Feature Partitioning Approaches to Principal Component Analysis

A Thesis submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
in
Computer Science

by

K. Vijaya Kumar
(Regn. No: 03MCPC18)

Supervisor
Dr. Atul Negi

Department of Computer and Information Sciences
School of Mathematics and Computer/Information Sciences
University of Hyderabad
P.O. Central University
Hyderabad - 500 046, A.P., India

March 2009

UNIVERSITY OF HYDERABAD

P.O. CENTRAL UNIVERSITY
HYDERABAD - 500 046, A.P. (INDIA)

School of Mathematics and Computer/Information Sciences
Department of Computer and Information Sciences

CERTIFICATE

This is to certify that the thesis entitled **"Feature Partitioning Approaches to Principal Component Analysis"**, being submitted by **Mr. K. Vijaya Kumar (Regn. No.: 03MCPC18)**, in fulfillment of the requirements for the award of degree of Doctor of Philosophy in Computer Science, University of Hyderabad, is a record of bonafide research work carried out by him. He has worked under my guidance and supervision in conformity of the rules and regulations of University of Hyderabad. The matter embodied in this thesis has not been submitted in part or in full to any other university or institute for the award of any degree or diploma.

Dr. Atul Negi
Supervisor
Reader
Dept. of Computer and Information Sciences
University of Hyderabad

Prof. Arun Agarwal
Head
Dept. of Computer and Information Sciences
University of Hyderabad

Prof. T. Amarnath
Dean
School of Mathematics and Computer/Information Sciences
University of Hyderabad

Note: Signatures are available on hard copy.

DECLARATION

I, **Mr. K. Vijaya Kumar**, hereby declare that the work presented in this thesis has been carried out by me under the supervision of **Dr. Atul Negi**, Reader, Department of Computer and Information Sciences, University of Hyderabad, India, for the full period as per the PhD ordinances of University of Hyderabad.

I declare to the best of my knowledge that no part of this thesis was earlier submitted for the award of a research degree of any University.

Date: **Signature of the candidate**

Name : K. Vijaya Kumar
Regn. No. : 03MCPC18
Email : kadappakumar@gmail.com
Phone : 91-40-9440946151

Note: Signature is available on hard copy.

Acknowledgments

Completing this doctoral work has been a joyful and often overwhelming experience. I have been very privileged to have undoubtedly the most caring, inspiring, smart and supportive supervisor Sri Atul Negi. He made me to learn right from core concepts of PR. He constantly boosted my confidence by his wonderful personality. He never treated me like a student, but as a good friend. In fact, whenever I speak to him I used to get high levels of energy and confidence not only with respect to my doctoral work, also in personal life, he is such a wonderful person. I profoundly thank him for his exceptional guidance throughout my doctoral study.

I am grateful to my Doctoral Review Committee (DRC) members Sri Chakravarthy Bhagvati, Sri S. Bapi Raju for their most inspiring feedback on my progress, every semester. At times, I was not confident of the significance of my work, DRC members feedback was immense.

I am thankful to Sri C. Raghavendra Rao who helped me, during lien period of my supervisor.

I thank administration and management of Vasavi College of Engineering, Hyderabad, in particular Sri P. Balaji, Hon. Secretary, for funding me to register in various conferences.

My thanks are very much due for my colleagues, in particular, Sri P. Hemagiri Rao and Smt. S. Sreelakshmi, Department of Computer Applications, Vasavi College of Engineering and staff of University of Hyderabad for their selfless support.

Most importantly, I express my heartfelt thanks to my parents (Smt. K. Naga Rathnamma and Sri. K. V. Veeranna), my in-laws (Smt. D. Vijaya and Sri. D. Veera

Bhaskar) and my wife (Smt. D. V. Rajeshwari Devi) for their invaluable support at home, without which I would have not completed my doctoral work.

Last, but not the least, I am indebted to my wonderful and naughty daughter, Adithi (4 Yrs. 11 months), for sparing me for having not spent much time in playing with her.

I thank one and all who directly or indirectly helped me to complete my doctoral work.

Dedicated to my father Sri Veeranna,
my mother Smt Naga Rathnamma,
and my wife Smt Rajeshwari (Anju)

Supervisor

Atul Negi

Author

K. Vijaya Kumar

Feature Partitioning Approaches to Principal Component Analysis

Abstract

Dimensionality reduction (Feature extraction) plays a crucial role in developing a pattern recognition (PR) system. Principal Component Analysis (PCA) is one of the well-known techniques for dimensionality reduction with numerous and diverse applications. Despite the popularity of PCA, its major drawbacks are seen as low selectivity of local (i.e. block based) features and poor scalability to high dimensional data due to large computational complexity. To address these problems, block based PCA methods (here referred to as 'Feature Partitioning based PCA' (FP-PCA) methods) were proposed. These are based on the idea of dividing each pattern into blocks (sub-patterns). In the literature, some FP-PCA methods were seen to be practically superior to classical PCA method in terms of reduced computational complexity and improved recognition rate (for prominent local variations). However, these existing FP-PCA methods have several shortcomings. They do not use the entire covariance structure due to limitations in the partitioning scheme, which results in more principal components (poorer dimensionality reduction). These methods do not have a proper conceptual basis and their formal properties are not studied. In addition, FP-PCA methods may not perform well when global variations across patterns are predominant. Further, these FP-PCA methods do not take advantage of matrix structure of image data which is used by popular 2DPCA methods. From our study,

we find that FP-PCA methods have not been applied to cluster analysis and subspace classification. In this work, we propose novel methods (i) which perform well in both the contexts, that is, when local and global variations are prominent, (ii) which overcome the problems faced by both PCA and the existing FP-PCA methods. First, we propose a novel framework that generalizes the FP-PCA methods into a common framework and bring out several basic issues to be addressed in this context. Subsequently, we propose a novel FP-PCA approach, SubXPCA, to address 'loss of covariance due to partitioning', which improves classification and yields better dimensionality reduction. Further, we extend our work for application on image data by proposing new FP-PCA methods SIMPCA and FLPCA, which make use of feature partitioning concept and the more appropriate matrix structure of image data. To understand the variance-covariance structure captured by FP-PCA methods, we perform a theoretical analysis and propose the properties of FP-PCA methods. The theoretical study may form a basis for future investigation and evolution of new FP-PCA methods. Finally, the feature partitioning concept is applied to correlation connected cluster analysis and subspace classification, which improves the efficiency of these pattern recognition techniques. From our experimentation on UCI repository of Machine Learning database, face data sets (Yale, ORL, UMIST, CMU) and PolyU palmprint data, we prove the superiority of our proposed FP-PCA approaches in terms of local and global feature extraction, classification, time complexity and reducing small sample size problem. The superiority is shown in comparison to holistic PCA methods (e.g. classical PCA, 2DPCA) and other FP-PCA methods.

Contents

List of Figures

List of Tables

Citations to Previously Published Work from this Thesis

The work presented in this thesis has been published in various International Journals and Conferences.

1. Kadappagari Vijaya Kumar and Atul Negi, "SubXPCA and a generalized feature partitioning approach to principal component analysis", *Pattern Recognition*, Vol. 41, No. 4, Apr. 2008, pp. 1398-1409.

2. Kadappagari Vijaya Kumar and Atul Negi, "Novel approaches to principal component analysis of image data based on feature partitioning framework", *Pattern Recognition Letters*, Vol. 29 Issue 3, Feb. 2008, pp. 254-264.

3. Kadappagari Vijaya Kumar and Atul Negi, "A feature partitioning approach to subspace classification", *In Proceedings of IEEE TenCon 2007 Conference*, Taipei, Taiwan, pp. 1-4, 30.10.2007-Nov. 2^{nd} 2007.

4. Kadappagari Vijaya Kumar and Atul Negi, "A novel approach to eigenpalm features using feature partitioning framework", *In Proceedings of IAPR conference on Machine Vision and Applications*, Japan, pp. 29-32, May 16-18^{th} 2007.

5. Kadappagari Vijaya Kumar and Atul Negi, "An attribute partitioning approach to correlation connected clusters", *In Proceedings of International Conference on Advances in Pattern Recognition (ICAPR-2007)*, ISI Kolkata, India, pp. 93-98, Jan. 2-4^{th} 2007.

6. Atul Negi and Kadappagari Vijaya Kumar, "An experimental study of subpattern based principal component analysis and cross-subpattern-correlation based principal component analysis (SubXPCA)", *In Proceedings of Image and Vision Computing Conference (IVCNZ-2005)*, New Zealand, pp. 20-25, Nov. 28^{th}-29^{th} 2005.

7. Kadappagari Vijaya Kumar and Atul Negi, "A review of principal component analysis methods", **submitted to** *Journal of Pattern Recognition Research*, on Jan. 13^{th} 2009.

