes de caractéristiques [1,2], quelques unes étant listées ci-dessous. Dans cette liste, le terme *squelette* signifie le tracé aminci à la largeur d'un pixel, *graphe* signifie une approximation simplifiée du squelette, *zonage*, la division de l'image du caractère en zones contigues et *profil*, la portion du contour visible dans une direction d'observation.

Quelques familles courantes de caractéristiques :

- caractéristiques après zonage :
  - histogramme des angles du contour dans chaque zone ;
  - longueur du squelette/graphe dans chaque zone ;
  - moyenne de niveau de gris dans chaque zone ;
- caractéristiques géométriques et topologiques du squelette/graphe :
  - directions du squelette/graphe selon le code de Freeman ;
  - nombre de boucles/ demi-cercles ;
  - nombre de points terminaux/isolés ou de croisement/ branchement ;
- paramètres de splines d'approximation du contour ;
- positions des points sur un profil donné et fonctions de ces positions ;
- niveaux de gris des points ;
- histogramme des pixels noirs sur chaque ligne et colonne ;
- *loci* i.e. nombre de croisements des quatre côtés d'un point quelconque ;
- transformées globales/locales du contour/squelette/graphe/image.

Parmi ces familles courantes, celles considérées dans ce mémoire sont les *transformées*. Les transformées possèdent les propriétés pratiques suivantes :

- possibilité de *représentations complètes* (RC) de tout signal, i.e. vecteurs de valeurs contenant toute l'information du signal ;
- possibilité de caractéristiques peu sensibles au bruit ;

Le but des transformées est d'obtenir une RC d'entropie moindre que le signal, i.e. de concentrer l'information totale du signal dans moins de composantes de la RC. Une des propriétés désirées de toute RC est l'*orthonormalité*. Dans une RC orthonormale (ROC), les composantes ne partagent aucune information semblable sur le signal.

Une transformée est *globale* lorsque l'information extraite du signal est diluée tout le long de celui-ci. Or, l'information pertinante dans un même signal peut être concentrée en régions locales et de tailles variées. Pour extraire l'information sur chacune de ces régions, on utilise plutôt une transformée à localité variable : la *transformée en ondelettes* (TO), ou sinon la *transformée en paquets d'ondelettes* (TPO), qui permet une plus grande variété de RC que la TO.

Depuis un certain temps, la TO est utilisée pour la compression de signaux (réduction de la taille de leurs représentations), l'élimination du bruit et la détection de discontinuités (changements instantanés, donc locaux) dans les signaux. Depuis plus récemment, elle a été utilisée dans l'extraction de caractéristiques à évaluer sur certaines formes, dont celles de caractères imprimés ou manuscrits.

Lors d'une recherche d'articles sur ces familles de caractéristiques issues d'une TO et ayant été évaluées sur des classes de caractères, nous avons énuméré 6 familles. Celles-ci sont l'objet des articles [3-8]. Dans chaque article, un (ou plusieurs) ensemble de caractéristiques d'une famille issue d'une TO a été évalué sur des exemples de caractères. Dans certains de ces articles [3,6], des ensembles de familles issues de transformées globales ont aussi été évalués sur les mêmes exemples : la performance a été plus grande avec les caractéristiques issues d'une TO que celles issues d'une transformée globale.

Ces familles issues d'une TO étant donc prometteuses et récentes, cela nous a motivé à étudier les articles pour y trouver des points à améliorer sur l'extraction des caractéristiques et sur le choix des valeurs de leurs paramètres. Ainsi, après avoir étudié les articles [3-8], nous avons estimé que les points suivants méritent une attention particulière :

- Dans les articles [3,4], le choix des valeurs de certains paramètres n'est pas expliqué;
- Dans l'article [3], l'orthonormalité d'une possible RC de l'image est empêchée par une seule des opérations d'extraction ;
- Dans aucun de ces articles, la TPO n'est utilisée dans l'extraction ;
- Dans l'article [4], l'extraction est effectuée avec des erreurs possibles d'approximation.

Ces quatre points constituent la problématique de ce mémoire. On y observe que le seul article cité à chacun de ces quatre points est l'article [3], qui porte sur *les invariants de*

moments d'ondelette (IMO). L'objectif de ce mémoire est donc d'évaluer des ensembles de IMO modifiés et de paramètres réglés de façon à contribuer à la résolution de ces quatre points. Cette famille résulte de deux transformées consécutives, dont une TO. Soit une grille polaire dans le plan de l'image du caractère. La première transformée est effectuée sur l'image dans la direction angulaire et la deuxième, sur le résultat de la première dans la direction radiale (figure 2).

Ce mémoire est développé comme suit :

- Le chapitre 1 explique le fonctionnement général d'un classifieur, puis présente et explique quelques types courants de classifieur, quelques méthodes et critères de sélection d'ensembles de caractéristiques puis explique quelques transformées globales utilisées comme caractéristiques et montre comment celles-ci sont globales.
- Le chapitre 2 explique une transformée globale classique, une transformée locale classique, puis la TO, puis explique comment la TO permet des représentations complètes ou de bon compromis information-taille, et enfin présente quelques ondelettes courantes puis explique la TPO.

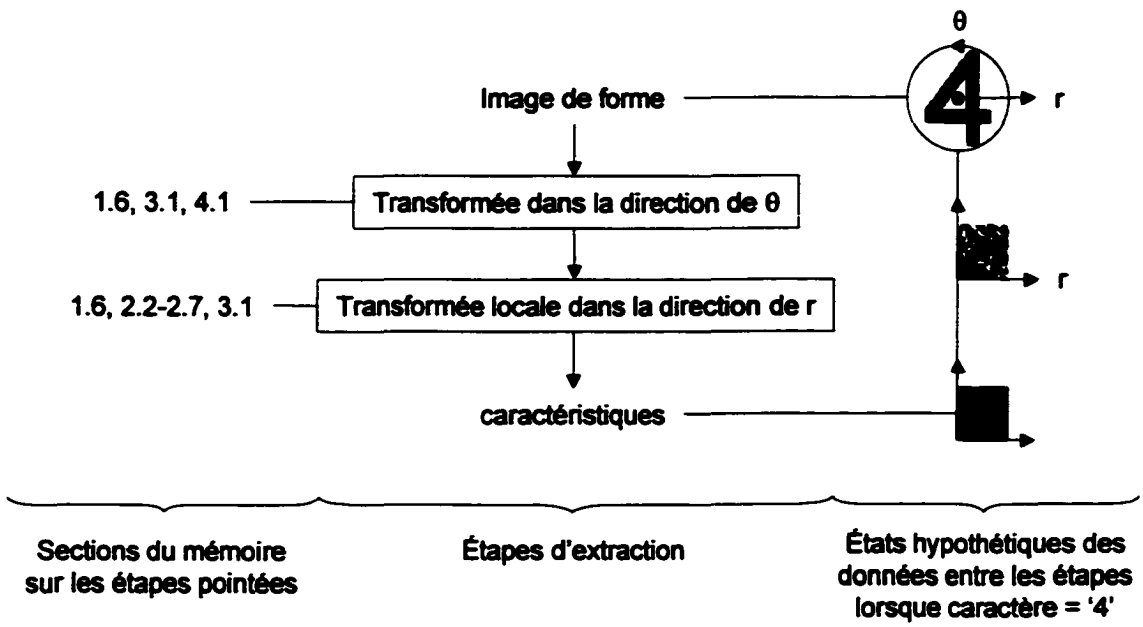


Figure 2 Extraction des moments d'ondelettes

- Le chapitre 3 explique les familles de caractéristiques issues d'une TO, spécifie pour quelles valeurs de paramètres les ensembles de caractéristiques ont été évalués, précise chacun des points de la problématique exposée ci-dessus, puis propose pour chacun une solution puis enfin, montre en quoi chaque solution contribue à la résolution de la problématique.
- Le chapitre 4 développe la méthodologie (algorithme de mise en oeuvre de l'objectif) puis en présente une application avec les résultats.

CHAPITRE 1

RECONNAISSANCE DE FORMES : GÉNÉRALITÉS

1.1 Fonction discriminante

Soit X, l'ensemble des caractéristiques et soit $\mathbf{x} \equiv [x]_{x \in \mathrm{X}}$. La ressemblance de la forme avec une classe donnée, k, est mesurée par $g_k(\mathbf{x})$, g_k étant une fonction *discriminante* [9]. Soit $\mathrm{K} = \{1 \ldots K\}$, l'ensemble des indices des classes. Les paramètres de chaque fonction $g_k|_{k=1\ldots K}$ sont estimés à partir des valeurs de caractéristiques d'exemples de classe k, alors appelés exemples d'*entraînement* [9]. Désignons par j_{ki}, l'indice du $i$ème exemple de classe k, et

$$\mathrm{E} \equiv \{j_{ki}\}_{k=1\ldots K, i=1\ldots I}, \tag{1.1}$$

l'ensemble des indices des exemples d'entraînement.

Un premier exemple de fonction g courante est p [9], $p_k(\mathbf{x})$ (noté aussi $p(k|\mathbf{x})$) étant la probabilité que la forme appartienne à la classe k connaissant $\mathbf{x}$. Un deuxième exemple est $g_{n\mathrm{PPV}}$ [9], $g_{n\mathrm{PPV},k}(\mathbf{x})$ étant le nombre d'exemples dans E de classe k parmi les nPPV ("n plus proches voisins") de la forme dans l'espace $\mathbf{x}$ i.e.

$$g_{n\mathrm{PPV},k}(\mathbf{x}) \equiv \left| \left\{ j_{ki} \middle| j_{ki} \in \{j_\alpha\}_{\alpha=1}^{n}; j \in \mathrm{E} \setminus \{j_\alpha\}_{\alpha=1}^{n}; i = 1 \ldots I; d_{j_\alpha} < d_{j_{\alpha+1}} < d_j \right\} \right| \tag{1.2}$$

où

$$d_j \equiv \sum_{x \in \mathrm{X}} |x - x_j|^s, \tag{1.3}$$

d_j étant une mesure de distance entre la forme et l'exemple j dans l'espace X [9], x_j, la valeur de x lorsque la forme est l'exemple j et s, une valeur plus grande que 0.

Évidemment, peu importe le choix de g, plus $g_k(\mathbf{x})$ est élevé, plus la forme ressemble à la classe k et donc plus la classe k a de chance d'être la classe réelle de la forme. La classe k la plus probable de la forme est donc celle qui maximise $g_k(\mathbf{x})$.

1.2 Classifieur

Le *classifieur* [9] est une fonction qui assigne à $\mathbf{x}$ la classe k qui maximise $g_k(\mathbf{x})$. Soit $\hat{k}$, cette classe i.e. $\hat{k} \equiv l$ tel que $g_l(\mathbf{x}) > g_{l'}(\mathbf{x}), \forall l, l' \in \mathrm{K}, l \neq l'$. (1.4)

Lorsque la classe assignée n'est pas la classe réelle de la forme, il y a *erreur de classification*. Soit e, le taux d'erreur de classification i.e.

$e \equiv$ nombre de formes mal classées / nombre de formes.

Soit $p(k)$, la probabilité d'occurance de la classe k, $p(\mathbf{x}|k)$, la densité de probabilité de $\mathbf{x}$ connaissant k et $\bar{e}$, le taux d'erreur espéré i.e.

$$\bar{e} \equiv 1 - \sum_{k \in \mathrm{K}} p(k) \int_{\{\mathbf{x}|\hat{k}=k\}} p(\mathbf{x}|k) d\mathbf{x} . \tag{1.5}$$

Il est démontrable [9] que $\bar{e}$ est minimal lorsque $g_k(\mathbf{x}) = p(k|\mathbf{x})$; il s'agit du classifieur de *Bayes* [9] : $\bar{e}_{\text{bayes}} \equiv \bar{e}|_{g_k(\mathbf{x})=p(k|\mathbf{x})} = \min_g \bar{e}$.

Soit $\bar{e}_{nPPV} \equiv \bar{e}|_{g=g_{nPPV}}$, il est aussi démontrable [9] que pour un nombre infini d'exemples d'entraînement,

$$\bar{e}_{\text{bayes}} \leq \bar{e}_{nppv} \leq \bar{e}_{\text{bayes}} \left(2 - \frac{K-1}{K} \bar{e}_{\text{bayes}} \right). \tag{1.6}$$

Pour s'assurer que les formes à reconnaître puissent tomber assez proches d'exemples de leur classe réelle, il faut avoir assez d'exemples pour occuper suffisamment l'espace.

1.3 Taille de l'ensemble de caractéristiques

On démontre [9] que pour un nombre fini et fixe d'exemples d'entraînement, le taux d'erreur de classification diminue avec la taille du vecteur de caractéristiques pour atteindre un minimum puis remonter ; il s'agit de la *malédiction de la dimensionnalité*. Par ailleurs, l'extraction des caractéristiques et le calcul des distances demande d'autant plus d'opérations qu'il y a de caractéristiques ; par exemple, dans le calcul de d_j (1.3),

on voit que pour chaque caractéristique, la séquence d'opérations "$+,-,(\cdot)^2$" est requise. Pour ces deux raisons, il est préférable de limiter le vecteur de caractéristiques à celles contenant le plus d'information discriminante.

1.4 Mesure de la quantité d'information discriminante

Pour mesurer la quantité d'information discriminante contenue dans un ensemble de variables, on utilise des fonctions appelées *fonctions objectives*, notées J [9]. La fonction objective la plus effective est $\bar{e}$ (1.5). Pour estimer le taux d'erreur du système de reconnaissance, celui-ci est utilisé sur un ensemble de formes de chaque classe, alors appelées exemples de *test* : le système assigne à chaque exemple de test la classe qui maximise $g_k(\mathbf{x})$. Le taux d'erreur estimé, $\hat{e}$, est alors le nombre d'exemples mal classés sur le nombre total d'exemples de test [9]. Le but du système de reconnaissance étant de reconnaître des formes imprévues, les exemples de test doivent être différents des exemples d'entraînement.

Ceci est le cas dans la méthode d'estimation d'erreur par *validation croisée* [9]. Dans cette méthode, l'ensemble E (1.1) est divisé en sous-ensembles de taille égale, puis chaque sous-ensemble sert à tour de rôle d'ensemble de test pendant que les autres servent d'ensemble d'entraînement. Cet ensemble d'entraînement n'étant qu'une partie de l'ensemble E, l'ensemble d'entraînement effectif, cela ajoute un biais à l'estimation. Pour une taille fixe de E, plus l'ensemble d'entraînement est large, plus il est semblable à E, et donc plus l'estimation est fiable.

Pour le cas où la taille de l'ensemble d'entraînement est maximale, et donc celle de test, minimale, soit 1, l'estimateur s'appelle *taux d'erreur leave-one-out* [9]. Soit $g_k(\mathbf{x},\mathrm{E}) \equiv g_k(\mathbf{x})$, soit $\mathbf{x}_j$ la valeur de $\mathbf{x}$ lorsque la forme est l'exemple j et soit $\hat{k}(j)$, la classe assignée à l'exemple de test j lors d'une estimation d'erreur *leave-one-out* i.e.

$$\hat{k}(j) \equiv k \text{ tel que } g_k(\mathbf{x}_j, \mathrm{E}\setminus j) > g_{k'}(\mathbf{x}_j, \mathrm{E}\setminus j) k \neq k', \forall k,k' \in \mathrm{K}, \qquad (1.7)$$

alors le taux d'erreur leave-one-out, $\hat{e}$, est défini par

$$\hat{e} \equiv \frac{1}{|\mathbb{E}|} \sum_{J_{ik} \in \mathbb{E}} \left| \{ j_{ik} | \hat{k}(j_{ik}) \neq k \} \right| \overset{(1.1)}{=} \frac{1}{KI} \sum_{k=1}^{K} \sum_{i=1}^{I} \left| \{ j_{ik} | \hat{k}(j_{ik}) \neq k \} \right| . \tag{1.8}$$

Notons qu'à chaque échange de rôles, l'ensemble d'entraînement n'étant plus le même, les paramètres des fonctions g_k doivent être réestimés pour toutes les classes k des exemples de test, ce qui demande beaucoup d'opérations. À cause de cela, on utilise parfois d'autres fonctions J, estimées aussi à partir des exemples d'entraînement.

Un exemple d'autres fonctions J est la *divergence de Kullback-Leibler* (DKL). Ainsi, soit (1.1) et soit x_{ik}, la valeur de x du $i$ème exemple de classe k, alors il existe une version de la DKL [10] de x, estimée sur l'ensemble d'exemples E et définie par

$$D(x)\big|_{\forall x \in X} \equiv \sum_{k=1}^{K-1} \sum_{l=k+1}^{K} (E_k(x) - E_l(x)) \log\left(\frac{E_k(x)}{E_l(x)} \right) \text{ où } E_k(x) \equiv \frac{\sum_{i=1}^{I} x_{ik}^2}{\sum_{x \in X} \sum_{i=1}^{I} x_{ik}^2} . \tag{1.9}$$

1.5 Sélection des caractéristiques

La *sélection des caractéristiques* [9] a pour but de trouver, dans un ensemble de variables, un sous-ensemble de caractéristiques qui possède un compromis taille-information discriminante meilleure que certains autres sous-ensembles. Dans un ensemble de taille n, il y a $n!/p!(n-p)!$ sous-ensembles possibles de taille p. Calculer la fonction objective pour chaque sous-ensemble possible peut être très long, même pour des valeurs modérées de n.

Pour cette raison, les méthodes courantes de sélection ne calculent la fonction que pour une fraction des sous-ensembles possibles. Soit Y, l'ensemble de variables et Y_i, le sous-ensemble de i caractéristiques. Une première méthode de sélection est celle que nous appelerons *sélection individuelle*, dont le principe est le suivant : parmi les

variables x tel que $x \in Y$, on déplace dans Y_i celle qui maximise $J(x)$ i.e.

$$y_{i+1} = \{x | x, x' \in Y \setminus Y_i, J(x) \geq J(x')\}. \quad (1.10)$$

Une deuxième méthode de sélection est celle de la *sélection directe séquentielle* (SDS) [9] : parmi les variables x tel que $x \in Y$, on déplace dans Y_i celle qui maximise $J(Y_{i+1})$ i.e.

$$y_{i+1} = \{x | x, x' \in Y \setminus Y_i, J(x \cup Y_i) \geq J(x' \cup Y_i)\}. \quad (1.11)$$

Soit $x, x' \in Y$ et $J(x) = J(x')$. Lors d'une sélection individuelle, x, x' seraient sélectionnées immédiatement l'une après l'autre i.e. si $y_i = x$ alors $y_{i+1} = x'$. Pourtant, elles peuvent être identiques i.e. $x = x'$, donc contenir la même information. Si c'est le cas, une des deux n'ajouterait aucune information à l'autre. Lors d'une sélection par SDS, dès qu'une des deux serait sélectionnée, soit x' i.e. $x' \in Y$, alors comme $J(Y \cup x) = J(Y)$, x ne serait sélectionnée qu'après toutes les autres variables dans Y, lesquelles pourraient ajouter de l'information supplémentaire dans Y. Par conséquent, la SDS est à priori plus fiable que la sélection individuelle.

1.6 Transformée

Tout signal, tel que l'image ou le contour d'une forme, peut-être représenté dans un espace orthogonal par un point dont chaque coordonnée est l'amplitude à un instant/position du signal (luminance d'un point de l'image ou position d'un point du contour). Une *transformée* sur le signal consiste à déplacer les coordonnées entre les points de l'espace (déformer ou réorienter les axes) et a pour but de concentrer toute l'information discriminante du signal dans le moins d'axes possible. Ainsi, les autres axes pourront être éliminés en perdant moins d'information discriminante. Un type de transformée appelé *projection* consiste à simplement réorienter sans les déformer les axes de coordonnées dans l'espace.

Soit f et g, deux fonctions et A, un ensemble de L-uplets de nombres, et soit $x \equiv [x(1),\ldots,x(L)] \in \mathrm{A}$. La projection de f sur g dans A est définie par

$$\langle f|g\rangle_{\mathrm{A}} \equiv \begin{cases} \sum_{k\in\mathrm{A}} f(k)g(k) & \text{A discret} \\ \int_{\mathrm{A}} f(x)g(x)dx & \text{A continu} \end{cases} \tag{1.12}$$

Par simplicité, $\langle f|g\rangle \equiv \langle f|g\rangle_{\mathbb{R}}$.

Soit $[f(\mathrm{A})] \equiv [f(\min \mathrm{A})\ldots f(x)\ldots f(\max \mathrm{A})]$. La *norme* de la fonction f dans A, ou longueur du vecteur $[f(\mathrm{A})]$, est définie par $\|f\|_{\mathrm{A}} \equiv \sqrt{\langle f|f\rangle_{\mathrm{A}}}$ (figure 3a). L'*énergie* de la fonction f dans A, est définie par $\|f\|_{\mathrm{A}}^2$. La longueur de la composante de $[f(\mathrm{A})]$ parallèle à $[g(\mathrm{A})]$ est égale à $\langle f|g\rangle_{\mathrm{A}}/\|g\|_{\mathrm{A}}$ (figure 3b). Lorsque f et g sont parallèles i.e. $g/\|g\| = f/\|f\|$, cette longueur est maximale i.e. $\dfrac{\langle f|g\rangle}{\|g\|} \le \dfrac{\langle f|g\rangle}{\|g\|}\Big|_{g/\|g\|=f/\|f\|} = \langle f|f\rangle/\|f\| = \|f\|^2/\|f\| = \|f\|$.

Pour être certain que (3.12) converge ($\langle f|g\rangle_{\mathrm{A}} < \infty$), il faut que $\{f,g\} \subset L^2(\mathrm{A})$, où $L^2(\mathrm{A})$ est défini comme l'*ensemble des fonctions de carré sommable sur* A i.e.

$$L^2(\mathrm{A}) \equiv \left\{ f \,\middle|\, \|f\|_{\mathrm{A}} < \infty \right\}.$$

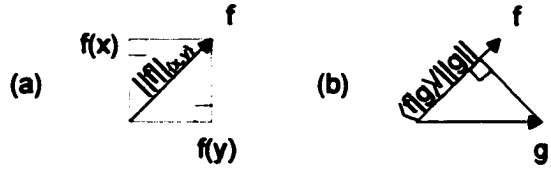


Figure 3 Exemples de norme et de projection

(a) Exemple de norme : $\|f\|_{\{x,y\}}$;

(b) Exemple de projection : $\langle f|g\rangle$.

Soit B, un autre ensemble de L-uplets, et

$$c(n)\big|_{\forall n\in B} \equiv \langle f | g_n \rangle_A . \tag{1.13}$$

Il est intuitif que si tous les vecteurs $[g_n(A)]_{n\in B}$ sont linéairement indépendants et en nombre égal aux axes de l'espace A i.e. $|B|=|A|$, pour tout vecteur $[f(A)]$ on obtient un et un seul vecteur $[c(B)]$. On dit alors que $[c(B)]$ est une *représentation complète* (RC) de $[f(A)]$ ou que $\{g_n\}_{n\in B}$ est une *base complète* de $L^2(A)$. Si au moins un vecteur autre que $[g_n(A)]_{n\in B}$ est ajouté à $[g_n(A)]_{n\in B}$, par exemple $\{g_n\}_{n\in B'\setminus B}$, on dit que $\{g_n\}_{n\in B'}$ est une *base complète redondante* de $L^2(A)$.

