Geometry-Aware Analysis of High-Dimensional Visual Information Sets

THÈSE N^{O} 4296 (2008)

PRÉSENTÉE LE 20 FÉVRIER 2009
À LA FACULTE SCIENCES ET TECHNIQUES DE L'INGÉNIEUR
LABORATOIRE DE TRAITEMENT DES SIGNAUX 4
PROGRAMME DOCTORAL EN INFORMATIQUE, COMMUNICATIONS ET INFORMATION

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Effrosyni KOKIOPOULOU

acceptée sur proposition du jury:

Prof. P. Vandergheynst, président du jury
Prof. P. Frossard, directeur de thèse
Prof. N. Paragios, rapporteur
Prof. J.-Ph. Thiran, rapporteur
Dr O. Verscheure, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE
Suisse
2009

We are drowning in information, but we are starving for knowledge

...heard it somewhere

Acknowledgments

First, I would like to thank my PhD advisor, Prof. Pascal Frossard, for giving me the opportunity to participate in his research group and for his great flexibility in giving me the crucial room I needed to explore my own ideas and directions. Working closely with him, I had the opportunity to learn a lot and benefit from his experience and his vision. I am grateful for his genuine and continuous interest in my research progress and for standing always by my side with guidance and understanding. I would also like to thank him for ensuring a great atmosphere in the laboratory and in general for being such a kind boss!

I am thankful to the members of my thesis committee, Prof. N. Paragios, Prof. J-P. Thiran and Dr. O. Verscheure, as well as the president of the committee Prof. P. Vandergheynst, for the time spent in reviewing this manuscript, and for their helpful comments on improving its content. The work presented here has been sponsored by NCCR IM2. I am grateful for this support.

My deepest gratitude goes to Prof. Yousef Saad for his guidance during my graduate studies at UMN, and his continuous support and encouragement during the last years. I was very fortunate to work with him and get exposed to his enlighted way of thinking, his vision and personal integrity. I am grateful to Prof. Efstratios Gallopoulos, as his role was central in stimulating my interest in research during my early steps. I would like also to thank Prof. Nikos Papanikolopoulos for his encouragement and for his valuable advice. I would like to extend my deepest thanks to Prof. Stergios Roumeliotis and Prof. Georgios Giannakis for the numerous discussions I enjoyed with them; it is not rare that great ideas come while enjoying a good coffee at Starbucks. I would also like to thank prof. Paul Schrater for his excellent course on Pattern Recognition in Fall 2003 at UMN. This course has played decisive role in shaping my research directions the years that followed.

I thank all my former and current colleagues in the LTS4 laboratory for the nice moments we spent together and for making life at EPFL a great deal more enjoyable: Nikos, Zorana, Zafer, Eirini, Jean-Paul, Dan, Ivana, Ivana, Tamara, Luigi, Vijay, Jari and Jacob. I would like also to thank my officemate Zafer Arican as well as Laurent Jacques for the exciting research discussions we had. Special thanks go to Nikos for our tasteful discussions about Greek politics. I also thank the ITS staff, in particular Mariane Marion, Rosie di Pietro and Gilles Auric for making the administrative work less painful. My kindest thanks go to my students for giving me the opportunity to see how it feels to be on the advisor side. I am proud of them and I hope that they learned from me, as I learned from them.

Special thanks go to Zorana, Yuga and Tanja for kindly accepting me as roommate during the last year of my PhD. I thank all my friends in Greece and abroad for their support and for all the nice moments we had together.

Finally, I am indebted to my family, my mother Vassiliki, my father Xenofon, for their love and dedication towards me during all my life. Special thanks go to my sister, Ioanna and my brother, Nikos for their support all these years. All my gratitude goes to my husband Costas and his family for their unconditional love and support.

Abstract

Over the past few decades we have been experiencing a data explosion; massive amounts of data are increasingly collected and multimedia databases, such as `YouTube` and `Flickr`, are rapidly expanding. At the same time rapid technological advancements in mobile devices and vision sensors have led to the emergence of novel multimedia mining architectures. These produce even more multimedia data, which are possibly captured under geometric transformations and need to be efficiently stored and analyzed. It is also common in such systems that data are collected distributively. This very fact poses great challenges in the design of effective methods for analysis and knowledge discovery from multimedia data. In this thesis, we study various instances of the problem of classification of visual data under the viewpoint of modern challenges. Roughly speaking, classification corresponds to the problem of categorizing an observed object to a particular class (or category), based on previously seen examples. We address important issues related to classification, namely flexible data representation for joint coding and classification, robust classification in the case of large geometric transformations and classification with multiple object observations in both centralized and distributed settings.

We start by identifying the need for flexible data representation methods that are efficient in both storage and classification of multimedia data. Such flexible schemes offer the potential to significantly impact the efficiency of current multimedia mining systems, as they permit the classification of multimedia patterns directly in their compressed form, without the need for decompression. We propose a framework, called semantic approximation, which is based on sparse data representations. It formulates dimensionality reduction as a matrix factorization problem, under hybrid criteria that are posed as a trade-off between approximation for efficient compression and discrimination for effective classification. We demonstrate that the proposed methodology competes with state-of-the-art solutions in image classification and face recognition, implying that compression and classification can be performed jointly without performance penalties with respect to expensive disjoint solutions.

Next, we allow the multimedia patterns to be geometrically transformed and we focus on transformation invariance issues in pattern classification. When a pattern is transformed, it spans a manifold in a high dimensional space. We focus on the problem of computing the distance of a certain test pattern from the manifold, which is also closely related to the image alignment problem. This is a hard non-convex problem that has only been sub-optimally addressed before. We represent transformation manifolds based on sparse geometric expansions, which results in a closed-form representation of the manifold equation with respect to the transformation parameters. When the transformation consists of a synthesis of translations, rotations and scalings, we prove that the objective function of this problem can be decomposed as a difference of convex functions (DC). This very property allows us to solve optimally our optimization problem with a cutting plane algorithm, which is well known to successfully find the global minimizer in practice. We showcase applications in robust face recognition and image alignment.

The classification problem with multiple observations is addressed next. Multiple observations are typically produced in practice when an object is observed over successive time instants or under different viewpoints. In particular, we focus on the problem of classifying an object when multiple geometrically transformed observations of it are available. These multiple observations typically belong to a manifold and the classification problem resides in determining which manifold the observations belong to. We show that this problem can be viewed as a special case of semi-supervised learning, where all unlabelled examples belong to the same class. We design a graph-based algorithm, which exploits the structure of the manifold. Estimating the unknown object class then results into a discrete optimization problem that can be solved efficiently. We show the performance of our algorithm in classification of multiple handwritten digit images and in video-based face recognition.

Next, we study the problem of classification of multiple observations in distributed scenarios, such as camera networks. In this case the classification is performed iteratively and distributively, without the presence of a central coordinator node. The main goal is to reach a global classification decision using only local computation and communication, while ensuring robustness to changes in network topology. We propose to use consensus algorithms in order to design a distributed version of the aforementioned graph-based algorithm. We show that the distributed classification algorithm has similar performance as its centralized counterpart, provided that the training set is sufficiently large. Finally, we delve further into the convergence properties of consensus-based distributed algorithms and we propose an acceleration methodology for fast convergence that uses the memory of the sensors. Our simulations show that the convergence is indeed accelerated in both static and dynamic networks, and that distributed classification in sensor networks can significantly benefit from them.

Overall, the present thesis addresses a few important issues related to pattern analysis and classification in modern multimedia systems. Our solutions for semantic approximation and transformation invariance can impact the efficiency and robustness of classification in multimedia systems. Furthermore, our graph-based framework for multiple observations is able to perform effective classification in both centralized and distributed environments. Finally, our fast consensus algorithms can significantly contribute to the accelerated convergence of distributed classification algorithms in sensor networks.

Keywords: pattern classification, transformation invariance, sparse representations, dimensionality reduction, multiple observations, distributed classification, distributed consensus.

Résumé

Pendant les dernières décennies nous avons constaté une explosion de données; des quantités des données volumineuses sont de plus en plus accumulées et des bases de données multimédia, comme par exemple `YouTube` et `Flickr`, deviennent de plus en plus répandues. En même temps, les avancements rapides de la technologie des mobiles et des capteurs de vision ont mené à l'apparition de nouvelles architectures d' analyse de données multimédia. Ceux-ci produisent même encore plus de données multimédia, qui peuvent être capturées avec des transformations géométriques et doivent être efficacement stockées et analysées. C'est également habituel dans de tels systèmes que les données soient collectées de façon distribuée. Ce fait introduit des grands défis pour la conception des méthodes efficaces d'analyse de données multimédia. Dans cette thèse, nous étudions divers exemples du problème de la classification des données visuelles sous le point de vue des défis actuels. En général, la classification correspond à l' attribution d' une classe (ou d' une catégorie) à un objet observé, basée sur des exemples vus précédemment. Nous abordons des questions importantes liées à la classification, comme la représentation flexible des données pour un codage et une classification jointe, la classification robuste dans le cas de grandes transformations géométriques et la classification d' observations multiples d'un objet dans des scénarios centralisés et distribués.

Nous proposons tout d' abord des méthodes de représentation de données flexibles qui sont efficaces à la fois pour le stockage et la classification des données multimédia. Ces schémas flexibles offrent le potentiel d'influencer de manière significative l'efficacité des systèmes d' exploration des données multimédia actuels, puisqu'ils permettent la classification des signaux multimédia directement sous leur forme compressée, sans besoin de décompression. Nous proposons un algorithm, nommé approximation sémantique, qui est basé sur la représentation de données parcimonieuse. Il formule la réduction de la dimensionnalité comme un problème de factorisation d'une matrice, en utilisant des critères hybrides qui sont posés comme un compromis entre l'approximation pour une compression efficace et la discrimination pour une classification efficace. Nous démontrons que la méthodologie proposée est concurrente avec les solutions actuelles de classification d'image et d'identification de visage, illustrant que la compression et la classification peuvent être effectuées conjointement sans pénalité de performance en comparaison avec des solutions disjointes coûteuses.

Nous étudions ensuite les problèmes d' invariance aux transformations géométriques des signaux. Un signal transformé géométriquement génère un manifold dont la dimension est plus petite que la dimension du signal original. Nous abordons le problème du calcul de la distance d'un signal observé au manifold generé par transformation géométrique. Ce problème est étroitement lié au problème d' alignement d' image. Ceci est un problème difficile non convexe qui a été seulement adressé avant avec une façon sous-optimale. Nous représentons les manifolds créés par transformation géométrique à l'aide de décompositions parcimonieuses des signaux. Ceci permet de représenter le manifold de façon analytique. Lorsque la transformation du signal est composée de déplacements, rotations et changement d'échelle, nous démontrons que le calcul de la distance au manifold peut être formulé comme une différences de fonctions convexes (DC). Cette propriété permet de résoudre de façon optimale le problème d'optimisation, pour lequel nous présentons des applications dans le contexte de l'identification de visages ou d'alignement d'images.

Nous décrivons ensuite le problème de la classification avec des observations multiples. Ces observations correspondent par exemple aux mesures effectuées à partir de différents points de vue, ou à des instants différents. En particulier, nous nous intéressons au cas où les observations multiples sont liées par des transformations géométriques. Ces observations appartiennent typiquement au même manifold, et le problème de classification revient à déterminer à quel manifold les observations correspondent. Nous démontrons que ce problème est similaire à un cas particulier d'apprentissage semi-supervisé, où tous les échantillons appartiennent à la même classe. Nous proposons un algorithme basé sur des graphes, qui exploitent la régularité du manifold. L'estimation de la classe de l'objet observé devient un problème d'optimisation discret qui peut être résolu très efficacement. Nous démontrons les performances de l'algorithme proposé pour la reconnaissance d'écriture ou l'identification de visages dans des séquences vidéo.

Nous étendons ensuite le problème de classification d'observations multiples à des scénarios distribués comme des réseaux de caméras. Dans ce cas, la classification s'effectue de façon itérative et distribuée, sans l'intervention d'un noeud central. L'objectif principal est d'obtenir une décision globale en utilisant seulement des informations locales, d'une manière qui soit robuste aux changements de la topologie du réseau. Nous proposons d'utiliser des algorithmes de consensus afin de construire une version distribuée de l'algorithme de classification ci-dessus. Nous démontrons que les performances de l'algorithme distribué sont comparables à celles de l'algorithme centralisé si l'ensemble de données d'apprentissage est suffisamment grand. Finalement, une étude approfondie des propriétés de convergence des algorithmes distribués basés sur des méthodes de consensus conduit à une méthode d'accélération de la convergence qui utilisent la mémoire des capteurs. Les simulations démontrent que la convergence est accélérée pour des réseaux statiques comme pour des réseaux dynamiques, ce qui est très avantageux pour le problème de classification distribuée.

Cette thèse aborde plusieurs problèmes importants liés à l'analyse et la classification de données dans des systèmes multimédia modernes. Nos solutions qui se rapportent à l'approximation sémantique et l'invariance vis-à-vis des transformations géométriques peuvent influencer les performances et la robustesse de la classification dans des systèmes multimédia. En outre, les méthodes basées sur les graphes pour la classification d'observations multiples se montrent efficaces dans des environnements centralisés ou distribués. Finalement, les nouveaux algorithmes de consensus rapide contribuent de manière significative à la convergence accélérée des algorithmes de classification distribués dans des réseaux de capteurs.

Mots-clés: classification, invariance aux transformations géométriques, représentations parcimonieuses, réduction de dimensionnalité, observations multiples, classification distribuée, consensus.

Contents

1 Introduction 1
- 1.1 Analysis of high-dimensional data . . . 1
- 1.2 Modern challenges . . . 2
- 1.3 Thesis outline . . . 3
- 1.4 Summary of contributions . . . 5

2 Prior Art 7
- 2.1 Overview . . . 7
- 2.2 Dimensionality reduction . . . 7
- 2.3 Transformation invariance in pattern analysis . . . 10
- 2.4 Classification of multiple observations . . . 13
  - 2.4.1 Centralized algorithms . . . 13
  - 2.4.2 Distributed algorithms . . . 16
- 2.5 Distributed consensus . . . 17
  - 2.5.1 Background . . . 17
  - 2.5.2 Accelerating the convergence . . . 19
- 2.6 Summary . . . 19

3 Flexible dimensionality reduction 21
- 3.1 Introduction . . . 21
- 3.2 Dimensionality reduction using simultaneous sparse approximations . . . 22
- 3.3 Supervised dimensionality reduction . . . 25
  - 3.3.1 Supervised atom selection . . . 25
  - 3.3.2 Analysis of SAS . . . 27
- 3.4 Semantic approximation of images . . . 28
  - 3.4.1 Dictionary design . . . 28
  - 3.4.2 Implementation Issues . . . 29
  - 3.4.3 Experimental setup . . . 30
  - 3.4.4 Classification performance . . . 34
  - 3.4.5 Approximation performance . . . 35
  - 3.4.6 Approximation and Classification trade-off . . . 36
- 3.5 Application to face recognition . . . 37
  - 3.5.1 2D face recognition . . . 37
  - 3.5.2 3D face recognition . . . 39

3.6 Related work 41
3.7 Conclusions 42

4 Transformation Invariance 45
4.1 Introduction 45
4.2 Problem formulation 46
    4.2.1 Transformation manifolds 46
    4.2.2 Estimation of transformation parameters 48
4.3 DC decomposition 49
    4.3.1 Properties of DC functions 49
    4.3.2 DC decomposition of transformed atoms 50
    4.3.3 DC form of the objective function 53
4.4 Image alignment experiments 56
    4.4.1 Comparison of different methods 56
    4.4.2 Application to medical image registration 57
4.5 Transformation invariant face recognition 59
4.6 Related work 61
4.7 Conclusions 63

5 Classification of Multiple Observations 65
5.1 Introduction 65
5.2 Problem definition 66
5.3 Classification with modified TSVM 67
5.4 Graph-based classification 68
    5.4.1 Label propagation with multiple observations 68
    5.4.2 The MASC algorithm 71
5.5 Classification of multiple images 72
5.6 Video-based face recognition 74
    5.6.1 Experimental setup 74
    5.6.2 Classification results on VidTIMIT 75
    5.6.3 Classification results on Honda/UCSD 77
    5.6.4 Video-based face recognition overview 78
5.7 Conclusions 80

6 Distributed Classification of Multiple Observations 81
6.1 Introduction 81
6.2 Problem formulation 82
6.3 Distributed MASC 83
    6.3.1 Preliminaries 83
    6.3.2 Algorithmic description 84
    6.3.3 Discussion 85
    6.3.4 Cost analysis 86
6.4 Simulation results 87
    6.4.1 Setup 87
    6.4.2 Distributed vs centralized MASC 88
    6.4.3 Effect of consensus on distributed MASC 90
6.5 Related work 91

6.6 Conclusions 92

7 Fast Convergence in Distributed Consensus 93
7.1 Introduction 93
7.2 Convergence in distributed consensus averaging 94
7.2.1 Static network topology 94
7.2.2 Dynamic network topology 97
7.3 Accelerated consensus with polynomial filtering 99
7.3.1 Exploiting memory 99
7.3.2 Polynomial filtering 99
7.3.3 Newton's interpolating polynomial 100
7.4 Polynomial filter design with semi-definite programming 102
7.4.1 Polynomial filtering for static network topologies 102
7.4.2 Polynomial filtering for dynamic network topologies 103
7.5 Simulation results 104
7.5.1 Setup 104
7.5.2 Static network topologies 105
7.5.3 Dynamic network topologies 109
7.6 Consensus-based distributed classification 112
7.7 Related work 112
7.8 Conclusions 114

8 Conclusions 115
8.1 Thesis achievements 115
8.2 Future directions 116

A Appendix of Chapter 4 121
A.1 Proof of Proposition 3(b) 121
A.2 Proof of Lemma 4.3.1 121
A.3 The cutting plane algorithm 122

Resume 135

List of Figures

1.1 Analysis of high-dimensional visual information sets. 2
1.2 Ad-hoc network of vision sensors 3
1.3 Typical scenarios of producing multiple observations of an object. 4

2.1 Manifold distance is the distance from a reference point p to the transformation manifold of s. The tangent distance is the distance from p to the tangent space of the manifold with respect to s. 11
2.2 Typical structure of $\mathcal{G}_d$. 14
2.3 Graph-based network model. 18

3.1 Sample face images from the ORL database. There are 10 available facial expressions and poses for each subject. 31
3.2 Impact of different dictionaries on the classification performance. 32
3.3 Recovered basis vectors from the handwritten digit collection. 33
3.4 Recovered basis vectors from the ORL face data set. 33
3.5 DCT vs Gaussian features on the digits data set. 34
3.6 Image recognition experiments. 35
3.7 Approximation of facial images (CBCL dataset), using NMF and SOMP. . . 36
3.8 Approximation-classification trade-off with SAS algorithm on the handwritten digit dataset. 37
3.9 Face recognition experiments. 38
3.10 Block diagram of the 3D face recognition system. 39
3.11 Gaussian atoms 40
3.12 Classification error rates on the FRGC v.1.0 database. 41

4.1 Progressive OMP approximation of the digit “5” (leftmost) with 10, 20, 30, 40 and 50 Gaussian atoms (from left to right). 47
4.2 Samples from the transformation manifold of the digit “5”. From left to right, the samples correspond to rotation angles from 0 to 2π with step $\pi/4$. 47
4.3 Manifold distance is the minimum distance from a reference point p to the transformation manifold of s. 48
4.4 Invariance of various methods with respect to scale and rotation. 56
4.5 The MRI brain image and its successive approximations. 57
4.6 The transformed MRI brain images and their estimated transformed images. First row corresponds to η_1 and second row to η_2. 58

4.7 The transformed test face images used for face recognition. 59
4.8 Histogram of the absolute error of the rotation angle (first row) and of the scale (second row). 60

5.1 Structure of the class-conditional label matrix Z_p. 69
5.2 Classification results measured on two different data sets. 72
5.3 Head pose variations in the VidTIMIT database. 75
5.4 A typical face manifold from the VidTIMIT database. Observe the four clusters corresponding to the four different head poses (face looking left, right, up and down). 75
5.5 Video face recognition results on the VidTIMIT database. 76
5.6 Head pose variations in the Honda/UCSD database. 77
5.7 A typical face manifold from the Honda/UCSD database. 78
5.8 Video face recognition results on the Honda/UCSD database. 79

6.1 Ad-hoc network of vision sensors 82
6.2 Conceptual distinction between the two graphs of the problem. $\mathcal{G}_s$ (resp. $\mathcal{G}_d$) denotes the graph of the sensor network topology (resp. the data graph). In $\mathcal{G}_d$, the filled (resp. empty) circles correspond to labelled (resp. unlabelled) examples. 83
6.3 Flow of computation, which is repeated for each hypothesis p, $p = 1, \ldots, c$. The stars in the last row of each similarity matrix correspond to the nearest neighbors of the observation $x^{(u)}$ among the labelled examples. 86
6.4 Sample face images from the UMIST database. The number of different poses for each subject is varying. 87
6.5 Distributed multi-view face recognition in a vision sensor network. Each facial image corresponds to the observation of a sensor. The problem is to estimate the unknown class in a distributed fashion. 88
6.6 Difference in performance between MASC and its distributed version versus the number of training samples (per class). 89
6.7 Classification performance versus number of multiple observations, for both methods. Each curve corresponds to different number of training samples per class. 90
6.8 Average classification error rate vs consensus iterations. 91

7.1 Static network topology. The parameter $W(i, j) \neq 0$ describes the weight of the edge that permits the sensor i to communicate with its neighbor sensor j. 95
7.2 Dynamic network topology. Each link is active with a probability p_{ij}. When a link is active, the parameter $W(i, j)$ describes the weight of the edge that lets the sensors i and j to communicate. 97
7.3 Newton's polynomial $p_k(\lambda)$ of various degrees k. 101
7.4 Effect of polynomial filtering on the spectrum of the Maximum-degree matrix W of eq. (7.26). 106
7.5 Average consensus performance of polynomial filtering for variable degree k on fixed topology with the Laplacian weight matrix. 107
7.6 Average consensus performance of Newton's polynomial and SDP polynomial ($k = 4$) on fixed topology. 107

7.7 Average number of consensus iterations versus the network size m. Results with different weight matrices are shown. 108
7.8 Results of SDP polynomial filtering applied to the optimal weight matrix. . . 109
7.9 Average consensus performance using the Laplacian weight matrix on random network topologies: the case of small k. 110
7.10 Average consensus performance using the Laplacian weight matrix on random network topologies: the case of large k. 111
7.11 Classification error rate vs consensus iterations with SDP polynomial filtering, where the sensor network uses the Metropolis weight matrix. 112
7.12 Classification error rate vs consensus iterations with SDP polynomial filtering, where the sensor network uses the Maximum-degree weight matrix. 113

List of Tables

3.1 The data sets used in the experimental evaluation, where n_i denotes the number of samples per class. 32
3.2 The data sets used in 2D face recognition, where n_i denotes the number of samples per class. 38
3.3 Test configurations. 40

4.1 Transformation invariant face recognition results. 61

5.1 Video face recognition results on the Honda/UCSD database. 77

7.1 Summary of methods and their input parameters. 105

Chapter 1

Introduction

1.1 Analysis of high-dimensional data

Moore's law has predicted the doubling of our computing capacity roughly every eighteen months over the last four decades. This progress, unmatched by any other technology, has enabled us to use computers in almost every aspect of human life. Today, computation is everywhere, from our cars, cell phones and desktop computers in the office, to ubiquitous computing in the household. Moreover, our computing resources are now interconnected to such an extent that scientists have begun to consider the Internet as a single entity and the information it contains invaluable.

This combination of outstanding computing and networking advancements has led to a data deluge. Over the last fifteen years we have been experiencing a revolution in the amount of data that we collect and publish. Multimedia databases such as `YouTube`, `Flickr` and `Picasa` are continuously expanding. Digital data is everywhere and refers to virtually every piece of information around us. These huge volumes of data that "simply" wait to be exploited, lend themselves as a valuable source of information for powerful knowledge discovery. This very fact creates a grand challenge: How can we create useful knowledge out of an ocean of bits and bytes? This question that refers to the problem of data analysis, lies at the heart of the present thesis. Although some problems of this nature may look rather simple to the human brain, they are however extremely challenging for any machine that is capable of performing arithmetic operations.

In this thesis we focus on computationally viable algorithms, in order to address a few of the most important problems in pattern recognition and in analysis of high-dimensional data. High dimensions stem from the representation of data in high-dimensional vector spaces, such as the number of pixels of an image, the numerous frames of a video sequence or the several points in a 3D point cloud. Of particular importance to this thesis are visual information sets that come from modern multimedia applications. We focus on the problem of classification of patterns in multimedia signals, such as images and video.

We study different instances of the classification problem that are posed by modern challenges in novel multimedia processing systems. We investigate novel ways of representing visual information by means of meaningful geometric features, which further facilitate classification. We study the properties of such a representation in the context of geometric transformation invariance in pattern analysis. We also consider the problem of classification with

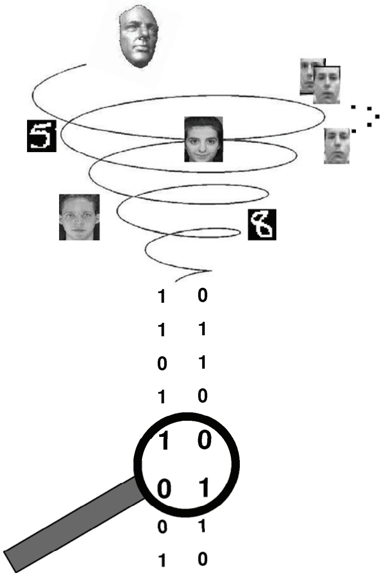


Figure 1.1: *Analysis of high-dimensional visual information sets.*

multiple transformed observations of the test pattern and we design distributed algorithms for addressing it in distributed settings.

1.2 Modern challenges

The emergence of novel multimedia architectures and the explosion in the volume of multimedia data that are collected by them, has strongly motivated the development of pattern analysis methods that are effective in both classification and efficient storage of such data. Classical media coding for pure compression performance is certainly sub-optimal in this context, since it can discard useful information that may be crucial for the learning task, and it generally requires a decompression step before feature extraction that represents a computational bottleneck in large systems. Hence, flexible representation methods that typically address jointly compression and feature extraction for data mining problems become of particular importance, as they may offer computational efficiency to current media processing systems and robustness against noise [23, 30].

Additionally, it is often the case that patterns are captured under geometric transformations (e.g., translations, rotations and scaling in the case of visual objects). Methods that are able to handle large geometric pattern transformations, will contribute to the development of robust classification algorithms. Moreover, an object may be captured in multiple observations, which may correspond to different viewing angles or successive time instants, for example. In this case, one may want to exploit the diversity of information provided by the multiple observations, in order to obtain increased classification accuracy. This problem becomes even more challenging in its distributed version, where multiple observations are collected distributively and the classification has to be done distributively too. This occurs, for instance, in ad-hoc vision sensor networks, such as the one shown in Fig. 1.2 that are typically networks with arbitrary topology, characterized by the absence of a central coordinator

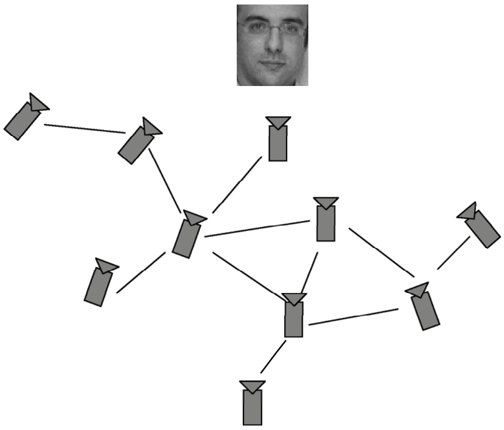


Figure 1.2: *Ad-hoc network of vision sensors*

node.

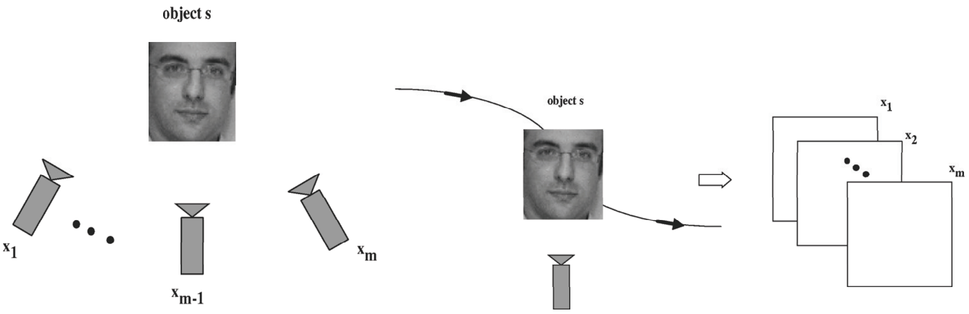
Thus, there is a dire need for pattern classification frameworks that are able to harness the modern challenges. These exciting issues that lie on the interface between pattern recognition and signal processing have not been thoroughly investigated so far. For instance, little attention has been given to developing solutions that jointly address approximation and classification. At the same time, although a lot of research efforts have been devoted in matching transformed versions of a certain pattern, this problem has only been sub-optimally addressed before. Additionally, the problem of pattern classification under multiple observations has not been thoroughly investigated. Hence, important problems concerning the classification of multimedia patterns have not been thoroughly addressed so far.

In this thesis, we provide solutions to the above issues. Our proposed semantic approximation framework offers an interesting trade-off between approximation and classification and our non-convex optimization methodology, based on differences of convex functions, offers globally optimal solutions for transformation invariant pattern analysis. We present also our graph-based framework for both centralized and distributed classification of multiple observations. In the latter case, we further provide our solutions for fast distributed computation based on convex programming.

1.3 Thesis outline

We present our ideas, solutions as well as in-depth analysis of a few of the most important issues that arise in the classification of multimedia patterns in modern multimedia systems.

We first address in Chapter 3 the problem of **flexible dimensionality reduction for joint approximation and classification**. We propose a *semantic approximation* methodology, which aims at reducing multimedia information under a compact form that directly permits efficient classification. The semantic approximation problem starts from a subspace method where dimensionality reduction is formulated as a matrix factorization problem. Data samples are jointly represented in a common subspace extracted from a redundant parametric dictionary of basis functions. We first build on greedy pursuit algorithms for simultaneous



(a) Network of vision sensors (b) Video frames of a moving object

Figure 1.3: *Typical scenarios of producing multiple observations of an object.*

sparse approximations to solve the dimensionality reduction problem. The method is extended into a supervised algorithm, which further encourages class separability in the extraction of the signal features. The resulting supervised dimensionality reduction scheme provides an interesting trade-off between approximation (or compression) and discriminant feature extraction (or classification). The semantic approximation methodology provides a compressed signal representation that can be directly used for multimedia data analysis. The application of the proposed algorithms to image recognition problems further demonstrates classification performances that are competitive with state of the art solutions in handwritten digit as well as in 2D and 3D face recognition.

Next, in Chapter 4, we allow the objects of interest to be geometrically transformed and we study **invariance to geometric pattern transformations**. Transformation invariance is an important property in pattern recognition, where different versions of the same object should typically receive the same label. We focus on a transformation invariant distance measure that represents the minimum distance between the transformation manifolds that are spanned by patterns of interest under geometric transformations. Since these manifolds are typically nonlinear, the computation of the manifold distance becomes a non-convex optimization problem, which has been only sub-optimally addressed before. We propose to represent a pattern of interest as a linear combination of a few geometric functions extracted from a structured and redundant basis. Transforming the pattern results in the transformation of its constituent parts. We show that when the transformation is restricted to a synthesis of translations, rotations and isotropic scalings, such a pattern representation results in a closed-form expression of the manifold equation with respect to the transformation parameters. We demonstrate that the manifold distance computation can then be formulated as a minimization problem, whose objective function is expressed as the *difference of convex functions* (DC). This crucial property permits to solve *optimally* the optimization problem with DC programming solvers that are globally convergent. We present experimental evidence showing that our method is able to find the globally optimal solution, outperforming state of the art methods that yield only sub-optimal solutions.

Then, we study in Chapter 5 the extension of the above problem in the case of **multiple transformed pattern observations**. The problem now is to classify an object based on multiple transformed observations of it. Figure 1.3 shows various scenarios of producing multiple observations in practice. We view this problem as a special case of semi-supervised learning where all unlabelled examples belong to the same unknown class. Semi-supervised learning is the learning paradigm where the test data are available during the training process. We propose a graph-based algorithm that optimizes the unknown label matrix in a way that exploits the special structure of the problem. In particular, we formulate an optimization problem whose objective function measures the smoothness of a candidate label matrix on the manifold. This results into a discrete optimization problem, which can be solved by an efficient and low complexity algorithm. We provide experimental results that show the effectiveness of the proposed methodology in diverse problems, ranging from the classification of multiple handwritten digit images to face recognition from video.

In Chapter 6, we study the problem of **distributed classification of multiple observations**. In particular, we consider the scenario where multiple observations of an object are collected distributively in an ad-hoc sensor network, where each sensor captures an observation that is different from its peers. The problem now is to classify the observed object into the unknown class by aggregating information across the network, such that all sensors reach a consensus classification decision. We employ *distributed consensus* as the main computational tool in order to propose the distributed version of the previous graph-based algorithm. Distributed consensus has become increasingly popular and it has been employed in several distributed aggregation tasks (e.g., distributed averaging). We provide simulation results that show that the difference in classification performance between the distributed and the centralized graph-based algorithm becomes negligible as the number of training examples increases. It is worth noting that the communication cost of the distributed algorithm is directly driven by the convergence properties of the particular consensus algorithm that is employed.

Motivated by the above fact, we address in Chapter 7 the problem of **optimizing the convergence rate of distributed consensus** by exploiting the sensors' memory. The consensus averaging problem has been typically addressed in the literature by distributed linear iterative algorithms, with asymptotic convergence of the consensus solution. The convergence rate of such distributed algorithms typically depends on the network topology and the weights given to the edges between neighboring sensors, as described by the network matrix. We show that one can impact the convergence rate for given network matrices by the use of polynomial filtering. The main idea of the proposed methodology is to apply a polynomial filter on the network matrix that will shape its spectrum in order to increase the convergence rate. We show that such an algorithm is equivalent to periodic updates in each of the sensors by aggregating a few of its previous estimates. We formulate the computation of the coefficients of the optimal polynomial as a semi-definite program that can be efficiently and globally solved for both static and dynamic network topologies. We provide simulation results that demonstrate the effectiveness of the proposed solutions in accelerating the convergence of distributed consensus averaging problems as well as in contributing to fast distributed classification.

1.4 Summary of contributions

Compared to previous work on classification of multimedia patterns, this thesis brings the following contributions:

- We introduce the virtue of joint approximation and discrimination criteria for dimensionality reduction and semantic data analysis in general. We show also that simultaneous sparse approximations can be beneficial for effective dimensionality reduction.

- We represent pattern manifolds using sparse geometric expansions over parametric structured and redundant bases. This results in a closed form representation of the manifold equation with respect to the transformation parameters. This is valuable for optimization.

- We solve the non-convex problem of manifold distance computation problem *optimally*. We prove that the objective function of this problem can be decomposed in DC form (i.e., difference of convex functions). Hence, the globally optimal solution can be efficiently computed.

- We propose a graph-based method for addressing the problem of classifying a pattern with multiple observations, which exploits the fact that all observations belong to the same class.

- We show that distributed classification in ad-hoc sensor networks can be performed by means of consensus. We illustrate the feasibility of such an algorithm in the context of pattern classification in vision sensor networks and show its competitiveness to its centralized counterpart, when the number of training examples becomes sufficiently large.

- We propose the use of sensors' memory in accelerating the convergence process of distributed consensus problems. We formulate our idea as a semi-definite program, whose solution provides the optimal weights of the linear combination of the sensors' previous estimates.

Chapter 2

Prior Art

2.1 Overview

In the present chapter, we review the most relevant methods from the literature, which are linked with the general problems addressed in this thesis. First, in Section 2.2, we review the methods that seek a meaningful and parsimonious representation of the data, a process which is also known as dimensionality reduction. Next, in Section 2.3, we focus on transformation invariance issues in pattern analysis and discuss various methods that attack classification under geometric pattern transformations, as well as various ways of introducing invariance in pattern recognition. In the sequel, in Section 2.4, we focus on the problem of classification of multiple observations of an object. We view this problem as a special case of semi-supervised learning and we review the most relevant methods in this area. We also consider distributed aspects of the above problem and provide an overview of consensus-based distributed classification. Finally, we focus on accelerating the convergence rate of distributed consensus and in Section 2.5 we review state of the art methods for fast distributed consensus. Additionally, for the sake of completeness, each following chapter includes an overview of some additional methods that are either application-oriented state of the art methods or methods that are mostly conceptually related to the methodologies proposed in this thesis.

2.2 Dimensionality reduction

In general, given a set of high dimensional data samples, the goal of dimensionality reduction is to map the data to a low dimensional space such that certain properties of the initial set are preserved. Dimensionality reduction is a very broad concept that encompasses numerous methods proposed in the literature. One may mostly distinguish the following families of methods: (a) linear methods (e.g., LDA [119, Ch.4], LPP [36], ONPP [61, 62]), (b) nonlinear methods (e.g., LLE [89], Laplacian Eigenmaps [5], Isomap [107]) and (c) low rank approximation methods (e.g., PCA [47], NMF [64, 80]). The first two categories employ a mapping from the high dimensional space to a low dimensional space, which is linear in the former case and nonlinear (implicit) in the latter case. However, these methods have not been designed with approximation in mind, and they typically target at pure discriminant objectives. We discuss briefly the LDA method below due to its large popularity, but we will not discuss further any other of the above methods as this would diverge the focus. The third family that is the closest to the methods proposed in this thesis, includes methods that use a low rank approximation of the data matrix. In other words, they use only a small number of basis vectors to approximate the high dimensional data of interest. After introducing LDA, we provide further details about these methods.

Denote by $S \in \mathbb{R}^{m\times n}$ the data matrix whose columns hold the high dimensional data samples, and by $s_j^{(i)}$ the jth sample of the ith class. Linear Discriminant Analysis (LDA) extracts a set of discriminant axes, such that the data are well separated into their respective classes when they are projected on them. Assume that we have c classes and that each class i has n_i data samples. Define the *between-class scatter matrix*

$$S_B = \sum_{i=1}^{c} n_i(\mu^{(i)} - \mu)(\mu^{(i)} - \mu)^\top$$

and the *within-class scatter matrix*

$$S_W = \sum_{i=1}^{c} \left(\sum_{j=1}^{n_i} (s_j^{(i)} - \mu^{(i)})(s_j^{(i)} - \mu^{(i)})^\top \right)$$

where $\mu^{(i)}$ is the sample mean vector of the ith class and μ the global sample mean vector. In LDA the projected data are produced as $\tilde{S} = W^\top S,\ W \in R^{m\times r}$ and the columns of W are the eigenvectors associated with the r largest eigenvalues of the following generalized eigenvalue problem

$$S_B w = \lambda S_W w. \tag{2.1}$$

Intuitively, the matrix W of LDA maximizes the ratio of inter-class dispersion over the intra-class dispersion. Note that the rank of S_B is at most $c - 1$, which implies that the above problem has only $c - 1$ generalized eigenvalues. Therefore, LDA can yield at most $c - 1$ discriminant axes.

Low-rank approximation methods The low rank approximation methods seek matrices $W \in \mathbb{R}^{m\times r}$ and $H \in \mathbb{R}^{r\times n}$ such that $r \ll m$ and $W \cdot H$ is close to S. Hence, the S-data are represented as H-data in the W-space, which is of much smaller dimension r relative to m.

The most popular subspace method for dimensionality reduction is Principal Component Analysis (PCA) [47]. In PCA, a subspace is constructed from the eigenvectors of the sample covariance matrix,

$$A = (S - \mu e^\top)(S - \mu e^\top)^\top,$$

where $\mu = \frac{1}{n}\sum_i^n s_i$ is the sample mean vector of the data samples. Dimensionality reduction is accomplished by discarding the eigenvectors of A that correspond to its smallest eigenvalues. PCA maximizes the variance of the projected data. The obtained basis vectors from PCA are typically holistic and of global support.

Non-negative Matrix Factorization (NMF), introduced in [64, 80], is another popular dimensionality reduction method with empirical success in real life data sets. It certainly represents the closest solution to the strategies presented in this thesis, although it cannot be used easily for signal compression and compact data representation. It has been proposed as a subspace method for a parts-based representation of objects by imposing non-negativity constraints, typical to digital imaging applications, for example. Given a data matrix $S \in$

$\mathbb{R}^{m\times n}$ with non-negative entries, NMF seeks two non-negative factors $W \in R^{m\times r}$ and $H \in R^{r\times n}$ such that

$$S \approx WH. \tag{2.2}$$

The columns of the matrix W contain the basis vectors and the matrix H contains the corresponding coefficients (or encoding) vectors for the approximation of the columns of S. Consider the generalized Kullback-Leibler (KL) divergence between X and Y

$$D(X||Y) = \sum_{i=1}^{n}\sum_{j=1}^{m} [x_{ij}\log\frac{x_{ij}}{y_{ij}} - x_{ij} + y_{ij}]. \tag{2.3}$$

The KL divergence is the most popular objective function used in NMF algorithms. The *Standard NMF* can be formulated as the following optimization problem

Optimization problem: **NMF**
$\min_{W,H} D(S||WH)$,
subject to
$W, H \geq 0$,
$\sum_i w_{ij} = 1,\ \forall j$.

A local minimum solution to the above problem can be obtained by iterating the multiplicative rules introduced in [64]. The *Local NMF* (LNMF) [68] is a variant of NMF, which tries to enforce the spatial locality of the basis vectors. In particular, it differs from the standard NMF by imposing three additional constraints expressed by the following rules: (a) the number of basis components should be minimized, (b) different basis vectors should be as orthogonal as possible and (c) only the most important components are retained. LNMF can be formulated as the following optimization problem.

Optimization problem: **LNMF**
$\min_{W,H} D(S||WH) + \alpha\sum_{i,j} u_{ij} - \beta\sum_i z_{ii}$,
subject to
$W, H \geq 0$,
$\alpha, \beta > 0$,
$U = W^\top W$,
$Z = HH^\top$.

In the objective function we have introduced the scalars α and β, which are the Lagrange multipliers corresponding to the additional constraints on spatial locality of features. A local minimum solution to the above problem can be obtained by iterating the three multiplicative rules introduced in [68].

Other variants of NMF have also been proposed recently. For example, a sparsity controlled NMF algorithm based on a measure of sparsity that is a combination of the L1 and L2 norm, has been proposed in [42]. Along the same ideas of controlling sparsity of the reduced subspaces, NMF variants using convex programming have been proposed in [37, 38]. Yet another variant of NMF has been presented in [82], where the authors describe an extension of standard NMF by imposing smoothness constraints on the non-negative factors. In particular, they apply their algorithm for the analysis of non-negative spectral data generated from astronomical spectrometers. Finally, in [81], the NMF model is modified by introducing a smoothing symmetric matrix which controls the sparsity of both non-negative factors.

Although the NMF optimization problem is convex with respect to W or H individually, it is however non-convex with respect to both of them. Thus, all algorithms that have been proposed in the literature are not guaranteed to converge to the global minimum and they are prone to local minima. Moreover, it has been observed that they are also sensitive to the initializations of the two non-negative factors. If the initialization is not good it may happen that the algorithm gets trapped in a bad local minimum, which leads to clearly sub-optimal performances.

Finally, extension to classification problems have been proposed with supervised variants of NMF, which takes into account class labels information. The authors in [118] and [127] independently propose a supervised NMF algorithm by incorporating the Fisher constraints into the objective function of NMF and they propose multiplicative update rules.

Positioning Although the above methods provide good performances at low rank approximation of the data, they present certain shortcomings in terms of compactness of the representation. In particular, it can be shown that PCA is optimal in terms of L2 approximation error [35, Sec. 3.4.3]. However, it generally fails to identify features that are spatially localized, since its basis vectors are dense and have global support. This represents a clear drawback for applications that rely on parts-based representations of data objects, or where the most relevant information is contained in localized features.

The NMF optimization problems on the other hand generally require sophisticated constraints, in order to shape the properties of the basis functions (see e.g., [37, 38]). It represents a clear drawback with respect to solutions based on flexible dictionaries of functions, as presented in this thesis. In addition, even if NMF results in good signal approximation, it cannot lead to compact representations as the resulting basis vectors W are specifically tuned to the data S. Therefore, they are as hard to compress as the initial images themselves. On the contrary, as we will show in the next chapter, the basis vectors in a flexible structured dictionary have a parametric mathematical description and can be compactly represented by a few parameters only.

Finally, one may want to address the problem by compressing first, and then performing classification on the compressed data. Several works have been proposed in the past few years, where learning tasks are directly performed on the compressed signal built by coding standards (see e.g., [116, 20] and references therein). However, the extracted features are not optimal, since the compression does not target any classification task, and the signal analysis becomes quite sensitive to the coding rate and the testing conditions. In the semantic approximation framework proposed in this thesis, compression is rather accomplished by a flexible dimensionality reduction method that is designed to be aware of the subsequent learning task.

2.3 Transformation invariance in pattern analysis

In this section, we review the main research efforts on introducing transformation invariance in pattern recognition. It often happens that an object is observed or represented under geometric transformations. In visual object classification for example, it is generally desirable that the classification function be invariant to object transformations such as scalings, rota-

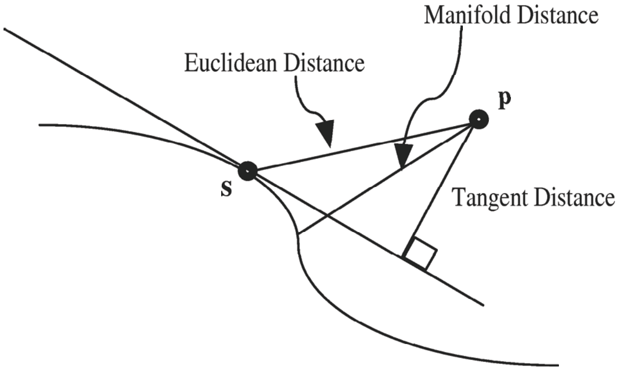


Figure 2.1: *Manifold distance is the distance from a reference point p to the transformation manifold of s. The tangent distance is the distance from p to the tangent space of the manifold with respect to s.*

tions and translations. Thus, transformation invariance is a very important property of any learning algorithm.

One possible approach to introduce invariance into pattern recognition algorithms is to use transformation invariant features. However, there are two main disadvantages with this approach; (i) crucial information may be discarded and (ii) it is hard to evaluate the impact of feature extraction on the classification error [113]. For the above reasons, a lot of research efforts have concentrated on seeking for invariance by the computation (or the approximation) of appropriate distance measures in the pattern space or by simulating transformations on the training examples. In the sequel, we review the related techniques.

Transformation invariant distance measures When a pattern is geometrically transformed, it spans a low dimensional (usually nonlinear) manifold living in a high dimensional space. The intrinsic dimension of the manifold is typically low and it is equal to the degrees of freedom of the applied transformation. If the distance between two patterns is transformation invariant, it is equivalent to the distance between their corresponding transformation manifolds, which is moreover called the manifold distance (MD). Observe that the computation of the manifold distance requires the alignment of the involved patterns. However, alignment is a hard optimization problem, mostly due to the non-convexity of its objective function.

One solution consists in the local linearization of the manifold, in order to compute the distance between tangent spaces instead of the true manifold distance. P. Simard et al in [100] introduced the notion of the tangent distance (TD) of a reference pattern p from the manifold $\mathcal{T}_s$ spanned by the transformed versions of the pattern s. The TD has been successfully applied in digit image recognition [52]. The main idea is to use a locally linear approximation of $\mathcal{T}_s$ around s. The linear approximation is constructed using the tangent vectors $\frac{\partial \mathcal{T}}{\partial \eta}$, where η denotes the transformation parameters. The tangent distance is defined as the distance of p from the tangent space (see Fig. 2.1 for a schematic illustration). The effectiveness of this approach is highly sensitive to the nonlinearity of the manifold. Since the obtained linear approximation using the tangent space is accurate only locally, this method provides local invariance to transformations.

As we have already mentioned, the smallest distance of p from the manifold $\mathcal{T}_s$ is called the manifold distance (MD) (shown in Fig. 2.1) and is truly transformation invariant. Unfortunately, these manifolds have no analytic expression in general and they are typically nonlinear. Hence, the computation of MD involves the solution of a hard (typically nonconvex) optimization problem with an unknown number of local minima. One approach in computing the MD is to use a traditional method such as Newton's method or steepest descent. However, this yields a sub-optimal solution due to the presence of local minima. Along the same lines, in [24] the authors use probabilistic pattern models and they introduce priors both on the transformation parameters as well as the pattern that generates the manifold. They introduce the so-called joint manifold distance, defined as the distance between the two pattern manifolds that is optimized over the transformation parameters and the pattern themselves. The resulting optimization problem is solved using Levenberg-Marquardt, which is also a local method that is susceptible to local minima.