8. Kadappagari Vijaya Kumar and Atul Negi, "A generalized study of feature partitioning methods to principal component analysis", **submitted to** *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Feb. 2009.

Notation

Dimensionality/Size related

d	:Pattern size or dimensionality
k	:Number of blocks or sub-patterns or sub-images of a pattern
$k.r$	:Dimensionality of a locally-reduced pattern
m, n	:Number of rows and columns of an image pattern (matrix)
N	:Number of training patterns
r	:Locally-reduced sub-pattern size
u	:Block or Sub-pattern size
w	:Globally-reduced pattern size

Pattern/Sub-Pattern Data related

$(\mathbf{A}_i)_{m\times n}$	:i^{th} Image pattern (matrix) of size $m \times n$; $i \in \{1, 2, \ldots, N\}$
$\mathbf{A} = \{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N\}$	:Set of N training image patterns
$(\mathbf{A}_i^j)_{m\times u}$	:j^{th} Sub-image pattern or Block of i^{th} image pattern, $\mathbf{A}_i$
$(\mathbf{B}_i)_{m\times k.r}$	:Locally-reduced image pattern of $(\mathbf{A}_i)_{m\times n}$
$\mathbf{B} = \{\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_N\}$	:Set of locally-reduced image patterns of $\mathbf{A}$
$(\mathbf{D}_i)_{m\times w}$	:i^{th} Globally-reduced image pattern of $(\mathbf{B}_i)_{m\times k.r}$
$\mathbf{D} = \{\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_N\}$	:Set of globally-reduced image patterns of $\mathbf{B}$
$(\mathbf{P}^j)_{N\times u}$	:Set of j^{th} sub-patterns or blocks; $j \in \{1, 2, \ldots, k\}$
$\mathbf{Q}$	:Set of heterogeneous (need not be j^{th} ones) sub-patterns
$(\mathbf{R}^j)_{N\times r}$	:Locally-reduced sub-pattern set of $(\mathbf{P}^j)_{N\times u}$
$x_1,\ x_2,\ \ldots,\ x_d$	:d random feature variables of feature vector $\mathbf{x}$
$\mathbf{x}$	:Feature vector of d random feature variables $x_1,\ x_2,\ \ldots,\ x_d$
$\mathbf{y}$	:Transformed feature vector of $\mathbf{x}$
$(\mathbf{X}_i)_{d\times 1}$	:i^{th} Pattern; $i \in \{1, 2, \ldots, N\}$
$\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\}$	:Set of N training patterns (or objects)
$\mathbf{X^r} = \{\mathbf{X_1^r}, \mathbf{X_2^r}, \ldots, \mathbf{X_N^r}\}$	:Set of N reduced training patterns by classical PCA
$(\mathbf{X}_i^j)_{u\times 1}$	:j^{th} Sub-pattern or Block of i^{th} pattern, $(\mathbf{X}_i)_{d\times 1}$
$(\mathbf{Y}_i)_{k.r\times 1}$	:i^{th} Locally-reduced pattern of $(\mathbf{X}_i)_{d\times 1}$
$(\mathbf{Y}_i^j)_{r\times 1}$	:Locally-reduced sub-pattern of $(\mathbf{X}_i^j)_{u\times 1}$
$\mathbf{Y} = \{\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_N\}$	:Set of locally-reduced patterns (or objects) of $\mathbf{X}$
$(\mathbf{Z_i})_{w\times 1}$	:i^{th} Globally-reduced pattern of $(\mathbf{Y}_i)_{k.r\times 1}$
$\mathbf{Z} = \{\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_N\}$	:Set of globally-reduced patterns of $\mathbf{Y}$

Variance/Covariance/Eigenvector related

$\sigma_{i,i}$ or c_{ii}	:Variance in i^{th} feature
$\sigma_{i,j}$ or c_{ij}	:Covariance between i^{th} and j^{th} features
$(\rho)_{d\times d}$	:Correlation matrix of patterns, $\mathbf{X}$

$(\mathbf{C})_{d\times d}$	:Covariance matrix of patterns, $\mathbf{X}$
$(\mathbf{C}^g)_{(k.r)\times(k.r)}$	:(Inter-block) Covariance matrix of $\mathbf{Y}$
$\mathbf{C}^g_{q,t}$	:Matrix of inter-block covariances of $\mathbf{R}^q$ and $\mathbf{R}^t$
$(\mathbf{C}^j)_{u\times u}$ or $(\mathbf{C}^{j,j})_{u\times u}$	:(Intra-block) Sub-covariance matrix of j^{th} sub-patterns, $\mathbf{P}^j$
$\mathbf{C}^{q,t}$	:(Inter-block) Covariances of sub-patterns, $\mathbf{P}^q$ and $\mathbf{P}^t$, $q \neq t$
$(\mathbf{M})_{n\times n}$	:Image covariance matrix of original images, $\mathbf{A}$
$(\mathbf{M}^g)_{k.r\times k.r}$	:(Inter-block) Image covariance matrix of $\mathbf{B}$
$(\mathbf{M}^j)_{u\times u}$	:(Intra-block) Sub-image covariance matrix of j^{th} sub-images
$(\lambda_i, \mathbf{e}_i)$	:Pair of i^{th} eigenvalue and its corresponding eigenvector
$(\lambda^j_p, \mathbf{e}^j_p)$	:Pair of p^{th} eigenvalue and its eigenvector of $\mathbf{C}^j$, $p = 1, 2, \ldots, u$
$\mathbf{\Lambda}^g$	:Diagonal eigenvalue matrix of $\mathbf{C}^g$
$(\mathbf{E})_{d\times r}$	:Set of first $r\,(< d)$ global eigenvectors of $\mathbf{C}$
$(\mathbf{E})_{n\times r}$	:Set of first $r\,(< n)$ global eigenvectors of $\mathbf{M}$
$(\mathbf{E}^g)_{k.r\times w}$	:Set of first $w\,(< k.r)$ global eigenvectors of $\mathbf{C}^g$ or $\mathbf{M}^g$
$(\mathbf{E}^j)_{u\times r}$	:Set of first $r\,(< u)$ local eigenvectors of $\mathbf{C}^j$ or $\mathbf{M}^j$
$\mathbf{G}$	:Set of all eigenvectors of $\mathbf{C}^g$
$\mathbf{V}$	:Combined matrix of local eigenvectors of all the blocks
	Time complexity related
T_{4C}	:Time complexity of 4C method
T_C	:Time complexity of classical PCA
T_E	:Time complexity of Efficient classical PCA
T_F	:Time complexity of SubXPCA
T_{FP}	:Time complexity of FP-4C method
T_I	:Time complexity of IMPCA or 2DPCA
T_o	:Time complexity of Modular PCA
T_L	:Time complexity of FLPCA
T_M	:Time complexity of SIMPCA
T_S	:Time complexity of SubPCA
	Chapter 7 related
μ	:Minimum number of points or patterns in $N_\epsilon(\ldots)$
δ	:Threshold to select eigenvalues
θ	:Upper bound for correlation dimension
$\mathbf{C^g_O}$	:Covariance matrix with respect to $N_\epsilon(\mathbf{O})$ by FP-4C method
$\mathbf{\bar{C}^g_O}$	:Correlation similarity matrix w.r.t. object $\mathbf{O}$ by FP-4C method
$\mathbf{C_O}$	:Covariance matrix with respect to $N_\epsilon(\mathbf{O})$ by 4C method
$\mathbf{\bar{C}_O}$	:Correlation similarity matrix w.r.t. object $\mathbf{O}$ by 4C method
$\mathbf{CD}$	:List of candidate objects
$\mathbf{CL}$	:Cluster