Le vecteur $[c(B)]$ contient donc toute l'information sur $[f(A)]$. Par conséquent, le vecteur $[f(A)]$ peut être reconstruit à partir de $[c(B)]$. Ainsi, il est possible [11] de trouver un ensemble de fonctions $\{\tilde{g}_n\}_{n\in B}$ tel que $f(k)\big|_{\forall k\in A} = \sum_{n\in B} c(n)\tilde{g}_n(k)$, (1.14)

appelé *équation de reconstruction* de $[f(A)]$ à partir de $[c(B)]$.

Soit $B'\subset B$, alors $[c(B')]$ ne contient qu'une partie de l'information sur $[f(A)]$. Une mesure de l'information non contenue dans $[c(B')]$ est l'*erreur quadratique* de représentation de $[f(A)]$ par $[c(B')]$. Soit $[\hat{f}(A)]$, le vecteur reconstruit à partir de $[c(B')]$ i.e.

$$\hat{f}(k)\big|_{\forall k\in A} = \sum_{n\in B'} c(n)\tilde{g}_n(k),$$

alors l'erreur quadratique est définie par $\|\hat{f}-f\|_A$. Il est démontré [11] que cette erreur est minimale lorsque tous les vecteurs $[g_n(A)]_{n\in B}$ sont *orthonormaux* entre eux, c'est-à-dire orthogonaux et de longueur unitaire i.e.

$$\langle g_i | g_j^* \rangle_A \big|_{\forall i,j\in B} = \delta(i-j), \tag{1.15}$$

où δ est la fonction *d'impulsion* i.e.

$$\delta(x) \equiv \begin{cases} 1 & x(1) = \ldots = x(L) = 0 \quad \text{A discret} \\ \infty & x(1) = \ldots = x(L) = 0 \quad \text{A continu} \\ 0 & \text{sinon} \end{cases}. \tag{1.16}$$

Il est aussi démontré [11] que (3.15) implique que

$$\tilde{g}_n = g_n^* \tag{1.17}$$

et que

$$\|c\|_B = \|f\|_A . \tag{1.18}$$

On dit alors que $\{g_n\}_{n\in B}$ est une *base orthonormale complète* (BOC) de $L^2(A)$ ou que $[c(B)]$ est une *représentation orthonormale complète* (ROC) de $[f(A)]$.

Le résultat (3.18), appelé *équation de conservation de l'énergie* de f dans c, est intuitif. En effet, d'après (3.13), la prolongation de tout vecteur $[g_n(A)]_{n\in B}$ est l'axe $c(n)$ de l'espace $[c(B)]$. Par conséquent, si tous les vecteurs $[g_n(A)]_{n\in B}$ sont orthogonaux entre eux, tous les axes $c(n)$ le sont aussi. Aussi, en fixant l'échelle de tout axe de l'espace $[f(A)]$ à la valeur 1, l'échelle de l'axe $c(n)$ devient égale à $\|g_n(A)\|$. Donc si en plus tous les vecteurs $[g_n(A)]_{n\in B}$ sont de longueur unitaire, alors l'échelle de l'axe $c(n)$ est unitaire aussi et par conséquent tout vecteur apparaît de même longueur dans $[f(A)]$ ou dans $[c(B)]$ i.e. $\|c\|_B = \|f\|_A$, mais réorienté selon un angle de valeur et direction fixe.

1.7 Mesure d'entropie d'une représentation orthonormale

Dans la section précédente, l'information totale contenue dans le vecteur $[c(B)]$ est la même que celle contenue dans le vecteur $[f(A)]$.Par contre, elle n'est pas distribuée de la même façon entre les composantes, car $[c(B)] \neq [f(A)]$. Soit $A' \subseteq A$ et $B' \subseteq B$. Pour mesurer l'uniformité de cette distribution entre les variables dans $c(B')$, on utilise la DKL de $c(B')$, définie par

$$D(c(B')) \equiv \sum_{n\in B'} D(c(n)) . \tag{1.19}$$

Pour pouvoir comparer l'uniformité dans $c(B')$ à celle dans $f(A')$ par leur DKL, la condition [10] est que $c(B')=f(A')$ et $c(B)=f(A)$.

1.8 Normalisation

Soit l'image d'un caractère. Soit $f(x,y)$, le niveau de gris du point de coordonnées cartésiennes (x,y) et $(x(t),y(t))$ les coordonnées cartésiennes du point de position t le long du contour externe du caractère, et $z(t)\equiv x(t)+jy(t)$. Pour s'assurer que les caractéristiques vont être invariantes à la position et à la taille, l'origine et l'échelle de f sont *normalisées*.

Soit $\tilde{f}$, l'image contenant la même forme que f, mais rétrécie α fois dans toutes les directions, puis déplacée de (a,b) dans le sens négatif des axes de f i.e.

$$\tilde{f}(x,y)\equiv f(\alpha x+a,\alpha y+b). \tag{1.20}$$

Ces deux formes sont nécessairement de même classe mais pas nécessairement de mêmes valeurs de caractéristiques. Pour rendre celles-ci égales, on déplace et dilate l'échelle de f pour que les images deviennent identiques i.e. pour que f devienne $\tilde{f}$, ce qui assure que les caractéristiques vont être invariantes à la position et à la taille.

Il existe plusieurs mesures de la taille et de la position horizontale et verticale d'une forme. Désignons ces mesures par w,u,v pour la forme f et par $\tilde{w},\tilde{u},\tilde{v}$, pour la forme $\tilde{f}$. Elles respectent les identités suivantes :

$$w=\alpha\tilde{w},[u\quad v]=[\tilde{u}\quad \tilde{v}]+[a\quad b]. \tag{1.21}$$

α,a,b sont calculés à partir de ces trois équations. On dira que l'échelle et l'origine sont *normalisées* par rapport à w et (u,v) respectivement.

Lorsque les caractéristiques sont celles du contour, on peut soit normaliser l'origine et l'échelle de f comme précédemment, ou plutôt normaliser l'origine et l'échelle de z. Pour cela, il suffit de remplacer $f(x,y)$ par $z(t)$ dans (3.20).

1.9 Transformées globales

Dans cette section, on explique les transformées globales suivantes, utilisées en reconnaissance de caractères :

- moments et leurs invariants ;
- descripteurs de Fourier et leurs modules ;
- projections de *Karhunen-Loève.*

Soit $re^{j\theta} \equiv x + jy$ et $$f_p(r,\theta) \equiv f(x,y), \tag{1.22}$$

et soit g_{mn}, une fonction à choisir, et $g_{mnq}(x,y) \equiv g_{mn}(r)e^{jq\theta}$. Plusieurs familles de moments ont la forme suivante [3] : $$g_{mnq}(x,y) \equiv \iint_{0<\theta<2\pi} f_p(r,\theta) g_{mn}(r) e^{jq\theta}\, dr d\theta. \tag{1.23}$$

Les *invariants* de ces moments, définis par $|c(m,n,q)|$, sont appelés ainsi car ils sont invariants à la rotation et à la symétrie [1] i.e.

$$\iint_{0<\theta<2\pi} f_p(r,\theta+\Delta) g_{mn}(r) e^{jq\theta}\, dr d\theta = \iint_{0<\theta<2\pi} f_p(r,-\theta) g_{mn}(r) e^{jq\theta}\, dr d\theta = \iint_{0<\theta<2\pi} f_p(r,\theta) g_{mn}(r) e^{jq\theta}\, dr d\theta .$$

Parmi les fonctions g_{mn} utilisées, il y a les *polynômes de Zernike* [1], définis par

$$g_{mn}^{\text{zernike}}(r) \equiv \frac{n+1}{\pi} \sum_{k=0}^{(m-|n|)/2} (-1)^k \frac{(m-k)!}{k!\left(\frac{m+|n|}{2}-k\right)!\left(\frac{m-|n|}{2}-k\right)!} r^{m/2-k}. \tag{1.24}$$

$\left\{g_{mnq}^{\text{zernike}}\right\}_{m,\frac{m-|n|}{2}\in\mathbb{N},q=n}$ forme une BOC de $L^2(\{(x,y)\}_{0<r<1})$.

Certains *descripteurs de Fourier* (DF) [6] sont définis par

$$c(q) \equiv \langle z \,|\, g_q \rangle_{[0,2\pi]} \text{ avec } g_q(t) \equiv e^{jqt}. \tag{1.25}$$

Par similarité avec " $\int f_p(r,\theta)e^{jq\theta}d\theta$ " dans (3.23), le module des descripteurs de Fourier est invariant au déplacement le long du contour i.e.

$$\left|\int_a^b z(t+\Delta)e^{jqt}\,dt\right| = \left|\int_a^b z(t)e^{jqt}\,dt\right|.$$

$\{g_q\}_{q\in\mathbb{Z}}$ forme une BOC de $L^2([0,2\pi])$.

La projection de *Karhunen-Loève* a pour but de concentrer le mieux possible l'information totale de tout signal dans un seul et même axe. Pour cela, elle effectue sur l'espace du signal, parmi toute les rotations globales possibles, celle qui aligne le mieux possible des exemples de vecteur (signaux) dans la direction d'un seul axe. On peut mesurer l'alignement dans une direction par la variance entre les composantes des exemples dans cette direction : la projection de *Karhunen-Loève* est alors la coordonnée par rapport à l'axe de variance maximale.

Dans (3.23) et (3.24), on voit que chaque coefficient est une somme de tous les points de l'image $f_p(r,\theta)$ pondérés par $rg_{mn}(r)e^{jq\theta}\,dr d\theta$ ou de tous les points du contour $z(t)$ pondérés par $g_q(t)dt$. Ces poids sont non nuls sur tous les points ou presque, car $\left|rg_{mn}^{\text{zernike}}(r)\right|_{0<r<1} \neq 0$ et $|g_q(\cdot)| = 1 \neq 0$. De même, pour la projection de *Karhunen-Loève*, chaque coordonnée après rotation peut dépendre de toutes les coordonnées avant rotation. Par conséquent, l'information contenue dans chaque coefficient est *globale*, c'est-à-dire diluée sur toute l'image ou le contour.

Dans une même image, l'information discriminante peut être globale ou concentrée en régions plus ou moins locales. Il peut donc être préférable d'utiliser une fonction g ayant des valeurs $g(r)$ non négligeables seulement dans un intervalle de r de position et de largeur variables.

CHAPITRE 2

TRANSFORMÉE EN ONDELETTES

2.1 Transformées classiques

Parmi les transformées existantes, celle qui a longtemps été utilisée pour représenter certaines fonctions f, comme les signaux sonores, est la *transformée de Fourier* (TF) :

$$F(\omega) \equiv \langle f | g_\omega \rangle \text{ où } g_\omega(r) \equiv e^{-j\omega r}, \tag{2.1}$$

où $\{g_\omega\}_{\omega \in \mathbb{R}}$ forme une BOC de $L^2(\mathbb{R})$. Dans le reste du chapitre 2, les coefficients de la transformée de Fourier de toute fonction seront désignés par la même lettre que la fonction mais en majuscule. Comme $|g_\omega(\cdot)| = 1 \neq 0$, les fonctions g_ω sont globales. Par conséquent, les coefficients F ne peuvent être correctement estimés que lorsque le signal f est mesuré durant une période suffisamment longue.

Les signaux locaux à l'intérieur d'un signal global plus long sont parfois de courte durée, comme les syllabes dans une conversation. Supposons que l'on veuille estimer correctement les valeurs de leurs coefficients F. En exécutant la TF du signal global (la conversation) sur toute sa durée, chaque coefficient F sera biaisé par le reste du signal global (autres syllabes ou silences). Pour minimiser ce biais, le signal global doit être pondéré par une fonction w locale appelée *fenêtre*, à position réglable b, que l'on fait glisser tout le long du signal global. Il s'agit de la *TF à fenêtre glissante* i.e.

$$F(b,\omega) \equiv \langle f | g_{b\omega} \rangle = \frac{1}{2\pi} \langle F | G_{b\omega} \rangle, \tag{2.2}$$

$g_{b\omega}$ étant la fonction w déplacée de b vers la droite et modulée en fréquence par ω i.e.

$$g_{b\omega}(r) \equiv w(r-b)e^{-j\omega r}.$$

L'ensemble $\{g_{b\omega}\}_{b,\omega \in \mathbb{R}}$ forme une base complète de $L^2(\mathbb{R})$. L'identité de droite est un cas particulier du théorème de *Parseval* [11].

$F(b,\omega)$ peut être vue comme une estimation de $F(\omega)$ ou de $f(b)$, car d'après (2.2), $F(b,\omega)\big|_{\xi=\delta} = f(b)$ et $F(b,\omega)\big|_{\xi=1} = F(\omega)$. Soit $\hat{b},\hat{\omega}$, les valeurs estimées de b,ω telles que $F(b,\omega) = f(\hat{b}) = F(\hat{\omega})$, et le biais $|\hat{b}-b|$ et $|\hat{\omega}-\omega|$. L'espérance du biais, ou incertitude est mesurée par l'écart-type temporel $\sigma_r(b,\xi)$ et spectral $\sigma_\omega(b,\xi)$ de la fonction $g(b,\xi)$ utilisée. Notons que l'écart-type temporel peut être vu aussi comme une mesure de la durée de la fonction. Calculons les expressions de σ_r et σ_ω en fonction de b et ξ. Tout d'abord, $\{g_{b\omega}\}_{b,\omega\in\mathbb{R}}$ étant une BOC de $L^2(\mathbb{R})$, la conservation de l'énergie (1.17) implique

que $$\|G_{b\omega}\|^2 = \|g_{b\omega}\|^2 = \int |g_{b\omega}(r)|^2 dr = \int |w(r-b)|^2 dr = \|w\|^2 = 1. \qquad (2.3)$$

Aussi, $$|G_{b\xi}(\omega)| = \left|\int w(r-b)e^{-j(\omega+\xi)r} dr\right| = \left|e^{-j(\omega+\xi)b}\int w(r)e^{-j(\omega+\xi)r} dr\right| = |G_{00}(\omega+\xi)|. \qquad (2.4)$$

Soit $\bar{r}(b,\xi)$ et $\bar{\omega}(b,\xi)$ la position et la fréquence moyennes de $g(b,\xi)$, alors

$$\bar{\omega}(b,\xi) \equiv \int \omega \frac{|G_{b\xi}(\omega)|^2}{\|G_{b\xi}\|^2} d\omega \overset{(2.3,4)}{=} \int \omega |G_{00}(\omega+\xi)|^2 d\omega = \int \omega |G_{00}(\omega)|^2 d\omega - \xi \|G_{00}\| \overset{(2.3)}{=} \bar{\omega}(0,0) - \xi, \qquad (2.5)$$

$$\bar{r}(b,\xi) \equiv \int r |g_{b\xi}(r)|^2 dr \overset{(2.2)}{=} \int r |w(r-b)|^2 dr = \int (r+b)|w(r)|^2 dr = \int r|w(r)|^2 dr + b\|w\| \overset{(2.3)}{=} \bar{r}(0,0) + b \qquad (2.6)$$

et alors

$$\sigma_\omega(b,\xi)^2 \equiv \int \frac{|(\omega-\bar{\omega}(\xi))G_{b\xi}(\omega)|^2}{\|G_{b\xi}\|^2} d\omega \overset{(2.4,5)}{=} \int |(\omega+\xi-\bar{\omega}(0))G_{00}(\omega+\xi)|^2 d\omega = \sigma_\omega(0,0)^2,$$

$$\sigma_r(b,\xi)^2 \equiv \int |(r-\bar{r}(b,\xi))g_{b\xi}(r)|^2 dr \overset{(2.2,6)}{=} \int |(r-\bar{r}(0,0)-b)w(r-b)|^2 dr = \int |(r-\bar{r}(0,0))w(r)|^2 dr = \sigma_r(0,0)^2.$$

Ces deux dernières identités signifient que toutes les fonctions d'une base telle que $\{g_{b\omega}\}_{b,\omega\in\mathbb{R}}$ donnent la même incertitude sur la position et la fréquence.

D'après (2.5) et (2.6), ξ et b nous permettent de contrôler $\bar{\omega}$ et $\bar{r}$ en fonction de l'allure des signaux f à représenter. Pour les représenter le mieux possible, donc

maximiser $F(b,\xi)$, ξ et b doivent être réglés pour que $g_{b\xi}$ soit le plus près possible de $f/\|f\|$ (§1.6). Cette ressemblance est améliorée si les écart-types (temporel et spectral) de la fonction g sont proches de ceux de f.

2.2 Transformée en ondelettes

Les différents signaux locaux à représenter et situés à différentes positions d'un même signal global plus long peuvent être de durées différentes. Il est alors souhaitable que chaque fonction d'une même base complète aient non seulement une position et fréquence moyenne différente des autres, mais aussi une durée ajustable indépendamment des autres. Ainsi, on peut ajuster cette durée pour rendre la fonction davantage semblable à celle de n'importe quel des signaux locaux de positions différentes. C'est le cas des *ondelettes* ψ_{ab}. Il s'agit alors de la *transformée en ondelette* (TO) i.e.

$$d(a,b) \equiv \langle f | \psi_{ab} \rangle \text{ où } \psi_{ab}(r) \equiv a^{-1/2}\psi((r-b)/a) \qquad (2.7)$$

Le paramètre d'*échelle* a détermine la largeur et la hauteur de l'ondelette : la largeur varie en sens inverse de la hauteur pour que $\|\psi_{ab}\|$ reste normalisé à la valeur 1 (figure 4a) : $\|\psi_{ab}\| = \int \psi_{ab}(r)^2 dr = \int a^{-1}\psi(r/a)^2 dr = \int a^{-1}\psi(r/a)^2 a d(r/a) = \int \psi(r)^2 dr = \|\psi\| = 1$.

Ainsi, sur la figure 4a on observe que la courbe de ψ_{ab} est 2 fois plus large et $\sqrt{2}$ fois moins haute quand a est 2 fois plus grand.

Calculons les expressions de σ_r et σ_ω en fonction de a et b. Tout d'abord,

$$|\Psi_{ab}(\omega)| = \left| \int \psi_{ab}(r) e^{-j\omega r} d(r) \right| = \left| \int a^{-1/2}\psi(r/a) e^{-j\omega a(r/a)} a d(r/a) \right| = \left| \int \sqrt{a}\psi(r) e^{-j\omega a r} dr \right| = \left| \sqrt{a}\Psi(a\omega) \right|$$

$$\text{alors} \qquad \bar{\omega}(a) \equiv \int \omega |\Psi_{ab}(\omega)|^2 d\omega = \int \frac{a\omega}{a} a |\Psi(a\omega)|^2 \frac{d(a\omega)}{a} = \frac{1}{a}\int \omega |\Psi(\omega)|^2 d\omega = \frac{\bar{\omega}(1)}{a} \qquad (2.8)$$

$$\bar{r}(a,b) \equiv \int r |\psi_{ab}(r)|^2 dr \overset{(2.7)}{=} \int r \left| \frac{1}{\sqrt{a}} \psi\left(\frac{r-b}{a} \right) \right|^2 dr = \int (ar+b) |\psi(r)|^2 dr = \int ar |\psi(r)|^2 dr + b\|\psi\|^2 = a\bar{r}(1,0) + b$$

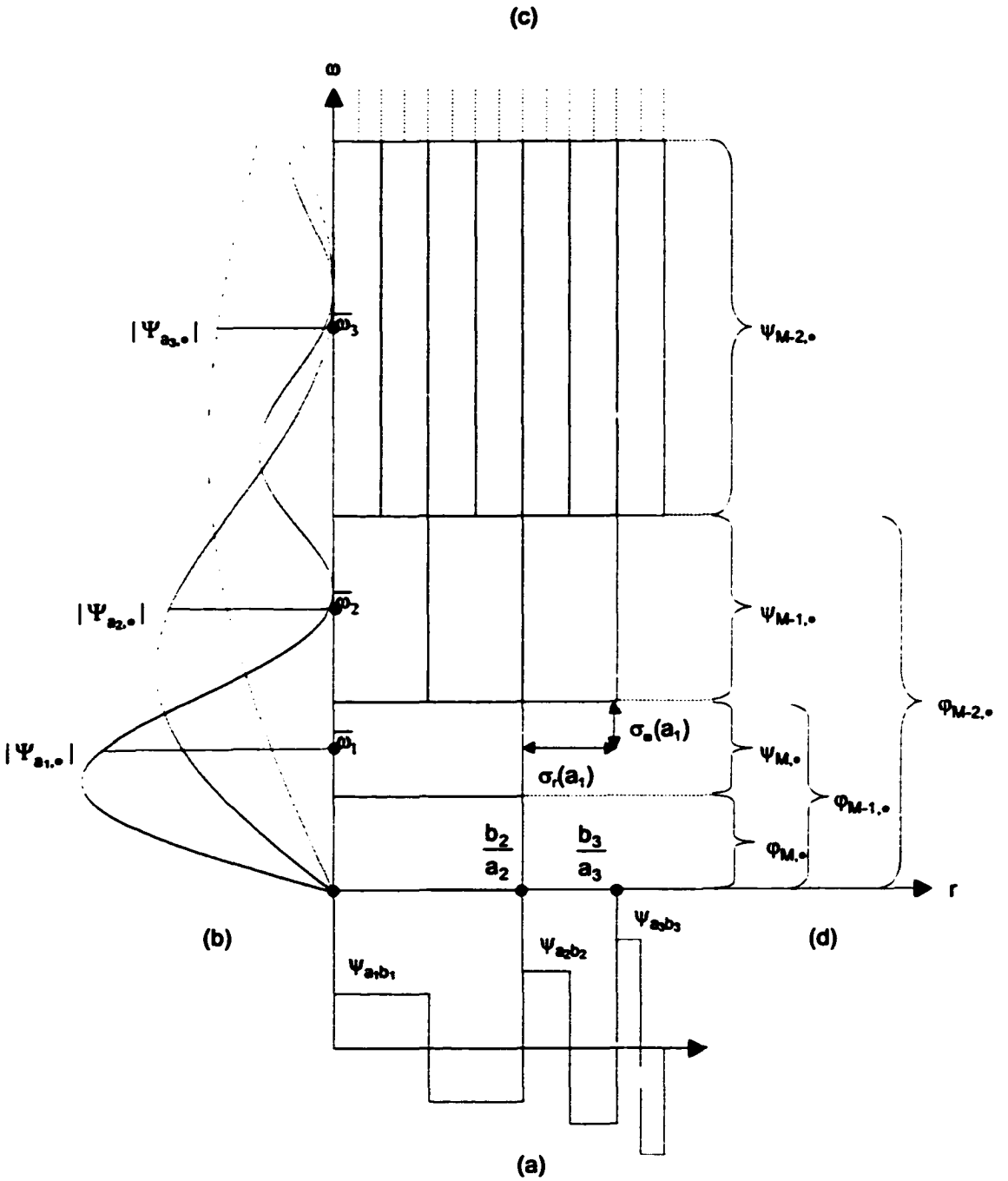


Figure 4 Ondelettes et plan (r, ω)

(a) exemple de fonction ψ ; $a_1 = 2a_2 = 4a_3$;
(b) spectre de chaque fonction de la figure a;
(c) plan (r, ω) de ψ ;
(d) "}" : largeur des boîtes des fonctions.

et alors

$$\sigma_\omega(a)^2 \equiv \int(\omega-\bar{\omega}(a))|\Psi_{ab}(\omega)|^2 d\omega \overset{(2.8)}{=} \int\frac{|(a\omega-\bar{\omega}(1))\Psi(a\omega)|^2}{a^2}d(a\omega) = \int\frac{|(\omega-\bar{\omega}(1))\Psi(\omega)|^2}{a^2}d\omega = \frac{\sigma_\omega^2(1)}{a^2},$$

$$\sigma_r(a)^2 = \int(r-\bar{r}(a,b))|\psi_{ab}(r)|^2 dr = \int(ar-a\bar{r}(1,0))|\psi(r)|^2 dr = a^2\int(r-\bar{r}(1,0))|\psi(r)|^2 dr = a^2\sigma_r^2(1).$$

D'après les identités (2.8) et suivantes $\bar{\omega}\propto 1/a$, $\sigma_\omega \propto 1/a$ et $\sigma_r \propto a$ ('$\propto$' signifie "proportionnel à"). Ainsi, sur la figure 4a on observe que la courbe $\Psi_{a:}$ est deux fois plus étroite et que son centre est deux fois plus proche de l'origine quand a est deux fois plus grand. Cela signifie que a nous permet de régler $\bar{\omega}, \sigma_\omega$ ou σ_r en fonction de l'allure des composantes f à représenter. En général, les variations rapides durent moins longtemps que les variations lentes. Or, comme $\bar{\omega}\propto 1/a$ et $\sigma_r \propto a$, alors $\sigma_r \propto 1/\bar{\omega}$ ce qui signifie que l'ondelette a aussi ce comportement, ce qui permet de régler a plus facilement pour maximiser la ressemblance entre ψ_{ab} et $f/\|f\|$.

