Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Abdelhamid Ibn Badis - Mostaganem
Faculté des Sciences Exactes et de l'Informatique
Département de Mathématiques et Informatique
Filière : Mathématiques

MÉMOIRE DE FIN D'ÉTUDES

Pour l'Obtention du Diplôme de Master en Mathématiques délivré par

Université de Mostaganem

Spécialité "Modélisation, Contrôle et Optimisation"

présentée par :

Nabila BENYOUCEF

Ondelettes de Gabor et Analyse Discriminante de Fisher pour l'identification d'un visage

soutenu publiquement le 26 Juin 2022 devant le jury composé de :

Président :	Hocine ABLAOUI	MAA	UMAB
Examinateur :	Oussama BOUANANI	MCB	UMAB
Encadreur :	Abdessamad AMIR	Professeur	UMAB
Co-encadreur :	Mokhtar El Amine KORICHI	Doctorant	UMAB

Année Universitaire : 2021 / 2022

MASTER

Remerciements

J'adresse mes plus sincères remerciements à **ALLAH**, de nous avoir donné la santé, la volonté et la patience pour mener à terme notre étude et pouvoir réaliser ce travail de recherche.
C'est avec un immense plaisir que j'exprime mes sincères remerciements et ma profonde gratitude à mon encadrant, le professeur **AMIR Abdessamad**, pour l'aide compétente qu'il m'a apportée, pour ses encouragements, et pour tous ses précieux conseils. Je salue sa gentillesse et son aide.
C'est certes avec joie et fierté que je dépose ce mémoire aujourd'hui.
Je voudrais également remercier Monsieur **KORICHI Mokhtar El Amine** pour son aide et ses conseils.
J'adresse aussi mes vifs remerciments aux membres des jurys Monsieur **ABLAOUI Hocine** et Monsieur **BOUANANI Oussama** qui ont accepté d'évaluer et juger mon travail.
J'adresse un grand merci à mes parents, la plus grande bénédiction que ALLAH m'a accordée, sans eux, je ne serais jamais là où je suis maintenant.
J'exprime ma profonde gratitude à ma soeur, mes frères la source de force et de bonheur dans ma vie, qui ont toujours été présents lorsque j'en ai eu besoin.
J'aimerais également à remercier Mlle BOUBEKEUR Marwa pour son aide et ses conseils cette année.
J'exprime aussi mes remerciements à mes amis qui m'ont toujours encouragé.
Enfin, je profite cette occasion pour exprimer mes remerciements à Monsieur FETTOUCH.

Ondelettes de Gabor et Analyse Discriminante de Fisher pour l'identification d'un visage

Résumé : Les travaux menés durant la préparation de ce mémoire concernent l'application de la méthode d'Analyse Discriminate de Fisher dans le système de la détection faciale. L'Analyse discriminante de Fisher (ADF) est réputée pour être le classifieur optimale lorsque les données sont linéairement séparables. L'étude de ce projet est donc double.
Tout d'abord, nous étudierons ADF comme algorithme de classification supervisé, une étude comparative avec d'autres classifieurs. Nous montrons ensuite que l'approche d'associer le filtre de Gabor pour l'extraction des caractéristiques avec ADF est robuste pour l'identification faciale.

Mots-Clés. Analyse discriminante, classification, filtre de Gabor, identification faciale.

Gabor wavelets and Fisher Discriminant Analysis for face identification

Abstract : The work carried out during the preparation of this thesis concerns the application of Fisher's Discriminate Analysis method in the facial detection system. Fisher Discriminant Analysis (FDA) is known to be the optimal classifier when the data are linearly separable. The study of this project is therefore twofold.
First, we will study FDA as a supervised classification algorithm, a comparative study with other classifier. We then show that the approach of combining Gabor's filter for feature extraction with FDA is robust for facial identification.

Keywords. Discriminant analysis, classification, Gabor filter, face identification.

مرشح غابور و تحليل فيشر التمييزي لتحديد الوجه

ملخص :
العمل الذي تم به أثناء إعداد هذه الأطروحة يتعلق بتطبيق طريقة فيشر للتحليل التمييزي في نظام الكشف عن الوجه. من المعروف أن تحليل فيشر التمييزي هو المصنف الأمثل عندما تكون البيانات منفصلة خطيا. لذلك فإن دراسة هذا المشروع مضاعفة.
أولا، سوف ندرس طريقة فيشر للتحليل التمييزي كخوارزمية تصنيف، دراسة مقارنة مع مصنف آخر. نوضح بعد ذلك أن نهج الجمع بين مرشح غابور لاستخراج الميزات مع طريقة فيشر للتحليل التمييزي قوي لتحديد الوجه.

الكلمات الرئيسية.
تحليل تمييزي، تصنيف، مرشح غابور، تحديد الوجه.

Table des figures

Liste des tableaux

Table des matières

Liste des abréviations

ACP : Analyse en Composantes Principales.
ADF : Analyse Discriminante de Fisher.
GDA : Analyse Discriminante Généralisée.
SVD : Décomposition en Valeurs Singulières.
SVM : Machines à Vecteurs de Support.

Index des notations

X	:	La matrice des données.
X_i	:	La matrice des données de la classe i.
$\mathrm{X}_{:j}$	:	La j^{ieme} colonne de la matrice X.
$\mathrm{X}^{(i)}_{:j}$	:	La j^{ieme} colonne de la matrice des données de la classe i.
y_j	:	La classe de $\mathrm{X}_{:j}$.
n	:	Le nombre de colonnes de X.
d	:	Le nombre de lignes de X.
n_i	:	Le nombre de données de la classe i.
μ_i	:	La moyenne des données de la classe i.
$var(\mathrm{X})$	:	La matrice de variance de X.
Σ_i	:	La matrice de variance des données de la classe i.
ϕ	:	La fonction caractéristique.
F	:	L'espace caractéristique.
λ_i	:	La i^{ieme} valeur propre.
K	:	La fonction noyau.
$*$	:	Le produit de convolution.
$<.,.>_{\mathrm{F}}$	:	Le produit scalaire dans l'espace F.
$\|.\|$	:	La norme euclidienne.
L	:	La fonction de Lagrange.
$\frac{\partial \mathrm{L}}{\partial \lambda}$	:	La dérivée partielle de L par rapport à λ.
TF	:	La transformée de Fourier.
TF^{-1}	:	La transformée de Fourier inverse.

Introduction

La détection d'un visage dans une image par un ordinateur, est un problème qui se pose depuis la naissance des sciences informatiques. Il s'agit d'identifier les critères qui permettent de certifier que quelque chose est un visage ou non. Ainsi, la question posée dans ce mémoire est de savoir comment une machine peut reconnaitre un visage dans une image ?

L'être humain, utilise pour reconnaître un visage, différentes caractéristiques qui varient entre la géométrie, la texture et les couleurs des différentes régions du visage : les yeux, les lèvres, le nez et les joues. Grâce à cette remarque, plusieurs études ont vu le jour dans le but de rapprocher ce concept sur une machine.

La détection de visage est un domaine de la vision par ordinateur consistant à détecter un visage humain dans une image numérique. C'est un exemple de la reconnaissance d'objet, où l'on cherche à développer un programme informatique qui, dans une image arbitraire, peut détecter et localiser des visages humains.

La détection de visage a de très nombreuses applications directes dans les domaines de la sécurité, le contrôle d'accès, l'interaction personne-machine et d'autres domaines de la vision par ordinateur et de l'apprentissage automatique.

La majorité des approches utilisées pour la détection de visage reposent sur deux tâches. La première concerne la description numérique précise de ce qui distingue les visages humains des autres objets, cette tâche est appelée extraction des caractéristiques dans le domaine du traitement d'image numérique. La deuxième tâche est généralement un algorithme de classification, qu'il soit supervisé ou non suivant la méthode, qui devra reconnaitre avec une bonne précision un visage d'un non-visage.

L'étape de l'extraction des caractéristiques est une étape primordiale dans le processus de la reconnaissance faciale. En traitement d'images numériques, on extrait les caractéristiques par des filtres ou des ondelettes, par exemple les filtres de Gabor, les ondelettes de Haar,..., etc, qui sont des techniques qui permettent d'extraire à partir d'une image un vecteur de grande dimension, nous fournissant des informations importantes qui nous aident à faire notre classification.

Plusieurs méthodes de classification ont été appliquées dans ce contexte afin d'obtenir plus de performances, parmi ces méthodes les méthodes des machines à vecteurs de support (SVMs) [20], les méthodes de la régression logistique,..., etc.

La méthode d'Analyse Discriminante de Fisher (ADF) [10] cherche à effectuer une réduction de dimension. De plus, ADF est considérée comme une méthode d'apprentissage supervisée, donc capable de faire une classification avec réduction de dimension. Ce travail représente une étude approfondie des analyses discriminantes, particulièrement ADF.

Dans ce mémoire, nous étudierons deux méthodes de classification qui sont basées sur la notion de la réduction de dimension ADF et la méthode d'Analyse Discriminante de Fisher Généralisée (GDA) [16] qui est une extension de ADF, dans le cas des données non linéairement séparables, ensuite nous étudierons les méthodes d'extraction des

caractéristiques par les filtres de Gabor, et enfin nous appliquons la méthode ADF et GDA dans le contexte de la détection de visage.

Nous décrivons maintenant brièvement le contenu du manuscrit :

Dans le premier chapitre, nous introduisons d'abord l'Analyse en Composantes Principales (ACP) [13], comme méthode de réduction de dimension. Ensuite nous rappelons la théorie de ADF introduite par Fisher [10]. À la fin de ce chapitre nous exposerons une exécution sur un problème réel.

Le second chapitre est dédié à une extension de ADF dans le cas où les données sont non linéairement séparables. En effet une introduction à la technique des noyaux est nécessaire pour l'étude. Ces deux premières parties représentent le travail théorique et le reste de ce travail est didié à la pratique.

Dans le troisième chapitre, nous présentons les outils du traitement d'images numériques et décrivons les étapes fondamentales de la détection faciale.

Le quatrième et le dernier chapitre est dédié aux simulations numériques, essentiellement une comparaison entre les deux méthodes de classification ADF et GDA dans le contexte de la détection faciale est présenté.

Chapitre 1

Analyse Discriminante de Fisher

1 Introduction

La classification vise à générer une fonction (appelé un classifieur) sur un ensemble de donné (appelé ensemble d'entrainement) préétiqueté pour les classifier correctement. Il existe plusieurs approches et méthodes de classification : nous nous intéressant principalement à la méthode dite Analyse Discriminante de Fisher (ADF) est une méthode de réduction de dimension pour des fins de classification. Comme introduction nous allons d'abord présenter une méthode classique de réduction de dimension.

2 Analyse en Composantes Principales

L'Analyse en Composantes Principales (ACP) est une méthode de réduction de dimension qui permet de projeter les données d'un espace de dimension d à un nouvel espace de dimension inférieur p, où $p \ll d$.
Il s'agit en fait, de filtrer les données non pertinentes et bruités, afin de réduire la taille des données et garder uniquement les variables contenants des informations utiles, appelées : **Composantes Principales**.
Évidemment, qui dit réduction de dimension dit perte d'informations. C'est là tout l'enjeu que représente une analyse en Composantes Principales. Il faut pouvoir réduire la dimension des données tout en conservant le maximum d'informations.