$DenReach(\mathbf{X}_2, \mathbf{X}_1)$	:Density reachability of $\mathbf{X}_1$ from core point, $\mathbf{X}_2$
$DirCorReach(\mathbf{O}_2, \mathbf{O}_1)$	:Direct correlation reachability of object $\mathbf{O}_1$ from object $\mathbf{O}_2$
$DirReach(\mathbf{X}_2, \mathbf{X}_1)$	:Direct density reachability of $\mathbf{X}_1$ from core point, $\mathbf{X}_2$
$N_\epsilon(\mathbf{X}_i)$	:ϵ-neighbourhood of a point or pattern, $\mathbf{X}_i$
$\lvert N_\epsilon(\mathbf{O})\rvert$	:Number of points or patterns in $N_\epsilon(\mathbf{O})$
$N_\epsilon^{\bar{\mathbf{C}}\mathbf{o}}(\mathbf{O})$	:Correlation ϵ-neighbourhood of object $\mathbf{O}$ by 4C method
$N_\epsilon^{\bar{\mathbf{C}}\bar{\mathbf{g}}\mathbf{o}}(\mathbf{O})$	:Correlation ϵ-neighbourhood of object $\mathbf{O}$ by FP-4C method
$\mathbf{U}$	:Current object selected from the list of candidate objects, **CD**
	Chapter 8 related
c	:Number of classes
h_q	:q^{th} class label, where $q \in \{1, \ldots, c\}$
N_q	:Number of patterns which belongs to class, h_q
$(\mathbf{C}_j^q)_{u\times u}$	:j^{th} Sub-covariance matrix of $\mathbf{P}_j^q$
$(\mathbf{C}^g)^q_{(k.r)\times(k.r)}$	:(Inter-block) Covariance matrix of $(\mathbf{Y}_1)^q, (\mathbf{Y}_2)^q, \ldots, (\mathbf{Y}_{N_q})^q$
$(\mathbf{E}_j^q)_{u\times r}$	:Set of first $r\,(<u)$ local eigenvectors of $\mathbf{C}_j^q$
$(\mathbf{E}^g)^q_{k.r\times w}$	:Set of first $w\,(<k.r)$ global eigenvectors of $(\mathbf{C}^g)^q$
$(\mathbf{P}_j^q)_{N_q\times u}$	:Set of j^{th} sub-patterns of $\{(\mathbf{X}_i)^q\ \forall i = 1, 2, \ldots, N_q\}$ of class h_q
$(\mathbf{R}_j^q)_{N_q\times r}$	:Locally-reduced sub-pattern set of $(\mathbf{P}_j^q)_{N_q\times u}$
T_C^q	:Time complexity of PCA based subspace classifier for class h_q
T_F^q	:Time complexity of FP-SC subspace classifier for class h_q
$(\mathbf{X}_i^j)^q$	:j^{th} Sub-pattern or Block of i^{th} pattern, $(\mathbf{X}_i)^q$, of class h_q
$(\mathbf{X}_i)^q$	:i^{th} pattern of class h_q
$(\mathbf{Y}_i^j)^q_{r\times 1}$	:Locally-reduced sub-pattern of $(\mathbf{X}_i^j)^q_{u\times 1}$
$(\mathbf{Y}_i)^q_{k.r\times 1}$	:i^{th} Locally-reduced pattern of $(\mathbf{X}_i)^q_{d\times 1}$
	Miscellaneous
c	:Number of classes
$\mathbf{f} = \{f_1, f_2, \ldots, f_d\}$	:Features of a pattern $\mathbf{X}_i$; $i \in \{1, 2, \ldots, N\}$
$\mathbf{F}_i$	:a Feature order (a permutation) of $\mathbf{f}$
h_q	:q^{th} class label, where $q \in \{1, \ldots, c\}$
$\mathbf{N}$	:Set of natural numbers
$\Re$	:Set of real numbers
$g_i(\mathbf{x})$	:Discriminant function with respect to class h_i, $i = 1, 2, \ldots, c$
H	:Holistic PCA
$\mathbf{I}_{d\times d}$ or $\mathbf{I}_d$	:Identity matrix of size $d \times d$
$(\mathbf{0})_{r\times r}$	:Zero matrix of size $r \times r$
T1	:Feature Partitioning PCA Type-I method
T2	:Feature Partitioning PCA Type-II method
T3	:Feature Partitioning PCA Type-III method
T4	:Feature Partitioning PCA Type-IV method

Abbreviations

2DLDA	:Two-Dimensional LDA
2DPCA	:Two Dimensional Principal Component Analysis
$(2D)^2$PCA	:Two-directional 2DPCA
4C	:Computing Clusters of Correlation Connected objects
AA-MLP	:Auto-Associative Multi Layer Perceptron
AA-NN	:Auto-Associative Neural Network
ALA	:Adaptive Learning Algorithm
AMD	:Assembled Matrix Distance
ANOVA	:Analysis of Variance
ANN	:Artificial Neural Network
APEX	:Adaptive Principal Component Extractor
Aw-SpPCA	:Adaptively weighted PCA
B2DPCA	:Bilateral Projection based 2DPCA
BDPCA	:Bi-directional PCA
CCC	:Cumulative Correlation Coefficient
CLAFIC	:Class-Featuring Information Compression
DBSCAN	:Density-Based Spatial Clustering of Applications of Noise
DiaPCA	:Diagonal PCA
DWT	:Discrete Wavelet Transform
EM	:Expectation Maximization
EVD	:Eigen Value Decomposition
FAR	:False Acceptance Ratio
FD	:Fourier Descriptor
FLD	:Fisher Linear Discriminant
FLPCA	:FLexible image Principal Component Analysis
FP-4C	:Feature Partitioning Approach to 4C
FP-PCA	:Feature Partitioning based PCA
FP-PCA-Type-I	:Feature Partitioning PCA Type-I method
FP-PCA-Type-II	:Feature Partitioning PCA Type-II method
FP-PCA-Type-III	:Feature Partitioning PCA Type-III method
FP-PCA-Type-IV	:Feature Partitioning PCA Type-IV method
FP-SC	:Feature Partitioning approach to Subspace Classification
FRR	:False Rejection Ratio
GA	:Genetic Algorithm
GDA	:Generalized Discriminant Analysis
GHA	:Generalized Hebbian Algorithm
HANN	:Hebbian type ANN
HPCA	:Holistic PCA

ICA	:Independent Component Analysis
IMPCA	:IMage Principal Component Analysis
K2DPCA	:Kernel version of 2DPCA
KPCA	:Kernel PCA
LDA	:Linear Discriminant Analysis
MatPCA	:Matrixized version of PCA
MLP	:Multi Layer Perceptron
modPCA	:Modular PCA
MSE	:Mean Squared Error
MSS	:Multiple Similarity Method
NLNN	:Non Linear Neural Network
NLPCA	:Non Linear PCA
PC	:Principal Component
PCA	:Principal Component Analysis
PR	:Pattern Recognition
PV	:Projection Vector (Eigenvector selected for projection)
RT	:Rough Set Theory
SGA	:Stochastic Gradient Ascent
SHPCA	:Sub-Holistic PCA
SIMPCA	:Sub-IMage Principal Component Analysis
SN	:Subspace Network
SSS	:Small Sample Size
SubPCA	:Sub-pattern based PCA
SubXPCA	:Cross-Sub-pattern correlation based PCA
SVD	:Singular Value Decomposition
SVM	:Support Vector Machine
TER	:Total Error Rate
TS	:Tabu Search
WSA	:Weighted Subspace Algorithm
WT	:Wavelet Transform

Chapter 1

Introduction

In this chapter, we give the background and motivation of our investigation. We present a brief review of pattern recognition, feature extraction (dimensionality reduction) and Principal Component Analysis (PCA) to setup the context of this work. At the end of this Chapter we summarize our contributions that are to come in the following chapters.

1.1 A Brief View of Pattern Recognition

Pattern Recognition (PR) is the study of how machines can observe the environment, learn to distinguish patterns of interest from their background, and make sound and reasonable decisions about the categories of the patterns [72]. Given a pattern (e.g. a face image or a character or a fingerprint image), its recognition or classification may consist of one of the following two tasks [173]: (i) Supervised classification (e.g. Nearest Neighbour Rule), in which an input pattern is identified as

a member of a predefined class. (ii) Unsupervised classification (e.g. Clustering), in which an input pattern is assigned to an unknown class. In many of the applications, it is obvious that no single approach for classification is optimal and that multiple approaches have to be used. Consequently combining several approaches is one of the commonly used practices in pattern recognition.

Pattern Recognition systems [144] often play a vital role in machine intelligence systems and are used for both data processing and decision making. In broad terms, pattern recognition is the science that concerns the description or classification (recognition) of measurements (patterns). Pattern recognition systems overlap with other areas such as Signal Processing, Artificial Intelligence, Neural Modelling, Optimization/Estimation theory, etc. PR applications include image processing, segmentation and image analysis, computer vision, seismic analysis, radar signal classification, biometric identification (e.g. face, fingerprint), speech recognition or understanding, character recognition and handwriting analysis.

Pattern Recognition may be classified as an information reduction, information mapping or information labelling process. The structure of a typical PR system is shown in Fig. 1.1. The PR system consists of a sensor (e.g. camera), a feature extraction mechanism/algorithm (e.g. Principal Component Analysis) and a classification or descriptive algorithm (e.g. k-Nearest Neighbour method). Using camera/transducer, we measure world pattern data, which may contain noise, measurement error, etc (for e.g. camera may capture a face of a person in a rainy day or sunny day or in overcast weather in a typical surveillance system). Further, the measured data captured by transducer at different scenarios is required to be preprocessed

to make it suitable for pattern recognition tasks (for e.g. enhancing or cropping or segmenting the face image). The preprocessed pattern may be of high dimensionality. It is well known that high-dimensional data demands high computational and storage requirements. In addition, classification performance may come down with increase in dimensionality due to *curse of dimensionality* phenomenon. Therefore, there is a need of feature extraction algorithms, which reduce dimensionality and also extract a few salient features useful for better classification. Next, classification algorithms such as Nearest Neighbour method, Bayes classifier, Support Vector machines, Clustering algorithms, etc, make use of the salient features to classify a given pattern. In general, we assume some data is available which is already classified or described to train the system. Such data is called as *training data*.