On remarque que comme $\sigma_r \propto 1/\bar{\omega}$ et $\sigma_\omega \propto \bar{\omega}$, $\sigma_r\sigma_\omega$ = constante. Il est démontré [11] que constante $\geq 1/2$ quelque soit ψ : il s'agit du *principe d'incertitude*.

2.3 Représentation complète ou non

L'ensemble $\{\psi_{ab}\}_{b\in\mathbb{R},a\in\mathbb{R}^*_+}$ forme une base redondante complète de $L^2(\mathbb{R})$. Il est démontré [11] que l'équation de reconstruction de f à partir de d, ou TO *inverse* est :

$$f(r) = \frac{1}{C_\psi}\int_0^\infty\int_{-\infty}^{\infty} d(a,b)\psi_{ab}(r)db\frac{da}{a^2} \text{ où } C_\psi \equiv \int_0^\infty\frac{|\Psi(\omega)|^2}{\omega}d\omega.$$

Pour que cette équation converge, il faut que

$$C_\psi < \infty \Rightarrow \frac{|\Psi(0)|^2}{0} < \infty \rightarrow \Psi(0) = 0. \qquad (2.9)$$

D'un autre côté, pour qu'une ondelette ψ_{ab} ne représente que les variations de fréquences non nulles, elle ne doit avoir aucune fréquence nulle, ce qui impose la même condition :

$$0 = d(a,b) = f(r)\int \psi_{ab}(r)dr \rightarrow \int \psi_{ab}(r)dr = \Psi_a(0) = 0 .$$

En pratique, on ne peut pas calculer un ensemble continu de coefficients. Sur le plan (r,ω) (figure 4c), les régions centrées sur différentes valeurs de r et ω sont délimitées par des boîtes de tailles $2\sigma_r(a = k/\omega) \times 2\sigma_\omega(a = k/\omega)$. Ainsi, une boîte centrée sur (b,ω) délimite la région du signal f ayant le plus grand poids sur la valeur du coefficient $d(a = k/\omega, b)$. Un critère pour choisir un ensemble discret ayant un bon compromis information-taille est que ces boîtes couvrent tout le plan (r,ω) sans se recouper (figure 4c). Un choix possible est $a = a_0^m$ $(a_0 > 1, m \in \mathbb{Z})$, avec la constante a_0 convenablement fixée et bien sûr si le pas de b est égal à la largeur $2\sigma_r(a)$, elle-même proportionnelle à a i.e. $b = nb_0a_0^m$ $(b_0 > 0, n \in \mathbb{Z})$. Dans ce cas,

$$d(m,n) \Leftarrow d(a,b) \text{ et } \psi_{mn}(r) \Leftarrow \psi_{ab}(r).$$

Dans le reste du texte, $a_0 = 2$ et $b_0 = 1$.

Certaines fonctions ψ sont conçues pour que l'ensemble $\{\psi_{mn}\}_{m,n\in\mathbb{Z}}$ forme une base complète de $L^2(\mathbb{R})$. Dans ce cas, on fixe la taille des boîtes non plus exactement à $2\sigma_r \times 2\sigma_\omega$ mais selon des valeurs quand même proportionnelles à $1/a$.

À chaque ajout d'une boîte de haut en bas, on voit que seule la moitié supérieure de la région vide est couverte (figure 4d) et il restera donc toujours une moitié inférieure vide. Pour couvrir celle-ci avec une seule boîte sans recouper la région supérieure, on remplace ψ dans (2.7) par une autre fonction, la *fonction d'échelle* φ, Il s'agit de la transformée d'échelle (TE) i.e.

$$c(m,n) \equiv \langle f | \varphi_{mn} \rangle \text{ où } \varphi_{mn}(r) \equiv 2^{-m/2}\varphi\left(2^{-m}r - n\right). \qquad (2.10)$$

La fonction φ est telle que $\{\varphi_{m+1,n}, \psi_{m+1,n}\}_{m,n\in\mathbb{Z}}$ forme une base complète de $L^2(\mathbb{R})$.

2.4 Calcul de la TO et TE

La TO et la TE d'un signal f, comme on le voit (2.7. 2.10), est une intégrale, en fait celle de $f(r)\psi_{mn}(r)$ et $f(r)\varphi_{mn}(r)$ le long de r pour $r \in (\text{support de } \psi_{mn} \text{ et } \varphi_{mn})$. Or, les valeurs de $f(r)$ ne peuvent être enregistrées puis traitées que pour un nombre fini de valeurs de r.

Sur la figure 4d, on voit que les deux moitiés inférieure et supérieure à l'échelle 2^{m+1} forment la moitié inférieure de l'échelle précédente 2^m. Par conséquent, l'ensemble $\{\varphi_{m+1,n}, \psi_{m+1,n}\}_{n\in\mathbb{Z}}$ forme une base complète de $\{\varphi_{mn}\}_{n\in\mathbb{Z}}$ i.e. il existe deux fonctions $\tilde{g}, \tilde{h}$ tel que

$$\left[\varphi_{m+1,n} \quad \psi_{m+1,n}\right] = \sum_k \varphi_{m,2n-k} \left[\tilde{g}(k) \quad \tilde{h}(k)\right] \tag{2.11}$$

et donc que

$$\begin{bmatrix} c(m+1,n) \\ d(m+1,n) \end{bmatrix} \overset{(2.10)}{=} \left\langle f \middle| \begin{bmatrix} \varphi_{m+1,n} \\ \psi_{m+1,n} \end{bmatrix} \right\rangle \overset{(2.11)}{=} \sum_k \begin{bmatrix} \tilde{g}(k) \\ \tilde{h}(k) \end{bmatrix} \left\langle f \middle| \varphi_{m,2n-k} \right\rangle = \sum_k c(m,2n-k) \begin{bmatrix} \tilde{g}(k) \\ \tilde{h}(k) \end{bmatrix}. \tag{2.12}$$

L'équation précédente, comme on le voit, est une somme discrète, en fait celle de $c(m,2n-k)\left[\tilde{g}(k) \quad \tilde{h}(k)\right]^T$ le long de k pour $k \in (\text{support de } \tilde{g} \text{ ou } \tilde{h})$. Par conséquent, si le support de $\tilde{g}$ ou $\tilde{h}$ est compact, le calcul de $[c(m+1,n) \quad d(m+1,n)]$ avec (2.12) est réalisable, contrairement au même calcul mais avec (2.7, 2.10), bien sûr à condition de connaître $[c(m,2n-k)]_{k\in(\text{support de } \tilde{g} \text{ ou } \tilde{h})}$.

Pour connaître $c(m,n)$, il faudrait exécuter (2.10), ce qui est irréalisable. Or, $\lim_{m\to-\infty} 2^{-m}\varphi(2^{-m}r) = \delta(r)$, δ étant la fonction d'impulsion (1.16). Donc si m est suffisamment petit, on peut estimer $c(m,n)$ ainsi :

$$c(m,n) \overset{(2.10)}{=} 2^{m/2} \int f(r) 2^{-m} \varphi\left(2^{-m}\left(r - 2^m n\right)\right) dr \approx 2^{m/2} \int f(r)\delta\left(r - 2^m n\right) dr = 2^{m/2} f\left(2^m n\right). \tag{2.13}$$

Une autre méthode de calcul de $c(m,n)$, expérimentée par Abry et Flandrin [12], est d'approximer l'intégrale (2.10) par une somme discrète i.e.

$$c(m,n) \approx \sum_k f(k)\varphi_{mn}(k).$$

Comme $\{\varphi_{m+1,n}, \psi_{m+1,n}\}_{n\in\mathbb{Z}}$ forme une base complète de $\{\varphi_{mn}\}_{n\in\mathbb{Z}}$, il est clair que $[c(m+1,\mathbb{Z}) \quad d(m+1,\mathbb{Z})]$ forme une RC de $[c(m,\mathbb{Z})]$ et donc il existe deux fonctions g,h tel que $h(-2n-k), g(-2n-k)$ sont à l'équation suivante ce que $\tilde{g}_{2n}(k), \tilde{g}_{2n+1}(k)$ sont à (1.14) :

$$c(m,k)\big|_{\forall k\in\mathbb{Z}} = \sum_n c(m+1,n)g(2n-k) + \sum_n d(m+1,n)h(2n-k). \qquad (2.14)$$

Les équations (2.12) et (2.14) sont en fait celles du *banc de filtres dyadique*, formé de deux parties : le banc d'analyse et le banc de synthèse (figure 5). L'équation (2.12) est celle du banc d'analyse et l'équation (2.14), celle du banc de synthèse. Si on relie plusieurs bancs l'un à l'autre par leur sortie/entrée, il s'agit de l'algorithme de l'*arbre dyadique* (figure 6).

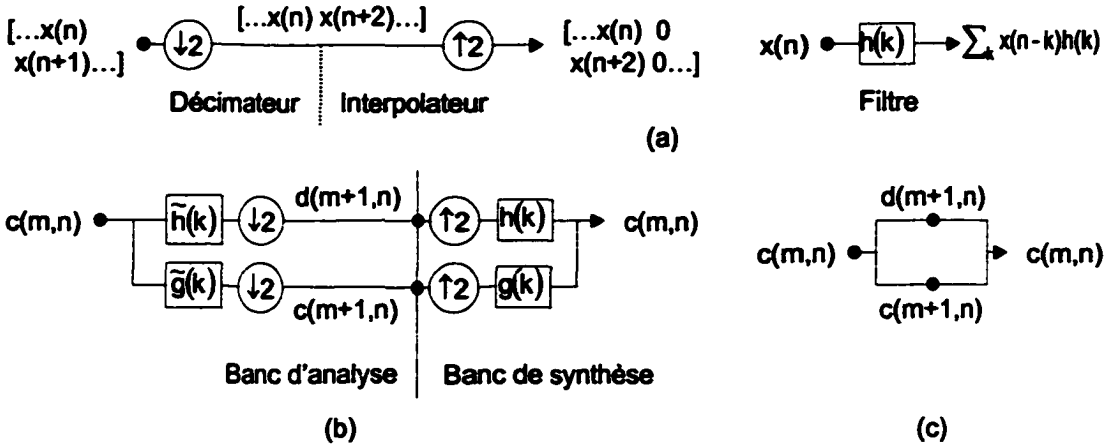


Figure 5 Banc de filtres dyadique et ses composantes

(a) Composantes : décimateur, interpolateur et filtre;
(b) Banc de filtres dyadique;
(c) Représentation simplifiée de la figure de gauche.

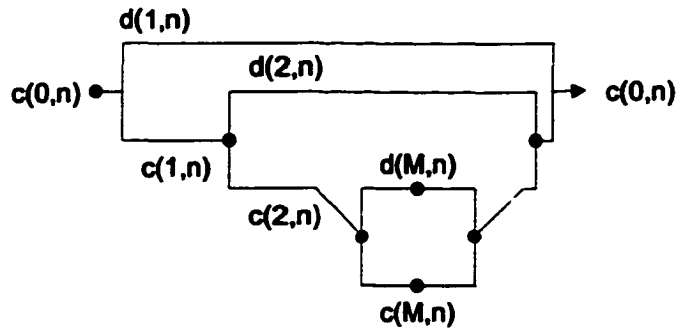


Figure 6 Arbre dyadique à M bancs.

2.5 Propriétés de $\tilde{g}$ et $\tilde{h}$

Déterminons les propriétés de $\tilde{g}$ et $\tilde{h}$. Soit $\tilde{G},\tilde{H}$, la TF *discrète* [11] de $\tilde{g},\tilde{h}$ respectivement i.e. $\left[\tilde{G}(e^{j\omega})\ \ \tilde{H}(e^{j\omega})\right] \equiv \sum_k \left[\tilde{g}(k)\ \ \tilde{h}(k)\right] e^{jk\omega}$, alors d'après (2.1),

$$\Phi(\omega) = \sum_k \tilde{g}(k) \int \sqrt{2}\varphi(2r+k) e^{-j\omega r} dr = \sum_k \tilde{g}(k) \sqrt{2} \int \varphi(u) e^{-j.5\omega(u-k)} d\frac{u}{2},$$

et alors
$$\Phi(\omega) = \sum_k \tilde{g}(k) e^{j.5\omega k} 2^{-1/2} \int \varphi(u) e^{-j.5\omega u} du = 2^{-1/2} \tilde{G}(e^{j\omega/2}) \Phi(\omega/2). \qquad (2.15)$$

De façon similaire
$$\Psi(\omega) = 2^{-1/2} \tilde{H}(e^{j\omega/2}) \Phi(\omega/2). \qquad (2.16)$$

De (2.15), on déduit que
$$\tilde{G}(e^0) = \sum_k \tilde{g}(k) = 2^{1/2} \Phi(0) / \Phi(0) = \sqrt{2} \qquad (2.17)$$

et de (2.9) et (2.16), que
$$\tilde{H}(e^0) = 2^{1/2}\, \Psi(0) / \Phi(0) = 0. \qquad (2.18)$$

Il est aussi démontrable [11] que $\tilde{G}(\infty) = \tilde{H}(\infty) = 0$, ce qui signifie avec (2.17) et (2.18), que $\tilde{g}$ et $\tilde{h}$ sont des filtres *passe-bas* et *passe-bande* respectivement. Ainsi, le banc de filtres $\tilde{g}$ et $\tilde{h}$ décompose le signal d'entrée $[c(m,\mathbb{Z})]$ en signaux de sorties $[c(m+1,\mathbb{Z})]$ et $[d(m+1,\mathbb{Z})]$ de bandes moins larges et de fréquences moyenne inférieure et supérieure respectivement. On appelle $\tilde{g}$ et $\tilde{h}$ les *filtres de* ψ.

Les signaux de sorties étant de moins larges bandes, leur *résolution spectrale* est plus haute i.e. l'incertitude sur leur fréquence moyenne est plus basse. Dans un arbre dyadique (figure 6), la décomposition du signal $[c(m,\mathbb{Z})]$ étant itérée par rapport à m, ces bandes rétrécissent à chaque itération. Pour cette raison principalement, l'arbre d'analyse dyadique est en fait un algorithme d'*analyse multirésolution* [11].

Si la base $\{\varphi_{m+1,n}, \psi_{m+1,n}\}_{n\in\mathbb{Z}}$ est orthonormale (§1.6) i.e. pour tout $n,n' \in \mathbb{Z}$, si

$$\langle \varphi_{mn'} | \varphi_{m,n'-n} \rangle = \delta(n), \tag{2.19}$$

$$\langle \psi_{mn'} | \psi_{m,n'-n} \rangle = \delta(n) \tag{2.20}$$

et

$$\langle \varphi_{mn} | \psi_{mn'} \rangle = 0, \tag{2.21}$$

alors les filtres $\tilde{g}$ et $\tilde{h}$ ont aussi les propriétés suivantes et sont dits *orthonormaux* :

$$\delta(n) \overset{(2.11)}{=} \sum_k \tilde{g}(k) \sum_l \tilde{g}(l) \langle \varphi_{m-1,-l} | \varphi_{m-1,-2n-k} \rangle \overset{(2.19)}{=} \sum_k \tilde{g}(k) \sum_l \tilde{g}(l) \delta(-2n-k+l) = \sum_k \tilde{g}(k)\tilde{g}(k+2n), \tag{2.22}$$

$$\delta(n) \overset{(2.11)}{=} \sum_k \tilde{h}(k) \sum_l \tilde{h}(l) \langle \varphi_{m-1,-l} | \varphi_{m-1,-2n-k} \rangle \overset{(2.20)}{=} \sum_k \tilde{h}(k) \sum_l \tilde{h}(l) \delta(-2n-k+l) = \sum_k \tilde{h}(k)\tilde{h}(k+2n) \tag{2.23}$$

et

$$0 \overset{(2.11)}{=} \sum_k \tilde{g}(k) \sum_l \tilde{h}(l) \langle \varphi_{m-1,-l} | \varphi_{m-1,-k} \rangle \overset{(2.21)}{=} \sum_k \tilde{g}(k) \sum_l \tilde{h}(l) \delta(-k+l) = \sum_k \tilde{g}(k)\tilde{h}(k). \tag{2.24}$$

Parmi les solutions possibles à (2.24), la plus simple est la solution *paraunitaire* [11] i.e.

$$\tilde{h}(n) = (-1)^n \tilde{g}(p-1-n)\big|_{p \text{ pair}}, \tag{2.25}$$

et les filtres $\tilde{g}, \tilde{h}$ sont alors dits paraunitaires.

2.6 Fonctions d'ondelettes courantes

Dans cette section, nous montrons d'abord comment calculer une fonction d'ondelette à partir de leur filtre $\tilde{g}$. Ensuite, dans §2.7.1 et §2.7.2, nous calculons deux fonctions d'ondelettes courantes : *Haar* et *Daubechies d'ordre 4* [11].

On peut noter que pour $m=-1, n=0$, l'équation (2.11) est un cas particulier de

$$\left[\varphi^{i+1} \quad \psi^{i+1}\right] = \sum_k \tilde{g}(k)\left[\varphi^{i}_{-1,-k} \quad \psi^{i}_{-1,-k}\right]. \tag{2.26}$$

Il est clair que si $[\varphi \quad \psi] = \lim_{i\to\infty}\left[\varphi^{i+1} \quad \psi^{i+1}\right] = \left[\varphi^{i} \quad \psi^{i}\right]$, alors (2.26) est équivalent à (2.11). Pour calculer toute fonction ϕ paramétrique, on peut donc calculer la forme paramétrique de φ^i puis de φ^i pour $i\to\infty$.

Pour calculer toute fonction φ ou ψ non paramétrique, on peut effectuer l'opération suivante pour $l=1,2\ldots$ et $\pm n=1,3,5\ldots$:

$$\left[\varphi(n/2^l) \quad \psi(n/2^l)\right] = \sum_k \varphi(n/2^{l-1}+k)\left[\tilde{g}(k) \quad \tilde{h}(k)\right]. \tag{2.27}$$

Étant donné que pour $l=1$ on effectue $[\varphi(n/2) \quad \psi(n/2)] = \sum_k \varphi(n+k)\left[\tilde{g}(k) \quad \tilde{h}(k)\right]$, il est clair qu'il faut connaître $[\varphi(\mathbb{Z})]$. Il est démontré [11] que si $\Phi(0)=\sqrt{2}$, alors

$$\sum_k \varphi(k) = 1 \tag{2.28}$$

et

$$\varphi(r)\big|_{-r\in(\text{support de } g)} = 0. \tag{2.29}$$

2.6.1 Haar

Le filtre $\tilde{g}$ de l'ondelette de Haar est tel que

$$\tilde{g}(n) = \begin{cases} 1/\sqrt{2} & n=-1,0 \\ 0 & \text{sinon} \end{cases}, \tag{2.30}$$

donc $\tilde{g}$ possède la propriété essentielle (2.17) car

$$\sum_k \tilde{g}(k) = \tilde{g}(-1)+\tilde{g}(0) = \sqrt{.5}+\sqrt{.5} = \sqrt{2}.$$

Ce filtre $\tilde{g}$ est orthonormal i.e. il possède la propriété (2.22), car

$$\begin{aligned}\sum_k \tilde{g}(k)\tilde{g}(k+2n) &= \tilde{g}(-1)\tilde{g}(-1+2k)+\tilde{g}(0)\tilde{g}(2k) = \sqrt{.5}\tilde{g}(-1+2k)+\sqrt{.5}\tilde{g}(2k) \\ &= \begin{cases} \sqrt{.5}\tilde{g}(-1)+\sqrt{.5}\tilde{g}(0) = .5+.5 = 1 & n=0 \\ 0 & n\neq 0 \end{cases} = \delta(n)\end{aligned}$$

Soit $$\varphi^0(r)=\begin{cases}1 & 0<r<1\\ 0 & \text{sinon}\end{cases},$$

alors d'après (2.26),

$$\varphi^1(r)=\sum_k \tilde{g}(k)\sqrt{2}\varphi^0(2r+k)=\sqrt{2}\tilde{g}(-1)\varphi(2r-1)+\sqrt{2}\tilde{g}(0)\varphi(2r)=\varphi^0(2r-1)+\varphi^0(2r)=\varphi^0(r)$$

et alors il est clair que $$\varphi=\varphi^\infty=\ldots=\varphi^1=\varphi^0.$$

Les filtres $\tilde{g},\tilde{h}$ de l'ondelette de Haar sont paraunitaires i.e. d'après (2.25),

$$\tilde{h}(n)\big|_{p=0}=\begin{cases}\left[(-1)^0\tilde{g}(-1)\ \ (-1)^{-1}\tilde{g}(0)\right]=\left[1/\sqrt{2}\ \ -1/\sqrt{2}\right] & n=[-1,0]\\ 0 & \text{sinon}\end{cases} \qquad (2.31)$$

et alors d'après (2.11),

$$\psi(r)=\sum_k h(k)\sqrt{2}\varphi(2r+k)=\sqrt{2}h(-1)\varphi(2r-1)+\sqrt{2}h(0)\varphi(2r)=-\varphi(2r-1)+\varphi(2r)=\begin{cases}1 & 0<r<0.5\\ -1 & 0.5<r<1\\ 0 & \text{sinon}\end{cases},$$

ψ étant donc la fonction d'ondelette de *Haar* (figure 4a).

2.6.2 Daubechies d'ordre 4

Le filtre $\tilde{g}$ de l'ondelette de Daubechies d'ordre 4 est tel que

$$\tilde{g}(n)=\frac{1}{\sqrt{32}}\begin{cases}\left[1-\sqrt{3}\ \ 3-\sqrt{3}\ \ 3+\sqrt{3}\ \ 1+\sqrt{3}\right] & n=[-3\ \ -2\ \ -1\ \ 0]\\ 0 & \text{sinon}\end{cases},$$

ce qui respecte la propriété (2.17).