To alleviate the problem of local mimima, N. Vasconcelos et al [113] proposed the multiresolution manifold distance, which attempts to compute the MD by using a multiresolution decomposition of the involved images and then employ Newton's method in each resolution level. Starting from the coarsest level, Newton's method is used in order to converge to a local minimum solution, which is used as the initial guess to Newton in the next level. This is iterated until the finest image resolution is reached. The intuition is that at the coarsest level, the MD objective function will be less "bumpy" with less local minima. Thus, the hope is that Newton's method will be less susceptible to them. Although this methodology provides robustness to a wider range of transformations relative to TD and MD, there is no guarantee that it will converge to the global minimizer.

Virtual examples and jittering Another approach to introduce invariance in pattern recognition algorithms is by learning from transformed examples, the so-called *virtual samples*. Typically, one applies various transformations on the training examples resulting in an expanded training set. This has the advantage that can be readily combined with any classification algorithm. However, the training set rapidly becomes large and this approach is costly in terms of memory requirements. Motivated by this shortcoming, DeCoste et al [17] introduced the concept of jittered queries. The main idea is to apply various transformations on the test samples resulting in an expanded set of transformed test samples. Shifting the responsibility of the invariance from the training to the testing phase has certain advantages in terms of computational cost and memory usage.

Positioning While the above research efforts provide robustness to small transformations, they generally fail in the presence of large transformations. This is due to the hardness of the optimization problem which is non-convex. There is no guarantee for local methods, such as Newton and Levenberg-Marquardt, that they will converge to the globally optimal solution. Additionally, methods based on virtual samples can partially alleviate the shortcomings of local methods, but this comes at the cost of large memory requirements. Moreover, the effectiveness of the above methods is sensitive to the nonlinearity of the manifold. Hence, they provide only local invariance to pattern transformations and clearly there is a need for global methods.

In this thesis, we offer a novel solution to this problem when the transformation consists of a synthesis of translations, rotations and (isotropic) scalings. Our method does not rely on any sampling or discretization of the manifold; it is rather based on continuous optimization.

We represent transformation manifolds over structured and redundant bases, which allows for closed form representation of the manifold equation with respect to the transformation parameters. Next, we prove that the objective function can be decomposed in a DC form; that is, we formulate the optimization problem as a DC program. We employ an efficient cutting plane method to solve the DC program which has theoretical guarantees to converge to the globally optimal solution.

2.4 Classification of multiple observations

In many realistic scenarios, multiple observations of an object or pattern are captured over successive time instants or under different geometric transformations. In this context, the problem of pattern classification with multiple observations becomes increasingly important. Since all observations belong to the same (unknown) class, the problem resides in estimating the unknown class. Depending on whether the observations are collected in a centralized or distributed fashion, one may categorize the methods in two main corresponding classes. In what follows, we review first the centralized methods and next we discuss the distributed ones.

2.4.1 Centralized algorithms

One straightforward approach is to apply a local (transformation invariant) classifier on each observation and then estimate the unknown class by fusing the local classification decisions. The fusion step may be performed by applying a classifier combination scheme, such as majority voting (the reader is referred to [7, Ch. 14] for more details on classifier combination). However, such an approach would treat the observations as independent (which is typically not the case in practice) and it would not exploit the relative geometric structure of the observations; for instance, in cases that they belong to a manifold. Therefore, there is a need for methods that treat the multiple observations jointly (and not independently). This is reminiscent of semi-supervised learning (SSL), where the test examples are available during training and they can be jointly used for refining the training process. Before we go on, let us revisit quickly the main categories of learning.

There are three paradigms of learning [13]; unsupervised, supervised and semi-supervised. In *unsupervised* learning, the learner is given a set $X = [x_1, \dots, x_n]$ of n examples (or data samples or points), where $x_i \in \mathcal{X}$, $i = 1, \dots, n$. The goal of unsupervised learning is to identify interesting structure in the data X. For instance, in clustering, the goal is to partition the examples in groups, such that similar examples are included in the same group and dissimilar examples in different ones. In *supervised* learning, the learner is additionally provided with the corresponding labels $y_i \in \mathcal{Y}$ for each training example x_i, and the goal is to learn a mapping from $\mathcal{X}$ to $\mathcal{Y}$. When the labels y_i are discrete (resp. continuous) the task is called *classification* (resp. regression). *Semi-supervised* learning lies in between the previous two learning paradigms. In this case, the learner is provided with the training data and corresponding labels, but not for all examples.

In the sequel, we review the most relevant methods from the literature on semi-supervised learning (SSL). For a complete survey on SSL the reader is referred to [13]. Let us first introduce the necessary notation. Assume that we are given a data set $X = \{X^{(l)}, X^{(u)}\}$, where $X^{(l)} = \{x_1, x_2, \dots, x_l\} \subseteq \mathbb{R}^d$, where $l = |X^{(l)}|$ and $X^{(u)} = \{x_{l+1}, \dots, x_n\} \subseteq \mathbb{R}^d$, where $m = |X^{(u)}|$ and $n = l + m$. We are also given a label set $\mathcal{L} = \{1, \dots, c\}$, where c is the number

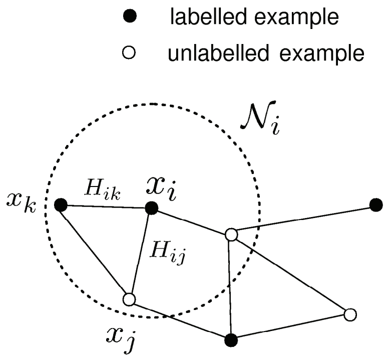


Figure 2.2: *Typical structure of* $\mathcal{G}_d$.

of classes. The examples in $X^{(l)}$ are labelled $\{y_1, y_2, \dots, y_l\}$, $y_i \in \mathcal{L}$, and the m examples in $X^{(u)}$ are unlabelled. The problem in semi-supervised learning is to predict the labels of $X^{(u)}$, given the data samples $X^{(l)}$ and $X^{(u)}$, as well as the labels of $X^{(l)}$.

We start with the graph-based methods, since they belong to the state of the art of SSL. The main idea of these methods is to build a graph that captures the geometry of the data manifold as well as the proximity of the data samples. We first review the label propagation algorithm [16], which is the most representative among the graph-based methods.

Label propagation This algorithm is based on the *smoothness assumption* for SSL, which states that if x_1 and x_2 are close by, then their corresponding labels y_1 and y_2 should be close as well. Denote by $\mathcal{M}$ the set of matrices with nonnegative entries, of size $n \times c$. Notice that any matrix $M \in \mathcal{M}$ provides a labelling of the data set by applying the following rule: $y_i = \max_{j=1,\dots,c} M_{ij}$. We denote the initial label matrix as $Y \in \mathcal{M}$ where $Y_{ij} = 1$ if x_i belongs to class j and 0 otherwise (this is also known as 1-of-c encoding). The label propagation algorithm first forms the k nearest neighbor (k-NN) graph defined as

$$\mathcal{G}_d = (\mathcal{V}_d, \mathcal{E}_d),$$

where the vertices $\mathcal{V}_d$ correspond to the data samples X. An edge $e_{ij} \in \mathcal{E}_d$ is drawn if and only if x_j is among the k nearest neighbors of x_i.

It is common practice to assign weights on the edge set of $\mathcal{G}_d$. One typical choice is the Gaussian weights

$$H_{ij} = \begin{cases} \exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2}) & \text{when } (i,j) \in \mathcal{E}_d, \\ 0 & \text{otherwise.} \end{cases} \tag{2.4}$$

The similarity matrix $S \in R^{n \times n}$ is further defined as

$$S = D^{-1/2} H D^{-1/2}, \tag{2.5}$$

where D is a diagonal matrix with entries $D_{ii} = \sum_{j=1}^{n} H_{ij}$. Figure 2.2 shows schematically the graph structure as well as some related notation.

Next, the algorithm computes a real valued $M^* \in \mathcal{M}$ based on which the final classification is performed using the rule $y_i = \max_{j=1,\dots,c} M^*_{ij}$. This is done via a regularization framework, where the cost function is defined as

$$\mathcal{Q}(M) = \frac{1}{2}\Big(\sum_{i,j=1}^{n} H_{ij}\Big\|\frac{1}{\sqrt{D_{ii}}}M_i - \frac{1}{\sqrt{D_{jj}}}M_j\Big\|^2 + \mu\sum_{i=1}^{n}\|M_i - Y_i\|^2\Big). \tag{2.6}$$

In the above, M_i denotes the ith row of M. The computation of M^* is done by solving the quadratic optimization problem $M^* = \arg\min_{M\in\mathcal{M}} \mathcal{Q}(M)$.

Intuitively, we are seeking an M^* that is smooth along the edges of similar pairs (x_i, x_j) and at the same time close to Y when evaluated on the labelled data $X^{(l)}$. The first term in (2.6) is the *smoothness* term and the second is the *fitness* term. Notice that when two examples x_i and x_j are similar (i.e., the weight H_{ij} is large) minimizing the smoothness term in (2.6) results in M being smooth across similar examples. Thus, similar data examples will probably share the same class label. It can be shown [16] that the solution to problem (2.6) is given by

$$M^* = \beta(I - \alpha S)^{-1}Y, \tag{2.7}$$

where $\alpha = \frac{1}{1+\mu}$ and $\beta = \frac{\mu}{1+\mu}$.

Note in passing that several other variants of label propagation have been proposed in the past few years. We mention for instance, the method of [130] and the variant of label propagation that was inspired from the Jacobi iteration algorithm [13, Ch. 11]. Finally, it is interesting to note that there have also been found connections to Markov random walks [105] and electric networks [131].

Transductive SVM One of the most fundamental methods in SSL, is the Transductive Support Vector Machines (TSVM) [112, 46]. This method is based on the *cluster assumption* or (*low density separation assumption*), which simply states that the data density is low near the decision surface. Assume in this case that we are interested in binary classification i.e., $\mathcal{L} = \{+1, -1\}$. Given m unlabelled test examples $X^{(u)}$, the main idea of TSVM is to find the label configuration of $X^{(u)}$ that results in the largest margin of both training and test data. In the non-separable case, the TSVM solves the following optimization problem [46]

Optimization problem: **TSVM**
$\min_{y^*,w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{l}\xi_i + C^*\sum_{j=0}^{m}\xi_j^*$
subject to
$y_i[w^\top x_i + b] \geq 1 - \xi_i,$
$y_j^*[w^\top x_j^* + b] \geq 1 - \xi_j^*,$
$\xi_i > 0,$
$\xi_j^* > 0$

In the above problem, the variables annotated with '$*$' denote the variables that correspond to the unlabelled data. Observe that the unknown labels y^* are among the unknowns, which makes the constraint $y_j^*[w^\top x_j^* + b] \geq 1 - \xi_j^*$ non-convex, and hence the TSVM optimization problem turns out to be non-convex as well. Since the label of each example can only be $\{+1, -1\}$, the size of the TSVM search space is 2^m i.e., exponential with respect to the test set size. Thus, a simple algorithm that tries all possible label assignments, would have been intractable when m grows large, due to its exponential complexity.

Given the hardness of the above problem, several research efforts have tried to propose sub-optimal algorithms that work sufficiently well in practice. For instance, T. Joachims [46] has proposed an optimization heuristic for solving the above problem. It starts with a label configuration obtained by a standard (non transductive) SVM and it improves it by switching the labels of a well picked pair of negative and positive samples, until convergence. For more information on recent research efforts attacking the TSVM optimization problem, the reader is referred to [13, Ch. 2].

Positioning The problem of classification of multiple observations may be seen as a special case of SSL, where all test samples belong to the same class. Note that the above methods refer to the generic scenario in SSL, where the unlabelled examples come from different classes. Therefore, in their current form, they do not exploit the specificities of the problem of classifying multiple observations. In this thesis, we introduce a graph-based classification algorithm that exploits the special structure of classifying an object with multiple observations. We particularly opt for a graph-based approach, as it offers the possibility to exploit the geometric structure of the data (e.g., manifold structure). The proposed method is based on the smoothness assumption, similarly to the graph-based methods. However, it differs from them, by constraining the unknown label matrix to fit the problem at hand. We will show that this framework results in a simple, low complexity, yet effective algorithm for classifying multiple observations, with application to video-based face recognition.

2.4.2 Distributed algorithms

We focus now on distributed classification in sensor networks with *ad-hoc* architecture. The latter implies that there is no central coordinator node (e.g., fusion center). Also the network topology may be arbitrary and it may be dynamic as well. For instance, such networks arise in the deployment of sensors in a geographic region. Typically, the sensors are very simple devices will limited memory and computation capabilities as well as small communication range. Hence, they can only communicate with their closest neighbors. It is also often possible that the network topology changes dynamically due to various reasons, such as link failures (e.g., in unstable environment conditions), sensor motion and so on.

We are interested in the distributed version of the problem of the previous section. In particular, we assume that an object is observed in a sensor network (see e.g. Fig. 1.2). Each sensor captures a single observation of this object and the problem is to estimate its class by aggregating information from all available observations over the network, such that all sensors reach a consensus decision. Each sensor is moreover assumed to have a copy of the training set.

To the best of our knowledge, this particular problem is in its infancy and it is only very recently that a few methods have started to emerge. These methods use distributed consensus to attack the problem of distributed classification in ad-hoc sensor networks. In general, the main goal of consensus is to reach a global solution iteratively using only local computation and communication, while staying robust to changes in the network topology (see Section 2.5 below for more details on distributed consensus). Although these classification methods do not exactly correspond to our problem setup, we review them below.

K. Flouri et al [25] propose algorithms for distributed SVM training based on consensus. They assume that each sensor has access to a small (different) subset of the training set. Two training algorithms are proposed, whose core idea is to exchange support vector sets, obtained from local training. In both algorithms, the exchange of support vectors between the sensors is performed using consensus. In the first algorithm, sensors exchange the minimum number of support vectors but the algorithm is sub-optimal (since a support vector of the whole set may not be included in the union of support vectors of the local training sets). In the second algorithm, the sensors exchange sufficient amount of information in order to reach optimality.

R. Tron and R. Vidal in [109] propose a consensus-based algorithm for distributed pose estimation algorithm in camera networks, with application to distributed face recognition. The main contribution is to extend the average consensus to handle scalars living in non-Euclidean manifolds, in particular rotations in SO(3). This is performed by considering an appropriate generalized distance measure in SO(3). The average pose is estimated by consensus while taking also into account the relative poses of the camera nodes. Once consensus has converged, the final classification decision is taken on the average pose.

Positioning In this thesis, we propose a consensus-based algorithm for distributed classification, which is suitable for ad-hoc architectures. It is a distributed version of the graph-based algorithm that has been developed for the centralized problem of the previous section. The distributed algorithm employs consensus in order to aggregate information from all observations across the network. At the end of the computation, all sensors reach a consensus classification decision.

We should mention that the method in [25] refers to the problem of distributed training, in contrast to our method that refers to distributed classification (i.e., testing phase). Hence, the problem setup is different in the two methods. Note finally that the method in [109] targets mostly at distributed face pose estimation, and face recognition is simply an application of that. On the contrary, our method targets at recognition directly.

2.5 Distributed consensus

Distributed consensus [6] algorithms are becoming increasingly popular and they attract numerous research efforts. It has recently become an important computational tool for various aggregation tasks in ad-hoc sensor networks. A few examples include applications in multimedia data analysis (see e.g., distributed pose estimation for face recognition in [109]), as well as other generic tasks; distributed estimation [95], distributed compression [87], coordination of networks of autonomous agents [8] and computation of averages and least-squares in a distributed fashion (see e.g., [121, 122, 123, 79] and references therein).

A very important problem in consensus is how to ensure its convergence and further how to increase its convergence rate. Fast consensus algorithms can contribute to significant energy savings and hence in fast distributed classification. In what follows, we first provide an introduction to the general problem of distributed consensus and in the sequel we provide an overview of state of the art methods for addressing the problem of increasing the convergence rate.

2.5.1 Background

Assume an ad-hoc network of m sensors. We model the network topology as an undirected graph $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s)$ with nodes $\mathcal{V}_s = \{1, \ldots, m\}$ corresponding to sensors. An edge $(i, j) \in \mathcal{E}_s$ is drawn if and only if sensor i can communicate with sensor j, as illustrated in Fig. 2.3. We denote the set of neighbors for node i as $\mathcal{N}_i = \{j|\ (i, j) \in \mathcal{E}_s\}$.

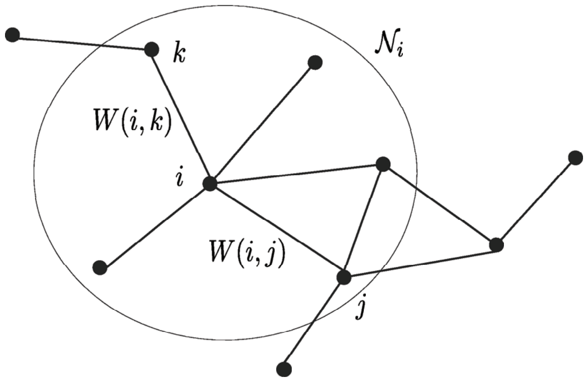


Figure 2.3: *Graph-based network model.*

In this work, we consider distributed linear iterations of the following form

$$z_{t+1}(i) = W(i,i)z_t(i) + \sum_{j \in \mathcal{N}_i} W(i,j)z_t(j) + \tilde{z}(i), \tag{2.8}$$

for $i = 1, \dots, m$, where $z_t(j)$ represents the value computed by sensor j at iteration t, and $\tilde{z}(i)$ denotes a constant scalar value (e.g., a bias term). Since the sensors communicate in each iteration t, we assume that they are synchronized. The parameters $W(i,j)$ denote the edge weights of $\mathcal{G}_s$. Since each sensor communicates only with its direct neighbors, $W(i,j) = 0$ when $(i,j) \notin \mathcal{E}_s$. The above iteration can be compactly written in the following form

$$z_{t+1} = Wz_t + \tilde{z}. \tag{2.9}$$

We call the matrix W that gathers the edge weights $W(i,j)$, as the weight matrix. Note that W is a sparse matrix whose sparsity pattern is driven by the network topology. We call such a matrix *network conformant*, since its sparsity structure conforms to the network topology.

Distributed averaging Let us see now how consensus can be employed for the problem of distributed averaging. Assume that initially each sensor i reports a scalar value $z_0(i) \in \mathbb{R}$. We denote by $z_0 = [z_0(1), \dots, z_0(m)]^\top \in \mathbb{R}^n$ the vector of initial values on the network. Denote by

$$\bar{z}_0 = \frac{1}{m} \sum_{i=1}^{m} z_0(i) \tag{2.10}$$

the average of the initial values of the sensors. The problem of distributed averaging therefore becomes typically to compute $\bar{z}_0$ *at each sensor* by distributed linear iterations of the form of (2.9), where $\tilde{z} = 0$. In other words, for this particular problem, we consider distributed linear iterations of the following form

$$z_{t+1} = Wz_t. \tag{2.11}$$

Iteration (2.11) converges to the average for every z_0 if and only if

$$\lim_{t \to \infty} W^t = \frac{\mathbf{1}\mathbf{1}^\top}{m}, \tag{2.12}$$

where **1** is the vector of ones [121]. Indeed, notice that in this case

$$z^* = \lim_{t\to\infty} z_t = \lim_{t\to\infty} W^t z_0 = \frac{\mathbf{1}\mathbf{1}^\top}{m} z_0 = \bar{z}_0 \mathbf{1}.$$

2.5.2 Accelerating the convergence

Now that the problem of distributed consensus has been introduced, we are ready to review the most relevant methods that attack the problem of increasing the convergence rate of distributed consensus. Several works have studied the convergence properties of distributed consensus algorithms. In particular, the authors in [121] and [50, 51] have shown that the convergence rate depends on the second largest eigenvalue of the network weight matrix, for fixed and random networks, respectively. This implies that optimizing the convergence rate of distributed consensus, is equivalent to an eigenvalue optimization problem. In this context, learning the optimal weight matrix can be typically formulated as a semi-definite program (SDP). First, L. Xiao and S. Boyd in [121] formulate the computation of the optimal weight matrix as an SDP in the case of fixed network topologies. Later, along the same lines, S. Kar and J. Moura in [51] use semi-definite programming to optimize the topology of random networks, where links may fail at random.

Notice that the standard distributed consensus as introduced in Section 2.5.1 above, does not use the memory of the sensors. It turns out that taking advantage of the memory of the sensors by applying extrapolation-based ideas, can provide better estimates. Hence, in this context, fast convergence can be achieved, even with fixed weights. Recent works have proposed to use the sensors' memory. The main idea in [4] is to update the value of each sensor by a convex combination of the value obtained by linear prediction (on its own previous values), and the standard consensus operation (i.e., aggregating neighbors' values). In a different context, the effect of quantization has been studied in [126] for average consensus problems. The authors propose scalar quantizers based on predictive coding. The predictive coding scheme relies upon linear prediction using the past values of the sensors.

Positioning In this thesis, we propose to use the memory of the sensors in order to accelerate the convergence of distributed consensus. The main idea is that each sensor updates periodically its own estimate using a linear combination of its own previous estimates. We show that this is equivalent to applying a polynomial filter on the weight matrix. The coefficients of the filter (i.e., weights of the linear combination) can be optimized by semi-definite programming, such that the convergence rate is maximized. We show in Chapter 6 that our polynomial filtering methodology can be combined with optimal weight matrices (such as those discussed above), resulting even in faster convergence rates.

Note also that while the method proposed in [4] derives the optimal convex combination parameter, the linear prediction coefficients are not optimized. This is to be contrasted to our SDP polynomial filtering methodology, where we establish the optimality of the polynomial coefficients. Finally, we show that distributed classification of multimedia data can benefit from the proposed fast consensus algorithms.

2.6 Summary

Based on the above facts, we would like to capitalize on the following points:

- Most existing dimensionality reduction methods target at either approximation or classification. Flexible methods are required that are able to address both criteria jointly and at the same time provide compact data representation.

- Existing methods are sub-optimal in computing the manifold distance. Clearly, there is a need for a novel method that is optimal and globally convergent.

- Current methods in SSL consider the generic scenario where unlabelled examples come from different classes. This necessitates the design of new classification algorithms that are able to exploit the fact that multiple observations of an object belong to the same class.

- Little attention has been given to the problem of distributed classification over ad-hoc networks. There is a need for methods addressing this problem, which should be moreover communication and cost efficient as well as robust to dynamic network topologies.

- Existing algorithms for fast distributed consensus do not exploit the memory of the sensors in an optimal way for accelerating its convergence rate.

Chapter 3

Flexible dimensionality reduction

3.1 Introduction

In this chapter, we introduce a dimensionality reduction algorithm[1] that identifies relevant multidimensional patterns in multimedia signals, which represent an effective trade-off between approximation and classification performance [59]. We formulate the dimensionality reduction problem as a matrix factorization problem, where the basis vectors are extracted from a redundant and structured dictionary of localized basis functions. The design of such a dictionary provides direct control on the shape and the properties of the basis functions, such as spatial locality and sparse support. Spatial locality typically characterizes those signals whose energy and support do not cover the whole signal area, but it is rather concentrated around local regions. It naturally permits to incorporate a priori and application-driven knowledge into the learning process.

In order to solve the matrix factorization problem, we build on greedy pursuit algorithms from simultaneous sparse approximations [111] that have been previously proposed in the context of joint signal compression. These algorithms proceed by selecting sequentially the basis vectors from the dictionary in order to provide the best match to the training data. In this chapter, we extend the simultaneous sparse approximations algorithms to relevant features extraction in classification problems. We build on [55] and design a greedy algorithm for supervised dimensionality reduction, which exploits available class labels information and uses the inter-class variance as a class separability cost function. The selection of the basis functions from the dictionary is thus driven by a trade-off between the approximation error (for efficient compression) and class separability (for good classification). Based on these properties, we call the proposed framework *semantic approximation.* The convergence rate of the supervised greedy decomposition algorithm is therefore penalized by a class separability constraint, which permits to achieve efficient and robust classification, jointly with effective signal approximation.

The novel dimensionality reduction algorithm is applied to image classification problems, in the context of handwritten digit and face recognition. The features selected by the supervised algorithm are shown to provide jointly interesting approximation and classification performance. When combined with Linear Discriminant Analysis (LDA), the dimensionality

---

[1] Part of this chapter has been published in: E. Kokiopoulou and P. Frossard, "Semantic coding by supervised dimensionality reduction", IEEE Transactions on Multimedia, vol. 10, no. 5, pp. 806-818, August, 2008.

reduction strategy even reaches classification performances that are competitive with state of the art methods in 2D and 3D face recognition. At the same time, the extracted features can lead to efficient compression strategies since they are chosen from a pre-defined parametric dictionary of functions. It certainly represents one of the main advantages of the proposed method compared to state of the art subspace methods whose signal-specific features can be as difficult to code as the original signal. Compression and classification can be performed jointly, without important performance penalty with respect to expensive disjoint solutions.

In summary, the contribution of this chapter amounts to: (i) formulating the joint approximation and feature extraction problem as a supervised dimensionality reduction algorithm based on simultaneous sparse approximation (ii) designing a greedy dimensionality reduction algorithm that reflects the trade-off between compression and feature extraction, as desired in current media processing and mining systems, and (iii) application of the proposed solution to image classification, as well as 2D and 3D face recognition problems.

The rest of the chapter is organized as follows. In Section 3.2, we discuss our semantic approximation framework for dimensionality reduction using redundant dictionaries. The supervised method that jointly targets efficient approximation and classification is presented in Section 3.3, and its convergence properties are discussed. Next, Section 3.4 presents the properties of the proposed dimensionality reduction scheme to images. Then, Section 3.5 provides the experimental results of its application to 2D and 3D face recognition, which shows that classification performance is competitive with state of the art solutions, while it additionally provides compact signal representation. We provide some additional related work in Section 3.6 and finally discuss our concluding remarks in Section 3.7.

3.2 Dimensionality reduction using simultaneous sparse approximations

We propose to formulate dimensionality reduction as a matrix factorization problem, where the basis vectors are extracted from a generic dictionary of localized basis functions. We assume the existence of a redundant dictionary $\mathcal{D}$ that spans the Hilbert space $\mathcal{H}$ of the data of interest. Redundancy offers flexibility in the construction of the dictionary, and in general improves the approximation rate, especially for multidimensional data. A redundant dictionary is an overcomplete basis in the sense that it includes a number of vectors that is larger than the dimension of the subspace. The elements of the dictionary, which are indexed by $\gamma \in \Gamma$ i.e.,

$$\mathcal{D} = \{\phi_\gamma,\ \gamma \in \Gamma\}, \tag{3.1}$$

are usually called *atoms*. The atoms have unit norm i.e., $\|\phi_\gamma\|_2 = 1,\ \forall \gamma \in \Gamma$, where $\|\cdot\|_2$ denotes the L2 norm. It is important to note that we do not set any particular assumption on the dictionary design, and that the following analysis holds for any redundant dictionary. The only assumption that we make is that the dictionary spans the input space $\mathcal{H}$ (i.e., the basis is (at least) complete).

Then, we consider a data sample s_i as an element of $\mathcal{H} \subseteq R^m$. The training data forms a data matrix

$$S = [s_1, s_2, \dots, s_n] \in \mathbb{R}^{m \times n}, \tag{3.2}$$

Algorithm 1 The SOMP algorithm

```
1:  Input:
    $S \in R^{m \times n}$: data matrix
    tol: approximation error tolerance
    $N_a$: number of atoms
2:  Output:
    $\Psi$: Set of selected atoms,
    $A$: approximation
    $R$: residual matrix
3:  Initialization: $R_0 = S$, $\Psi = []$, $t = 1$.
4:  Main iteration:
5:  Find index $\gamma_t$ which solves the optimization problem
        $$\gamma_t = \arg\max_{\gamma \in \Gamma} \|R_t^\top \phi_\gamma\|_1$$
6:  Augment $\Psi = [\Psi,\ \phi_{\gamma_t}]$.
7:  Compute an orthonormal basis $V = [v_1, \ldots, v_t]$ of the span$\{\Psi\}$.
8:  Compute the orthogonal projector $P_t = V_t V_t^\top$ on the span$\{\Psi\}$.
9:  Compute the new approximation and residual
    $A_t = P_t S$
    $R_t = (I - P_t)S$.
10: if $\|R\|_F \leq$ tol or $t = N_a$ then
11:     stop.
12: else
13:     increment iteration $t = t + 1$, and go to step (2).
14: end if
```

where s_i denotes the ith column of S. For dimensionality reduction, our goal is to decompose S in the following form

$$S = \Psi C, \ \Psi \in \mathbb{R}^{m \times r}, \ C \in \mathbb{R}^{r \times n}, \tag{3.3}$$

where Ψ are the basis vectors drawn from the dictionary and C are the corresponding coefficients. In other words, every column of S is represented in the same set of basis functions Ψ using different coefficients. This is a dimensionality reduction step, where each data sample (column of S) is represented in the subspace spanned by the columns of Ψ, using only $r \ll m$ coefficients.

If the columns of Ψ are spatially localized basis functions then the decomposition given in Eq. (3.3) results in a parts-based representation. Note that the characteristics of the dictionary determines the properties of Ψ. Therefore, one has direct control on the shape and the properties of the basis functions due to the flexible design of the dictionary. This is to be contrasted with the NMF methods, where one has only implicit control on the properties of the basis functions.

If we denote by $\|\cdot\|_F$ the Frobenius norm, then we formulate the above problem as the following optimization problem.

Optimization problem: **OPT1**
$\min_{\Psi,C} \|S - \Psi C\|_F^2$
subject to
$\Psi \subseteq \mathcal{D}$.

In order to solve OPT1 one may employ greedy algorithms that have been proposed for simultaneous sparse signal approximations [111, 110, 66, 15] in the context of joint signal compression. We have chosen to use the Simultaneous Orthogonal Matching Pursuit (SOMP) algorithm [111], since it lends itself as an efficient algorithm for solving OPT1 in practice. SOMP is not prone to local minima and not sensitive to initializations.

SOMP is a generalization of Matching Pursuit [74] to the case of joint signal compression, and it can be extended directly to dimensionality reduction. It is a greedy algorithm that extracts a subset Ψ of the dictionary $\mathcal{D}$, such that all the columns of S are simultaneously approximated. Initially, SOMP sets the residual matrix $R = S$. The atom from the dictionary that best matches all the vectors, is selected. The algorithm then updates the residual matrix by projection on its orthogonal complement, i.e.,

$$R = (I - \phi_\gamma \phi_\gamma^\top)S,$$

where $I - \phi_\gamma \phi_\gamma^\top$ is the projector on the orthogonal complement of $\mathrm{span}\{\phi_\gamma\}$. The above step will remove the components of ϕ_γ from R. The same procedure is repeated iteratively on the updated residual matrix. Thus, it greedily selects at step t, the best matching atom ϕ_{γ_t} by solving the simple optimization problem

$$\gamma_t = \max \arg_{\gamma \in \Gamma} \|R^\top \phi_\gamma\|_1, \tag{3.4}$$

and includes the selected ϕ_{γ_t} in Ψ. In the above equation, $\|\cdot\|_1$ denotes the L1 norm. The residual matrix is updated by $R = (I - P)S$, where P is the orthogonal projector on the $\mathrm{span}\{\Psi\}$. The main steps of the SOMP algorithm are summarized in Algorithm 1. We should mention that the Orthogonal Matching Pursuit (OMP) converges in a finite number of iterations [73, Sec.9.5.3] since the norm of the residual is decreasing strictly monotonically in each step. This can be generalized to the case of SOMP [15]. Note in passing that the L2 norm could be alternatively used in (3.4). However, we consider both L1 and L2 choices equivalent and we will not pursue this issue further.

It can be noted that other methods for simultaneous approximation could be alternatively used for dimensionality reduction with redundant dictionaries. Interestingly, an algorithm called M-OMP, which is identical to SOMP, has been independently proposed in [15]. However, for notational convenience we will keep using the term SOMP while referring to any of these two algorithms.

Note finally that a simpler Simultaneous Matching Pursuit (SMP) algorithm can be derived from SOMP by modifying the residual update i.e., Line 9 in Algorithm 1. In SMP, the residual update at iteration t, would simply remove the contribution of the selected atom

$$r_t^{(i)} = r_{t-1}^{(i)} - \langle r_{t-1}^{(i)}, \phi_{\gamma_t} \rangle \phi_{\gamma_t},$$

where $r_t^{(i)}$ denotes the ith column of R_t and corresponds to the residual of signal s_i. However, in contrast to SOMP, there is no guarantee that SMP will converge in a finite number of iterations.

3.3 Supervised dimensionality reduction

3.3.1 Supervised atom selection

We now extend the previous algorithm to classification problems, and we propose a supervised learning solution where class labels are available a priori [119]. In order to develop a supervised dimensionality reduction method, we modify the objective function in OPT1 by including an additional term that encourages the separability between different classes. First, let us denote the number of classes by c and assume without loss of generality that

$$S = [S^{(1)}, \ldots, S^{(c)}] \in \mathbb{R}^{m \times n}, \tag{3.5}$$

where $S^{(i)} \in R^{m \times n_i}$ denotes the data samples that belong to the ith class of cardinality n_i. Then we formulate a supervised dimensionality reduction problem by modifying the optimization problem OPT1 as follows.

Optimization problem: **OPT2**
$\min_{\Psi,C} \|S - \Psi C\|_F^2 - \lambda J(\Psi)$
subject to
$\Psi \subseteq \mathcal{D}$.

In the above optimization problem $J(\Psi)$ denotes the cost function that captures the separability of different classes. The scalar λ drives the trade-off between the approximation error and the class separability. In order to solve OPT2, we propose a new algorithm where the atom selection step is modified in order to include the class separability term. The intuition is that in each step, the algorithm should select the atom that best approximates all data and also discriminates between data samples of different classes. We call the modified supervised algorithm SAS (i.e., Supervised Atom Selection).

The separability cost function is chosen to capture the projected between-class variance. By the projected class variance, we mean the restriction of the scatter matrix S_b, on the candidate atom ϕ. This is given as $\phi^\top S_b \phi$, where the scatter matrix S_b is defined as

$$S_b = \frac{1}{n} \sum_{i=1}^{c} n_i (\mu^{(i)} - \mu)(\mu^{(i)} - \mu)^\top. \tag{3.6}$$

In the above formula we have introduced

$$\mu^{(i)} = \frac{1}{n_i} \sum_{j=1}^{n_i} s_j^{(i)} \tag{3.7}$$

which denotes the centroid of the ith class and

$$\mu = \frac{1}{n} \sum_{j=1}^{n} s_j \tag{3.8}$$

which represents the global centroid. The notation $s_j^{(i)}$ denotes the jth sample of the ith class. Note that we can write $S_b = G_b G_b^\top$, where $G_b \in \mathbb{R}^{m \times c}$ is defined as

$$G_b = \frac{1}{\sqrt{n}} [\sqrt{n_1}(\mu^{(1)} - \mu), \ldots, \sqrt{n_c}(\mu^{(c)} - \mu)].$$

Algorithm 2 The SAS algorithm

```
1: Input:
   $S \in R^{m \times n}$: data matrix
   tol: approximation error tolerance
   $N_a$: number of atoms
2: Output:
   $\Psi$: Set of selected atoms,
   $A$: approximation
   $R$: residual matrix
3: Initialization: $R_0 = S$, $\Psi = []$, $t = 1$.
4: Main iteration:
5: Find index $\gamma_t$ which solves the optimization problem
   $$\gamma_t = \arg\max_{\gamma \in \Gamma} \|R_t^\top \phi_\gamma\|_1 + \lambda J(\phi_\gamma)$$
6: Augment $\Psi = [\Psi,\ \phi_{\gamma_t}]$.
7: Compute an orthonormal basis $V = [v_1, \dots, v_t]$ of the span$\{\Psi\}$.
8: Compute the orthogonal projector $P_t = V_t V_t^\top$ on the span$\{\Psi\}$.
9: Compute the new approximation and residual
   $A_t = P_t S$
   $R_t = (I - P_t)S$.
10: if $\|R\|_F \leq$ tol or $t = N_a$ then
11:    stop.
12: else
13:    increment iteration $t = t + 1$, and go to step (2).
14: end if
```

The above implies that the scatter matrix is symmetric and positive semi-definite. Note that with the definition of G_b the projected between-class variance is $\phi^\top S_b \phi = \phi^\top G_b G_b^\top \phi = \|G_b^\top \phi\|^2$.

We define the class separability term as

$$J(\phi) = \|G_b^\top \phi\|_2^2 - \kappa \|\Psi^\top \phi\|_2^2. \tag{3.9}$$

The first term captures the discriminant properties of the chosen atom, and the second term ensures that the selected atom is as orthogonal as possible with respect to Ψ, which represents the previously selected atoms. The scalar $\kappa > 0$ determines the significance of the discriminant value of ϕ relatively to its orthogonality level with respect to the previous atoms. If $\kappa = 0$ then the algorithm selects an atom that is certainly discriminant (due to the first term) but possibly very similar to the previous one, depending on the value of λ. Thus, the second term is necessitated by the greedy nature of the expansion and the redundancy of the dictionary. The role of κ however depends on the value of λ. In particular, if λ is very small or zero, then the role of κ is deemphasized. The selected basis vectors are therefore not exactly orthogonal but semi-orthogonal and the level of their orthogonality is driven by κ. Finally, we can write the optimization problem that we solve in each step of the supervised SAS algorithm as,

$$\gamma_t = \arg\max_{\gamma \in \Gamma} \left(\|R^\top \phi_\gamma\|_1 + \lambda J(\phi_\gamma) \right). \tag{3.10}$$

Note that more complex cost functions could be proposed, but this goes beyond the scope of the present chapter that rather focuses on the semantic approximation framework. Algorithm 2 summarizes the main steps of SAS.

3.3.2 Analysis of SAS

The residual of SOMP has been shown to converge to zero as the number of iteration increases [15]. In SAS, the class separability is strengthened, which results in an effective algorithm for classification tasks but also introduces a penalty on the convergence rate of the SAS algorithm. In the extreme case where the penalty term is very large, it can even cause stagnation of the progress of the residual. This is explained by the following proposition, which is rather intuitive but provided here for the sake of completeness.

Proposition 1. *The residual of the SAS algorithm decreases strictly monotonically in each step t, if the following condition is satisfied,*

$$\|R_t^\top \phi_{\gamma_t}\|_2^2 > 0, \quad \forall t. \tag{3.11}$$

Proof. Assume that in iteration t, the condition (3.11) is violated and the selected atom ϕ_{γ_t} is orthogonal to all columns of the residual matrix. In other words,

$$\|R_t^\top \phi_{\gamma_t}\|_2^2 = 0. \tag{3.12}$$

First, note that condition (3.12) implies that

$$\|R_t^\top v_{t+1}\|_2^2 = 0. \tag{3.13}$$

Indeed, it holds that

$$v_{t+1} = \phi_{\gamma_t} - \sum_{i=1}^{t} \zeta_i v_i, \tag{3.14}$$

where $\zeta_i = v_i^\top \phi_{\gamma_t}$ are the weights of the linear combination that make v_{t+1} orthogonal to $v_1, \dots, v_t$. Note that they can be also computed using the Gram Schmidt orthogonalization process [29]. At the same time $R_t \perp \operatorname{span}\{v_1, \dots, v_t\}$, due to the construction of the algorithm. Combined with Eq. (3.14), it leads to the condition given in Eq. (3.13).

Then, we call $V_{t+1} = [v_1, \dots, v_{t+1}]$ an orthogonal basis for the $\operatorname{span}\{\Psi \bigcup \phi_{\gamma_t}\}$ obtained in the first $t+1$ iterations. The orthogonal projector on the $\operatorname{span}\{\Psi \bigcup \phi_{\gamma_t}\}$ is

$$P_{t+1} = V_{t+1} V_{t+1}^\top = \sum_{i=1}^{t+1} v_i v_i^\top = P_t + v_{t+1} v_{t+1}^\top. \tag{3.15}$$

Using the above formula, we observe that

$$\begin{aligned} R_{t+1} &= S - P_{t+1} S = (I - P_{t+1}) S \\ &= (I - P_t - v_{t+1} v_{t+1}^\top) S \\ &= R_t - v_{t+1} v_{t+1}^\top S. \end{aligned} \tag{3.16}$$

However it holds that $v_{t+1} v_{t+1}^\top S = 0$ because

$$\begin{aligned} v_{t+1} v_{t+1}^\top S &= v_{t+1} v_{t+1}^\top (R_t + A_t) \\ &= v_{t+1} v_{t+1}^\top R_t + v_{t+1} v_{t+1}^\top A_t \\ &= 0, \end{aligned} \tag{3.17}$$

where the first term is zero because of Eq. (3.13). The second term cancels out since A_t belongs to the span$\{V_t\}$ and $v_{t+1} \perp$ span$\{V_t\}$ (see also Eq. (3.14)). In this case we therefore have $R_{t+1} = R_t$ due to Eqs. (3.16) and (3.17). We conclude that if condition (3.11) is violated, the progress of the residual stops. □

In summary, SAS converges in a finite number of steps, and is not sensitive to any particular initialization. However, the approximation rate is now driven by λ, which controls the trade-off between approximation and extraction of discriminative features. If one wants to avoid the possibility of residual stagnation, the choice of λ has to ensure that the condition given in Proposition 1, is satisfied. In particular, the selected atoms have to participate to the approximation of the signal. One could devise an automatic way of tuning λ for avoidance of rare residual stagnation. Starting with an initial value of λ, violation of condition (3.11) is checked. If the condition is violated, λ is divided by 2 and the same process is repeated until the condition is satisfied. In the worst case, λ becomes 0 and the selected atom satisfies (3.11). Such an atom certainly exists due to the fact that the dictionary spans the signal space.

3.4 Semantic approximation of images

3.4.1 Dictionary design

We first discuss in detail how one may build redundant dictionaries for dimensionality reduction in the context of digital images. Driven by the need for efficient compression, we propose to use a structured dictionary $\mathcal{D}$ that is built by applying geometric transformations to a generating mother function ϕ. In such a case, efficient coding simply proceeds by describing each basis vector or atom with the parameters of these transformations [45]. The atom parameters are carefully sampled such that the resulting dictionary consists an overcomplete basis of the image space. The sampling of the atom parameters typically drives the dictionary size and therefore its redundancy. A geometric transformation $\gamma \in \Gamma$ is represented by a unitary operator $U(\gamma)$ and in the simplest case it may be one of the following three types.

- *Translation* by $\vec{b} = [b_1\ b_2]^\top$. $U(\vec{b})$ moves the generating function across the image
$$U(\vec{b})\phi(x,y) = \phi(x - b_1, y - b_2).$$
- *Rotation* by θ. $U(\theta)$ rotates the generating function by angle θ i.e.,
$$\begin{aligned} U(\theta)\phi(x,y) &= \phi(x',y'). \\ x' &= \cos(\theta)x + \sin(\theta)y \\ y' &= \cos(\theta)y - \sin(\theta)x \end{aligned}$$
- *Anisotropic scaling* by $\vec{a} = [a_1\ a_2]^\top$. $U(\vec{a})$ scales the generating function anisotropically in the two directions i.e.,
$$U(\vec{a})\phi(x,y) = \phi(\frac{x}{a_1}, \frac{y}{a_2}).$$

Composing all the above transformations yields a transformation $\gamma = \{\vec{b}, \vec{a}, \theta\} \in \Gamma$. Finally, an atom in the structured dictionary

$$D = \{U(\gamma)\phi,\ \gamma \in \Gamma\}$$

is built as

$$\begin{aligned} U(\gamma)\phi(x,y) &= \phi(x',y'), \\ x' &= \frac{\cos(\theta)(x-b_1)+\sin(\theta)(y-b_2)}{a_1} \\ y' &= \frac{\cos(\theta)(y-b_2)-\sin(\theta)(x-b_1)}{a_2}. \end{aligned}$$

According to the above, notice that if we know ϕ, then an atom in a structured dictionary can be completely described by its corresponding transformation parameters γ. It is exactly this handy representation property that permits the use of structured redundant dictionaries for efficient image coding. This is to be contrasted to the basis vectors of NMF that are as hard to compress as the initial images in S. Moreover, the properties of ϕ completely determine the geometric properties (e.g., shape) of the basis vectors. This flexibility in the design of ϕ further permits to incorporate a-priori knowledge in the learning process.

In image classification applications, we consider three different structured dictionaries generated by different ϕ, where ϕ is

- *Gaussian* function:
$$\phi(x,y) = \frac{1}{\sqrt{\pi}}\exp(-(x^2+y^2)). \tag{3.18}$$

- *Anisotropic refinement* (AnR) function. This generating function has an edge-like form and has been successfully used for image coding [45]. It is Gaussian in one direction and the second derivative of Gaussian in the orthogonal direction. It can be mathematically expressed as,
$$\phi(x,y) = \frac{2}{\sqrt{3\pi}}(4x^2-2)\exp(-(x^2+y^2)). \tag{3.19}$$

- *Gabor* function. This generating function is very popular in face recognition. It consists of a Gaussian envelope modulated by a complex exponential. We have used the real part of a simplified version of the Gabor function,
$$\phi(x,y) = \cos(2\pi x)\exp(-(x^2+y^2)). \tag{3.20}$$

3.4.2 Implementation Issues

We discuss here the computational aspects of the proposed methods. Note that one of the advantages of structured dictionaries lies in the fact that they enable a fast FFT-based implementation of the SOMP algorithm for images. Recall that in each step of SOMP, we need to compute the inner product of the candidate atom with the residual signals. In practice we construct the atoms only in their centered position. The inner product of a residual signal r with all translated versions of an atom g, is computed via 2D convolution which can be effectively computed using 2D FFT. Using this computational trick the algorithm becomes more computationally efficient, even in the context of high dimensional signals, like digital images.

There is also another computational trick that can be employed in order to save significant computation time by trading memory utilization. Observe that in SOMP the residual $r_t^{(i)}$

of the ith data sample s_i in the tth iteration, can be alternatively expressed in the following form,

$$r_t^{(i)} = s_i - \sum_{k=1}^{t-1} \alpha_k^i \phi_k, \ \forall i = 1, \ldots, n,$$

for some scalars α_k^i that are determined by the orthogonal projection process in order to minimize the residual of approximating s_i from the span of ϕ_k's. In the next iteration $t+1$, SOMP will need to compute the inner product of each residual vector with each candidate atom ϕ from the dictionary. It other words, it will compute

$$\langle r_t^{(i)}, \phi \rangle = \langle s_i, \phi \rangle - \sum_{k=1}^{t-1} \alpha_k^i \langle \phi_k, \phi \rangle, \ \forall i = 1, \ldots, n.$$

The above observation suggests that the computation of $\langle s_i, \phi \rangle$ needs to be performed only once before the first iteration of the algorithm (off-line), since it will be used in each subsequent iteration. Furthermore, at the kth iteration, the projection of the selected atom ϕ_k on the dictionary (i.e., $\langle \phi_k, \phi \rangle, \forall \phi \in \mathcal{D}$) can be computed only once, stored and then re-used for the evaluation of all n residuals at the next iteration $k+1$. Using the above tricks, the SOMP methods become computationally attractive since the main computational cost is amortized among all n images.

Overall, we should note that the feature extraction part in semantic approximation is the most computationally intensive task. Since the main operations involve FFTs and inner product calculations, this task can become very efficient by careful design. For instance, one may organize the data or the dictionary in a more structured form in order to get a feature extraction algorithm of reduced complexity (see for example the Tree-based Pursuit algorithm [48], the Sparse Greedy Matrix Approximation [101, 96] and references therein). Alternatively, one may use specific generating functions (such as Haar or box-like basis functions) that are known to result in very efficient inner product calculations. Thus, the proposed semantic approximation principle is certainly applicable in media system architectures.

3.4.3 Experimental setup

The construction of the atoms in each dictionary proceeds by sampling uniformly 10 orientation angles in $[0, \pi]$ and 5 logarithmically equi-distributed scales in $[1, N/6]$ horizontally and $[1, N/4]$ vertically, where N is the image size. The translation parameters are all possible pixel locations. For our experimental comparisons, we use the provided implementations of NMF and LNMF in `nmfpack` [42], which is a MATLAB software package developed by P. Hoyer.

In the learning stage that produces the matrix Ψ of basis vectors, we use 5 samples per class. In all experiments that follow, the parameter κ was set to 0.01. Recall that κ is the parameter that determines the trade-off between discrimination capability of each atom and its orthogonality towards the previous atoms. We have observed experimentally that $\kappa = 0.01$ works reasonably well and we use the same value of κ in all experiments. We should also mention that in the recognition experiments the emphasis is on the classification performance. For that reason, in each step of the SAS algorithm the best atom is selected by pure discrimination criteria i.e., using only the right-hand side term $J(\phi)$ shown in Eq. (3.9). This is equivalent to setting λ to a large value.