Recognition depends upon the concept of a *pattern*. Generally, a pattern can be described as a set of measurements or observations and may be represented in vector or matrix form. The measurements could be items such as height, weight, color, etc. In addition, patterns may be converted from one representation to another. Some examples of patterns are shown in Fig. 1.2. In broad terms, *features* represent any extractable measurement (e.g. pixel intensities of a face image). Features may be symbolic, numeric or both (e.g. height, color, texture). Features may also be obtained from applying a *feature extraction or feature selection algorithm* (e.g. PCA or LDA) to the input data. Such algorithms may need significant computational effort and the extracted features may contain errors or noise. Features may be represented by continuous, discrete or discrete-binary variables. The key issue of feature extraction/selection algorithms is to extract or choose features that (i) are computationally

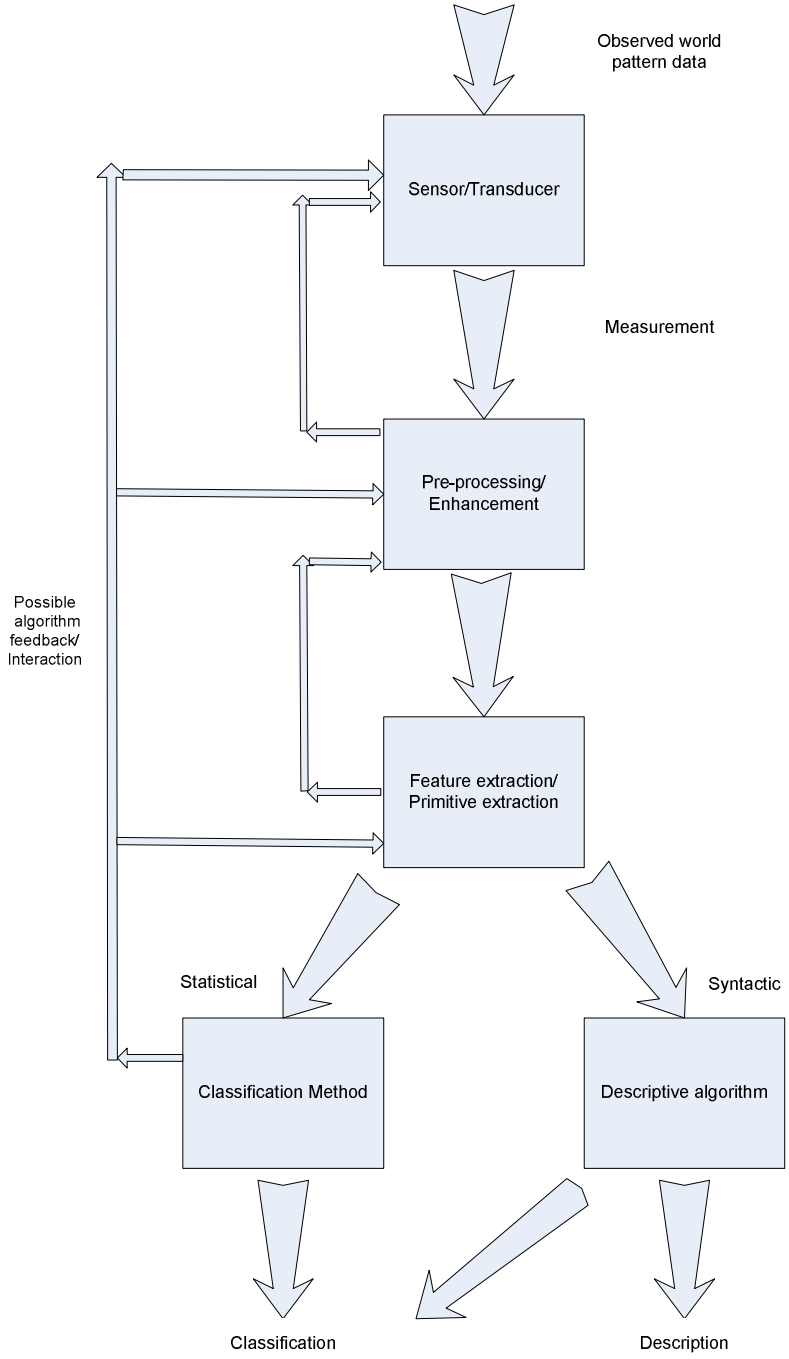


Figure 1.1: Typical pattern recognition system structure [144]

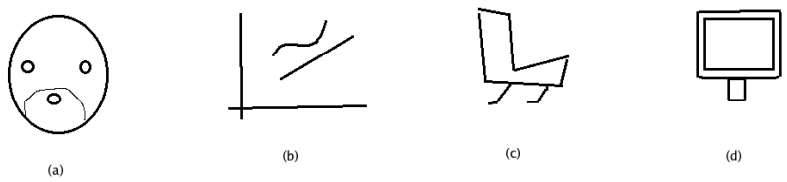


Figure 1.2: *Examples of visual patterns.* (a) a Face (b) a Line drawing (c) a Chair (d) a Computer monitor

Figure 1.3: Rotations of pattern 'A'

feasible, (ii) lead to good PR system success (i.e. in terms of high classification rate), (iii) reduce the problem data (raw measurements of patterns) into a significant amount of information without discarding vital information.

In general, we wish to have classification or recognition of a pattern that is *invariant to some changes or deviations* in the pattern from the ideal case. One of the causes for such deviations is noise. In many situations, a set of patterns from a single class show wide variations. For example, a character from English alphabet (Figs. 1.3-1.6). Classifying such characters involves feature analysis of each character. Some examples of *pattern distortions* are shown in Figs. 1.3-1.6. Therefore, a careful choice of invariant features (e.g. with respect to Rotation, Scale and Translation (RST)) and pattern structure is to be used for recognition. For example, RST-invariant moment features may be used for shape recognition application.

Figure 1.4: Scaling of pattern ‘A’

Figure 1.5: Variants of pattern ‘A’

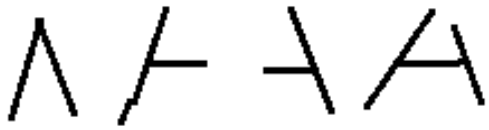


Figure 1.6: Some occlusions of pattern ‘A’

The goal of pattern recognition or classification is to distinguish between different classes of patterns, hence it is good to find the basis for this *discriminative ability*. One obvious answer we can give is that patterns of different classes are composed of features with different numerical values. However, this may not be always true. For example, two halftone pictures, say car and bus, comprise of same black and white dots (features). However, spatial arrangements of those two pictures are different and may be used for classification.

Features are arranged in a d-dimensional *feature vector*, denoted by $\mathbf{x}$, which gives multi-dimensional feature space. If each feature is a real number, the feature space is $\Re^d$ i.e. hypercube. Quite often classification is carried out by partitioning feature space into regions for each class. If dimensionality of a feature vector is large (e.g. an image vector of size 300×300), the feature extraction techniques (e.g. PCA) play a vital role in dimensionality reduction (Fig. 1.7(a)). A classifier partitions *feature space* into class-labelled *decision regions*. Decision regions must cover feature space and be disjointed (one exception to this is the notion of fuzzy sets). With these ideas, classification of a feature vector, $\mathbf{x}$, is done as follows: we find the decision region (in $\Re^d$) into which $\mathbf{x}$ falls and assign $\mathbf{x}$ to this class. Although the classification principle looks simple, the determination of decision regions is a challenging task. We can also come out with a number of classifiers based on *discriminant functions*. In the c-class case, discriminant functions, denoted by $g_i(\mathbf{x})$, $i = 1, \ldots, c$ are used to partition $\Re^d$ as follows.

Decision Rule: Assign $\mathbf{x}$ to class h_m (Region $\mathbf{R_m}$), where $g_m(\mathbf{x}) > g_i(\mathbf{x})$ $\forall i = 1, \ldots, c$ and $i \neq m$.