Les fonctions φ et ψ de Daubechies ne sont pas paramétriques. Soit $\Phi(0)=\sqrt{2}$, alors d'après (2.29), $$\varphi(r)\big|_{r\le 0,r\ge 3}=0,$$

donc $$\begin{bmatrix}1\\ \varphi(1)\end{bmatrix}\underset{(2.11)}{=}\begin{bmatrix}\varphi(1)+\varphi(2)\\ \frac{1}{4}\left(\left(3+\sqrt{3}\right)\varphi(1)+\left(1+\sqrt{3}\right)\varphi(2)\right)\end{bmatrix}\rightarrow\begin{bmatrix}1 & 1\\ \sqrt{3}-1 & 1+\sqrt{3}\end{bmatrix}\begin{bmatrix}\varphi(1)\\ \varphi(2)\end{bmatrix}=\begin{bmatrix}1\\ 0\end{bmatrix}$$

et donc $$[\varphi(1)\ \ \varphi(2)]=\frac{\begin{bmatrix}-1-\sqrt{3} & \sqrt{3}-1\end{bmatrix}}{\left(\sqrt{3}-1\right)-\left(1+\sqrt{3}\right)}=\frac{1}{2}\left[1+\sqrt{3}\ \ 1-\sqrt{3}\right]$$

Pour $l=1$,

$$[\varphi(1/2) \quad \varphi(3/2) \quad \varphi(5/2)] = \frac{1}{4}\left[\left(1+\sqrt{3}\right)\varphi(1) \quad \left(3-\sqrt{3}\right)\varphi(1)+\left(3+\sqrt{3}\right)\varphi(2) \quad \left(1+\sqrt{3}\right)\varphi(2)\right] = \frac{1}{8}\left[4+2\sqrt{3} \quad 0 \quad -2\right].$$

Les filtres $\tilde{g}, \tilde{h}$ des ondelettes de Daubechies sont paraunitaires.

2.7 La transformée en paquets d'ondelettes

Dans §2.4, nous avons vu que $[c(m+1,\mathbb{Z}) \quad d(m+1,\mathbb{Z})]$ est une RC de $[c(m,\mathbb{Z})]$ et que dans l'arbre de décomposition dyadique (figure 6), parmi les branches des coefficients c et d, seules celles de c sont ramifiées en d'autres branches. Lorsque d'autres branches que c sont également ramifiées, on a alors un arbre de *paquets d'ondelette* et les résultats s'appellent *coefficients* de paquets d'ondelette (CPO). Soit c_0, le signal à l'entrée de l'arbre. L'identité suivante est à la figure 7 ce que (2.12) est à la figure 5c :

$$[c_{2i+1}(n) \quad c_{2i+2}(n)] \equiv \sum_k c_i(2n-k)\left[\tilde{g}(k) \quad \tilde{h}(k)\right]. \tag{2.32}$$

Aussi, $[c_{2i+1}(\mathbb{Z}) \quad c_{2i+2}(\mathbb{Z})]$ est une RC de $[c_i(\mathbb{Z})]$. Donc, toute RC de $[c_{2i+1}(\mathbb{Z}) \quad c_{2i+2}(\mathbb{Z})]$ est aussi une RC de $[c_i(\mathbb{Z})]$. Par conséquent, dans un arbre à m niveaux de ramification avec c_0 à l'entrée, il y a 2^m RC possibles de c_0, incluant c_0 lui-même.

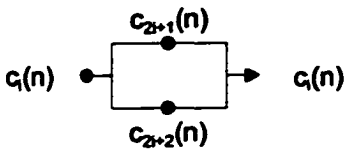


Figure 7 Banc de filtres dyadique

2.8 Sélection d'un arbre de paquets d'ondelettes

Soit $\mathbb{D}_0$, un ensemble tel que $\mathbb{D}_0 \subset \mathbb{Z}$ et $\mathbb{D}_i$, l'ensemble tel que le signal $[c_i(\mathbb{D}_i)]$ dépend entièrement et au moins d'un échantillon du signal $[c_0(\mathbb{D}_0)]$. Soit $\mathbb{D}$, un ensemble de valeurs quelconques et *arbre*, l'ensemble des valeurs de i des signaux c_i à toutes les

sorties de l'arbre, et c une fonction telle que $[c(\mathbb{D})] = [c_i(\mathbb{D}_i)]_{\text{dearbre}}$. On peut trouver, parmi plusieurs ROC possibles dans l'arbre, celle qui minimise $D(c(\mathbb{D}))$ (1.9) à l'aide de la méthode *local discriminant basis* (LDB), de Saito [10]. Le principe de la LDB est le suivant : dans un arbre initialement à une branche, nommée $i=0$, toute branche i de l'arbre est divisée en deux autres branches $2i+1$ et $2i+2$ si et seulement si la comparaison suivante est vérifiée : $D(\{c_{2i+1}(\mathbb{D}_{2i+1}), c_{2i+2}(\mathbb{D}_{2i+2})\}) > D(\{c_i(\mathbb{D}_i)\})$. L'arbre ainsi ramifié est celui qui minimise $D(c(\mathbb{D}))$.

Nous avons vu (§1.7) que pour $A' \subseteq A$ et $B' \subseteq B$, la comparaison $D(c(B')) > D(f(A'))$ ne peut être effectuée que si $|c|_{B'} = |f|_{A'}$ et $|c|_B = |f|_A$. Cette dernière comparaison est à la précédente ce que $|c|_{B'} = |f|_{A'}$ est à l'identité suivante :

$$\|c_i\|^2_{\mathbb{D}_i} = \|c_{2i+1}\|^2_{\mathbb{D}_{2i+1}} + \|c_{2i+2}\|^2_{\mathbb{D}_{2i+2}} . \qquad (2.33)$$

CHAPITRE 3

REVUE DE LA LITTÉRATURE

Les six familles de caractéristiques issues d'une TO et ayant été évaluées sur des classes de caractères sont expliquées dans §3.1-5. Dans le texte, $\mathbb{D}(m)$ désigne l'ensemble tel que le signal $[c(m,\mathbb{D}(m))]$ dépend entièrement et au moins d'un échantillon du signal $[c(0,\mathbb{D}(0))]$. Le terme X désigne l'ensemble des caractéristiques, et $\tilde{f}$, l'image f après normalisation (1.20).

3.1 Famille F1 : invariants des moments d'ondelettes

Les moments d'ondelettes (MO) sont un cas des moments définis par (1.23) lorsque $g_{mn} = \psi_{mn} = 2^{-m/2}\psi\left(2^{-m}r - b_0 n\right)$.

Soit
$$F(r,q) \equiv r\int_0^{2\pi} \tilde{f}_p(r,\theta)e^{jq\theta}d\theta \tag{3.1}$$

et
$$d(m,n,q) \equiv \int \psi_{mn}(r)F(r,q)dr\,, \tag{3.2}$$

alors
$$X \subset \left\{d(m,\mathbb{Z},q)\right\}_{-m=0\ldots-M,\,q=0\ldots Q}$$

Les auteurs Shen et Ip [3] utilisent $b_0 = 0.5$, $Q = 3$, $M = 3$. L'ensemble X, ainsi qu'un ensemble d'invariants de moments de Zernike ((1.24) dans le module de (1.23)), sont chacun évalués sur des exemples synthétiques des majuscules A à Z de différentes tailles et angles de rotation : le taux d'erreur de classification obtenu est plus bas avec les invariants de moments d'ondelette qu'avec ceux de Zernike.

3.2 Famille F2 : descripteurs de Fourier-ondelette

Tout d'abord, les images de caractère sont *polarisées* [4] i.e. elles sont filtrées dans la direction des axes d'une grille polaire uniforme (figure 8) puis rééchantillonnées sur

cette grille. Soit Θ, le nombres d'échantillons sur chaque cercle de cette grille, w_θ et w_r, les filtres angulaire et radial, $\bar{f}_\theta$, l'image $\tilde{f}_p$ filtrée le long de l'axe angulaire, et $\bar{f}$, l'image $\bar{f}_\theta$ filtrée le long de l'axe radial i.e.

$$\bar{f}_\theta(r,k) \equiv \int \tilde{f}_p(r,\theta) w_\theta(\theta - 2\pi k/\Theta) d\theta \tag{3.3}$$

et

$$\bar{f}(n,k) \equiv \int \bar{f}_\theta(r,k) w_r(r-n) dr\,. \tag{3.4}$$

Les filtres w_θ et w_r sont rectangulaires. Plus précisément :

$$w_\theta(\cdot) = \begin{cases} \Theta/2\pi & 0 < \cdot < 2\pi/\Theta \\ 0 & \text{sinon} \end{cases},\quad w_r(\cdot) = \begin{cases} 1 & 0 < \cdot < 1 \\ 0 & \text{sinon} \end{cases}. \tag{3.5}$$

Ensuite,

$$c(0,n,q) = \left| \frac{1}{\Theta} \sum_{k=0}^{\Theta-1} \bar{f}(n,k) e^{j2\pi qk/\Theta} \right|, \tag{3.6}$$

et alors

$$\mathrm{X} = \{c(m, \mathbb{D}(m), q), d(k, \mathbb{D}(m), q)\}_{k=1\ldots m, q=0\ldots Q},$$

avec

$$\mathbb{D}(0) = \{1 \ldots N\}, \tag{3.7}$$

N étant une valeur à choisir.

Les auteurs Chen et Bui [4] utilisent $m = 6$. Les images de caractères étant de 64×64 pixels, les auteurs utilisent $N = \Theta = 64, Q = \Theta/2 = 32$. X est évalué sur des caractères chinois imprimés de différentes tailles et angles de rotation : le taux d'erreur de classification obtenu est nul dans la plupart des tests.

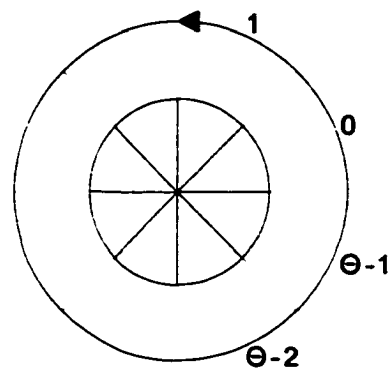


Figure 8 Grille polaire à Θ échantillons par cercle

3.3 Famille F3 : Signatures Ring-Projection-Wavelet-Fractal

Soit l'ensemble de courbes $\{[c(0,\mathbf{D}(0))],[c(m,\mathbf{D}(m))],[d(m,\mathbf{D}(m))]\}_{m=1}^{3}$ lorsque

$$c(0,n)=\int_0^{2\pi}\tilde{f}_p(n,\theta)d\theta\approx\frac{1}{\Theta}\sum_{k=0}^{\Theta-1}\tilde{f}_p(n,2\pi k/\Theta). \tag{3.8}$$

Toute courbe qui ne se croise pas possède une *dimension de Bouligand-Minkowski* (*divider dimension* [5]), définie par

$$divd(courbe)\equiv-\lim_{\delta\to 0}(\log M_\delta(courbe)/\log\delta).$$

Pour calculer la dimension de Bouligand-Minkowski de toute courbe, on insère l'un à la suite de l'autre des segments entre des points de la courbe uniformément séparés d'une distance euclidienne δ : $M_\delta(courbe)$ est le nombre de segments nécessaire pour couvrir toute la courbe. Les signatures *Ring-Projection-Wavelet-Fractal*, de Tang et *al.* [5], sont alors définis par

$$\mathrm{X}=\left\{divd(courbe)\middle|courbe\in\{[c(0,\mathbf{D}(0))],[c(m,\mathbf{D}(m))],[d(m,\mathbf{D}(m))]\}_{m=1}^{3}\right\}$$

3.4 Famille F4 : descripteurs d'ondelette

Soit $$c(0,n)=z(n),$$

alors $$\mathrm{X}=\{\mathrm{Re}(c(m,n)),\mathrm{Im}(c(m,n))\}_{n\in\mathbb{Z}}.$$

Pour rendre les descripteurs d'ondelette (DO) invariants au déplacement le long du contour, l'origine de coordonnée n peut être normalisée de deux façons. Dans la première, l'origine est fixée au point le plus proche du coin supérieur gauche du *minimum bounding rectangle*. Dans la deuxième, l'origine est fixée pour annuler la phase du DF (1.25) à $q=1$ i.e. $\angle c(1)\equiv 0$.

Les auteurs Wunsch et Laine [6] utilisent $m=3,4,5$. L'ensemble X, ainsi qu'un ensemble de DF (1.25) sont chacun évalués sur des chiffres : le taux d'erreur de classification obtenu est plus bas avec les DO (0%) qu'avec les DF.

3.5 Familles F5 et F6 : coefficients de l'arbre dyadique bidimensionnel

Dans un arbre d'analyse dyadique *monodimensionnel* (figure 5), tous les bancs opèrent le long de la même dimension n_1 ou n_2 de leur signal d'entrée $[c(m,n_1,n_2)]_{n_1,n_2\in\mathbb{Z}}$ i.e.

$$\begin{bmatrix} c(m+1,n_1,n_2) \\ d(m+1,n_1,n_2) \end{bmatrix} \overset{(2.12)}{=} \sum_k \begin{bmatrix} c_0(m,2n_1-k,n_2) \\ c_0(m,2n_1-k,n_2) \end{bmatrix} \begin{bmatrix} g(n_1) \\ h(n_1) \end{bmatrix}. \tag{3.9}$$

Si à chacune des sorties c et d on ajoute un autre banc d'analyse qui opère le long de l'autre dimension n_2, on obtient un arbre d'analyse dyadique *bidimensionnel* (figure 9)

i.e.
$$\begin{bmatrix} c_0(m+1,n_1,n_2) & c_1(m+1,n_1,n_2) \\ c_2(m+1,n_1,n_2) & c_3(m+1,n_1,n_2) \end{bmatrix} \overset{(2.12)}{=} \sum_l \begin{bmatrix} c(m+1,n_1,2n_2-l)\begin{bmatrix} g(n_2) & h(n_2) \end{bmatrix} \\ d(m+1,n_1,2n_2-l)\begin{bmatrix} g(n_2) & h(n_2) \end{bmatrix} \end{bmatrix}$$
$$\overset{(3.9)}{=} \sum_{k,l} \begin{bmatrix} c(m,2n_1-k,2n_2-l)g(n_1)\begin{bmatrix} g(n_2) & h(n_2) \end{bmatrix} \\ d(m,2n_1-k,2n_2-l)h(n_1)\begin{bmatrix} g(n_2) & h(n_2) \end{bmatrix} \end{bmatrix}.$$

Dans Lee et *al.* [7], $c_0(0,n_1,n_2)=\tilde{f}(n_1,n_2)$.

$$\mathbf{X}_1 \equiv \{c_0(2,\mathbf{D}(2)), c_i(m,\mathbf{D}(m))\}_{i=1,2,3,m=1,2}, \mathbf{X}_2 \equiv \mathbf{X}_1 \cup \{c_0(3,\mathbf{D}(3))\}$$

et $$\mathbf{X} = \mathbf{X}_1, \mathbf{X}_2.$$

Dans Laine et Schuler [8], l'image est d'abord rééchantillonnée sur une autre grille cartésienne par interpolation bilinéaire selon

$$\tilde{f}_1(n_1,n_2) \equiv \frac{\tilde{f}(\lceil n_1/3 \rceil, n_2) + \tilde{f}(\lfloor n_1/3 \rfloor, n_2)}{2}, \quad \tilde{f}_2(n_1,n_2) \equiv \frac{\tilde{f}_1(n_1,\lceil n_2/2 \rceil) + \tilde{f}_1(n_1,\lfloor n_2/2 \rfloor)}{2},$$

puis rééchantillonnée sur une grille hexagonale. Soit f_h, cette image "hexagonalisée" i.e.

$$f_h(n_1,n_2) \equiv \tilde{f}(n_1-n_2,n_2).$$

Enfin, $$c_0(0,n_1,n_2)=f_h(n_1,n_2),$$

$$\mathbf{X} = \{c_i(4,\mathbf{D}(4))\}_{i=0,1,2,3}.$$

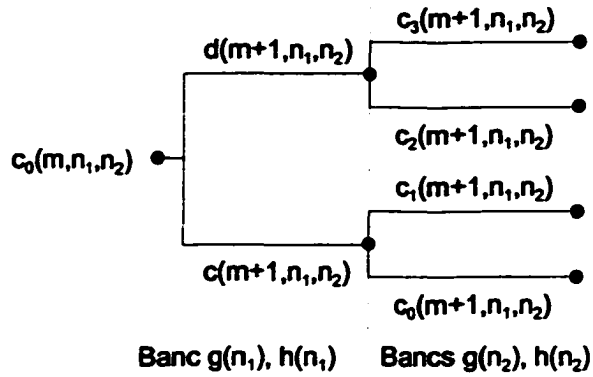


Figure 9 Banc de filtres bidimensionnel

3.6 Problématique à résoudre

Après avoir étudié les familles 1 à 6, nous avons estimé que les quatre points suivants méritent une attention particulière :

P1. Dans l'article sur les IMO, le choix de Q ($Q=3$) n'est pas expliqué par les auteurs. Dans l'article sur les DFO, le choix de N et Θ est plutôt heuristique ($N=\Theta=64$), et le choix de Q est imposé par celui de Θ ($Q=\Theta/2$), lui-même heuristique.

P2. L'opération "$|\cdot|$", qui transforme les moments en leurs invariants (§1.9), n'est pas une projection (1.12). Par conséquent, les IMO ne peuvent former de ROC de l'image. Les MO le peuvent, car on voit (3.1,3.2) qu'ils sont le résultat de deux transformées consécutives à partir de l'image $\left|\widetilde{rf}_p(r,\theta)\right|_{0<r<1,0<\theta<2\pi}$: la TF de $\left|\widetilde{rf}_p(r,\theta)\right|_{0<\theta<2\pi}$ pour chaque valeur réelle de r entre 0 et 1, suivie de la TO de $[F(r,\cdot)]_{0<r<1}$. Nous avons vu que la TF permet des ROC (§1.9) et que la TO le peut aussi (§2.5). Si la TF et la TO le sont, il est clair que les MO le sont aussi. Par contre, les IMO sont invariants à la rotation et réflexion (§1.9). De plus, les invariants sont simples à calculer à partir des moments, car ce ne sont que leurs modules (§1.9).

P3. Les 6 familles sont issues d'une TO, qui n'est qu'un exemple de la TPO (§2.6).

P4. Dans les équations (3.1, 3.6, 3.8), on voit que l'intégrale le long de θ d'une fonction de $\tilde{f}_p(r,\theta)$ est estimée. Dans l'équation (3.8), on voit que cette intégrale ($\int_0^{2\pi} \tilde{f}_p(r,\theta)d\theta$) est estimée par $\frac{1}{\Theta}\sum_{k=0}^{\Theta-1} f_p(n, 2\pi k/\Theta)$. Dans l'article sur la famille 1 [3], la façon d'estimer cette intégrale ($\int_0^{2\pi} \tilde{f}_p(r,\theta)e^{jq\theta}d\theta$ dans (3.1)), n'est pas spécifiée. Dans l'article sur la famille 2, le terme $\frac{1}{\Theta}\sum_{k=0}^{\Theta-1} \bar{f}_\theta(r,k)e^{j2\pi kq/\Theta}$ nous semble servir à estimer l'intégrale $\int_0^{2\pi} \tilde{f}_p(r,\theta)e^{jq\theta}d\theta$. En effet, séparons dans l'équation (3.6) les termes en θ des termes en r i.e.

$$c(0,n,q) \overset{(3.3.4)}{=} \left|\sum_{k=0}^{\Theta-1}\frac{1}{\Theta}\left(\int \bar{f}_\theta(r,k)w(r-n)dr\right)e^{j2\pi qk/\Theta}\right| = \left|\int w(r-n)\times\sum_{k=0}^{\Theta-1}\frac{1}{\Theta}\bar{f}_\theta(r,k)e^{j2\pi qk/\Theta}dr\right|.$$

Lorsque $\Theta=\infty$, et qu'alors $w_\theta \overset{(3.5)}{=} \delta$, et donc que $\bar{f}_\theta(r,k) \overset{(3.3)}{=} \tilde{f}_p(r,2\pi k/\Theta)$, on voit que le terme global en θ du membre de droite de l'équation précédente devient

$$\lim_{\Theta\to\infty}\frac{1}{\Theta}\sum_{k=0}^{\Theta-1}\bar{f}_\theta(r,k)e^{j2\pi kq/\Theta} = \int_0^{2\pi}\bar{f}_\theta(r,\theta\Theta/2\pi)e^{jq\theta}d\theta = \int_0^{2\pi}\tilde{f}_p(r,\theta)e^{jq\theta}d\theta.$$

Or, pour $\Theta<\infty$, les estimations dans (3.6) et (3.8) ne donnent qu'une approximation de l'intégrale et pour $\Theta=\infty$, le calcul direct de (3.6) et (3.8), qui sont chacun une somme de Θ termes, prendrait un temps infini.

3.7 Solutions proposées

Pour contribuer à la résolution des 4 points mentionnés ci-haut, notre démarche consistera, respectivement, à :

1. Augmenter Q et N tant que cela fait diminuer le taux d'erreur de classification ;
2. Éliminer l'opération "$|\cdot|$" présente dans le calcul des IMO, puis la rajouter après avoir évalué les MO ;

3. Remplacer les arbres dyadiques par des arbres de paquets d'ondelettes. Les MO deviendront alors les MPO (moments de paquets d'ondelettes) ;
4. Fixer Θ à $\Theta=\infty$ puis remplacer le calcul direct de (3.6) par une équation paramétrique équivalente.

Les sections suivantes §3.7.1-4 expliquent en quoi consistent les solutions 1 à 4 respectivement et à l'intérieur de quelles valeurs de paramètres elles doivent être appliquées. Déterminons donc ces valeurs. Tout d'abord, on peut noter que (3.2) est à $c(m,n,\cdot)\approx 2^{m/2}F(2^m n,\cdot)$ ce que (2.7) est à (2.13), et donc d'après la solution 3,

$$c_0(n,\cdot)=F(2^m n,\cdot). \tag{3.10}$$

Pour que la TF donne une ROC de $\left|\tilde{f}_p(r,\theta)\right|_{0<\theta<2\pi}$, il faut que $q\in\mathbb{Z}_+^*$ dans (3.1), car $\{g_q | g_q(\theta)=e^{iq\theta}, q\in\mathbb{Z}_+^*\}$ forme une BOC de $L^2([0,2\pi])$ (§1.9). Pour que la TPO donne une ROC, il faut que les filtres $\tilde{g},\tilde{h}$ soient orthonormaux (§2.5).

Soit Y, l'ensemble des IMO modifiés et de paramètres réglés selon ces quatre points. Soit $d,P,Q_{\max}$, trois paramètres à choisir et soit $d^P\equiv N$ et $d^{P-p}\equiv 2^m$, alors d'après (3.7),

$$\mathbb{D}_0=\{1\ldots d^P\} \tag{3.11}$$

et d'après (3.10), $$c_0(n,\cdot)=F(d^{P-p}n,\cdot). \tag{3.12}$$

Soit (1.1) et (1.5), et soit $\varepsilon(p,\cdot)$, la valeur de $\bar{e}$ lorsque $\mathrm{Y}=c(\mathbb{D}\times\{0\ldots\cdot\})$. Soit (p_{op},Q_{op}), la paire de valeurs dans $\{0\ldots P\}\times\{0\ldots Q_{\max}\}$ qui minimise ε i.e.

$$(p_{op},Q_{op})\equiv\arg\min([\varepsilon(\{0\ldots P\}\times\{0\ldots Q_{\max}\})]). \tag{3.13}$$

Soit $\varepsilon(\cdot,\cdot,I)$, la valeur estimée de $\varepsilon(\cdot,\cdot)$ sur I exemples par classe. Pour être certain de pouvoir trouver la valeur exacte de $\varepsilon(\cdot,\cdot)$, il faudrait que I soit infini, ce qui est irréalisable. Nous supposons que lorsque I augmente, $\varepsilon(\cdot,\cdot,I)$ oscille autour de $\varepsilon(\cdot,\cdot)$ de moins en moins. Dans le texte, I_{op} désigne la valeur minimale de I à partir de laquelle $\varepsilon(p_{op},Q_{op},I)$ n'oscille presque plus.