2.1 Principes de l'Analyse en Composantes Principales

Considérons une matrice des données $\mathrm{X} \in \mathbb{R}^{d \times n}$, où chaque donnée $\mathrm{X}_{:j}$ est un vecteur unidimensionnel de $\mathbb{R}^d$, pour tout $j = \overline{1,n}$.
La technique ACP consiste à projeter les données sur une direction $u \in \mathbb{R}^d$, tout en maximisant la variation des données.
En effet :
La projection des données sur le vecteur $u \in \mathbb{R}^d$ est $u^{\mathrm{T}}\mathrm{X}_{:j} \in \mathbb{R}$, pour $j = \overline{1,n}$.
La matrice de variance de X est donnée par [15] :

$$\mathrm{S} = \frac{1}{d}\tilde{\mathrm{X}}\tilde{\mathrm{X}}^{\mathrm{T}} \in \mathbb{R}^{d \times d}, \tag{1.1}$$

où $\tilde{X}_{ij} = X_{ij} - \mu_i$, avec $\tilde{X}_{ij} \in \mathbb{R}$, pour $i = \overline{1,d}$ et $j = \overline{1,n}$, où $\mu_i = \frac{1}{n}\sum_{j=1}^{n} X_{ij}$ est la moyenne de $X_{i:}$, pour $i = \overline{1,d}$.
La variance des données projetées [15] :

$$var(u^T X) = u^T S u. \tag{1.2}$$

Le vecteur directeur u donnant une variance maximale, est solution du problème d'optimisation :

$$\begin{cases} \max u^T S u \ ; \\ \quad u \in \mathbb{R}^d, \end{cases} \tag{1.3}$$

comme la fonction objective du problème (1.3) est semi définie positive, ce dernier est mal posé, car la fonction objective tend vers $+\infty$. Pour résoudre ce problème, on peut se restreindre au u vérifiant $\|u\| = 1$, vu que nous cherchons une direction qui peut être définie par n'importe quel vecteur u de norme finie. (1.3) devient :

$$\begin{cases} \max\limits_{u \in \mathbb{R}^d} u^T S u \ ; \\ \quad \|u\|^2 = 1. \end{cases} \tag{1.4}$$

La maximisation de la variance de la projection des données, sous condition que le vecteur u soit de norme égale à 1, est un problème d'optimisation quadratique sous contraintes égalités qui peut être résolu au moyen de la méthode des multiplicateurs de Lagrange [14]. Sous cette dernière forme, le problème d'optimisation sous contrainte égalité maximisant la variance de la projection des données est formulé par la fonction de Lagrange :

$$L(u, \lambda) = u^T S u - \lambda(u^T u - 1), \tag{1.5}$$

où $\lambda \in \mathbb{R}$ est le multiplicateur de Lagrange. Donc la solution optimale u du problème (1.4) est obtenue en annulant les dérivées premières de la fonction de Lagrange.

$$\begin{cases} \frac{\partial L}{\partial u}(u, \lambda) = 0, \\ \\ \frac{\partial L}{\partial \lambda}(u, \lambda) = 0. \end{cases} \tag{1.6}$$

Ceci implique la solution du système

$$\begin{cases} S u = \lambda u, \\ \\ u^T u = 1. \end{cases} \tag{1.7}$$

La résolution de ce système consiste à trouver les valeurs propres et les vecteurs propres de la matrice de variance S sous contrainte de normalisation des vecteurs propres.
Comme $u^T S u = \lambda$, alors la valeur optimale du problème (1.4) est λ. Notons par $u_1, u_2, ..., u_d$ les d vecteurs propres qui sont associés au $\lambda_1, \lambda_2, ..., \lambda_d$, les d valeurs propres de la matrice de variance S (avec $\lambda_1 > \lambda_2 > ... > \lambda_d$).
Les vecteurs propres u_i sont appelées **les Composantes Principales** pour $i = 1, 2, ..., d$.

Comme S est une matrice symétrique et semi définie positive par définition, alors les valeurs propres $\lambda_1, \lambda_2, ..., \lambda_d$ de S sont non négatives. La direction du vecteur propre u_1 associée à la plus grande valeur propre λ_1 contient le maximum d'informations. Inversement, la direction du vecteur propre directeur u_d associée à la plus petite valeur propre λ_d est celle qui présente le minimum d'informations.
Nous pouvons également utiliser la décomposition en valeurs singulières (SVD) [9] de la matrice $\frac{1}{\sqrt{d}}\tilde{X}$ pour trouver les composantes principales qui sont les solutions du problème (1.4).
La décomposition de $\frac{1}{\sqrt{d}}\tilde{X}$ en valeurs singulières est donnée par :

$$\frac{1}{\sqrt{d}}\tilde{X} = U\Sigma V^T, \tag{1.8}$$

où :
• Les colonnes de U sont les vecteurs propres de $S = \frac{1}{d}\tilde{X}\tilde{X}^T$.
• Σ est une matrice diagonale qui contient les valeurs propres de $\frac{1}{d}\tilde{X}\tilde{X}^T$ en ordre décroissant.
• Les colonnes de V^T sont les vecteurs propres de $\frac{1}{d}\tilde{X}^T\tilde{X}$.
Comme Σ contient les valeurs propres de $\frac{1}{d}\tilde{X}\tilde{X}^T$ en ordre décroissant, alors la i^{eme} colonne de U représente la i^{eme} Composante Principale.

Algorithm 1.1 Analyse en Composantes Principales (ACP)

Données : La matrice des données $X \in \mathbb{R}^{d\times n}$ et $p \in \mathbb{N}$ avec $p < d$.
Résultat : Z.

1: Calculer m, la moyenne des colonnes de X ;
2: Calculer $\tilde{X} = X - m * ones(1, n)$;
3: Calculer les matrices U, Σ et V par la décomposition en valeurs singulières de $\frac{1}{\sqrt{d}}\tilde{X}$;
4: $u_i = U(:, i),\ i = \overline{1, p}$, les p Composantes Principales ;
5: Calculer $Z = u^T X$.

L'algorithme (1.1), donne la projection des données sur les p composantes principales.

La méthode de réduction de dimension présentée précédemment, ne prend cependant pas compte de la classe des données, c'est une méthode non supervisée. L'Analyse Discriminante de Fisher combine quant à elle, la réduction de dimension et la classification.

3 Analyse Discriminante de Fisher (ADF)

L'Analyse Discriminante de Fisher (ADF) est un classifieur linéaire qui projette un vecteur de caractéristique de dimension d sur un hyperplan qui divise l'espace en deux demi-espaces, chaque demi-espace représente une classe. Donc ADF est une méthode de réduction de dimension mais à but de classification [8].

3.1 Principes de l'Analyse Discriminante de Fisher

Supposons que nous n'ayons que deux classes $y_i \in \{0, 1\}, i = \overline{1, n}$. Soit X la matrice des données telle que $X = [X_{:1}\ X_{:2}\ ...\ X_{:n}] \in \mathbb{R}^{d\times n}$, dans tout ce qui suit, nous allons représenter l'ensemble des données de la classe 0 par l'indice 0 et ceux de la classe 1 par

l'indice 1.
L'idée de l'Analyse Discriminante de Fisher est de réduire la dimensionnalité des données à une seule dimension. C'est-à-dire, nous projetons toutes les données de dimension d en une dimension en trouvant $w^{\mathrm{T}}\mathrm{X}$ avec $w \in \mathbb{R}^d$ et $\mathrm{X}_{:j} \in \mathbb{R}^d$ tel que :

$$\begin{aligned} \mathrm{Z}_j &= w^{\mathrm{T}}\mathrm{X}_{:j}, \\ &= \sum_{i=1}^{d} w_i \mathrm{X}_{ij}. \end{aligned} \tag{1.9}$$

$\mathrm{Z} \in \mathbb{R}^n$ est la nouvelle représentation des données de dimension 1 que nous utiliserons pour la classification.
Le but de ADF est de trouver une direction telle que les données projetées $w^{\mathrm{T}}\mathrm{X}_{:j}$ soient bien séparé.
Notons par :

$$\mu_0 = \frac{1}{n_0} \sum_{j:y_j=0} \mathrm{X}_{:j} \ , \qquad \mu_1 = \frac{1}{n_1} \sum_{j:y_j=1} \mathrm{X}_{:j},$$

où :
n_0 est le nombre des données de la classe 0,
n_1 est le nombre des données de la classe 1,
et
μ_0 est la moyenne des données de la classe 0,
μ_1 est la moyenne des données de la classe 1.
Le principe de ADF est de trouver une représentation des données qui permette de discriminer les classes le mieux possible. Pour ce faire, il faut :

1- Maximiser la distance entre les moyennes de classes projetées ;
2- Minimiser la variance de chaque classe.

La distance entre les moyennes de classe projetées est :

$$\begin{aligned} (d(w^{\mathrm{T}}\mu_0, w^{\mathrm{T}}\mu_1))^2 &= \| w^{\mathrm{T}}\mu_0 - w^{\mathrm{T}}\mu_1 \|^2, \\ &= (w^{\mathrm{T}}\mu_0 - w^{\mathrm{T}}\mu_1)^{\mathrm{T}}(w^{\mathrm{T}}\mu_0 - w^{\mathrm{T}}\mu_1), \\ &= (w^{\mathrm{T}}\mu_0 - w^{\mathrm{T}}\mu_1)(w^{\mathrm{T}}\mu_0 - w^{\mathrm{T}}\mu_1)^{\mathrm{T}}, \\ &= w^{\mathrm{T}}(\mu_0 - \mu_1)(\mu_0 - \mu_1)^{\mathrm{T}} w, \\ &= w^{\mathrm{T}}\mathrm{S}_{\mathrm{B}} w, \end{aligned} \tag{1.10}$$

où $\mathrm{S}_{\mathrm{B}} = (\mu_0 - \mu_1)(\mu_0 - \mu_1)^{\mathrm{T}}$ est la matrice de variance entre-classes [8].
La variance de projection des données de la première classe est : $w^{\mathrm{T}}\Sigma_0 w$ et de la deuxième classe est : $w^{\mathrm{T}}\Sigma_1 w$, où Σ_0 est la variance des données de la première classe et Σ_1 est la variance des données de la deuxième classe (voir 1.1).
Minimiser la variance de projection de chaque classe équivaut à minimiser la somme de toutes les variances.
La somme de projection des variances est :

$$\begin{aligned} w^{\mathrm{T}}\Sigma_0 w + w^{\mathrm{T}}\Sigma_1 w &= w^{\mathrm{T}}(\Sigma_0 + \Sigma_1) w, \\ &= w^{\mathrm{T}}\mathrm{S}_w w, \end{aligned} \tag{1.11}$$

où $\mathrm{S}_w = \Sigma_0 + \Sigma_1$ est la matrice de variance intra-classes [8].
Afin de maximiser la distance entre les moyennes de classes projetées $(\max_{w \in \mathbb{R}^d} w^{\mathrm{T}}\mathrm{S}_{\mathrm{B}} w)$

et minimiser la variance de projection de chaque classe ($\min_{w\in\mathbb{R}^d} w^{\mathrm{T}}\mathrm{S}_w w$) en même temps dans un seul critère, nous pouvons maximiser la quantité :

$$\max_{w\in\mathbb{R}^d} \frac{w^{\mathrm{T}}\mathrm{S}_{\mathrm{B}} w}{w^{\mathrm{T}}\mathrm{S}_w w}, \tag{1.12}$$

de plus, comme $\forall\alpha\in\mathbb{R}^*$:

$$\frac{(\alpha w)^{\mathrm{T}}\mathrm{S}_{\mathrm{B}}(\alpha w)}{(\alpha w)^{\mathrm{T}}\mathrm{S}_w(\alpha w)} = \frac{w^{\mathrm{T}}\mathrm{S}_{\mathrm{B}} w}{w^{\mathrm{T}}\mathrm{S}_w w}.$$

Si w^* est solution du problème (1.12), alors $\forall\alpha\in\mathbb{R}^*$, αw^* est aussi solution. On s'intéresse à une infinité de solutions normalisée ($w^{\mathrm{T}}\mathrm{S}_w w = 1$) de ce problème, on considère la solution vérifiant $w^{\mathrm{T}}\mathrm{S}_w w = 1$. Le problème (1.12) devient :

$$\begin{cases} \max\limits_{w\in\mathbb{R}^d} w^{\mathrm{T}}\mathrm{S}_{\mathrm{B}} w \ ; \\ \\ \quad w^{\mathrm{T}}\mathrm{S}_w w = 1. \end{cases} \tag{1.13}$$

Le problème (1.13) est un problème d'optimisation quadratique avec contrainte égalité qui peut être résolu par la méthode des multiplicateurs de Lagrange, avec L représente la fonction de Lagrange associée au problème (1.13) [14] :

$$\mathrm{L}(w,\lambda) = w^{\mathrm{T}}\mathrm{S}_{\mathrm{B}} w - \lambda(w^{\mathrm{T}}\mathrm{S}_w w - 1), \tag{1.14}$$

où $\lambda\in\mathbb{R}$ représente le multiplicateur de Lagrange. La fonction de Lagrange permet de trouver les solutions optimales du problème (1.12) sous la contrainte
$w^{\mathrm{T}}\mathrm{S}_w w = 1$.
Les valeurs optimales de (1.13) sont obtenues en annulant la dérivée première de L par rapport à w :

$$\frac{\partial \mathrm{L}}{\partial w}(w,\lambda) = 2\mathrm{S}_{\mathrm{B}} w - 2\lambda \mathrm{S}_w w,$$

donc,

$$\begin{aligned} \frac{\partial \mathrm{L}}{\partial w}(w,\lambda) = 0 &\iff \mathrm{S}_{\mathrm{B}} w = \lambda \mathrm{S}_w w, \\ &\iff \mathrm{S}_w^{-1}\mathrm{S}_{\mathrm{B}} w = \lambda w. \end{aligned} \tag{1.15}$$

w est le vecteur propre de $\mathrm{S}_w^{-1}\mathrm{S}_{\mathrm{B}}$ associé à la valeur propre λ dans le cas où la matrice S_w est inversible. Dans le cas de non inversibilité ceci est un autre problème, dans l'algorithme ADF nous allons uniquement considérer le cas inversible.
Trouver un vecteur propre est souvent un problème couteux en grande dimension, il est souvent préférable de contourner ce problème.
Comme le rang de la matrice S_{B} est égal à 1, alors le rang de $\mathrm{S}_w^{-1}\mathrm{S}_{\mathrm{B}}$ est égal à 1 (par propriété du rang de produit de matrices). Alors il existe un seul vecteur propre w qui représente la meilleure direction qui sépare les données projetées.
Revenons à nos problème :

$$\mathrm{S}_w^{-1}\mathrm{S}_{\mathrm{B}} w = \lambda w.$$

Nous allons réécrire le problème précédent sous la forme suivante :

$$\begin{aligned} S_w^{-1}(\mu_0-\mu_1)(\mu_0-\mu_1)^T w = \lambda w \Leftrightarrow S_w^{-1}(\mu_0-\mu_1)C = \lambda w, \quad C = (\mu_0-\mu_1)^T w, \\ \Leftrightarrow S_w^{-1}(\mu_0-\mu_1) \sim \lambda w. \end{aligned} \tag{1.16}$$

Donc, w est représenté par $S_w^{-1}(\mu_0-\mu_1)$, alors :

$$w^* = S_w^{-1}(\mu_0-\mu_1), \tag{1.17}$$

est la solution du problème

$$\begin{cases} \max\limits_{w\in\mathbb{R}^d} w^T S_B w \ ; \\ \\ \quad w^T S_w w = 1. \end{cases} \tag{1.18}$$

Algorithm 1.2 Analyse Discriminante de Fisher (ADF)

Données : La matrice de données X_0 de la classe 0 et la matrice de données X_1 de la classe 1.
Résultat : Z.