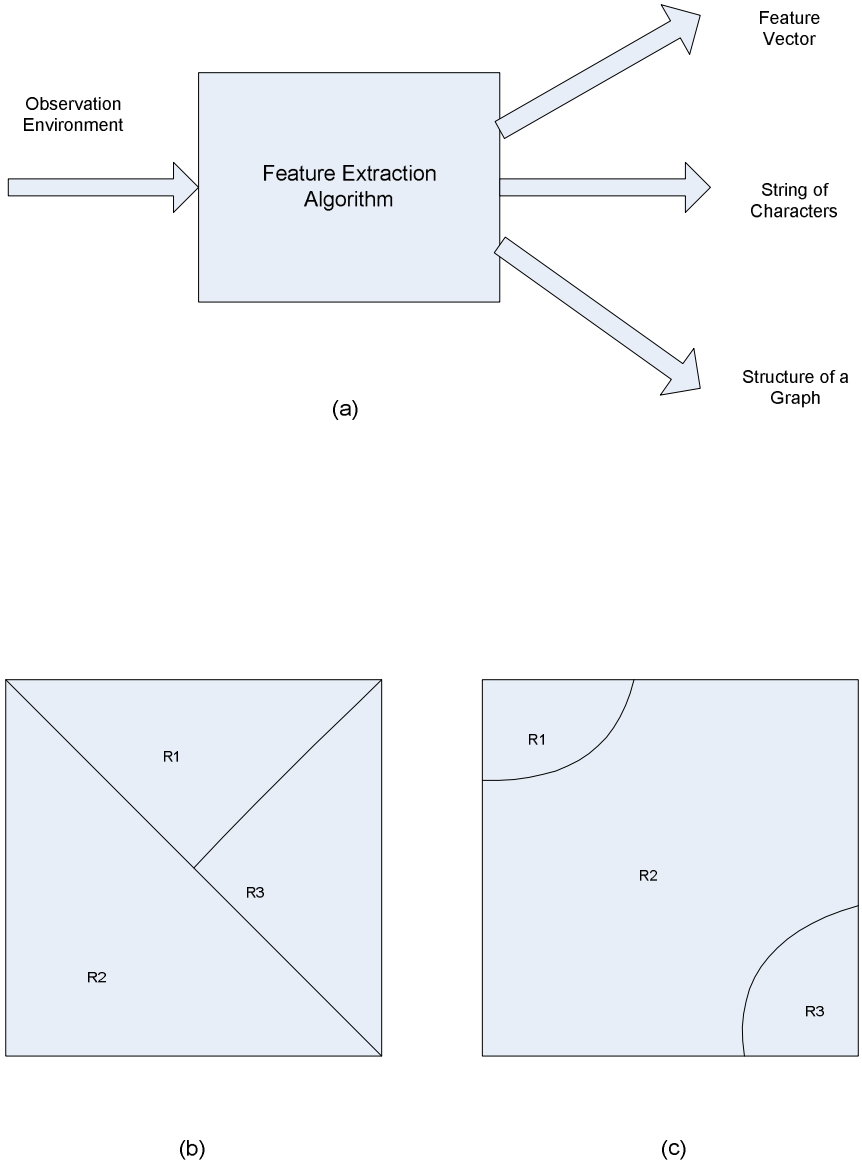


Figure 1.7: (a) Feature extraction process in pattern recognition (b) Piecewise (Linear) decision regions. (c) Hyperbolic (Quadratic) decision regions.

It is obvious that $g_k(\mathbf{x}) = g_l(\mathbf{x})$ defines a decision boundary. Fig. 1.7(b) shows some sample decision regions.

To get the best achievable performance for a PR system, it is good to use maximum amount of information available (*a priori information*) such as sample patterns with known class origin. A set of typical patterns, where salient attributes or the class label of each is known, forms a training set, $\mathbf{X}$. In general, the training set provides significant information required to associate input data with class labels. The training set is used to make the system to learn relevant information. *Supervised learning* assumes a labelled training set, $\mathbf{X}$ (i.e. patterns with class labels), where as in *unsupervised learning*, the class labels of $\mathbf{X}$ are not known and the system must determine natural groups of training data. Each group is named with a class label.

We need to use appropriate PR approaches for the application under consideration. The *Statistical PR* (StatPR) approach may be used if there is an underlying and quantifiable statistical basis for the generation of patterns. In StatPR, features are assumed to be generated by a state of nature and therefore the underlying model is of a state of nature or class-conditioned set of probabilities and/or probability density functions. The *Syntactic PR* (SyntPR) approach may be used if interrelationships or interconnections of features give some important information, which facilitates structural description or classification (e.g. musical pattern analysis, speech recognition, handwriting analysis, etc). However, in SyntPR we must be able to quantify and extract structural information and to assess structural similarity of patterns. In some other cases, where neither statistical basis of patterns nor structural description is available, we may use other PR approaches, for example *Neural Network based PR*

may be suitable for pattern association applications. That is we can train a Neural Network to correctly associate input patterns with desired classes [144]. The main difference between neural network and other approaches to pattern recognition are that these networks have the ability to learn complex non-linear input-output relationships and use sequential training procedures. In addition, neural networks have the general characteristic of adapting themselves to the data. The most commonly used family of neural networks for pattern classification tasks is the feed forward networks such as Multi Layer Perceptron (MLP), Radial Basis Function (RBF) networks [73] and Self-Organizing Map (SOM) [86] (which is mainly used for data clustering and feature mapping). Neural networks provide unified approaches for feature extraction and classification and flexible procedures for finding good, moderately nonlinear solutions. *Template matching* is another simple and well known approach to pattern recognition. In template matching a 'template' or an ideal prototype of the pattern to be recognized is available. The pattern to be recognized is matched against the stored templates while taking into account all allowable translation, rotation and scale changes [72]. Fu introduced the notion of attributed grammars which unifies Syntactic and Statistical Pattern Recognition approaches [48].

1.2 An Overview of Feature Extraction (Dimensionality Reduction)

Feature extraction is an important stage of pattern recognition (Figs. 1.1 and 1.7(a)). Extraction of features is an important step and strongly influences classifier

design. That is, if the extracted features show significant differences from one class to another, the classifier can be designed more easily with better performance. Therefore, the extraction of features is a key issue in pattern recognition.

Feature extraction is generally considered as a process of mapping the original features (measurements) into more effective features. If the mapping is linear, the mapping function is well defined and the task is simply to find the coefficients of a linear function so as to maximize or minimize a criterion. To determine these mapping coefficients, we can use the linear algebra techniques for simple criteria and we can apply optimization techniques for complex criteria. Unfortunately, in many applications of pattern recognition, there are salient features which are non-linear mappings of original measurements. Since there is no general theory to generate such nonlinear functions systematically and find the optimum one, extraction of features becomes problem-specific.

In large multi-dimensional data sets, it is usually advantageous to discover some structure from the data. Thus we assume that the data are governed by a certain number of underlying parameters (features). The minimum number of features required to account for the observed properties of the data is called *intrinsic dimensionality* of the data set. Geometrically, the entire data lies on a topological hyperspace of dimensions equal to intrinsic dimensionality [49]. Some recent efforts to compute intrinsic dimensionality may be found in [124] [180] [28].

An ideal feature extraction technique yields a set of features that makes the job of the classifier trivial. The goal of the feature extraction technique is to characterize a pattern to be recognized by measurements (features) whose values are very similar

for patterns in the same class and very different for patterns in different classes. This leads to the problem of finding distinguishing features that are invariant to transformations (e.g. Rotation, Scale, Translation) of the input pattern (Figs. 1.3–1.6). In speech recognition, we want the features that are invariant to translations in time and to changes in the overall amplitude. We may also want the features that are sensitive to the duration of the word, i.e. invariant to the rate at which the pattern evolves. Rate variation is a serious problem because even the same person speaks at different rates causing the speech signal to change in complex ways. Similarly cursive handwriting also varies as the writer speeds up.

The feature extraction may be domain-dependent and thus requires knowledge of the domain. A good feature extractor for fingerprint classification, may not be useful for face recognition or palmprint recognition.

From our experience, we know that classification of patterns as done by human beings is based on a very few of the important attributes (features). Similarly we attempt to design PR systems on the basis of a few significant features characterizing the class membership of the patterns, preferably those that would be used by humans for classification. The main motivation for keeping the number of pattern dimensions to the absolute minimum is to curtail the effect of the *curse of dimensionality* phenomenon (i.e. the number of feature points is the exponential function of feature dimension [11]) on the complexity of the classifier. It is obvious that probability of misclassification does not increase with the number of features increased provided the class-conditional densities are completely known (or equivalently the number of training data is large and representative of underlying densities). However in prac-

tice the performance of classifier degrades as the number of features increases if the number of features is relatively smaller than number of training samples [131][132]. The difficult task is to find out significant features for classification among available features. This problem may be increased if we do not know the process by which patterns are generated. To alleviate this problem, it is necessary to use some feature extraction technique (transducer) which is able to retain as much information about patterns as possible. Some recent work on to avoid curse of dimensionality may be found in [3] [8] [155].