3.7.1 Solution 1

La solution 1 consiste à effectuer (3.13) pour $I > I_{op}$.

3.7.2 Solution 2

La solution 2 consiste à effectuer $Y = \{c(\mathbb{D},q)\}_{q=0}^{Q_{op}}$ puis $Y = \{c(\mathbb{D},0), \mathrm{Re}\, c(\mathbb{D},q), Im\ c(\mathbb{D},q)\}_{q=1}^{Q_{op}}$ pour $p = p_{op}$ et $I = I_{op}$.

3.7.3 Solution 3

La solution 3 consiste à sélectionner, parmi plusieurs ROC possibles dans l'ensemble de tous les arbres d'entrée $c_0(n,q)$ tel que $q \in \{0...Q_{op}\}$, celle qui minimise $D(\{c(\mathbb{D}\times\{0...Q_{op}\})\})$. Nous avons vu (§2.7) que pour trouver, parmi plusieurs ROC possibles dans un arbre d'entrée $c_0(n)$, celle qui minimise $D(c(\mathbb{D}))$, on utilise la méthode LDB et que celle-ci inclut une suite de comparaisons $D(\{c_{2i+1}(\mathbb{D}_{2i+1}), c_{2i+2}(\mathbb{D}_{2i+2})\}) > D(\{c_i(\mathbb{D}_i)\})$ ne pouvant être effectuées que si la condition (2.33) est vérifiée.

Comme $D(c(\mathbb{D}))$ est à cette condition ce que $D(\{c(\mathbb{D}\times\{\cdot\})\})$ est à la condition $\|c_{2i+1}\|^2_{\mathbb{D}_{2i+1}\times\{\}} + \|c_{2i+2}\|^2_{\mathbb{D}_{2i+2}\times\{\}} = \|c_i\|^2_{\mathbb{D}_i\times\{\}}$, déterminons si celle-ci peut être vérifiée.

D'après (2.32), si g et h sont réels, alors

$$\mathrm{Re}\begin{bmatrix} c_{2i+1}(n) \\ c_{2i+2}(n) \end{bmatrix} + j\,\mathrm{Im}\begin{bmatrix} c_{2i+1}(n) \\ c_{2i+2}(n) \end{bmatrix} = \sum_k \left(\mathrm{Re}c_i(2n-k) + j\,\mathrm{Im}c_i(2n-k)\right)\begin{bmatrix} g(k) \\ h(k) \end{bmatrix}$$

et alors

$$\begin{matrix} \mathrm{Re}\begin{bmatrix} c_{2i+1}(n) & c_{2i+2}(n) \end{bmatrix} \\ \mathrm{Im}\begin{bmatrix} c_{2i+1}(n) & c_{2i+2}(n) \end{bmatrix} \end{matrix} = \begin{matrix} \sum_k \mathrm{Re}c_i(2n-k)\begin{bmatrix} g(k) & h(k) \end{bmatrix} \\ \sum_k \mathrm{Im}c_i(2n-k)\begin{bmatrix} g(k) & h(k) \end{bmatrix} \end{matrix}. \qquad (3.14)$$

Donc, si g et h sont orthonormaux, alors d'après (2.33),

$$\begin{array}{ccccc}
\sum_{n\in D_{2i+1}\times\{\}}\left(\mathrm{Re}(c_{2i+1}(n))^2 + \mathrm{Re}(c_{2i+2}(n))^2\right) & = & \sum_{n\in D_i\times\{\}}\mathrm{Re}(c_i(n))^2 \\
+ & & + \\
\sum_{n\in D_{2i+1}\times\{\}}\left(\mathrm{Im}(c_{2i+1}(n))^2 + \mathrm{Im}(c_{2i+2}(n))^2\right) & = & \sum_{n\in D_i\times\{\}}\mathrm{Im}(c_i(n))^2 \\
= & & = \\
\sum_{n\in D_{2i+1}\times\{\}}\left(|c_{2i+1}(n)|^2 + |c_{2i+2}(n)|^2 \right) & = & \sum_{n\in D_i\times\{\}} |c_i(n)|^2 \\
= & & = \\
\left\| |c_{2i+1}| \right\|^2_{D_{2i+1}\times\{\}} + \left\| |c_{2i+2}| \right\|^2_{D_{2i+2}\times\{\}} & = & \left\| |c_i| \right\|^2_{D_i\times\{\}}
\end{array} \qquad (3.15)$$

ce qui vérifie la dite condition.

3.7.4 Solution 4

Définissons le *pixel* associé à tout échantillon de l'image comme étant la région des points dont le dit échantillon est le plus proche ; cette région est évidemment un carré centré sur l'échantillon. Dans ce mémoire, f est échantillonné sur une grille $\mathbb{Z}^2$ et donc les contours de tous les pixels de cette grille forment eux-mêmes une autre grille, décalée de (0.5,0.5) de la grille $\mathbb{Z}^2$ (figure 10). Assignons à tout point de l'image f la valeur de l'échantillon de la grille le plus proche i.e.

$$f(x,y) = f([x],[y]).$$

D'après cette dernière équation, $f(x,y)$ est constant (donc paramétrique) par pixel. Par conséquent, l'intégrale de toute fonction paramétrique de $f(x,y)$ a aussi une forme paramétrique. La solution 1 consiste donc à calculer la forme paramétrique de l'intégrale le long de θ de la fonction de $\tilde{f}_p(r,\theta)$ entre chaque intersection cercle-grille décalée consécutive ; l'intégrale étant effectuée tout le long du cercle i.e. pour $0<\theta<2\pi$, sa forme paramétrique est alors la somme des équations paramétriques obtenues à toutes les intersections cercle-grille décalée.

Soit θ_i, l'angle d'une des intersections du cercle avec la grille décalée, tel que

$$\theta_i < \theta_{i+1}, \theta_{8L} = \theta_0 + 2\pi, \qquad (3.16)$$

et soit $(x_{i+.5}, y_{i+.5})$, le centre du pixel entre θ_i et θ_{i+1} i.e.

$$\tilde{f}_p(n,\theta_i<\theta<\theta_{i+1}) \overset{(1.22)}{=} f(x_{i+.5},y_{i+.5}). \tag{3.17}$$

Soit $$G_{i+.5}(n,q) \equiv \int_{\theta_i}^{\theta_{i+1}} \tilde{f}_p(n,\theta) e^{-jq\theta}\, d\theta, \tag{3.18}$$

alors $$F(n,q) \overset{(3.1)}{=} n \sum_{i=0}^{8L-1} G_{i+.5}(n,q). \tag{3.19}$$

Comme $\tilde{f}_p(n,\theta)$ est constant entre θ_i et θ_{i+1}, on peut le sortir de l'intégrale (3.11), et

alors $$G_{i+.5}(n,q) = f(x_{i+.5},y_{i+.5}) \times \begin{cases} \theta_{i+1}-\theta_i & \text{si } q=0 \\ \dfrac{e^{-jq\theta_{i+1}} - e^{-jq\theta_i}}{jq} & \text{sinon} \end{cases}. \tag{3.20}$$

En insérant (3.13) dans (3.12), on obtient l'équation paramétrique cherchée.

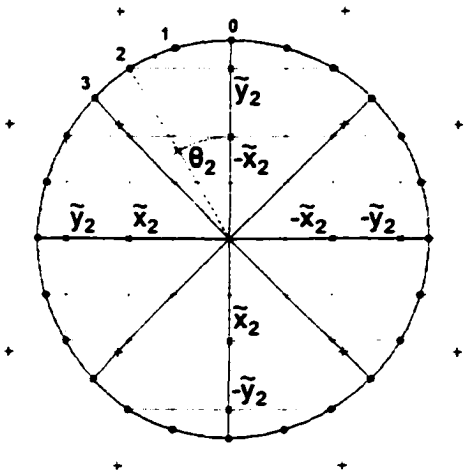


Figure 10 Grille décalée et cercle centré sur un pixel

'·' : intersections ;
'0...3' : numéros des intersections ;
'+' : échantillons de valeur $f(\pm\tilde{x}_{1.5}+a,\pm\tilde{y}_{1.5}+b), f(\pm\tilde{y}_{1.5}+a,\pm\tilde{x}_{1.5}+b)$;
droites continues : axes de symétrie passant par le centre et directions de réflexion sur ces axes.

3.8 Contribution à la résolution de la problématique

Rappelons (introduction) que l'objectif de ce mémoire est d'évaluer des ensembles de IMO modifiés et de paramètres réglés de façon à contribuer à la résolution des quatre points de la problématique (§3.6). Nous venons (§3.7) effectivement de modifier (solutions 2 et 3) les IMO et de régler (solutions 1 et 4) certains paramètres à des valeurs différentes des auteurs. Pour montrer en quoi chaque solution contribue à la résolution de la problématique, on compare les caractéristiques des choix des paramètres dans la littérature avec les caractéristiques de ceux que nous proposons (Tableau I).

Tableau I

Caractéristiques des choix de paramètres (§3.6) avant et après application des solutions (§3.7)

Problème	Paramètre	Famille	Valeur [3,4]	Caractéristique [3,4]	Valeur proposée	Caractéristique résultante
P4	Θ	F2	$64 (< \infty)$	permet approximation des caractéristiques	∞	permet calcul exact des caractéristiques
P1	N	F2	Θ	choix heuristique	d^{p_∞}	choix qui minimisent un critère (3.13)
	Q	F2	$\Theta/2$	choix heuristique	Q_{op}	
		F1	3	choix inexpliqué		
P2	X	F1	invariants	ne permet pas de ROC	moments	permet des ROC
P3	arbre	F1	dyadique	permet moins de RC	PO	permet plus de RC

CHAPITRE 4

MÉTHODOLOGIE

Les ensembles à évaluer étant des ensembles de MPO et de leurs invariants, la méthodologie consiste à calculer plusieurs MPO et leurs invariants, y sélectionner des sous-ensembles, puis évaluer ceux-ci (figure 11).

La sélection de caractéristiques dans un ensemble de variables X, d'un sous-ensemble Y_i de taille i et selon une fonction J a été expliquée dans §1.5. L'étape de sélection des MPO et IMPO consiste donc à effectuer quelques sélections pour $\mathrm{X} = \{c(\mathbb{D},0), \mathrm{Re}\, c(\mathbb{D},q), \mathrm{Im}\, c(\mathbb{D},q)\}_{q=1}^{Q_{\mathrm{op}}}$ puis $\mathrm{X} = \{c(\mathbb{D},q)\}_{q=0}^{Q_{\mathrm{op}}}$ avec $p = p_{\mathrm{op}}$. L'étape d'évaluation consiste à estimer $e(\mathrm{Y}_i)$ pour différentes valeurs de i.

Dans ce chapitre, chacune des sections §4.1 à §4.5 explique la conception d'une des étapes de la méthodologie, d'abord en présentant le cadre théorique de l'étape, puis en développant l'algorithme simulant l'étape et basé sur le cadre théorique. Ensuite, chacune des sections §4.6 à §4.9 explique comment les valeurs de certains paramètres de la méthodologie ont été délimités ou fixés. Chaque algorithme a été implanté dans un programme écrit en langage compilable/interprétable (annexes 1 et 2).

Dans tout algorithme, chaque variable a pour rôle de contenir la valeur d'un paramètre du monde réel. Aussi, une même variable présente à différents endroits peut représenter différents paramètres. Dans les algorithmes de ce chapitre, nous avons donné à la plupart des variables, le nom des paramètres qu'elle représentent, et nous avons listées les autres ainsi :

<variable> : < paramètre représenté 1>, < paramètre représenté 2>

À chaque ligne de cette liste, les paramètres sont listés dans l'ordre selon lequel ils le sont représentés. Dans les algorithme, tel que celui de la figure 11, l'expression non encadrée et située au coin gauche de toute boîte est la condition d'exécution de la routine dans la boîte. Dans ce texte, f_{ik} désigne l'image f du $i$ème exemple de classe k, et $F_{ik} \equiv F|_{f=f_{ik}}$.

Voici la liste de quelques variables de l'algorithme de la figure 11 :

- (p,Q) : $(p,Q),(p_{op},Q_{op})$;
- I : I, I_{op} ;

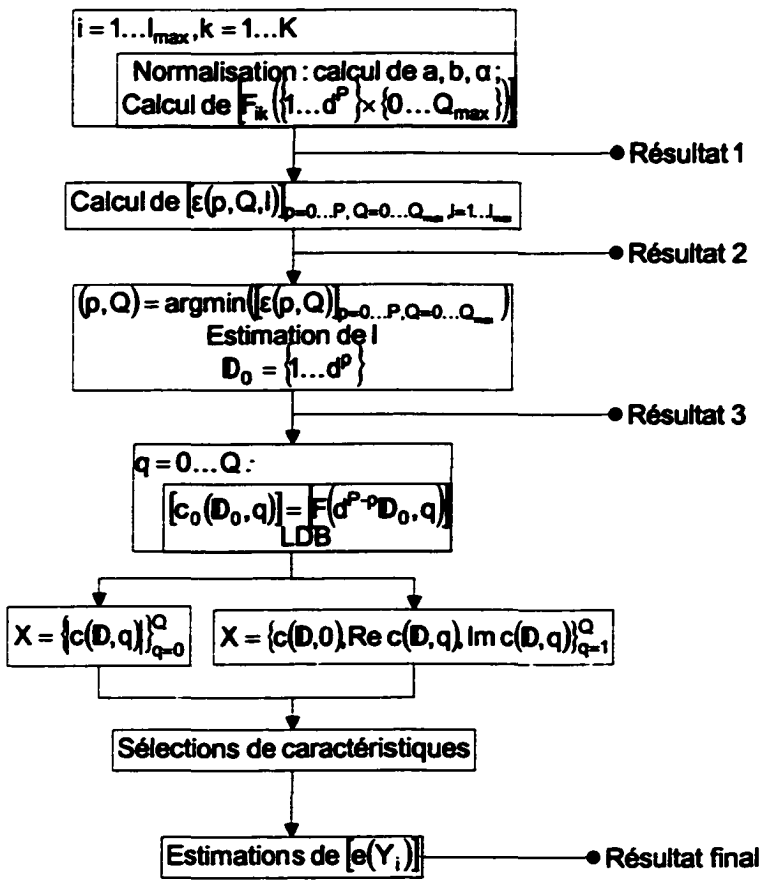


Figure 11 Méthodologie

4.1 Calcul de F

4.1.1 Cadre théorique

Les équations de calcul de $G_{i+.5}$ (3.17,3.18) contiennent des termes indicés : $x_i, y_i, x_{i+.5}, y_{i+.5}, \theta_i$. Expliquons comment les calculer. Chaque intersection cercle-grille est, bien sûr, sur une barre horizontale ou verticale de la grille (figure 9). Soit i, toute intersection cercle-grille ou cercle-axe de symétrie située en haut et à gauche du centre du cercle i.e. $x_i \le 0, y_i \ge 0$, soit $y_{h,i}$ l'ordonnée de la barre horizontale la plus proche sous l'intersection i, et soit

$$(\tilde{x}_i, \tilde{y}_i) \equiv (x_i - a, y_i - b) \tag{4.1}$$

et

$$re^{j\theta_i} \equiv \tilde{x}_i + j\tilde{y}_i . \tag{4.2}$$

Si l'intersection i est sur la barre horizontale,

alors

$$y_{i+1} = y_{h,i} \tag{4.3}$$

$$\tilde{x}_{i+1} \overset{(4.2)}{=} \sqrt{r^2 - \tilde{y}_{i+1}^2}, \tag{4.4}$$

et

$$(x_{i+.5}, y_{i+.5}) = (\lceil x_{i+1} \rceil, \lceil y_{i+1} \rceil), \tag{4.5}$$

sinon

$$x_{i+1} = \begin{cases} \lceil x_i \rceil - 0.5 & \text{si } \lceil x_i \rceil - 0.5 \ge \lceil x_i \rceil \\ \lceil x_i \rceil - 1.5 & \text{sinon} \end{cases}, \tag{4.6}$$

$$\tilde{y}_{i+1} \overset{(4.2)}{=} \sqrt{r^2 - \tilde{x}_{i+1}^2}, \tag{4.7}$$

et

$$(x_{i+.5}, y_{i+.5}) = (\lceil x_{i+1} \rceil, \lceil y_{i+1} \rceil). \tag{4.8}$$

Finalement,

$$\theta_{i+1} \overset{(4.2)}{=} \arccos(\tilde{x}_{i+1}/r). \tag{4.9}$$

Soit $8L$, le nombre d'intersections sur tout le cercle. On peut aussi trouver $x_{i+1}, y_{i+1}, x_{i+.5}, y_{i+.5}$ et $\theta_{i+1} - \theta_i$ pour certaines valeurs de i plus rapidement que par (4.3-9), à condition que l'origine soit sur un coin ou centre de pixel i.e. que

$$(a + .5, b + .5) \in \mathbb{Z}^2 \text{ ou } (a, b) \in \mathbb{Z}^2, \tag{4.10}.$$

En effet, toute grille cartésienne est symétrique par rapport à 4 axes passant par un coin ou par le centre de tout pixel (figure 9). Par conséquent, si (4.10) est respecté, alors

$$\left\{(\tilde{x}_{2i},\tilde{y}_{2i})\middle|2i=0\ldots16L-1\right\}=\left\{(\rho\tilde{x}_{2i},\lambda\tilde{y}_{2i}),(\rho\tilde{y}_{2i},\lambda\tilde{x}_{2i})\middle|\rho,\lambda=\pm1,2i=0\ldots2L-1\right\}, \qquad (4.11)$$

ce qui signifie que l'on peut trouver, par réflexion de tout point $2i|_{2i=0..2L-1}$ sur les 4 axes, les positions de 7 autres points, soit $(-\tilde{x}_{2i},\pm\tilde{y}_{2i}),(\pm\tilde{y}_{2i},\pm\tilde{x}_{2i})$ (figure 9). Ainsi, en assignant 0 à l'une ou l'autre des 4 intersections cercle-axes, cela implique que

$$F(n,q)\overset{(3.13)}{=}n\sum_{i=0}^{L-1}\sum_{\rho,\lambda=\pm1;(\tilde{u},\tilde{v})=(\rho\tilde{x},\lambda\tilde{y}),(\rho\tilde{y},\lambda\tilde{x})}\rho\lambda G_{i+.5}(u,v,q). \qquad (4.12)$$

On peut noter que l'équation (4.2) insérée dans (3.20) donne

$$G_{i+.5}(n,q)=f(x_{i+.5},y_{i+.5})\times\begin{cases}\theta_{i+1}-\theta_i & \text{si } q=0\\ \dfrac{(\tilde{x}_{i+1}+j\tilde{y}_{i+1})^q-(\tilde{x}_i+j\tilde{y}_i)^q}{jqr^q} & \text{sinon}\end{cases}.$$

Finalement, soit $$F_l(n,q)\equiv\begin{cases}0 & \text{si } l=0\\ n\sum_{i=0}^{l-1}\sum_{\rho,\lambda=\pm1;(\tilde{u},\tilde{v})=(\rho\tilde{x},\lambda\tilde{y}),(\rho\tilde{y},\lambda\tilde{x})}\rho\lambda G_{i+.5}(u,v,q) & \text{sinon}\end{cases},$$

alors $$F_{l+1}(n,q)=F_l(n,q)+\sum_{\rho,\lambda=\pm1;(\tilde{u},\tilde{v})=(\rho\tilde{x},\lambda\tilde{y}),(\rho\tilde{y},\lambda\tilde{x})}\rho\lambda G_{l-.5}(u,v,q), \qquad (4.13)$$

et alors $$F_L(n,q)=F(n,q). \qquad (4.14)$$

4.1.2 Algorithme

Le principe de l'algorithme est le suivant : pour $n=1\ldots N$, $q=0\ldots Q_{\max}$, $l=0\ldots L-1$, effectuer (4.13). Pour fixer $\tilde{x}_0,\tilde{y}_0,\tilde{x}_{L-1},\tilde{y}_{L-1}$, nous avons assigné 0 à l'intersection sur l'axe de symétrie vertical positif et $L-1$ à l'intersection sur l'axe de symétrie du deuxième quadrant (figure 9) i.e.

$$\begin{cases}\theta_0=\pi/2\\ \theta_{L-1}=2\pi/8+\theta_0=3\pi/4\end{cases}\leftrightarrow\begin{cases}(\tilde{x}_0,\tilde{y}_0)\overset{(4.2)}{=}(r\cos\theta_0,r\sin\theta_0)=(0,r)\\ (\tilde{x}_{L-1},\tilde{y}_{L-1})=(r\cos\theta_{L-1},r\sin\theta_{L-1})=\left(-r/\sqrt{2},r/\sqrt{2}\right)\end{cases}. \qquad (4.15)$$

Voici la liste de quelques variables de l'algorithme (figure 12) :

- r : αn ;
- $F_{ik}(n,q)$: $\begin{cases} \frac{F_l(n,0)}{n}, \frac{F_{l+1}(n,0)}{n}, F_{l+1}(n,0) & \text{si } q=0 \\ F_l(n,q)\frac{jqr^q}{n}, F_{l+1}(n,q)\frac{jqr^q}{n}, F_{l+1}(n,q) & \text{sinon} \end{cases}$;
- (x_p, y_p, θ_p) : $(x_{l+p}, y_{l+p}, \theta_{l+p})$;
- $(\tilde{x}_p, \tilde{y}_p)$: $(\tilde{x}_{l+p}, \tilde{y}_{l+p})$;
- (x, y) : (x_{l+5}, y_{l+5}).