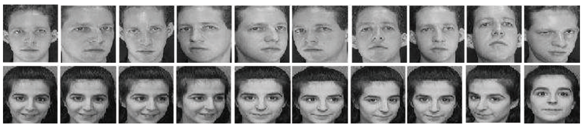


Figure 3.1: *Sample face images from the ORL database. There are 10 available facial expressions and poses for each subject.*

As it is common practice for classification, each training signal s_i is projected using the basis vectors Q, where Q denotes Ψ for the SOMP or SAS methods and W for the NMF methods (see Section 2.2). In particular, we project the samples in the reduced space using the transpose of Q i.e.,

$$y_i = Q^\top s_i, \quad i = 1, \ldots, n.$$

Then, classification is accomplished in the reduced space by nearest neighbor (NN) classification. The test signal s_t is also projected by $y_t = Q^\top s_t$ and then classified and assigned the label of its nearest neighbor among all the training signals. Note in passing that the selected atoms in SOMP are linearly independent due to the orthogonal projection in each step. This is also the case for SAS due to the orthogonality term in eq. (3.9). Hence, redundancy issues do not interfere with the nearest neighbor computation in the reduced space.

We measure performance in terms of classification error rate, which is the percentage of the test samples that have been misclassified. In our experiments, we use the following data sets:

- **Handwritten digit image collection** We use the handwritten digit collection that is publicly available at S. Roweis web page[2]. This collection contains 20 × 16 bit binary images of “0” through “9”, and each class contains 39 samples. We form the training set by a random subset of 10 samples per class and the remaining 29 samples are assigned in the test set.

- **ORL face database** The ORL (formerly Olivetti) database [91] contains 40 individuals and 10 different images for each individual including variation in facial expression (smiling/non smiling) and pose. Figure 3.1 illustrates two sample subjects of the ORL database along with variations in facial expression and pose. The size of each facial image is downsampled to 28 × 23 for computational efficiency. We form the training set by a random subset of 5 different facial expressions/poses per subject and use the remaining 5 as a test set.

- **CBCL face database** The CBCL face database [26] consists of 2,429 facial images of size 19 × 19. Note that for this data set, there are no class labels available for the individuals.

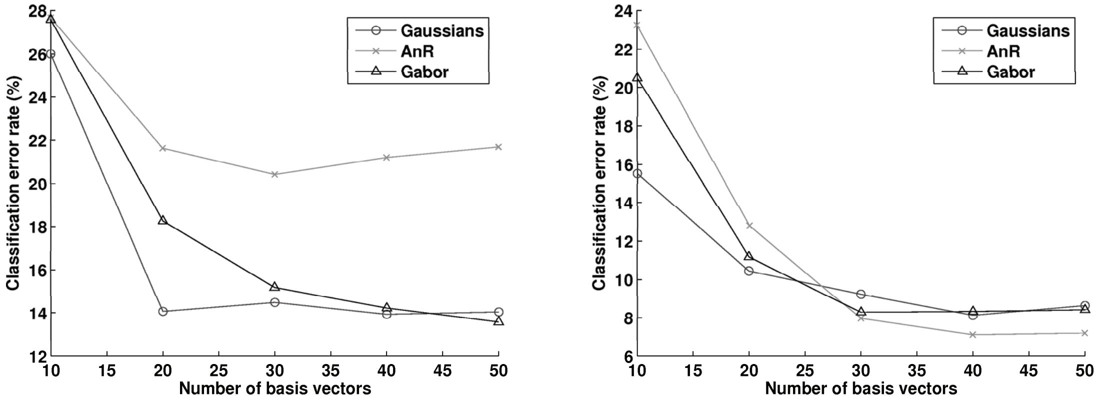
All data sets that are used in the experimental evaluation along with their main properties, are summarized in Table 3.1.

---

[2] http://www.cs.toronto.edu/∼roweis/data/binaryalphadigs.mat

	Size of S	n_i
Handwritten digits	320×390	39
ORL faces	644×400	10
CBCL faces	361×2429	-

Table 3.1: *The data sets used in the experimental evaluation, where* n_i *denotes the number of samples per class.*



(a) Digits data set

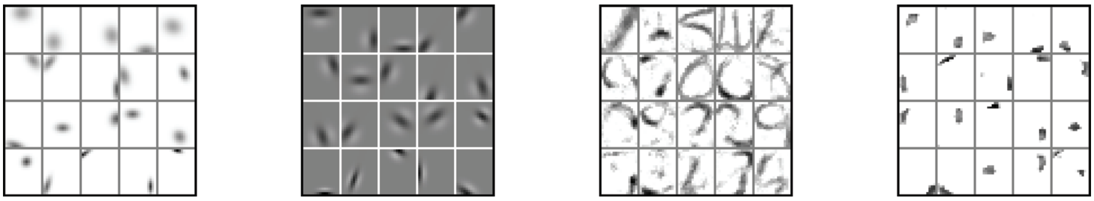
(b) ORL face data set

Figure 3.2: *Impact of different dictionaries on the classification performance.*

Dictionary choice First, we investigate the impact of the dictionary on the classification performance, by comparing the effectiveness of the three generating functions presented earlier. We run SOMP on both digit and ORL face data sets and compare the classification performance with respect to different dimensions $r = [10 : 10 : 50]$ (in MATLAB notation) of the reduced space. Sub-figures 3.2(a) and 3.2(b) depict the classification error rates obtained via the different dictionaries, for the digits and the face data set respectively. Note that for each value of r we report the average classification error rate across 100 random realizations of the training/test set.

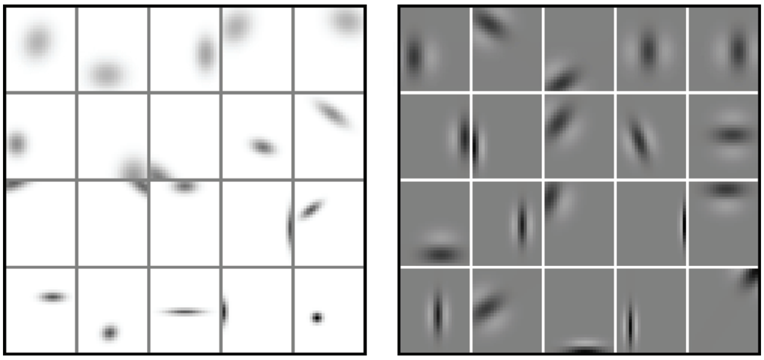
Observe that for the digits data set, the dictionary built from Gaussian functions is the best performer among the three candidates under test. However, for the face data set the behavior is quite different and the AnR dictionary seems to be competitive and even superior to the other dictionaries, especially for large dimensions. This is likely due to the fact that the AnR atoms can represent the edge-like fine details of facial characteristics like the eyes or the mouth, for example. In the following experiments, we have therefore chosen to use the Gaussian dictionary for the digit data set and the AnR dictionary for the face data sets.

We also propose a simple comparison with orthogonal dictionaries based on Discrete Cosine Transform (DCT) functions, as they represent the most commonly used features in methods that perform learning tasks in the compressed domain. We compare Discrete Cosine Transform (DCT) features with Gaussian features using the handwritten digits data set. We extract the DCT features by dividing the image in blocks of size 8-by-8 and using 2D-DCT in each block. Then, we keep the most important coefficients which are those residing in the

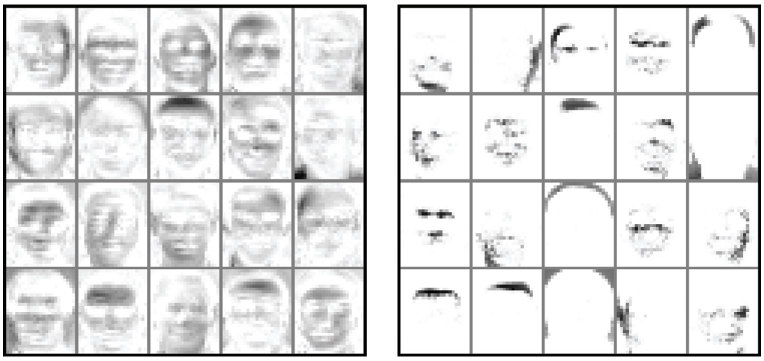


(a) Gaussian atoms (b) AnR atoms (c) NMF (d) LNMF

Figure 3.3: *Recovered basis vectors from the handwritten digit collection.*



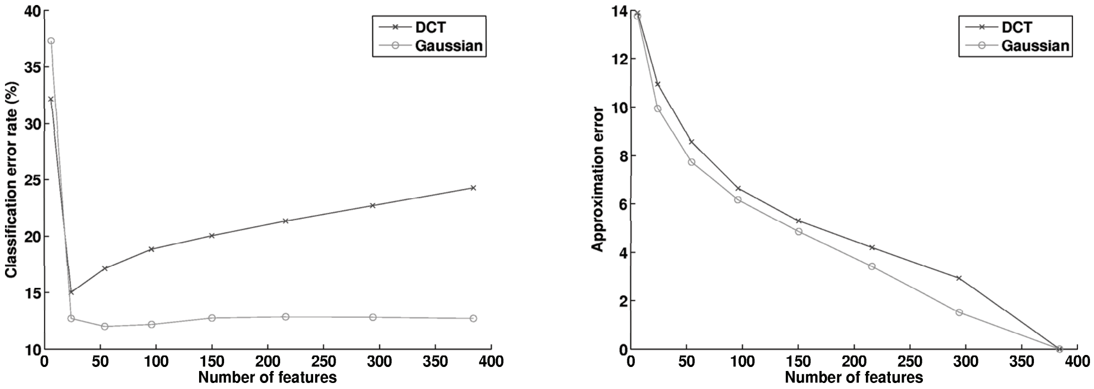
(a) Gaussian atoms (b) AnR atoms



(c) NMF (d) LNMF

Figure 3.4: *Recovered basis vectors from the ORL face data set.*

left upper (square) part of the block of certain size, say n_f. We report the classification error rate and the approximation error versus different number of features, produced by varying n_f from 1 up to 8. We measure the approximation error by computing the Frobenius norm of the residual matrix i.e., $\|S - \hat{S}\|_F$, where $\hat{S}$ denotes the reconstructed data matrix from the number of features that are available. Figure 3.5 illustrates the classification error rates and the approximation errors for both DCT and Gaussian features. Although DCT features work



(a) Classification error rate

(b) Approximation error

Figure 3.5: *DCT vs Gaussian features on the digits data set.*

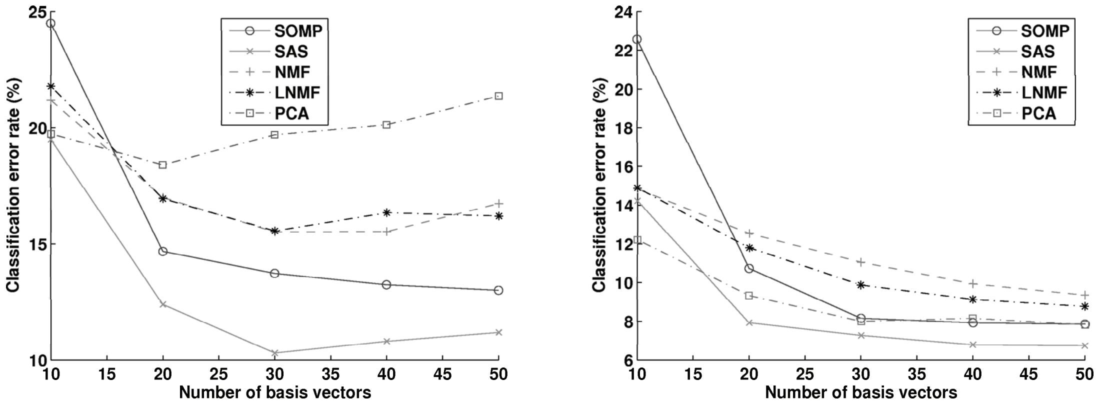
nicely for compression purposes, they are not optimal for classification purposes. Unsurprisingly, the features provided by the standards may not be optimal for the application at hand, since they have been optimized with respect to compression performance. Note however that this example is certainly not conclusive, and it does not exclude the use of DCT features for all applications. Our generic semantic approximation methodology however revisits compression from the viewpoint of the subsequent learning task, by performing both compression and feature extraction jointly and flexibly.

3.4.4 Classification performance

We analyze and compare the classification performance obtained by the proposed algorithms with several variants of parts-based dimensionality reduction algorithms. The basis functions obtained respectively from SOMP, NMF and LNMF algorithms are given in Figures 3.3 and 3.4, for the digits and faces data sets, respectively. The basis functions in sub-figures 3.3(a) and 3.4(a) are obtained from SOMP using the Gaussian dictionary. Similarly, the basis function in panels 3.3(b) and 3.4(b) are obtained from SOMP using the AnR dictionary. The figures also depict the corresponding recovered basis functions from NMF and LNMF. Note that the features obtained from NMF are not localized and seem to be of global support. On the contrary, the features of LNMF are spatially localized and for the digits data set they seem quite similar to the Gaussian atoms.

We now compare SOMP and SAS with NMF, LNMF and PCA in terms of classification performance. We compare with the above methods since they provide a low-rank approximation of the data and they are closely related to the proposed algorithms. In both data sets, we experiment with the dimension of the reduced space $r = [10 : 10 : 50]$ and in the classification experiments, for each value of r, we report the classification performance in terms of average error rate across 50 random realizations of the training/test set.

Figure 3.6(a) first depicts the average classification error rate for various values of the dimension r of the reduced space, for the handwritten digit image recognition task. The average is computed over 50 random realizations of the training/test set, where we use the Gaussian dictionary for SOMP and SAS. We observe that both proposed algorithms are superior to



(a) Digits data set

(b) ORL face data set

Figure 3.6: *Image recognition experiments.*

the NMF algorithms. Furthermore, the SAS method seems to outperform SOMP, mainly due to its supervised nature. Notice also that PCA does not have satisfactory performance for this data set. This is due to the fact that the basis vectors of PCA are holistic and of global support. Hence, they have trouble to capture the geometric structure of the handwritten digits that are of localized and sparse support.

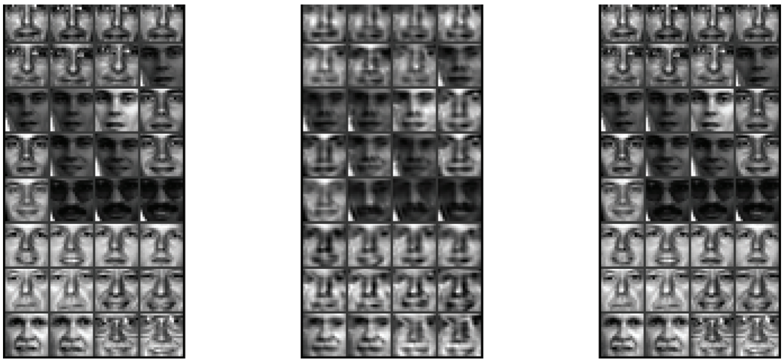
Then, Figure 3.6(b) depicts the average classification error rate across 50 random realizations of the training/test set for the face recognition task, measured on the ORL data set. Recall that for this data set we use the AnR dictionary, in both proposed algorithms. Notice that SAS and SOMP with PCA outperform the NMF methods. Observe also that for small dimensions r of the reduced space, the performance of SOMP is poor. However, as r increases the SOMP method becomes more discriminant and finally superior to the NMF methods. This can be explained by the greedy nature of SOMP. In the first steps, SOMP usually select atoms of large scale in order to reduce quickly the approximation error, but that do not consist in highly discriminating functions. The large scale atoms typically correspond to low frequency information which may not contribute a lot to the classification task.

Note finally that the authors in [118, Fig. 5] report the performance of their proposed (supervised) Fisher NMF (FNMF) on the ORL database with the same experimental setup as here (see the ORL description in Sec. 3.4.3). Hence it is possible to compare directly the performances of SOMP and SAS with that of FNMF on this database. From their reported results, one may observe that (i) SOMP and PCA compete with FNMF and (ii) SAS seems to slightly outperform FNMF.

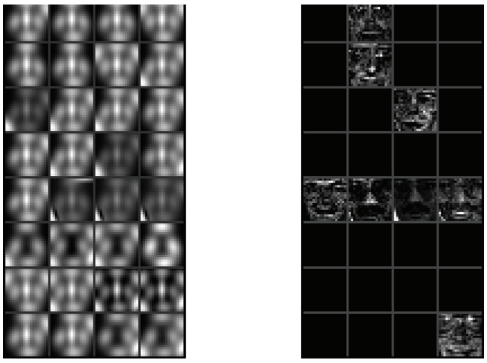
3.4.5 Approximation performance

This section presents a few results that illustrate the approximation performance of both NMF and SOMP algorithms. Figure 3.7 illustrates facial images from the CBCL database along with reconstructed images for the NMF solution (50 vectors), and the SOMP algorithm (50 Gaussian atoms). It also represents the reconstructed images when the coefficients of the vectors have been quantized uniformly with a step size q, before reconstruction.

From Figures 3.7(b) and 3.7(c) it can be observed that the approximation performances



(a) Original (b) SOMP (c) NMF



(d) SOMP, $q = 0.3$ (e) NMF, $q = 0.3$

Figure 3.7: *Approximation of facial images (CBCL dataset), using NMF and SOMP.*

are quite similar in both cases, even if NMF seems to provide slightly better visual results. This is due to the fact that the vectors in NMF are specifically adapted to the images under consideration, while atoms are selected from a pre-defined, generic dictionary in the case of SOMP. However, recall that NMF cannot lead to an effective coding strategy, as the basis vectors are essentially images, which are as difficult to compress as the initial images. On the contrary, recall that the vectors selected by SOMP can be efficiently described by the parameters of the corresponding atoms in the redundant dictionary. It has been shown that such a signal decomposition leads to effective image compression schemes [45].

Finally, the quantization experiments, shown in Figures 3.7(d) and 3.7(e), hint that SOMP is more robust than NMF to noise that can alter the representation of the images. In particular, SOMP is shown to be more robust to coarse uniform quantization of the vector coefficients. This is mostly due to the non-uniform distribution of the magnitude of its coefficients, where most of the signal energy is concentrated on a few atoms only. These atoms are of large scale and capture the main geometric characteristics of the facial shape. Therefore, reconstructed images still appear like faces even when coefficients of these atoms are coarsely quantized.

3.4.6 Approximation and Classification trade-off

Finally, we demonstrate the approximation and classification trade-off which is driven by the parameter λ in the SAS algorithm. Figure 3.8 illustrates the classification error rate

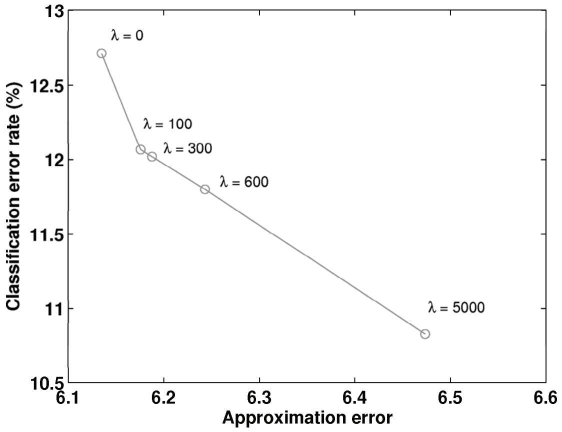


Figure 3.8: *Approximation-classification trade-off with SAS algorithm on the handwritten digit dataset.*

versus the approximation error, for different values of λ, using the handwritten digit data set. The approximation error is measured by the Frobenius norm of the residual matrix i.e., $\|S - \Psi C\|_F$. In this experiment, the dimension of reduced space was fixed to $r = 40$. Observe that when $\lambda = 0$, SAS simplifies to SOMP and as λ increases, more emphasis is given to the classification performance. This improves the classification error rate but at the same time the approximation quality deteriorates due to the fact that Ψ is not selected any more by pure approximation criteria (see eq. (3.10)). This trade-off is very important since it permits to build systems that are efficient not only in the compression of the multimedia data, but also in the desired data mining task.

3.5 Application to face recognition

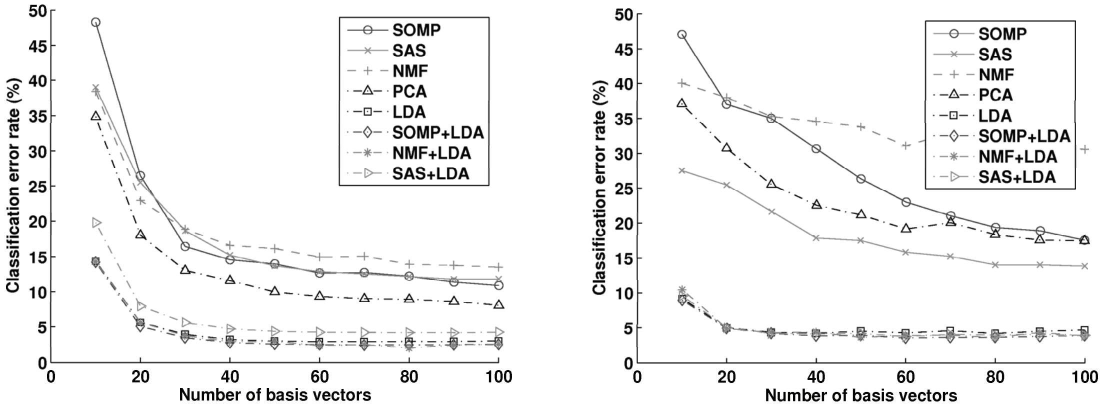
So far, we have considered the behavior of the proposed methods in the generic image classification problem. In this section, we investigate their application to face recognition. We show that the proposed dimensionality reduction methods can be used as preprocessing blocks in systems that are optimized for specific classification applications, namely 2D and 3D face recognition.

3.5.1 2D face recognition

Let us focus first on the particular problem of 2D face recognition to illustrate the potential of the proposed methodology. While our aim is not to provide a novel face recognition system, we provide experimental evidence that suggests that the proposed methods can be combined with subsequent supervised methods and yield effective hybrid methods. These are competitive with the state of the art in face recognition, while they provide simultaneously efficient and flexible coding solutions. In particular, we combine SOMP and SAS with Linear Discriminant Analysis [119, ch.4] (see also Section 2.2), and we denote the hybrid algorithms respectively by SOMP-LDA, and SAS-LDA. We compare them with Eigenfaces (PCA) and Fisherfaces

	Size of S	n_i
XM2VTS faces	1280×2360	8
AR faces	1728×1008	8

Table 3.2: *The data sets used in 2D face recognition, where n_i denotes the number of samples per class.*



(a) XM2VTS (b) AR

Figure 3.9: *Face recognition experiments.*

(LDA) as well as with the corresponding hybrid solution of NMF, denoted as NMF-LDA. We use the following data sets for our evaluation.

- **XM2VTS face database** The XM2VTS database contains 295 individuals and 8 different images for each individual including variation in lighting. The size of each facial image has been downsampled to 32×40. Note that the frontal faces have been extracted with respect to the ground truth eye positions. We form the training set by a random subset of 4 different facial images per subject and use the remaining 4 as a test set.
- **AR face database** The AR face database contains 126 individuals and 8 different images for each individual including variation in facial expression and lighting. The size of each facial image has been downsampled to 36×48. We form the training set by a random subset of 4 different facial images per subject and use the remaining 4 as a test set.

Table 3.2 summarizes the characteristics of the data sets. Figure 3.9(a) illustrates the classification performances for all methods on the XM2VTS database. We report the average error rate across 50 random realization of the training/test set for different number of basis vectors $r = [10 : 10 : 100]$ (in MATLAB notation). First, notice that the proposed methods outperform NMF. The main observation though is that combining SOMP and SAS with LDA yields effective hybrid methods that have similar performance with Fisherfaces (LDA) which is among the state of the art for face recognition.

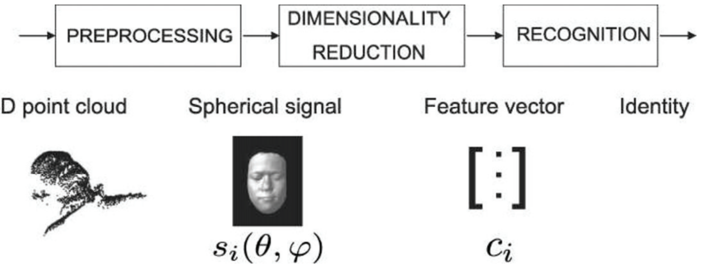


Figure 3.10: *Block diagram of the 3D face recognition system.*

Figure 3.9(b) illustrates the same experiment using the AR database. Notice that in this database SAS outperforms the other methods and SOMP competes with PCA. We observe again that the hybrid methods with SOMP and SAS compete with Fisherfaces resulting in state of the art performance. As the proposed algorithms work directly in the compressed domain, it certainly shows their potential in the design of more efficient multimedia processing systems.

3.5.2 3D face recognition

We focus now on the problem 3D face recognition [9]. We propose a 3D face recognition methodology [71, 72], which relies on sparse representation of faces on the sphere (see Fig. 3.10 for the main architecture of the system). In our scheme, 3D faces are represented as spherical signals, which permits to preserve the geometry information and to provide important advantages with respect to depth images (i.e., 2D arrays of face to camera distance values). A spherical signal $s(\theta, \varphi)$ represents the distance from the center of the sphere as a function of θ (azimuth angle) and φ (elevation angle). The reader is referred to [72] for more details on facial region extraction, registration and construction of the spherical facial signals.

We perform dimensionality reduction on the sphere with the spherical version of the Simultaneous Matching Pursuit (SMP) algorithm that was discussed before in Section 3.2. Facial images are jointly represented using a few localized and orientated features in the sphere, which are able to capture discriminant geometrical characteristics. Recall that the signal space here is the space of spherical signals and that $\langle \cdot, \cdot \rangle$ represents the inner product on the sphere, which is defined as $\langle f, g \rangle = \int_\theta \int_\varphi f(\theta, \varphi) g(\theta, \varphi) \sin\theta d\theta d\varphi$. We call SSMP i.e., Spherical SMP the resulting algorithm, to distinguish it from the standard SMP performed on 2D images.

After dimensionality reduction, the spherical facial images are projected in the reduced subspace by computing the corresponding low dimensional coefficients vectors. Those are computed by successive projections of the residual signals over the atoms, in a way that "mimics" the SSMP process. Recognition is performed in the reduced space, among coefficient vectors, by NN classification. Also, we have observed experimentally that the selected atoms are linearly independent. The matching uses the L1 distance metric, which has been found to outperform other related distances in this particular application.

We evaluate the performance of the proposed algorithm on the University of Notre Dame (UND) Biometric database[3], known also as FRGC v1.0 database. The database consists of 277

---

[3] http://www.nd.edu/∼cvrl/UNDBiometricsDatabase.html

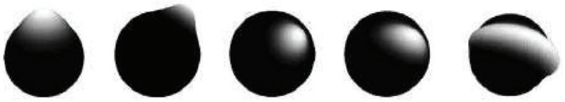


Figure 3.11: *Gaussian atoms*

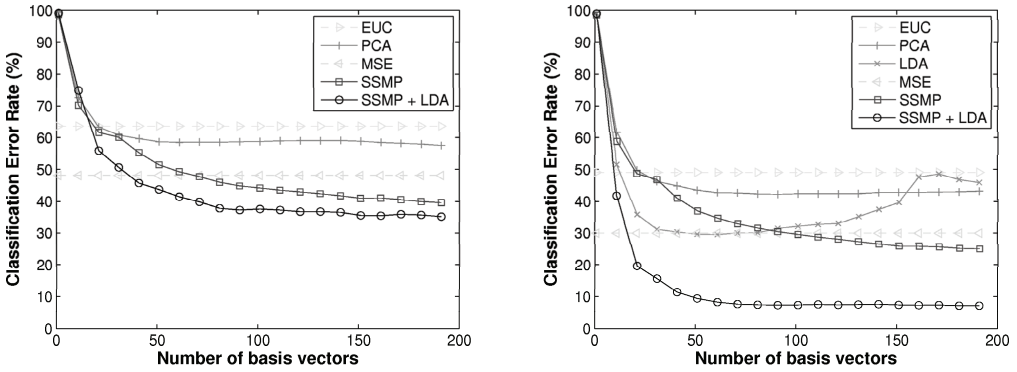
Test configuration	p	Number of subjects	Training set	Test set
T_1	1	200	200	673
T_2	2	166	332	474
T_3	3	121	363	308
T_4	4	86	344	187

Table 3.3: *Test configurations.*

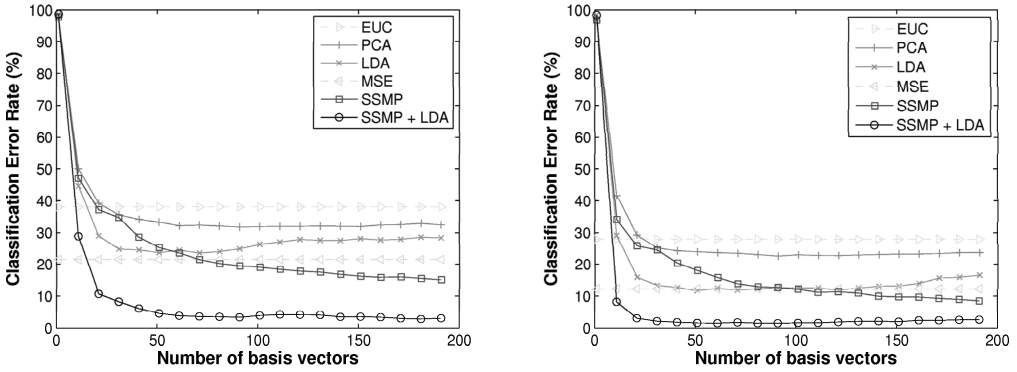
subjects, and each subject has between one and eight scans. We define several configurations for our experiments, driven by the number of samples p per class in the formation of the training set (see Table 3.3 for more details). The dictionary is constructed in a structured way i.e., the atoms are constructed by applying translation, rotation and anisotropic scaling on a generating function, which has been chosen to be the 2D Gaussian function on the sphere. Figure 3.11 shows a few sample Gaussian atoms that are used in the representation of spherical faces. The reader is referred to [71] for more details on the construction of the spherical dictionary.

In order to cope with the limited size of the training set (in some configurations) and with possible registration errors, we enrich the training data by the addition of *virtual faces.* These represent artificial faces that are produced by slight pose variations of the training faces, which are easily computed in the spherical representation. We use virtual samples in the configurations T_1, T_2 and T_3, where the training set is limited. In particular, each spherical training image is expanded by generating 8 slightly shifted versions of it (one pixel in each possible direction).

For the sake of completeness, we include in our comparisons a hybrid method of SSMP followed by LDA, denoted as SSMP+LDA. We compare our methods with PCA and LDA on the (preprocessed) depth images. Additionally, we provide recognition results with the Euclidean distance (EUC) on the depth images as well as the Mean Square Error (MSE) on the spherical functions. We report rank-1 recognition results across random experiments. For each experiment, we split randomly the samples between the training and the test sets. In Figure 3.12 we show the average classification error rate obtained for each configuration, over all 10 random splits. First, we observe that Euclidean distance on the depth images (EUC) is inferior to Mean Square Error (MSE) on the spherical signals. This is the main observation that encourages the use of spherical signals for representing 3D faces. Next, notice that SSMP outperforms PCA, which in turn mainly explains the fact that SSMP+LDA outperforms LDA (on the depth images), since the latter implements a PCA decomposition at first step. Furthermore, we can see that SSMP+LDA gives the best performances among all the tested schemes. Finally, taking into account that the proposed method provides at the same time



(a) Test Configuration T_1 (b) Test Configuration T_2



(c) Test Configuration T_3 (d) Test Configuration T_4

Figure 3.12: *Classification error rates on the FRGC v.1.0 database.*

efficient coding solutions, it shows its potential impact in efficient multimedia mining systems.

3.6 Related work

In Section 2.2, we reviewed the most relevant methods from the literature on low rank approximation and dimensionality reduction. Furthermore, we provided an experimental evaluation of our methodology with respect to these methods in semantic approximation of images (see Section 3.4) and in face recognition (see Section 3.5). In what follows, for the sake of completeness, we review some additional methods from the literature that bridge sparse representations with classification, face recognition or object recognition in general. Although these methods are conceptually related to the methodologies proposed in this chapter, they are quite different from them. We start by methods that study Matching Pursuit for classification.

P. Vincent and Y. Bengio in [114] introduced Kernel Matching Pursuit (KMP). First, the authors show how MP can be used for sparse function approximation, under the machine learning perspective. They propose three variants of MP and they discuss their generalization to other loss functions, different than the squared-error loss one. Furthermore, an alternative formulation of the SVM problem is introduced, which involves a margin loss function, and allows for using MP to build an alternative to SVMs. Finally, relations to other machine learning paradigms, such as SVMs, radial basis functions networks and boosting (with kernels centered on training points) are discussed.

In a recent article, V. Popovici et al in [84] proposed a stochastic version of KMP in order to reduce its computational complexity. The main idea is to avoid the full search in the dictionary by examining only a random subset of it. The experimental results show that comparable classification performances can be obtained with the advantage of being more computationally efficient. Finally, the same authors in [85] propose to adapt the dictionary in the context of KMP in order to encourage sparser solutions. The dictionary functions are assumed to have a parametric form and the parameters are individually optimized in order to provide a better fit to the problem at hand. The authors show empirically that the proposed approach provides competitive or better classification results to the original KMP.

Other methods have used sparse representations for face recognition. We mention the work in [120], which use sparse representations and the theory of compressed sensing for robust face recognition. The main idea is based on computing a sparse representation of the test image over an overcomplete dictionary whose atoms are the training images. The authors also show that the sparse representation can be employed for outlier detection. Furthermore, they study two important issues in face recognition; (i) the role of feature extraction and (ii) robustness to occlusions. One striking observation regarding the proposed framework is that the choice of feature extraction is not critical, as long as the feature space dimension is sufficiently large to guarantee the correct recovery of the sparsest solution. The proposed framework can be also adapted in order to provide robustness to image occlusions, by exploiting the fact that occlusions are typically sparse in the canonical (i.e., pixel) basis.

Sparse representations have been also used for generic object recognition. The authors in [83] propose the use of sparse representations for coarse and fine object recognition. They consider the problem of object recognition under different views and they call the pose estimation problem as coarse recognition and that of object class estimation as fine recognition problem. The main idea is to treat the (training) images of an object under viewpoint variations as a three dimensional intensity function of spatial coordinates and pose. The authors propose to represent this function sparsely over a redundant dictionary of three dimensional atoms, which are built by separable basis functions. Then, they show that the pose estimation reduces to a polynomial evaluation problem, which is further used for recognition of a test object image (i.e., after the correct pose has been estimated first).

Finally we should mention that methods that combine joint approximation and discrimination criteria (similarly to our semantic approximation scheme) for dictionary learning have started recently to emerge. F. Rodriguez and G. Sapiro [88] proposed a methodology for non-parametric dictionary learning, which relies on an alternating optimization scheme, similarly to the K-SVD algorithm [1].

3.7 Conclusions

We have proposed in this chapter a semantic approximation framework where supervised dimensionality reduction achieves an interesting trade-off between compression and classification. First, we have presented a subspace method that uses greedy algorithms from simultaneous sparse approximations of multiple signals. Next, we have extended these algorithms to classification problems using a supervised dimensionality reduction strategy. This strategy utilizes a class separability penalty term in the objective function of the optimization problem, which improves on the classification performance and provides the desired trade-off between the two objectives. The proposed algorithms and their hybrid versions lead to performances that are competitive or even outperform solutions that run state of the art algorithms in 2D and 3D face recognition applications, with the additional advantage of enabling efficient coding solutions. It certainly represents a promising solution for multimedia data mining applications, where relevant feature extraction and signal compression are to be performed jointly. In the following chapter, we will allow the signals to be geometrically transformed and we will study transformation invariance in pattern analysis.

Chapter 4

Transformation Invariance

4.1 Introduction

In this chapter, we allow the patterns to be geometrically transformed and we are interested in comparing transformed versions of patterns for matching and recognition purposes. The comparison of two patterns is generally meaningful if they are aligned, so that their distance reflects their structural and geometric differences. Alignment consists of estimating the relative transformation between patterns. The transformed version of a pattern can be described as a point of a (generally nonlinear) manifold in a high dimensional space, which is usually called the *transformation manifold*. If the distance between two patterns is truly invariant to transformations, it becomes equivalent to the distance between their corresponding transformation manifolds. This is called the two sided manifold distance or the one sided manifold distance (MD) when one of the pattern is fixed. Observe that the MD between a pattern and its transformed version is zero; hence, MD is close to the semantic distance between any two patterns. The minimum manifold distance thus corresponds to the distance between patterns after proper alignment.

In this chapter, we introduce a method[1] for pattern alignment and manifold distance computation, which is able to find the globally optimal solution when the transformation consists of a synthesis of translations, rotation and isotropic scaling [57]. We represent the pattern of interest as a linear combination of geometric atoms, which are chosen from a structured dictionary. When the pattern is transformed, the transformation is essentially applied on each constituent atom individually, resulting in a synthesis of transformations. Using group theory of transformations we build on [56] and show that the proposed representation allows for a closed form expression of the manifold equation with respect to the transformation parameters. Next, we formulate the pattern alignment problem as a DC program, by showing that the objective function is DC, i.e., that it can be written as a *difference of convex functions*. DC programs are non-convex problems that can be globally solved by exploiting their special structure. In order to solve the DC program, we employ an outer approximation - cutting plane method that converges to the globally optimal solution. The algorithm permits to estimate jointly the translation, rotation and scaling parameters, and ensures convergence to the global minimizer. Once the transformation parameters have been estimated, manifold

---

[1] To appear in: E. Kokiopoulou and P. Frossard, “Minimum distance between pattern transformation manifolds: Algorithm and Applications”, IEEE Transactions on Pattern Analysis and Machine Intelligence.

distance can be readily computed by comparing the aligned patterns, and it can be further used for matching and classification purposes. We finally provide experimental results that show the effectiveness of the proposed algorithm in robust pattern recognition and image alignment applications, where it outperforms existing solutions that are typically based on (sub-optimal) manifold, tangent or Euclidean distance computation.

The rest of this chapter is organized as follows. In Section 4.2 we introduce the representation of transformation manifolds using sparse geometric expansions and we formulate the optimization problem. Then, we show in Section 4.3 that the objective function is DC; hence our optimization problem is a DC program. We present experimental results on image alignment and robust face recognition in Sections 4.4 and 4.5 respectively. For the sake of completeness, in Sec. 4.6, we review some additional related work concerning transformation invariance issues in pattern analysis. Finally, our concluding remarks are provided in Sec. 4.7.

4.2 Problem formulation

In this section, we introduce the concept of transformation manifolds. In particular, we show that the pattern representation based on sparse geometric expansions yields a parametric model of the pattern. Such a geometric pattern representation leads to a closed form expression of the transformation manifold equation with respect to the transformation parameters. Then, we use this handy property to formulate the problem of pattern alignment, which refers to the estimation of the transformation parameters.

4.2.1 Transformation manifolds

We propose to represent the pattern of interest as a linear combination of geometric atoms, taken from a structured and possibly redundant dictionary $\mathcal{D}$ spanning the input space. This representation generally allows to capture the most prominent features of the pattern of interest that are generally also meaningful components. Recall from Chapter 3 that the atoms in a structured dictionary are constructed by applying geometric transformations $\gamma \in \Gamma$ to a generating mother function denoted by ϕ, so that a structured dictionary takes the following form,

$$\mathcal{D} = \{\ \phi_\gamma = U(\gamma)\phi,\ \ \gamma \in \Gamma\}. \tag{4.1}$$

In what follows, a transformation with parameters $\gamma_i = \{b_i, a_i, \omega_i\} \in \Gamma$ denotes a synthesis of translations, anisotropic scalings and rotations. It can be observed that applying a transformation on the mother function is equivalent to transforming the coordinate system from $\{x, y\}$ to $\{\tilde{x}, \tilde{y}\}$ before applying $\phi(\cdot)$. When the ith atom in the structured dictionary (4.1) is built as $\phi_{\gamma_i} = U(\gamma_i)\phi(x, y)$, where $\gamma_i = \{b_i, a_i, \omega_i\} \in \Gamma$, it forms the same 2D function as $\phi(\tilde{x}, \tilde{y})$, where

$$\begin{aligned} \begin{bmatrix} \tilde{x} \\ \tilde{y} \end{bmatrix} &= \underbrace{\begin{bmatrix} \frac{1}{a_{ix}} & 0 \\ 0 & \frac{1}{a_{iy}} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} \cos\omega_i & \sin\omega_i \\ -\sin\omega_i & \cos\omega_i \end{bmatrix}}_{R(\omega_i)} \underbrace{\begin{bmatrix} x - b_{ix} \\ y - b_{iy} \end{bmatrix}}_{t} \\ &= AR(\omega_i)t. \end{aligned} \tag{4.2}$$

The approximation of the pattern of interest s with atoms from the dictionary $\mathcal{D}$ can be obtained in different ways. Even if finding the sparsest approximation of s is generally a hard

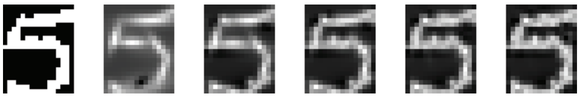


Figure 4.1: *Progressive OMP approximation of the digit "5" (leftmost) with 10, 20, 30, 40 and 50 Gaussian atoms (from left to right).*

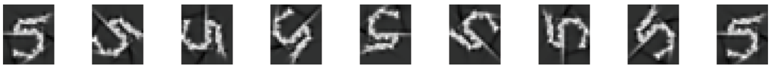


Figure 4.2: *Samples from the transformation manifold of the digit "5". From left to right, the samples correspond to rotation angles from* 0 *to* 2π *with step* $\pi/4$.

problem, effective sub-optimal solutions are usually sufficient to capture the salient semantic and geometric structure of a pattern with only a few atoms. Here, we have chosen to use Orthogonal Matching Pursuit (OMP) [73, Sec. 9.5.3], which is a simplified version of SOMP discussed in Section 3.2, in the case where there is only a single pattern to be approximated. After K steps of OMP, the pattern s is approximated by a sparse linear combination of a few atoms i.e.,

$$s = \sum_{k=1}^{K} \xi_k \phi_{\gamma_k} + r_K, \tag{4.3}$$

where r_K is the residual of the approximation. Figure 4.1 shows the progressive approximation of the digit '5' from a Gaussian dictionary using OMP. Observe that only a few atoms are sufficient to capture the main geometric characteristics of the pattern and the representation (4.3) does not need to be very accurate before it is useful for alignment purposes.

When a pattern s undergoes a geometric transformation η, it spans a high dimensional manifold $\mathcal{T}$ in the ambient space. Assume that the transformation is $\eta = (\beta, \alpha, \theta)$; that is, it consists of a synthesis of translations $\beta = [\beta_x, \beta_y]$, isotropic scaling α and rotation θ. The manifold $\mathcal{T}$ can be expressed mathematically as follows,

$$\mathcal{T} = \{s(\eta) \triangleq U(\eta)s,\ \eta = (\beta, \alpha, \theta)\}. \tag{4.4}$$

Although the manifold is high dimensional, its intrinsic dimension is rather small and equal to the number of transformation parameters, which is 4 in our case. Figure 4.2 shows a few samples from the transformation manifold of the digit "5", when the transformation is simply a rotation.

The transformations η form a group, namely the similitude group SIM(2) on the 2D plane. Let denote by

$$R(\omega) = \begin{bmatrix} \cos\omega & \sin\omega \\ -\sin\omega & \cos\omega \end{bmatrix}, \quad 0 \le \omega < 2\pi,$$

the rotation matrix by angle ω in the 2D plane. If (b, a, ω) and (b', a', ω') are two elements of the SIM(2) group, then the group law is

$$(b, a, \omega) \circ (b', a', \omega') = (b + aR(\omega)b', aa', \omega' - \omega). \tag{4.5}$$

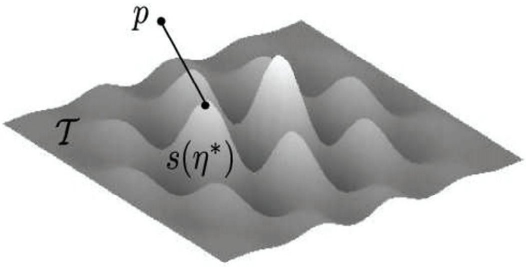


Figure 4.3: *Manifold distance is the minimum distance from a reference point p to the transformation manifold of s.*

Using (4.3) and dropping the residual term r_K, it turns out that applying the transformation η on the pattern s, results in

$$s(\eta) = U(\eta)s = \sum_{k=1}^{K} \xi_k U(\eta)\phi_{\gamma_k} = \sum_{k=1}^{K} \xi_k \phi_{\eta\circ\gamma_k}, \tag{4.6}$$

where $\eta \circ \gamma_k$ is a product of transformations. In words, the transformation is applied on each constituent atom individually. The group law (4.5) indeed applies [45, 27] and can be further employed to determine the updated parameters of the transformed atoms. Equation (4.6) is of great importance, since it expresses the manifold equation (4.4) in closed form with respect to the transformation parameters η. This is very valuable for the estimation of η, and in particular for the applicability of the DC programming methodology that is proposed in the next section.

4.2.2 Estimation of transformation parameters

In this chapter, we are interested in computing the distance between visual patterns under transformation invariance. Suppose that we want to compute the manifold distance from a reference pattern p to the transformation manifold $\mathcal{T}$ described by equation (4.4). Fig. 4.3 provides a graphical illustration of the problem. Essentially, the manifold distance corresponds to the distance between the aligned patterns. Hence, we need to estimate the optimal transformation parameters η^* first. We formulate the parameter estimation problem as follows

$$\eta^* = \arg \max_{\eta=(\beta,\alpha,\theta)} f(\eta), \quad \text{where} \quad f(\eta) = |\langle s(\eta), p\rangle|. \tag{4.7}$$

Recall that $s(\eta) \in \mathcal{T}$ denotes the transformed pattern s when it is subject to transformation $\eta = (\beta, \alpha, \theta)$. We assume that the pattern of interest s has been well approximated with a sparse expansion over $\mathcal{D}$ i.e.,

$$s = \sum_{k=1}^{K} \xi_k \phi_{\gamma_k}, \tag{4.8}$$

which results from (4.3) by dropping the residual term. When the atom parameters γ_k are fixed, the transformed pattern $s(\eta)$ in (4.6) provides a parametric pattern model with respect to η. Note that in the above optimization problem, only the pattern s is expanded in the redundant basis; i.e., the reference pattern p is used as is.

The purpose of the optimization problem is to compute the exact transformation parameters η^*. This is generally a non-convex nonlinear optimization problem [78] which is hard to solve using the traditional methods, such as steepest descent or Newton-type methods due to their local convergence property and the presence of an unknown number of local minima. However, it can be shown that the above objective function is a DC function, or equivalently that it can be expressed as the difference of two convex functions.

Theorem 4.2.1. *The objective function*

$$f(\eta) = |\langle s(\eta), p\rangle| = |\sum_{k=1}^{K} \xi_k \langle \phi_{\eta_k}, p\rangle|, \tag{4.9}$$

where $\eta_k = \eta \circ \gamma_k$, *is DC.*

We show in the next section that this proposition is true. We first recall basic properties of DC functions. Then we show that the geometric transformation of an atom by translation, scaling and rotation is equivalent to a change in the coordinate system. The transformed coordinate system $(\tilde{x}, \tilde{y})$ of the kth atom ϕ_{η_k} depends on the transformation parameters η. With this transformation in the coordinate system, we can demonstrate that $\tilde{x}(\eta_k)^2 + \tilde{y}(\eta_k)^2$ is a DC function of η, which in turn will allow us to express the pixels of each atom ϕ_{η_k} in a DC form. Then, we prove that the inner product $\langle \phi_{\eta_k}, p\rangle$ between the pattern p and each of the atoms is also DC. Finally, we obtain the DC formulation of the objective function $|\langle s(\eta), p\rangle|$.

4.3 DC decomposition

4.3.1 Properties of DC functions

We show in this section that the objective function f in (4.7) is DC, in several steps. We start with some definitions and background material about DC functions [41, 40, 39] and their properties.

First, let X be a convex subset of $\mathbb{R}^n$. A function $f : X \subseteq \mathbb{R}^n \to \mathbb{R}$ is called DC on X, if there exist two convex functions $g, h : X \to \mathbb{R}$ such that f is expressed as

$$f(x) = g(x) - h(x). \tag{4.10}$$

A representation of the above form is called the DC decomposition of f. The DC decomposition is clearly not unique, since if $c(x)$ is a convex function, then $f(x) = (g(x) + c(x)) - (h(x) + c(x))$ is another DC decomposition of f. In what follows, the notations $\succ 0$ and $\succeq 0$ denote positive definiteness and positive semi-definiteness accordingly. We present now a few properties of DC functions.