1: Calculer μ_0 et μ_1, les moyennes de X_0 et X_1 respectivement ;
2: Calculer Σ_0 et Σ_1, les variances de X_0 et X_1 respectivement ;
3: Calculer $S_w^{-1} = (\Sigma_0 + \Sigma_1)^{-1}$;
4: Calculer $w = S_w^{-1}(\mu_0 - \mu_1)$;
5: Calculer $Z = w^T X$.

L'algorithme (1.2), nous donne une projection des données sur une droite, où les données associées aux deux classes sont bien séparées par un seuil. La méthode ADF est ullistré par l'exemple illustratif ci-dessous.

3.2 Exemple illustratif

Base de donnée MNIST

MNIST Handwritten Digit [19] est une base des données des chiffres écrits manuellement. C'est un jeu de données très utilisé en apprentissage automatique. Elle regroupe 60000 images d'apprentissages et 10000 images de tests. Chaque image dans cette base de donnée représente une calligraphie différente des chiffres entre 0 et 9. Ces images sont en niveaux de gris (28×28 pixels). La classification revient à pouvoir correctement reconnaitre le nombre associé à la calligraphie. C'est un problème multi-classes. Dans notre exemple nous nous contentons d'une classification binaire en appliquant ADF uniquement sur deux chiffres, à savoir le deux et le trois (voir 1.1).

FIGURE 1.1 – Différentes représentations des nombres 2 et 3 dans MNIST.

Classification par ADF

Nous allons dans un premier temps scinder les données en deux sous ensembles, notée X_2 et X_3, contenant les vecteurs des données relatives aux images de 2 et 3 respectivement. La figure (1.2) représente l'application de l'algorithme (1.2) sur les bases des données X_2 et X_3. La projection des images sur l'axe des x, donne l'indice des données. Alors que, la projection des images sur l'axe des y, donne le point Z de l'algorithme (1.2). Tous les points entre l'intervalle $[-40,0]$ peuvent être considérer comme un seuil. Cependant, le seuil peut être calculer par l'expression suivante :

$$\delta = \frac{w^T \mu_2 + w^T \mu_3}{2}, \tag{1.19}$$

où μ_2, μ_3 représentent les moyennes de X_2 et X_3 respectivement.

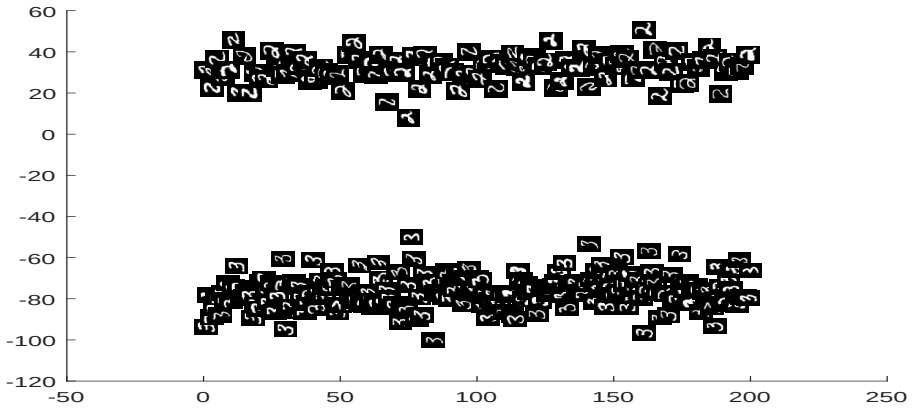

FIGURE 1.2 – Discrimination des classes avec la fonction ADF

Nous avons calculé la valeur du δ dans MATLAB qui est égale à -22.67, nous remarquons dans la figure (1.2) que les valeurs des données projetées de la classe 2 sont supérieurs à δ et les valeurs des données projetées de la classe 3 sont inférieurs à δ et les données projetées sont bien séparées.

Dans ce chapitre, nous avons vu une méthode utile sur les problèmes de la classification, mais cette méthode échoue lorsqu'il s'agit d'un problème où les données sont non

linéairement séparables. Dans le chapitre suivant, nous allons discuter une méthode qui généralise l'Analyse Discriminante de Fisher.

Chapitre 2

Analyse Discriminante Généralisée

1 Introduction

La méthode ADF étant une méthode robuste dans le cas linéaire, elle reste non fiable dans le cas où les données sont non linéairement séparables. Dans ce contexte, une nouvelle méthode a été introduite [4] afin d'exploiter les propriétés des noyaux. Dans ce chapitre nous allons d'abord introduire les notions relatives aux noyaux et généraliser celles de ADF pour le cas des noyaux et enfin développer la méthode GDA qui généralise ADF, dans le cas non linéaire.
Soit X la matrice des données, tel que $X = [X_0, X_1] \in \mathbb{R}^{d \times n}$,
où $X_i = [X^{(i)}_{:1} X^{(i)}_{:2} \ ... \ X^{(i)}_{:n_i}] \in \mathbb{R}^{d \times n_i}$, avec $n_i = card(X_i)$ pour $i = 0, 1$ et $n = n_0 + n_1$.

2 Introduction aux noyaux

Les méthodes des noyaux consistent à projeter les données de l'espace $\mathbb{R}^d$ dans un autre espace de dimension plus élevée où les données qui étaient non linéairement séparables peuvent le devenir.
Soit E un ensemble non vide, une fonction k est dite fonction noyau s'il existe un $\mathbb{R}$-espace de Hilbert F et une transformation $\phi : E \to F$ telle que [1] :

$$\begin{aligned} k : E \times E &\longrightarrow \mathbb{R} \\ (x, y) &\longmapsto k(x, y) = < \phi(x), \phi(y) >_F, \end{aligned}$$

F est appelé l'espace des caractéristiques. Dans notre cas $E = \mathbb{R}^d$.
Le théorème suivant permet de caractériser les fonctions noyaux sans passer explicitement par l'espace caractéristique.

Théorème de Mercer

Une fonction $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ est un noyau si et seulement si k est symétrique et semi-définie positive [1].

Exemples de noyaux

- **Noyau polynomial** [1] : Soient $m \in \mathbb{N}$, $d \in \mathbb{N}^*$ et $c \geq 0$. Alors la fonction k définie par :

$$k(x, y) = (x^T y + c)^m, \text{ pour tout } x, y \ \in \mathbb{R}^d.$$

est un noyau polynomial de degré m sur $\mathbb{R}^d$.

- **Noyau RBF** [1] : Pour $d \in \mathbb{N}$, $\sigma > 0$, $x \in \mathbb{R}^d$ et $y \in \mathbb{R}^d$, le noyau RBF sur $\mathbb{R}^d$ est définie par :

$$k(x, y) = e^{-\frac{||x-y||^2}{\sigma^2}},$$

où $\|.\|$ représente la norme 2 (distance euclidienne) dans l'espace $\mathbb{R}^d$.
Le noyau RBF est appelé aussi le noyau Gaussien.

3 Notations

La matrice de variance de X est donnée par :

$$\mathrm{C} = \frac{1}{n}\sum_{j=1}^{n} \mathrm{X}_{:j}\mathrm{X}_{:j}^{\mathrm{T}}.$$

Supposons que l'espace $\mathbb{R}^d$ soit mappé dans un espace de Hilbert F au moyen d'une fonction caractéristique ϕ [4] :

$$\begin{aligned} \phi : \mathbb{R}^d &\longrightarrow \mathrm{F} \\ \mathrm{X}_{:j} &\longmapsto \phi(\mathrm{X}_{:j}). \end{aligned}$$

La matrice de variance de X dans l'espace des caractéristiques F est donnée par [4] :

$$\mathrm{Q} = \frac{1}{n}\sum_{j=1}^{n} \phi(\mathrm{X}_{:j})\phi^{\mathrm{T}}(\mathrm{X}_{:j}).$$

Nous supposons que les données sont centrées en F. Cependant, la façon de centrer les données dans l'espace F est donnée dans la référence [4].
La matrice de variance des moyennes de classes dans l'espace F est donnée par [4] :

$$\mathrm{B} = \frac{1}{n}\sum_{l=0}^{1} n_l \overline{\phi_l}\ \overline{\phi_l^{\mathrm{T}}}, \tag{2.1}$$

où B représente l'inertie inter-classes et $\overline{\phi_l}$ est la moyenne de la classe l dans l'espace F pour $l = 0, 1$, tel que :

$$\overline{\phi_l} = \frac{1}{n_l}\sum_{k=1}^{n_l} \phi(\mathrm{X}_{:k}^{(l)}). \tag{2.2}$$

La somme des variances des données (X_0 et X_1) dans l'espace des caractéristiques F est donnée par [4] :

$$\mathrm{V} = \frac{1}{n}\sum_{l=0}^{1}\sum_{k=1}^{n_l} \phi(\mathrm{X}_{:k}^{(l)})\phi^{\mathrm{T}}(\mathrm{X}_{:k}^{(l)}), \tag{2.3}$$

la matrice V représente l'inertie intra-classe.
Afin de pouvoir généraliser ADF au cas non linéaire, une reformulation basée sur le produit scalaire est nécessaire. Par conséquent, nous considérons une expression de produit scalaire sur l'espace F donnée par la fonction de noyau suivante [4] :

$$k(\mathrm{X}_{:i}, \mathrm{X}_{:j}) = k_{ij} = \phi^{\mathrm{T}}(\mathrm{X}_{:i})\phi(\mathrm{X}_{:j}).$$

Pour des classes p et q données, on exprime cette fonction noyau par :

$$(k_{ij})_{pq} = \phi^{\mathrm{T}}(\mathrm{X}_{:i}^{(p)})\phi(\mathrm{X}_{:j}^{(q)}).$$

Soit K une matrice $(n \times n)$ définie sur les éléments de classe par $(\mathrm{K}_{pq})_{p=0,1;\ q=0,1}$, où (K_{pq}) est une matrice composée de produit scalaire dans l'espace des caractéristiques F :

$$\mathrm{K} = (\mathrm{K}_{pq})_{p=0,1;\ q=0,1} \text{ où } \mathrm{K}_{pq} = (k_{ij})_{i=1,\ldots,n_p;\ j=1,\ldots,n_q},$$

K_{pq} est une matrice $(n_p \times n_q)$ et K est une matrice $(n \times n)$ symétrique.
Nous présentons également la matrice :

$$\mathrm{W} = (\mathrm{W}_l)_{l=0,1}, \tag{2.4}$$

où W_l est une matrice carré $(n_l \times n_l)$, d'éléments égaux à : $\frac{1}{n_l}$. W est une matrice carré $(n \times n)$.

4 Principes de l'Analyse Discriminante Généralisée

L'idée de base de l'Analyse Discriminante Généralisée est de résoudre le problème de l'Analyse Discriminante de Fisher (1.12) dans l'espace F. En utilisant les fonctions noyau, nous généralisons ADF. Le but de GDA est donc de maximiser l'inertie inter-classes et minimiser l'inertie intra-classes [4, 16] :

$$\max_{v} \frac{v^{\mathrm{T}}\mathrm{B}v}{v^{\mathrm{T}}\mathrm{V}v}. \tag{2.5}$$

Pour maximiser le quotient de (2.5), il faut que la dérivée relative à v disparaisse [4] :

$$\begin{aligned}
\frac{(v^{\mathrm{T}}\mathrm{V}v)(2\mathrm{B}v) - (v^{\mathrm{T}}\mathrm{B}v)(2\mathrm{V}v)}{(v^{\mathrm{T}}\mathrm{V}v)^2} = 0 &\Longrightarrow (v^{\mathrm{T}}\mathrm{V}v)(2\mathrm{B}v) - (v^{\mathrm{T}}\mathrm{B}v)(2\mathrm{V}v) = 0, \\
&\Longrightarrow (v^{\mathrm{T}}\mathrm{V}v)(\mathrm{B}v) = (v^{\mathrm{T}}\mathrm{B}v)(\mathrm{V}v), \\
&\Longrightarrow \mathrm{B}v = \frac{v^{\mathrm{T}}\mathrm{B}v}{v^{\mathrm{T}}\mathrm{V}v}\mathrm{V}v.
\end{aligned} \tag{2.6}$$

Posons,

$$\lambda = \frac{v^{\mathrm{T}}\mathrm{B}v}{v^{\mathrm{T}}\mathrm{V}v}, \tag{2.7}$$

nous obtiendrons :

$$\mathrm{B}v = \lambda \mathrm{V}v, \tag{2.8}$$

v est le vecteur propre de $\mathrm{B}^{-1}\mathrm{V}$ associée à la plus grande valeur propre λ.
Puisque les vecteurs propres v sont des combinaisons linéaires des éléments de F, alors il existe des coefficients α_{pq} ($p = 0, 1$ et $q = \overline{1, n_p}$) tels que :

$$v = \sum_{p=0}^{1}\sum_{q=1}^{n_p} \alpha_{pq}\phi(\mathrm{X}_{:q}^{(p)}). \tag{2.9}$$

Le quotient de (2.7) est équivalent au quotient suivant (voir référence [4]) :

$$\lambda = \frac{\alpha^{\mathrm{T}}\mathrm{KWK}\alpha}{\alpha^{\mathrm{T}}\mathrm{KK}\alpha}. \tag{2.10}$$

Le but est de résoudre le système de vecteur propre (2.10), qui nécessite une décomposition algébrique de la matrice K.