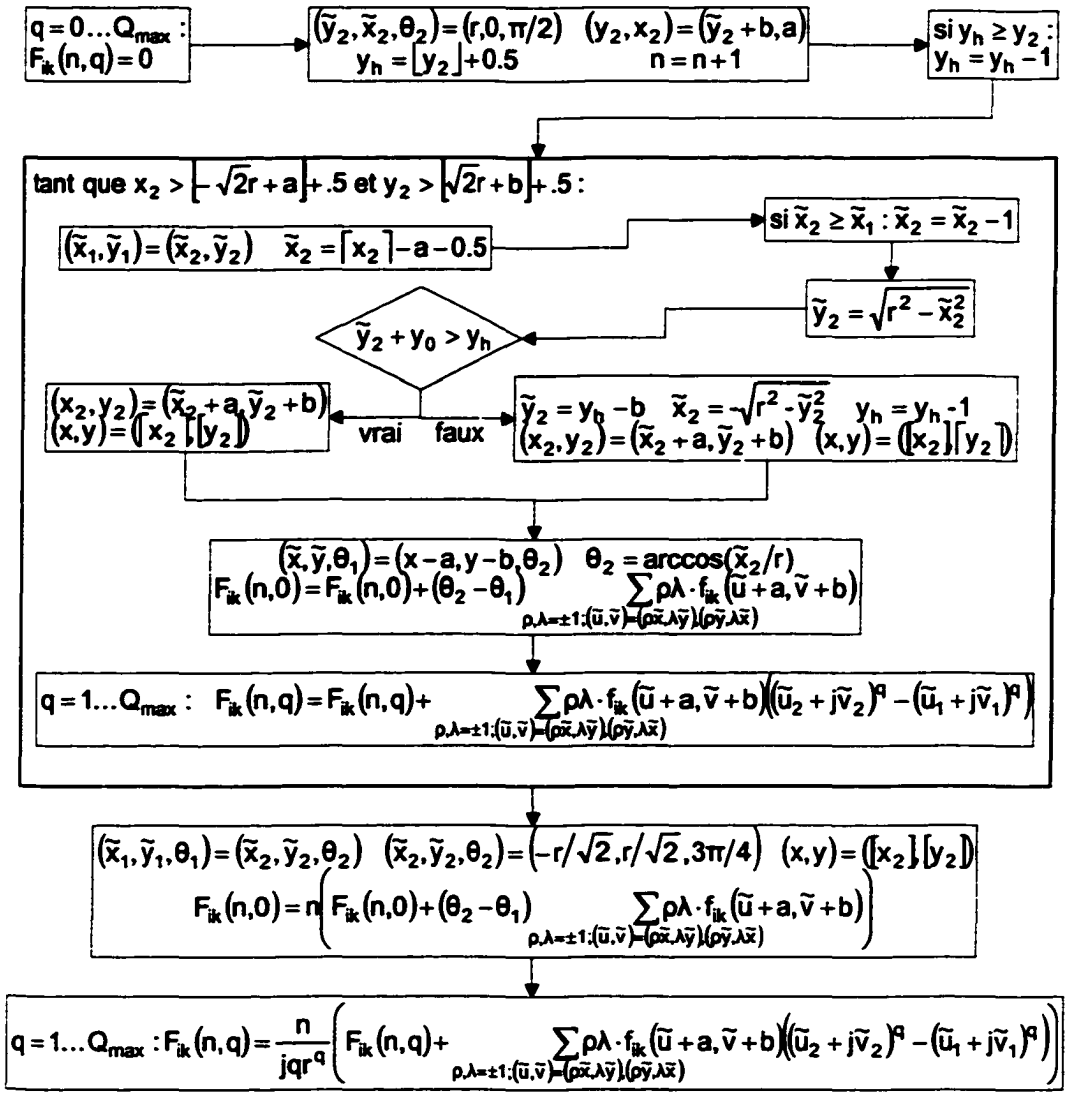


Figure 12 Algorithme de calcul de $[F_{ik}(n, \{0 \ldots Q_{max}\})]$.

4.2 Calcul de α, a, b

Précisons d'avance que f est binaire i.e.

$$f(x,y)=\begin{cases}1 & \text{si } (x,y)\in \text{caractère}\\ 0 & \text{sinon}\end{cases}.$$

Soit $[\cdot]$, l'entier le près de $\cdot$ et $\lfloor\cdot\rfloor$, l'entier inférieur le plus près de $\cdot$. Pour respecter la condition (4.11), l'origine est fixée sur un coin ou un centre de pixel, en fait le plus près du centroïde du caractère $(\bar{x},\bar{y})$ i.e.

$$[\bar{x}\quad \bar{y}]\equiv\frac{\sum_{x,y}[x\quad y]f(x,y)}{\sum_{x,y}f(x,y)} \tag{4.16}$$

$$a=0.5[2\bar{x}],\ b=0.5\lfloor 2\bar{y}\rfloor+a-\bar{y}-\lfloor a-0.5\lfloor 2\bar{y}\rfloor\rfloor. \tag{4.17}$$

L'échelle est normalisée par rapport à R, le rayon externe du caractère partant de (a,b) i.e. dans (1.21),

$$w=R\equiv\max_{x,y\in\mathbb{Z}}\sqrt{f(x-a,y-b)\left((x-a)^2+(y-b)^2\right)} \tag{4.18}$$

et donc d'après (1.21) :

$$\alpha=R/\tilde{R}. \tag{4.19}$$

Le principe de l'algorithme est alors le suivant : effectuer (4.16) à (4.19). Dans cet algorithme, la variable a représente $\bar{x}$, puis a. (figure 13).

$$(a\quad b)=\frac{\sum_{x,y}(x\quad y)f_{ik}(x,y)}{\sum_{x,y}f_{ik}(x,y)}$$

$$a=0.5[2a]\quad b=0.5\lfloor 2b\rfloor+a-b-\lfloor a-0.5\lfloor 2b\rfloor\rfloor$$

$$R=\sqrt{\max_{x,y}\left(f_{ik}(x-a,y-b)\left((x-a)^2+(y-b)^2\right)\right)}$$

$$\alpha=R/\tilde{R}$$

Figure 13 Algorithme de calcul de α, a, b.

4.3 Calcul de ε

Comme estimateur de $\varepsilon(p,Q)$, c'est l'estimateur *leave-one-out* (1.8) qui a été choisi avec, comme type de classifieur, *n*PPV (1.2) i.e. $g_k = g_{nPPV,k}$. En effet, lors du calcul du taux d'erreur *leave-one-out* ou de validation croisée, l'ensemble d'entraînement varie en contenu (§1.4). De plus, dans notre méthodologie, cet ensemble varie aussi en taille (§3.7). Chaque fois qu'il varie en contenu ou en taille, les paramètres varient aussi et doivent être réestimés à partir des valeurs de caractéristiques des exemples d'entraînement. Lorsque le classifieur est de type *n*PPV, on voit dans (1.2) et (1.3) que les paramètres de g_k sont ces valeurs de caractéristiques elles-mêmes, soit $x_j\big|_{x\in X, j\in E}$, ce qui permet d'économiser le temps de réestimation des paramètres.

Finalement, dans (1.2) il est clair que pour chaque instance de $(j_1,\ldots,j_n,j)$, le nombre de comparaisons "$d(\cdot)\le d(\cdot\cdot)$" à évaluer augmente avec n. Pour écourter au maximum le temps de cette évaluation, la valeur de n a été choisie minimale : $n=1$.

Soit (1.1), (1.3) et $$d_{jj'} \equiv d_j\big|_{x=x_{j'}} = \sum_{x\in X}\left(\mathrm{Re}\left|x_j - x_{j'}\right|\right)^r + \left(\mathrm{Im}\left|x_j - x_{j'}\right|\right)^r . \tag{4.20}$$

Dans le texte, $d_{jj'}(Q)$ désigne la valeur de $d_{jj'}$ lorsque $X = c(\mathbb{D}_0 \times \{0\ldots Q\})$ i.e.

$$d_{jj'}(Q) = \sum_{q=0}^{Q}\sum_{n=1}^{N}\left(\mathrm{Re}\left|c_j(n,q) - c_{j'}(n,q)\right|\right)^r + \left(\mathrm{Im}\left|c_j(n,q) - c_{j'}(n,q)\right|\right)^r . \tag{4.21}$$

4.3.1 Cadre théorique

4.3.1.1 Calcul de $d_{jj'}$

Nous avons vu (§1.6) que si $c(n)\big|_{\forall n\in B} = \langle f \,|\, g_n\rangle_A$ et si tous les vecteurs $[g_n(A)]\big|_{n\in B}$ sont orthonormaux entre eux, alors tout vecteur apparaît de même longueur dans $[f(A)]$ ou

dans $[c(B)]$, mais réorienté selon un angle de valeur et direction fixe. Par conséquent, la distance euclidienne entre deux points quelconques devrait apparaître comme la même dans $[f(A)]$ ou dans $[c(B)]$. Comme d'après (3.12), $c_0(n,\cdot)=F\left(d^{P-p}n,\cdot\right)$ et g,h ont été choisis orthonormaux (§3.7), elle devrait apparaître comme la même dans $c(\mathbb{D}_0\times\{0\ldots Q\})$ ou dans $\left[F\left(d^{P-p}\mathbb{D}_0,\cdot\right)\right]$. Ainsi, soit F_j, la valeur de F de l'exemple j et $d_{jj'}(N,Q)$, la valeur de d_j lorsque $X=\left\{F\left(d^{P-p}\mathbb{D}_0\times\{0\ldots Q\}\right)\right\}$, i.e.

$$d_{jj'}(N,Q)\equiv\sum_{q=0}^{Q}\sum_{n=1}^{N}\left(\mathrm{Re}\left|F_{j'}\left(d^{P-p}n,q\right)-F_j\left(d^{P-p}n,q\right)\right|^s+\left(\mathrm{Im}\left|F_{j'}\left(d^{P-p}n,q\right)-F_j\left(d^{P-p}n,q\right)\right|\right)^s,\quad(4.22)$$

alors si $s=2$, $\qquad d_{jj'}(Q)=d_{jj'}(N,Q)$.

On voit que le calcul de $d_{jj'}(Q)$ par (4.22) plutôt que (4.21) permet d'économiser le calcul de $c(n,q)$ pour $n=1\ldots N, q=Q_{op}+1\ldots Q$. Pour cette raison, s a été fixé à $s=2$.

Il est clair d'après (4.22) que la fonction $d_{jj'}$ possède les deux propriétés suivantes :

$$d_{jj'}(N,Q)=\left|\mathrm{Re}\left(F_j(N,Q)-F_{j'}(N,Q)\right)\right|^s+\left|\mathrm{Im}\left(F_j(N,Q)-F_{j'}(N,Q)\right)\right|^s+\begin{cases}d_{jj'}(N_{\max},Q-1)\text{si } N=1\\ d_{jj'}(N-1,Q)\text{sinon}\end{cases}\quad(4.23)$$

$$d_{jj'}=d_{j'j}\quad(4.24)$$

On voit que le calcul de $d_{jj'}(N,Q)$ par (4.23) ou (4.24) est plus rapide que par (4.22).

4.3.1.2 Calcul de ε

Comme $g_k=g_{1PPV,k}$ (§4.3), alors d'après (1.2), $g_k(\mathbf{x})=\left|\left\{j_{ki}\middle|d_{j_{ki}}<d_j,j_{ki}\neq j,j_{ki},j\in E\right\}\right|$ et il est donc clair qu'une et une seule classe k rend $g_k(\mathbf{x})$ égal à 1 alors que les autres classes le rend égal à 0 i.e. $g_k(\mathbf{x})=1\leftrightarrow g_l(\mathbf{x})\big|_{l\neq k}=0$, ce qui implique que la classe assignée à tout exemple est celle de son unique PPV i.e. selon (1.4),

$$\hat{k}(j)=k \text{ tel que } g_k(j,E\setminus j)=1.\quad(4.25)$$

Soit $E_0 \equiv \{j_{ki}\}_{k=1\ldots K, i=1\ldots I-1}$, $E_l \equiv E_{l-1} \cup j_{lI}$ et $PPV_l(i)$, le PPV de l'exemple i dans E_l et dans l'espace X i.e. $PPV_l(i) \equiv j$ tel que $d_{ij} < d_{ij'}, j \neq j'; j, j' \in E_l \setminus i$. (4.26)

Si E_{l-1} est l'ensemble courant, il est clair qu'un nouvel exemple j_{lI} lui étant ajouté est le seul pouvant détrôner les PPV courants i.e.

$$PPV_l(j_{ki}) = j \text{ tel que } d_{j_{ki}j} < d_{i_{ki}j'}, j \neq j'; j, j' \in \{PPV_{l-1}(j_{ki}), j_{lI}\}. \quad (4.27)$$

Dans l'équation (4.26), on voit que la comparaison $d(j_{ki}, j) < d(j_{ki}, j')$ serait évaluée pour chaque paire de valeurs dans $\{(j, j') | j, j' \in E_l \setminus j_{ki}, j \neq j'\}$; dans l'équation (4.27), elle l'est pour chaque paire de valeurs dans $\{(j, j') | j, j' \in \{PPV_{l-1}(j_{ki}), j_{lI}\}\}$. Comme $\{PPV_{l-1}(j_{ki}), j_{lI}\} \subset E_l \setminus j_{ki}$, le calcul de $PPV_l(j_{ki})$ est plus rapide avec (4.27) qu'avec (4.26).

Soit
$$\tilde{e}(k,i) \equiv \begin{cases} 0 & k = 1, i = 1 \\ \sum_{i_{ki}=1}^{i} \left| \left\{ j_{ki_{ki}} \middle| \hat{k}(j_{ki_{ki}}) \neq k \right\} \right| + \sum_{k_{ki}=1}^{k-1} \sum_{i_{ki}=1}^{I} \left| \left\{ j_{k_{ki}i_{ki}} \middle| \hat{k}(j_{k_{ki}i_{ki}}) \neq k_{ki} \right\} \right| & \text{sinon} \end{cases}. \quad (4.28)$$

alors
$$\tilde{e}(k,i) \overset{(4.28)}{=} \left| \left\{ j_{ki} \middle| \hat{k}(j_{ki}) \neq k \right\} \right| + \sum_{i_{ki}=1}^{i-1} \left| \left\{ j_{ki_{ki}} \middle| \hat{k}(j_{ki_{ki}}) \neq k \right\} \right| + \sum_{k_{ki}=1}^{k-1} \sum_{i_{ki}=1}^{I} \left| \left\{ j_{k_{ki}i_{ki}} \middle| \hat{k}(j_{k_{ki}i_{ki}}) \neq k_{ki} \right\} \right|$$

$$\overset{(4.28)}{=} \begin{cases} 1 \text{ si } \hat{k}(j_{ki}) \neq k \\ 0 \text{ sinon} \end{cases} + \begin{cases} \tilde{e}(k-1, I) \text{ si } i = 1 \\ \tilde{e}(k, i-1) \text{ sinon} \end{cases}. \quad (4.29)$$

et
$$\hat{e} \overset{(1.8)}{=} \frac{\tilde{e}(K, I)}{KI}. \quad (4.30)$$

4.3.2 Algorithme

Soit I_{max}, une valeur à choisir. Le principe de l'algorithme est le suivant. Pour $p = 0 \ldots P$, $Q = 0 \ldots Q_{max}$: (1) pour $n = 1 \ldots d^p, j, j' \in \{j_{kI_{max}}\}_{k=1\ldots K, j>j'}$ effectuer (4.23) puis (4.24) ; pour $I = 1 \ldots I_{max}, k = 1 \ldots K$: (2) pour $i = 1 \ldots I-1$, effectuer (4.27), (4.29) puis (4.30). Pour $i = I$, effectuer (4.26) , (4.29) puis (4.30) ; (3) effectuer (4.30), puis $\varepsilon(p, Q, I) = \hat{e}$.

Pour fixer PPV_0, on peut assumer que le PPV de chaque exemple $j_{kl}|_{k=1..K}$ est n'importe quel autre exemple plus ancien, donc de l'ensemble $\mathrm{E}_K = \{j_{kl}\}_{k=1..K,\, l=1..I}$, i.e.

$$\mathrm{PPV}_0(j_{kl}) \in \mathrm{E}_K \setminus j_{kl}\,.$$

Par exemple, on a assumé que

$$\mathrm{PPV}_0(j_{1,\cdot}) = j_{2,1}\,.\mathrm{PPV}_0(j_{k,\cdot})\big|_{k\neq 1} = j_{1,1}\,. \tag{4.31}$$

L'indice du premier exemple de chaque classe $k+1$ a été fixé pour suivre l'indice du dernier exemple de la classe k i.e. $j_{k+1,1} \equiv j_{kI_{\max}} + 1$, (4.32)

ce qui implique que $j_{ki} = kI_{\max} + i \rightarrow k = \lfloor (j_{ki} - 1)/I_{\max} \rfloor + 1$. (4.33)

En effet, $j_{k+1,1} \overset{(4.33)}{=} (k+1)I_{\max} + 1 = kI_{\max} + I_{\max} + 1 \overset{(4.33)}{=} j_{kI_{\max}} + 1$,

ce qui est la définition (4.32).

Voici la liste de quelques variables de l'algorithme :

- d : $d_{j_{\mathrm{tr}} j_{\mathrm{te}}}(n, Q)$;
- $(j_{\mathrm{tr}}, j_{\mathrm{te}})$: $(j_{k_{\mathrm{tr}} i_{\mathrm{tr}}}, j_{k_{\mathrm{te}} i_{\mathrm{te}}})$;
- PPV : $\mathrm{PPV}_{k_{\mathrm{te}}}$;
- $\hat{\mathrm{k}}$: $\hat{k}(j_{\mathrm{te}})$;
- $\varepsilon(\mathrm{p,q,I})$: $e(k_{\mathrm{te}}-1, i_{\mathrm{te}}-1), e(k_{\mathrm{te}}-1, i_{\mathrm{te}}), e(k_{\mathrm{te}}, i_{\mathrm{te}}-1), e(k_{\mathrm{te}}, i_{\mathrm{te}})$ pour $\mathrm{X} = c(\{1\ldots n\} \times \{0\ldots Q\})$

Dans cette liste, les indices "tr" et "te" signifient "entraînement" et "test" respectivement.

4.4 LDB

Dans la LDB, qui inclut une suite de comparaisons $D(\{c_{2i+1}(\mathbf{D}_{2i+1}), c_{2i+2}(\mathbf{D}_{2i+2})\}) > D(\{c_i(\mathbf{D}_i)\})$ à effectuer (§2.7), les termes c_{2i+1} et c_{2i+2} doivent être calculés par (2.32). L'opération (2.32), ainsi qu'une méthode identique à celui de LDB mais utilisant une autre fonction J que D, sont déjà implantés dans deux routines du logiciel *Matlab* : *wpdec.m* et *besttree.m*. Dans la routine *wpdec.m*, les opérations (2.32) et le calcul de J sont effectués sur un seul exemple de signal à la fois. Or, le calcul de D (1.9) doit être

effectué sur plusieurs exemples de classes. Pour obtenir la méthode LDB à partir de *wpdec.m* et *bestree.m*, on remplace donc dans *wpdec.m*, l'estimation de $J(c_i(\mathbb{D}_i))$ sur un seul exemple par celle de $D(c_i(\mathbb{D}_i))$ sur l'ensemble d'exemples $\{j_{ki}\}_{k=1\ldots K, i=1\ldots I_y}$.

4.5 Sélections de caractéristiques et estimation de $e(Y_i)$

Dans ces deux étapes, J et $e(Y_i)$ sont estimés sur l'ensemble d'exemples $\{j_{ki}\}_{k=1\ldots K, i=1\ldots I_y}$. Deux sélections sont effectuées : une SI avec $J = J_{DKL}$ et une SDS avec $J = \hat{e}$ (§1.4, §1.5) et comme estimateur de e, le taux d'erreur *leave-one-out* sur classifieur 1PPV. L'algorithme de cette deuxième sélection est déjà implanté dans la routine *featSelectMinErr_SFS* du logiciel *Tooldiag*. Dans cette routine, l'équation (1.11) est effectuée pour $i = 0\ldots|X|-1$.

4.6 Délimitation du choix de d, p

Les paramètres d, p peuvent être limités à certaines valeurs permettant d'économiser des calculs. En effet, soit $\mathbb{D}_0(p) \equiv \{1\ldots d^p\} = \mathbb{D}_0$ (3.11), alors pour $d, p \in \mathbb{Z}_+^*$, il est clair que $\mathbb{D}_0(p) = \{1\ldots d^p\} \subset \{1\ldots d^p \ldots d^{p+1}\} = \mathbb{D}_0(p+1)$, ce qui signifie que si $c_0(\mathbb{D}_0(p), \{\cdot\})$ est connu, on peut obtenir $c_0(\mathbb{D}_0(p+1), \{\cdot\})$ en ne calculant que $c_0(\{d^p+1\ldots d^{p+1}\}, \{\cdot\})$ et en l'ajoutant à $c_0(\mathbb{D}_0(p), \{\cdot\})$, ce qui est plus rapide que de le calculer au complet. Pour cette raison, d, p ont été choisis dans $\mathbb{Z}_+^*$.

4.7 Choix de $\tilde{g}, \tilde{h}$

Soit $[\tilde{c}_{2i+1}(n)\ \ \tilde{c}_{2i+2}(n)] \equiv \sum_k c_i(n-k)[\tilde{g}(k)\ \ \tilde{h}(k)]$, alors d'après (2.32), $[c_{2i+1}(n)\ \ c_{2i+2}(n)] = [\tilde{c}_{2i+1}(2n)\ \ \tilde{c}_{2i+2}(2n)]$ (décimation). Soit $\tilde{\mathbb{D}}_i$, l'ensemble tel que le signal $[c_i(\tilde{\mathbb{D}}_i)]$ dépend entièrement et au moins d'un échantillon du signal $[c_0(\mathbb{D}_0)]$ et $\ell(\cdot)$, la

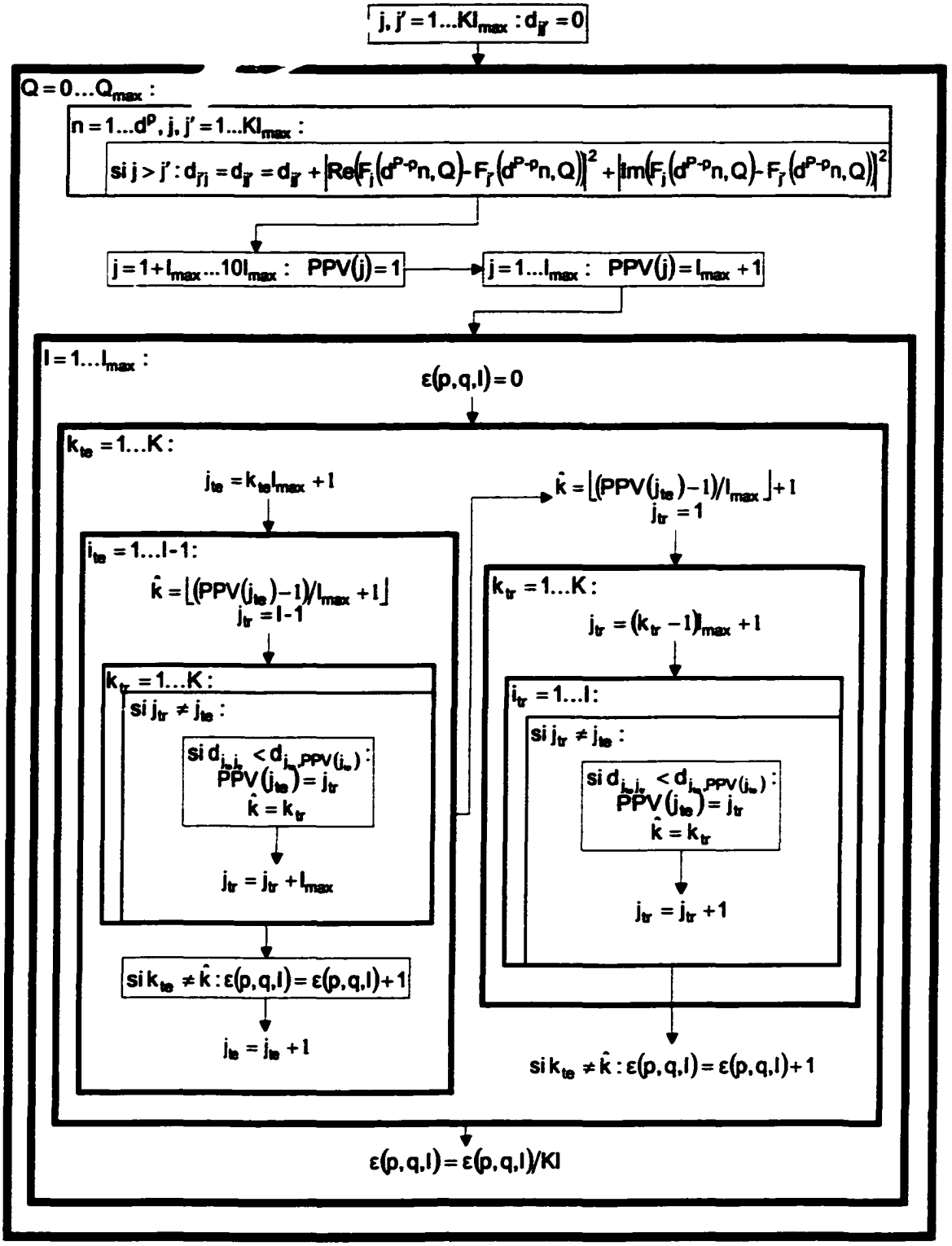


Figure 14 Algorithme de calcul de $[\varepsilon(p, Q, l)]_{Q=0...Q_{max}, l=1...l_{max}}$

longueur du support de la fonction $\cdot\cdot$. Il est clair (figure 15) que

$$\left|\mathbb{D}_{2i+1}\right| = \ell(\tilde{g}) + \left|\mathbb{D}_i\right| - 1 \tag{4.34}$$

et par analogie, que $\left|\mathbb{D}_{2i+2}\right| = \ell(\tilde{h}) + \left|\mathbb{D}_i\right| - 1$.