It is well known that often patterns contain a large number of features (e.g. a face image of 300×400 size consists of 120000 features). These patterns may contain features which are redundant or irrelevant to the classification task. Moreover, the pattern generating mechanism and the feature extraction techniques (transducers) are likely to introduce some distortion and noise, in addition to natural pattern variability. Therefore the fundamental task of feature extraction technique is to extract most useful information from the original pattern and present it in a form of a lower dimensionality vector, whose components represent the most significant features of input pattern. The goal of feature extraction technique is not merely dimensionality reduction, but to remove any redundant and irrelevant features which may reduce the classifier performance. Further role of the feature extraction process is to establish whether it is necessary to seek additional features which would contain discriminatory information allowing the improvement of classifier performance.

Most feature extraction techniques compress the observed information into a lower dimensional space to facilitate its transmission or storage or classification. Here the

elimination of the irrelevant information and redundancy (using feature extraction techniques) is an integral part of the transformation which maps original pattern vector $\mathbf{x}$ into $\mathbf{y}$, the transformed vector, a new lower dimensional feature space which is given by

$$\mathbf{y} = f(\mathbf{x}) \tag{1.1}$$

The mapping $f(\ldots)$ is obtained by optimizing a criterion function $J(\ldots)$.

As the number of dimensions increases the generalization ability is likely to come down for finite training data. Feature extraction may be seen as the one which improves the generalization ability of the recognition system by keeping number of dimensions at minimum. Therefore its performance on unknown patterns is a trade-off error probability for estimation errors. The benefits of dimensionality reduction by feature extraction may be limited when we restrict the form or criterion of mapping. At times we implicitly assume an over-simplistic model of the pattern recognition system. For example, if the classes are not linearly separable and mapping of feature extractor is restricted to a linear form [35].

Some commonly used feature extraction techniques include Principal Component Analysis (PCA), Linear Discriminant Analysis [49], Independent Component Analysis [15][26][97], Projection Pursuit [47], Random Projections [33][43]. A good overview of feature extraction techniques may be found in Refs. [16][72].

Here we shall focus on PCA henceforth in this work.

1.3 Classical Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is one of the widely used techniques for dimensionality reduction with widespread applications to data reduction, image processing, visualization, pattern recognition, exploratory data analysis, etc, [76][35][39][75]. PCA is known by different names *Karhunen-Loeve transform*, *Hotelling transform*, *Signal Subspace* or *Eigenstructure approach*. In pattern recognition, PCA is used in various forms for optimal feature extraction and data compression [35]. In image processing PCA defines the Hotelling or KL transform that is applied in image data compression [71]. In signal processing, a useful characterization of signals is to assume that they roughly lie in signal subspace defined by PCA. Several methods of signal modelling, spectrum estimation, and array processing are based on this concept [161].

PCA is concerned with explaining the *variance-covariance structure* through a few linear combinations of the original features. Although d components are required to reproduce the total system variability, often much of this variability can be accounted for by a small number, $r\,(< d)$, of the principal components (PCs) from the original d features. The N measurements with r PCs are used to replace original N measurements with d features [75].

PCA is used as an intermediate step in many investigations such as cluster analysis or multiple regression. Algebraically, PCs are particular linear combinations of d random variables (features), $x_1, x_2, \ldots, x_d$. Geometrically these linear combinations represent the selection of a new coordinate system obtained by rotating the original system with $x_1, x_2, \ldots, x_d$ as coordinate axes. The new axes represent the directions with maximum variability and provide a simple description of covariance structure.

PCs depend solely on the covariance matrix, $\mathbf{C}$ (or the correlation matrix, ρ) of $x_1, x_2, \ldots, x_d$. Computation of PCs does not require a multivariate normal distribution assumption. Let the random vector $\mathbf{x} = [x_1, x_2, \ldots, x_d]^T$ have the covariance matrix $\mathbf{C}$ with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d \geq 0$.

Consider the linear combinations

$$y_1 = \mathbf{l}_1^T.\mathbf{x} = l_{11}.x_1 + l_{12}.x_2 + \ldots + l_{1d}.x_d \tag{1.2}$$

$$y_2 = \mathbf{l}_2^T.\mathbf{x} = l_{21}.x_1 + l_{22}.x_2 + \ldots + l_{2d}.x_d \tag{1.3}$$

$$\vdots \quad \vdots \quad \vdots \quad \vdots$$

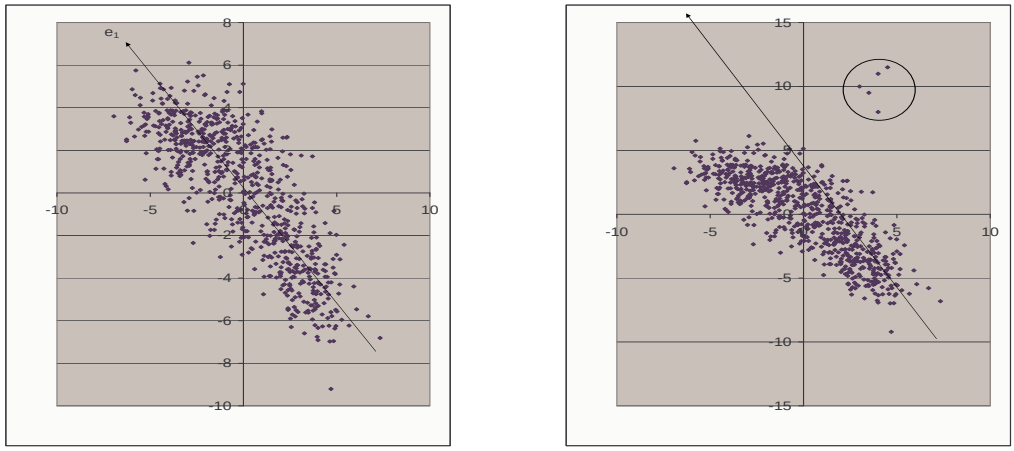
$$y_d = \mathbf{l}_d^T.\mathbf{x} = l_{d1}.x_1 + l_{d2}.x_2 + \ldots + l_{dd}.x_d \tag{1.4}$$

where $\mathbf{l}_i = [l_{i_1}, l_{i_2}, \ldots, l_{i_d}]^T$ is a coefficient vector. Then the variance (Var) and covariance (Cov) of these linear combinations are given as follows. Here $E(\ldots)$ indicates expectation function.

$$Var(y_i) = E[\mathbf{l}_i^T.\mathbf{x}.\mathbf{x}^T.\mathbf{l}_i] = \mathbf{l}_i^T.E[\mathbf{x}.\mathbf{x}^T].\mathbf{l}_i = \mathbf{l}_i^T.\mathbf{C}.\mathbf{l}_i\,; i = 1, 2, \ldots, d \tag{1.5}$$

$$Cov(y_i, y_j) = E[\mathbf{l}_i^T.\mathbf{x}.\mathbf{x}^T.\mathbf{l}_j] = \mathbf{l}_i^T.E[\mathbf{x}.\mathbf{x}^T].\mathbf{l}_j = \mathbf{l}_i^T.\mathbf{C}.\mathbf{l}_j;\ i, j = 1, 2, \ldots, d \tag{1.6}$$

The PCs are those uncorrelated linear combinations $y_1, y_2, \ldots, y_d$ (eqs. 1.2-1.4) whose variances (eq. (1.5)) are as large as possible. It is clear that variance specified in eq. (1.5) can be increased by multiplying $\mathbf{l}_i$ by some constant. Thus to eliminate this problem it is good to restrict the length of coefficient vectors (eigenvectors) to unity. Therefore we can enumerate:



(a) Correlations without Outliers data. (b) Correlations with Outliers data.

Figure 1.8: *Effect of outliers in Waveform data [165].* (a) The eigenvector in the direction of maximum variance (b) Outliers data are circled. Observe that the eigenvector is pulled towards outliers thus spoiling the direction of maximum variance.

First Principal Component = linear combination $\mathbf{l}_1^T.\mathbf{x}$ that maximizes $Var(\mathbf{l}_1^T.\mathbf{x})$ subject to $\mathbf{l}_1^T.\mathbf{l}_1 = 1$,

Second Principal Component = linear combination $\mathbf{l}_2^T.\mathbf{x}$ that maximizes $Var(\mathbf{l}_2^T.\mathbf{x})$ subject to $\mathbf{l}_2^T.\mathbf{l}_2 = 1$ and $Cov(\mathbf{l}_1^T.\mathbf{x}, \mathbf{l}_2^T.\mathbf{x}) = 0$, and so,

at the i^{th} step,

i^{th} Principal Component = linear combination $\mathbf{l}_i^T.\mathbf{x}$ that maximizes $Var(\mathbf{l}_i^T.\mathbf{x})$ subject to $\mathbf{l}_i^T.\mathbf{l}_i = 1$ and $Cov(\mathbf{l}_i^T.\mathbf{x}, \mathbf{l}_j^T.\mathbf{x}) = 0$, $\forall j < i$.