5 La résolution en valeur propre

Nous utilisons la décomposition en valeurs propres de la matrice K :

$$\mathrm{K} = \mathrm{P}\Gamma\mathrm{P}^{\mathrm{T}}, \tag{2.11}$$

où Γ la matrice diagonale contenant les valeurs propres non nulles de K et P la matrice composée des vecteurs propres. Ainsi Γ^{-1} existe. P est une matrice orthonormale ($\mathrm{P}^{\mathrm{T}}\mathrm{P} = \mathrm{I}_d$).
En remplaçant K dans l'équation (2.10), on obtient [4] :

$$\lambda = \frac{(\Gamma\mathrm{P}^{\mathrm{T}}\alpha)^{\mathrm{T}}\mathrm{P}^{\mathrm{T}}\mathrm{W}\mathrm{P}(\Gamma\mathrm{P}^{\mathrm{T}}\alpha)}{(\Gamma\mathrm{P}^{\mathrm{T}}\alpha)^{\mathrm{T}}\mathrm{P}^{\mathrm{T}}\mathrm{P}(\Gamma\mathrm{P}^{\mathrm{T}}\alpha)}. \tag{2.12}$$

Posons

$$\beta = \Gamma\mathrm{P}^{\mathrm{T}}\alpha. \tag{2.13}$$

En remplaçant cette dernière formule dans l'équation (2.12), on obtient :

$$\lambda = \frac{\beta^{\mathrm{T}}\mathrm{P}^{\mathrm{T}}\mathrm{W}\mathrm{P}\beta}{\beta^{\mathrm{T}}\mathrm{P}^{\mathrm{T}}\mathrm{P}\beta}. \tag{2.14}$$

Nous obtenons donc :

$$\lambda\mathrm{P}^{\mathrm{T}}\mathrm{P}\beta = \mathrm{P}^{\mathrm{T}}\mathrm{W}\mathrm{P}\beta. \tag{2.15}$$

Comme P est orthonormale, cette dernière équation peut être simplifiée et donne (2.16), pour laquelle des solutions doivent être trouvées en maximisant λ :

$$\lambda\beta = \mathrm{P}^{\mathrm{T}}\mathrm{W}\mathrm{P}\beta, \tag{2.16}$$

β est le vecteur propre de $\mathrm{P}^{\mathrm{T}}\mathrm{W}\mathrm{P}$ associée à la plus grande valeur propre λ. Pour un vecteur β donné, il existe au moins un α satisfaisant (2.13) sous la forme :

$$\alpha = \mathrm{P}\Gamma^{-1}\beta. \tag{2.17}$$

Les coefficients α sont normalisés avec la normalisation des vecteurs propres v en F :

$$v^{\mathrm{T}}v = 1.$$

En utilisant l'équation (2.9) :

$$\begin{aligned} v^{\mathrm{T}}v &= \sum_{p=0}^{1}\sum_{q=1}^{n_p}\sum_{l=0}^{1}\sum_{h=1}^{n_l}\alpha_{pq}\alpha_{lh}\phi^{\mathrm{T}}(\mathrm{X}_{:q}^{(p)})\phi(\mathrm{X}_{:h}^{(l)}) = 1, \\ v^{\mathrm{T}}v &= \sum_{p=0}^{1}\sum_{l=0}^{1}\alpha_p^{\mathrm{T}}\mathrm{K}_{pl}\alpha_l = 1, \\ v^{\mathrm{T}}v &= \alpha^{\mathrm{T}}\mathrm{K}\alpha = 1. \end{aligned} \tag{2.18}$$

Les coefficients α sont divisés par $\sqrt{\alpha^{\mathrm{T}}\mathrm{K}\alpha}$ afin d'obtenir des vecteurs normalisés v.
La projection de x sur v est :

$$v^{\mathrm{T}}\phi(x) = \sum_{p=0}^{1}\sum_{q=1}^{n_p}\alpha_{pq}\mathrm{K}(\mathrm{X}_{:q}^{(p)}, x). \tag{2.19}$$

Algorithm 2.1 Analyse Discriminante Généralisée (GDA)

Données : La matrice de données X_0 de la classe 0 et la matrice de données X_1 de la classe 1.
Résultat : La projection des données.

1: Calculer K et W ;
2: Calculer Γ et P ;
3: Calculer β, le vecteur propre de $P^T W P$;
4: Calculer $\alpha = P\Gamma^{-1}\beta$;
5: Calculer v ;
6: Calculer la projection des données.

L'algorithme (2.1), offre une projection des données sur une droite permettant de maximiser leur séparation et génère un seuil de classification Ψ. Nous pouvons calculer le seuil Ψ comme suit :
Pour un point x donnée (à classifier),

1. Nous calculons $Z = v^T \phi(x)$, la projection de x sur v.
2. Nous calculons la moyenne des données projetées de chaque classe, soit μ_0 la moyenne des données projetées de la classe 0 et soit μ_1 la moyenne des données projetées de la classe 1.
3. Nous calculons les distances suivantes :
   $d_0 = d(\mu_0, Z)$: la distance entre la moyenne des données projetées de la classe 0 et la projection de x sur v.
   $d_1 = d(\mu_1, Z)$: la distance entre la moyenne des données projetées de la classe 1 et la projection de x sur v.
4. Nous calculons le seuil de la classification :

$$\Psi = \min(d_0, d_1). \tag{2.20}$$

Chapitre 3

Détection faciale

1 Introduction

L'une des applications les plus importantes de la classification est la reconnaissance d'objets. Dans notre étude, nous nous intéressons principalement à la détection faciale. Dans sa définition exacte, un système de détection faciale est un programme capable de reconnaitre dans une image quelconque de façon indépendante les visages humains présents sur cette image. Ce processus présente plusieurs aspects dont le traitement d'image et la classification. Nous allons dans un premier temps introduire une méthode d'extraction des caractéristiques utilisant les filtres de Gabor, ensuite nous allons détailler le processus d'obtention du vecteur de données pour la classification.

2 Traitement d'image

Le traitement d'images est l'ensemble des techniques permettant de modifier une image dans le but de l'améliorer ou d'en extraire des informations.

2.1 Image numérique

Une image numérique est représentée par un tableau, chaque case de ce tableau stock une valeur, se nomme un pixel [12].
Il existe trois types d'images :

1. Les images en noir et blanc : si chaque pixel prend deux valeurs 0 ou 1, 0 pour le noir et 1 pour le blanc ;
2. Les images en niveau de gris : si chaque pixel prend des valeurs entre 0 et 255, 0 pour le noir, 255 pour le blanc et entre 0 et 255 pour des niveaux de gris allant du noir au blanc ;
3. Les images en couleurs (RVB) : si chaque pixel prend trois valeurs, soient R la valeur du rouge, V la valeur du vert et B la valeur du blue.

2.2 Filtrage d'une image

Le principe du filtrage est de modifier les valeurs des pixels d'une image, généralement dans le but d'améliorer son aspect. En pratique, il s'agit de créer une nouvelle image en se servant des valeurs des pixels de l'image d'origine. L'un des buts du filtrage est d'éliminer plus de bruit possible tout en préservant le maximum d'informations. La

figure (3.1) montre une application du filtre médian sur une image (de gauche) bruité (bruit poivre et sel) qui résulte en une image nette (image de droite).

FIGURE 3.1 – Application d'un filtre médian.

Le filtre médian est utilisé pour la réduction de bruits [12], la fonction MATLAB utilisée pour ce filtre est "**medfilt2**".
Comme il existe des filtres pour la réduction de bruits, il existe aussi des filtres qui rajoutent le bruit sur une image, par exemple le bruit poivre et sel, le flou,... etc. La figure (3.2) illustre une application d'un bruit poivre et sel sur l'image de gauche donnant l'image de droite.

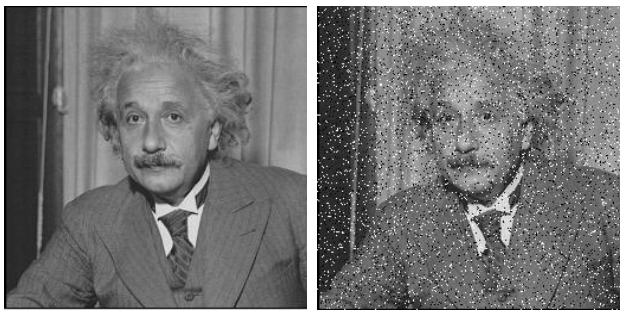

FIGURE 3.2 – Application d'un bruit poivre et sel.

Le bruit poivre et sel est obtenu en ajoutant des pixels noirs et des pixels blancs aléatoirement dans une image [12]. La fonction MATLAB utilisée pour ajouter le bruit à une image est "**imnoise**".

2.3 Filtrage par convolution

Le filtrage par convolution se pratique couramment sur les images numériques, par exemple pour rendre les images légèrement floues afin de réduire le bruit, ou bien au contraire pour accentuer les détails.

La convolution

La convolution est liée à la notion du filtrage et le traitement du signal, lorsque l'on utilise des filtres.
En traitement d'image numérique, la convolution est l'opérateur de base du traitement linéaire des images. C'est le remplacement de la valeur d'un pixel par une combinaison linéaire de ses voisins. Elle consiste à faire balayer une fenêtre (masque) sur l'ensemble

des points de l'image [2].
Sa formule est définie par :

$$(f * g)(x, y) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x-u, y-v) g(u, v) du \; dv, \tag{3.1}$$

et dans le cas discret :

$$(f * g)(x, y) = \sum_{u=-\infty}^{+\infty} \sum_{v=-\infty}^{+\infty} f(x-u, y-v) g(u, v), \tag{3.2}$$

g : masque de convolution. La figure (3.3) montre un exemple de convolution entre deux images.

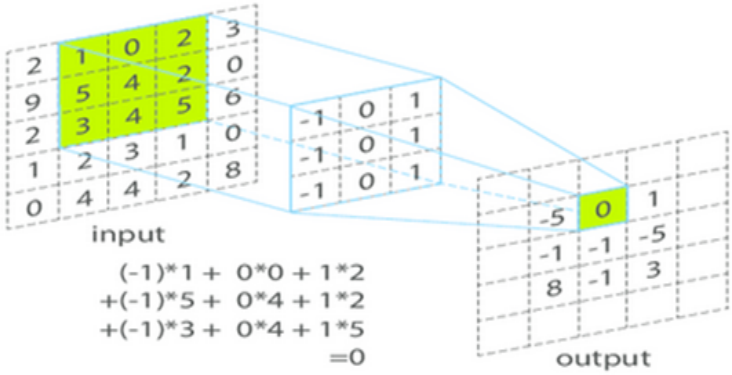

FIGURE 3.3 – Exemple de convolution.

La convolution entre une image et un filtre est appelé aussi masque de convolution.

Transformée de Fourier

Soit une fonction f définie sur $\mathbb{R}^2$, avec $f \in L^1(\mathbb{R}^2)$. On appelle la transformée de Fourier de f, la fonction notée TF, définie sur $\mathbb{R}^2$ par

$$\mathrm{TF}(f(x, y)) = \tilde{f}(u, v) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x, y) e^{-2\pi i(ux+vy)} du dv. \tag{3.3}$$

Théorème de Plancherel

La transformée de Fourier du produit de convolution de deux signaux $f(x, y)$ et $g(x, y)$ est égale au produit des transformées de Fourier de ces deux signaux [6].

$$\mathrm{TF}((f * g)(x, y)) = \mathrm{TF}(f(x, y)) \; \mathrm{TF}(g(x, y)). \tag{3.4}$$

Il est possible de calculer le produit de convolution de deux signaux, en calculant l'inverse de la transformée de Fourier.

$$(f * g)(x, y) = \mathrm{TF}^{-1}(\mathrm{TF}(f(x, y)) \; \mathrm{TF}(g(x, y))). \tag{3.5}$$

Le calcul de produit de convolution par transformée de Fourier au lieu de calculer directement le produit de convolution nous permet de réduire le temps de calcul.