Proposition 2. *[40, Sec 4.2] Properties of DC functions. Let* $f = g - h$ *and* $f_i = g_i - h_i$, $i = 1 \ldots, m$ *be DC functions. Then the following functions are also DC:*

(a) $\sum_{i=1}^{m} \lambda_i f_i = \left[\sum_{\{i:\lambda_i \geq 0\}} \lambda_i g_i - \sum_{\{i:\lambda_i < 0\}} \lambda_i h_i\right] - \left[\sum_{\{i:\lambda_i \geq 0\}} \lambda_i h_i - \sum_{\{i:\lambda_i < 0\}} \lambda_i g_i\right].$

(b) $|f| = 2\max\{g, h\} - (g + h)$.

(c) *If f_1 and f_2 are DC functions, then the product $f_1 \cdot f_2$ is DC. Moreover, if f_1 and f_2 have nonnegative convex parts, the following DC decomposition holds*

$$f_1 \cdot f_2 = \frac{1}{2}[(g_1 + g_2)^2 + (h_1 + h_2)^2] - \frac{1}{2}[(g_1 + h_2)^2 + (g_2 + h_1)^2].$$

In addition, it can be shown that the synthesis of a convex function and a DC function is also DC, which is particularly relevant for our particular objective function, as shown below.

Proposition 3. *Let $f(x) : \mathbb{R}^n \to \mathbb{R}$ be DC and $q : \mathbb{R} \to \mathbb{R}$ be convex. Then,*

(a) *the composition $q(f(x))$ is DC [40, Sec 4.2].*

(b) *$q(f(x))$ has the following DC decomposition,*

$$q(f(x)) = p(x) - K[g(x) + h(x)], \tag{4.11}$$

where $p(x) = q(f(x)) + K[g(x) + h(x)]$ is a convex function and K is a constant satisfying $K \geq |q'(f(x))|$ [21].

Note that in part (b) of the above proposition, we use slightly different conditions than those in [21]. For this reason, we provide a proof of part (b) in the appendix.

4.3.2 DC decomposition of transformed atoms

We show here that the transformed atom ϕ_{η_k} can be expressed in a DC form. For notational ease, we drop the subscript k since it is clear from the context that we refer to the kth atom. Note first that the transformation of an atom by scaling, rotation and translation, is equivalent to a change in the coordinate system.

Lemma 4.3.1. *The transformed coordinates of an atom in (4.9) have the following form*

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = \begin{pmatrix} \mu_1 \frac{\cos\theta}{\alpha} + \mu_2 \frac{\sin\theta}{\alpha} + \mu_3 \frac{\tau_x}{\alpha} + \mu_4 \frac{\tau_y}{\alpha} + \mu_5 \\ \nu_1 \frac{\cos\theta}{\alpha} + \nu_2 \frac{\sin\theta}{\alpha} + \nu_3 \frac{\tau_x}{\alpha} + \nu_4 \frac{\tau_y}{\alpha} + \nu_5 \end{pmatrix}, \tag{4.12}$$

where the μ_i's and ν_i's are constants depending on the atom parameters which are fixed, and τ_x and τ_y are related to β_x and β_y by the following relation

$$\begin{pmatrix} \tau_x \\ \tau_y \end{pmatrix} = -R(-\theta) \begin{pmatrix} \beta_x \\ \beta_y \end{pmatrix} \tag{4.13}$$

Proof. The proof is given in the appendix. □

With the change of variables in (4.13), the optimization variables become τ_x, τ_y, α and θ. This representation of the transformation parameters is equivalent to the previous one, since from the τ's we can recover the β's and vice versa (using (4.13)). Hence, for notational ease, we will continue using η for denoting the transformation parameters with respect to τ_x, τ_y, α and θ.

The above lemma implies that,

$$\begin{aligned}\tilde{x}(\eta)^2 &= \mu_1^2\frac{\cos^2\theta}{\alpha^2} + \mu_2^2\frac{\sin^2\theta}{\alpha^2} + \mu_3^2\frac{\tau_x^2}{\alpha^2} + \mu_4^2\frac{\tau_y^2}{\alpha^2} + \mu_5^2 + 2\mu_1\mu_2\frac{\cos\theta\sin\theta}{\alpha^2} + \\ &\quad +2\mu_1\mu_3\frac{\cos\theta\tau_x}{\alpha^2} + 2\mu_1\mu_4\frac{\cos\theta\tau_y}{\alpha^2} + 2\mu_1\mu_5\frac{\cos\theta}{\alpha} + 2\mu_2\mu_3\frac{\sin\theta t_x}{\alpha^2} \\ &\quad +2\mu_2\mu_4\frac{\sin\theta\tau_y}{\alpha^2} + 2\mu_2\mu_5\frac{\sin\theta}{\alpha} + 2\mu_3\mu_4\frac{\tau_x\tau_y}{\alpha^2} + 2\mu_3\mu_5\frac{\tau_x}{\alpha} + 2\mu_4\mu_5\frac{\tau_y}{\alpha} \qquad (4.14)\end{aligned}$$

and similarly for $\tilde{y}(\eta)$ by replacing the μ's by ν's. In order to show that ϕ_η is DC we need to show that every single term in the above equation is DC as well. In other words, we have the following proposition.

Proposition 4. *The expression corresponding to the transformed coordinate system $\tilde{x}(\eta)^2$ (and similarly $\tilde{y}(\eta)^2$) in (4.14) is DC.*

In what follows, we provide a few lemmas that will help us showing that this is true. In particular, we show that several of the constituent functions are DC: $f(\theta) = \cos\theta$, $f(\theta) = \sin\theta$, $f(\theta,\alpha) = \frac{\theta}{\alpha}$, $f(\theta,\alpha) = \frac{\cos\theta}{\alpha}$ and $f(\theta,\alpha) = \frac{\sin\theta}{\alpha}$.

Lemma 4.3.2. *The following functions are DC*

(i) $f_1(\theta) = \cos\theta,\ \theta \in [0, 2\pi)$.

(ii) $f_2(\theta) = \sin\theta,\ \theta \in [0, 2\pi)$.

Proof. **(i)** From Taylor's theorem it is known that

$$\cos\theta = \sum_{n=0}^{\infty}\frac{(-1)^n\theta^{2n}}{(2n)!} = 1 - \frac{\theta^2}{2!} + \frac{\theta^4}{4!} - \frac{\theta^6}{6!} + \frac{\theta^8}{8!} - \dots$$

By grouping the terms of the same sign we have that

$$\cos\theta = \left[1 + \frac{\theta^4}{4!} + \frac{\theta^8}{8!} + \frac{\theta^{12}}{12!} + \dots\right] - \left[\frac{\theta^2}{2!} + \frac{\theta^6}{6!} + \frac{\theta^{10}}{10!} + \dots\right]. \qquad (4.15)$$

Since $\theta \in [0, 2\pi)$, all the powers of the form of θ^n, $n \geq 0$ are convex [10, Ch. 3]. Thus, the above equation provides a DC decomposition of the cosine function when $\theta \in [0, 2\pi)$.

(ii) Similarly, the sine function is written in the following form,

$$\begin{aligned}\sin\theta &= \sum_{n=0}^{\infty}\frac{(-1)^n\theta^{2n+1}}{(2n+1)!} \\ &= \left[\theta + \frac{\theta^5}{5!} + \frac{\theta^9}{9!} + \frac{\theta^{13}}{13!} + \dots\right] - \left[\frac{\theta^3}{3!} + \frac{\theta^7}{7!} + \frac{\theta^{11}}{11!} + \dots\right], \qquad (4.16)\end{aligned}$$

which is DC using similar arguments as above.

□

Lemma 4.3.3. *The function $f(\theta,\alpha) = \frac{\theta}{\alpha} : \mathbb{R} \times \mathbb{R}^+ \rightarrow \mathbb{R}$ is DC with the following DC decomposition*

$$f(\theta,\alpha) = \frac{\theta}{\alpha} = g(\theta,\alpha) - h(\theta,\alpha) = \frac{1}{2}\frac{(\theta+1)^2}{\alpha} - \frac{1}{2}\frac{(\theta^2+1)}{\alpha}. \qquad (4.17)$$

Proof. We need to show that both parts are convex. The first part $g(\theta,\alpha) = \frac{(\theta+1)^2}{\alpha}$ is convex since

$$\begin{aligned}\nabla^2 g(\theta,\alpha) &= \frac{2}{\alpha^3}\begin{bmatrix} \alpha^2 & -(\theta+1)\alpha \\ -(\theta+1)\alpha & (\theta+1)^2 \end{bmatrix} \\ &= \frac{2}{\alpha^3}\begin{bmatrix} \alpha \\ -(\theta+1) \end{bmatrix}\begin{bmatrix} \alpha \\ -(\theta+1) \end{bmatrix}^\top \succeq 0,\end{aligned}$$

when $\alpha > 0$. For the second part $h(\theta,\alpha) = \frac{(\theta^2+1)}{\alpha}$ we have that

$$\nabla^2 h(\theta,\alpha) = \frac{2}{\alpha^3}\begin{bmatrix} \alpha^2 & -\theta\alpha \\ -\theta\alpha & \theta^2+1 \end{bmatrix} \succeq 0,$$

since $\alpha > 0$ and $v^\top \nabla^2 h(\theta,\alpha) v \geq 0$, for every $v = [v_1\ v_2]^\top \neq 0$. To see why this is true observe that

$$\begin{aligned}&[v_1\ v_2]\begin{bmatrix} \alpha^2 & -\theta\alpha \\ -\theta\alpha & \theta^2+1 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \\ &= \alpha^2 v_1^2 - \theta\alpha v_1 v_2 - \theta\alpha v_1 v_2 + \theta^2 v_2^2 + v_2^2 \\ &= (\theta v_2 - \alpha v_1)^2 + v_2^2 \geq 0.\end{aligned}$$

□

Lemma 4.3.4. *The function* $f(\theta,\alpha) = \frac{\theta^k}{\alpha} : [0,2\pi) \times \mathbb{R}^+ \to \mathbb{R},\ k \geq 2,$ *is convex.*

Proof. The Hessian matrix of f is

$$\nabla^2 f(\theta,\alpha) = \frac{\theta^{k-2}}{\alpha^3}\begin{bmatrix} k(k-1)\alpha^2 & -k\theta\alpha \\ -k\theta\alpha & 2\theta^2 \end{bmatrix}.$$

Observe that the term $\frac{\theta^{k-2}}{\alpha^3}$ is positive, so we only need to prove that the remaining matrix is positive semi-definite. Call λ_1 and λ_2 its eigenvalues. Then observe that its determinant is

$$\lambda_1\lambda_2 = \theta^2\alpha^2[2k(k-1) - k^2] = \theta^2\alpha^2 k(k-2) \geq 0,$$

since $k \geq 2$. Thus λ_1 and λ_2 have always the same sign. Now, observe also that the trace of this matrix is

$$\lambda_1 + \lambda_2 = k(k-1)\alpha^2 + 2\theta^2 > 0$$

Thus, we conclude that both eigenvalues are non-negative. Therefore, the Hessian matrix is positive semi-definite and f is convex. □

Lemma 4.3.5. *The following functions are DC*

(i) $f_1(\theta,\alpha) = \frac{\cos\theta}{\alpha} : [0,2\pi) \times \mathbb{R}^+ \to \mathbb{R}$

(ii) $f_2(\theta,\alpha) = \frac{\sin\theta}{\alpha} : [0,2\pi) \times \mathbb{R}^+ \to \mathbb{R}$

Proof. **(i)** Using equation (4.15) we have that

$$\frac{\cos\theta}{\alpha} \quad = \quad \Big[\frac{1}{\alpha}+\frac{\theta^4}{\alpha 4!}+\frac{\theta^8}{\alpha 8!}+\frac{\theta^{12}}{\alpha 12!}+\ldots\Big]-\Big[\frac{\theta^2}{\alpha 2!}+\frac{\theta^6}{\alpha 6!}+\frac{\theta^{10}}{\alpha 10!}+\ldots\Big].$$

Each of the individual parts is convex because it consists of a sum of convex functions according to Lemma 4.3.4.

(ii) Similarly, using equation (4.16) we have that,

$$\frac{\sin\theta}{\alpha} \quad = \quad \Big[\frac{\theta}{\alpha}+\frac{\theta^5}{\alpha 5!}+\frac{\theta^9}{\alpha 9!}+\frac{\theta^{13}}{\alpha 13!}+\ldots\Big]-\Big[\frac{\theta^3}{\alpha 3!}+\frac{\theta^7}{\alpha 7!}+\frac{\theta^{11}}{\alpha 11!}+\ldots\Big].$$

The second part of the sine is convex, again due to Lemma 4.3.4. Unfortunately, this is not the case for the first part, since the function $\frac{\theta}{\alpha}$ is not convex. However, we know from Lemma 4.3.3 that it is DC. If we substitute the DC decomposition of $\frac{\theta}{\alpha}$ into the above formula we get,

$$\frac{\sin\theta}{\alpha}=\frac{1}{\alpha}\Big[\frac{(\theta+1)^2}{2}+\frac{\theta^5}{5!}+\frac{\theta^9}{9!}+\frac{\theta^{13}}{13!}+\ldots\Big]-\frac{1}{\alpha}\Big[\frac{(\theta^2+1)}{2}+\frac{\theta^3}{3!}+\frac{\theta^7}{7!}+\frac{\theta^{11}}{11!}+\ldots\Big] \tag{4.18}$$

The above formula yields now a DC decomposition of $\frac{\sin\theta}{a}$ since both parts are convex.

Note in passing that in the DC decompositions of both $\frac{\cos\theta}{a}$ and $\frac{\sin\theta}{a}$ the convex parts are non-negative. This property will be used later on. □

Lemma 4.3.6. *The following functions*

$$\{\frac{\cos\theta\sin\theta}{\alpha^2},\ \frac{\cos\theta\tau_x}{\alpha^2},\ \frac{\sin\theta\tau_x}{\alpha^2},\ \frac{\cos\theta\tau_y}{\alpha^2},\ \frac{\sin\theta\tau_y}{\alpha^2},\ \frac{\cos^2\theta}{\alpha^2},\ \frac{\sin^2\theta}{\alpha^2},\ \frac{\tau_x^2}{\alpha^2},\ \frac{\tau_y^2}{\alpha^2},\ \frac{\tau_x\tau_y}{\alpha^2}\}$$

defined on $\theta\in[0,2\pi),\ \alpha\in\mathbb{R}^+,\ \tau_x\in\mathbb{R},\ \tau_x\in\mathbb{R}$, *are DC.*

Proof. According to Lemma 4.3.5, the functions $\frac{\cos\theta}{\alpha}$ and $\frac{\sin\theta}{\alpha}$ are DC with nonnegative convex parts. Also, the functions $\frac{\tau_x}{\alpha}$ and $\frac{\tau_y}{\alpha}$ are DC with nonnegative convex parts, due to Lemma 4.3.3. The fact that all the above functions are DC, results from property (c) of Proposition 2 which states that the product of two DC functions (with nonnegative convex parts) is also DC. □

Coming back to the transformed coordinates axes of the kth atom ϕ_{η_k} in (4.9), we can now show that the transformed coordinates system is represented by DC functions.

Proof of Proposition 4: According to Lemma 4.3.6, $\tilde{x}(\eta)^2$ (and similarly $\tilde{y}(\eta)^2$) in (4.14) is DC, since it is a linear combination of DC functions (property (a) of Proposition 2). Thus, $z(\eta)=\tilde{x}(\eta)^2+\tilde{y}(\eta)^2$ is also DC and suppose that $z(\eta)=g_z(\eta)-h_z(\eta)$ is its DC decomposition. □

4.3.3 DC form of the objective function

Now we are ready to prove the main result, namely Theorem 4.2.1, which claims that the objective function of the optimization problem (4.7) is DC. The construction of a geometric atom by transformation of the generating function is equivalent to apply the generating functions on the transformed coordinates computed above. Given the above developments, it finally remains to show that the transformed generating functions are DC, and that the inner products between the atoms and the pattern of interest p are also DC functions. We prove these properties for the two different generating functions ϕ used in this chapter.

Proof of Theorem 4.2.1:

- If ϕ is Gaussian then

$$\begin{aligned} \phi_\eta &\triangleq \phi(\tilde{x}(\eta), \tilde{y}(\eta)) = \frac{e^{-(\tilde{x}(\eta)^2+\tilde{y}(\eta)^2)}}{\alpha\|\phi\|} = \frac{e^{-z(\eta)}}{\alpha\|\phi\|} \\ &= e^{-z(\eta)-\ln\alpha-\ln\|\phi\|} = e^{-[z(\eta)+\ln\alpha+\ln\|\phi\|]} \\ &= e^{-w(\eta)}, \end{aligned}$$

where we have introduced the function

$$\begin{aligned} w(\eta) &= z(\eta) + \ln\alpha + \ln\|\phi\| \\ &= \big[g_z(\eta) + \ln\|\phi\|\big] - \big[h_z(\eta) - \ln\alpha\big] \\ &= g_w(\eta) - h_w(\eta), \end{aligned} \tag{4.19}$$

which is also DC, since $\ln\alpha$ is a concave function of α and therefore $-\ln\alpha$ is convex. In the above formulas, $\|\phi\|$ denotes the norm of the non-transformed atom and $\alpha\|\phi\|$ is the norm of the transformed atom. Putting these pieces together, we conclude that ϕ_η is DC with the following decomposition $e^{-w(\eta)} = [e^{-w(\eta)} + K(g_w(\eta)+h_w(\eta))] - [K(g_w(\eta)+h_w(\eta))]$, according to Proposition 3(b).

- If ϕ is AR then

$$\begin{aligned} \phi_\eta &\triangleq \phi(\tilde{x}(\eta), \tilde{y}(\eta)) = \frac{e^{-(\tilde{x}(\eta)^2+\tilde{y}(\eta)^2)}}{\alpha\|\phi\|} \cdot [4\tilde{x}(\eta)^2 - 2] \\ &= 4\tilde{x}(\eta)^2 \frac{e^{-z(\eta)}}{\alpha\|\phi\|} - 2\frac{e^{-z(\eta)}}{\alpha\|\phi\|}, \end{aligned}$$

which is also DC, since it is linear combination of two DC terms (the first term is DC since it is the product of two DC functions (property (c) of Proposition 2)).

Next, we need to show that the inner product $\zeta(\eta) \triangleq \langle\phi_\eta, p\rangle$ is also DC. Assume that M is the number of pixels of the images and $\phi_m = g_m - h_m$ is the DC decomposition of the mth pixel of ϕ_η. Then, $\zeta(\eta)$ is DC with the following decomposition

$$\begin{aligned} \zeta(\eta) &= \langle\phi_\eta, p\rangle = \sum_{m=1}^{M} \phi_m p_m \\ &= \Big[\sum_{\{m:p_m\geq 0\}} p_m g_m - \sum_{\{m:p_m<0\}} p_m h_m\Big] - \Big[\sum_{\{m:p_m\geq 0\}} p_m h_m - \sum_{\{m:p_m<0\}} p_m g_m\Big], \end{aligned}$$

based on property (a) of Proposition 2.

Assume now that $\langle\phi_{\eta_k}, p\rangle = g_k(\eta) - h_k(\eta)$ is the DC decomposition of the inner product of the kth atom with p. In the sequel, we use again the property (a) of Proposition 2 to come up with the DC decomposition of $\sum_{k=1}^{K} \xi_k \langle \phi_{\eta_k}, p \rangle$ which reads

$$\begin{aligned}
\sum_{k=1}^{K} \xi_k \langle \phi_{\eta_k}, p \rangle &= \underbrace{\Big[\sum_{\{k:\xi_k \geq 0\}} \xi_k g_k - \sum_{\{k:\xi_k < 0\}} \xi_k h_k \Big]}_{g} - \underbrace{\Big[\sum_{\{k:\xi_k \geq 0\}} \xi_k h_k - \sum_{\{k:\xi_k < 0\}} \xi_k g_k \Big]}_{h} \\
&= g(\eta) - h(\eta).
\end{aligned}$$

Finally, the objective function in (4.9) is DC with the following decomposition, following from the property (b) of Proposition 2,

$$f(\eta) = |\sum_{k=1}^{K} \xi_k \langle \phi_{\eta_k}, p \rangle| = 2 \max\{g, h\} - (g + h).$$

□

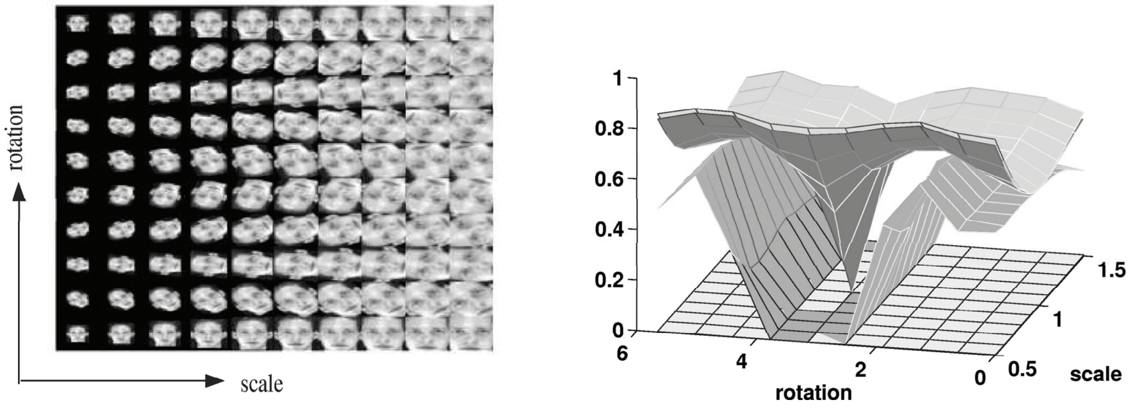
Due to Theorem 4.2.1, the objective function is a DC function and the optimization problem can therefore be formulated as a DC program. This permits the application of DC programming methods for finding the global minimizer to the manifold distance computation. To the best of our knowledge, this is the first globally optimal framework that is proposed in the context of transformation invariant distance computation. We use an outer-approximation cutting plane method [41, Thm 5.3] for solving the DC program, due to its simplicity and the fact that the parameter space in our case is four-dimensional (see Section A.3 in the Appendix for more details on the cutting plane algorithm). However, we should mention that our framework could be also combined with other DC solvers such as Branch-and-Bound schemes [41, Sec 5.1, Sec 5.2] and DCA [2].

Note in passing that the DC decomposition of f is both inefficient and complicated to be obtained in closed form with respect to η. On the contrary, the values of g and h can be obtained in practice by applying sequentially the above properties. Actually this can be done at the cost of evaluating only one of them (that is, the evaluation of g yields the corresponding value of h for free and vice versa).

Computational cost analysis Let us provide now an analysis of the computational complexity of the proposed methodology. Notice that typically each step of a DC solver calls for function and subgradient evaluations. Since each subgradient evaluation calls for two function evaluations, it is sufficient to study the cost of function evaluations alone. Before delving into the cost analysis, it is important first to differentiate between the complexity of evaluating the DC decomposition of f and the complexity of the DC solver. The latter depends on the particular characteristics of the DC solver that is employed, and it is beyond the scope of the present work.

We turn our attention now to the cost of evaluating the convex parts g and h of the objective function f in Eq. (4.9). Recall that the evaluation of g yields the corresponding value of h for free and vice versa. This is due to the fact that the values of $g(\eta_0)$ and $h(\eta_0)$ at a certain point η_0 of the parameter space, are obtained simultaneously by applying sequentially the properties described in Section 4.3. Overall, the computational complexity of each function evaluation is

$$C_{\text{comp}} = O(K \cdot n_1 \cdot n_2),$$



(a) Transformed faces

(b) Distance surfaces: (from top to bottom) ED, TD, MD and MDDC

Figure 4.4: *Invariance of various methods with respect to scale and rotation.*

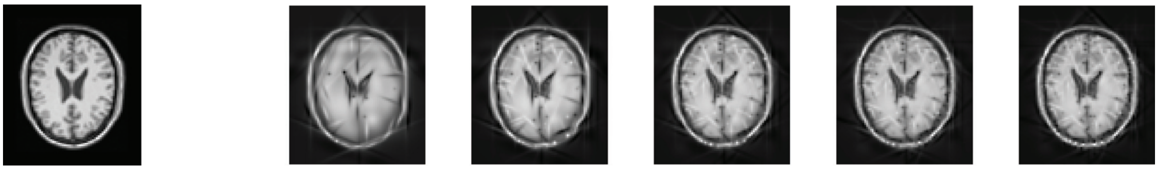
where K is the number of atoms and $M = n_1 \cdot n_2$ is the number of pixels of the image. This arises from the fact that a pattern is represented by K atoms and every pixel of each atom is expressed in DC form. Note that K is typically small (e.g., less than 100) and it is independent of the image size. It solely depends on the geometric complexity (shape) of the pattern and the properties of the dictionary. Therefore, we observe that the sparsity of the pattern and the image resolution are the main factors affecting the cost of function evaluations.

4.4 Image alignment experiments

In this section we compare the proposed method, called MDDC (i.e., manifold distance measure using DC programming) with related methods from the literature. We have observed in practice that our method reaches quickly the vicinity of the global minimizer, but then the convergence rate drops before it computes the global minimizer with high accuracy. This is a common characteristic of cutting plane methods in general. On the other hand, Newton is a very accurate and fast method but it requires an initial guess that is close to the global solution. For this reason, we combine both methods to get the best of both worlds. Thus, what we call MDDC is essentially a hybrid method which first employs the cutting plane method to reach the vicinity of the global minimizer and then switches to Newton with the initial parameters that are obtained from the cutting plane algorithm.

4.4.1 Comparison of different methods

In the first experiment, we compare MDDC with ED (Euclidean distance), TD (tangent distance) and MD (manifold distance using Newton's method). For our comparison we use a facial image from the ORL database [91] (see Section 4.5 for more details on this database). Fig. 4.4(a) shows how this face transforms in the scale-rotation transformation space. Since the manifold (semantic) distance between all these facial images is zero, we would like our



(a) MRI image

(b) OMP approximations using 100, 200, 300, 400, 500 atoms respectively (left to right).

Figure 4.5: *The MRI brain image and its successive approximations.*

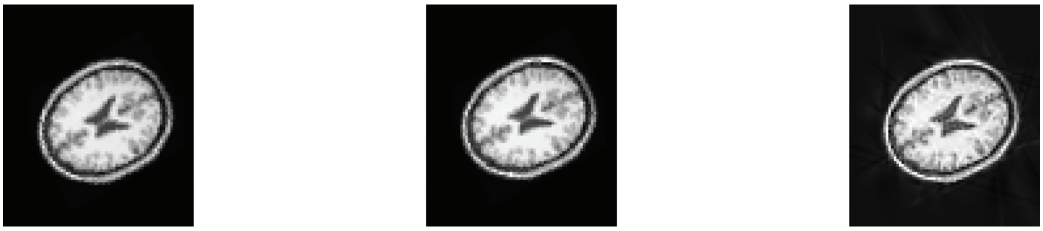
methods to give a distance between them which is as small as possible. Hence, the smaller the obtained distance, the better the method.

We build a parametric model of the facial image with $K = 50$ Gaussian atoms (see (4.8)) using OMP. Then, we sample the transformation parameter space by sampling the rotation angle θ uniformly in $[0, 2\pi)$ with step $\pi/5$ and the scale α in $[0.5, 1.5]$ with step 0.1. For each combination of scale and rotation values we build a transformed image $s(\eta)$ using (4.6) and employ it as the reference image p. Then, we compute the distance between s and p using all methods and compare the results. In the TD method, the tangent vectors are computed by finite differences using (4.6) and the manifold is linearized around $\theta = 0$ and $\alpha = 1$ (which corresponds to the non-transformed pattern). In the case of MD and MDDC we report the distance $\|s(\hat{\eta}) - s(\eta^*)\|_2$, where η^* and $\hat{\eta}$ is the exact and estimated transformation respectively. Note that in MDDC we use 100 iterations of the cutting plane method before switching to Newton.

Figure 4.4(b) illustrates the obtained distance surfaces from all methods. First observe that the ED surface is non convex, which reveals the difficulty of the problem in practical scenarios. Notice that ED and TD have comparable performance, with the latter bringing a minor improvement over the former. Next, observe that Newton's method provides a dramatic improvement over the previous methods. We see also that its local convergence property is also verified in practice i.e., for small transformations it is able to converge to the global minimizer. However, for transformations of larger magnitude it gets trapped in local minima. Finally, MDDC seems to provide superior results compared to the other methods. It provides almost zero distance over all pairs of scale and rotation, verifying its ability to converge to the globally optimal solution. Thus, it corresponds indeed to a close approximation of the semantic distance between the transformed faces.

4.4.2 Application to medical image registration

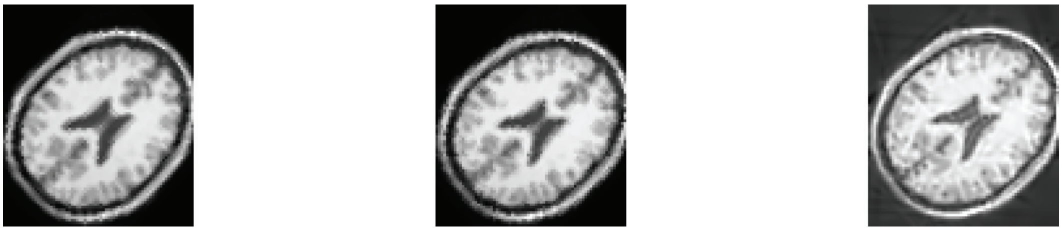
Next, we provide an illustrative example of the application of our method to medical image registration. In this example, we consider η to be a synthesis of translations, scalings and rotations. We consider a typical brain MRI image s as shown in Fig. 4.5(a). We build a pattern model of s using OMP with $K = 500$ Gaussian atoms. Fig. 4.5(b) shows the obtained successive approximations while increasing the number of atoms from 100 to 500 with step 100. Observe that a few atoms are sufficient to capture the main geometric structure of the pattern. Hence, in the context of alignment, one does not need to compute a very accurate and expensive approximation of s. In other words, it is possible to estimate the transformation parameters using only a crude pattern approximation, provided that it captures its salient



(a) Transformed image with η_1

(b) Estimated transformed image

(c) $s(\hat{\eta}_1)$



(d) Transformed image with η_2

(e) Estimated transformed image

(f) $s(\hat{\eta}_2)$

Figure 4.6: *The transformed MRI brain images and their estimated transformed images. First row corresponds to* η_1 *and second row to* η_2*.*

geometric characteristics. For this reason, we use only $K = 100$ atoms in the following alignment experiments.

We perform two experiments with two different transformations. First, we apply a geometric transformation $\eta_1 = (\beta_1, \alpha_1, \theta_1)$ on s, where $\beta_1 = [3,\ 2]^\top$, $\alpha_1 = 0.8$ and $\theta_1 = \pi/3 \simeq 1.047$. The transformed image after image warping with η_1 is shown in Fig 4.6(a). We apply the cutting plane method and after 500 iterations, the estimated transformation parameters are: $\hat{\beta}_1 = [3.15,\ 0.61]^\top$, $\hat{\alpha}_1 = 0.8$ and $\hat{\theta}_1 = 1.08$. Fig. 4.6(b) shows the warped image using the estimated transformation parameters $\hat{\eta}_1$. Also, in Fig. 4.6(c) we show the transformed pattern $s(\hat{\eta}_1)$ (see eq. (4.6)) using 500 atoms (for illustration purposes).

In the second experiment, we test with a new transformation $\eta_2 = (\beta_2, \alpha_2, \theta_2)$, with $\beta_2 = [1,\ 4]^\top$, $\alpha_2 = 1.2$ and $\theta_2 = 5\pi/4 \simeq 3.93$. The transformed image after image warping with η_2 is shown in Fig 4.6(d). After 500 iterations of the cutting plane algorithm the estimated transformation parameters are: $\hat{\beta}_2 = [1.54,\ 4.33]^\top$, $\hat{\alpha}_2 = 1.19$ and $\hat{\theta}_2 = 4$. In Fig. 4.6(e) and 4.6(f) we show the warped image with $\hat{\eta}_2$ and $s(\hat{\eta}_2)$ respectively. Observe that in both cases the cutting plane algorithm is able to converge very close to the global minimizer. Therefore, if it is combined with more accurate local convergence methods it can potentially yield a very accurate method for image registration.

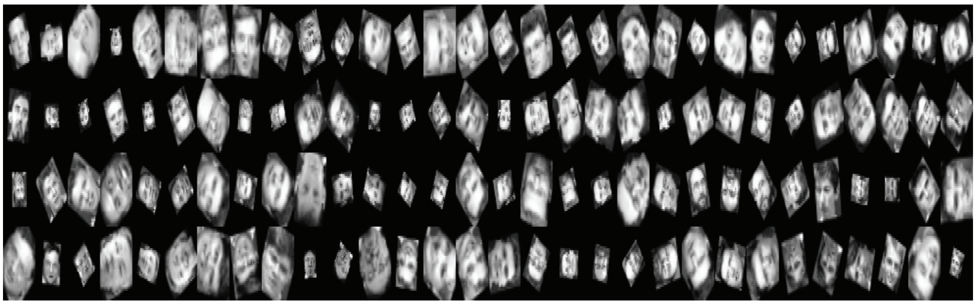


Figure 4.7: *The transformed test face images used for face recognition.*

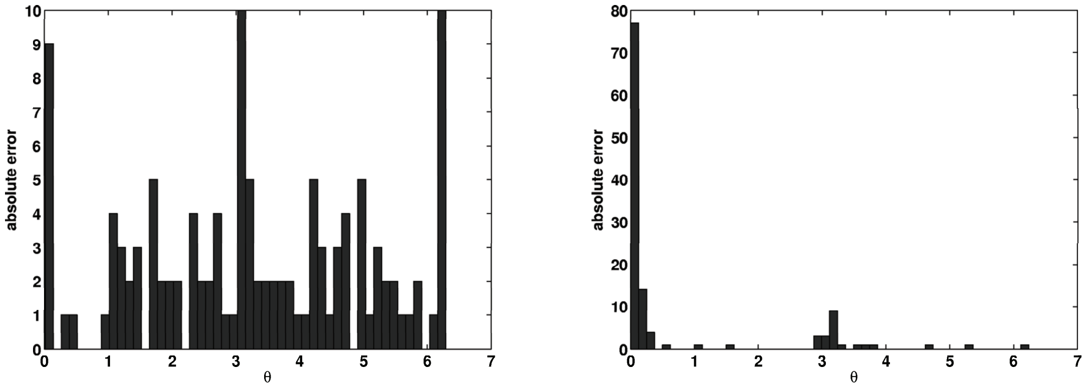
4.5 Transformation invariant face recognition

In this section we compare all methods in the context of transformation invariant face recognition. We use the ORL (formerly Olivetti) database [91] which contains 40 individuals and 10 different images for each individual, including variation in facial expression (smiling/non smiling) and pose. We form the training set by picking the first 7 facial expressions/poses per subject and use the remaining 3 as a test set. This results into 280 training samples and 120 test samples. Our evaluation metric is the classification error rate which is the percentage of test samples that are misclassified.

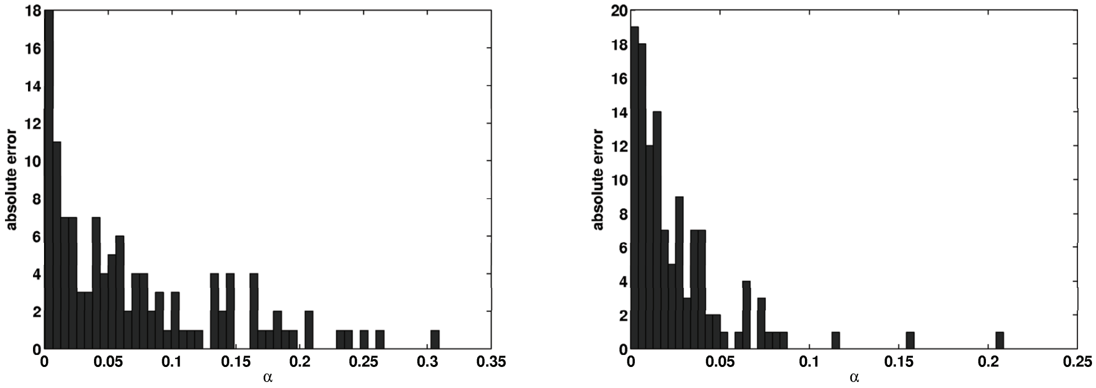
First, we compute the pattern model of all training images using $K = 50$ Gaussian atoms in OMP. This is an off-line step and is performed once and for all. Then, each method is combined with the nearest-neighbor (NN) classifier yielding a transformation-invariant classifier. In particular, each test face image is first geometrically transformed and then compared to each training face image. The test sample is given the class label of the “closest” training image. The distance between the test and the training image is computed according to the method that is tested; (i) for ED it is the Euclidean distance between the two, (ii) for TD it is the tangent distance, and (ii) for MD and MDDC it is the Euclidean distance obtained after alignment. In the latter case, alignment is accomplished by estimating the transformation parameters first, and then by image warping using the estimated parameters. For computational ease, the test image is aligned with only one representative image per subject; namely the first facial expression/pose. Then, the same transformation is used for aligning the test image with the remaining images of the same subject.

Note that this experiment is more challenging than the previous one, because the training and the test images (which are to be aligned) correspond to different expressions/poses of the same person or to different persons. In MDDC, we use 30 iterations of the cutting plane method before switching to Newton. In the testing phase, each test facial image is scaled by a random α, uniformly distributed in $[0.5, 1.5]$ and rotated by a random θ, uniformly distributed in $[0, 2\pi)$. Each test image is transformed independently from its peers. Observe that the testing phase scales linearly with the training set size. Fig. 4.7 shows the transformed versions of the test facial images.

For the sake of completeness, we also include in our comparisons the virtual samples (VS) method. The virtual samples are simply transformed versions of the training samples that



(a) $|\theta - \hat{\theta}|$, Newton (b) $|\theta - \hat{\theta}|$, MDDC



(c) $|\alpha - \hat{\alpha}|$, Newton (d) $|\alpha - \hat{\alpha}|$, MDDC

Figure 4.8: *Histogram of the absolute error of the rotation angle (first row) and of the scale (second row).*

artificially generated. The latter method is employed as a baseline in order to show that expanding the training set with transformed samples can be helpful in robust recognition against the presence of transformations in the test set. In particular, for each training image we build a set of virtual samples by applying geometric transformations corresponding to each possible pair $(\theta_i, \alpha_j),\ i = 1, \ldots, n_\theta, j = 1 \ldots, n_s$. We discretize the rotation space $[0, 2\pi)$ with $n_\theta = 10$ points and the scale space $[0.5, 1.5]$ with $n_s = 5$ points. In the VS method the size of the training set is expanded by $n_\theta \cdot n_s$ times, which is 50 in our case. This makes it memory intensive, which is one of the main pitfalls of this method. Note that one may discretize even more finely the transformation space, but we should bear in mind that in practice this may not be always feasible. This method typically bridges the gap between local methods (i.e., that are built for small transformations) and global methods, like ours.

Table 4.1 shows the classification error rates of all methods, under both L1 and L2 metrics. Notice that ED, TD and MD break down due to the presence of large transformations. However, MDDC yields a satisfactory performance which shows its truly transformation invariant properties. The VS method has comparable performance to MDDC, at the cost of

Classif. error rate(%)	ED	TD	MD	VS	MDDC
L2	95.8	98.3	80.8	37.5	26.6
L1	94.2	-	81.6	29.2	22.5

Table 4.1: *Transformation invariant face recognition results.*

highly increased memory requirements though.

We further discuss the alignment performance of Newton and MDDC. We compute the histogram of the absolute errors of both scale and rotation using 50 bins. In particular, we report the absolute errors $|\theta - \hat{\theta}|$ and $|\alpha - \hat{\alpha}|$, where $\hat{\theta}$ and $\hat{\alpha}$ denote the estimated transformation parameters of rotation and scale respectively. We observed that in many cases the global minimizer in the alignment between *different* persons does not coincide with the exact transformation parameters. For this reason, in the histogram computation, we include only the alignment performance of the pairs corresponding to different expressions/poses of the *same* subject. Fig. 4.8 show the results of both Newton and MDDC. Notice that, as expected, Newton is not able to converge to the true transformation parameters, due to its local convergence property. On the other hand, MDDC is successful in the vast majority of the cases. The very few outlier cases are due to the fact that we run only 30 iterations of the cutting plane algorithm (before switching to Newton).

Discussion The above experimental results indicate that our algorithm converges to the global minimizer in the presence of large transformations, when they consist of synthesis of translations, rotations and scalings. Our goal is to show that our method is applicable to robust pattern classification and image registration problems, while enjoying at the same time theoretical merits of convergence to the globally optimal solution. These examples are however mostly illustrative, and do not permit to extrapolate that the proposed framework alone will provide state of the art performances in more generic problems, like image registration in the presence of occlusion, or background noise.

Also, we can note that the proposed framework has targeted transformations that can be represented as a composition of rotation, translation and isotropic scaling. While such transformations represent of a wide range of image processing applications, the proposed framework could also be applied to more generic transformations models, such as fully affine, non-rigid etc, as long as the effect of the transformation on a set of atoms can be properly expressed in a DC form.

4.6 Related work

In Section 2.3, we reviewed the most related methods from the literature on transformation invariant distance measures. Furthermore, we provided an experimental evaluation of our framework with respect to these methods in Sections 4.4 and 4.5, and the results show that our method is superior to them in both pattern alignment as well as robust face recognition. In what follows, for the sake of completeness, we review some additional methods from the literature that touch upon transformation invariance issues in pattern analysis, although they are not directly related to the framework proposed in this chapter. We start by methods that introduce invariance in pattern recognition.

A lot of research efforts have focused on introducing transformation invariance into Support Vector Machines (SVMs). This can be attempted by the following approaches (see [18] for more details and references). (i) Design of invariant kernels [34]. (ii) Introduction of virtual support vectors [97, 98]. In particular, Schölkopf et al. introduced invariance into SVMs by generating virtual examples from the support vectors, rather than the whole training set (Virtual SVM). This yields the training phase more computationally efficient since the training set size becomes controlled now. (iii) Kernel jittering [17], where the main idea is the following; instead of performing the transformations on the training examples before training, they are performed on-demand during kernel evaluation.

In the context of maximum margin classifiers, Graepel et al [31] proposed the semidefinite programming machines (SDPM). They represent the transformation manifold of each training sample by a polynomial trajectory (using Taylor expansion) and then formulate the margin maximization as a semidefinite program. Experimentally, SDPM performed slightly better than the virtual SVM. However, extending their methodology to more generic transformations consisting of more than one parameters is complex.

Another approach towards invariance in pattern recognition is to encode invariance into the classification algorithm itself. For example, Simard et al in [100] modified the error function of neural networks in order to make it invariant to transformations. In particular, they included an additional term in the error function, which captures the magnitude of the gradient of the classifying function with respect to the transformation parameters. Thus, minimizing the total error function results in a classification function that is smooth with respect to the desired transformations.

We should also mention that there are research efforts that attack the problem of interest from a different combinatorial perspective. The authors in [43] provide a combinatorial treatment of pattern matching by considering solely discrete images. The main result in [43] is that the image matching problem under affine transformations can be solved in polynomial time. They propose an exhaustive search algorithm over a database of all possible affine transformations of the discrete image. They show that the size of the database is bounded polynomially with respect to the size of the image. The main idea of the proof relies on the fact that the parameter space of continuous affine transformations in R^6 can be partitioned into equivalence classes. Two parameter vectors belong to the same equivalence class if they map to the same discrete transformed image (under the assumption of nearest neighbor interpolation). The authors show that the number of equivalence classes is bounded polynomially and that the complexity of their combinatorial algorithm scales as $\mathrm{O}(N^{18})$, where N is the image size. Although, the complexity is polynomial, the algorithm becomes inefficient for large image sizes. In a follow up article [44], the same authors study the particular problem of pattern matching under combined scale and rotation. They propose sharper bounds on the size of the database of all possible transformed patterns, and they provide an improved algorithm of smaller complexity for this case, which is however still based on exhaustive search.

Finally, it is important to stress [113] that the problem of computing the manifold distance is closely related to the problem of image alignment or registration. For instance, the article in [108] proposes the use of multiresolution decomposition combined with Levenberg-Marquardt for image registration, which is close to the idea proposed in [113]. The authors in [76, 11] propose a maximum likelihood (ML) approach for translation, rotation and scale estimation. The idea is to loop over the pixels and for each pixel to estimate scale and rotation by maximizing an approximation of the log-likelihood using quasi-Newton. The latter approximation is obtained by expanding the images in the Laguerre-Gauss transform domain.

This approach does an exhaustive search over the possible translation parameters and solves the scale-rotation problem for each pixel. On the contrary our methodology solves the full problem all at once. The reader is referred to [132] for a survey on image registration.

4.7 Conclusions

In this chapter we considered the problem of estimating the transformation parameters for pattern alignment and manifold distance computation. This problem allows for meaningful comparison between geometrically transformed patterns. We have proposed a globally optimal method, which uses sparse geometric expansions to represent the transformation manifold. A few atoms are sufficient to capture the main geometric structure of the pattern, which is further used for alignment. We formulate the minimization of the distance of the reference pattern from the manifold, as a DC program, by proving that the objective function is DC. We solve the DC program using an outer approximation - cutting plane method which converges to the globally optimal solution. The experimental results show that the proposed method is successful in finding the global minimizer in practice, with applications to robust face recognition and image alignment. In the next chapter, we study the problem of pattern classification, when multiple transformed observations of the test pattern are available.

Chapter 5

Classification of Multiple Observations

5.1 Introduction

In this chapter, we focus on the problem of classifying multiple transformed observations of the same test object. We assume that each observation is produced from the same object under a certain generic transformation, and the problem is to predict the unknown class. In this context, classification methods have typically to exploit the diversity of the multiple observations in order to provide increased classification accuracy [102]. For instance, high-dimensional visual information sets typically exhibit a low-dimensional manifold structure and exploiting this particular structure is important.

The problem of classification of multiple observations can be identified as a particular case of semi-supervised learning (SSL) [13]. SSL refers to the type of learning where the test unlabelled data are available in the training phase; the challenge is to exploit this extra information in order to improve the classification performances. Notice that all unlabelled examples typically belong to the same unknown class in our problem of classification of multiple observation sets. A classification algorithm for multiple observations should be able to benefit from this special structure of the problem. We design two methodologies, which build on SSL and take explicitly into account this key observation.

We first show that this particular classification problem can be solved with a modified Transductive Support Vector Machine algorithm. Due to the complexity limitations of such a solution, we further propose a novel graph-based algorithm built on label propagation [16]. Label propagation methods typically assume that the data lie on a low dimensional manifold living in a high dimensional space. They rely upon the *smoothness assumption*, which states that if two data samples x_1 and x_2 are close, then their labels y_1 and y_2 should be close as well. The main idea of these methods is to build a graph that captures the geometry of this manifold as well as the proximity of the data samples. The labels of the test examples are derived by "propagating" the labels of the labelled data along the manifold, while making use of the smoothness property. We exploit the specificities of our particular classification problem and constrain the unknown labels to correspond to one single class. This leads to the formulation of a discrete optimization problem that can be optimally solved by a simple and low complexity algorithm.

We apply the proposed algorithms to the classification of sets of multiple images in handwritten digit recognition or video-based face recognition. In particular, we illustrate the high potential in graph-based methods for efficient classification of images that belong to the same data manifold. For example, we show that the graph-based solution outperforms state of the art subspace or statistical classification methods in video-based face recognition. Hence, this chapter establishes new connections between semi-supervised learning and video-based face recognition, where graph-based solutions are certainly very promising.

The rest of the chapter is organized as follows. We first formulate the problem of classification of multiple observation sets in Section 5.2. We propose a modified TSVM solution in Section 5.3 and a graph-based algorithm inspired by label propagation in Section 5.4. Then we demonstrate the performance of the proposed classification methods for handwritten digit recognition and video-based face recognition in Section 5.5 and Section 5.6, respectively.

5.2 Problem definition

We address the problem of the classification of multiple observations of the same object, possibly with some transformations. In particular, the problem is to assign multiple observations of the test pattern/object s to a single class of objects. We assume that we have m transformed observations of s of the following form

$$x_j^{(u)} = U(\eta_j)s, \;\; j = 1, \dots, m,$$

where $U(\eta)$ denotes a (geometric) transformation with parameters η, which is applied on s. For instance, in the case of visual objects, $U(\eta)$ may correspond to a rotation, scaling, translation, or perspective projection of the object. We assume that each observation $x_j^{(u)}$ is obtained by applying a transformation η_j on s, which is different from its peers (i.e., $\eta_i \neq \eta_j$, for $i \neq j$). The problem is to classify s in one of the c classes under consideration, using the multiple observations $x_j^{(u)}, \; j = 1, \dots, m$.