A sample PC direction is shown in Fig. 1.8(a). Let $\mathbf{C}$, the covariance matrix associated with random feature vector $\mathbf{x} = [x_1, x_2, \ldots, x_d]^T$, have the eigenvalue-eigenvector pairs $(\lambda_1, \mathbf{e}_1)$, $(\lambda_2, \mathbf{e}_2)$,..., $(\lambda_d, \mathbf{e}_d)$, where $\lambda_1 \geq \lambda_2, \ldots, \lambda_d \geq 0$. Then the

i^{th} PC is given by

$$y_i = \mathbf{e}_i^T.\mathbf{x} = e_{i1}.x_1 + e_{i2}.x_2 + \ldots + e_{id}.x_d;\ i = 1, 2, \ldots, d \tag{1.7}$$

The variance summarized by i^{th} PC, y_i, is given by

$$Var(y_i) = E[\mathbf{e}_i^T.\mathbf{x}.\mathbf{x}^T.\mathbf{e}_i] = \mathbf{e}_i^T.E[\mathbf{x}.\mathbf{x}^T].\mathbf{e}_i = \mathbf{e}_i^T.\mathbf{C}.\mathbf{e}_i = \lambda_i\,; i = 1, 2, \ldots, d \tag{1.8}$$

and the covariance between PCs, y_i and y_j is given by

$$Cov(y_i, y_j) = 0,\ \forall i, j = 1, 2, \ldots, d; i \neq j \tag{1.9}$$

If some eigenvalues are equal, the choices of corresponding eigenvectors, $\mathbf{e}_i$ and hence y_i, are not unique. It is also true that the sum of variance of original variables (features), $x_1, x_2, \ldots, x_d$, is equal to the sum of eigenvalues. In other words,

$$\sigma_{11} + \sigma_{22} + \ldots + \sigma_{dd} = \sum_{i=1}^{d} Var(x_i) = \lambda_1 + \ldots + \lambda_d = \sum_{i=1}^{d} Var(y_i) \tag{1.10}$$

Each component of the coefficient vector (eigenvector), $\mathbf{e}_i = [e_{i1}, e_{i2}, \ldots, e_{id}]^T$ needs to be inspected. The magnitude of e_{ip} measures the importance of the p^{th} variable to the i^{th} PC, irrespective of other variables (features). e_{ip} is proportional to the correlation coefficient between y_i (i.e. i^{th} PC) and x_p (i.e. p^{th} variable). In other words,

$$\rho_{y_i, x_p} = \frac{(e_{ip}.\sqrt{\lambda_i})}{\sqrt{\sigma_{pp}}}\,; i, p \in \{1, 2, \ldots, d\} \tag{1.11}$$

Also it was observed that eigenvalues and eigenvectors obtained from covariance matrix ($\mathbf{C}$) are, in general, not the same as the ones derived from correlation matrix (ρ). Further, PCs derived from $\mathbf{C}$ are not a simple function of the ρ. Variables (features) should probably be standardized if they are measured on scales with widely

differing ranges or if the measurement units are not commensurable. For example, if a variable x_1 represents annual sales in the range Rupees 1 million to 10 millions and x_2 is the ratio (net annual income)/(total assets) that falls in the range 0.01 to 0.6, then the total variation will be just due to sales variable, x_1. Here first PC shows weighting heavily x_1 and weight for variable x_2 is quite negligible. Alternatively, if both variables are standardized, their subsequent magnitudes are of the same order and both the variables, x_1 and x_2, may have significant weights in the construction of the PCs. When attempting to interpret subject matter of the PCs (i.e. to interpret a PC in terms of variables), the correlations, ρ_{y_i,x_p} (eq. (1.11)) may be more reliable guides than the PC coefficients, e_{ip}. The correlations allow for differences in the variances of the original variables and therefore avoid the interpretive problem caused by different measurement scales.

An unusually small eigenvalues from either covariance or correlation matrix may indicate an unnoticed linear dependency in the data set. If this is the case, one or more of the variables (features) is redundant and should be deleted. For example, let x_1, x_2, x_3 be student scores in various subjects and the total score, x_4, is the sum $x_1 + x_2 + x_3$. Then even though the linear combination $\mathbf{e}^T.\mathbf{x} = [1, 1, 1, -1].\mathbf{x} = x_1 + x_2 + x_3 - x_4$ is always *zero*, rounding error in the computation of eigenvalues may lead to a small nonzero value. If the linear expression (dependency) $x_4 = x_1 + x_2 + x_3$ was unnoticed, the smallest eigenvalue-eigenvector pair provide a hint to its existence. Thus although large eigenvalues and associated eigenvectors are important in PCA, eigenvalues very close to *zero* should not be overlooked in a routine way. The eigenvectors associated with these latter eigenvalues may reveal linear dependencies in

the data set that can cause interpretive and computational problems in a subsequent analysis [75].

1.3.1 How to Perform PCA on a Given Set of Patterns?

Consider $\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\}$, the set of N training patterns of dimensionality d. The steps of classical PCA are given as follows [120].

1. Perform a 'mean subtraction' operation upon the training data patterns.

2. Calculate the Covariance Matrix:

$$(\mathbf{C})_{d\times d} = \frac{1}{N}.\sum_{i=1}^{N}[\mathbf{X}_i - \bar{\mathbf{X}}].[\mathbf{X}_i - \bar{\mathbf{X}}]^T \tag{1.12}$$

   where $\bar{\mathbf{X}}$ is the mean of training patterns, $\mathbf{X}$.

3. Determine eigenvalues, λ and eigenvectors, $\mathbf{e}$ of the covariance matrix, $\mathbf{C}$ such that

$$(\mathbf{C})_{d\times d}.(\mathbf{e})_{d\times 1} = (\mathbf{e})_{d\times 1}.(\lambda)_{1\times 1} \tag{1.13}$$

   If the rank of $\mathbf{C}$ is d then one can find d eigenvalues.

4. Sort the eigenvalues and the corresponding eigenvectors in non-increasing order.

5. Choose first $r\,(< d)$ column eigenvectors, (denoted by $\mathbf{E}_{d\times r}$) and project the data set $\mathbf{X}$ on $\mathbf{E}_{d\times r}$ to get the reduced data $\mathbf{X^r}$.

$$\mathbf{X^r}_{N\times r} = \mathbf{X}_{N\times d}.\mathbf{E}_{d\times r} \tag{1.14}$$

The original data can be reconstructed with most of the variance (information) from compressed or reduced data $\mathbf{X^r}_{N\times r}$ as given by

$$\hat{\mathbf{X}}_{N\times d} = \mathbf{X^r}_{N\times r}.\mathbf{E}^T_{r\times d} \tag{1.15}$$

where $\hat{\mathbf{X}}_{N\times d}$ is reconstructed data.

1.3.2 Why is PCA so Popular in Dimensionality Reduction?

It is observed that the use of PCA is widespread for dimensionality reduction. Here we enumerate some reasons:

- PCA is a global scheme because it considers covariances (correlations) between every pair of original d variables (features). Therefore it is highly effective in dimensionality reduction when global variations are prominent.

- PCA is an optimal linear scheme (in terms of mean squared error) for compressing a set of high dimensional vectors into a set of low dimensional vectors and for reconstructing the original vectors (Section 1.3).

- The model parameters (eigenvectors and eigenvalues) can be computed directly from the data i.e. by diagonalizing the sample covariance (eq. 1.13).

- Compression and Decompression are easy operations and they require only matrix multiplications (eqs. 1.14-1.15).

- Further, PCA minimizes representation entropy and it makes output variables (PCs) mutually uncorrelated [79].

1.3.3 Fundamental Problems/Issues with Classical PCA

Although PCA is popular in its application, there are several issues which inhibit an analyst. Here we describe a few of them based upon our study.

1. *High computational complexity.* The classical PCA methods are not suitable for high dimensional data (d) or large number of data points (N), because PCA needs high computational requirements for such data in particular for huge image data. Computing the sample covariance matrix or correlation matrix itself is expensive ($O(N.d^2)$). To understand the severity of the problem, consider 50 training images, each of size 100×100 pixels, that is $d = 100.100 = 10^4$. The time complexity to calculate covariance matrix is $O(50.10^4.10^4)$, which is quite expensive [103][176]. The problem is multiplied with bigger images. Now the question arises whether one can avoid computation of sample covariance matrix explicitly or any other means to reduce the time complexity.

2. *Poor performance (that is less generalization ability) with data of prominent local variations.* PCA is a global feature extraction technique which may perform well when *global* variations among patterns are dominant (e.g. screaming face expression spreads across entire face), however it may not perform well when *local* variations among patterns are dominant (e.g. smile of a face is limited to mouth region) [53]. Can we find a way which extracts *both local and global features* to adapt to different scenarios, which can improve classification performance?