Convolution entre deux images

Dans le domaine du traitement d'images, la convolution 2D est un outil puissant en traitement d'images, il s'applique sur une image à travers un noyau pour améliorer différentes caractéristiques telles que le contraste, la netteté...etc. Le principe est basé sur des calculs purement mathématiques et matriciels.
Le produit de convolution est une opération entre deux images en niveau de gris f et g, noté $f * g$ défini dans l'expression (3.2).
La figure (3.4) montre un produit de convolution d'une image (matrice gauche) avec noyau (matrice centre) fournit une image filtrée (matrice droite).

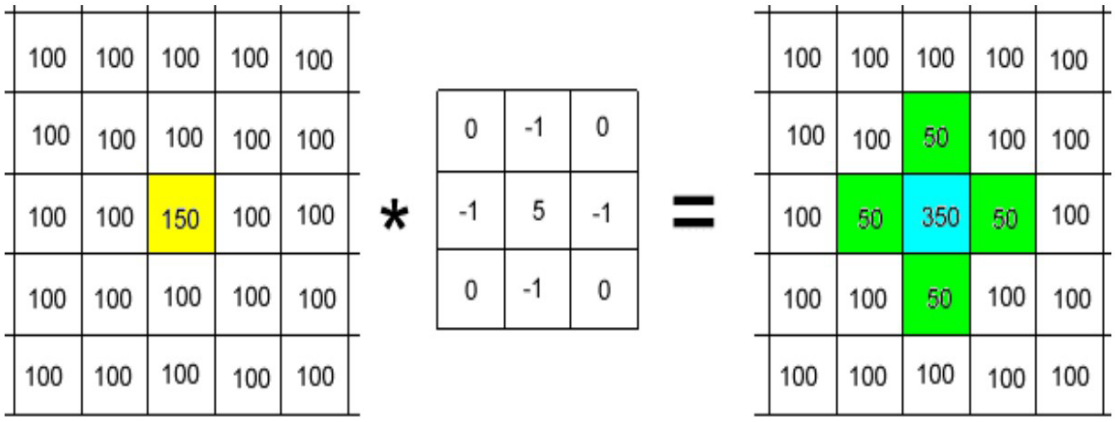

FIGURE 3.4 – Convolution d'une image avec un noyau.

Sur MATLAB la fonction utilisée pour le produit de convolution 2D est "**conv2**".
Le filtrage d'une image peut se pratiquer dans deux domaines différents : spatial ou fréquentiel (voir la figure 3.5)

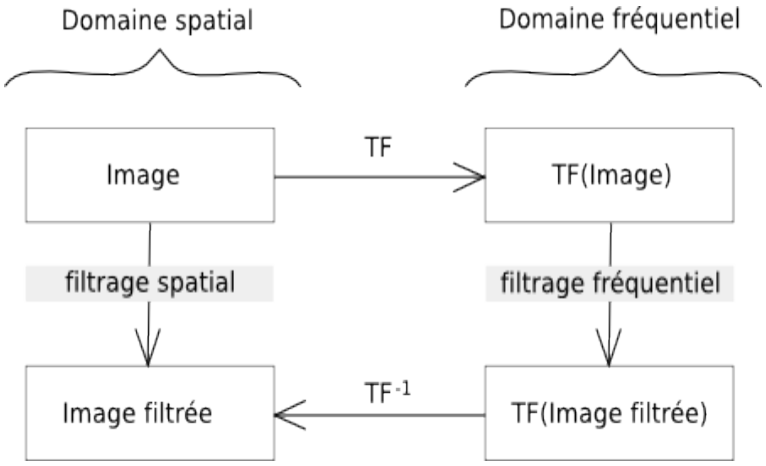

FIGURE 3.5 – Le filtrage dans le domaine fréquentiel et spatial.

Le filtrage dans le domaine spatial consiste à multiplier (convolution) le signal (image) par le filtre, la figure (3.6) représente le filtrage dans le domaine spatial.

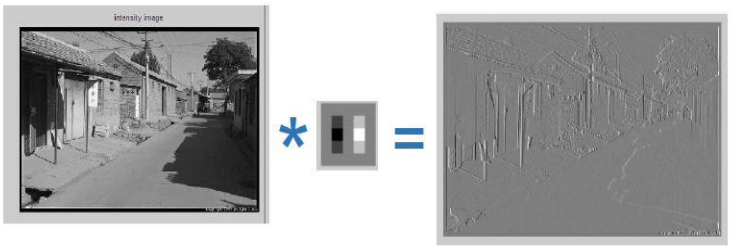

FIGURE 3.6 – Le filtrage dans le domaine spatial.

Et le filtrage dans le domaine fréquentiel consiste à multiplier la transformée de Fourier du signal (image) par la transformée de Fourier du filtre, après on applique l'inverse de la transformée de Fourier (voir 2.3), la figure (3.7) représente le filtrage dans le domaine fréquentiel.

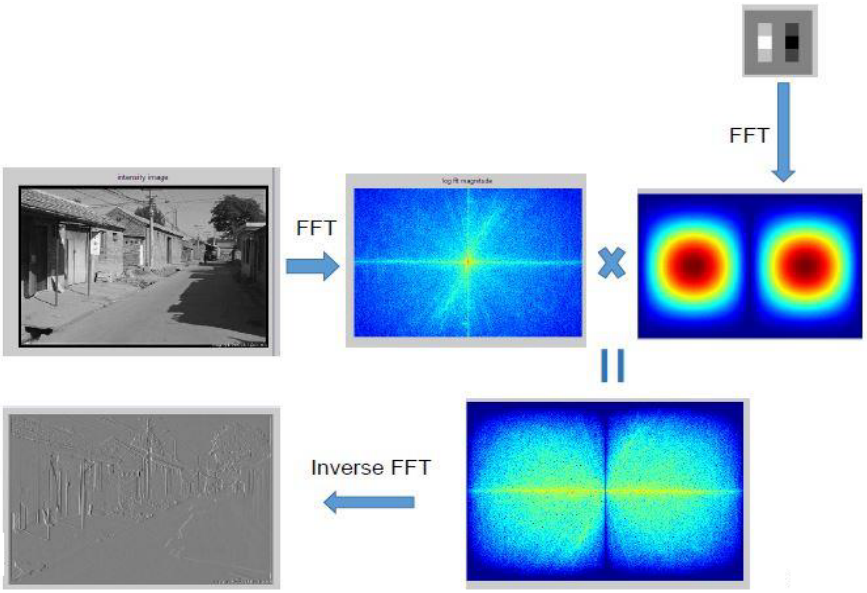

FIGURE 3.7 – Le filtrage dans le domaine fréquentiel.

Il existe des filtres linéaires et non-linéaires, mais nous nous intéressons à des filtres linéaires.

2.4 Filtre linéaire

Un filtre linéaire transforme un ensemble de données d'entrée en un ensemble de données de sortie selon une opération mathématique linéaire nommée convolution.
Le filtrage linéaire sera effectué en convoluant un filtre exprimé dans le domaine spatial à l'image. Parmi les filtres linéaires nous avons le filtre gaussien, le filtre moyenneur et le filtre de Gabor,..., etc.

Filtre moyenneur

Le filtre moyenneur est utilisé pour réduire le bruit dans une image et/ou flouter une image [12]. La figure (3.8) montre une application d'un filtre moyenneur sur l'image de gauche donne l'image de droite.

FIGURE 3.8 – Application d'un filtre moyenneur sur une image.

Ce filtre remplace chaque pixel par la moyenne des valeurs des pixels adjacents.

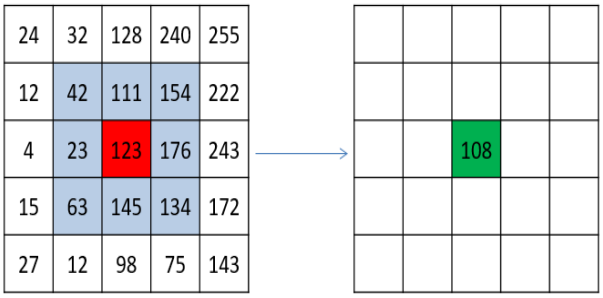

FIGURE 3.9 – Filtre moyenneur.

Avec un filtre moyenneur de largeur 3, pour calculer la nouvelle valeur du pixel rouge de l'image originale de gauche, nous calculons la valeur moyenne des pixels situés dans un carré de dimension 3×3 centré sur ce pixel. Cela donne la nouvelle valeur du pixel sur l'image transformée (pixel vert sur l'image de droite) :

$$\frac{42+111+154+23+123+176+63+145+134}{9} = 108.$$

Filtre Gaussien

Ce type de filtre est utilisé pour diminuer le bruit ou appliquer un flou sur une image. Le Filtre Gaussien est donnée par la formule suivante [12] :

$$g(x,y,\sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{x^2+y^2}{2\sigma^2}}. \tag{3.6}$$

σ représente la déviation standard de l'enveloppe Gaussienne (elle permet de contrôler l'ampleur de la cloche Gaussienne).
x représente la distance entre l'origine et laxe (xx').
y représente la distance entre l'origine et laxe (yy').
La figure (3.10) montre une application d'un filtre Gaussien sur l'image de gauche donne l'image de droite.

FIGURE 3.10 – Application d'un filtre Gaussien sur une image.

2.5 Filtres de Gabor

Un filtre du Gabor, du nom physicien anglais d'origine hongroise Dennis Gabor, est un filtre linéaire. Dans le domaine spatial, un filtre Gabor est un noyau gaussien modulé par une onde sinusoïde [11].

$$g(x, y) = s(x, y)\, w_r(x, y), \tag{3.7}$$

où $s(x, y)$ est une sinusoïde complexe, connue sous le nom de support, et $w_r(x, y)$ est une fonction gaussienne en forme de 2-D, connue sous le nom d'enveloppe.

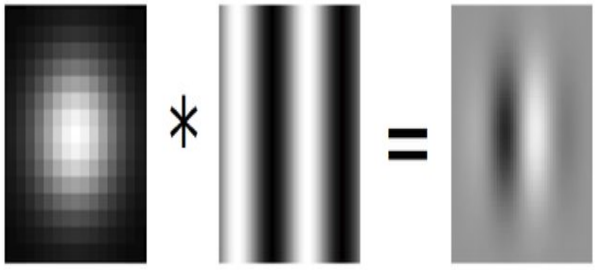

FIGURE 3.11 – Gaussiene × Sinusoïde = Gabor.

- **La sinusoïde complexe :** la fonction $s(x, y)$ est définie par :

$$s(x, y) = e^{j(2\pi(u_0 x + v_0 y) + \mathrm{P})}, \tag{3.8}$$

où les paramètres (u_0, v_0) et P représentent la fréquence spatiale et la phase du sinusoïde respectivement.
Nous pouvons considérer cette sinusoïde comme deux fonctions réelles distinctes, distribué dans la partie réelle et imaginaire d'une fonction complexe (voir la figure 3.12).

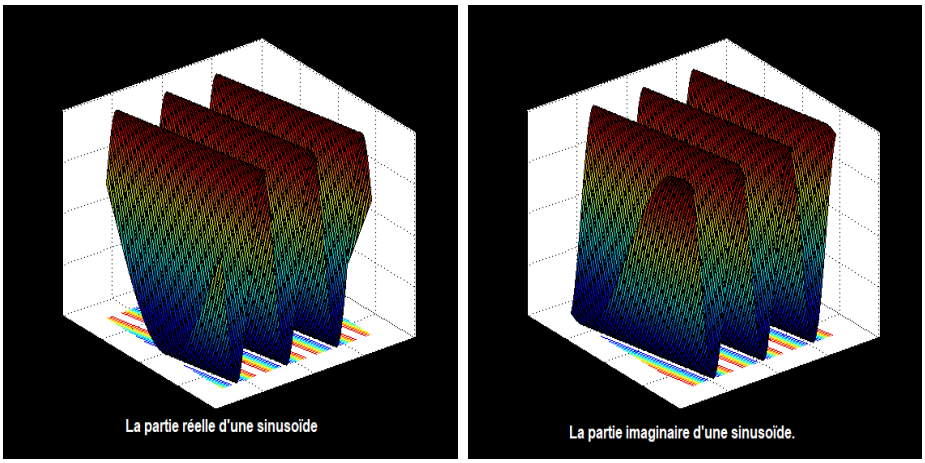

FIGURE 3.12 – La partie réelle et imaginaire d'une sinusoïde.