Assume further that the data set is organized in two parts $X = \{X^{(l)}, X^{(u)}\}$, where $X^{(l)} = \{x_1, x_2, \dots, x_l\} = \{x_1^{(l)}, x_2^{(l)}, \dots, x_l^{(l)}\} \subset \mathbb{R}^d$ and $X^{(u)} = \{x_{l+1}, \dots, x_n\} = \{x_1^{(u)}, \dots, x_m^{(u)}\} \subset \mathbb{R}^d$, where $n = l + m$. Let also $\mathcal{L} = \{1, \dots, c\}$ denote the label set. The l examples in $X^{(l)}$ are labelled $\{y_1, y_2, \dots, y_l\}, \; y_i \in \mathcal{L}$, and the m examples in $X^{(u)}$ are unlabelled. The classification problem can be formally defined as follows.

Problem 1. *Given a set of labelled data* $X^{(l)}$*, and a set of unlabelled data* $X^{(u)} \triangleq \{x_j^{(u)} = U(\eta_j)s, \; j = 1, \dots, m\}$ *that correspond to multiple transformed observations of* s*, the problem is to predict the correct class* c^* *of the original pattern* s*.*

This problem is a particular case of semi-supervised learning [13], which generally consists in predicting the labels of $X^{(u)}$, based on the knowledge of the data points (both $X^{(l)}$ and $X^{(u)}$) and the labels of the labelled points. Note that in the generic scenario of semi-supervised learning, the test examples may belong to different classes. This problem however presents an important additional constraint, where all the observations belong to the same class. Thus, one may view Problem 1 as a special case of semi-supervised learning, where the unlabelled data $X^{(u)}$ represent the multiple observations and they have the extra constraint that all unlabelled data examples belong to the same (unknown) class. The problem then resides in estimating the unknown class.

It has to be noted here that robustness to pattern transformations is a very important property of the classification of multiple observations. Transformation invariance can be introduced into classification algorithms by augmenting the labelled examples with the so-called *virtual samples*, denoted hereby as $X^{(\mathrm{vs})}$ (see [86] for a similar approach). Recall that the virtual samples are essentially data samples that are generated artificially, by applying transformations to the original data samples. It is common practice to generate the virtual samples by uniform sampling of the transformation parameter space. They are given the class labels of the original examples that they have been generated from, and are treated as labelled data. By including the virtual samples in the data set, any classification algorithm becomes more robust to transformations of the test examples[1]. We therefore adopt this strategy in the proposed methods and we include n_{vs} virtual samples $X^{(\mathrm{vs})}$ in our original data set that is finally written as $X = \{X^{(l)}, X^{(\mathrm{vs})}, X^{(u)}\}$.

In the next sections, we propose two novel methods to solve Problem 1, which are respectively based on Transductive Support Vector Machines (TSVM) and label propagation (see Section 2.4.1 for more details on these two methods).

5.3 Classification with modified TSVM

Algorithm 3 The TSVM algorithm

```
Input:
  $X^{(l)} \in \mathbb{R}^{d \times l}$: labelled data.
  $X^{(u)} \in \mathbb{R}^{d \times m}$: unlabelled data.
  $t$: RBF kernel width.
  $C$: the soft margin parameter.
Output:
  $\hat{p}$: estimated unknown class.
Initialization:
$r = [0, \dots, 0] \in \mathbb{R}^c$
for $p = 1 : c$ do
    {Assume that $X^{(u)}$ belongs to the $p$th class}
    Form the positive examples $X_+ = \{X^{(l)}_{y_i = p} \bigcup X^{(u)}\}$.
    Form the negative examples $X_- = \{X^{(l)}_{y_i \neq p}\}$.
    Form $Y_+$ and $Y_-$.
    {Train a binary SVM}
    $\{\alpha, b_0, X_{SV}\}$ = trainSVM($[X_+, Y_+]$, $[X_-, Y_-]$, $C$, $t$);
    Compute the margin $r(p) = \frac{1}{w^\top w} \left( = \frac{1}{\alpha^\top K \alpha} \right)$
end for
$\hat{p} = \arg\max_p r(p)$
```

We propose now to modify the standard TSVM algorithm (see Section 2.4.1) in order to solve Problem 1 and exploit its special structure. Since all the unlabelled samples belong to the same class, we can rely on the maximum margin principle and modify the TSVM algorithm to solve the problem of classification of multiple observations.

---

[1] Although the inclusion of virtual samples induces a memory cost, we should mention however that this cost gets amortized among the multiple test examples.

The main idea is simply to loop over the different class hypotheses and output as the estimated class $\hat{p}$ the one that maximizes the margin, as presented in Algorithm 3. For each hypothesized class p, the modified TSVM algorithm trains a binary standard SVM (Line 11), where:

- the positive examples X_+ are formed by the union of the unlabelled data $X^{(u)}$ and the labelled data $X^{(l)}_{y_i=p}$ of the class p (Line 7), and
- the negative examples X_- consist of the rest of the labelled data $X^{(l)}_{y_i \neq p}$ (Line 8).

The algorithm computes the margin $r(p)$ for each hypothesis p (Line 12), and finally, the hypothesis that results in the largest margin is provided as the estimated class $\hat{p}$.

Algorithm 3 accommodates the special structure of Problem 1 and instantiates the TSVM classification for the classification of multiple observation patterns. Note however that Algorithm 3 does not explicitly model the manifold structure of the data. It is possible to employ a nonlinear SVM to capture the nonlinearities that are present in the data, but this comes at the price of choosing an appropriate kernel and its hyperparameters.

Finally, the total computational cost of the above TSVM algorithm is $O(c \cdot n^2)$, since the SVM training that scales as $\mathrm{O}(n^2)$, is repeated for each class hypothesis. Thus, the cost of the above method becomes quite considerable when the data set grows large and when the number of classes increases. For the above reasons, we propose in the next section, a low complexity graph-based method that captures the manifold structure of the data by means of a graph.

5.4 Graph-based classification

5.4.1 Label propagation with multiple observations

We propose now to build on graph-based algorithms to solve the problem of classification of multiple observations. In general, label propagation assumes that the unlabelled examples come from different classes. As Problem 1 presents the specific constraint that all unlabelled data belong to the same class, label propagation does not fit exactly the definition of the problem as it falls short of exploiting its special structure. Therefore, we propose in the sequel a novel graph-based algorithm, which (i) uses the smoothness criterion on the manifold in order to predict the unknown class labels and (ii) at the same time, it is able to exploit the specificities of Problem 1.

We represent the data labels with a 1-of-c encoding, which allows to form a binary label matrix of size $n \times c$, whose ith row encodes the class label of the ith example. The class label is basically encoded in the position of the nonzero element.

Suppose now that the correct class for the unlabelled data is the pth one. In this case, we denote by $Z_p \in R^{n \times c}$ the corresponding label matrix and we call it the pth *class-conditional label matrix*. Note that there are c such label matrices; one for each class hypothesis. Each matrix Z_p has the following form

$$Z_p = \left[\begin{array}{c} Y_l \in R^{l \times c} \\ \hline \mathbf{1} e_p^\top \in R^{m \times c} \end{array} \right] \in R^{n \times c}, \tag{5.1}$$

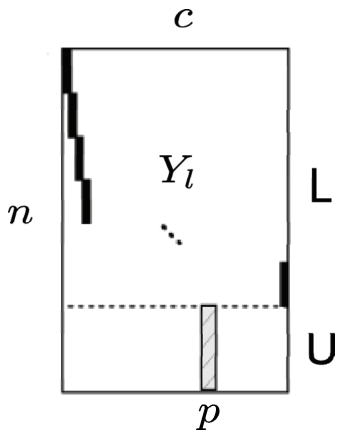


Figure 5.1: *Structure of the class-conditional label matrix Z_p.*

where $e_p \in R^c$ is the pth canonical basis vector and $\mathbf{1} \in R^m$ is the vector of ones. Fig. 5.1 shows schematically the structure of matrix Z_p. The upper part corresponds to the labelled examples and the lower part to the unlabelled ones. Z_p holds the labels of all data samples, assuming that all unlabelled examples belong to the pth class. Observe that the Z_p's share the first part Y_l and differ only in the second part.

Since all unlabelled examples share the same label, the class labels have a special structure that reflects the special structure of Problem 1, as outlined in our previous work [60]. We could then express the unknown label matrix M as,

$$M = \sum_{p=1}^{c} \lambda_p Z_p, \quad Z_p \in R^{n \times c}, \tag{5.2}$$

where Z_p is given in (5.1), $\lambda_p \in \{0, 1\}$ and

$$\sum_{p=1}^{c} \lambda_p = 1. \tag{5.3}$$

In the above, $\lambda = [\lambda_1, \dots, \lambda_c]$ is the vector of linear combination weights, which are discrete and sum to one. Ideally, λ should be sparse with only one nonzero entry pointing to the correct class.

The classification problem now resides in estimating the proper value of λ. We propose to make use of the smoothness assumption property, similar to the graph-based methods for semi-supervised learning. Notice that the fitting term in (2.6) is not needed anymore due to the structure of the Z matrices. Hence, we propose the following objective function

$$\tilde{\mathcal{Q}}(\lambda) = \frac{1}{2}\Big(\sum_{i,j=1}^{n} H_{ij} \| \frac{1}{\sqrt{D_{ii}}} M_i - \frac{1}{\sqrt{D_{jj}}} M_j \|^2 \Big), \tag{5.4}$$

where the optimization variable now becomes the λ vector. In the above, M_i (resp. M_j)

denotes the ith (resp. jth) row of M. In the case of normalized Laplacian, we have

$$\mathcal{Q}(\lambda) = \frac{1}{2}\sum_{i,j=1}^{n} S_{ij}\|M_i - M_j\|^2, \tag{5.5}$$

where S is defined as in (2.5). It can be seen that the objective function directly relies on the smoothness assumption. When two examples x_i, x_j are nearby (i.e., H_{ij} or S_{ij} is large), minimizing $\tilde{Q}(\lambda)$ and $Q(\lambda)$ results in class labels that are close too. In addition, we observe that λ is implicitly represented in the above equation through M, defined in eq. (5.2). The following proposition further shows the dependence of Q on λ.

Proposition 5. *Assume the data set is split into l labelled examples $X^{(l)}$ and m unlabelled examples $X^{(u)}$, i.e., $X = [X^{(l)}, X^{(u)}]$. Then, the objective function (5.5) can be written in the following form,*

$$Q(\lambda) = C + \frac{1}{2}\sum_{i\le l, j>l} S_{ij}\|Y_i - \lambda\|^2 + \frac{1}{2}\sum_{i>l, j\le l} S_{ij}\|Y_j - \lambda\|^2 \tag{5.6}$$

where $C = \sum_{i\le l, j\le l} S_{ij}\|Y_i - Y_j\|^2$.

Proof. From equation (5.5) observe that

$$\begin{aligned}\mathcal{Q}(\lambda) &= \underbrace{\frac{1}{2}\sum_{i,j\le l}^{n} S_{ij}\|M_i - M_j\|^2}_{Q_1} + \underbrace{\frac{1}{2}\sum_{i,j>l}^{n} S_{ij}\|M_i - M_j\|^2}_{Q_2} \\ &\quad + \underbrace{\frac{1}{2}\sum_{i\le l, j>l}^{n} S_{ij}\|M_i - M_j\|^2}_{Q_3} + \underbrace{\frac{1}{2}\sum_{i>l, j\le l}^{n} S_{ij}\|M_i - M_j\|^2}_{Q_4}.\end{aligned}$$

We consider the following cases

(i) $i \le l$ and $j \le l$ i.e., both data examples x_i and x_j are labelled. Then, $M_i = (\sum_{p=1}^{c}\lambda_p)Y_i = Y_i$, due to the special structure of the Z matrices (see (5.1)) and also due to the constraint (5.3). Similarly, $M_j = Y_j$. This results in $Q_1 = \frac{1}{2}\sum_{i,j\le l} S_{ij}\|Y_i - Y_j\|^2 = C$, which is a constant term and does not depend on λ.

(ii) $i > l$ and $j > l$ i.e., both data samples x_i and x_j are unlabelled. In this case, $M_i = \lambda$ and $M_j = \lambda$, again due to (5.1). Therefore the second term Q_2 is zero.

(iii) $i \le l$ and $j > l$ i.e., x_i is labelled and x_j is unlabelled. In this case, $M_i = Y_i$ and $M_j = \lambda$. This results in $Q_3 = \frac{1}{2}\sum_{i\le l, j>l} S_{ij}\|Y_i - \lambda\|^2$.

(iv) $i > l$ and $j \le l$ is analogous to the case (iii) above, where the roles of x_i and x_j are switched. Thus, $Q_4 = \frac{1}{2}\sum_{i>l, j\le l} S_{ij}\|Y_j - \lambda\|^2$.

Putting the above facts together yields (5.6). □

The above proposition suggests that only the interface between labelled and unlabelled examples matters in determining the smoothness value of a candidate solution vector λ. We use this observation in order to design an efficient graph-based classification algorithm that is described below.

Algorithm 4 The MASC algorithm

```
Input:
  $X \in \mathbb{R}^{d \times n}$: data examples.
  $m$: number of observations.
  $l$: number of labelled data.
Output:
  $\hat{p}$: estimated unknown class.
Initialization:
Form the $k$-NN graph $\mathcal{G}_d = (\mathcal{V}_d, \mathcal{E}_d)$.
Compute the weight matrix $H \in \mathbb{R}^{n \times n}$ and the diagonal matrix $D$, where $D_{i,i} = \sum_{j=1}^{n} H_{ij}$.
Compute $S = D^{-1/2} H D^{-1/2}$.
for $p = 1 : c$ do
    $M = \begin{bmatrix} Y_l \\ \hline \mathbf{1} e_p^\top \end{bmatrix}$
    $q(p) = \sum_{i \le l, j > l} S_{ij} \|M_i - M_j\|^2 + \sum_{i > l, j \le l} S_{ij} \|M_i - M_j\|^2$.
end for
$\hat{p} = \arg\min_p q(p)$
```

5.4.2 The MASC algorithm

We propose in this section a simple, yet effective graph-based algorithm for the classification of multiple observations from the same class. Based on Proposition 5 and ignoring the constant term, we need to solve the following optimization problem

Optimization problem: **OPT**
$\min_\lambda \sum_{i \le l, j > l} S_{ij} \|Y_i - \lambda\|^2 + \sum_{i > l, j \le l} S_{ij} \|Y_j - \lambda\|^2$
subject to
$\lambda_p \in \{0, 1\}$, $p = 1, \ldots, c$,
$\sum_{p=1}^{c} \lambda_p = 1$.

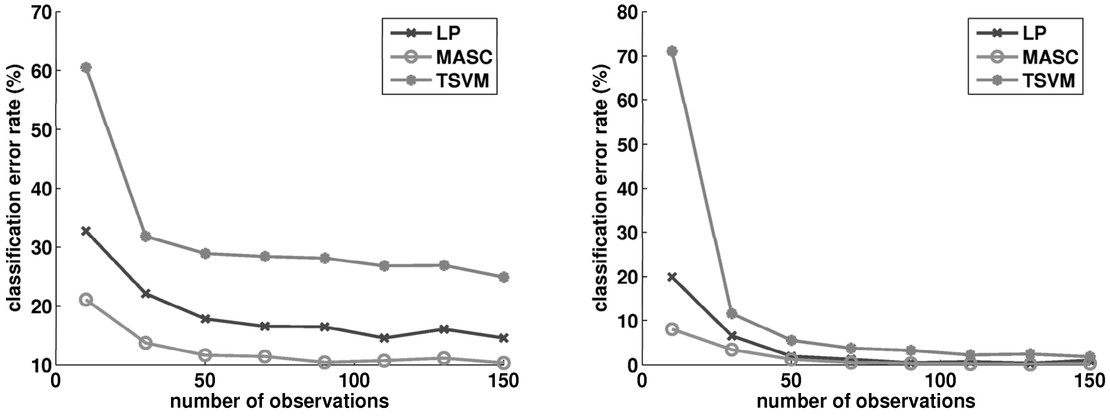
Intuitively, we seek the class that corresponds to the smoothest label assignment between labelled and unlabelled data. Observe that the above problem is a discrete optimization problem due to the constraints imposed on λ, that can be collected in a set Λ, where

$$\Lambda = \{\lambda \in R^{c \times 1} : \ \lambda_p \in \{0, 1\}, p = 1, \ldots, c, \sum_{p=1}^{c} \lambda_p = 1\}.$$

Notice that the search space Λ is small. In particular, it consists of the following c vectors:

$$\begin{array}{l} [1, 0, \ldots, 0, \ldots, 0] \\ [0, 1, \ldots, 0, \ldots, 0] \\ \ldots \\ [0, 0, \ldots, 1, \ldots, 0] \\ [0, 0, \ldots, 0, \ldots, 1]. \end{array}$$

Thus, one may solve OPT by enumerating all above possible solutions and pick the one λ^* that minimizes $Q(\lambda)$. Then, the position of the nonzero entry in λ^* yields the estimated unknown



(a) Binary digits (b) USPS digits

Figure 5.2: *Classification results measured on two different data sets.*

class. We call this algorithm **MA**nifold-based **S**moothing under **C**onstraints (MASC) and we show its main steps in Algorithm 4.

The MASC algorithm has computational complexity that is linear with the number of classes, and quadratic with the number of samples. The construction of k-NN graph (Lines 4-6) scales as $\mathrm{O}(n^2)$. Once the graph has been constructed, the enumeration of all possible solutions scales as $\mathrm{O}(c)$. We conclude that the total computational cost is $\mathrm{O}(n^2 + c)$, which is much lower than the complexity $\mathrm{O}(c \cdot n^2)$ of the modified TSVM solution proposed before, especially when the number of classes increases.

In what follows, we evaluate the performance of the proposed methods in classification of sets of multiple images in handwritten digit recognition and video-based face recognition. One may want to use particular image models or features depending on the context and the target application. However, our main goal here is to evaluate the behavior of the classification methods independently of particular choices on image models or feature representations. For this reason, we treat images in the classical pixel-based representation in all experiments that follow.

5.5 Classification of multiple images

We evaluate the performance of the classification algorithms based on TSVM, and label propagation, in the context of handwritten digit classification. Multiple transformed images of the same digit class form a set of observations, which we want to assign to the correct class. We use two different data sets for our experimental evaluation; (i) a handwritten digit image collection[2] and (ii) the USPS handwritten digit image collection. The first collection contains 20 × 16 bit binary images of "0" through "9", where each class contains 39 examples. The USPS collection contains 16 × 16 grayscale images of digits and each class contains 1100 examples.

We compare the classification performance of the MASC algorithm with the label prop-

---

[2] http://www.cs.toronto.edu/~roweis/data.html

agation (LP) method. In LP, the estimated class is computed by majority voting on the estimated class labels computed by (2.7). In our experiments, we use the same k-NN graph in combination with the Gaussian weights (2.4) in both LP and MASC methods. In order to determine the value of σ in (2.4) we adopt the following process; we pick randomly 1000 examples, compute their pairwise distances and then set σ equal to half of its median.

We first split the data sets into training and test sets by including 2 examples per class in the training set and the remaining are assigned to the test set. Each training sample is augmented by 4 virtual examples generated by successive rotations of it, where each rotation angle is sampled regularly in $[-40°, 40°]$. This interval has been chosen to be sufficiently small in order to avoid the confusion of digits '6' and '9'. Next, in order to build the unlabelled set $X^{(u)}$ (i.e., multiple observations) of a certain class, we choose randomly a sample from the test set of this class and then we apply a rotation on it by a random (uniformly sampled) angle $\theta \in [-40°, 40°]$.

The number of nearest neighbors was set to $k = 5$ for both binary digit collection and the USPS data set, in both methods. These values of k have been obtained by the best performance of LP on the test set. We try different sizes of the unlabelled set (i.e., multiple observations), namely $m = [10 : 20 : 150]$ (in MATLAB notation). For each value of m, we report the average classification error rate across 100 random realizations of $X^{(u)}$ generated from each one of the 10 classes. Thus, each point in the plot is an average over 1000 random experiments.

We also compare the graph-based algorithms with the TSVM method, discussed in Section 5.3. For the SVM implementation, we use the SPIDER machine learning library for MATLAB that is publically available[3]. The standard SVM used in Line 11 of Algorithm 3, consists of a nonlinear SVM with an RBF kernel. We rather opt for a nonlinear type of SVM, due to the manifold structure of the data (caused by the pattern transformations). In this case, the hyperparameters of the TSVM are the width t of the RBF kernel and the soft margin parameter C. Since the number of training examples is small, the hyperparameters are selected based on the test error. We used the following two-dimensional grid for the hyperparameters,

$$\{(t_i, C_j), t_i \in [10^{-2} : 10^{0.5} : 10], C_j \in [10^{-1} : 10^{0.5} : 100]\}.$$

This resulted in $t_1 = 0.3162$ and $C_1 = 10$ for the binary digits data set and $t_1 = 0.3162$ and $C_1 = 31.6228$ for the USPS data.

Figures 5.2(a) and 5.2(b) show the results over the binary digits and the USPS digits image collections, respectively. Observe first that increasing the number of observations gradually improves the classification error rate of all methods. This is expected since more observations of a certain pattern give more evidence, which in turn results in higher confidence in the estimated class label. Notice also that the graph-based methods outperform TSVM, which does not consider explicitly the manifold structure of the data. Finally, observe that the proposed MASC algorithm unsurprisingly outperforms LP in both data sets., since it is designed to exploit the particular structure of Problem 1.

---

[3] http://www.kyb.tuebingen.mpg.de/bs/people/spider/main.html

5.6 Video-based face recognition

5.6.1 Experimental setup

In this section we evaluate our graph-based algorithm in the context of face recognition from video sequence. In this case, the different video frames are considered as multiple observations of the same person, and the problem becomes the correct classification of this person. We evaluate in this section the behavior of the MASC algorithm in realistic conditions, i.e., under variations in head pose, facial expression and illumination. Note in passing that our algorithm does not assume any temporal order between the frames; hence, it is also applicable to the generic problem of face recognition from image sets.

We use two publically available databases; the VidTIMIT [92] and the first subset of the Honda/UCSD [65] database. The VidTIMIT database[4] contains 43 individuals and there are three face sequences obtained from three different sessions per subject. The data set has been recorded in three sessions, with a mean delay of seven days between session one and two, and six days between session two and three. In each video sequence each person performed a head rotation sequence. In particular, the sequence consists of the person moving his/her head to the left, right, back to the center, up, then down and finally return to center.

The Honda/UCSD database[5] contains 59 sequences of 20 subjects. In contrast to the previous database, the individuals move their head freely, in different speed and facial expressions. In each sequence, the subjects perform various in-plane and out-of-plane head rotations, improvising their own head movements. Each person has between 2 and 5 video sequences and the number of sequences per subject is variable.

For preprocessing, in both databases, we used first P. Viola's face detector [115] in order to automatically extract the facial region from each frame. Note that this typically results in misaligned facial images. Next, we downsampled the facial images to size 32×32 for computational ease. No further preprocessing has been performed, which brings our experimental setup closer to real testing conditions.

The proposed MASC method implements Gaussian weights (2.4) and sets $k = 5$ in the construction of the k-NN graph. It does not use any virtual samples, since the training sequence contains already sufficient number of data examples. We compare MASC to two well-known methods from the literature, which mostly gather algorithms based on either subspace analysis or density estimation (statistical methods):

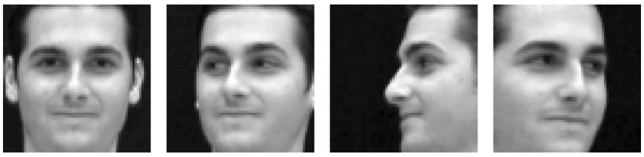
- MSM. The Mutual Subspace Method [28, 124], which is the most well known representative of the subspace analysis methods. It represents each sequence by a subspace spanned by the principal components, i.e., eigenvectors of the covariance matrix. The comparison of a test sequence with a training one is then achieved by computing the *principal angles* [29] between the two subspaces. In our experiments, the number of principal components has been set to nine, which has been found to provide the best performance.

- KLD. The KL-divergence algorithm by Shakhnarovich et al [99] is the most popular representative of density-based statistical methods. It formulates the video-based face recognition problem as a statistical hypothesis testing problem. Under the i.i.d and the Gaussian assumptions on the face sequences, this classification problem typically boils

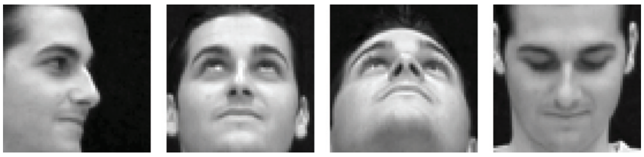
---

[4] http://users.rsise.anu.edu.au/~conrad/vidtimit/

[5] http://vision.ucsd.edu/ leekc/HondaUCSDVideoDatabase/HondaUCSD.html



(a) pose 1 (b) pose 2 (c) pose 3 (d) pose 4



(e) pose 5 (f) pose 6 (g) pose 7 (h) pose 8

Figure 5.3: *Head pose variations in the VidTIMIT database.*

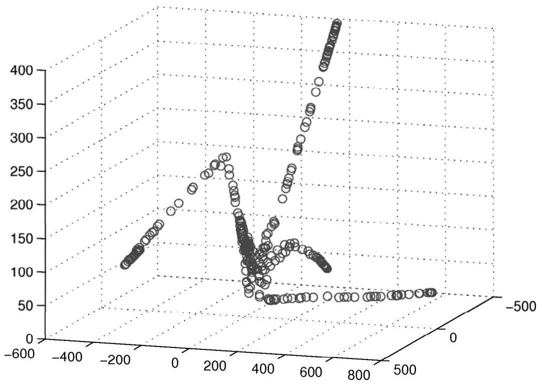


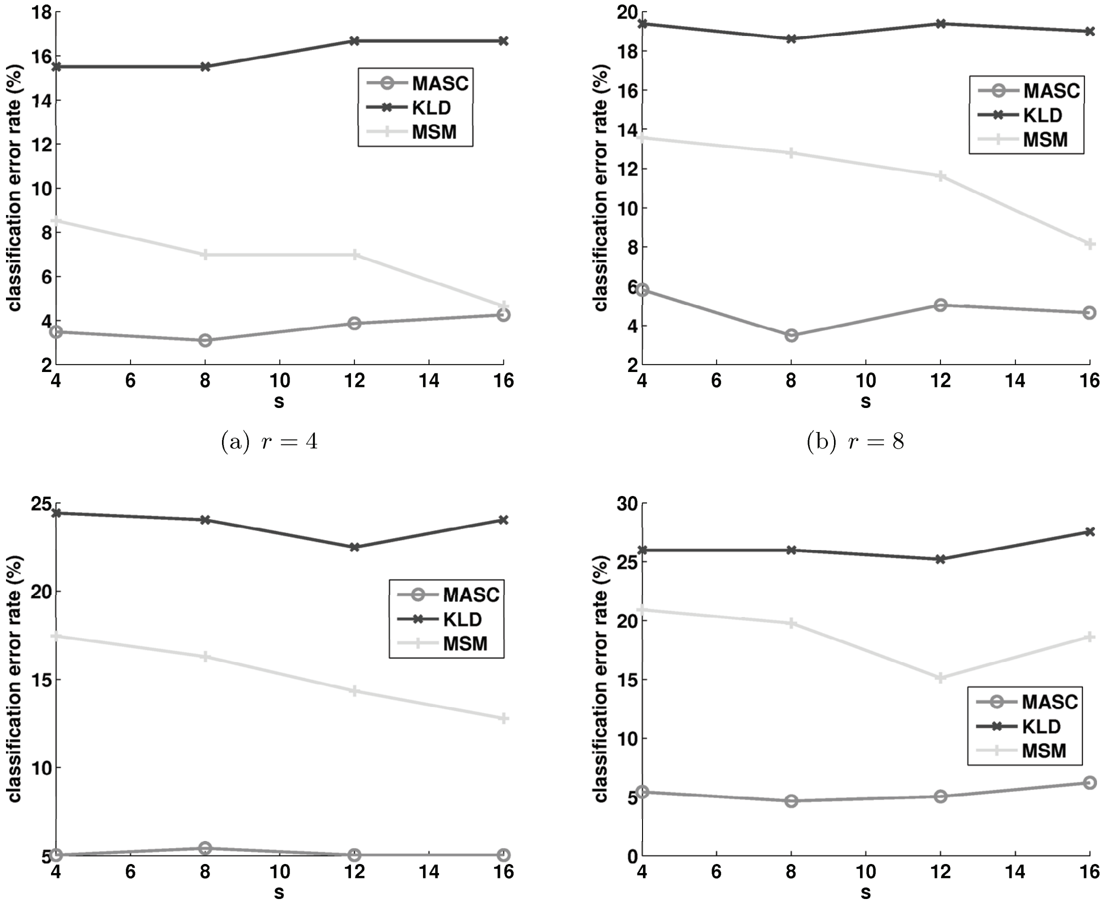
Figure 5.4: *A typical face manifold from the VidTIMIT database. Observe the four clusters corresponding to the four different head poses (face looking left, right, up and down).*

down to a computation of the KL divergence between sequences. The energy cut-off, which determines the number of principal components used in the regularization of the covariance matrices, has been set to 0.96.

5.6.2 Classification results on VidTIMIT

We first study the performance of the MASC algorithm with the VidTIMIT database. Figure 5.3 shows a few representative images from a sample face manifold in the VidTIMIT database. Observe the presence of large head pose variations. Figure 5.4 shows the 3D projection of the manifold that is obtained using the ONPP method [62], which has been shown to be an effective tool for data visualization. Notice the four clusters corresponding to the four different head poses i.e., looking left, right, up and down. This indicates that a graph-based method should be able to capture the geometry of the manifold and propagate class labels based on the manifold structure.

Since there are three sessions, we use the following metric for evaluating the classification



(a) $r = 4$ (b) $r = 8$

(c) $r = 12$ (d) $r = 16$

Figure 5.5: *Video face recognition results on the VidTIMIT database.*

performances

$$\overline{e} = \frac{1}{6}\sum_{i=1}^{3}\sum_{j=1,j\neq i}^{3} e(i,j), \tag{5.7}$$

where $e(i,j)$ is the classification error rate when the ith session is used as training set and the jth session is used as test set. In words, $\overline{e}$ is the average classification error rate calculated over the following six experiments, namely (1,2), (2,1), (1,3), (3,1), (2,3) and (3,2).

We evaluate the video face recognition performance of all methods for diverse training and test set sizes. The objective is to assess the robustness of the methods with respect to the size of the training and/or the test set. For this reason, each image set, say the ith, is re-sampled as follows

$$X_{\text{train}}^{(i)} = X_i(:, 1:r:n),\ i = 1,\ldots,c \tag{5.8}$$

$$X_{\text{test}}^{(i)} = X_i(:, 1:s:n),\ i = 1,\ldots,c. \tag{5.9}$$

In the above, each image set is re-sampled with step r if it is used as training set, and with

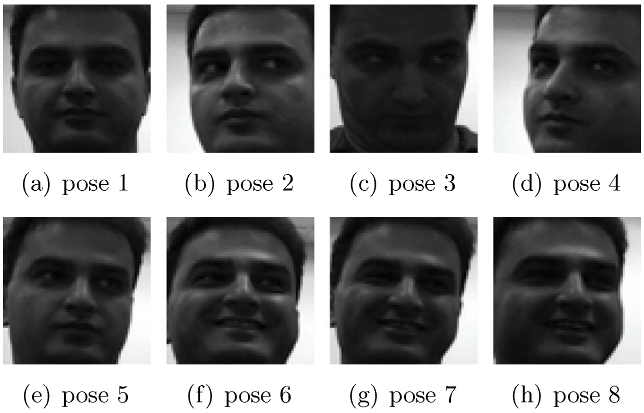
Recognition rate (%)	MASC	MSM	KLD
$r = 4$	100	84.62	84.62
$r = 6$	100	84.62	79.49
$r = 8$	97.44	84.62	61.54
$r = 10$	97.44	87.18	66.67
$r = 12$	97.44	76.92	61.54

Table 5.1: *Video face recognition results on the Honda/UCSD database.*

step s if it represents a test set. In our experiments, we use $r \in [4:4:16]$ and $s \in [4:4:16]$ (in MATLAB notation). For each pair of r and s values, we measure the average classification error rate according to relation (5.7).

Figures 5.5(a)-5.5(d) show the classification performance, for r from 4 to 16 with step 4, respectively. Observe that the KLD method that relies on density estimation is sensitive to the number of the available data. Also, notice that MSM is superior to KLD, which is expected since KLD relies on the imprecise assumption that data follow a Gaussian distribution. Finally, we observe that MASC clearly outperforms its competitors in the vast majority of cases. At the same time, it stays robust to significant re-sampling of the data, since its performance remains almost the same for each value of r and s.

5.6.3 Classification results on Honda/UCSD


(a) pose 1 (b) pose 2 (c) pose 3 (d) pose 4

(e) pose 5 (f) pose 6 (g) pose 7 (h) pose 8

Figure 5.6: *Head pose variations in the Honda/UCSD database.*

We further study the video-based face recognition performance on the Honda/UCSD database. Figure 5.6 shows a few representative images from a sample face manifold in the Honda/UCSD database. Observe the presence of large head pose variations along with facial expressions. The projection of the manifold on the 3D space using ONPP shows again clearly the manifold structure of the data (see Figure 5.7), which implies that a graph-based method may be more suitable for such kind of data.

The Honda/UCSD database comes with a default splitting into training and test sets, which contains 20 training and 39 test video sequences. We use this default setup and we

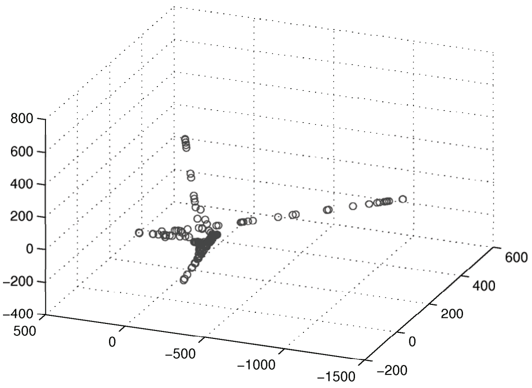


Figure 5.7: *A typical face manifold from the Honda/UCSD database.*

report the classification performance of all methods, under different data re-sampling rates. In particular, both training and test image sets are re-sampled now with step r i.e., $X^{(i)} = X_i(:, 1 : r : n),\ i = 1, \ldots, c$. Table 5.1 shows the recognition rates, when r varies from 4 to 12 with step 2. Figure 5.8 shows the same results graphically. Recall that larger values of r imply sparser image sets. Observe again that KLD is mostly affected by r, by suffering loss in performance. This is not surprising since it is a density-based method and densities cannot be accurately estimated (in general) with a few samples. MSM seems to be more robust, yielding better results than KLD. Finally, MASC is again the best performer and it exhibits very high robustness against data re-sampling.

Regarding the relative performance of MASC and MSM, we should finally stress out that MSM uses a linear subspace model of the image sets, which fails to capture the nonlinearities in the data. On the other hand, the MASC method relies on a graph model that is more realistic and fits much better the manifold structure of the data. Furthermore, it provides a means to cope with the curse of dimensionality, since the intrinsic dimension of the manifolds is typically very small. We believe that graph methods have a great potential in this field.

5.6.4 Video-based face recognition overview

For the sake of completeness, we review briefly in this last section the state of the art in video-based face recognition. Typically, one may distinguish between two main families of methods; those that are based on subspace analysis and those that are based on density estimation (statistical methods). The most representative methods for these two families are respectively the MSM [28, 124] method and the solution based on KLD [99], which have been used in the experiments above.

Among the methods based on subspace analysis, we should mention the extension of principal angles from subspaces, to nonlinear manifolds. In a recent article [117] it was proposed to represent the facial manifold by a collection of linear patches, which are recovered by a non-iterative algorithm that augments the current patch until the linearity criterion is violated. This manifold representation allows for defining the distance between manifolds as integration of distances between linear patches. For comparing two linear patches, the authors propose a distance measure that is a mixture between (i) the principal angles and (ii)

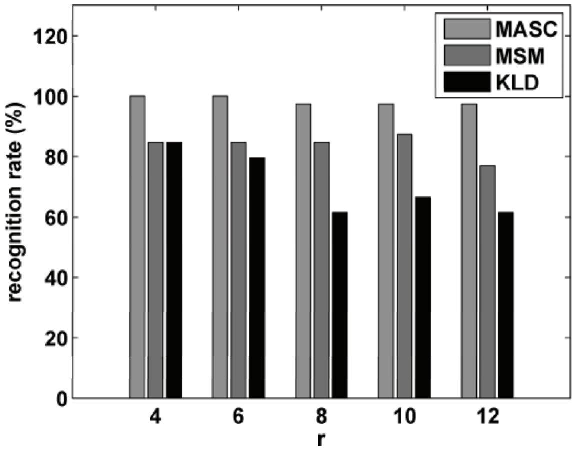


Figure 5.8: *Video face recognition results on the Honda/UCSD database.*

exemplar-based distance. However, it is not clearly justified why such a mixture is needed and what is the relative benefit over the individual distances. Moreover, their proposed method requires the computation of both geodesic and Euclidean distances as well as setting four parameters. On the contrary, our MASC method needs only one parameter (k) to be set and it requires the computation of the Euclidean distances only. Note finally that their method achieves comparable results with MASC on the Honda/UCSD database, but at a higher computational cost and at the price of tuning four parameters.

Along the same lines, the authors in [53] propose a similarity measure between manifolds that is a mixture of similarity between subspaces and similarity between local linear patches. Each individual similarity is based on a weighted combination of principal angles and those weights are learnt by AdaBoost for improved discriminative performance. In contrast to the previous paper [117], the linear patches are extracted here using mixtures of Probabilistic PCA (PPCA). PPCA mixture fitting is a highly non-trivial task, which requires an estimate of the local principal subspace dimension and it also involves model selection. Furthermore, as the authors claim themselves, this step is computationally intensive.

The main limitation of the statistical methods such as KLD [99] is the inadequacy of the Gaussianity assumption of face images sets; face sequences rather have a manifold structure. The test video frames are moreover not independent, so that the i.i.d assumption is unrealistic as well. The authors in [3] therefore extend the work of KL divergence by replacing the Gaussian densities by Gaussian Mixture Models (GMMs), which provides a more flexible method for density estimation. However, the KL divergence in this case cannot be computed in a closed form, which makes the authors to resort to Monte Carlo simulations that are quite computationally intensive.

Finally, there have been a few other methods that cannot be directly categorized in the above families of methods. The authors in [128] propose ensemble similarity metrics that are based on probabilistic distance measures, evaluated in Reproducing Kernel Hilbert spaces. All computations are performed under the Gaussianity assumption, which is unfortunately not realistic for facial manifolds.

In [129], the authors provide a probabilistic framework for face recognition from image sets. They model the identity as a discrete or continuous random variable and they provide a statistical framework for estimating the identity by marginalizing over face localization, illumination and head pose. Illumination-invariant basis vectors are learnt for each (discretized) pose and the resulting subspace is used for representing the low dimensional vector that encodes the subject identity. However, the statistical framework requires the computation of several integrals that are numerically approximated. Also, the proposed method assumes that training images are available for every subject at each possible pose and illumination, which is hard to satisfy in practice.

5.7 Conclusions

In this chapter, we have studied the problem of classification of multiple transformed observations of an object, which is often faced in modern multimedia architectures. We proposed a graph-based algorithm that exploits the fact that the multiple test observations share the same class label. The main idea is to rely on the smoothness assumption on the manifold, in order to learn the unknown label matrix, under the constraint that it satisfies the specificities of the problem. We have formulated this process as a discrete optimization problem, which can be solved efficiently by the graph-based algorithm. We provide experimental results that show that the proposed method outperforms (i) the label propagation and TSVM methods in handwritten digit classification, and (ii) well known methods that run state of the art solutions in video face recognition. In the next chapter, we will drop the assumption that the algorithm has access to all multiple observations and we will consider the problem in distributed settings.

Chapter 6

Distributed Classification of Multiple Observations

6.1 Introduction

In the previous chapter we studied the problem of classification of multiple object observations, assuming that these observations are collected in a central computer and they are all available to the algorithm. In this chapter, we drop this assumption and we consider the case where the multiple observations are collected in a distributed fashion and therefore, the algorithm has partial access to them. This scenario often arises in ad-hoc vision sensor networks (VSN), such as the one illustrated in Fig. 6.1. In such cases each vision sensor captures an observation of the object and the multiple observations are collected distributively. Ad-hoc network architectures may have an arbitrary topology and they are characterized by the fact that there is no central coordinator node. Moreover, due to the limited communication range, it is typical that each sensor can only communicate with a few of its closest neighbors. In this context, classification becomes even more challenging, since the decision has to be reached globally by means of collaborative processing, which moreover has to be effective in terms of both communication and computational cost. The problem now is to classify the observed object *at all sensors* such that they reach a consensus decision, by aggregating information across all observations.

At the same time, distributed consensus [6] is becoming increasingly popular, attracting numerous research efforts. In general the main goal of consensus is to reach a global solution iteratively using only local computation and communication, while staying robust to changes in the network topology. Having in mind that the sensors are very simple devices with limited computing capabilities, these characteristics of consensus are very appealing. Additionally, consensus-based methods for distributed classification in ad-hoc sensor networks have recently started to emerge. For instance, we mention the work in [25] for distributed SVM training based on consensus, and in [109] for consensus-based pose estimation applied to distributed face recognition. Hence, the problem of distributed classification in ad-hoc sensor networks becomes increasingly important.

In this chapter, we present a distributed version of the MASC algorithm, introduced in the previous chapter, which (i) is of low complexity that is suitable for sensor networks and (ii) is solely based on consensus-based distributed averaging. Each sensor captures an observation

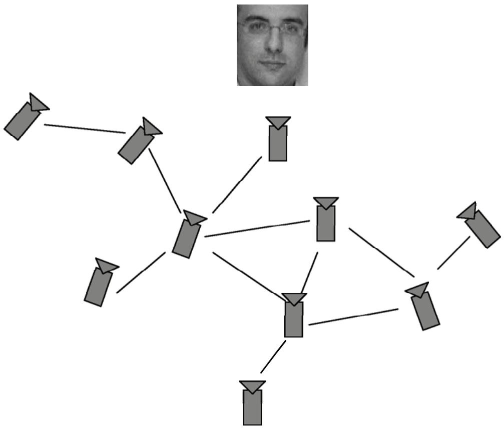


Figure 6.1: *Ad-hoc network of vision sensors*

of the object and computes its nearest neighbors among the labelled examples. Under a certain class hypothesis, those neighbors contribute to the (local) computation of a portion of the value of the objective function, which captures the smoothness of the current label assignment. Those portions are averaged distributively by means of average consensus, so that all observations are progressively taken into account and the total value of the objective function is computed *at all sensors*. This process is repeated for all class hypotheses and eventually the sensors reach a consensus classification decision. We show the feasibility of our algorithm in the context of distributed multi-view face recognition in a VSN, where observations correspond to facial images captured under different viewpoints.

The rest of this chapter is organized as follows. First, we formulate the problem formally in Section 6.2 and in Section 6.3 we introduce the distributed MASC algorithm, which is solely based on consensus-based distributed averaging. In the sequel, in Section 6.4, we show the feasibility of our algorithm in the context of distributed multi-view face recognition. Next, for the sake of completeness, we provide in Section 6.5 an overview of methods for distributed classification in more general settings and finally, we provide our concluding remarks in Section 6.6.

6.2 Problem formulation

Let us formally define the problem of distributed classification in an ad-hoc sensor network. We consider a network of m sensors, which is modelled by a graph $\mathcal{G}_s = (\mathcal{V}_s, \mathcal{E}_s)$. As we have already seen, W is used to denote the weight matrix of $\mathcal{G}_s$. We assume that each sensor j captures a single (unlabelled) observation $x_j^{(u)}$ of an object f. Each observation is different from its peers and has the following form,

$$x_j^{(u)} \triangleq U(\eta_j)f, \; j = 1, \ldots, m. \tag{6.1}$$

In the above, $U(\eta_j)$ denotes the transformation applied on the object f with parameters η_j. Hence, there are m observations of the object f that are collected distributively over the

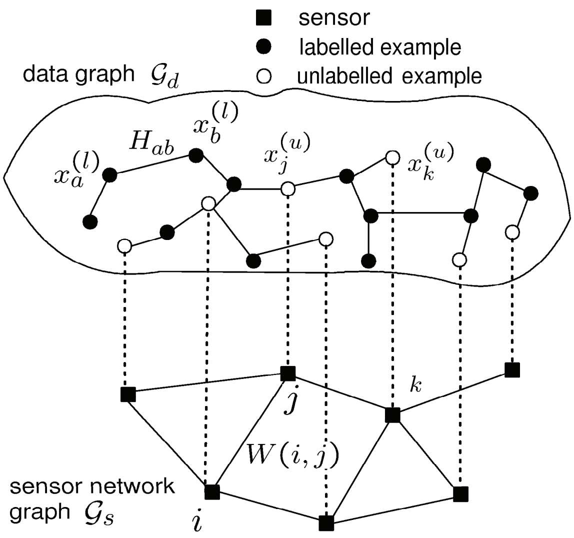


Figure 6.2: *Conceptual distinction between the two graphs of the problem. $\mathcal{G}_s$ (resp. $\mathcal{G}_d$) denotes the graph of the sensor network topology (resp. the data graph). In $\mathcal{G}_d$, the filled (resp. empty) circles correspond to labelled (resp. unlabelled) examples.*

sensor network and there is one-to-one correspondence among sensors and observations. The problem of distributed classification can be formally defined as follows.

Problem 2. *Assume that each sensor j has a copy of the labelled set $\{X^{(l)}, Y^{(l)}\}$ in addition to its single observation $x_j^{(u)}$ defined in (6.1). The topology of the sensor network is modelled by a graph $\mathcal{G}_s$ along with its weight matrix W. Assume also that each sensor knows only its neighbors and the weights of the links to them. The problem is to predict the correct class c^* of the object of interest f, by aggregating information from all available observations over the network, such that all sensors reach a consensus decision.*

In the following section, we introduce our methodology for distributed classification in ad-hoc sensor networks, using distributed consensus as our main computational tool. In order to attack Problem 2, we propose a distributed version of the MASC algorithm introduced in the previous chapter. Recall that MASC employs a graph to capture the pairwise similarities among the data samples. To avoid any confusion with the graph $\mathcal{G}_s$ of the sensor network, we denote the data graph by $\mathcal{G}_d$ and its weight matrix by H. Figure 6.2 shows the conceptual distinction between the two graphs and summarizes the related notation.

6.3 Distributed MASC

6.3.1 Preliminaries

Recall from the previous chapter that the objective function of MASC captures the smoothness of a candidate label assignment over the manifold, and it is of the following form

$$Q(\lambda) = \sum_{i=1}^{l} \sum_{j=l+1}^{n} S_{ij} \|Y_i - \lambda\|^2 + \sum_{j=l+1}^{n} \sum_{i=1}^{l} S_{ji} \|Y_i - \lambda\|^2, \tag{6.2}$$

where we have dropped the constant terms. Note that when i is fixed, the first term involves a sum over all unlabelled examples. However, due to the restrictions of the distributed settings, this term is hard to compute, since each sensor does not have access to all unlabelled examples (except from its own). Hence, for reasons of computational complexity, we drop the first term and turn our attention on the second one. Then, the objective function becomes

$$Q_d(\lambda) = \sum_{j=l+1}^{n} \Big(\sum_{i=1}^{l} S_{ji} \|Y_i - \lambda\|^2 \Big) = \sum_{j=l+1}^{n} r(j), \tag{6.3}$$

where we have introduced the subscript d to associate it with the distributed settings. In the above sum, the index j runs over the unlabelled examples and the index i over the labelled ones. Observe that for a certain j, the computation within the parentheses can be performed locally within each sensor, as it involves a weighted summation over the labelled examples. Hence, once the $r(j)$'s have been computed, the computation of $Q_d(\lambda)$ involves a distributed sum across sensors. Based on the above facts, we introduce below the distributed MASC algorithm.

6.3.2 Algorithmic description

The main steps of distributed MASC are the following. Each sensor computes the nearest neighbors of its observation $x_j^{(u)}$ among the labelled examples. Next, it computes the value of the objective function for each possible class p; this involves first a local and then a distributed computation step. In particular, for each class p, the neighbors of $x_j^{(u)}$ contribute to the calculation of a portion $(r(j))$ of the value of the objective function, which involves only local computation. Next, those portions are averaged distributively, by means of average consensus, so that all observations are taken into account and the total value of the objective function is computed *at all sensors*. The above process is repeated for all possible classes and eventually the sensors reach a consensus classification decision.

In what follows, we discuss in detail the distributed MASC algorithm. For notational ease, we drop the subscript j from $x_j^{(u)}$ when it is clear from the context that we refer to sensor j. The main steps are shown in Algorithm 5. First, each sensor computes the k-NN graph of its own data set $\{X^{(l)} \bigcup x^{(u)}\}$ and forms the corresponding S matrix of size $(l+1) \times (l+1)$ (Lines 4-7). Next, each class hypothesis is tested (loop 8-12). For each class hypothesis p, each sensor j first computes a scalar number $r(j)$ that involves local computation only; namely a weighted sum of the nonzero entries of the last row of S (i.e., $(l+1)$th row). This corresponds to a portion of the value of the objective function, which captures the smoothness of the label assignment under the current class hypothesis.