À cause de cela, il est préférable que les filtres $\tilde{g}$ et $\tilde{h}$ soient le plus courts possible, pour économiser le plus d'opérations (2.32) possible et réduire la taille de l'information sur c_i contenu dans c_{2i+1} et c_{2i+2}.

Pour garder le signal c le plus court possible, les filtres $\tilde{g}$ et $\tilde{h}$ ont été choisis de longueur minimale possible i.e. 2. Les filtres de l'ondelette de *Haar* (2.30,2.31) sont effectivement de longueur 2 et orthonormaux (§2.5.1) et ont donc été choisis.

$c_i(n)$:				+	···	+		
$\tilde{g}(n)$:	+	···	+					
$\bar{g}(n)$:						+	···	+
$c_{i+1}(n)$:				+	···	···	···	+
n:	1	···	ℓ	···	···	···		L

Figure 15 Filtrage et décimation du signal c_i

+ : échantillons de bord ;
n : instant de la nième colonne d'échantillons ;
ℓ : $\ell(g)$; $\bar{g}(n) : \tilde{g}(n - |\mathbb{D}_i|)$;
1 : échantillon du bord gauche de $\bar{g}$;
L : échantillon du bord droit de $\bar{g}$.

4.8 Choix de $\tilde{R}$

Comme d'après (3.11) et (3.1), $d^P = \max(\mathbb{D}_0)$ et $c_0(d^P, \cdot) = d^P \int_0^{2\pi} \tilde{f}_p(d^P, \theta) d\theta$, alors le caractère sur $\tilde{f}$ doit être suffisamment petit pour être englobé par le cercle de rayon d^P et de centre (0,0). Par contre ce caractère, de rayon externe $\tilde{R}$, ne devrait pas être plus

petit, car en-dehors du cercle de rayon $\tilde{R}$, les points de l'image ne contiennent bien sûr aucune information sur le caractère. Cela implique donc que $\tilde{R} = d^p$.

4.9 Choix de d

Tout décimateur (figure 5a) rétrécit le signal d'entrée. Lorsqu'un signal de longueur impaire est décimé d'un facteur 2 à partir de l'échantillon de bord, il est clair que le signal de sortie du décimateur a un échantillon de moins que s'il est décimé à partir de l'échantillon suivant. À cause de cela, il est avantageux que le décimateur soit réglé pour décimer à partir du deuxième échantillon, ce qui est le cas avec la fonction *wpdec.m*, et

alors $$|\mathbb{D}_{2i+1}| = \left\lfloor \frac{|\tilde{\mathbb{D}}_{2i+1}|}{2} \right\rfloor = \left\lfloor \frac{\ell(g)+|\mathbb{D}_i|+1}{2} \right\rfloor = \left\lceil \frac{\ell(g)+|\mathbb{D}_i|}{2} \right\rceil .$$

Dans le cas de l'ondelette choisie i.e. celle de Haar, comme $\ell(\tilde{g}) = \ell(\tilde{h}) = 2$ (2.30,2.31),

alors $$|\mathbb{D}_{2i+2}| = |\mathbb{D}_{2i+2}| = \left\lceil \frac{|\mathbb{D}_i|}{2} \right\rceil = \begin{cases} |\mathbb{D}_i|/2 & \text{si } |\mathbb{D}_i| \text{ pair} \\ 1+|\mathbb{D}_i|/2 & \text{si } |\mathbb{D}_i| \text{ impair} \end{cases}.$$

D'après l'équation précédente, lorsque $|\mathbb{D}_i|$ est pair (divisible par 2), les signaux c_{2i+1}, c_{2i+2} perdent un échantillon de plus que lorsque $|\mathbb{D}_i|$ est impair. À cause de cela, et comme d'après (3.11), $|\mathbb{D}_0| = |\{1 \ldots d^p\}| = d^p$, et d'après §4.6, $d, p \in \mathbb{Z}_+^*$, alors d a été choisi pour que $|\mathbb{D}_0|$ soit divisible par deux le plus de fois possibles : $d = 2$.

4.10 Application de la méthodologie

Nous avons implanté chaque étape de la méthodologie (figure 11) dans un programme écrit en langage *c* ou *Matlab* (annexes), tout en y fixant certains paramètres (§4.6 à §4.9). Pour appliquer cette méthodologie, on exécute donc les programmes. Ainsi, nous avons appliqué cette méthodologie pour évaluer des ensembles de MPO et de leurs invariants sur l'ensemble de classes que sont les dix chiffres arabes i.e. {'0'...'9'}. Les indices k de ces 10 classes ont été fixés à $k = 1 \ldots K$ respectivement, et alors $K = 10$.

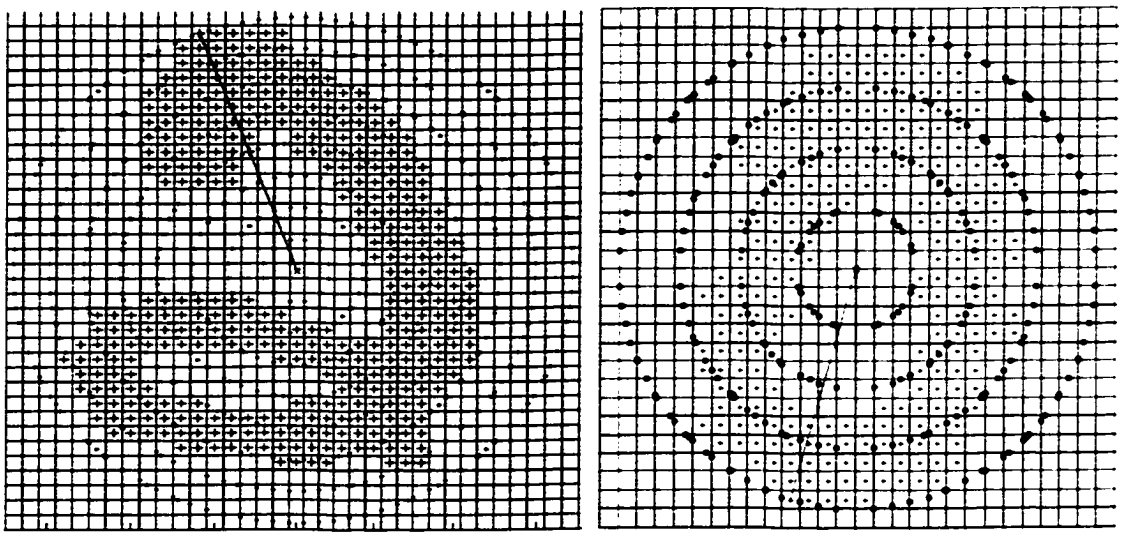
Nous avons extrait tous nos exemples de classes dans la base de données NIST, qui contient des fichiers d'images binaires échantillonnées sur une grille cartésienne de 300 points par pouce. Ces images sont celles de caractères manuscrits écrits dans la même direction. Chaque fichier contient des exemples de même classe écrits par différents scripteurs et est contenu dans un répertoire dont le nom est la valeur ASCII de la classe. Ainsi, nos exemples sont les I_{max} premiers dans chacun des fichiers *30/hsf_0.mis* à *39/hsf_0.mis*. Chacun de ces fichiers contient entre 4603 et 6008 exemples. Les résultats de la méthodologie (figure 16, 17, 20) sont référencés sur la figure 11.

4.10.1 Résultats du programme de calcul de F_{ik}

Le résultat 1 (figure 16) sert à vérifier la fonctionnalité du programme de calcul de F_{ik} (annexe 1). Sur la figure 16a et b respectivement, on reconnaît à l'œil, dans la région formée par les '+' un caractère de la classe '0' et '2', car $k-1=0$ et $k-1=2$. On compte respectivement 4 et 8 cercles de centre '×' et le plus grand englobe tout le caractère en y touchant que par le pixel le plus éloigné ; ce pixel est celui à l'extrémité du segment de droite, de longueur R, car par définition R est le rayon du plus petit cercle centré sur (a,b) et contenant tout le caractère. Les '·' sont sur toutes et uniquement les intersections grille-cercles, car par définition (x_2,y_2) est la coordonnée d'une de ces intersections. Le programme de l'annexe 1 fonctionne donc tel que prévu.

4.10.2 Résultats suivants

Dans notre application, nous avons fixé le paramètre P à 5, et Q_{max} à 24. Sur la figure 17, $\varepsilon(p,Q,1)=100\%$ pour toute les courbes. Cela est trivial, car lors d'une estimation *leave-one-out* avec classifieur 1NN, au moment où le seul exemple d'une classe devient l'exemple de test, son PPV ne peut être qu'un exemple d'une autre classe.



(a) (b)

Figure 16 Résultat 1 avec le programme de l'annexe 1
'+' : positions (x,y) où $f_{ik}(x,y)=1$; ailleurs : $f_{ik}(x,y)=0$;
'x' : position (a,b) ; '·' : positions (x_2,y_2) ;
droite partant du 'x' : longueur R ;
(a) $P=3$, $k=3$ et $i=6$;
(b) $P=2$, $k=1$ et $i=475$.

Sur la figure 18, on voit que les courbes deviennent plus ou moins constantes, surtout dans la zone à droite de $I=300$. Dans cette zone, la courbe qui semble être en moyenne plus basse que les autres est celle formée de '∘' (figure 19). Par conséquent, nous avons assumé que $p_{op}=3, Q_{op}=10$. Sur cette courbe et dans la majeure partie inférieure de cette zone $(300<I\leq 456)$, le point le plus bas est à $I=351$. Cela peut signifier que les exemples ajoutés entre le 352ème et le 455ème sont mauvais i.e. qu'ils nuisent à la reconnaissance de certains des 351 exemples précédents lors des estimations *leave-one-out*. Par conséquent, nous ne les avons pas ajoutés et avons alors assumé que $I_{op}=351$ (tableau II).

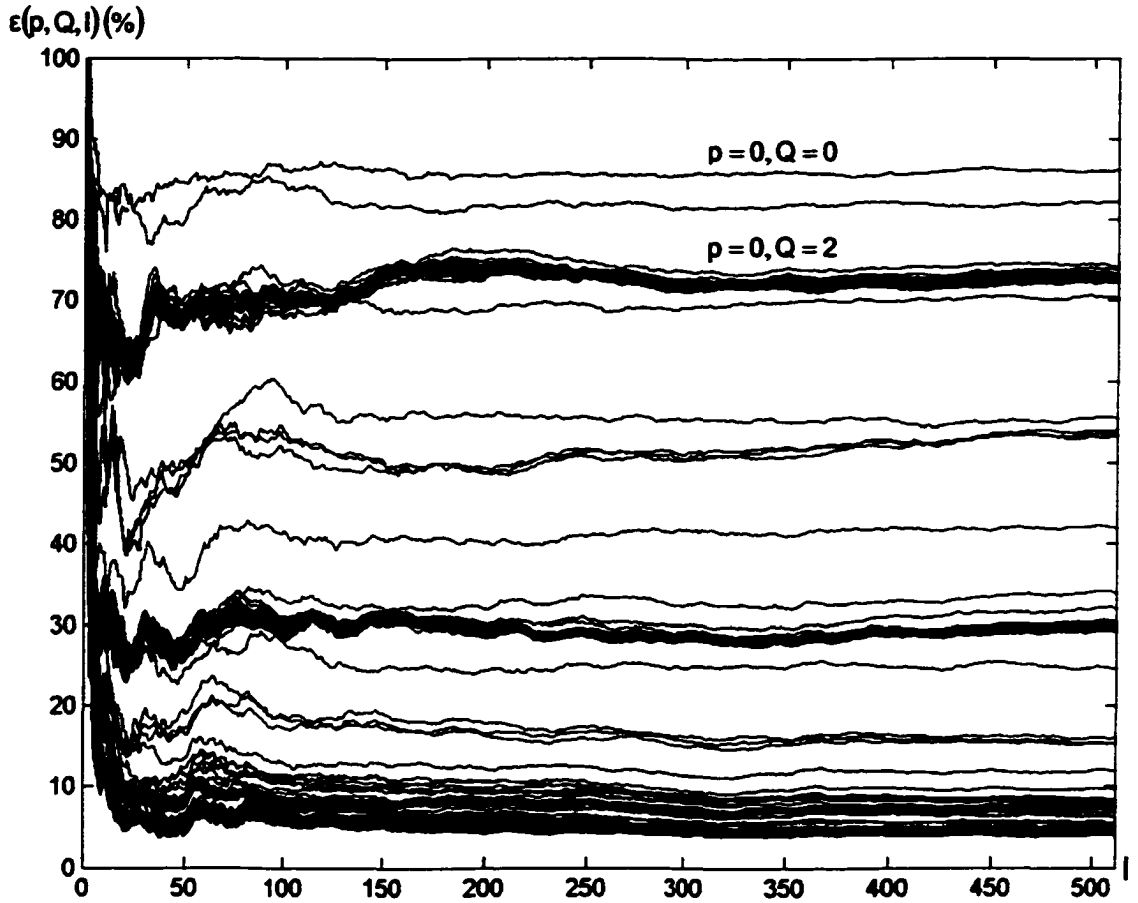


Figure 17 Résultat 2 ($\varepsilon(p,Q,l)$: taux d'erreur estimé sur l exemples par classe)

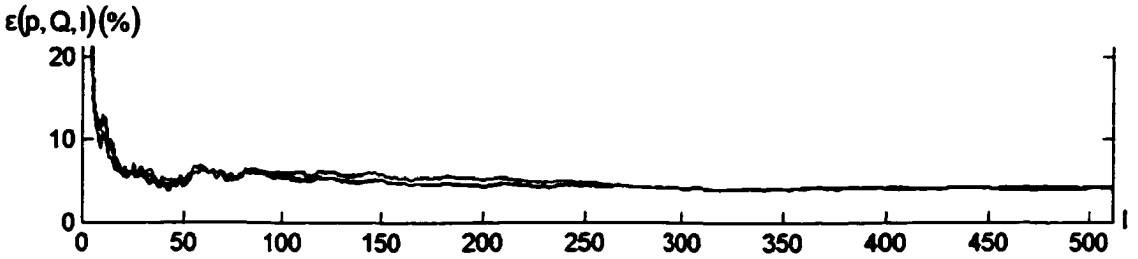


Figure 18 Quatre courbes les plus basses de la figure 17

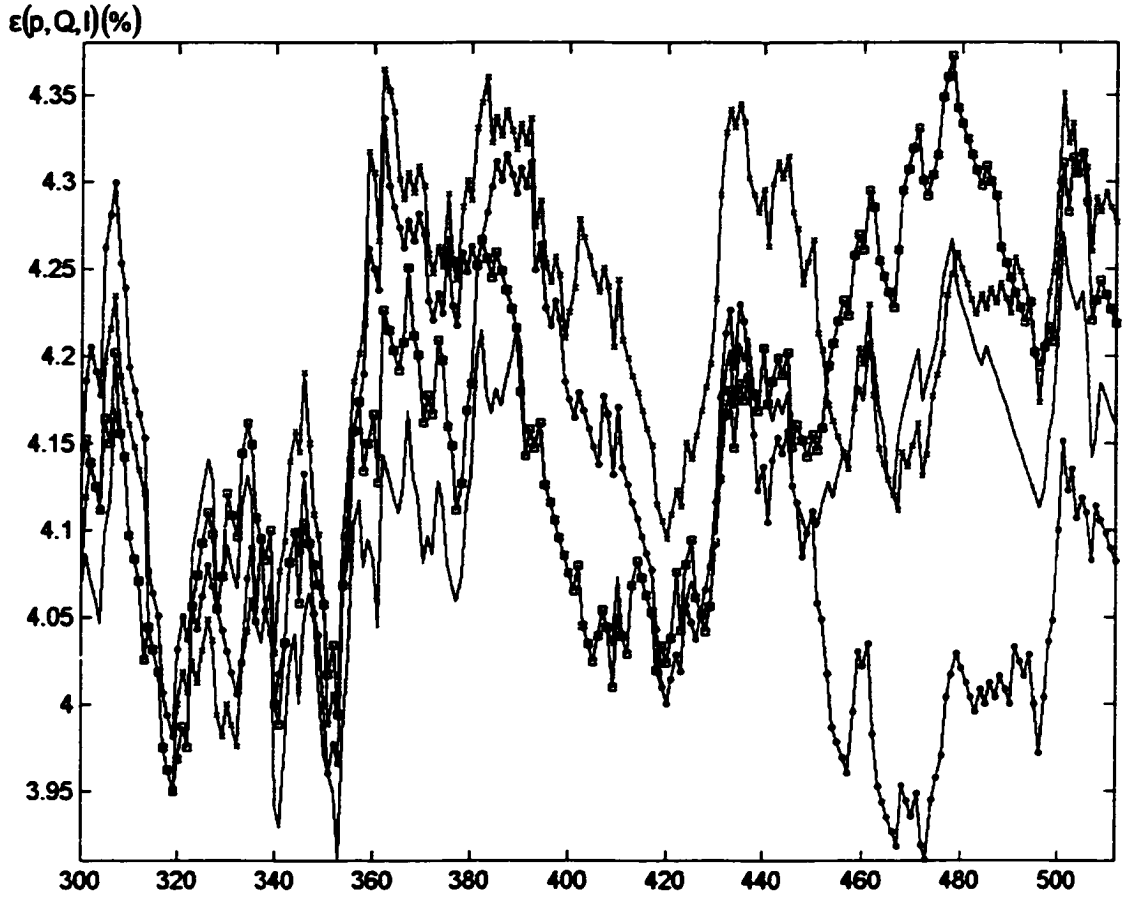


Figure 19 Portion agrandie de la figure 18.

'.' : $(p,Q)=(4,9)$;
'x' : $(p,Q)=(4,10)$;
'-' : $(p,Q)=(3,9)$;
'o' : $(p,Q)=(3,10)$.

Tableau II

Résultat 3

p_{op}	Q_{op}	I_{op}
3	10	351

Dans notre application, nous avons effectué une estimation de $e(Y_i)$ pour chaque ensemble Y et pour $i = 1...|Y|$ et avec comme estimateur de e, le taux d'erreur *leave-one-out* avec classifieur 1PPV à distance euclidienne. Sur la figure 20, on voit que la courbe la plus basse est la courbe *a* et qu'elle cesse relativement de décroître autour de $i \approx 40$, où

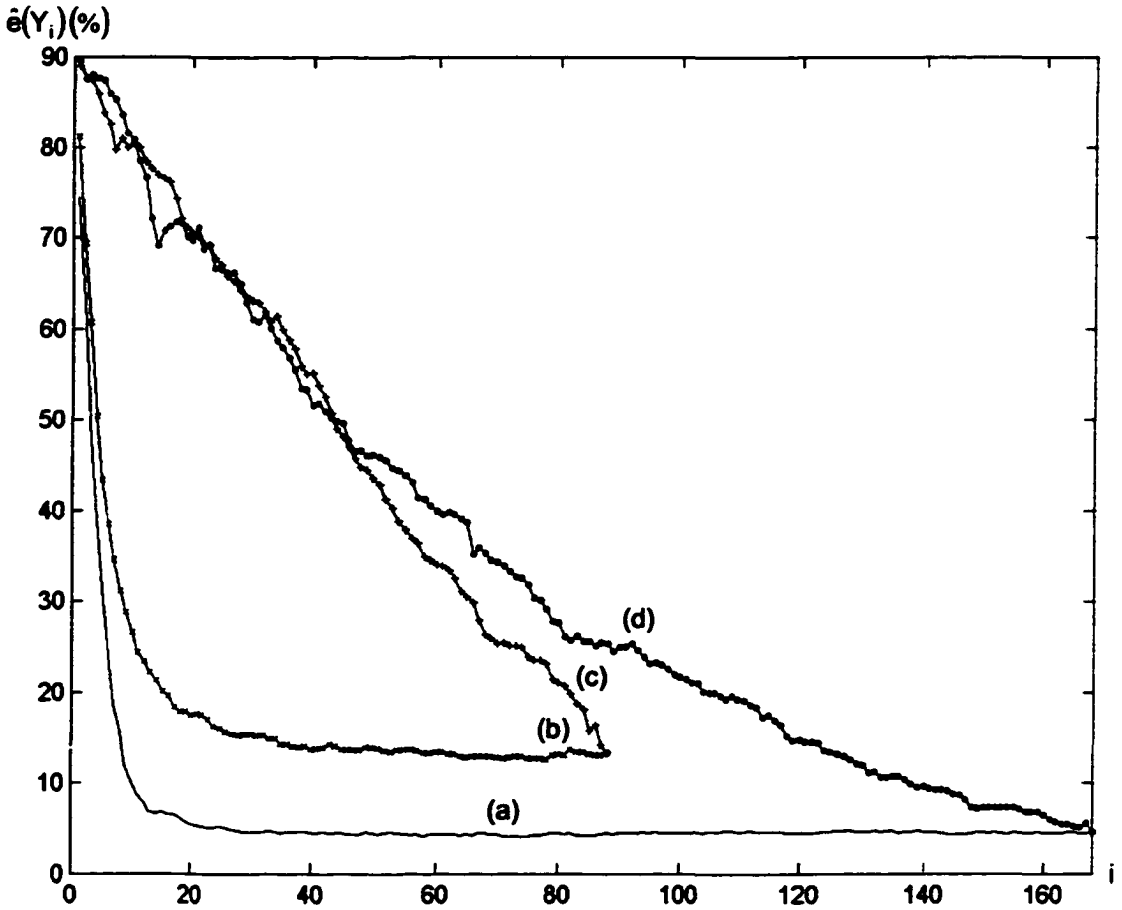


Figure 20 Résultat final avec classifieur 1PPV

Y_i : ensemble de i caractéristiques sélectionné dans Y ;

$\hat{e}(Y_i)$: taux d'erreur de Y_i estimé sur I_{op} exemples ;

(a) $Y = \{c(\mathbb{D},0), \mathrm{Re}\, c(\mathbb{D},q), \mathrm{Im}\, c(\mathbb{D},q)\}_{q=1}^{10}$, $J = \hat{e}$, sélection : SDS ;

(b) $Y = \{c(\mathbb{D},q)\}_{q=0}^{10}$, $J = \hat{e}$, sélection : SDS ;

(c) $Y = \{c(\mathbb{D},q)\}_{q=0}^{10}$, $J = D$, sélection : SI ;

(d) $Y = \{c(\mathbb{D},0), \mathrm{Re}\, c(\mathbb{D},q), \mathrm{Im}\, c(\mathbb{D},q)\}_{q=1}^{10}$, $J = D$, sélection : SI.