Les équations de la partie réelle et la partie imaginaire de la sinusoïde sont :

$$\mathrm{Re}(s(x,y)) = \cos(2\pi(u_0 x + v_0 y) + \mathrm{P})$$
$$\mathrm{Im}(s(x,y)) = \sin(2\pi(u_0 x + v_0 y) + \mathrm{P})$$

Les paramètres u_0 et v_0 définissent la fréquence spatiale de la sinusoïde en coordonnées cartésiennes. Cette fréquence spatiale peut également être exprimée en coordonnées polaires comme magnitude F_0 et direction w_0 :

$$\mathrm{F}_0 = \sqrt{u_0^2 + v_0^2}$$
$$w_0 = \tan^{-1}(\frac{v_0}{u_0})$$

c'est-à-dire

$$u_0 = \mathrm{F}_0 \cos w_0$$
$$v_0 = \mathrm{F}_0 \sin w_0$$

Donc, la sinusoïde complexe devient

$$s(x,y) = e^{j(2\pi \mathrm{F}_0(x\cos w_0 + y \sin w_0) + \mathrm{P})}. \tag{3.9}$$

• **L'enveloppe Gaussienne :** la fonction $w_r(x,y)$ est définie par :

$$w_r(x,y) = k e^{-\pi(a^2(x-x_0)_r^2 + b^2(y-y_0)_r^2))}, \tag{3.10}$$

où k l'amplitude, (x_0, y_0) les sommets de la fonction, a, b sont des paramètres de la Gaussienne, et l'indice r représente une opération de rotation, telle que :

$$(x-x_0)_r = (x-x_0)\cos\theta + (y-y_0)\sin\theta,$$
$$(y-y_0)_r = (x-x_0)\sin\theta + (y-y_0)\cos\theta.$$

Transformée de Fourier d'un Filtre de Gabor 2D

La transformée de Fourier pour un filtre de Gabor 2D est donnée par la formule suivante [11] :

$$\tilde{g}(u,v) = \frac{k}{ab} e^{j(-2\pi(x_0(u-u_0)+y_0(v-v_0))+\mathrm{P})} e^{-\pi(\frac{(u-u_0)^2}{a^2} + \frac{(v-v_0)^2}{b^2})}. \tag{3.11}$$

3 Filtre de Gabor 2D pour la détection de visage

La méthode de reconnaissance que nous avons adoptée repose, principalement, sur l'extraction des caractéristiques du visage par les filtres de Gabor 2D.

3.1 Filtre de Gabor 2D

Un filtre de Gabor 2D est une fonction à noyau Gaussienne modulé par une onde plane sinusoïde orientée selon un angle [16] :

$$\varphi_{\pi(f,\theta,\gamma,\eta)}(x,y) = \frac{f^2}{\pi\gamma\eta} e^{(-(\alpha^2 x'^2 + \beta^2 y'^2))} e^{(j2\pi f x')}, \tag{3.12}$$

où :

$$\begin{aligned} x' &= x\cos\theta + y\sin\theta, \\ y' &= -x\sin\theta + y\cos\theta. \end{aligned}$$

avec, f la fréquence de l'onde plane sinusoïde, θ l'angle de rotation de la Gaussienne et l'onde plane, α la netteté, le long du grand axe gaussien (parallèle à l'onde), β la netteté, le long du petit axe (perpendiculaire à l'onde), $\gamma = \frac{f}{\alpha}$ et $\eta = \frac{f}{\beta}$ sont définis pour conserver le rapport entre la fréquence et la netteté constante.
Chaque filtre se présente sous la forme d'ondes planes de fréquence f, limité par une fonction d'enveloppe Gaussienne avec la largeur relative à α et β. Pour extraire des caractéristiques utiles d'une image (visage), il faut normalement une banque de filtres Gabor avec des fréquences et des orientations différentes :

$$\varphi_{uv} = \varphi_{\pi(f_u,\theta_v,\gamma,\eta)}, \tag{3.13}$$

$$\begin{aligned} f_u &= \frac{f_{max}}{\sqrt{2}^u}, \\ \theta_v &= \frac{v}{8}\pi, \\ u &= 0,...,U-1, \\ v &= 0,...,V-1. \end{aligned} \tag{3.14}$$

L'expression de $\varphi_{\pi(f_u,\theta_v,\gamma,\eta)}$ dans (3.13) est obtenue en remplaçant f par f_u et θ par θ_v.

3.2 Filtres Gabor pour l'extraction des traits du visage

Pour la reconnaissance faciale nous choisissons $f_{max} = 0.25$. Les paramètres γ et η déterminent le rapport entre la fréquence centrale et la taille de l'enveloppe Gaussienne. Nous supposons $\alpha = \beta$ et $\gamma = \eta = \sqrt{2}$ (à la même hypothèse que l'on trouve dans la littérature scientifique). Comme il n'existe pas de base théorique pour le choix des échelles et des orientations, plusieurs travaux ont été effectués pour évaluer les performances sous différentes paramètres. Les résultats montrent que 5 échelles et 8 orientations doivent être utilisées à des fins de reconnaissance faciale (voir la figure (3.13)) [16].

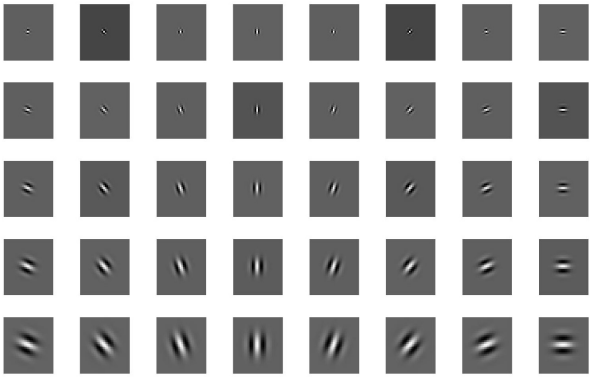

FIGURE 3.13 – Filtres de Gabor

3.3 La représentation des visages

La représentation de Gabor d'une image de visage $I(x, y)$ est obtenue par la convolution de l'image avec la famille des filtres de Gabor, définie par [16] :

$$G_{u,v}(x) = I(x, y) * \varphi_{u,v}, \tag{3.15}$$

où $G_{u,v}(x)$ représente le résultat de la convolution de l'image par le filtre de Gabor à l'orientation u et l'échelle v. La figure (3.14) montre le résultat de la convolution d'une image de visage avec deux filtres de Gabor à différentes orientations et échelles.

FIGURE 3.14 – Résultat de convolution d'une image visage avec deux filtres de Gabor.

4 Principe de la détection faciale

La reconnaissance faciale comporte trois étapes principales. La première étape explique le prétraitement effectué pour préparer les imagettes. La deuxième étape est l'extraction des caractéristiques pour extraire les informations les plus pertinentes de l'imagette en utilisant les filtres de Gabor. Et la dernière étape est la classification. La figure (3.15) montre un résumé général du système de la reconnaissance faciale [5].

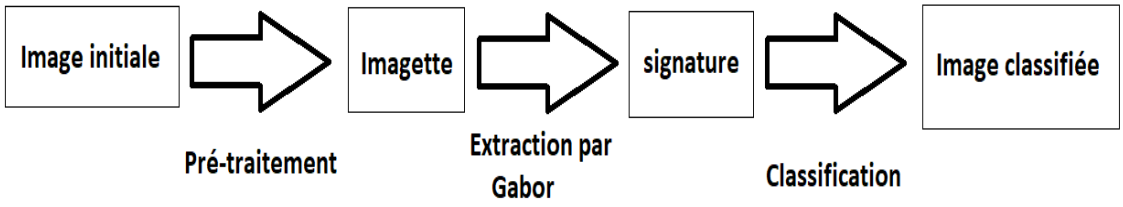

FIGURE 3.15 – Processus de classification des visages.

La figure (3.16) représente l'exemple utilisé dans ce chapitre (extraite de [18]).

FIGURE 3.16 – Exemple d'une image sur laquelle on cherche à détecter les visages.

5 Prétraitement

Dans le contexte de la détection faciale, le point de départ est de faire un prétraitement qui permet d'extraire depuis une image un ensemble d'imagettes. À partir de ces imagettes, il faut extraire les caractéristiques que l'on appelle signature pour faire la différence entre un visage et un non-visage [5]. La figure (3.17) illustre quelques exemples d'imagettes extraites de notre image originale (3.16).

FIGURE 3.17 – Exemples de résultats du prétraitement

Pour la détection faciale, cette étape est essentielle parce qu'elle permet de classer les données sous la forme la plus appropriée pour l'algorithme. Dans les expériences numériques, nous utilisons une application de la détection de visage qui extrait les caractéristiques par filtres de Gabor à travers l'implémentation MATLAB faite par Omid Sakhi [18].
Nous employons un noyau en tant qu'image (en particulier un visage), qui sera appliquée dans la convolution (le processus est précisé dans la figure (3.18)).

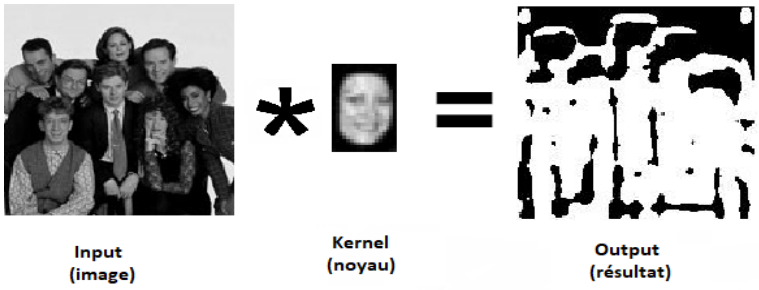

FIGURE 3.18 – Produit de convolution avec un noyau visage.

Ensuite, une fois que l'amélioration a été faite sur l'image, le prétraitement doit détecter les parties de l'image qui peuvent représenter un visage, puis les extraire sous forme d'imagettes. Pour ce faire, l'algorithme doit rechercher les points de forte intensité. Pour un bon fonctionnement, il est important que toutes les imagettes soient de taille 27×18. Dans MATLAB, la fonction utilisée est "**imregionalmax**" qui retourne le maximum d'une certaine région.

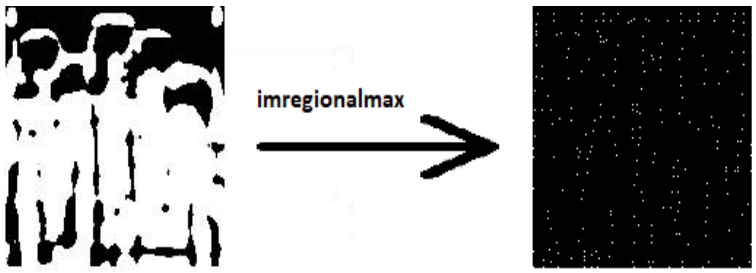

FIGURE 3.19 – Les points de forte intensité.

6 Extraction par filtres de Gabor

L'extraction consiste à extraire les informations les plus pertinentes afin de faire notre classification. Pour une image, on peut la transformer en un vecteur potentiellement de grande dimension, mais qui doit nous donner des informations importantes peuvent nous aider à faire notre classification. Dans le domaine du traitement d'image, il est possible de le faire par des filtres ou des ondelettes. Dans notre travail, nous employons l'extraction avec les filtres de Gabor. La figure (3.20) montre une application d'un filtre de Gabor sur une image (visage).

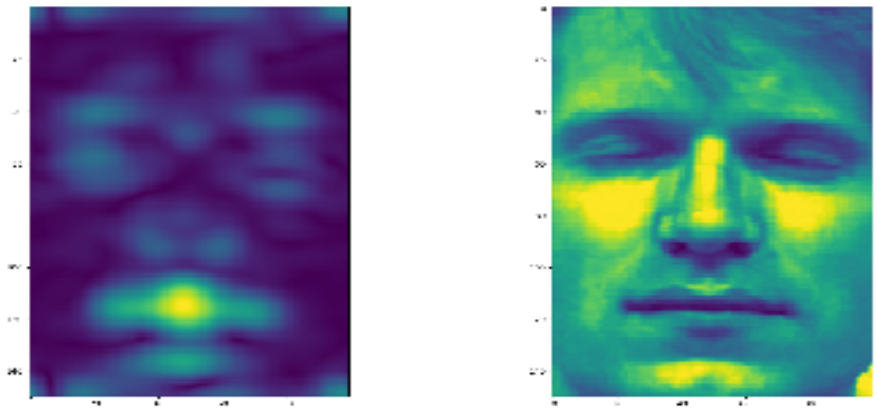

FIGURE 3.20 – Application d'un filtre de Gabor à un visage par convolution.

Il s'agit d'abord de générer une banque de filtres de Gabor, dans notre exemple nous utilisons **40** filtres de Gabor, résultant de 5 échelles différentes et 8 angles employés dans la paramétrisation de la fonction de Gabor [17]. La figure (3.21) montre les 40 filtres de Gabor utilisés dans l'extraction des caractéristiques.