In order to compute the value of the objective function $q(p)$, the partial sums $r(j)$ need to be summed together and this involves distributed computation. This step can be performed by *distributed average consensus* (Line 11), where the summation of all r's is computed *at each sensor*. Note that this will result in a scaled version of $q(p)$, due to presence of $1/m$ in the average. However, this has no influence on the classification decision, which is taken in Line 13 by all sensors, after all hypotheses have been tested. At the end of the algorithm, all sensors reach a consensus decision.

Figure 6.3 shows schematically the flow of the distributed computation in Line 11 of Algorithm 5 for a single hypothesis p. We show the general structure of the similarity matrix S formed at each sensor j, $j = 1, \dots, m$. Observe that the upper left block of S corresponding

Algorithm 5 The distributed MASC algorithm

```
1: Input to each sensor:
   $l$: number of labelled data.
   $X^{(l)} \in \mathbb{R}^{d \times l}, Y^{(l)}$: labelled examples.
   $x^{(u)} \in \mathbb{R}^{d \times 1}$: unlabelled example (observation).
2: Output at each sensor:
   $\hat{p}$: estimated unknown class.
3: Initialization at each sensor:
4: Form the $k$-NN graph $\mathcal{G}_d$ of the data set $\{X^{(l)} \bigcup x^{(u)}\}$.
5: Compute the weight matrix $H \in \mathbb{R}^{(l+1)\times(l+1)}$ of $\mathcal{G}_d$.
6: Compute the diagonal matrix $D$, where $D_{i,i} = \sum_{j=1}^{l+1} H_{ij}$.
7: Compute $S = D^{-1/2} H D^{-1/2}$.
8: for $p = 1 : c$ do
9:     Each sensor sets $\lambda = [0, \ldots, \underbrace{1}_{p}, \ldots, 0]$.
10:    Each sensor $j$ computes $r(j) = \sum_{i=1}^{l} S_{l+1,i} \|Y_i - \lambda\|^2$.
11:    $q(p) = \sum_{j=1}^{m} r(j)$ :=average_consensus($r$).
12: end for
13: $\hat{p} = \arg\min_p q(p)$
```

to the labelled set is common to all similarity matrices of the sensors as they all have a copy of $X^{(l)}$. The only difference is in their last row, whose non-zero entries correspond to the nearest neighbors of their own observation $x^{(u)}$ among the labelled examples (indicated by stars). Notice that those entries contribute to the computation of the partial sums $r(j)$ in Line 10, which involves only local computation. Then, the sum of all values $r(j),\ j = 1, \ldots, m$ is computed distributively by average consensus, which yields the value of the objective function $q(p)$ for the current class hypothesis p. Interestingly, all observations contribute to the final classification decision, thanks to the employment of average consensus. Note finally that the computation shown in Figure 6.3 is repeated for each class hypothesis p, from $p = 1$ to $p = c$ (see loop 8-12 in Algorithm 5).

6.3.3 Discussion

Notice that each sensor is able to provide an estimate of the unknown class even before the consensus process starts. This is possible by looping over all class hypotheses and using its local r value as a (crude) approximation to the objective function value. In the sequel, while distributed consensus progresses, information from all observations is propagated over the network, the approximations to the objective function are refined and the partial classification decisions are updated. Eventually, the approximations to the objective function values converge and the sensors reach to a consensus decision. The latter may even occur way before the function values stabilize. In what follows, we analyze why this is the case.

Observe that consensus decision is reached when the approximation error of consensus at each sensor becomes smaller than half of the gap between the smallest $q(p_i)$ and second smallest $q(p_j)$ value of the objective function. Denote the gap between them by $\delta = q(p_j) - q(p_i) > 0$ as shown below. The knots on the horizontal axis represent the sorted list (in ascending order) of the objective function values $q(p)$ for $p = 1, \ldots, c$. Therefore, as long as

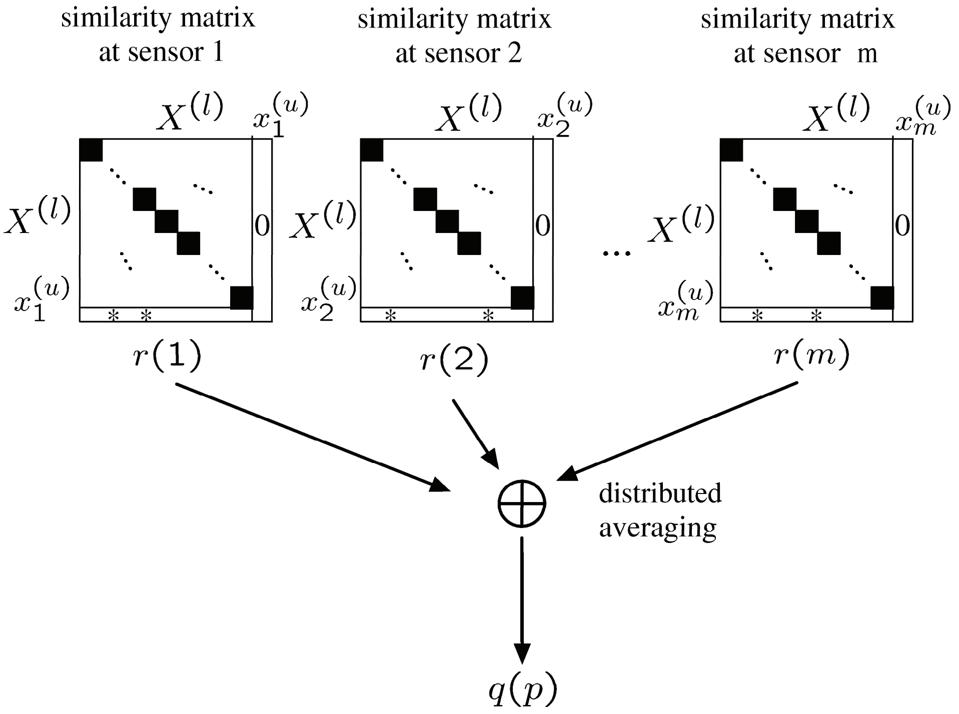
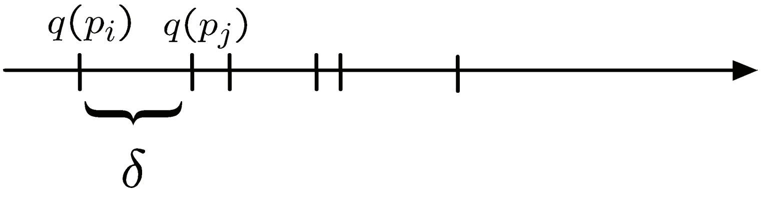


Figure 6.3: *Flow of computation, which is repeated for each hypothesis p, $p = 1, \ldots, c$. The stars in the last row of each similarity matrix correspond to the nearest neighbors of the observation $x^{(u)}$ among the labelled examples.*



the approximation error of consensus at each sensor is smaller than $\delta/2$, the order between the estimates $\tilde{q}(p_i)$ and $\tilde{q}(p_j)$ cannot change, and consensus decision has been reached. From this point on, further consensus iterations will decrease the approximation error, but they will have no influence on the consensus decision.

6.3.4 Cost analysis

In light of the above discussion, it is interesting to study the communication cost of distributed MASC. First, observe that c rounds of average consensus are needed (one for each class). Hence, the communication cost and the convergence properties of the particular consensus algorithm that is employed, drives the communication cost of distributed MASC. It should be clear that a fast consensus algorithm that is able to reduce the approximation error quickly within a few iterations, will have direct impact on reducing the communication cost of distributed MASC, resulting in significant energy savings. This provides the main motivation for studying fast consensus algorithms in Chapter 7.

Regarding the computational cost of distributed MASC, we should recall that it consists of the following main steps:

- The construction of k-NN graph among the labelled examples scales as $\mathrm{O}(l^2)$, where l denotes the number of labelled examples. However, this can be performed off-line (e.g., before the deployment of the sensor network).
- Each sensor j computes the nearest neighbors of $x_j^{(u)}$. This requires the computation of the distance of $x_j^{(u)}$ to all labelled examples and scales as $O(l)$.
- The enumeration of all possible hypotheses (Lines 8-12) scales as $\mathrm{O}(c)$.

If we omit the off-line cost of forming the graph among the labelled examples, we conclude that the total computational cost *per sensor* is $\mathrm{O}(l + c)$. Hence, distributed MASC is of very low complexity that is appropriate for sensor networks.

6.4 Simulation results

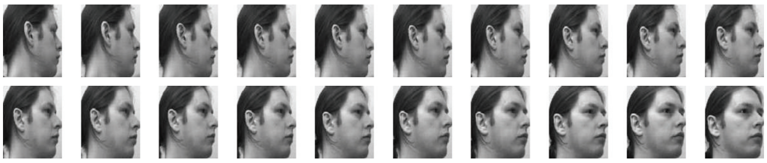


Figure 6.4: *Sample face images from the UMIST database. The number of different poses for each subject is varying.*

We show the feasibility of the distributed MASC algorithm in the context of distributed multi-view face recognition. We consider the case of a vision sensor network, such as the one shown in Fig. 6.1, where the face of a subject is captured by different cameras organized in an ad-hoc network. Each observation in this case represents a facial image captured under different viewing angles. Such a scenario may be of practical interest in airports or high security areas. Observe again that all observations belong to the same class and the problem resides in estimating the unknown class i.e., recognizing the subject. In this context, it is interesting to investigate how our distributed MASC algorithm behaves. Note that our main goal here is not to present a new method for multi-view face recognition, but rather to use this application as a showcase in order to illustrate the behavior and feasibility of our algorithm.

6.4.1 Setup

We use the UMIST database [32] in our simulations. The UMIST database contains 20 people under different poses. The number of different views per subject varies from 19 to 48. We used a cropped version of the UMIST database that is publically available from S. Roweis'

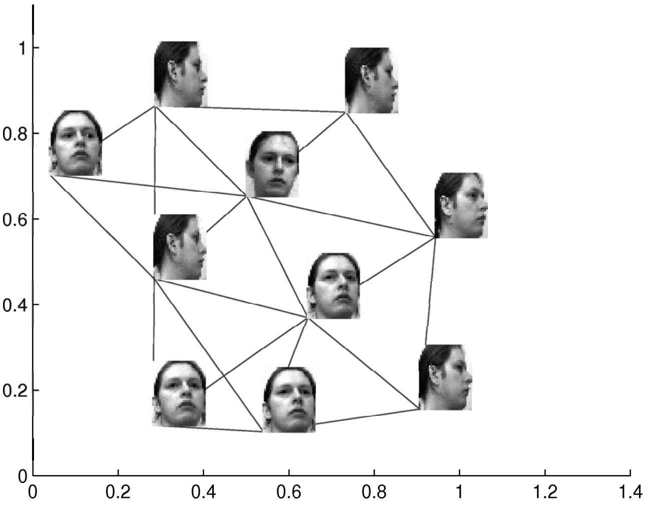


Figure 6.5: *Distributed multi-view face recognition in a vision sensor network. Each facial image corresponds to the observation of a sensor. The problem is to estimate the unknown class in a distributed fashion.*

web page[1]. Figure 6.4 illustrates a sample subject from the UMIST database along with its first 20 views.

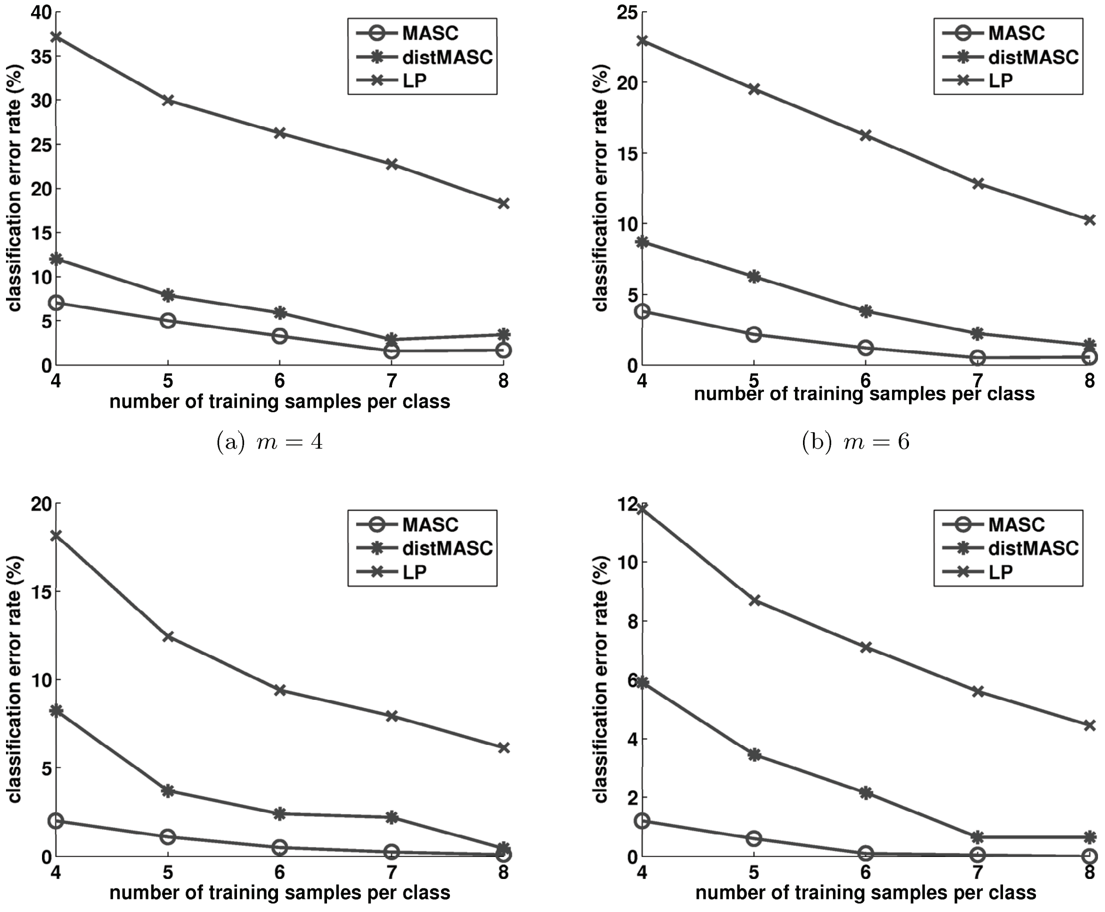
Fig. 6.5 illustrates a snapshot of the simulated network, where the facial image next to each sensor corresponds to its own observation.We use the random geographic graph model [33] for constructing the sensor network. According to this model, we randomly distribute m nodes on a 2-dimensional unit area. Two nodes are adjacent if their Euclidean distance is smaller than a threshold ϵ. More information about this model and the construction of the sensor network will be given in Section 7.5.1.

6.4.2 Distributed vs centralized MASC

In the first experiment we will investigate the classification performance of distributed MASC with respect to that of centralized MASC and centralized label propagation (LP). We assume that the distributed average consensus in Line 11 of Algorithm 5 has converged to the asymptotic solution. In other words, we assume that the distributed summation is exact. The purpose of this experiment is to investigate whether the distributed algorithm suffers any loss in performance and what are the factors that influence this phenomenon. We set the number of nearest neighbors k to 3 in all methods, and $\mu = 0.1$ in label propagation. We investigate the behavior of all methods, when the number of multiple observations m varies from 4 to 10 with step 2. For each particular value of m, we measure the classification error rate for different sizes of training set. In particular, we increase gradually the number of training examples per class and measure the average classification error rate over 100 random experiments (i.e., 100 random splits of the data into training (labelled) and test (unlabelled)

---

[1] `http://www.cs.toronto.edu/~roweis/data.html`



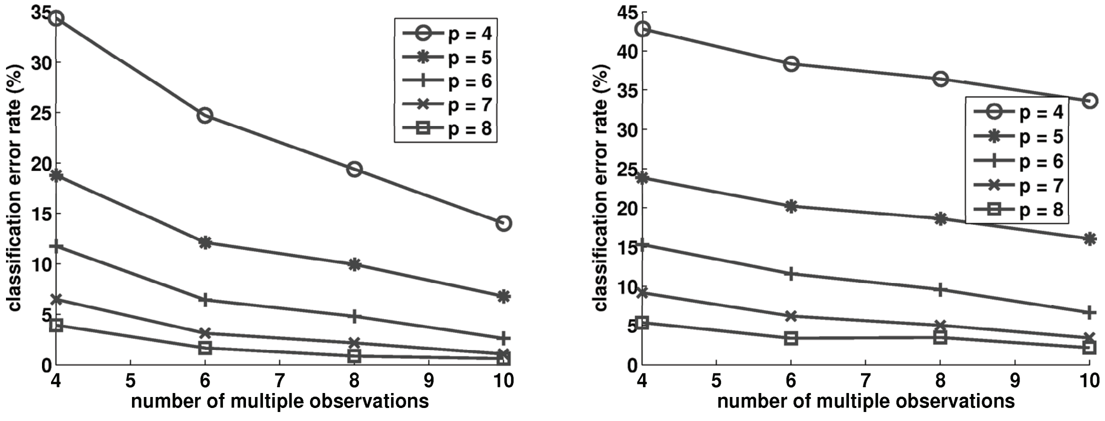
(a) $m = 4$ (b) $m = 6$

(c) $m = 8$ (d) $m = 10$

Figure 6.6: *Difference in performance between MASC and its distributed version versus the number of training samples (per class).*

sets).

Figs 6.6(a)-6.6(d) show the obtained results for different number m of multiple observations, when the number of training examples per class increases from 4 to 8 with step 1. Notice that there is a small loss in performance of distributed MASC with respect to its centralized counterpart. To see why this happens, it is important to observe that the k-NN graph in the distributed case is different than that in the centralized case. This is due to the fact that the multiple observations are collected distributively. Hence, the neighbors of an observation $x^{(u)}$ can only be selected among the labelled examples, whereas in the centralized case they may be selected among all labelled and unlabelled examples. This is the main reason for the difference in performance in Fig. 6.6, which is more pronounced when the training set is small. However, when the training set becomes larger, this phenomenon decays and the difference between MASC and its distributed variant becomes negligent. Notice finally, that even with this small loss in performance, distributed MASC is still superior to (centralized) label propagation.



(a) MASC (b) distMASC

Figure 6.7: *Classification performance versus number of multiple observations, for both methods. Each curve corresponds to different number of training samples per class.*

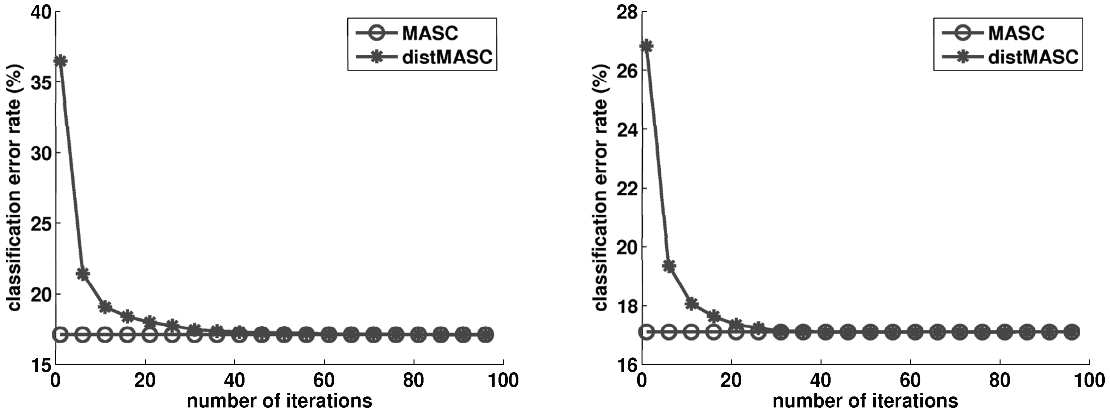
Note however that it is exactly this difference in the construction of the k-NN graph that allows the distributed MASC algorithm have much lower computational cost ($O(l + c)$) than that of centralized MASC ($O((l + m)^2 + c)$). Essentially, this is the main characteristic that makes it efficient and feasible in distributed settings. However, this comes at the cost of a small performance loss, which becomes even smaller when the training set is sufficiently large.

Fig. 6.7 summarizes the same results of Fig. 6.6 in a different way. In particular, it illustrates the behavior of classification performances of both MASC methods with respect to the number of multiple observations, when the size of the training set is fixed. The number of multiple observations m varies from 4 to 10 with step 2. Each curve corresponds to a fixed number of training samples per class, denoted by p. Unsurprisingly, observe that increasing the number of multiple observations, the general tendency in both algorithms is to improve the classification performance. We have seen the same phenomenon in the previous chapter (see Section 5.5).

6.4.3 Effect of consensus on distributed MASC

In the previous experiments, we assumed that the distributed summation in Line 11 of Algorithm 5 is exact. In this experiment we drop this assumption and we investigate the effect of employing distributed consensus for the computation of this sum. Note that our goal in this particular experiment is to study the effect of consensus on the classification performances. For this reason, we use the same k-NN graph of distributed MASC in its centralized counterpart. This way, the performance difference of the two algorithms will only be due to the summation part. First, we split randomly the data set into training and test set, by including two examples per class in the labelled set $X^{(l)}$ and the rest is assigned to the test set. We form $m = 10$ multiple observations, which are drawn randomly from the test set, and we use $k = 1$ in the construction of the k-NN graph.

Fig. 6.8 shows the average classification error rate (over 500 random experiments) measured on a certain sensor, say the first one, when the number of iterations in distributed



(a) Maximum degree (b) Metropolis

Figure 6.8: *Average classification error rate vs consensus iterations.*

consensus varies from 1 to 100 with step 5. We use two different weight matrices from the literature, namely the Metropolis and Maximum-degree matrices [121, 122], which are known to lead iteration $z_{t+1} = Wz_t$ to asymptotic convergence to the average $\mu = \frac{1}{m}\sum_{i=1}^{m} z_0(i)$ (i.e., these matrices satisfy the condition (2.12)). More details about these two matrices will be given later in Section 7.5. Observe that fairly few iterations, namely between 30 and 40, provide sufficient accuracy in the computation of the distributed sum, in order to offer similar performance as the centralized MASC algorithm.

6.5 Related work

We have already mentioned that the problem of distributed classification in ad-hoc sensor networks has just started to receive attention. In Section 2.4.2 we reviewed two very recent methodologies that study distributed SVM training and distributed head pose estimation respectively, in ad-hoc networks using consensus. It is important to note that the general problem of distributed classification has been mostly studied so far in sensor network architectures with a central coordinator node (e.g. fusion center). Typically, the fusion center communicates directly with the other nodes, gathers local information from them and takes the final global classification decision at the end. For the sake of completeness, we mention below a few representative schemes that attack the general problem of distributed classification, although they are quite different to the methods and problem setup studied in this chapter.

The authors in [63] propose a distributed multi-target classification algorithm for sensor networks. The authors formulate the classification problem as a multiple hypothesis testing problem and propose a decision fusion methodology by aggregating local classifier decisions to a fusion center. Since the number of hypothesis grows exponentially with the number of targets, the authors propose a sub-optimal approach of partitioning the hypothesis space, which moreover shows satisfactory performance.

Concerning the distributed binary classification problem, the authors in [77] propose a block coordinate descent algorithm which determines the decision rules at the sensors and at the fusion center. They assume that the joint distribution of the sensor observations is unknown. Their scheme is based on the principles of empirical risk minimization and marginalized kernels.

In another study [75], the authors investigate the problem of learning the aggregation function in distributed classification, where each classifier has its own training data set (i.e., no common labelled training data among classifiers). The learning is transductive, in the sense that the rule is optimized over objective functions measured on batches of test data. They provide experimental evidence showing that the learned rules outperform fixed rules, such as sum or product rules.

Moreover, there are schemes that attack the problem by distributed feature extraction followed by (centralized) classification at the fusion center. For instance, A. Yang et al in a recent article [125] propose a distributed scheme for segmentation and classification of human actions using a network of wearable motion sensors. It is assumed that sensors are able to transmit local feature vectors to a central computer, where the global classification is performed. Human actions are represented using subspace models for each class. Recognition is performed by recovering the sparse representation of a test action sample over the training set using L1 minimization. As a byproduct, the sparsity of the obtained solution can be also employed as a mechanism for outlier detection.

Finally, we should mention that the problem of classification can be generally treated as an inference problem over graphical models. In this case, message passing algorithms (e.g., belief propagation) can be employed for distributed classification. However, combining graphical models and sensor networks is not a trivial task and many issues need to be addressed (see [12] for more details). For instance, it is not straightforward how to assign random variables (nodes of the graphical model) to sensors such that communication cost is minimized (see e.g., [93]). Moreover, inference in arbitrary (loopy) graphs is intractable in general. On the contrary, our methodology is straightforward, as well as computation and communication efficient.

6.6 Conclusions

In this chapter, we studied the problem of classification of multiple observations in the scenario where the observations are collected distributively. We showed that distributed classification in ad-hoc sensor networks can be effectively performed using distributed consensus. In particular, we proposed a distributed version of the MASC algorithm that aggregates information from all observations across the network and results in a consensus classification decision among the sensors. We have illustrated its behavior in the context of distributed multi-view face recognition. The simulation results have shown that (i) MASC and its distributed version have similar performance when the training set size is sufficiently large and (ii) distributed MASC can accommodate slightly inexact estimates of the objective function values, which encourages the use of consensus for its computation. Based on the asymptotic convergence property of distributed consensus, one may wonder whether it is possible to accelerate its convergence process in order to impact the efficiency of consensus-based distributed classification algorithms. In the next chapter, we pursue this question further.

Chapter 7

Fast Convergence in Distributed Consensus

7.1 Introduction

We have seen that average consensus is a valuable tool for distributed classification in ad-hoc sensor networks. It provides a computational platform for aggregating information across the network, such that all observations are progressively taken into account before reaching a consensus decision. However, the sensors are typically very simple devices with limited computational and communication capabilities. Therefore, there is a need for fast distributed consensus techniques that are efficient in terms of communication and computational cost. Such techniques would in turn contribute to the efficiency of distributed classification in sensor networks.

In this chapter, we consider the problem of accelerating the convergence process of distributed consensus by exploiting the memory of the sensors[1]. One important characteristic of distributed consensus algorithms is the rate of convergence to the asymptotic solution. Recall from Section 2.5.1 that each sensor updates its own local estimate of the average by a weighted linear combination of the corresponding estimates of its neighbors. The weights that are represented in a network weight matrix W typically drive the importance of the measurements of the different neighbors. In many cases, the average consensus solution can be reached by linear iterations i.e., successive multiplications of W with the vector of initial sensor values. It has been shown in [121] that in the case of fixed network topology, the convergence rate depends on the second largest (in magnitude) eigenvalue of W, $\lambda_2(W)$. In particular, the convergence is faster when the value of $|\lambda_2(W)|$ is small. Similar convergence results have been proposed recently in the case of dynamic random network topology [50, 51], where the convergence rate is governed by the expected value of $|\lambda_2(W)|$.

The main research direction so far, in accelerating consensus, focuses on the computation of the optimal weights W that yield the fastest convergence rate to the asymptotic solution [121, 122, 123]. In this thesis, we diverge from methods that are only based on successive multiplications of W, and we rather allow the sensors to use their previous estimates, in order to accelerate the convergence rate [58]. This is similar in spirit to the works proposed

---

[1] To appear in: E. Kokiopoulou and P. Frossard, “Polynomial Filtering for Fast Convergence in Distributed Consensus”, IEEE Transactions on Signal Processing.

[54, 104] that reach the consensus solution in a finite number of steps, where extrapolation methods and linear dynamical system formulation are respectively used for fixed network topologies. In order to address more generic network topologies, we propose here to use a matrix polynomial p applied on the weight matrix W in order to shape its spectrum. Given the fact that the convergence rate is driven by $|\lambda_2(W)|$, it is therefore possible to impact on the convergence rate by careful design of the polynomial p. We show that, in the implementation viewpoint, working with $p(W)$ is equivalent to each sensor aggregating its value periodically using its own previous estimates. We further formulate the problem of the computation of the polynomial coefficients for both static and dynamic network topologies. We propose a methodology for the computation of the coefficients based on semi-definite programming (SDP), which results into an optimal solution in the case of static network topologies. In the case of dynamic topologies, we provide an effective sub-optimal solution where the filter coefficients are computed based on the average weight matrix.

The rest of this chapter is organized as follows. In Section 7.2 we review the main convergence results of average consensus in both fixed and dynamic random network topologies. Next, in Section 7.3 we introduce the polynomial filtering methodology and discuss its implementation for distributed consensus problems. We discuss the computation of the polynomial filter coefficients in Section 7.4 for both static and dynamic network topologies. In Section 7.5 we provide simulation results that verify the validity and the effectiveness of our method in both static and dynamic network topologies. Then, in Section 7.6, we show the impact of polynomial filtering on the efficiency of distributed classification. Related work is finally presented in Section 7.7.

7.2 Convergence in distributed consensus averaging

Assume a network of m sensors and that initially each sensor i reports a scalar value $z_0(i) \in \mathbb{R}$. We denote by $z_0 = [z_0(1), \dots, z_0(m)]^\top \in \mathbb{R}^m$ the vector of initial values on the network. Denote by

$$\mu = \frac{1}{m} \sum_{i=1}^{m} z_0(i) \tag{7.1}$$

the average of the initial values of the sensors. Recall from Section 2.5 that the problem of distributed averaging becomes to compute μ *at each sensor* by distributed linear iterations. In what follows, we review the main convergence results for distributed consensus algorithms on both fixed and dynamic network topologies.

7.2.1 Static network topology

The sensor network topology is typically modelled by an undirected graph, which has been denoted by $\mathcal{G}_s$ so far. In this chapter we deal only with graphs corresponding to sensor networks; hence, we drop the subscript s for notational ease. In what follows, the graph will be simply denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V} = \{1, \dots, m\}$ corresponding to sensors. An edge $(i, j) \in \mathcal{E}$ is drawn if and only if sensor i can communicate with sensor j, as illustrated in Figure 7.1. We denote the set of neighbors for node i as $\mathcal{N}_i = \{j | \, (i, j) \in \mathcal{E}\}$. Unless otherwise stated, we assume that each graph is simple i.e., no loops or multiple edges are allowed.

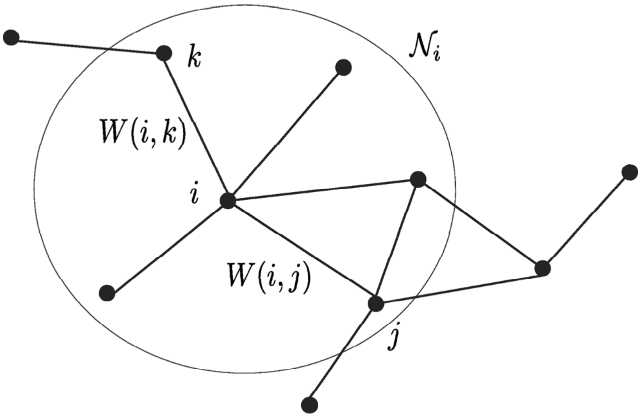


Figure 7.1: *Static network topology. The parameter $W(i,j) \neq 0$ describes the weight of the edge that permits the sensor i to communicate with its neighbor sensor j.*

We consider distributed linear iterations of the following form

$$z_{t+1}(i) = W(i,i)z_t(i) + \sum_{j \in \mathcal{N}_i} W(i,j)z_t(j), \tag{7.2}$$

for $i = 1, \dots, m$, where $z_t(j)$ represents the value computed by sensor j at iteration t. Since the sensors communicate in each iteration t, we assume that they are synchronized. The parameters $W(i,j)$ denote the edge weights of $\mathcal{G}$. Since each sensor communicates only with its direct neighbors, $W(i,j) = 0$ when $(i,j) \notin \mathcal{E}$. The above iteration can be compactly written in the following form

$$z_{t+1} = W z_t, \tag{7.3}$$

or more generally

$$z_t = \Big(\prod_{i=0}^{t-1} W\Big) z_0 = W^t z_0. \tag{7.4}$$

We call the matrix W that gathers the edge weights $W(i,j)$, as the weight matrix. Note that W is a sparse matrix whose sparsity pattern is driven by the network topology. We assume that W is symmetric, and we denote its eigenvalue decomposition as $W = Q\Lambda Q^\top$. We also denote by $\lambda_2(W)$ the second largest (in magnitude) eigenvalue of W.

We have also seen that the distributed linear iterations given in eq. (7.3) converge to the average for every z_0 if and only if

$$\lim_{t \to \infty} W^t = \frac{\mathbf{1}\mathbf{1}^\top}{m}, \tag{7.5}$$

where $\mathbf{1}$ is the vector of ones [121].

It has been shown that for fixed network topology the convergence rate of eq. (7.3) depends on the magnitude of the second largest eigenvalue $|\lambda_2(W)|$ [121]. The asymptotic convergence factor is defined as

$$r_{\text{asym}}(W) = \sup_{z_0 \neq \mu\mathbf{1}} \lim_{t \to \infty} \Big(\frac{\|z_t - \mu\mathbf{1}\|_2}{\|z_0 - \mu\mathbf{1}\|_2}\Big)^{1/t}, \tag{7.6}$$

and the per-step convergence factor is written as

$$r_{\text{step}}(W) = \sup_{z_0 \neq \mu\mathbf{1}} \frac{\|z_{t+1} - \mu\mathbf{1}\|_2}{\|z_t - \mu\mathbf{1}\|_2}. \tag{7.7}$$

Furthermore, it has been shown that the convergence rate relates to the spectrum of W, as given by the following theorem [121].

Theorem 7.2.1. *The condition in (7.5) holds if and only if*

$$\begin{aligned} \mathbf{1}^\top W &= \mathbf{1}^\top, && (7.8)\\ W\mathbf{1} &= \mathbf{1}, && (7.9)\\ \rho\big(W - \frac{\mathbf{1}\mathbf{1}^\top}{m}\big) &< 1, && (7.10) \end{aligned}$$

where $\rho(\cdot)$ denotes the spectral radius of a matrix. Furthermore,

$$r_{asym}(W) = \rho\big(W - \frac{\mathbf{1}\mathbf{1}^\top}{m}\big), \tag{7.11}$$

$$r_{step}(W) = \|W - \frac{\mathbf{1}\mathbf{1}^\top}{m}\|_2. \tag{7.12}$$

According to the above theorem, $\frac{1}{\sqrt{m}}\mathbf{1}$ is a left and right eigenvector of W associated with the eigenvalue one, and the magnitude of all other eigenvalues is strictly less than one. Note finally that since W is symmetric, the asymptotic convergence factor coincides with the per-step convergence factor, which implies that the relations (7.11) and (7.12) are equivalent.

We propose now an alternate proof of the above theorem that illustrates the importance of the second largest eigenvalue in the convergence rate. We expand the initial state vector z_0 to the orthogonal eigenbasis Q of W; that is,

$$z_0 = \nu_1 \frac{\mathbf{1}}{\sqrt{m}} + \sum_{j=2}^{m} \nu_j q_j,$$

where $\nu_1 = \langle \frac{\mathbf{1}}{\sqrt{m}}, z_0 \rangle$ and $\nu_j = \langle q_j, z_0 \rangle$. We further assume that $\nu_1 \neq 0$. Then, eq. (7.4) implies that

$$\begin{aligned} z_t &= W^t z_0 = W^t\big(\frac{\nu_1}{\sqrt{m}}\mathbf{1} + \sum_{j=2}^{m} \nu_j q_j\big)\\ &= \frac{\nu_1}{\sqrt{m}}\mathbf{1} + \sum_{j=2}^{m} \nu_j W^t q_j\\ &= \frac{\nu_1}{\sqrt{m}}\mathbf{1} + \sum_{j=2}^{m} \nu_j \lambda_j^t q_j. \end{aligned}$$

Observe now that if $|\lambda_j| < 1$, $\forall j \geq 2$, then in the limit, the second term in the above equation decays and

$$z^* = \lim_{t \to \infty} z_t = \frac{\nu_1}{\sqrt{m}}\mathbf{1} = \mu\mathbf{1}.$$

We see that the smaller the value of $|\lambda_2(W)|$, the faster the convergence rate. Analogous convergence results hold in the case of dynamic network topologies that are discussed next.

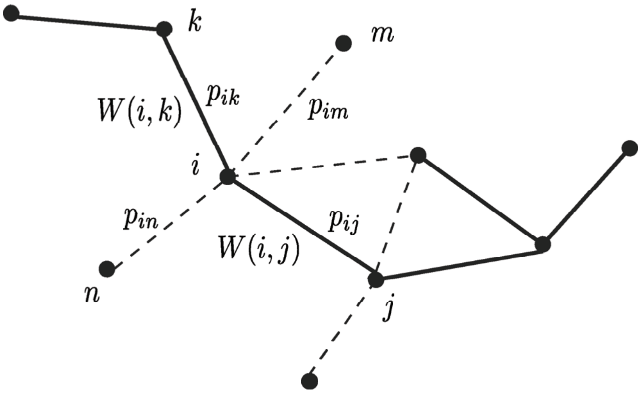


Figure 7.2: *Dynamic network topology. Each link is active with a probability p_{ij}. When a link is active, the parameter $W(i,j)$ describes the weight of the edge that lets the sensors i and j to communicate.*

7.2.2 Dynamic network topology

Let us consider now networks with random link failures, where the state of a link changes over the iterations (see Figure 7.2). In particular, we use the random network model proposed in [50, 51]. We assume that the network at any arbitrary iteration t is $\mathcal{G}_t = (\mathcal{V}, \mathcal{E}_t)$, where $\mathcal{E}_t$ denotes the edge set at iteration t, or equivalently at time instant t. Since the network is dynamic, the edge set changes over the iterations, as links fail at random. We assume that $\mathcal{E}_t \subseteq \mathcal{E}^*$, where $\mathcal{E}^* \subseteq \mathcal{V} \times \mathcal{V}$ is the set of realizable edges when there is no link failure.

We also assume that each link (i,j) fails with a probability $(1-p_{ij})$, independently of the other links. Two random edge sets $\mathcal{E}_{t_1}$ and $\mathcal{E}_{t_2}$ at different iterations t_1 and t_2 are independent. The probability of forming a particular $\mathcal{E}_t$ is thus given by $\prod_{(i,j)\in\mathcal{E}_t} p_{ij}$. We define the matrix P as

$$P_{ij} = \begin{cases} p_{ij} & \text{if } (i,j) \in \mathcal{E}^* \text{ and } i \neq j, \\ 0 & \text{otherwise.} \end{cases} \tag{7.13}$$

It represents the probabilities of edge formation in the network, and the edge set $\mathcal{E}_t$ is therefore a random subset of $\mathcal{E}^*$ driven by the P matrix. The matrix P is symmetric and its diagonal elements are zero, since it corresponds to a simple graph. The weight matrix W becomes dependent on the edge set since only the weights of existing edges can take non zero values. Note finally that one may further introduce a probability q of a network topology change in each iteration. In this case, q allows for controlling the dynamicity of the network (with respect to the synchronous sensor updates). The network then follows the above random network model only when a change is triggered.

In the dynamic case, the distributed linear iteration of eq. (7.2) becomes

$$z_{t+1}(i) = W_t(i,i)z_t(i) + \sum_{j\in\mathcal{N}_i} W_t(i,j)z_t(j) \tag{7.14}$$

or in compact form,

$$z_{t+1} = W_t z_t, \tag{7.15}$$

where W_t denotes the weight matrix corresponding to the graph realization $\mathcal{G}_t$ of iteration t and $W_t(i,j)$ is its corresponding weight of entry (i,j). The iterative relation given by eq.

(7.15) can be written as

$$z_t = \Big(\prod_{s=0}^{t-1} W_s\Big) z_0.$$

Clearly, z_t now represents a stochastic process since the edges are drawn randomly. In what follows, when it is clear from the context that we refer to the random matrix W, we drop the subscript t for notational ease. The convergence rate to the consensus solution therefore depends on the behavior of the product $\prod_{s=0}^{t-1} W_s$. We say that the algorithm converges if

$$\forall z_0 \in \mathbb{R}^m, \ \lim_{t \to \infty} E\|z_t - \mu \mathbf{1}\| = 0. \tag{7.16}$$

We review now some convergence results from [51], which first shows that

Lemma 7.2.1. *For any* $z_0 \in \mathbb{R}^m$,

$$\|z_{t+1} - \mu \boldsymbol{1}\| \leq \Big(\prod_{s=0}^{t} \rho\big(W_s - \frac{\boldsymbol{1}\boldsymbol{1}^\top}{m}\big)\Big) \|z_0 - \mu \boldsymbol{1}\|.$$

Taking the expectation of both sides and using the fact that the W_t's are i.i.d, results in

$$E\|z_{t+1} - \mu \mathbf{1}\| \leq \Big(E\Big[\rho\big(W - \frac{\mathbf{1}\mathbf{1}^\top}{m}\big)\Big]\Big)^t \|z_0 - \mu \mathbf{1}\|.$$

It leads to the following convergence theorem [51] for dynamic networks.

Theorem 7.2.2. *If* $E[\rho(W - \frac{\boldsymbol{1}\boldsymbol{1}^\top}{m})] < 1$*, the vector sequence* $\{z_t\}_{t=0}^{\infty}$ *converges in the sense of eq. (7.16).*

We define the convergence factor in dynamic network topologies as

$$r_d(W) = E\Big[\rho\big(W - \frac{\mathbf{1}\mathbf{1}^\top}{m}\big)\Big].$$

This factor depends, in general, on the spectral properties of the induced network matrix and drives the convergence rate of eq. (7.15). More generally, the authors in [106] show that $|\lambda_2(E[W])| < 1$ is a necessary and sufficient condition for asymptotic (almost sure) convergence of the consensus algorithm in the case of random networks where both network topology and weights are random (more precisely i.i.d and independent over time).

Finally, it is interesting to note that the consensus problem in a random network relates to gossip algorithms. Distributed averaging under the synchronous gossip constraint implies that multiple node pairs may communicate simultaneously only if these node pairs are disjoint. In other words, the set of links implied by the active node pairs forms a matching of the graph. Therefore, the distributed averaging problem described above is closely related to the distributed synchronous algorithm under the gossip constraint that has been proposed in [90, Sec. 3.3.2]. It has been shown in this case that the averaging time (or convergence rate) of a gossip algorithm depends on the second largest eigenvalue of a doubly stochastic network matrix.

7.3 Accelerated consensus with polynomial filtering

7.3.1 Exploiting memory

As we have seen above, the convergence rate of the distributed consensus algorithms depends, in general, on the spectral properties of an induced network matrix. This is the case for both fixed and dynamic network topologies. Motivated by this fact, we exploit the memory of sensors or the values of previous estimates, in order to augment to convergence rate, since memory and computation use is cheaper than communication costs.

We have proposed [54] the Scalar Epsilon Algorithm (SEA) for accelerating the convergence rate to the consensus solution. SEA belongs to the family of extrapolation methods for accelerating vector sequences, such as eq. (7.3). These methods exploit the fact that the fixed point of the sequence belongs to the subspace spanned by any $\ell + 1$ consecutive terms of it, where ℓ is the degree of the minimal polynomial of the sequence generator matrix (for more details, see [54] and references therein). SEA is a low complexity algorithm, which is ideal for sensor networks and it is known to reach the consensus solution in (at most) 2ℓ steps. However, ℓ is unknown in practice, so one may use all the available terms of the vector sequence. Hence, the memory requirements of SEA are $O(T)$, where T is the number of terms. Moreover, SEA assumes that the sequence generator matrix (e.g., W in the case of eq. (7.3)) is fixed, so that it does not adapt easily to dynamic network topologies.

In this chapter, we propose a more flexible algorithm based on the polynomial filtering technique. Polynomial filtering permits to “shape” the spectrum of a certain symmetric weight matrix, in order to accelerate the convergence to the consensus solution. Similarly to SEA, it allows the sensors to use their previous estimates. However, the polynomial filtering methodology introduced below presents three main advantages: (i) it is robust to dynamic topologies (ii) it has explicit control on the convergence rate and (iii) its memory requirements can be adjusted to the memory constraints imposed by the sensor.

7.3.2 Polynomial filtering

Starting from a given (possibly optimal) weight matrix W, we propose the application of a polynomial filter on the spectrum of W in order to impact the magnitude of $\lambda_2(W)$, which mainly drives the convergence rate. Denote by $p_k(\lambda)$ the polynomial filter of degree k that is applied on the spectrum of W,

$$p_k(\lambda) = \sum_{l=0}^{k} \alpha_l \lambda^l = \alpha_0 + \alpha_1 \lambda + \alpha_2 \lambda^2 + \ldots + \alpha_k \lambda^k. \tag{7.17}$$

Accordingly, the matrix polynomial is given as

$$p_k(W) = \sum_{l=0}^{k} \alpha_l W^l = \alpha_0 I + \alpha_1 W + \ldots + \alpha_k W^k. \tag{7.18}$$

Observe now that

$$\begin{aligned} p_k(W) &= p_k(Q \Lambda Q^\top) \\ &= \alpha_0 I + \alpha_1 (Q \Lambda Q^\top) + \ldots + \alpha_k (Q \Lambda^k Q^\top) \\ &= Q p_k(\Lambda) Q^\top, \end{aligned} \tag{7.19}$$

Algorithm 6 Polynomial Filtered Distributed Consensus

```
Input: polynomial coefficients $\alpha_0, \ldots, \alpha_{k+1}$, tolerance $\epsilon$.
Output: average estimate $\bar{\mu}$.
Initialization:
  $\bar{\mu}_0 = z_0(i)$.
  Set the iteration index $t = 1$.
repeat
  if mod$(t, k+1)$ ==0 then
    $z_t(i) = \alpha_0 z_{t-k-1}(i) + \alpha_1 z_{t-k}(i) + \alpha_2 z_{t-k+1}(i) + \ldots + \alpha_k z_{t-1}(i)$ {polynomial filter update}
    $z_t(i) = W(i,i) z_t(i) + \sum_{j \in \mathcal{N}_i} W(i,j) z_t(j)$.
  else
    $z_t(i) = W(i,i) z_{t-1}(i) + \sum_{j \in \mathcal{N}_i} W(i,j) z_{t-1}(j)$.
  end if
  Increase the iteration index $t = t + 1$.
  $\bar{\mu}_t = z_t(i)$.
until $\bar{\mu}_t - \bar{\mu}_{t-1} < \epsilon$
```

which implies that the eigenvalues of $p_k(W)$ are simply the polynomial filtered eigenvalues of W, i.e., $p_k(\lambda_i(W)),\ i = 1, \ldots, m$.

In the implementation level, working on $p_k(W)$ implies a periodic update of the current sensor's value with a linear combination of its previous values. To see why this is true, we observe that:

$$\begin{aligned} z_{t+k+1} &= p_k(W) z_t && (7.20) \\ &= \alpha_0 z_t + \alpha_1 W z_t + \ldots + \alpha_k W^k z_t \\ &= \alpha_0 z_t + \alpha_1 z_{t+1} + \ldots + \alpha_k z_{t+k}. && (7.21) \end{aligned}$$

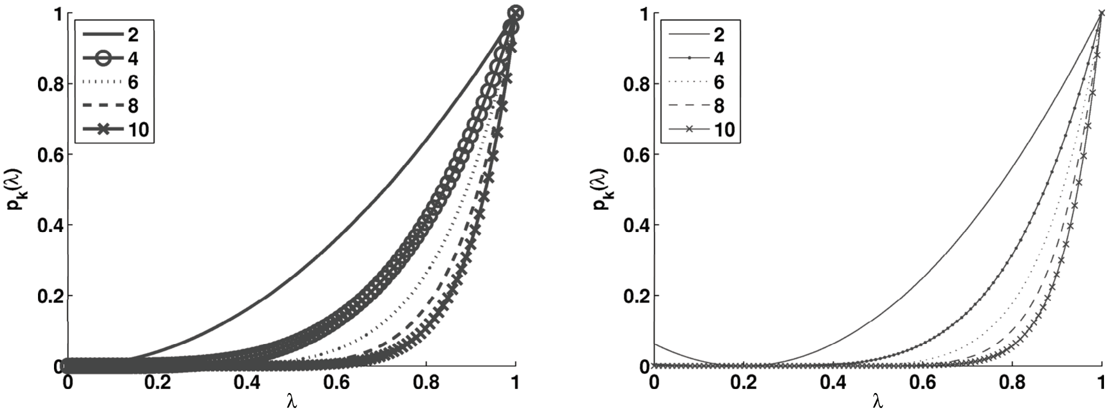
A careful design of p_k can impact the convergence rate dramatically. Then, each sensor typically applies polynomial filtering for distributed consensus by following the main steps that are tabulated in Algorithm 6. In what follows, we propose different approaches for computing the coefficients α_l of the filter p_k.