$\hat{e}(Y_i) \approx 4.5\%$. On voit aussi que la courbe *b* est plus basse sur toute sa longueur que la courbe *a* i.e. que pour $J = \hat{e}$, $\hat{e}(Y_i)$ est moins élevé pour $Y = \{c(\mathbb{D},0), \mathrm{Re}\, c(\mathbb{D},q), \mathrm{Im}\, c(\mathbb{D},q)\}_{q=1}^{10}$ que pour $Y = \{c(\mathbb{D},q)\}_{q=0}^{10}$.

Nous avons effectué une autre estimation de $e(Y_i)$ pour $i = 39$ et pour les paramètres de la courbe *a* et avec, comme estimateur de e, le taux d'erreur *hold-out* [9] sur classifieur SVM [13]. Durant cette estimation, 2/3 des I_{op} premiers exemples de chaque classe précédents ont servi d'exemples d'entraînement et les autres ($351 \cdot 1/3$), d'exemples de test. Nous avons obtenu $\hat{e}(Y_{39}) = 2.9915\%$ (tableau III).

Tableau III

Taux $\hat{e}(Y_{39})$ pour deux classifieurs différents

Classifieur	1NN	SVM
Estimateur	Leave-one-out	Hold-out
$\hat{e}(Y_{39})$	4.56%	2.99%

DISCUSSION ET INTERPRÉTATION DES RÉSULTATS

Le résultat final (figure 20) signifie que, dans un système de reconnaissance de chiffres manuscrits écrits dans la même direction, avec classifieur 1PPV à distance euclidienne, et avec, comme ensemble de caractéristiques, un des ensembles Y_i évalué dans l'application précédente, il serait plus économique que Y_i soit un de ceux sélectionnés par SDS avec $J = \hat{e}$ et tel que $i \leq 40$. On a vu aussi que, dans un cas (le cas $J = \hat{e}$), les ensembles de MPO ($c(\cdot)$) ont donné un taux d'erreur de classification moindre que ceux à tailles égales des invariants ($|c(\cdot)|$). En effet, les IMPO étant invariants à la rotation et à la réflexion (§3.6), cela rend probablement certains caractères plus difficiles à distinguer entre eux, comme le '6' avec le '9' et le '2' avec le '5', car les deux de chaque paire ne diffèrent entre eux en grande partie que par une rotation ou une réflexion.

Maintenant, essayons d'identifier les causes des erreurs de classification. Dans l'introduction, nous avons mentionné que les exemples de caractères manuscrits de même classe peuvent varier entre eux par leur inclinaison, leur taille et la largeur de leur tracé, ce qui peut modifier leurs caractéristiques. Dans la méthodologie, l'échelle est normalisée par rapport à R (4.19), ce qui rend les exemples de taille identique. Or, la normalisation de l'échelle modifie la largeur du tracé. Il est donc impossible que deux exemples de même largeur de tracé mais de taille différente deviennent identiques même après normalisation d'échelle. Une des causes d'erreurs peut donc être la taille et la largeur du tracé. Dans la méthodologie, lors du calcul de $[\varepsilon(\cdot,\cdots,I)]_{I=1\ldots I_{\max}}$ (§4.3), les I exemples de chaque classe sur lesquels est calculé $\varepsilon(\cdot,\cdots,I)$ sont inclus dans les $I+1$ exemples sur lesquels est calculé $\varepsilon(\cdot,\cdots,I+1)$. Comme l'estimateur choisi de $\varepsilon(\cdot,\cdots)$ est le *leave-one-out*, il est clair que si $\varepsilon(\cdot,\cdots,I+1) < \varepsilon(\cdot,\cdots,I)$, on ne peut savoir si c'est parce que les exemples de chaque classe ajoutés aux I autres exemples sont bons comme exemples de test ou d'entraînement i.e. s'ils sont bien reconnus ou s'ils contribuent à la reconnaissance des I autres exemples précédents. Une autre cause d'erreurs peut donc être la présence d'exemples mauvais comme exemples d'entraînement, mais qui immédiatement après avoir été ajoutés, ont fait baisser le taux d'erreur de reconnaissance parce qu'ils sont de bons exemples de test.

CONCLUSION

Notre objectif était d'évaluer des ensembles de IMO modifiés et de paramètres réglés de façon à contribuer à la résolution de quatre points (§3.6). Nous avons effectivement apporté une solution pour chacun (§3.7). L'évaluation sur classifieur 1PPV a donné comme résultat final la figure 20, qui inclut $\hat{e}(Y_{39}) = 4.56\%$ (tableau III) et l'évaluation sur classifieur SVM a donné comme résultat $\hat{e}(Y_{39}) = 2.9915\%$ (tableau III) pour les paramètres de la courbe 20a. Pour cela, rappelons que la méthodologie (figure 11) a été effectuée avec les valeurs suivantes de paramètres :

- fréquence d'échantillonnage : 300 points par pouce (§4.10) ;
- normalisation de l'origine : par rapport au coin/centre le plus près de $(\bar{x}, \bar{y})$ (4.18) ;
- normalisation de l'échelle : par rapport à R (4.19) ;
- estimateur de $\varepsilon(p, Q)$: *leave-one-out* avec 1PPV à distance euclidienne (§4.3) ;
- sélections : SI avec $J = D$ et SDS avec $J = \hat{e}$ (§4.5) ;
- $\tilde{g}, \tilde{h}$: filtres de l'ondelette de Haar (§4.7) ;
- $\tilde{R}$: d^P (§4.8) ;
- d : 2 (§4.9) ;
- ensemble de classes : {'0'...'9'} (§4.10) ;
- (P, Q_{max}) : (5,24) (§4.10.2) ;
- estimateur de $e(Y_i)$: *leave-one-out* (§4.10.2) ;
- classifieurs : 1PPV à distance euclidienne, SVM (§4.10.2).

Dans cette liste, les classifieurs, comme ceux inclus par la notion de *classifieur* telle que définie dans §1.2, sont de type à un seul *niveau* [9]. Or, il existe aussi le type à *arbre* [9]. Ainsi, pour avoir des chances d'améliorer le résultat i.e. de diminuer la hauteur des courbes de la figure 20, on peut réeffectuer la méthodologie mais en y apportant les modifications suivantes :

- estimateur de $e(Y_i)$: avec classifieur n PPV à valeurs de n supérieures ;
- $\tilde{g}, \tilde{h}$: filtres d'une fonction ψ de produit $\sigma_r \sigma_\omega$ inférieur (§2.1) ;
- ajout, avant l'étape de normalisation d'origine et d'échelle, d'une étape de normalisation d'inclinaison et après, d'une étape de normalisation d'épaisseur ou

d'amincissement du tracé. Cela rendrait les caractéristiques invariantes à l'inclinaison et à l'épaisseur du tracé ;

- remplacement de l'étape d'estimation de I par une étape de sélection des exemples d'entraînement. En effet, dans une région de l'espace X délimitée par un nuage d'exemples de même classe, seuls les exemples à la surface du nuage peuvent influencer la classe assignée par un classifieur nPPV . L'étape de sélection consisterait donc à ne conserver comme exemples d'entraînement que ceux qui sont à la surface des nuages d'exemples de même classe ;
- remplacement, dans l'étape d'évaluation des MPO, de l'arbre du classifieur à un seul niveau par un arbre à deux niveaux : un premier niveau pour la discrimination entre les deux ensembles de classe {'0','1','3','4','7','8'} et {'2','5','6','9'} avec des IMPO, et un deuxième niveau pour la discrimination dans {'2','5','6','9'} avec des MPO et dans {'0','1','3','4','7','8'} avec des IMPO. Les MPO n'étant pas invariants à la rotation, cela ferait peut-être diminuer le taux de confusion entre '2' et '5' et entre '6' et '9'.

Finalement, nous avons mentionné dans l'introduction que les transformées permettent la possibilité de calculer des caractéristiques peu sensibles au bruit. Les MPO et leurs invariants étant des transformées, il serait donc intéressant de réeffectuer la méthodologie sur des exemples de classe bruités. Aussi, les MPO n'étant pas invariants à l'orientation, il serait intéressant de réeffectuer la méthodologie sur des exemples de classe d'orientation aléatoire ; ainsi, les ensembles des invariants donneraient peut-être un taux d'erreur de classification moindre que ceux à tailles égales des MPO.

ANNEXE 1

Programme de calcul de a, a, b et $[F_{ik}(n, \{0 \ldots Q_{max}\})]$

L'algorithme de première étape d'extraction (figure 10) est écrit en langage *Matlab*:

```
clear all;close all;
F=[];save Fk F;

Q=1:24;
N=32;

for k=0:9
    Fk=[];
    unix(['showmis /home/bd/nist_cd/data/by_class/3' int2str(k)
'/hsf_0'
'.mis']);
    fid=fopen('image','rb');

    for ii=1:512
        f=[(fread(fid,[128,128],'ubit1'))]';
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        %debut de l'algorithme de la figure 10%
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        Sx=sum(f,1);
SSxy=sum(Sx);a=[1:128]*Sx'/SSxy;b=[1:128]*sum(f,2)/SSxy;
        a=round(2*a)/2;b=floor(2*b)/2;b=b+mod(a-b,1);
        [i_noir,j_noir]=find(f);
        R=sqrt( max(sum([i_noir-a j_noir-b].^2,2) ));

        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        %la matrice f est augmentée suffisamment pour%
        % pouvoir encadrer tout le cercle de rayon R %
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        rbar=floor(a-R);
        if rbar<1,
            f=[zeros(1-rbar,size(f,2));f];
            a=a+1-rbar;
        end
        rbar=ceil(R+a);
        if rbar>size(f,1),
            f=[f;zeros(rbar-size(f,1),size(f,2));];
        end
        rbar=floor(b-R);
        if rbar<1,
            f=[zeros(size(f,1),1-rbar) f];
            b=b+1-rbar;
        end
        rbar=ceil(R+b);
        if rbar>size(f,2),
            f=[f zeros(size(f,1),rbar-size(f,2))];
        end
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        %suite de l'algorithme de la figure 10%
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        alpha=R/N;
        Fki=zeros(1,(Q(end)+1)*N);
        n=0;
        for r=alpha:alpha:R
            n=n+1;
```

```
            yh=floor(r+b)+0.5;
            if r+b<=yh,yh=yh-1; end;

            y2_tilt=r;  teta2=pi/2;x2_tilt=0;
            x2=a; y2=y2_tilt+b;

            while x2>round(-0.7071068*r+a)+0.5 |
y2>round(0.7071068*r+b)+0.5
                y1_tilt=y2_tilt;   x1_tilt=x2_tilt;  x2_tilt=ceil(x2)-
a-0.5; if
x2_tilt>=x1_tilt, x2_tilt=x2_tilt-1; end;
                y2_tilt=sqrt(r^2-x2_tilt^2);
                if (y2_tilt+b)<yh,
                    y2_tilt=yh-b; x2_tilt=-sqrt(r^2-y2_tilt^2);
                    yh=yh-1;
                    x2=x2_tilt+a;x=round(x2);

                    y2=y2_tilt+b;y=ceil(y2);
                else
                    x2=x2_tilt+a;x=ceil(x2);


                    y2=y2_tilt+b;y=round(y2);
                end
                y_tilt =y-b; x_tilt =x-a;

                teta1 = teta2;     teta2 = acos(x2_tilt/r);

                rejt=([y1_tilt -x1_tilt x1_tilt -y1_tilt -[y1_tilt -
x1_tilt
x1_tilt -y1_tilt]]+j*[-x1_tilt y1_tilt y1_tilt -x1_tilt -[-x1_tilt
y1_tilt
y1_tilt -x1_tilt]]);
                rejt2=([y2_tilt -x2_tilt x2_tilt -y2_tilt -[y2_tilt -
x2_tilt
x2_tilt -y2_tilt]]+j*[-x2_tilt y2_tilt y2_tilt -x2_tilt -[-x2_tilt
y2_tilt
y2_tilt -x2_tilt]]);
                Fki(n)=Fki(n)+(teta1-teta2)*( f(-
x_tilt+a,y_tilt+b)+f(y_tilt+a,-
x_tilt+b)+f(y_tilt+a,x_tilt+b)+f(-x_tilt+a,-y_tilt+b)+f(x_tilt+a,-
y_tilt+b)+f(-
y_tilt+a,x_tilt+b)+f(y_tilt+a,-x_tilt+b)+f(x_tilt+a,y_tilt+b) );
                for q=Q,
                    Fki(n+N*(q))= Fki(n+N*(q))+(rejt2.^q-rejt.^q)*[ f(-
x_tilt+a,y_tilt+b) -f(y_tilt+a,-x_tilt+b) f(y_tilt+a,x_tilt+b) -f(-
x_tilt+a,-
y_tilt+b) f(x_tilt+a,-y_tilt+b) -f(-y_tilt+a,x_tilt+b) f(y_tilt+a,-
x_tilt+b) -
f(x_tilt+a,y_tilt+b) ]'   ;
                end;

            end;

            x1_tilt=x2_tilt;y1_tilt=y2_tilt;
            x2_tilt=-0.7071068*r; y2_tilt=abs(x2_tilt);
            x2=x2_tilt+a;x=round(x2);
```

```
            y2=y2_tilt+b;y=round(y2);
            y_tilt =y-b; x_tilt =x-a;

            teta1 = teta2;    teta2 = acos(x2_tilt/r);

            rejt=([y1_tilt -x1_tilt x1_tilt -y1_tilt -[y1_tilt -x1_tilt
x1_tilt
-y1_tilt]]+j*[-x1_tilt y1_tilt y1_tilt -x1_tilt -[-x1_tilt y1_tilt
y1_tilt -
x1_tilt]]);
            rejt2=([y2_tilt -x2_tilt x2_tilt -y2_tilt -[y2_tilt -
x2_tilt x2_tilt
-y2_tilt]]+j*[-x2_tilt y2_tilt y2_tilt -x2_tilt -[-x2_tilt y2_tilt
y2_tilt -
x2_tilt]]);
            Fki(n)=n*( Fki(n)+(teta1-teta2)*( f(-
x_tilt+a,y_tilt+b)+f(y_tilt+a,-
x_tilt+b)+f(y_tilt+a,x_tilt+b)+f(-x_tilt+a,-y_tilt+b)+f(x_tilt+a,-
y_tilt+b)+f(-
y_tilt+a,x_tilt+b)+f(y_tilt+a,-x_tilt+b)+f(x_tilt+a,y_tilt+b) ));
            for q=Q,
                Fki(n+N*(q))= ( n/(q*r^q) )*((Fki(n+N*(q))+(rejt2.^q-
rejt.^q)*[
f(-x_tilt+a,y_tilt+b) -f(y_tilt+a,-x_tilt+b) f(y_tilt+a,x_tilt+b) -f(-
x_tilt+a,-
y_tilt+b) f(x_tilt+a,-y_tilt+b) -f(-y_tilt+a,x_tilt+b) f(y_tilt+a,-
x_tilt+b) -
f(x_tilt+a,y_tilt+b)]'));
            end;
        end;
        Fk=[Fk; Fki]  ;
    end
    fclose(fid);
    delete image;
    load Fk; F=[F;Fk]; save Fk F;    F=[];
end
```

ANNEXE 2

Programme de calcul $[\varepsilon(p,Q,I)]_{p=0\ldots P,Q=0\ldots Q_{max},I=1\ldots I_{max}}$

L'algorithme de calcul du taux *leave-one-out* (figure 11) est implanté en langage C:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include "def.h"
#include "matrix.h"

extern universe *U;
FeatVector dis;
int I_MAX,I_MAX_K,*ppv;

/*
      étape 1: calcul des distances
*/
void estimate_error_KNN( n, FSV )
int n;
FeatSelectVector FSV;
{
float e, aux, x_n_i1;
int k1, k2, ii1, i1,i2, ii2,I_MAX, I_MAX_i_te,I_MAX_i2,rank;

for (i1=1;i1<I_MAX;ppv[i1++]=0);ppv[0]=U->C[0].numSampl;
i2 = 0;
for( k1 = 0; k1 < U->nrClass; k1++ )
{
            for( ii1 = 0; ii1 < U->C[0].numSampl; ii1++ )
            {
                  rank=FSV[n-1].rank;
                  x_n_i1=U->C[k1].S[ii1*U->nrFeat+rank];
                  I_MAX_i_te=I_MAX*i2;
                  i1 = 0;
                  for( k2 = 0; k2 < U->nrClass; k2++ )
                  {
                        for( ii2 = 0; ii2 < U->C[0].numSampl; ii2++ )
                        {
                              if( i1>i2 )
                              {
                                    aux = x_n_i1 - U->C[k2].S[(ii2)*U-
>nrFeat+rank];
                                    dis[I_MAX*i1+i2] =(dis[I_MAX_i2+i1]
=
                                    dis[I_MAX_i2+i1]+aux * aux);
                              }
                              i1++;
                        }
                  }
                  i2++;
             }
}
}
/*
      étape 2: calcul du taux leave-one out
*/
```

```
float estimate_error_KNN3( n, FSV)
int n;
FeatSelectVector FSV;
{
int e,KI, k_te, ii_te, i_te, kk, k_tr,i_tr,I_MAX_i_te;

for(U->C[0].numSampl= 0;U->C[0].numSampl<I_MAX_K;)
{
      KI = U->nrClass*(++U->C[0].numSampl);
      e = 0;
      for( k_te = 0; k_te < U->nrClass; k_te++ )
      {
            i_te=k_te*I_MAX_K;
            for( ii_te = 0; ii_te < U->C[0].numSampl-1; ii_te++ )
            {
                  kk = div(ppv[i_te],I_MAX_K).quot;
                  I_MAX_i_te=I_MAX*i_te;
                  i_tr = U->C[0].numSampl-1;
                  for( k_tr = 0; k_tr < U->nrClass; k_tr++ )
                  {
                        if( dis[I_MAX_i_te+i_tr]  <
dis[I_MAX_i_te+ppv[i_te]] )
                        {
                              ppv[i_te] = i_tr;
                              kk = k_tr;
                        }
                        i_tr+= I_MAX_K;
                  }
                  if( k_te != kk )  e++;
                  i_te++;
            }
            kk=div(ppv[i_te],I_MAX_K).quot;
            I_MAX_i_te=I_MAX*i_te;
            i_tr = 0;
            for( k_tr = 0; k_tr < U->nrClass; k_tr++ )
            {
                  i_tr= k_tr*I_MAX_K;
                  for( ii_te = 0; ii_te < U->C[0].numSampl; ii_te++ )
                  {
                        if( i_tr != i_te )
                              if( dis[I_MAX_i_te+i_tr]  <

                              dis[I_MAX_i_te+ppv[i_te]] )
                              {
                                    ppv[i_te] = i_tr;
                                    kk = k_tr;
                              }
                        i_tr++;
                  }
            }
            if( k_te != kk )  e++;
      }
      e/=(float)KI;
}
return( e );
}
```

```
void est_err_nearneib()
{
int n, numSampl;
FILE *ef = NULL;
str80 errFile, name;

strcpy( errFile, DATA_DIR );
gets( name ); strcat( errFile, name ); ef = fopen( errFile,
f_open_text_w );

I_MAX= U->nrClass*U->C[0].numSampl; I_MAX_K=I_MAX*0.1;
d=(float*)malloc(I_MAX*I_MAX*sizeof(float));
ppv=(int*)malloc(I_MAX*sizeof(int));
for (n=0;n<I_MAX*I_MAX;d[n++]=0);
for( n = 0; n < U->nrClass; n++ ) numSampl[n]=U->C[n].numSampl;
for (_2exp5moins_p=32; _2exp5moins_p>=1; _2exp5moins_p/=2)
{
      for (n=0;n<I_MAX*I_MAX;d[n++]=0);
      for( n = 0; n < U->nrClass; n++ ) numSampl[n]=U->C[n].numSampl;
      for(n=_2exp5moins_p; n <=32; n+=_2exp5moins_p )
      {
            U->C[0].numSampl= numSampl[0];
            calcul_d( n, U->FSV );
      }
      calcul_e( 32, U->FSV, ef, protocolFile);
      for(_2n_plus32plusQ64=31+2*_2exp5moins_p; _2n_plus32plusQ64 <U-
>nrSelFeat; _2n_plus32plusQ64+=2*_2exp5moins_p)
      {
            U->C[0].numSampl= numSampl[0];
            calcul_d( _2n_plus32plusQ64, U->FSV );calcul_d(
_2n_plus32plusQ64+1, U->FSV );
            if (div(_2n_plus32plusQ64-31,64).rem==0 )
            {
                  calcul_e( _2n_plus32plusQ64+1, U->FSV, ef,
protocolFile)
            }
      }
}
if( protocolFile ) fclose( ef );
free(ppv);free(d);U->C[0].numSampl=I_MAX*0.1;
}
void set_K(){}
void initMINERR(){}
float select_multivariate_minerr() {}
void featSelectMinErr_SFS(){}
int id_one_sample_KNN(){}
```

BIBLIOGRAPHIE

1. Mori, S., Nishida, H., Yamada, H. (1999). *Optical character recognition*, New York, N.Y. : J. Wiley and Sons.
2. Trier, O., Jain, A. K., Taxt, T. (1996). Feature extraction methods for character recognition - a survey. *Pattern Recognition, 29*(4), pp. 641-662.
3. Shen, D., Ip, H. (1999). H.s.discriminative wavelet shape descriptors for recognition of 2-d patterns. *Pattern Recognition, 32*(2) pp. 151-165.
4. Chen, G., Bui, T.D. (1999). Invariant fourier-wavelet descriptor for pattern recognition. *Pattern Recognition, 32*(6) pp. 1083-1088.
5. Tang, Y.Y., Li, B.F., Ma, H., Liu, J. (1998). Ring-projection-wavelet-fractal signatures: a novel approach to feature extraction. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, 45*(8) pp. 1130-1134.
6. Wunsch, P., Laine, A.F. (1995). Wavelet descriptors for multiresolution recognition of handprinted characters. *Pattern Recognition, 28*(8) pp. 1237-1249.
7. Lee, S.-W., Kim, C.-H., Ma, H., Tang, Y.Y. (1996). Multiresolution recognition of unconstrained handwritten numerals with wavelet transform and multilayer cluster neural network. *Pattern Recognition, 29*(12) pp. 1953-1961.
8. Laine, A. F., Schuler, S. (1994). Hexagonal wavelet representations for recognizing complex annotations. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Proceedings of the 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 740-745.
9. Duda, R. O., Hart, P. E., Stork, D. G. (2001). *Pattern classification*, New York, N.Y. : J. Wiley and Sons.
10. Saito, N., Coifman, R. R. (1994). Constructions of local orthonormal bases for classification and regression. *Comptes Rendus Acad. Sci. Paris*, Serie I , 319 pp.191-196.
11. Chan, Y. T. (1995). *Wavelet basics*, Boston, Mass. : Kluwer Academic.
12. Abry, P., Flandrin, P. (1994). On the initialization of the discrete wavelet transform algorithm. IEEE Signal Processing Letters, 1(2) pp. 32-34.
13. Ayat, N.E., Cheriet, M., Suen., C.Y. (2002). KMOD - A Two-parameter SVM Kernel for Pattern Recognition. À apparaître dans 16th International Conference on Pattern Recognition (ICPR 2002).