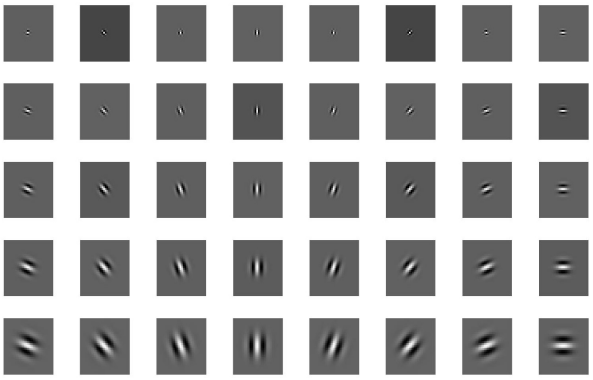

FIGURE 3.21 – Filtres de Gabor

L'étape suivante consiste à appliquer le filtre à l'imagette (résultante du prétraitement) que l'on souhaite classifier. Pour cela nous appliquons le produit des transformées de Fourier entre l'image et la banque de Filtres de Gabor (voir (3.4)), sur MATLAB la fonction utilisée est "**fft2**". Comme le produit de convolution est très "couteux", nous appliquons l'inverse de transformée de Fourier (voir (3.5)), nous utilisons la fonction MATLAB "**ifft2**" pour calculer l'inverse de la transformée de Fourier. L'image résultante est nommée une signature qui est de taille 135×144. La figure (3.22) illustre le processus.

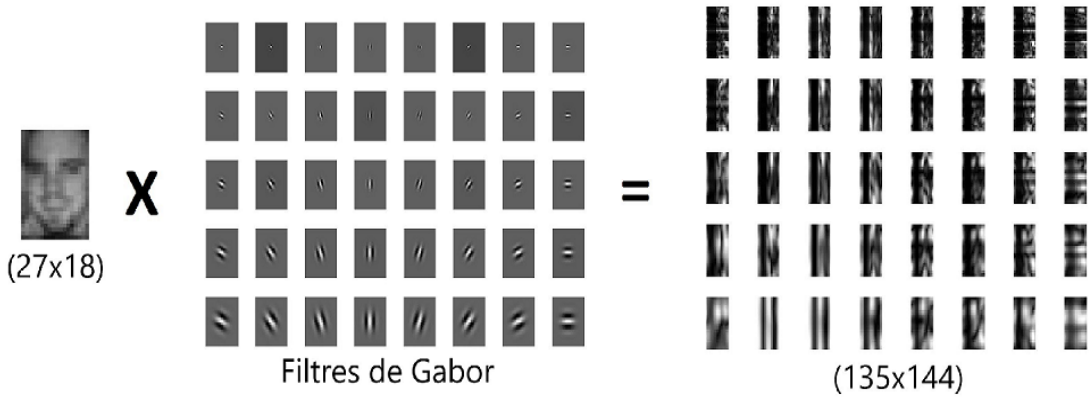

FIGURE 3.22 – Application des filtres de Gabor sur une imagette.

Lorsque cette partie est terminée, tout ce qui reste est de vectoriser l'image résultante (135×144) à un vecteur de dimension 19440 et classifier ou entrainer le classifieur sur les données numériques qui est la dernière étape de la détection faciale. La figure (3.23) récapitule les étapes de l'extraction des caractéristiques par les filtres de Gabor.

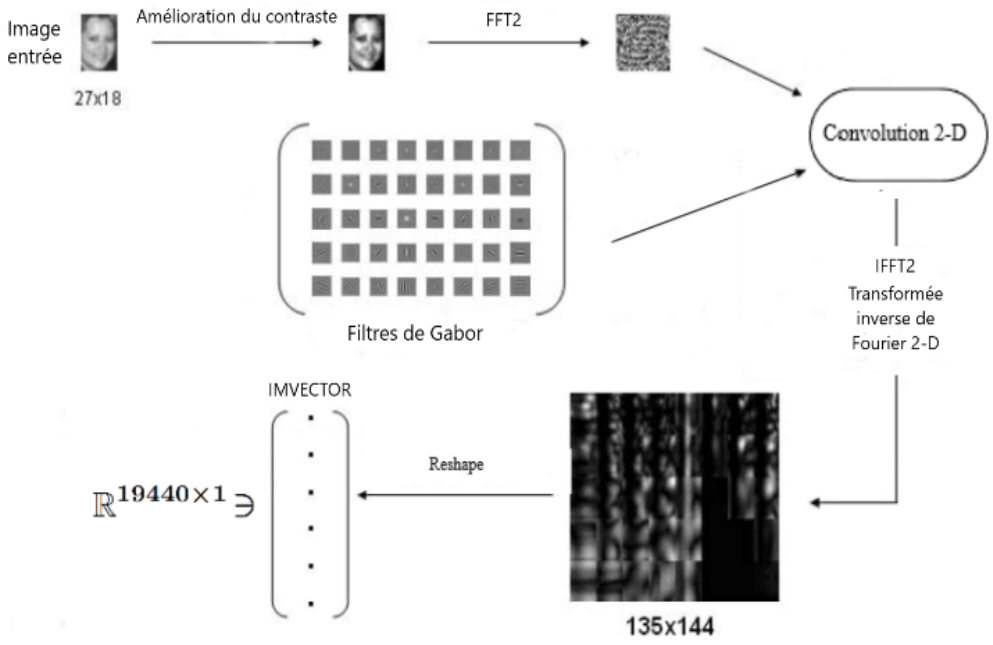

FIGURE 3.23 – Extraction des caractéristiques par les filtres de Gabor.

7 Classification

La dernière étape dans la détection faciale est la classification. Dans le code d'Omid Sakhi [18], la méthode de la classification utilisée est la méthode basée sur les machines à vecteurs de support (SVMs) [20], la fonction MATLAB utilisée dans ce cas est "**svmtrain**". Dans ce chapitre nous utilisons les méthodes vues en chapitres 1 et 2 au lieu SVM. Il s'agit de comparer les signatures obtenues grâce à l'application de

filtres de Gabor à une base de données déjà étiquetée comme visage ou non-visage. Dans notre cas, nous employons d'abord une base de 69 visages et 55 non-visages, la figure (3.24) donne quelques exemples de la base de visages et non-visages (extraite de [18]).

FIGURE 3.24 – Exemples de visages et non visages.

Une fois la classification terminée, l'algorithme encadrera les visages en vert et donnera ainsi le résultat de la détection faciale. La figure (3.25) montre le résultat de la reconnaissance faciale par la méthode ADF sur l'exemple.

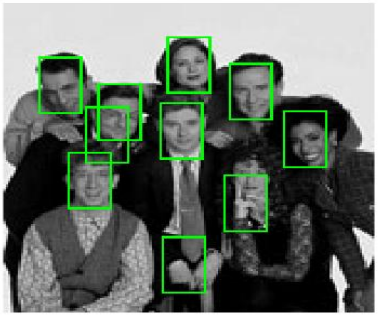

FIGURE 3.25 – Résultat de la détection faciale sur l'exemple.

Chapitre 4

Application au problème de la détection faciale

1 Introduction

Dans ce chapitre, nous expliquerons les fonctions MATLAB contenues dans le code d'Omid Sakhi [18] pour la détection de visage. Ensuite, nous appliquons les méthodes d'analyse discriminante vues aux chapitres 1 et 2 à la reconnaissance faciale du chapitre 3. Nous utilisons le code d'Omid Sakhi [18] avec une légère modification. Nous allons appliquer l'algorithme de ADF (voir chapitre 1, algorithme 1.2), ensuite l'algorithme de GDA (voir chapitre 2, algorithme 2.1) au lieu de l'algorithme de SVM.

2 Méthode d'évaluation

2.1 Précision

Pour tester la performance de la méthode ADF et GDA pour la détection faciale dans notre code, nous calculons le taux d'exactitude (accuracy en anglais) qui est donné dans l'expression (4.1) :

$$accuracy = \frac{\text{Nombre de données correctement prédites}}{\text{Nombre de données totales}} \times 100. \tag{4.1}$$

Nous choisissons des exemples d'images contenant plusieurs visages qui peuvent être détectés, ensuite nous calculons l'accuracy de la méthode ADF et GDA.

2.2 Code MATLAB

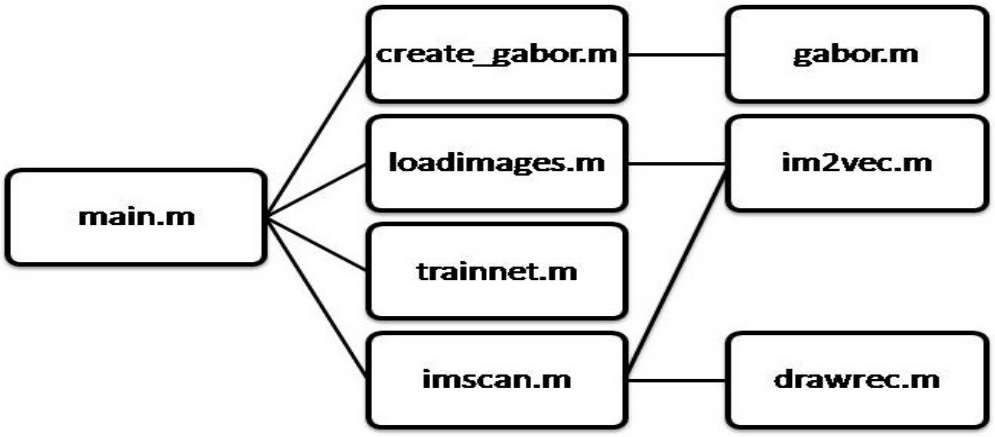

FIGURE 4.1 – Organisation du code MATLAB des tests numériques.

- **main.m :** Cette fonction représente la fonction d'exécution principale, elle propose un menu contextuel permettant de sélectionner l'action à exécuter.
- **gabor.m :** Cette fonction contient l'équation de Gabor pour générer les 40 filtres de Gabor suivant une paramétrisation que l'on définit (la figure (3.20)).
- **create_gabor.m :** Cette fonction fait appel à la sous fonction "**gabor.m**". La fonction ne devrait s'exécuter qu'une seule fois, après elle générera un fichier "**gabor.mat**" contenant les filtres de Gabor déjà générés dans les exécutions précédentes.
- **loadimages.m :** Cette fonction permet de préparer les images de la base de données pour entrainer le classifieur, elle génère une base de données sous forme d'une "cellule" de 3 lignes, chaque colonne représente un visage ou un non-visage, la première ligne contient le nom de l'image, la deuxième ligne contient la classe de l'image "visage" ou "non-visage" (1 pour visage, 0 pour non-visage), et la troisième ligne contient les données numériques de l'image.
  L'algorithme reçoit les données numériques de chaque image dans la ligne 3 à l'aide de la fonction "**imread**" de MATLAB. Après exécution, la fonction sauvegarde la cellule résultante sous le nom "**imgdb.mat**" pour des utilisations ultérieures.
- **im2vec.m :** Cette fonction permet de traiter l'imagette pour la préparer à la classification en appliquant des filtres de Gabor. Cette fonction utilise les transformées de Fourier et les transformées de Fourier inverse 2D (les fonctions MATLAB utilisées "**fft2**" et "**ifft2**"). La variable "**Features135x144**" contient le résultat de la convolution vue dans le chapitre précédent.
  Enfin, le résultat "**Features135x144**" est transformé en vecteur "**IMVECTOR**" à l'aide de la fonction "**reshape**" de MATLAB.
- **adf.m :** Cette fonction consiste à appliquer l'algorithme (1.2) de la méthode ADF (voir chapitre 1).
- **gda.m :** Cette fonction consiste à appliquer l'algorithme (2.1) de la méthode GDA (voir chapitre 2).
- **trainnet.m :** Cette fonction consiste à utiliser la base de données résultante de la fonction "**loadimages.m**" afin d'entraîner le classificateur. Nous employons

les deux méthodes présentées aux chapitres 1 et 2. Cette fonction fait appel à la sous fonction "**adf.m**" pour la méthode ADF et "**gda.m**" pour la méthode GDA.

- **drawrec.m :** Cette fonction permet d'encadrer les visages détectés.
- **classification_adf.m :** Cette fonction reprenait la direction w résultante de l'exécution de "**adf.m**" et la moyenne des données de chaque classe (μ_0 et μ_1), après elle calcule le seuil de classification δ (voir chapitre 1, l'équation 1.19), afin de classifier le point x (dans notre cas image).
  En effet, si $w^T x > \delta$, cette fonction retourne 1 (cette image est un visage), sinon elle retourne 0 (cette image est un non-visage).
- **classification_gda.m :** Cette fonction reprenait la direction v résultante de l'exécution de "**gda.m**" et la moyenne des données projetées de chaque classe (μ_0 et μ_1), après elle calcule le seuil de classification Ψ (voir chapitre 2, l'équation 2.20), afin de classifier le point x (dans notre cas image).
  En effet, si $v^T \phi(x) > \Psi$, cette fonction retourne 1 (cette image est un visage), sinon elle retourne 0 (cette image est un non-visage).
- **imscan.m :** Cette fonction doit scanner et détecter les imagettes qui seront ensuite classifiées suivant les méthodes vues aux chapitres 1 et 2. La figure (4.2) illustre la convolution et la recherche de points d'intensité, comme on l'a vu au chapitre 3.