7.3.3 Newton's interpolating polynomial

One simple approach for the design of the polynomial $p_k(\lambda)$ is to use Hermite interpolation [94]. Recall that the objective is to dampen the smallest eigenvalues of W, while keeping the eigenvalue one intact. Therefore, we assume (initially) that the spectrum of W lies in an interval $[a, 1]$ and we impose smoothness constraints of p_k at the left endpoint a. In particular, the polynomial $p_k(\lambda) : [a, 1] \to \mathbb{R}$ that we seek, will be determined by the following constraints:

$$\begin{aligned} p_k(a) &= 0, && (7.22) \\ p_k(1) &= 1, && (7.23) \\ p_k^{(i)}(a) &= 0,\ i = 1, \ldots, k-1 && (7.24) \end{aligned}$$

where $p_k^{(i)}(a)$ denotes the ith derivative of $p_k(\lambda)$ evaluated at a. Intuitively, we seek a polynomial $p_k(\lambda)$ that is zero at $\lambda = a$ (eq. (7.22)), worth one at $\lambda = 1$ (eq. (7.23)) and that is



(a) $a = 0$ (b) $a = 0.2$

Figure 7.3: *Newton's polynomial* $p_k(\lambda)$ *of various degrees* k*.*

very small in the region close to the left endpoint a. The latter is imposed by the smoothness constraints (7.24). The dampening is achieved by imposing smoothness constraints of the polynomial on the left endpoint of the interval. The computed polynomial will have a degree equal to k. Finally, the coefficients of p_k that satisfy the above constraints can be computed by Newton's divided differences [94].

Figure 7.3(a) shows an example of the shape of $p_k(\lambda)$ for $a = 0$ and different values of the degree k. As k increases, more smoothness constraints are imposed on a and the dampening of the small eigenvalues becomes more effective. Interestingly, notice that since the smoothness constraints hold for free on the left of the interval $[a, 1]$ as well, the filtering will work even in the case where a lies within the spectrum of W, provided that the magnitude of the filtered eigenvalues is strictly smaller than one (i.e., $|p_k(\lambda_j)| < 1$, $j \neq 1$). Therefore, we may drop the assumption that a encloses the spectrum of W. For instance, if the spectrum of W lies in $[0, 1]$ then one may choose $a = 0.2$. Fig. 7.3(b) shows the shape of the resulting polynomial filter in this case.

Finally, it is worth mentioning that the design of Newton's polynomial does not depend on the network topology or the network size. What is only needed is a left endpoint a that roughly corresponds to the left extreme of the spectrum of W, as well as the desired degree k, which is related to memory constraints. This feature of Newton's polynomial is very interesting and it is particularly appealing in the case of dynamic network topologies, as we will show later in the simulations section. Note however that the above polynomial design is mostly driven by intuitive arguments, which tend to obtain small eigenvalues for faster convergence. In the following section, we provide an alternative technique for computing the polynomial filter that optimizes the convergence rate.

7.4 Polynomial filter design with semi-definite programming

7.4.1 Polynomial filtering for static network topologies

Given weight matrix W and a certain degree k, we are now interested in finding the polynomial that leads to the fastest convergence of Eq. (7.20) that we reproduce for convenience here:

$$z_{t+k+1} = p_k(W) z_t.$$

Recall that $p_k(W) = \sum_{l=0}^{k} \alpha_l W^l$. Applying Theorem 7.2.1 to the above linear iteration, the optimal polynomial is the one that minimizes the spectral radius $\rho\big(p_k(W) - \frac{\mathbf{1}\mathbf{1}^\top}{m}\big)$. Therefore, we need to solve an optimization problem where the optimization variables are the $k+1$ polynomial coefficients $\alpha = [\alpha_0, \ldots, \alpha_k]^\top \in \mathbb{R}^{k+1}$ and the objective function is the spectral radius of $p_k(W) - \frac{\mathbf{1}\mathbf{1}^\top}{m}$.

Optimization problem: **OPT1**
$\min_{\alpha \in \mathbb{R}^{k+1}} \ \rho\big(\sum_{l=0}^{k} \alpha_l W^l - \frac{\mathbf{1}\mathbf{1}^\top}{m}\big)$
subject to
$\big(\sum_{l=0}^{k} \alpha_l W^l\big)\mathbf{1} = \mathbf{1}.$

Interestingly, the optimization problem OPT1 is convex. First, its objective function is convex, as stated in Lemma 7.4.1 below.

Lemma 7.4.1. *For a given symmetric weight matrix W and degree k, $\rho\big(\sum_{l=0}^{k} \alpha_l W^l - \frac{\mathbf{1}\mathbf{1}^\top}{m}\big)$ is a convex function of the polynomial coefficients α_l's.*

Proof. Let $\beta, \gamma \in \mathbb{R}^{k+1}$ and $0 \le \theta \le 1$. Since W is symmetric, $\sum_{l=0}^{k} \alpha_l W^l$ is also symmetric. Hence, the spectral radius is equal to the matrix 2-norm. Thus, we have

$$\begin{aligned}
&\rho\Big(\sum_{l=0}^{k}(\theta\beta_l + (1-\theta)\gamma_l)W^l - \frac{\mathbf{1}\mathbf{1}^\top}{m}\Big) \\
= \ & \Big\| \sum_{l=0}^{k}(\theta\beta_l + (1-\theta)\gamma_l)W^l - \frac{\mathbf{1}\mathbf{1}^\top}{m} \Big\|_2 \\
= \ & \Big\| \theta\Big(\sum_{l=0}^{k}\beta_l W^l - \frac{\mathbf{1}\mathbf{1}^\top}{m}\Big) + (1-\theta)\Big(\sum_{l=0}^{k}\gamma_l W^l - \frac{\mathbf{1}\mathbf{1}^\top}{m}\Big) \Big\|_2 \\
\le \ & \theta \Big\| \sum_{l=0}^{k}\beta_l W^l - \frac{\mathbf{1}\mathbf{1}^\top}{m} \Big\|_2 + (1-\theta) \Big\| \sum_{l=0}^{k}\gamma_l W^l - \frac{\mathbf{1}\mathbf{1}^\top}{m} \Big\|_2 \\
\le \ & \theta\rho\Big(\sum_{l=0}^{k}\beta_l W^l - \frac{\mathbf{1}\mathbf{1}^\top}{m}\Big) + (1-\theta)\rho\Big(\sum_{l=0}^{k}\gamma_l W^l - \frac{\mathbf{1}\mathbf{1}^\top}{m}\Big),
\end{aligned}$$

which proves the Lemma. □

In addition, the constraint of OPT1 is linear. This implies that the set of feasible α_l's is convex. As OPT1 minimizes a convex function over a convex set, the optimization problem is finally convex.

In order to solve OPT1, we use an auxiliary variable η to bound the objective function, and then we express the spectral radius constraint as a linear matrix inequality (LMI). Thus, we need to solve the following optimization problem.

Optimization problem: **OPT2**
$\min_{\eta\in\mathbb{R},\ \alpha\in\mathbb{R}^{k+1}}\ \eta$
subject to
$-\eta I \preceq \sum_{l=0}^{k}\alpha_l W^l - \frac{\mathbf{1}\mathbf{1}^\top}{m} \preceq \eta I,$
$\big(\sum_{l=0}^{k}\alpha_l W^l\big)\mathbf{1} = \mathbf{1}.$

Recall that since W is symmetric, $p_k(W) = \sum_{l=0}^{k}\alpha_l W^l$ will be symmetric as well. Hence, the constraint $p_k(W)\mathbf{1} = \mathbf{1}$ is sufficient to ensure that $\mathbf{1}$ will be also a left eigenvector of $p_k(W)$. The spectral radius constraint,

$$-\eta I \preceq p_k(W) - \frac{\mathbf{1}\mathbf{1}^\top}{m} \preceq \eta I$$

ensures that all the eigenvalues of $p_k(W)$ other than the first one, are less or equal to η. Due to the LMI, the above optimization problem becomes equivalent to a semi-definite program (SDP) [10]. SDPs are convex problems and can be globally and efficiently solved. The solution to OPT2 is therefore computed efficiently in practice, where the SDP only has a moderate number of $k+2$ unknowns (including η).

7.4.2 Polynomial filtering for dynamic network topologies

We extend now the idea of polynomial filtering to dynamic network topologies. Theorem 7.2.2 suggests that the convergence rate in the random network topology case is governed by $E\Big[\rho\big(W - \frac{\mathbf{1}\mathbf{1}^\top}{m}\big)\Big]$. Since W depends on a dynamic edge set, $p_k(W)$ now becomes stochastic. Following the same intuition as above, we could form an optimization problem, similar to OPT1, whose objective function would be $E[\rho\big(\sum_{l=0}^{k}\alpha_l W^l - \frac{\mathbf{1}\mathbf{1}^\top}{m}\big)]$. Although this objective function can be shown to be convex, its evaluation is hard and typically requires several Monte Carlo simulations steps.

For this reason, we use the following approximation. Let $\overline{W}$ denote the average weight matrix $E[W]$. We then observe that

$$E\Big[\rho\big(W - \frac{\mathbf{1}\mathbf{1}^\top}{m}\big)\Big] \geq \rho\Big(\overline{W} - \frac{\mathbf{1}\mathbf{1}^\top}{m}\Big), \tag{7.25}$$

which is due to Lemma 7.4.1 and Jensen's inequality. The above inequality implies that in order to have a small $E\Big[\rho\big(W - \frac{\mathbf{1}\mathbf{1}^\top}{m}\big)\Big]$, it is required that $\rho\Big(\overline{W} - \frac{\mathbf{1}\mathbf{1}^\top}{m}\Big)$ is small. Additionally, the authors provide experimental evidence in [51], which indicates that $\rho\Big(\overline{W} - \frac{\mathbf{1}\mathbf{1}^\top}{m}\Big)$ is closely related to the convergence rate of the linear iteration $z_{t+1} = W_t z_t$.

Based on the above approximation, we propose an algorithm for polynomial filtering applied to dynamic network topologies. In particular, we propose to build our polynomial filter based on $\overline{W}$. Hence, we formulate the following optimization problem for computing the polynomial coefficients α_l's in the dynamic network topology case.

Optimization problem: **OPT3**
$\min_{\eta\in\mathbb{R},\ \alpha\in\mathbb{R}^{k+1}} \ \eta$
subject to
$-\eta I \preceq \sum_{l=0}^{k} \alpha_l \overline{W}^l - \frac{\mathbf{1}\mathbf{1}^\top}{m} \preceq \eta I,$
$\big(\sum_{l=0}^{k} \alpha_l \overline{W}^l\big)\mathbf{1} = \mathbf{1}.$

The new optimization problem OPT3 could be viewed as the analog of OPT2 for the case of dynamic network topologies. The main difference is that we work on $\overline{W}$, whose eigenvalues can be easily obtained. Once $\overline{W}$ has been computed, this optimization problem is solved efficiently by a SDP, similarly to the case of static networks.

Finally, one would expect that the bound (7.25) will be tighter when the successive matrices W_t are not very different. Indeed, in this case, one may argue that for infinitesimal changes of W, the function $\rho(W - \frac{\mathbf{1}\mathbf{1}^\top}{m})$ is locally linear and therefore the bound in Eq. (7.25) is tight. Note also that these network changing conditions correspond to quite common scenarios. Furthermore, since $\overline{W}$ is symmetric, its eigenvalues are well behaved according to the Bauer-Fike theorem [29], which states that small perturbations of symmetric matrices cannot not result in large perturbations of their spectrum. Based on this fact, one would expect that small changes in $\overline{W}$ will lead to small changes to the optimal polynomial coefficients.

7.5 Simulation results

7.5.1 Setup

In this section, we provide simulation results that show the effectiveness of the polynomial filtering methodology in distributed averaging for both static and dynamic network topologies. We also show its impact in the efficiency of distributed classification that was presented in Chapter 6. First we introduce a few weight matrices for sensor networks that have been extensively used in the distributed averaging literature. Suppose that $d(i)$ denotes the degree of the ith sensor. It has been shown in [121, 122] that iterating eq. (7.3) with the following matrices leads to convergence to $\mu\mathbf{1}$.

- *Maximum-degree* weights. The Maximum-degree weight matrix is

$$W(i,j) = \begin{cases} \frac{1}{m} & \text{if } (i,j) \in \mathcal{E}, \\ 1 - \frac{d(i)}{m} & i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{7.26}$$

- *Metropolis* weights. The Metropolis weight matrix is

$$W(i,j) = \begin{cases} \frac{1}{1+\max\{d(i),d(j)\}} & \text{if } (i,j) \in \mathcal{E}, \\ 1 - \sum_{(i,k)\in\mathcal{E}} W(i,k) & i = j, \\ 0 & \text{otherwise.} \end{cases} \tag{7.27}$$

- *Laplacian* weights [121]. Suppose that A is the adjacency matrix of $\mathcal{G}$ and D is a diagonal matrix that holds the vertex degrees. The Laplacian matrix is defined as $L = D - A$ and the Laplacian weight matrix is defined as

$$W = I - \gamma L, \tag{7.28}$$

Method	Description	Input parameters
Iter	Liner iteration: see eq. (7.3)	W
SDP-PF	SDP polynomial filtering	k, W (or $\overline{W}$)
Newton-PF	Newton's polynomial filtering	k, a
SEA	Scalar Epsilon Algorithm [54]	$\{z_i\}_{i=0}^{t}$

Table 7.1: *Summary of methods and their input parameters.*

where the scalar γ must satisfy $0 < \gamma < 1/d_{\max}$.

- *Optimal* weights, introduced in [121]. The optimal weight matrix W is determined by solving an SDP corresponding to the minimization of $\rho(W - \frac{\mathbf{1}\mathbf{1}^\top}{m})$ with respect to the nonzero entries of W, under the constraint that W is graph conformant (see [121] for more details). It should be noted that the number of unknowns in this case is equal to the number of nonzero entries of W (i.e., number of links in the network, which is in the order of $\mathrm{O}(m^2)$). This represents a limitation on the use of such optimal weights for large or dynamic network topologies.

In addition, we will analyze the performance of polynomial filtering defined in this chapter and compare to state of the art weight matrices. Recall that the number of unknowns in our method is small, namely $k+2$, which is an important advantage with respect to optimal weights. Note in passing that in our SDP polynomial filtering method, in both fixed and dynamic network topology cases, the α_l's are computed off-line assuming that W and respectively $E[W]$ are known a priori. Finally, we should mention that the SDP programs are solved in MATLAB using the SeDuMi [103] solver[2] and the YALMIP toolbox [67].

The sensor networks are built using the random geographic graph model [33]. In particular, we uniformly distribute m nodes on the 2-dimensional unit area. Two nodes are considered to be adjacent if their Euclidean distance is smaller than $r = \sqrt{\frac{\log m}{m}}$ in order to guarantee connectedness of the graph with high probability [33].

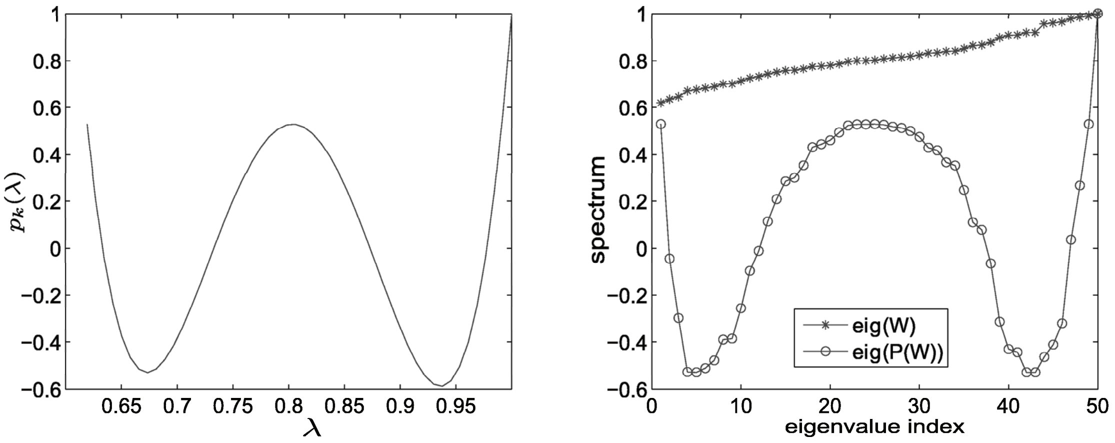
In what follows, we provide performance analysis of the different consensus methods, first in static and then in dynamic network topologies. For the sake of completeness, we also provide the results of the scalar epsilon algorithm (SEA) [54] that uses all previous estimates of each sensor. Before we delve into the performance analysis, we provide in Table 7.1 a summary of all methods that are tested, along with their description and their input parameters.

7.5.2 Static network topologies

Illustrative example First, we illustrate graphically the effect of polynomial filtering on the spectrum of W. We build a network of $m = 50$ sensors and we solve the optimization problem OPT2 with the Maximum-degree weight matrix W, defined in (7.26), and with $k = 4$. Figure 7.4(a) shows the obtained polynomial filter $p_k(\lambda)$, when $\lambda \in [\lambda_{\min}(W), 1]$, where $\lambda_{\min}(W)$ denotes the smallest (algebraically) eigenvalue of W. Next, we apply the polynomial on W and Figure 7.4(b) shows the spectrum of W before (star-solid line) and after (circle-solid line) polynomial filtering, versus the vector index. Observe that polynomial

---

[2] Publically available at: `http://sedumi.mcmaster.ca/`



(a) SDP polynomial filter $p_k(\lambda)$, $\lambda_{\min} \leq \lambda \leq 1$

(b) Effect on the spectrum $\lambda(W)$

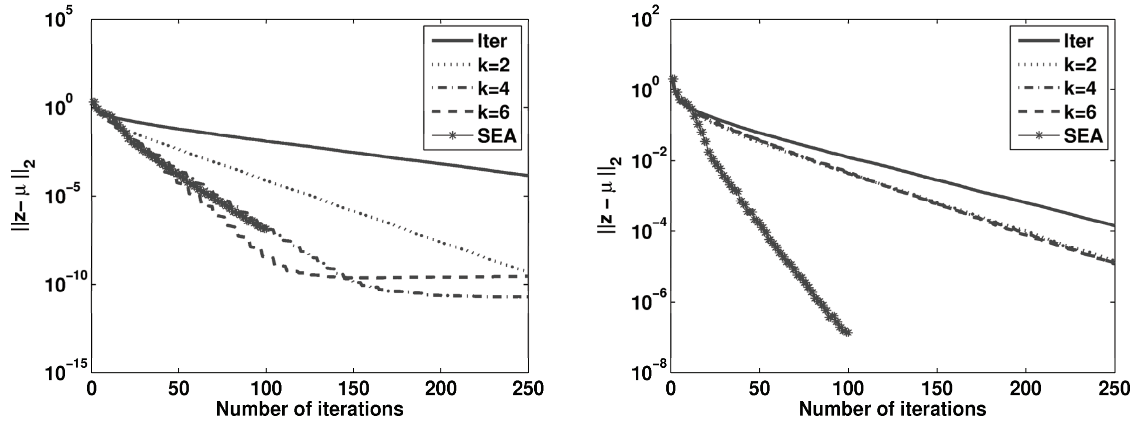
Figure 7.4: *Effect of polynomial filtering on the spectrum of the Maximum-degree matrix W of eq. (7.26).*

filtering dramatically increases the spectral gap $1-|\lambda_2(W)|$, which further leads to accelerating the distributed consensus, as we show later. This example illustrates the effect of a polynomial filter on the spectrum of a matrix. The detailed comparison between the convergence rates implied by the two matrices is further provided below.

Performance analysis We compare the performance of the different distributed consensus algorithms, with all the aforementioned weight matrices; that is, Maximum-degree, Metropolis, Laplacian and optimal weight matrices for distributed averaging. We compare both Newton's polynomial and the SDP polynomial (obtained from the solution of OPT2) with the standard iterative method, which is based on successive iterations of eq. (7.3) and is denoted by 'Iter'.

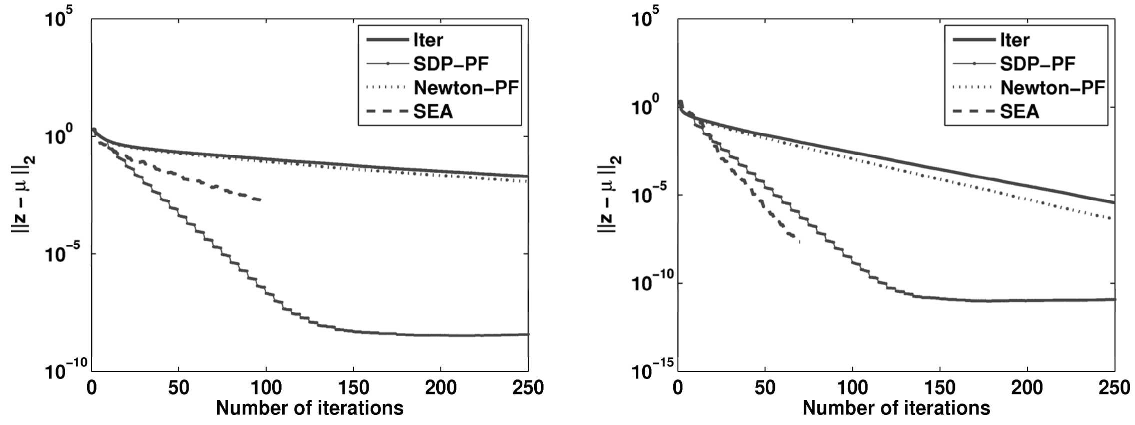
First, we explore the behavior of polynomial filtering methods under variable degree k from 2 to 6 with step 2. We use the Laplacian weight matrix for this experiment and we set $a = 0.3$. Figures 7.5(a) and 7.5(b) illustrate the evolution of the average consensus performance corresponding to the absolute error $\|z_t - \mu\mathbf{1}\|_2$ versus the iteration index t, for polynomial filtering with SDP and Newton's polynomials respectively. The curves correspond to averages over 500 random realizations of the sensor network and random (uniformly distributed) initial measurements. We also provide the curve of the standard iterative method ('Iter') as a baseline. Observe first that both polynomial filtering methods outperform the standard method by exhibiting faster convergence rates, across all values of k. Notice also that in the SDP method, the degree k clearly governs the convergence rate, since larger k implies more effective filtering and therefore faster convergence (i.e., the slope is steeper for larger k). Finally, we should note that the stagnation of the convergence process of the SDP polynomial filtering and large values of k is due to the limited accuracy of the SDP solver.

Next, we show the results obtained with the other two weight matrices. Figures 7.6(a) and 7.6(b) show the average convergence behavior of all consensus methods for the Maximum-degree and Metropolis matrices respectively, over 500 random experiments (network graphs



(a) SDP polynomial filtering (b) Newton polynomial filtering

Figure 7.5: *Average consensus performance of polynomial filtering for variable degree k on fixed topology with the Laplacian weight matrix.*

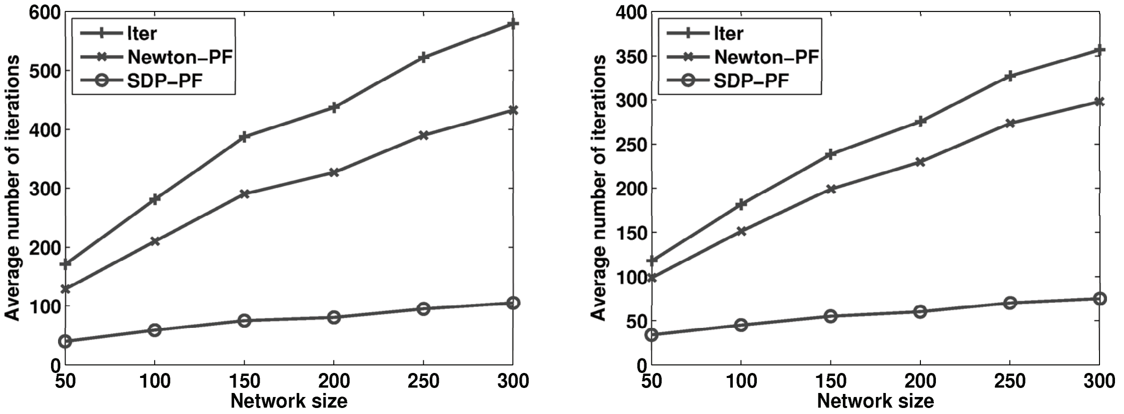


(a) Max-degree weight matrix (b) Metropolis weight matrix

Figure 7.6: *Average consensus performance of Newton's polynomial and SDP polynomial ($k = 4$) on fixed topology.*

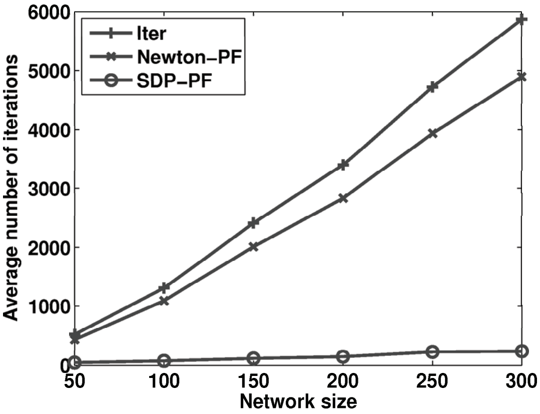
and random initial measurements). In both polynomial filtering methods we use a representative value of k, namely 4. In the case of Newton's polynomial filtering, we use $a = 0.2$ for both weight matrices. Notice again that polynomial filtering accelerates the convergence of the standard iterative method (solid line). As expected, the optimal polynomial computed with SDP outperforms Newton's polynomial, which is based on intuitive arguments only.

Furthermore, we can see from Figures 7.5 and 7.6 that in some cases the convergence rate is comparable for SEA and SDP polynomial filtering. Note however that the former uses all previous iterates, in contrast to the latter that uses only the $k+1$ most recent ones. Hence, the memory requirements are smaller for polynomial filtering, since they are directly driven by the degree of the filter k. This moreover permits a more direct control on the convergence rate, as



(a) Laplacian

(b) Metropolis

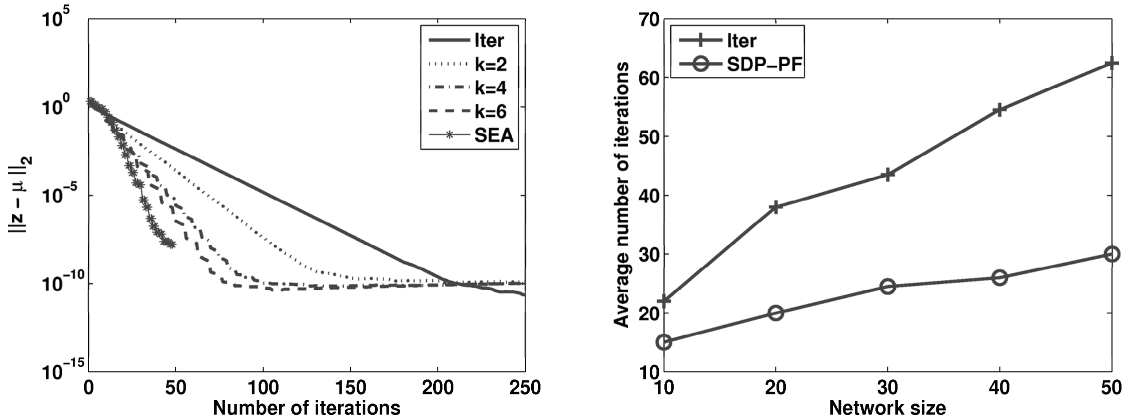


(c) Maximum-degree

Figure 7.7: *Average number of consensus iterations versus the network size m. Results with different weight matrices are shown.*

we have seen in Fig. 7.5. Interestingly, we see that the convergence process is smoother with polynomial filtering, which further permits easy extension to dynamic network topologies.

Different network sizes Next, we provide simulation results for different network sizes m, where m varies from 50 to 300 with step 50. In particular, we fix the tolerance δ of the absolute error $\|z_t - \mu\mathbf{1}\|_2$ to 10^{-3}, and we measure the average number of iterations needed by each method to reach the desired level δ of absolute error, across different network sizes. This provides an estimate on the average convergence time achieved by each method over 100 random experiments. The degree k is set to 4 for both polynomial filtering methods. Figures 7.7(a), 7.7(b) and 7.7(c) show the obtained results for the Laplacian, Metropolis and Maximum-degree weight matrices. Notice the superiority of polynomial filtering methods over the standard iterative method in terms of convergence speed, which is sustained over different network sizes.



(a) Average consensus performance of SDP polynomial filtering and SEA.

(b) Average number of iterations versus the network size m.

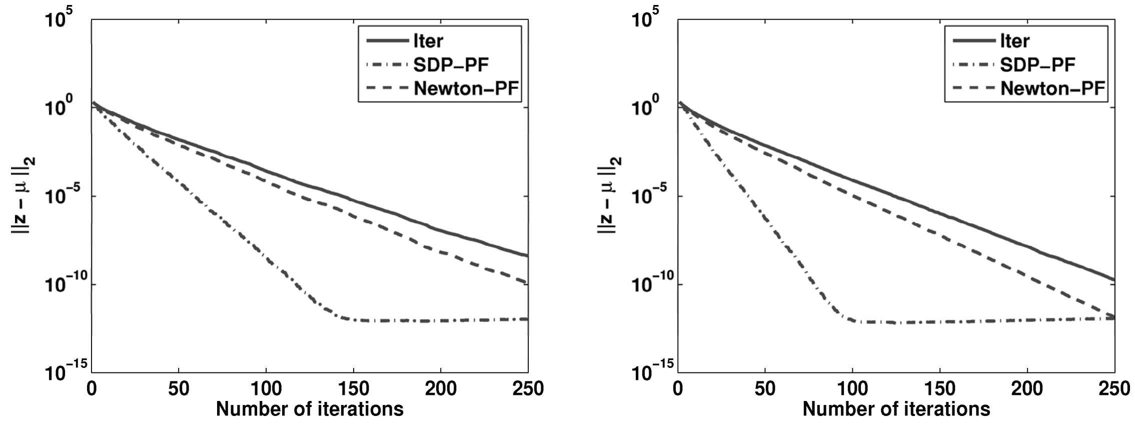
Figure 7.8: *Results of SDP polynomial filtering applied to the optimal weight matrix.*

Optimal weights Finally, for the sake of completeness, we provide simulation results with the optimal weight matrix (see Section 7.5.1). We apply our SDP polynomial filtering method on the optimal weight matrix and we report the results in Figures 7.8(a) and 7.8(b). Fig. 7.8(a) shows the average convergence behavior over 100 random experiments, for different values of the filter degree k. It can be noted that the SDP polynomial filtering is able to accelerate the linear iteration, even when the optimal weights are used. Also, increasing the degree k results in improvement on the convergence rate, but this seems to saturate for large values of k. Additionally, Fig. 7.8(b) shows the average number of iterations needed to obtain absolute error of $\delta = 10^{-3}$, when the network size m varies from 10 to 50 with step 10. In this experiment, k was set to 4 and the results correspond to averages over 100 random experiments. We observe that the improvement offered by polynomial filtering stays consistent for different network sizes.

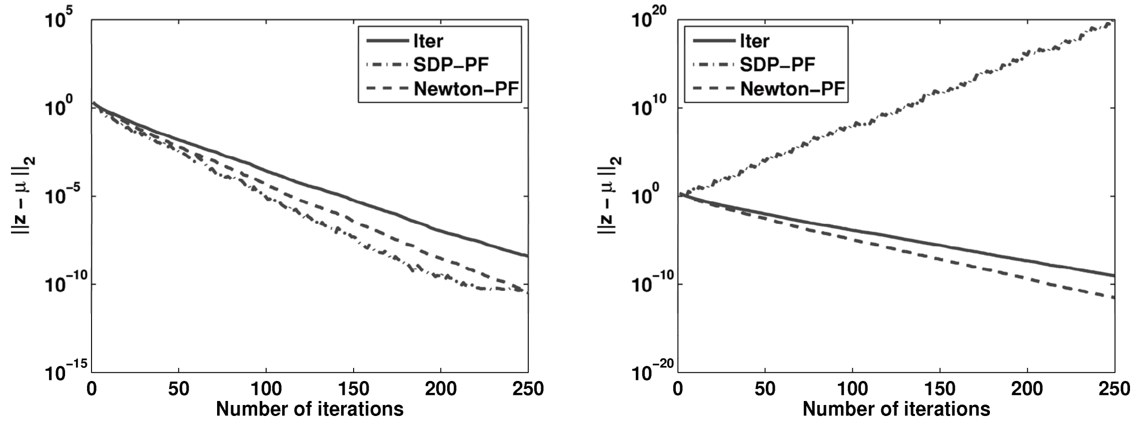
Note that in this case we had to perform simulations with fairly small network sizes, since computing the solution of the optimization problem for the optimal weights is computationally intensive. Recall that the number of unknowns is $\mathrm{O}(m^2)$ in this case. On the contrary, the number of unknowns in our SDP polynomial filtering method is only $k+2$, irrespectively of the network size.

7.5.3 Dynamic network topologies

We study now the performance of polynomial filtering for dynamic networks topologies. We build a sequence of random networks of $m = 50$ sensors, and we assume that in each iteration the network topology changes with probability q. With probability $1-q$ it remains the same as in the previous iteration. We compare all methods for different values of the probability q. We use the Laplacian weight matrix (7.28) and $a = 0.3$ in Newton's polynomial filtering. In the SDP polynomial filtering method, we solve the SDP program OPT3 (see Sec. 7.4.2). Fig. 7.9 shows the average performance of polynomial filtering for some representative values of the probability q and degree k. The average performance is computed using the median over 100 experiments. We have not reported the performance of the SEA algorithm, since it



(a) $k = 1$, $q = 0.2$ (b) $k = 1$, $q = 1$

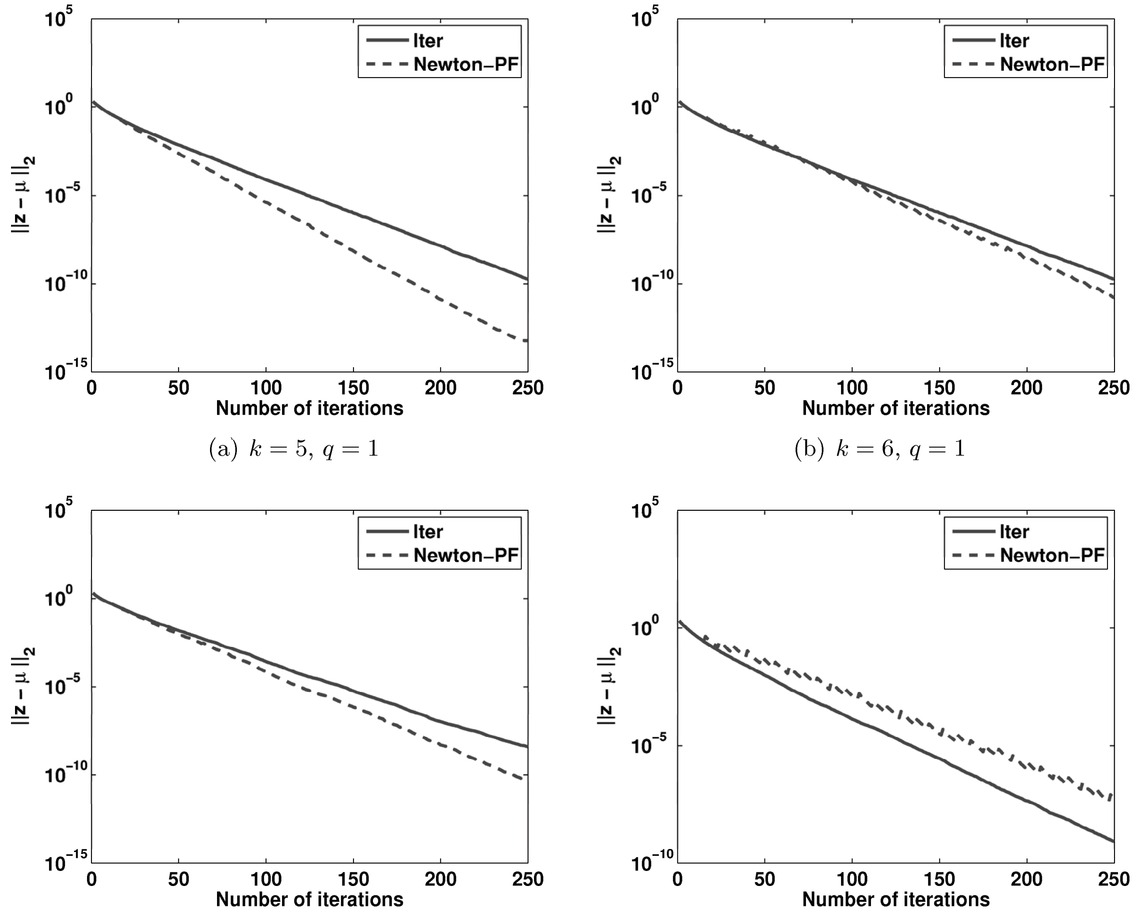


(c) $k = 2$, $q = 0.2$ (d) $k = 2$, $q = 0.8$

Figure 7.9: *Average consensus performance using the Laplacian weight matrix on random network topologies: the case of small k.*

is not robust to changes of the network topology.

Notice that when $k = 1$ (i.e., each sensor uses only its current value and the previous one) polynomial filtering accelerates the convergence over the standard iterative method for both small and large values of q. At the same time, it stays robust to network topology changes. Also, observe that in this case, the SDP polynomial outperforms Newton's polynomial. However, when $k = 2$, the roles between the two polynomial filtering methods change as the probability q increases. For instance, when $q = 0.8$, the SDP method even diverges. Thus, it appears that Newton's polynomial filtering is more robust to network changes than the SDP method. This is expected if we think that the coefficients of Newton's polynomial are computed using Hermite interpolation in a given interval and they do not depend on the specific realization of the underlying weight matrix. Thus, they are more generic than those of the SDP polynomial that takes $\overline{W}$ into account, and therefore they are less sensitive to the actual topology realization. Algorithms based on optimized polynomial filtering become therefore inefficient in a highly dynamic network whose topology changes very frequently, as



(a) $k = 5$, $q = 1$

(b) $k = 6$, $q = 1$

(c) $k = 7$, $q = 0.2$

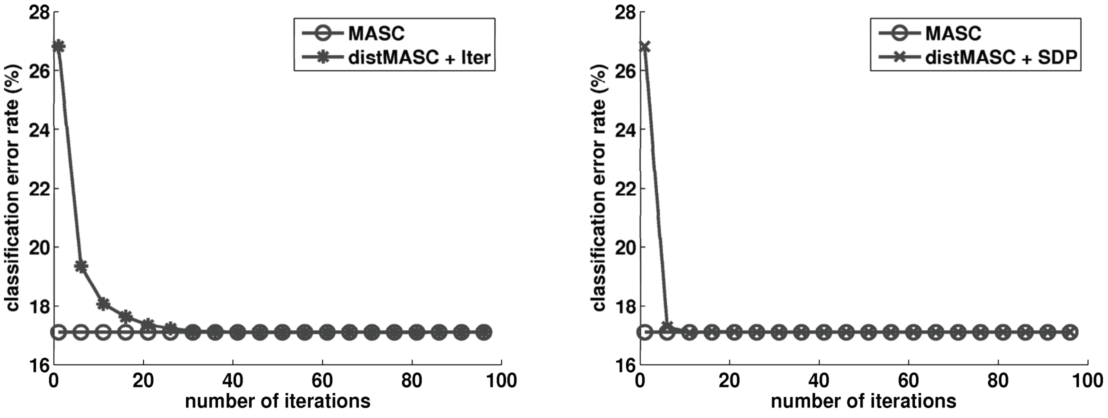
(d) $k = 7$, $q = 0.8$

Figure 7.10: *Average consensus performance using the Laplacian weight matrix on random network topologies: the case of large k.*

it becomes impossible to have any precise optimization in this case.

We perform further simulations that correspond to larger values of k in order to investigate the behavior of Newton's polynomial filtering. Fig. 7.10 shows the obtained results, when $k = 5$, 6 and $k = 7$. The experimental results suggest that polynomial filtering with Hermite interpolation stays robust to network changes when $k = 5$ and $k = 6$. This is even the case for $k = 7$ and small q. Finally, notice that for $k = 7$ and large q, the method is still able to converge, but it does not result into improvements over the standard iterative method any more. Therefore, we can conclude that Newton's polynomial filtering is robust to network changes for moderate values of k.

Let us summarize now the main findings from the simulation results of this section. For fixed network topologies, the SDP polynomial filtering method is the best performer, outperforming both the standard iterative method as well as Newton's polynomial filtering. This is consistent across different network sizes and all weight matrices that have been tested. For dynamic network topologies, SDP still outperforms the other methods when $k = 1$. However,



(a) Iterative

(b) SDP polynomial filtering

Figure 7.11: *Classification error rate vs consensus iterations with SDP polynomial filtering, where the sensor network uses the Metropolis weight matrix.*

as the degree k increases, the SDP method becomes unstable. On the other hand, Newton's polynomial filtering stays robust and outperforms the standard iterative method for moderate values of k up to 6.

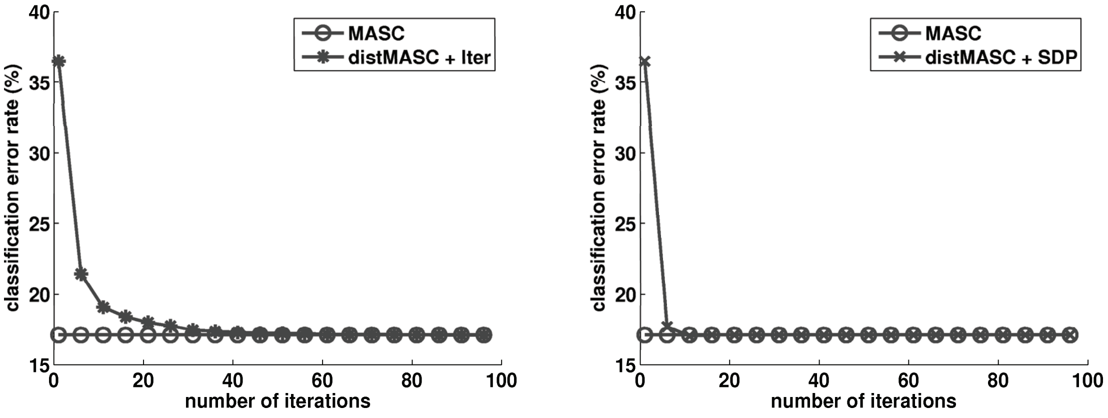
7.6 Consensus-based distributed classification

We turn now our attention on distributed classification that was presented in Chapter 6, and we analyze the efficiency of SDP polynomial filtering for faster convergence of the distributed classification applications. We use the same setup as in the experiment of Section 6.4.3. The goal of this experiment is to test whether consensus-based distributed classification can benefit from the polynomial filtering methodology. Note that in this case, the distributed MASC algorithm is based on fast consensus with SDP polynomial filtering for the computation of the sum in Line 11 of Algorithm 5. The degree of the polynomial filter is set to 4.

Figs. 7.11 and 7.12 show the obtained results using the Metropolis and Maximum-degree weight matrices respectively. We also provide the results with the standard iterative method for the sake of comparison. As expected, distributed MASC with polynomial filtering needs about 5 iterations (on the average) to achieve almost the same accuracy as the centralized MASC. This is to be contrasted with the iterative method, which needs about 30 and 40 iterations with the Maximum-degree and Metropolis matrices respectively. Therefore, polynomial filtering is able to offer efficiency for distributed classification via fast consensus.

7.7 Related work

In Section 2.5.2, we reviewed a few related methods, which try to accelerate the convergence of distributed consensus by optimizing the weights of the network links. Then, in Section 7.5 we showed that our fast consensus algorithms can accelerate the convergence of consensus, even when such optimal weight matrices are used. In the sequel, for the sake of completeness,



(a) Iterative (b) SDP polynomial filtering

Figure 7.12: *Classification error rate vs consensus iterations with SDP polynomial filtering, where the sensor network uses the Maximum-degree weight matrix.*

we mention a few representative articles that address the general problem of consensus in sensor networks. The approaches based on lifted Markov chains, gossiping or agreement, are however quite different from the work proposed in this chapter.

First we review the methods based on lifted Markov chains. It is known that the problem of fastest distributed consensus is closely related to the problem of finding the fastest mixing Markov chain (see e.g., [121]). Recently, a few articles have appeared that focus on accelerating the convergence rate to the consensus solution via lifted Markov chains. The main idea of lifting is to distinguish the graph nodes from the states of the Markov chain and to "split" the states into virtual states that are connected in such a way that permits faster mixing. The lifted graph is then "projected" back to the original graph, where the dynamics of the lifted Markov chain are simulated subject to the original graph topology. We mention the work in [70], which proposes a fast distributed averaging algorithm for geographic random graphs. In particular, the location information of the sensors is assumed to be known, which certainly represents a limitation in cases where sensor localization is not possible. The location information is used in order to construct a nonreversible lifted Markov chain that mixes faster than corresponding reversible chains. However, since the Markov chain is nonreversible, the stationary distribution is not uniform anymore. The authors introduce weights in order to overcome this problem, which in turn increases the communication cost. Moreover, the extension of their proposed methodology to dynamic network topologies is not straightforward. Along the same lines, K. Jung et al. [49], used nonreversible lifted Markov chains for fast gossip. The authors use the lifting scheme of [14] and they propose a deterministic gossip algorithm based on a set of disjoint maximal matchings, in order to simulate the dynamics of the lifted Markov chain.

In [69], the authors propose a cluster-based distributed averaging algorithm, applicable to both fixed linear iteration and random gossiping. By clustering the nodes, one may construct an overlay graph that is better connected, relatively to the original graph; hence, the random walk on the overlay graph mixes faster than the corresponding walk on the original graph. The improvement in mixing time comes nevertheless at the cost of performing the clustering and forming the overlay graph. The extension of this methodology to dynamic network topologies is also not easy. On the contrary, our polynomial filtering methodology is flexible, and as we have seen in the simulations section, it can be applied successfully in both fixed and dynamic network topologies.

Gossip algorithms have also been applied successfully to solving distributed averaging problems. Recall that the gossip constraint implies that multiple node pairs may communicate at the same time, provided that they are disjoint. In [90], the authors provide convergence results on randomized gossip algorithms in both synchronous and asynchronous settings. Based on the obtained results, they optimize the network topology (edge formation probabilities) in order to maximize the convergence rate of randomized gossip. This optimization problem is then formulated as a semi-definite program (SDP). In a recent study, the authors in [19, 22] have been able to improve the message complexity of the standard gossip protocols in cases where the sensors know their geometric positions. The main idea is to exploit geographic routing in order to aggregate values among random nodes that are far away in the network.

Finally, there are approaches related to agreement. For instance, we mention the work in [79] by A. Olshevsky and J. N. Tsitsiklis. They propose two consensus algorithms for fixed network topologies, which build on the "agreement algorithm". The proposed algorithms make use of spanning trees and the authors bound their worst-case convergence rate. For dynamic network topologies, they propose an algorithm which builds on a previously known distributed load balancing algorithm. In this case, the authors show that the algorithm has a polynomial bound on the convergence time (ϵ-convergence).

7.8 Conclusions

In this chapter, we have considered the problem of accelerating the convergence process of distributed averaging in ad-hoc sensor networks and its application to distributed classification. In particular, we have proposed a polynomial filtering methodology in order to accelerate distributed average consensus in both fixed and dynamic network topologies. The main idea of polynomial filtering is to shape the spectrum of the polynomial weight matrix in order to minimize its second largest eigenvalue and subsequently increase the convergence rate. We have constructed semi-definite programs to optimize the polynomial coefficients in both static and dynamic networks. Simulation results with several common weight matrices have shown that (i) the convergence rate is much higher than for state of the art algorithms in most scenarios, except in the specific case of highly dynamic networks and (ii) that distributed classification described in Chapter 6 can greatly benefit from the proposed fast consensus algorithms.

Chapter 8

Conclusions

8.1 Thesis achievements

This thesis studies particular instances of the problem of classification of multimedia patterns, as posed by new challenges faced by modern multimedia architectures. We first attacked semantic approximation and transformation invariance problems in multimedia pattern classification, using the handy properties of sparse geometric representations over structured redundant bases. Next, we studied classification with multiple transformed observations and designed a graph-based framework that is able to capture the geometric structure of the test examples and exploit the diversity offered by them, in order to perform effective classification in both centralized and distributed environments. In the latter case, we further offered a generic methodology for fast consensus with application to distributed classification and distributed data aggregation in general. In each particular problem studied in the thesis, we discussed important issues concerning high performance pattern classification, and we presented our analysis, results and conclusions.