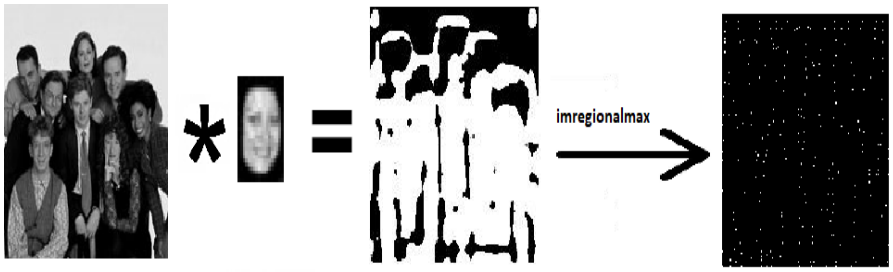

FIGURE 4.2 – Étapes d'exécution de la fonction imscan.

Ensuite, cette fonction fait appel à la sous fonction "**classification_adf.m**" ou bien "**classification_gda.m**" pour classifier les imagettes. Une fois les visages détectés, la fonction fait appel à la fonction "**drawrec.m**" afin d'encadrer les visages détectés.

Nous appliquons les méthodes ADF et GDA sur les trois images extraitent dans le code d'Omid [18].

Exemple 1 : Dans cet exemple le nombre de visages est 7.

FIGURE 4.3 – Exemple 1.

Exemple 2 : Dans cet exemple le nombre de visages est 15.

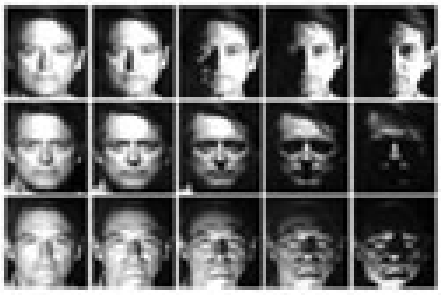

FIGURE 4.4 – Exemple 2.

Exemple 3 : Dans cet exemple le nombre de visages est 8.

FIGURE 4.5 – Exemple 3.

3 Résultat d'exécution

Le tableau suivant récapitule les résultats des expériences.

Méthodes	ADF	GDA
accuracy_training(%)	100%	100%
Exemple 1	85.71%	100%
Exemple 2	93.33%	93.33%
Exemple 3	100%	87.50%

TABLEAU 4.1 – Résultats d'exécution.

Les figures (4.6), (4.8) et (4.7) montrent les résultats d'exécution de l'algorithme avec ADF et GDA dans la détection des exemples 1,2 et 3 respectivement.

Exemple 1 :

(a) ADF (b) GDA

FIGURE 4.6 – Détection faciale de l'exemple 1.

Exemple 2 :

(a) ADF (b) GDA

FIGURE 4.7 – Détection faciale de l'exemple 2.

Exemple 3 :

(a) ADF (b) GDA

FIGURE 4.8 – Détection faciale de l'exemple 3.

Conclusion et perspectives

Dans ce mémoire, nous avons étudié la détection d'un visage en donnant une modélisation mathématique. Des méthodes de réduction de dimension à des fins des classifications ont été donné.

Dans ce travail nous confirmons d'abord les résultats obtenus dans la référence [3] que les données visages et non-visages sont linéairement séparables, vue que dans le cadre des SVMs uniquement le noyau linéaire a donné une satisfaction, là aussi GDA n'a pas donné une grande amélioration par rapport à ADF.

En général, dans le cadre de la méthode ADF, directement l'inverse ou la résolution d'un système est utilisé pour obtenir le vecteur de projection. Quand les données sont de petite dimension et il n'y a pas de corrélation entre les données, la matrice intra-classes est de rang maximal, ceci implique que la matrice de variance est automatiquement inversible et donc ADF donne de bons résultats. Par contre, dans ce contexte-là, les données sont de grande taille et il y a une grande corrélation entre les données et donc la matrice intra-classes n'est pas forcément inversible, donc la méthode ADF ne fonctionne pas correctement dans le système de la détection faciale (voir la figure 4.9). Pour résoudre ce problème d'inversibilité, nous avons utilisé le pseudo-inverse de Moore-Penrose [7] qui donne de bons résultats mais qui est très couteux. Ceci représente une voix de recherche intéressante qui a le compromis d'utiliser une méthode d'inverse généralisée et être très rapide.

FIGURE 4.9 – Résultat de ADF.

Annexe A

Dans cette annexe on réécrit l'équation (2.11) sous forme matricielle pour obtenir l'équation (2.10). On développe chaque terme de l'équation (2.11) selon les matrices K et W. A partir des équations (2.3) et (2.9), le terme de gauche de (2.11) donne :

$$\begin{aligned} \mathrm{V}v &= \frac{1}{n}\sum_{l=0}^{1}\sum_{k=1}^{n_l}\phi(\mathrm{X}^l_{:k})\phi^{\mathrm{T}}(\mathrm{X}^l_{:k})\sum_{p=0}^{1}\sum_{q=0}^{n_p}\alpha_{pq}\phi(\mathrm{X}^p_{:q}), \\ &= \frac{1}{n}\sum_{p=0}^{1}\sum_{q=0}^{n_p}\alpha_{pq}\sum_{l=0}^{1}\sum_{k=1}^{n_l}\phi(\mathrm{X}^l_{:k})[\phi^{\mathrm{T}}(\mathrm{X}^l_{:k})\phi(\mathrm{X}^p_{:q})]. \end{aligned}$$

$$\begin{aligned} \lambda\phi^{\mathrm{T}}(\mathrm{X}^m_{:j})\mathrm{V}v &= \frac{\lambda}{n}\sum_{p=0}^{1}\sum_{q=0}^{n_p}\alpha_{pq}\phi^{\mathrm{T}}(\mathrm{X}^m_{:j})\sum_{l=0}^{1}\sum_{k=1}^{n_l}\phi(\mathrm{X}^l_{:k})[\phi^{\mathrm{T}}(\mathrm{X}^l_{:k})\phi(\mathrm{X}^p_{:q})], \\ &= \frac{\lambda}{n}\sum_{p=0}^{1}\sum_{q=0}^{n_p}\alpha_{pq}\sum_{l=0}^{1}\sum_{k=1}^{n_l}[\phi^{\mathrm{T}}(\mathrm{X}^m_{:j})\phi(\mathrm{X}^l_{:k})][\phi^{\mathrm{T}}(\mathrm{X}^l_{:k})\phi(\mathrm{X}^p_{:q})]. \end{aligned}$$

En utilisant cette formule pour toute classe m et pour tout son élément j, on obtient :

$$\lambda(\phi^{\mathrm{T}}(\mathrm{X}^0_{:1}),...,\phi^{\mathrm{T}}(\mathrm{X}^0_{:n_1}),\phi^{\mathrm{T}}(\mathrm{X}^1_{:1}),...,\phi^{\mathrm{T}}(\mathrm{X}^1_{:n_1}) = \frac{\lambda}{n}\mathrm{KK}\alpha. \tag{4.2}$$

D'après les équations (2.1), (2.2) et (2.9), le terme de droite de l'équation (2.11) donne :

$$\begin{aligned} \mathrm{B}v &= \frac{1}{n}\sum_{p=0}^{1}\sum_{q=0}^{n_p}\alpha_{pq}\phi(\mathrm{X}^p_{:q})\sum_{l=0}^{1}n_l[\frac{1}{n_l}\sum_{k=1}^{n_l}\phi(\mathrm{X}^l_{:k})][\frac{1}{n_l}\sum_{k=1}^{n_l}\phi(\mathrm{X}^l_{:k})]^{\mathrm{T}}, \\ &= \frac{1}{n}\sum_{p=0}^{1}\sum_{q=0}^{n_p}\alpha_{pq}\sum_{l=0}^{1}[\sum_{k=1}^{n_l}\phi(\mathrm{X}^l_{:k})][\frac{1}{n_l}][\sum_{k=1}^{n_l}\phi^{\mathrm{T}}(\mathrm{X}^l_{:k})\phi^{\mathrm{T}}(\mathrm{X}^p_{:q})]. \end{aligned}$$

$$\phi^{\mathrm{T}}(\mathrm{X}^m_{:j})\mathrm{B}v = \frac{1}{n}\sum_{p=0}^{1}\sum_{q=0}^{n_p}\alpha_{pq}\sum_{l=0}^{1}[\phi^{\mathrm{T}}(\mathrm{X}^m_{:j})\sum_{k=1}^{n_l}\phi(\mathrm{X}^l_{:k})][\frac{1}{n_l}][\sum_{k=1}^{n_l}\phi^{\mathrm{T}}(\mathrm{X}^l_{:k})\phi^{\mathrm{T}}(\mathrm{X}^p_{:q})]. \tag{4.3}$$

En utilisant cette formule pour toute classe m et pour tout son élément j, on obtient :

$$(\phi^{\mathrm{T}}(\mathrm{X}^0_{:1}),...,\phi^{\mathrm{T}}(\mathrm{X}^0_{:n_1}),\phi^{\mathrm{T}}(\mathrm{X}^1_{:1}),...,\phi^{\mathrm{T}}(\mathrm{X}^1_{:n_1})\mathrm{B}v = \frac{1}{n}\mathrm{KWK}\alpha. \tag{4.4}$$

En combinant 4.2 et 4.4 ,on obtient :

$$\lambda\mathrm{KK}\alpha = \mathrm{KWK}\alpha \tag{4.5}$$

On multiplions 4.5) par un α^{T} pour obtenir (2.10).

Bibliographie

[1] Amir. A, Benmaghnia. N. Noyaux à base de polynômes orthogonaux pour les machines à vecteur de support (SVMs). Université Abdelhamid Ibn Badis, Mostaganem, 2017. 19, 20

[2] Amour. H, Boualam. L and Attaf. Y. Analyse d'images Par filtre de Gabor : Application aux images médicales. PhD diss. Université Mouloud Mammeri, 2010. 26

[3] Amir. A, Korichi. M. A. Reconnaissance Faciale Par les SVM. Université Abdelhamid Ibn Badis, Mostaganem, 2020. 44

[4] Baudat. G, Anouar. F. Generalized discriminant analysis using a kernel approach. Neural Computation 12 (10) (2000) 2385–2404. 19, 20, 21, 22

[5] Chaari. A. Reconnaissance de visages par réseaux d'ondelettes de Gabor. Thèse de Doctorat Université de Lille 1, 2009. 33, 34

[6] Didier MAQUIN. Institut National Polytechnique de Lorraine, 2000. 26

[7] DESPLAS. M. Matrice pseudo-inverse de Moore-Penrose et variables duales généralisées en programmation mathématique. RAIRO. Recherche opérationnelle, tome 26, 1992. 44

[8] Ghodsi. A. Statistical Learning-Classification. STAT 841, STAT 441, CM 463, 2007. 13, 14

[9] Guérin. J, Lahrichi. N and Le Digabel. S. Déomposition en valeurs singulières (SVD), 2022. 13

[10] Izenman. A. J. Linear Discriminant Analysis. In : Modern Multivariate Statistical Techniques. Springer Texts in Statistics. Springer, New York, NY, 2013. https ://doi.org/10.1007/978-0-387-781891_8. 9, 10

[11] Javier Movellan. R. A Comparison of Gabor Filter Methods for Automatic Detection of Facial Landmarks. University of California, 2002. 30, 31

[12] Moualkia. H, Sâadi. N and Boulgamh. F. Etude comparative entre le débruitage d'images par des méthodes classiques et par le filtre bilatéral. Application à des images à multi-copies bruitées. Université Larbi Ben M'hidi, Oum El-Bouaghi, 2015. 24, 25, 28, 29

[13] Mishra. S. P, Sarkar. U, Taraphder. S, Datta. S, Swain. D. P, Saikhom. R, Panda. S, and Laishram. M. Principal Component Analysis. International Journal of Livestock Research, 2017. 10

[14] Pin. C. E. K, Zak Stanislaw. H. An introduction to optimization. Wiley, 2013. 12, 15

[15] Saporta. G. Probabilits, analyse des donnes et statistique. Editions TECHNIP, 2006. 11, 12

[16] Shen. L, Bai. L and Fairhurst. M. Gabor wavelets and General Discriminat Analysis for face identification and verification. Image and vision computiong 25 (2007) 553-563. 9, 21, 32, 33

[17] Sharif. M, Khalid. A, Raza. M and Mohsin. S. Face Recognition using Gabor Filters. Journal of Applied Computer Science and Mathematics, 2011. 36

[18] Sakhi. O . Face Detection System, 2020.
https ://www.mathworks.com/matlabcentral/ ?leexchange/11073-face-detection-system.
MATLAB Central File Exchange. 34, 35, 37, 38, 39, 41

[19] Sheikh. R, Patel. M and Sinhal. A. Recognizing MNIST Handwritten Data Set Using PCA and LDA, 2020. In : G. Mathur , H. Sharma and Bundele. M, Dey. N,Paprzycki. M. (eds) International Conference on Artificial Intelligence : Advances and Applications 2019. Algorithms for Intelligent Systems. Springer, Singapore.
https ://doi.org/10.1007/978-981-15-1059-5_20. 16

[20] Vapnik. V. The Nature of Statistical Learning Theory, Springer-Verlag, New York (1995). 9, 37