First, we considered the problem of semantic approximation for classification. The goal is to identify relevant geometric primitives in multimedia patterns, which permit (i) compact representation for coding and (ii) effective classification. We proposed a flexible representation framework based on simultaneous sparse representations, which represents an effective trade-off between approximation and classification performances. Our experimental results reflect the desired trade-off and additionally show that the proposed methodology can lead to classification performances that are competitive to state of the art techniques, with the additional advantage of enabling efficient coding solutions. Therefore, compression and classification can be performed jointly, without important performance penalty with respect to expensive disjoint solutions. The proposed semantic approximation framework certainly represents a promising solution for efficient data mining in modern media processing systems.

Next, we studied transformation invariant issues in pattern analysis, in cases where the transformation consists of a synthesis of translations, rotations and scaling. We illustrated that a few atoms are sufficient to capture the salient geometric structure of the pattern of interest. We exploited the properties of such an expansion under geometric transformations, in order to formulate the manifold distance computation problem, which is moreover related to pattern alignment. Then, we proved that the optimization problem is a DC program, which can be optimally solved by any method from the arsenal of DC programming solvers. To the best of our knowledge, it is the first time that a globally optimal algorithm is proposed for this particular problem. The experimental results demonstrated that the proposed method is successful in finding the global minimizer in practice, with applications to robust face recognition and image alignment.

When the number of observations of a test object grows large, we designed a graph-based classification algorithm, which exploits the fact that the observations share the same class label. Our approach is suitable when the test examples present a manifold structure, which is typically the case with visual information sets. We have shown that learning the unknown label matrix under constraints that reflect the specificities of the problem, improves the classification performance. This permits to develop a low complexity, yet effective algorithm, which has been experimentally shown to be effective in diverse problems ranging from classification of multiple handwritten digit images to video-based face recognition.

Then, we considered distributed aspects of the classification problem, when the multiple observations are collected distributively in an ad-hoc sensor network. We provided an insight study of this scenario and we showed that distributed classification in ad-hoc sensor networks can be effectively performed using distributed average consensus. We proposed a distributed version of the above graph-based algorithm that is based on distributed averaging, in order to aggregate information from all observations across the network. We have illustrated the feasibility of such an algorithm in the context of distributed multi-view face recognition. The simulation results demonstrated that the proposed distributed algorithm and its centralized counterpart have similar performances when the training set is sufficiently large.

Finally, motivated by the above developments, we further studied the problem of accelerating the convergence of consensus, which directly drives the efficiency of any distributed algorithm relying on it. We exploited the memory of the sensors by means of polynomial filters in order to augment the convergence rate. We designed polynomial filtering algorithms based on semi-definite programming and Hermite interpolation, for both static and dynamic network topologies. The simulation results illustrated that the proposed algorithms improve the consensus convergence rate, outperforming state of the art solutions in static networks. At the same time, they stay robust to network topology changes in the dynamic case. Finally, we illustrated that distributed classification can certainly benefit from the proposed fast consensus algorithms.

8.2 Future directions

In the discussion of the semantic approximation framework in Chapter 3, we have seen that the trade-off of approximation and classification offers certain advantages to current multimedia mining systems. However, the behavior of sparse representations in practice depends a lot on the properties of the dictionary and the characteristics of the basis functions. This fact raises an interesting issue regarding what should be the properties of these basis functions. The recent advances in dictionary learning may contribute substantially in this direction. Most of the recent methods in dictionary learning are data-driven and optimize the basis functions according to some approximation-related criterion. Thus, in light of the semantic approximation framework, it would be very interesting to explore hybrid approximation-discrimination criteria for dictionary learning (see e.g., research efforts in [88]).

In Chapter 4, we introduced our transformation invariance methodology and we focused on pattern transformations that consist of synthesis of translations, rotations and scalings. It would be interesting to investigate extensions of this framework to more generic transformations, such as nonrigid deformations. This would require more generic laws that govern the parameter updates of the atoms. Such a methodology would significantly impact important vision areas, such as facial expression analysis and human body motion analysis.

We also mention that the problem of manifold sampling is very important. In Chapter 5 we introduced a graph-based method, where virtual examples are included in the graph vertex set, in order to provide robustness to pattern transformations. The virtual examples are essentially samples of the transformation manifold, which is spanned by continuous transformations of the pattern of interest. Manifold sampling in this context is a hard problem, mostly due to its high dimensionality. Intuitively, it should be performed in a way that takes the manifold geometry into account, such as curvature and geodesic distances. It certainly represents a very interesting and challenging problem.

Concerning our fast consensus algorithms introduced in Chapter 7, it is interesting to note that it may be possible to extend the polynomial filtering methodology to distributed averaging under the gossip constraint. For instance, we have already mentioned that the distributed averaging problem in random networks is closely related to the distributed gossip synchronous algorithm (see Section 7.2.2). Thus, this is certainly an issue worth investigating. Finally, it is also interesting to consider the problem of optimizing the convergence rate of distributed consensus jointly with respect to edge weights and polynomial filter coefficients.

APPENDIX

Appendix A

Appendix of Chapter 4

A.1 Proof of Proposition 3(b)

Proof. Let $f(x) = g(x) - h(x)$ be the DC decomposition of f. Then, the second part of (4.11) is convex since g and h are convex. Hence, we need to show that $p(x)$ is convex. Note that that the (i, j) entry of the Hessian of $p(x)$ is

$$\begin{aligned}
H_{ij} &= \frac{\partial^2 p(f(x))}{\partial x_i \partial x_j} = \frac{\partial^2 q(f(x))}{\partial f^2} \cdot \frac{\partial f(x)}{\partial x_i} \cdot \frac{\partial f(x)}{\partial x_j} + \\
&\quad \frac{\partial q(f(x))}{\partial f} \cdot \frac{\partial^2 f(x)}{\partial x_i \partial x_j} + K\Big[\frac{\partial^2 g(x)}{\partial x_i \partial x_j} + \frac{\partial^2 h(x)}{\partial x_i \partial x_j}\Big] \\
&= q''(f(x)) \cdot \frac{\partial f(x)}{\partial x_i} \cdot \frac{\partial f(x)}{\partial x_j} + q'(f(x)) \cdot \frac{\partial^2 f(x)}{\partial x_i \partial x_j} + \\
&\quad K\Big[\frac{\partial^2 g(x)}{\partial x_i \partial x_j} + \frac{\partial^2 h(x)}{\partial x_i \partial x_j}\Big] \\
&= q''(f(x)) \cdot \frac{\partial f(x)}{\partial x_i} \cdot \frac{\partial f(x)}{\partial x_j} + \Big[K + q'(f(x))\Big]\frac{\partial^2 g(x)}{\partial x_i \partial x_j} \\
&\quad + \Big[K - q'(f(x))\Big]\frac{\partial^2 h(x)}{\partial x_i \partial x_j}.
\end{aligned}$$

Therefore,

$$\begin{aligned}
H &= \nabla_p^2 = q''(f(x))\nabla_f \nabla_f^\top + \\
&\quad \Big[K + q'(f(x))\Big]\nabla_g^2 + \Big[K - q'(f(x))\Big]\nabla_h^2. \qquad \text{(A.1)}
\end{aligned}$$

The convexity of q implies that $q''(f(x)) \geq 0$. Also the convexity of g and h implies that ∇_g^2 and ∇_h^2 are positive semi-definite. Hence, when $K \geq |q'(f(x))|$, we conclude that H is positive semi-definite and $p(x)$ is convex. $\square$

A.2 Proof of Lemma 4.3.1

Proof. Suppose that the atom has parameters $\gamma = (b, a, \omega)$, with $a = [a_x, a_y]$ and $b = [b_x, b_y]$. If we denote by $\eta = (\beta, \alpha, \theta)$ the transformation parameters, then according to the group law, the transformed parameters of the atom will be

$$(\eta \circ \gamma) = (\beta + \alpha R(\theta) b,\ \alpha a,\ \omega - \theta).$$

If we denote $A = \begin{pmatrix} \frac{1}{\alpha a_x} & 0 \\ 0 & \frac{1}{\alpha a_y} \end{pmatrix}$ and $R(\omega) = \begin{pmatrix} \gamma_1 & \gamma_2 \\ \gamma_3 & \gamma_4 \end{pmatrix}$, then the transformed axes $\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix}$ will be

$$\begin{aligned}
& AR(\omega)R(-\theta)\Big[\begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} \beta_x \\ \beta_y \end{pmatrix} - \alpha R(\theta)\begin{pmatrix} b_x \\ b_y \end{pmatrix}\Big] \\
= \ & AR(\omega)\Big[R(-\theta)\begin{pmatrix} x \\ y \end{pmatrix} \underbrace{-R(-\theta)\begin{pmatrix} \beta_x \\ \beta_y \end{pmatrix}}_{[\tau_x\ \tau_y]^\top} - \alpha\begin{pmatrix} b_x \\ b_y \end{pmatrix}\Big] \\
= \ & AR(\omega)\Big[R(-\theta)\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} -\alpha b_x + \tau_x \\ -\alpha b_y + \tau_y \end{pmatrix}\Big] \\
= \ & AR(\omega)\begin{bmatrix} \cos\theta & -\sin\theta & -\alpha b_x + \tau_x \\ \sin\theta & \cos\theta & -\alpha b_y + \tau_y \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \\
= \ & A\begin{bmatrix} \gamma_1 & \gamma_2 \\ \gamma_3 & \gamma_4 \end{bmatrix}\begin{bmatrix} \cos\theta & -\sin\theta & -\alpha b_x + \tau_x \\ \sin\theta & \cos\theta & -\alpha b_y + \tau_y \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \\
= \ & \begin{bmatrix} \epsilon_1\frac{\cos\theta}{\alpha} + \epsilon_2\frac{\sin\theta}{\alpha} & -\epsilon_1\frac{\sin\theta}{\alpha} + \epsilon_2\frac{\cos\theta}{\alpha} & \epsilon_3 + \epsilon_4\frac{\tau_x}{\alpha} + \epsilon_5\frac{\tau_y}{a} \\ \zeta_1\frac{\cos\theta}{\alpha} + \zeta_2\frac{\sin\theta}{\alpha} & -\zeta_1\frac{\sin\theta}{\alpha} + \zeta_2\frac{\cos\theta}{\alpha} & \zeta_3 + \zeta_4\frac{\tau_x}{\alpha} + \zeta_5\frac{\tau_y}{a} \end{bmatrix} \\
& \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix},
\end{aligned}$$

where we have introduced $\epsilon_1 = \frac{\gamma_1}{a_x}$, $\epsilon_2 = \frac{\gamma_2}{a_x}$, $\epsilon_3 = \frac{-b_x\gamma_1 - b_y\gamma_2}{a_x}$, $\epsilon_4 = \epsilon_1$, $\epsilon_5 = \epsilon_2$ and similarly, $\zeta_1 = \frac{\gamma_3}{a_y}$, $\zeta_2 = \frac{\gamma_4}{a_y}$, $\zeta_3 = \frac{-b_x\gamma_3 - b_y\gamma_4}{a_y}$, $\zeta_4 = \zeta_1$ and $\zeta_5 = \zeta_2$. Then, the above formula reads

$$\begin{aligned}
& \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = \\
& \begin{pmatrix} (\epsilon_1 x + \epsilon_2 y)\frac{\cos\theta}{\alpha} + (\epsilon_2 x - \epsilon_1 y)\frac{\sin\theta}{\alpha} + \epsilon_4\frac{\tau_x}{\alpha} + \epsilon_5\frac{\tau_y}{\alpha} + \epsilon_3 \\ (\zeta_1 x + \zeta_2 y)\frac{\cos\theta}{\alpha} + (\zeta_2 x - \zeta_1 y)\frac{\sin\theta}{\alpha} + \zeta_4\frac{\tau_x}{\alpha} + \zeta_5\frac{\tau_y}{\alpha} + \zeta_3 \end{pmatrix},
\end{aligned}$$

which yields equation (4.12). □

A.3 The cutting plane algorithm

We discuss here the details of the outer-approximation cutting plane algorithm of [41, Thm 5.3]. A global optimization problem is called a DC program if it has the following form

$$\begin{aligned}
\min_x \quad & f(x) = g(x) - h(x), \\
\text{s.t.} \quad & x \in X = \{x \in \mathbb{R}^n : \delta(x) \le 0\},
\end{aligned} \tag{A.2}$$

Algorithm 7 Outer Approximation Cutting Plane

```
1: **Initialization**: Set $\omega^0 = g(y^0) - h(y^0)$, the first upper bound of the optimal value $\omega^*$ of the Problem (A.2).
2: Compute a subgradient $s \in \partial g(y^0)$ and construct the affine function $l(x) = (x - y^0)^\top s + g(y^0)$.
3: Construct a simplex $S^0 \supseteq X$ with vertex set $V(S^0) = \{v^1, \dots, v^{n+1}\}$. Choose $\bar{\omega}$ and $\bar{t}$ such that
       $\bar{\omega} = \min\{l(x) : x \in V(S^0)\} - \max\{h(x) : x \in V(S^0)\}$
       $\bar{t} > \max\{g(x) : x \in V(S^0)\} - \bar{\omega}$.
4: Construct a polytope $P^0 = \{(x,t) : x \in S^0,\ t \leq \bar{t},\ l(x) - t - \omega^0 \leq 0\} \supseteq \{(x,t) : x \in X,\ t \in \mathbb{R},\ g(x) - t - \omega^* = 0\}$ and compute its vertex set $V(P^0) = \{(v^i, \bar{t}), i = 1, \dots, n+1$ and $(v^i, l(v^i) - \omega^0), i = 1, \dots, n+1\}$.
5: Set $k = 0$.
6: **Iteration**:
7: Compute an optimal solution $(x^k, t^k)$ of the problem $\min\{-h(x) + t : (x,t) \in V(P^k)\}$
8: **if** $-h(x^k) + t^k = 0$ **then**
9:     $y^k$ is the optimal solution with optimal value $\omega^k$.
10: **else**
11:     Compute $s^k \in \partial g(x^k)$
12:     Compute the improved upper bound $\omega^{k+1} = \min\{\omega^k, g(x^k) - h(x^k)\}$.
13:     Update $y^{k+1}$ such that $g(y^{k+1}) - h(y^{k+1}) = \omega^{k+1}$.
14:     Construct the cutting plane
            $l^k(x,t) = (x - x^k)^\top s^k + g(x^k) - \omega^{k+1} - t$
15:     Set $P^{k+1} = P^k \cap \{(x,t) : l^k(x,t) \leq 0\}$ and compute $V(P^{k+1})$.
16: **end if**
17: Set $k = k + 1$ and go to step 7.
```

where $g, h : X \to \mathbb{R}$ are convex functions and $\delta : \mathbb{R}^n \to \mathbb{R}$ is a convex function. Assume that the DC Problem (A.2) is solvable and denote by ω^* its global minimum. The next proposition gives an optimality condition for Problem (A.2).

Proposition 6. *[41] The point $x^* \in X$ is an optimal solution to Problem (A.2) if and only if there exists a $t^* \in \mathbb{R}$ such that*

$$0 = \inf\{-h(x) + t :\ x \in X,\ t \in \mathbb{R},\ g(x) - t \leq g(x^*) - t^*\}. \tag{A.3}$$

The cutting plane algorithm (as well as any DC solver) needs an explicit DC decomposition of the objective function before it can be employed in practice. In the sequel, we discuss briefly the basics of the cutting plane algorithm, illustrated in Algorithm 7. The cutting plane algorithm seeks a point x^* that will satisfy the optimality condition (A.3). Each iteration involves the minimization of the concave function $-h(x) + t$ under the convex constraints $g(x) - t \leq \omega^k$, where ω^k is the best upper bound for ω^* as of iteration k. Thus, the right hand side of the convex constraint changes over the iterations. The algorithm starts with an initial feasible point y^0 which yields an initial upper bound ω^0. As soon as this bound improves, the convex constraint is refined. In the first iteration, the cutting plane method starts with a simple polyhedron P^0 which is required to contain the feasible set. The polyhedron P^0 is defined by the intersection of several halfspaces i.e., linear inequalities of the form $a_i^\top x + b_i \leq 0,\ i = 1, \ldots, m$.

It is known that a concave function defined over a convex polyhedron is minimized at one of its vertices [40, Thm 1.19]. In iteration k, the vertex x^k which minimizes $-h(x) + t$ over P^k (see Line 7) is used to define a cutting plane $l^k(x, t)$ which updates the polytope, by excluding points that do not satisfy the linear constraint $l^k(x, t) \leq 0$. The intersection of P^k with the cutting plane defines a new polytope P^{k+1} of smaller volume (see Line 15). The minimization of $-h(x) + t$ is repeated on the updated P^{k+1} and the algorithm continues by repeating the same process. The objective function f is evaluated on each vertex of P^k and whenever a better bound of ω^* is found, ω^k is updated (see Line 12) and the convex constraint is updated as well. This process is repeated until ω^k reaches ω^* and in this case the vertex which achieves $-h(x) + t = 0$ will satisfy the optimality condition (A.3). The cutting plane algorithm always converges to the global minimizer of Problem (A.2) [41, Thm 5.3].

The cutting plane algorithm is an efficient algorithm which typically reaches the vicinity of the global minimizer quickly. Each step involves only function and subgradient evaluations as well as vertex updates of the current polytope P^k. The latter process is called *vertex enumeration* and the reader is referred to [40, Sec 3.3.4] for further details.

Bibliography

[1] M. Aharon, M. Elad, and A. Bruckstein. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Trans. on Signal Processing*, 54:4311–4321, 2006.

[2] L. Thi Hoai An and P. Dinh Tao. The DC (Difference of Convex Functions) programming and DCA revisited with DC models of real world nonconvex optimization problems. *Annals of Operations Research*, 133:23–46, 2005.

[3] O. Arandjelović, G. Shakhnarovich, J. Fisher, R. Cipolla, and T. Darrell. Face recognition with image sets using manifold density divergence. *IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 1:581–588, 2005.

[4] T. C. Aysal B. N. Oreshkin and M. J. Coates. Distributed average consensus with increased convergence rate. *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, April 2008.

[5] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Comput.*, 15(6):1373–1396, 2003.

[6] D. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods.* Prentice Hall, 1989.

[7] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006.

[8] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis. Convergence in multiagent coordination, consensus and flocking. *IEEE Conf. on Decision and Control, and the European Control Conference*, pages 2996–3000, December 2005.

[9] K. W. Bowyer, K. Chang, and P. Flynn. A survey of approaches and challenges in 3D and multi-modal 3D+2D face recognition. *Computer vision and image understanding*, 101(1):1–15, January 2006.

[10] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[11] M. Carli, F. Coppola, G. Jacovitti, and A. Neri. Translation, orientation and scale estimation based on laguerre-gauss circular harmonic pyramids. *Proceedings of SPIE, Image Processing: Algorithms and Systems*, 4667:55–65, 2002.

[12] M. Çetin, L. Chen, J. W. Fisher III, A. T. Ihler, R. L. Moses, M. J. Wainwright, and A. S. Willsky. Distributed fusion in sensor networks. *IEEE Signal Processing Magazine*, 23(4):42–55, July 2006.

[13] O. Chapelle, B. Scholkopf, and A. Zien. *Semi-Supervised learning.* MIT Press, 2006.

[14] Fang Chen, László Lovász, and Igor Pak. Lifting markov chains to speed up mixing. *STOC '99: Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 275–281, 1999.

[15] S. F. Cotter, B. D. Rao, K. Engan, and K. Kreutz-Delgado. Sparse solutions to linear inverse problems with multiple measurement vectors. *IEEE Transactions on Signal Processing*, 53(7):2477–2488, July 2005.

[16] T. Navin Lal J. Weston D. Zhou, O. Bousquet and B. Scholkopf. Learning with local and global consistency. *Advances in Neural Information Processing Systems (NIPS)*, 2003.

[17] D. DeCoste and MC Burl. Distortion-invariant recognition via jittered queries. *IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2000.

[18] D. Decoste and B. Schölkopf. Training invariant support vector machines. *Machine Learning*, 46(1):161–190, 2002.

[19] A. Dimakis, A. D. Sarwate, and M. J. Wainwright. Geographic gossip: Efficient averaging for sensor networks. *IEEE Trans. on Signal Processing.* to appear.

[20] S. Eickeler, S. Müller, and G. Rigoll. Recognition of JPEG compressed face images based on statistical methods. *Image and Vision Computing Journal (special issue of Facial Image Analysis)*, 18(4):279–287, March 2000.

[21] S. Ellis and M. Nayakkankuppam. Phylogenetic analysis via DC programming (preprint). *Department of Mathematics and Statistics, UMBC, Baltimore, MD*, 2003.

[22] P. Thiran F. Bénézit, A. G. Dimakis and M. Vetterli. Order-optimal consensus through randomized path averaging. *submitted*, 2008.

[23] S. Fidler, D. Skočaj, and A. Leonardis. Combining reconstructive and discriminative subspace methods for robust classification and regression by subsampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(3):337–350, March 2006.

[24] AW Fitzgibbon and A. Zisserman. Joint manifold distance: a new approach to appearance based clustering. *IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2003.

[25] K. Flouri, B. Beferull-Lozano, and P. Tsakalides. Distributed consensus algorithms for SVM training in wireless sensor networks. *16th European Signal Processing Conference (EUSIPCO)*, 2008.

[26] MIT Center for Biological and Computation Learning. Cbcl face database #1,. `http://www.ai.mit.edu/projects/cbcl`.

[27] P. Frossard. *Robust and multi-resolution video delivery: from H.26x to Matching Pursuit based technologies.* PhD thesis, Swiss Federal Institute of Technology (EPFL), 2000.

[28] K. Fukui and O. Yamaguchi. Face recognition using multi-viewpoint patterns for robot vision. *Int. Symp. on Robotics Research*, 15:192–201, 2005.

[29] G. H. Golub and C. Van Loan. *Matrix Computations, 3rd edn.* The John Hopkins University Press, Baltimore, 1996.

[30] H. Grabner, P. M. Roth, and H. Bischof. Eigenboosting: combining discriminative and generative information. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, Minneapolis, MN, USA, June 18-23 2007.

[31] T. Graepel and R. Herbrich. Invariant pattern recognition by semidefinite programming machines. *Advances in Neural Information Processing Systems*, 16:33–40, 2004.

[32] D. B Graham and N. M Allinson. Characterizing virtual eigensignatures for general purpose face recognition. *Face Recognition: From Theory to Applications*, 163:446–456, 1998.

[33] P. Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Trans. on Information Theory*, 46(2):388–404, March 2000.

[34] B. Haasdonk and D. Keysers. Tangent distance kernels for support vector machines. *16th IEEE Int. Conf. on Pattern Recognition (ICPR)*, 2:864–868, 2002.

[35] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning.* Springer Series in Statistics, 2001.

[36] X. He and P. Niyogi. Locality preserving projections. *Advances in Neural Information Processing Systems 16 (NIPS)*, 2003.

[37] M. Heiler and C. Schnörr. Learning non-negative sparse image codes by convex programming. In *IEEE Intl. Conf. on Comp. Vision (ICCV)*, Beijing, China, 2005.

[38] M. Heiler and C. Schnörr. Learning sparse representations by non-negative matrix factorization and sequential cone programming. *Journal of Machine Learning Research*, 7:1385–1407, July 2006.

[39] R. Horst and P. M. Pardalos. *Handbook of Global Optimization*, volume 2 of *Nonconvex Optimization and Its Applications.* Kluwer Academic Publishers, 1995.

[40] R. Horst, P. M. Pardalos, and N. V. Thoai. *Introduction to Global Optimization*, volume 48 of *Nonconvex Optimization and Its Applications.* Kluwer Academic Publishers, 2nd edition, 2000.

[41] R. Horst and N. V. Thoai. DC Programming: Overview. *Journal of Optimization Theory and Applications*, 102(1):1–43, October 1999.

[42] P. O. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of Machine Learning Research*, 5:1457–1469, 2004.

[43] C. Hundt and M. Liśkiewicz. On the complexity of affine image matching. *Symposium on Theoretical Aspects of Computer Science (STACS)*, 2007.

[44] C. Hundt and M. Liśkiewicz. Two-dimensional pattern matching with combined scaling and rotation. *19th Annual Symposium on Combinatorial Pattern Matching (CPM)*, pages 5–17, June 2008.

[45] R. Figueras i Ventura, P. Vandergheynst, and P. Frossard. Low rate and flexible image coding with redundant representations. *IEEE Transactions on Image Processing*, 15(3):726–739, March 2006.

[46] T. Joachims. Transductive inference for text classification using support vector machines. *International Conference on Machine Learning (ICML)*, 1999.

[47] I.T. Jolliffe. *Principal Component Analysis.* Springer Verlag, New York, 1986.

[48] P. Jost, P. Vandergheynst, and P. Frossard. Tree-based pursuit: Algorithm and properties. *IEEE Transactions on Signal Processing*, 54(12):4685–4697, 2006.

[49] K. Jung and D. Shah. Fast gossip via nonreversible random walk. *IEEE Information Theory Workshop (ITW)*, March 2006.

[50] S. Kar and J. M. F. Moura. Distributed average consensus in sensor networks with random link failures. *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, April 2007.

[51] S. Kar and J. M. F. Moura. Sensor networks with random links: Topology design for distributed consensus. *IEEE Transactions on Signal Processing*, 56(7):3315–3326, July 2008.

[52] D. Keysers, J. Dahmen, T. Theiner, and H. Ney. Experiments with an extended tangent distance. *15th IEEE International Conference on Pattern Recognition*, 2:38–42, 2000.

[53] T-K. Kim, O. Arandjelovic, and R. Cipolla. Boosted manifold principal angles for image set-based recognition. *Pattern Recognition*, 40:2475–2484, 2007.

[54] E. Kokiopoulou and P. Frossard. Accelarating distributed consensus using extrapolation. *IEEE Signal Processing Letters*, 14(10):665–668, October 2007.

[55] E. Kokiopoulou and P. Frossard. Dimensionality reduction with adaptive approximation. In *IEEE Int. Conf. on Multimedia & Expo (ICME)*, Beijing, China, July 2-15 2007.

[56] E. Kokiopoulou and P. Frossard. Image alignment with rotation manifolds built on sparse geometric expansions. In *IEEE Int. Workshop on Multimedia Signal Processing*, Chania, Crete, Greece, October 1-3 2007.

[57] E. Kokiopoulou and P. Frossard. Minimum distance between pattern transformation manifolds: Algorithm and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008. to appear.

[58] E. Kokiopoulou and P. Frossard. Polynomial filtering for fast convergence in distributed consensus. *IEEE Transactions on Signal Processing*, 2008. to appear.

[59] E. Kokiopoulou and P. Frossard. Semantic coding by supervised dimensionality reduction. *IEEE Transactions on Multimedia*, 10(5):806–818, August 2008.

[60] E. Kokiopoulou, S. Pirillos, and P. Frossard. Graph-based classification for multiple observations of transformed patterns. *IEEE Int. Conf. Pattern Recognition (ICPR)*, December 2008.

[61] E. Kokiopoulou and Y. Saad. Orthogonal neighborhood preserving projections. In *IEEE Int. Conf. on Data Mining*, New Orleans, Louisiana, USA, November 26-30 2005.

[62] E. Kokiopoulou and Y. Saad. Orthogonal neighborhood preserving projections: A projection-based dimensionality reduction technique. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2143–2156, December 2007.

[63] J. H. Kotecha, V. Ramachandran, and A. M. Sayeed. Distributed multi-target classification in wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 23(4):703–713, April 2005.

[64] D.D. Lee and H.S. Seung. Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems*, 13:556–562, 2001.

[65] K. C. Lee, J. Ho, M. H. Yang, and D. Kriegman. Video-based face recognition using probabilistic appearance manifolds. *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 313–320, 2003.

[66] D. Leviatan and V. N. Temlyakov. Simultaneous approximation by greedy algorithms. *Advances in Computational Mathematics*, 25(1):73–90, June 2006.

[67] J. Lfberg. Yalmip : A toolbox for modeling and optimization in MATLAB. In *Proceedings of the CACSD Conference*, Taipei, Taiwan, 2004.

[68] S. Z. Li, X. Hou, H. Zhang, and Q. Cheng. Learning spatially localized, parts-based representation. In *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 1–6, 2001.

[69] W. Li and H. Dai. Cluster-based fast distributed consensus. *IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, April 2007.

[70] W. Li and H. Dai. Location-aided fast distributed consensus. *IEEE Transactions on Information Theory*, June 2007. submitted.

[71] R. Sala Llonch, E. Kokiopoulou, I. Tosic, and P. Frossard. 3D face recognition using sparse spherical representations. *IEEE Int. Conf. Pattern Recognition (ICPR)*, December 2008.

[72] R. Sala Llonch, E. Kokiopoulou, I. Tosic, and P. Frossard. 3D face recognition with sparse spherical representations. *Pattern Recognition*, October 2008. submitted.

[73] S. Mallat. *A Wavelet Tour of Signal Processing, 2nd edn.* Academic Press, 1998.

[74] S. Mallat and Z. Zhang. Matching pursuit with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397415, December 1993.

[75] D. J. Miller and S. Pal. Transductive methods for distributed ensemble classification. *40th Annual Conference on Information Sciences and Systems*, pages 1605–1610, March 2006.

[76] A. Neri and G. Jacovitti. Maximum likelihood localization of 2-d patterns in the gauss-laguerre transform domain: Theoretic framework and preliminary results. *IEEE Trans. on Image Processing*, 13(1):72–85, January 2004.

[77] X. Nguyen, M. J. Wainwright, and M. I. Jordan. Decentralized detection and classification using kernel methods. In *21st International Conference on Machine Learning (ICML)*, 2004.

[78] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 1999.

[79] A. Olshevsky and J. Tsitsiklis. Convergence rates in distributed consensus and averaging. *IEEE Conference on Decision and Control*, December 2006. San Diego, CA.

[80] P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5:11–126, 1994.

[81] A. Pascual-Montano, J. Carazo, K. Kochi, D. Lehmann, and R. Pascual-Marqui. Nonsmooth nonnegative matrix factorization (nsnmf). *IEEE Transactions on Pattern Analysis and Machine Inteligence*, 28(3):403–415, March 2006.

[82] V. P. Pauca, J. Piper, and R. J. Plemmons. Non-negative matrix factorization for spectral data analysis. *Linear Algebra and its Applications*, 416:29–47, 2006.

[83] T. V. Pham and A. W.M. Smeulders. Sparse representation for coarse and fine object recognition. *IEEE Trans. on Patt. Anal. and Mach. Intell.*, 28(4):555–567, April 2006.

[84] V. Popovici, S. Bengio, and J-P Thiran. Kernel matching pursuit for large datasets. *Pattern Recognition*, 38:2385–2390, 2005.

[85] V. Popovici and J-P Thiran. Adaptive kernel matching pursuit for pattern classification. *IASTED Int. Conf. on Artificial Intelligence and Applications*, 2004.

[86] A. Pozdnoukhov and S. Bengio. Graph-based transformation manifolds for invariant pattern recognition with kernel methods. *IEEE Int. Conf. on Pattern Recognition (ICPR)*, 2006.

[87] M. Rabbat, J. Haupt, A. Singh, and R. Nowak. Decentralized compression and predistribution via randomized gossiping. *5th ACM Int. Conf. on Information Processing in Sensor Networks (IPSN)*, pages 51 – 59, April 2006.

[88] F. Rodriguez and G. Sapiro. Sparse representations for image classification: learning discriminative and reconstructive non-parametric dictionaries. *IMA preprint series 2213*, June 2008.

[89] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

[90] B. Prabhakar S. Boyd, A. Ghosh and D. Shah. Randomized gossip algorithms. *IEEE Transactions on Information Theory*, 52:2508–2530, June 2006.

[91] F. Samaria and A. Harter. Parameterisation of a stochastic model for human face identification. In *2nd IEEE Workshop on Applications of Computer Vision*, Sarasota, FL, December 1994.

[92] C. Sanderson. *Biometric Person Recognition: Face, Speech and Fusion.* VDM-Verlag, 2008.

[93] J. Schiff, D. Antonelli, A. G. Dimakis, D. Chu, and M. J. Wainwright. Robust message-passing for statistical inference in sensor networks. *Int. Conf. on Information Processing in Sensor Networks (IPSN)*, April 2007.

[94] R. Schilling and S. Harris. *Applied numerical methods for engineers using MATLAB and C.* McGraw-Hill, 3rd edition, 1998.

[95] I. D. Schizas, A. Ribeiro, and G. B. Giannakis. Consensus in ad hoc WSNs with noisy links - Part I: Distributed estimation of deterministic signals. *IEEE Transactions on Singal Processing*, 56(1):350–364, January 2008.

[96] B. Schlkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, December 2001.

[97] B. Scholkopf, C. Burges, and V. Vapnik. Incorporating invariances in support vector learning machines. *Artificial Neural Networks— ICANN*, 96:47–52, 1996.

[98] B. Scholkopf, P. Simard, A. Smola, and V. Vapnik. Prior knowledge in support vector kernels. *Advances in Neural Information Processing Systems*, 10:640–646, 1998.

[99] G. Shakhnarovich, J. W. Fisher, and T. Darrel. Face recognition from long-term observations. *European Conference on Computer Vision (ECCV)*, 3:851–868, 2002.

[100] P. Simard, Y. Le Cun, J. Denker, and B. Victorri. Transformation invariance in pattern recognition - tangent distance and tangent propagation. *Neural networks: tricks of the trade*, 1524:239–274, 1998. LNCS, Springer.

[101] A. J. Smola and B. Scholkopf. Sparse greedy matrix approximation for machine learning. *Int. Conf. on Machine Learning (ICML)*, pages 911–918, 2000.

[102] C. Stauffer. Minimally-supervised classification using multiple observation sets. *IEEE Int. Conf. on Computer Vision (ICCV)*, 2003.

[103] J. F. Sturm. Implementation of interior point methods for mixed semidefinite and second order cone optimization problems. *EconPapers 73*, August 2002. Tilburg University, Center for Economic Research.

[104] S. Sundaram and C. N. Hadjicostis. Distributed consensus and linear functional calculation in networks: An observability perspective. *6th ACM Int. Conf. on Information Processing in Sensor Networks (IPSN)*, April 25-27 2007.

[105] M. Szummer and T. Jaakkola. Partially labeled classification with markov random walks. *Advances in Neural Information Processing Systems (NIPS)*, 2002.

[106] A. Tahbaz-Salehi and A. Jadbabaie. On consensus over random networks. *In Proceedings of 44th annual Allerton Conference on Communication, Control and Computing*, pages 1315–1321, September 2006.

[107] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[108] P. Thévenaz, U. E. Ruttimann, and M. Unser. A pyramidal approach to subpixel registration based on intensity. *IEEE Transactions on Image Processing*, 7(1):27–41, January 1998.

[109] R. Tron and R. Vidal. Distributed face recognition via consensus on SE(3). *8th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS)*, 2008.

[110] J. Tropp. Algorithms for simultaneous sparse approximation. part ii: Convex relaxation. *Signal Processing, special issue "Sparse approximations in signal and image processing"*, 86:589–602, April 2006.

[111] J. Tropp, A. Gilbert, and M. Strauss. Algorithms for simultaneous sparse approximation. Part I: Greedy pursuit. *Signal Processing, special issue "Sparse approximations in signal and image processing"*, 86:572–588, April 2006.

[112] V. Vapnik. *Statistical Learning Theory.* Wiley, New York, 1998.

[113] N. Vasconcelos and A. Lippman. A multiresolution manifold distance for invariant image similarity. *IEEE Transactions on Multimedia*, 7(1):127–142, February 2005.

[114] P. Vincent and Y. Bengio. Kernel matching pursuit. *Machine Learning*, 48(1):165–187, December 2001.

[115] P. Viola and M. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[116] H. Wang and G. Feng. Face recognition based on HMM in compressed domain. In *Image Processing: Algorithms and Systems, Neural Networks, and Machine Learning. Proceedings of the SPIE*, volume 6064, pages 523–530, 2006.

[117] R. Wang, S. Shan, X. Chen, and W. Gao. Manifold-manifold distance with application to face recognition based on image set. *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[118] Y. Wang, Y. Jia, C. Hu, and M. Turk. Fisher non-negative matrix factorization for learning local features. In *Asian Conference on Computer Vision*, pages 806–811, Jeju Island, Korea, Jan. 27-30th.

[119] A. Webb. *Statistical Pattern Recognition, 2nd edn.* Wiley, 2002.

[120] J. Wright, A. Y. Yang, Arvind Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Trans. on Patt. Anal. and Mach. Intell. (TPAMI).* in press.

[121] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. *Systems and Control Letters*, (53):65–78, February 2004.

[122] L. Xiao, S. Boyd, and S. Lall. A scheme for robust distributed sensor fusion based on average consensus. *Int. Conf. on Information Processing in Sensor Networks*, pages 63–70, April 2005. Los Angeles.

[123] L. Xiao, S. Boyd, and S. Lall. Distributed average consensus with time-varying metropolis weights. *Automatica*, June 2006. submitted.

[124] O. Yamaguchi, K. Fukui, and K. Maeda. Face recognition using temporal image sequence. *IEEE Int. Conf. on Automatic Face and Gesture Recognition*, pages 318–323, 1998.

[125] A.Y. Yang, S. Iyengar, S. Sastry, R. Bajcsy, P. Kuryloski, and R. Jafari. Distributed segmentation and classification of human actions using a wearable motion sensor network. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–8, June 2008.

[126] M. E. Yildiz and A. Scaglione. Differential nested lattice encoding for consensus problems. *ACM Int. Conf. on Information Processing in Sensor Networks (IPSN)*, April 2007.

[127] S. Zafeiriou, A. Tefas, I. Buciu, and I. Pitas. Exploiting discriminant information in non-negative matrix factorization with application to frontal face verification. *IEEE Transactions on Neural Networks*, 17(3):683–695, May 2006.

[128] S. Zhou and R. Chellappa. From sample similarity to ensemble similarity: Probabilistic distance measures in reproducing kernel hilbert space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(6):917–929, June 2006.

[129] S. K. Zhou and R. Chellappa. Probabilistic identity characterization for face recognition. *IEEE Int. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2:805–812, 2004.

[130] X. Zhu and Z. Ghahramani. Learning from labeled and unlabeled data with label propagation. *Technical report CMU-CALD-02-107*, 2002. Carnegie Mellon University, Pittsburgh.

[131] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. *20th Int. Conf. on Machine Learning (ICML)*, 2003.

[132] B. Zitová and J. Flusser. Image registration methods: a survey. *Image and Vision Computing*, 21(11):977–1000, 2003.

Curriculum Vitae

Full name: **Effrosyni Kokiopoulou**

Degrees: Master of Science, Computer Science and Engineering Department, University of Minnesota, USA.

Master of Science, Computer Engineering and Informatics Department, University of Patras, Greece.

Diploma of Engineering, Computer Engineering and Informatics Department, University of Patras, Greece.

Address:
Signal Processing Laboratory (LTS4)
Electrical Engineering Institute (IEL)
School of Engineering (STI)
Swiss Federal Institute of Technology (EPFL)
ELE 234, Batiment ELE
Station 11
CH-1015, Lausanne
Switzerland

Contact numbers:
Tel. (+41 21) 693 4329
Fax. (+41 21) 693 76 00
E-mail: effrosyni.kokiopoulou@epfl.ch

Civil status: Married

Date and place of birth: December 11th 1979, Athens, Greece.

Nationality: Greek

Professional Experiences

Since 2005	Research Assistant at the Signal Processing Laboratory (LTS4), Electrical Engineering Institute (IEL), Swiss Federal Institute of Technology at Lausanne (EPFL) - pursuing a Ph.D. degree in pattern recognition, under the direction of Prof. Pascal Frossard.
2003 - 2005 ***2 years***	Research Assistant at the Computer Science and Engineering Department, University of Minnesota, USA - pursuing an Msc degree in pattern recognition, under the direction of prof. Yousef Saad.
2002-2003 ***1 year***	Research Assistant at the Computer Engineering and Informatics Department, University of Patras, Greece - pursuing an Msc degree in numerical algorithms for SVD with applications to data analysis.

Honors and Awards

- "Emerging Leaders in Multimedia Research" Workshop 2007, T.J. Watson IBM Research, (total number of participants: 8 PhD students, worldwide selection).
- Finalist of the Google Europe Anita Borg Scholarship 2007.
- Succeeded in the PhD Qualifying exam (WPE) at the University of Minnesota, in Sept. 2004 (first attempt) with distinction.
- Bodossaki Foundation Fellow for the academic year 2002-2003.
- Academic excellence award from the Technical Chamber of Greece for undergraduate studies (2002).
- Class valedictorian award from the Hellenic Scholarship Foundation (2002).
- 1997-1998 (1st year of studies), 1998-1999 (2nd year of studies), 1999-2000 (3rd year of studies), 2000-2001 (4th year of studies): Scholarship Beneficiary of Hellenic Scholarship Foundation for superior academic performance.

Research Experience

Graduate Research at EFPL Advisor: prof. P. Frossard

- Transformation invariance analysis of visual data with applications to invariant image distances and transformation invariant pattern recognition.
- Design of semantic dimensionality reduction methods using sparse geometric representations. Applications to 2D/3D face recognition and speech pattern matching.
- Graph-based methods for classification of multiple observations. Applications to video-based face recognition.
- Design of fast algorithms for distributed consensus-based classification.

Graduate Research at the University of Minnesota Advisor: prof. Y. Saad

- Development of orthogonal graph-based dimensionality reduction techniques that respect the data geometry and topology.
- Development of dimensionality reduction techniques, based on polynomial filtering, which avoid completely any eigenvalue calculations.
- Development of a stochastic method for the estimation of the diagonal of a large matrix, which is available only via matrix-vector products. Applications of this problem appear in computational material science.

Undergraduate Research at the University of Patras Advisor: prof. E. Gallopoulos

- Development of an implicitly restarted Krylov subspace method for the computation of the smallest singular triplets of large sparse matrices.
- Parallel implementation of the above method in both shared memory and distributed memory platforms.
- Development of parallel algorithms for the efficient computation of matrix pseudospectra.

Teaching Experience

- Graduate Teaching Assistant, EPFL.
  Digital Signal Processing, Spring 2008,
  Instructor: Prof. P. Frossard.

- Graduate Teaching Assistant, EPFL.
  Image Communication, Spring 2006,
  Instructor: Prof. P. Frossard.

- Graduate Teaching Assistant, Univ. of Patras.
  Scientific Computing I, Fall 2002,
  Instructor: Prof. E. Gallopoulos.

- Undergraduate Teaching Assistant (assignments grader), Univ. of Patras.
  Numerical Methods in Finance, Fall 2001,
  Instructor: Prof. E. Houstis.

Languages

Greek:	mother tongue
English:	fluent
French:	basic

Computer Skills

Programming languages:	Matlab , C/C++, Java, Fortran 77/90.
Operating Systems:	Linux, Windows.
Parallel programming:	MPI, OpenMP, PVM.

Publications

Patents

P1 E. Kokiopoulou, P. Frossard, O. Verscheure and L. Amini, ”Method and system for robust pattern matching in continuous speech”, US patent, June 2008, filed.

Journal Papers

J11 E. Kokiopoulou and P. Frossard, “Classification of multiple observations by semi-supervised learning”, submitted.

J10 R. Sala Llonch, E. Kokiopoulou, I. Tosic and P. Frossard, “3D face recognition with sparse spherical representations”, submitted.

J9 E. Kokiopoulou and P. Frossard, “Polynomial Filtering for Fast Convergence in Distributed Consensus”, IEEE Transactions on Signal Processing, in press.

J8 E. Kokiopoulou and P. Frossard, “Minimum Distance between Pattern Transformation Manifolds: Algorithm and Applications”, IEEE Transactions on Pattern Analysis and Machine Intelligence, in press.

J7 E. Kokiopoulou and P. Frossard, “Semantic Coding by Supervised Dimensionality Reduction”, IEEE Transactions on Multimedia, vol. 10, no. 5, pp. 806-818, August 2008.

J6 E. Kokiopoulou and Y. Saad, “Orthogonal Neighborhood Preserving Projections: A Projection-based Dimensionality Reduction Technique”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 12, pp. 2143-2156, December 2007.

J5 E. Kokiopoulou and P. Frossard, “Accelerating Distributed Consensus Using Extrapolation”, IEEE Signal Processing Letters, Vol. 14, No. 10, October 2007.

J4 C. Bekas, E. Kokiopoulou and Y. Saad, “Computation of Large Invariant Subspaces Using Polynomial Filtered Lanczos Iterations with Applications in Density Functional Theory”, SIAM Journal on Matrix Analysis and Applications (SIMAX), vol. 30, no. 1, pp. 397-418, April 2008.

J3 C. Bekas, E. Kokiopoulou and Y. Saad, “An Estimator for the Diagonal of a Matrix”, J. Appl. Num. Math., vol. 57(11-12): pp. 1214-1229, November 2007.

J2 E. Kokiopoulou, C. Bekas and E. Gallopoulos, “Computing Smallest Singular Triplets with Implicitly Restarted Lanczos Bidiagonalization”, J. Appl. Num. Math., vol. 49(1): pp. 39-61, 2004.

J1 C. Bekas, E. Kokiopoulou and E. Gallopoulos, “The Design of a Distributed PSE for Computing Pseudospectra”, Future Generation Computer Systems, Vol. 21, No 6, pp. 930-941, April 2005.

Referred conference papers

C16 E. Kokiopoulou, S. Pirillos and P. Frossard, “Graph-based Classification for Multiple Observations of Transformed Patterns”, accepted to IEEE Int. Conf. on Pattern Recognition (ICPR), Tampa, Florida, December 2008.

C15 R. Sala Llonch, E. Kokiopoulou, I. Tosic and P. Frossard, “3D Face Recognition using Sparse Spherical Representations”, accepted to IEEE Int. Conf. on Pattern Recognition (ICPR), Tampa, Florida, December 2008.

C14 E. Kokiopoulou, P. Frossard and D. Gkorou “Optimal Polynomial Filtering for Accelerating Distributed Consensus”, IEEE Int. Symposium on Information Theory (ISIT), Toronto, Canada, July 2008.

C13 E. Kokiopoulou, P. Frossard and O. Verscheure, “Fast Keyword Detection with Sparse Time-frequency Models”, IEEE Int. Conf. on Multimedia and Expo (ICME), Hannover, Germany, June 2008.

C12 B. Ruf, E. Kokiopoulou and M. Detyniecki, “Mobile museum guide based on fast SIFT recognition”, 6th International Workshop on Adaptive Multimedia Retrieval (AMR), Berlin, June 26-27, 2008.

C11 E. Kokiopoulou and P. Frossard, “Image Alignment with Rotation Manifolds Built on Sparse Geometric Expansions”, IEEE Int. Workshop on Multimedia Signal Processing (MMSP), October 1-3, 2007, Chania, Crete, Greece.

C10 E. Kokiopoulou and P. Frossard, “Dimensionality Reduction with Adaptive Approximation”, IEEE Int. Conf. on Multimedia & Expo (ICME), July 2-5, 2007, Beijing, China.

C9 E. Kokiopoulou and P. Frossard, “Classification-Specific Feature Sampling for Face Recognition”, IEEE Int. Workshop on Multimedia Signal Processing (MMSP), October 3-6, 2006, BC, Canada.

C8 E. Kokiopoulou and P. Frossard , “Pattern Detection by Distributed Feature Extraction”, IEEE Int. Conf. on Image Processing (ICIP), October 8-11, 2006, Atlanta, GA, USA.

C7 E. Kokiopoulou and P. Frossard, “Distributed SVM Applied to Image Classification”, IEEE Int. Conf. on Multimedia and Expo (ICME), July 9-12, 2006, Toronto, Canada.

C6 E. Kokiopoulou and Y. Saad, “Face Recognition using OPRA-faces”, IEEE Int. Conf. on Machine Learning and Applications (ICMLA), December 15-17, 2005, Los Angeles, CA, USA.

C5 E. Kokiopoulou and Y. Saad, “Orthogonal Neighborhood Preserving Projections”, IEEE Int. Conf. on Data Mining (ICDM), November 27-30, 2005, Houston, TX, USA.

C4 E. Kokiopoulou and Y. Saad, “PCA without eigenvalue calculations: a case study on face recognition”, SIAM Data Mining Conference, April 21-23, 2005, Newport, CA.

C3 E. Kokiopoulou and Y. Saad, “Polynomial Filtering in Latent Semantic Indexing for Information Retrieval”, in Proc. ACM SIGIR ’04 Conference on Research and Development in Information Retrieval, July 25-29 2004, Sheffield, UK.

C2 C. Bekas, E. Kokiopoulou, E. Gallopoulos and V. Simoncini, “Parallel Computation of Pseudospectra using Transfer Functions on a MATLAB-MPI Cluster Platform”, 9th EuroPVM/MPI, pp.199-207, September 29-October 2 2002, Austria.

C1 C.Bekas, E. Kokiopoulou, I. Koutis and E. Gallopoulos, “Towards the Effective Parallel Computation of Matrix Pseudospectra”, ACM-ICS 2001, June 16-21 2001, Sorrento, Italy.