3. *Small Sample Size (SSS) problem.* PCA may not be good to reduce dimensions in the case of a Small Sample Size (SSS) problem. That is when number of training samples are less as compared to number of features or dimensions, PCA may not effectively perform dimensionality reduction or feature extraction. Therefore SSS problem has to be resolved before extraction of principal components [189]. How to resolve SSS problem? [131][132]

4. *Not suitable to handle missing data.* Another problem with classical PCA approaches is that it is not known how to deal properly with incomplete data set, in which some feature values are missing. The incomplete points are either discarded or completed using some interpolation methods. However, such approaches may not be useful if a significant part of measurements are unknown [18][62][101].

5. *Not suitable to handle outliers data.* In general, training data may contain some errors or noise from the underlying data generation method. Such data objects that distract the discovery of the underlying model are called as *outliers*. The classical PCA algorithm is based on the assumption that the data have not been spoiled by outliers. However in practice, outliers do exist in the data and can divert the principal components towards them and away from the direction of maximum variance, thus spoiling the projected data (Figs. 1.8(a) and 1.8(b)) [18][30].

6. *Not good for non-linear data.* Further, Classical PCA is suitable in applications where the underlying structure is linear. The linear PCA either needs more

principal components or unsuitable for the data sets where nonlinear structure is present [145][24].

7. *Choosing right number of principal components.* Right choice of principal components influence classifier performance as well as the total amount of variance (structure) in the reduced data [76][123]. Now the question arises as to how to choose right number of Principal Components?

In the next chapter, we discuss the state-of-the-art PCA methods existing in the literature to address the problems faced by classical PCA.

1.4 Summary of Contributions

Our ideas/contributions are layed out across the following chapters in this thesis. In this section we summarize all our ideas for ready reference. Our work focus on solving the problems of (i) high time complexity with high dimensional data, (ii) low generalization ability when either local variations or global variations among the patterns are dominant and (iii) SSS problem, related to classical PCA and other recent PCA methods. In addition, our approach retains the near optimality in terms of summarization of variance, which leads to near-optimal compression and reconstruction of the patterns.

First of all, we adopt a feature partitioning approach to PCA computation. Here a novel framework with a general scope is proposed for feature partitioning based PCA (FP-PCA) approaches. In this framework, each pattern is divided into $k\,(\geq 2)$ sub-patterns (blocks). Then we extract local features from these sub-patterns using

Principal Component Analysis technique. We call these features *local* because they are extracted from the sub-patterns, but not from whole patterns. Finally all these local features are combined by a systematic procedure.

Next, we analyze the generalized feature partitioning framework and bring out several fundamental issues to be addressed, which arise due to partitioning of patterns. These issues include (i) how to divide a given pattern into sub-patterns?, (ii) how to choose sub-pattern size or number of sub-patterns?, (iii) impact of feature ordering of patterns, (iv) how to combine locally-extracted features from sub-patterns?, (v) impact of overlapping features across sub-patterns, (vi) how to choose principal components from each of the sub-patterns?, (vii) impact of truncation of features in the last sub-pattern if its size is less than other sub-patterns, (viii) impact of loss of inter-sub-pattern correlations or covariances, etc. It is to be noted that any method which uses the feature partitioning framework is required to address at least some of these issues.

Further, we deduce from the general framework, a novel FP-PCA approach, SubXPCA. SubXPCA divides a given pattern by choosing a fixed number of contiguous features in the order of appearance. If a fewer number features are present in the last sub-pattern as compared to other sub-patterns, those features are truncated. However, we try to choose sub-pattern size so as to minimize the loss of features in the last sub-pattern. In SubXPCA, we form a set, $\mathbf{P}^j$ of all j^{th} sub-patterns and features are extracted (their scope is limited to sub-patterns) from every sub-pattern set using classical PCA technique. As a final step, SubXPCA combines these locally-extracted features systematically by applying PCA to extract global features (here

inter-sub-pattern covariances are exploited). SubXPCA chooses fixed number of Principal Components from every sub-pattern set (PCs may also be chosen based on some threshold).

We further investigate application of feature partitioning framework exclusively for image data by exploiting matrix structure of images. In this direction, we propose first of our FP-PCA approaches on image data, Sub-IMage Principal Component Analysis (SIMPCA), where in each image is divided vertically (number of rows remains same) into k sub-images. Then local image features are extracted from each of k sub-image sets using 2DPCA (also known as IMPCA) [189][187]. The features thus extracted are concatenated to form local feature vector. We improve upon SIMPCA by proposing another FP-PCA approach, FLexible Image Principal Component Analysis (FLPCA), which divides each image in the same way as SIMPCA. FLPCA uses more sophisticated way for combining local image features by utilizing inter-sub-image covariances or correlations. These correlations aid in removing redundant features across sub-images, which may improve subsequent recognition/classification. A noteworthy observation is that SIMPCA captures local variation of image features by finding image principal components, where as FLPCA captures global variation of locally-extracted image features by finding PCs across sub-images.

To have deeper insight of FP-PCA approaches, we perform a theoretical study of these approaches to establish general properties of them.

Further, we apply the FP-PCA approach, SubXPCA for cluster analysis. In our approach, we combine DBSCAN [40] with SubXPCA method to find clusters with correlation structures in subsets of data. Such correlation connected clusters may be

invisible when the entire data is considered. Such clusters may give some indication about feature dependencies, cause-effect relationships in some subset of the data. For example, in medical applications, one cluster may indicate dependency between age and dosage. In this feature partitioning approach to correlation connected clusters, we combine our ideas of SubXPCA with DBSCAN way of clustering to elicit correlation structures in the neighbourhood of a core data object.

As our last contribution, we investigate the relevance of feature partitioning paradigm to PCA based subspace classification. Here we use our FP-PCA approach, SubXPCA (or any such approach in principle), to compute subspace for each class (category). A test pattern is classified to the class, for which maximum norm is obtained for the pattern after projection onto corresponding PCs.

1.5 Organization of Thesis

The rest of the thesis is organized as follows. In Chapter 2 we review PCA literature in brief and state the problem. We propose a generalized feature partitioning framework and the issues arise in the framework are presented in Chapter 3. An instance of the proposed feature partitioning framework, Cross-Sub-Pattern Correlation based Principal Component Analysis (SubXPCA), its time complexity analysis, etc., is discussed in Chapter 4. Moving in this direction, we extend feature partitioning approach to image data based on Two-dimensional structure of images by proposing two approaches, Sub-IMage based Principal Component Analysis (SIM-PCA) and FLexible Image Principal Component Analysis (FLPCA) in Chapter 5. In Chapter 6, we perform a theoretical study of FP-PCA approaches with respect to

variance-covariance structure of the data. Subsequently we show how an FP-PCA approach can be used for cluster analysis and subspace classification in Chapters 7 and 8 respectively. We conclude in Chapter 9.

Chapter 2

Principal Component Analysis Methods: A Literature Survey

2.1 Introduction

Note: The work in this chapter has been submitted to *Journal of Pattern Recognition Research*[1].

In this Chapter we review the literature related to Principal Component Analysis (PCA) methods in brief. For better understanding we classify the literature (Figs. 2.1 to 2.4) into various categories viz, Feature Partitioning or Block based PCA (FP-PCA) methods, 2D structure based PCA methods, Artificial Neural Network based methods, Kernel PCA methods, EM algorithms to PCA, Hybrid methods, Choosing number of Principal Components, etc., as described in the following sections. In

[1]Kadappagari Vijaya Kumar and Atul Negi, "A review of principal component analysis methods", **submitted to** *Journal of Pattern Recognition Research*, on Jan. 13^{th} 2009.

addition, we discuss how the existing PCA methods solve the problems of classical PCA. At the end, we state the problem we address in our investigations in this thesis.

2.2 Feature Partitioning or Block based PCA (FP-PCA) Methods

It is now known that classical PCA suffers from the drawbacks of not coping well with high dimensional data and scaling up to large data set due to its prohibitive computational complexity ($O(N.d^2)$). Another shortcoming is that classical PCA may not perform well in terms of recognition for applications where local region based features have discriminant information (e.g. facial expressions, pose, illuminations, etc and change detection applications). To overcome these problems, Block-based PCA methods (henceforth we call them as feature partitioning based PCA (FP-PCA) methods) were emerged. Chen et al proposed Sub-pattern based PCA (SubPCA) technique [21] which divides each pattern into equally-sized sub-patterns and groups similar sub-patterns from all patterns into corresponding sub-pattern set. Local features are extracted from each sub-pattern set and are concatenated to form reduced patterns. Chen et al proved that SubPCA [21] outperforms PCA in terms of classification. Other approaches that appear similar to SubPCA method are Multi-Block PCA [128] and Region-based PCA [136]. Multi-Block PCA [128] is used for change detection in remote sensing. Region-based PCA [136] is used in clutter rejection technique for Forward Looking Infra Red (FLIR) imagery in Automated Target Detection application. Region-based PCA technique is proposed to categorize all target images by

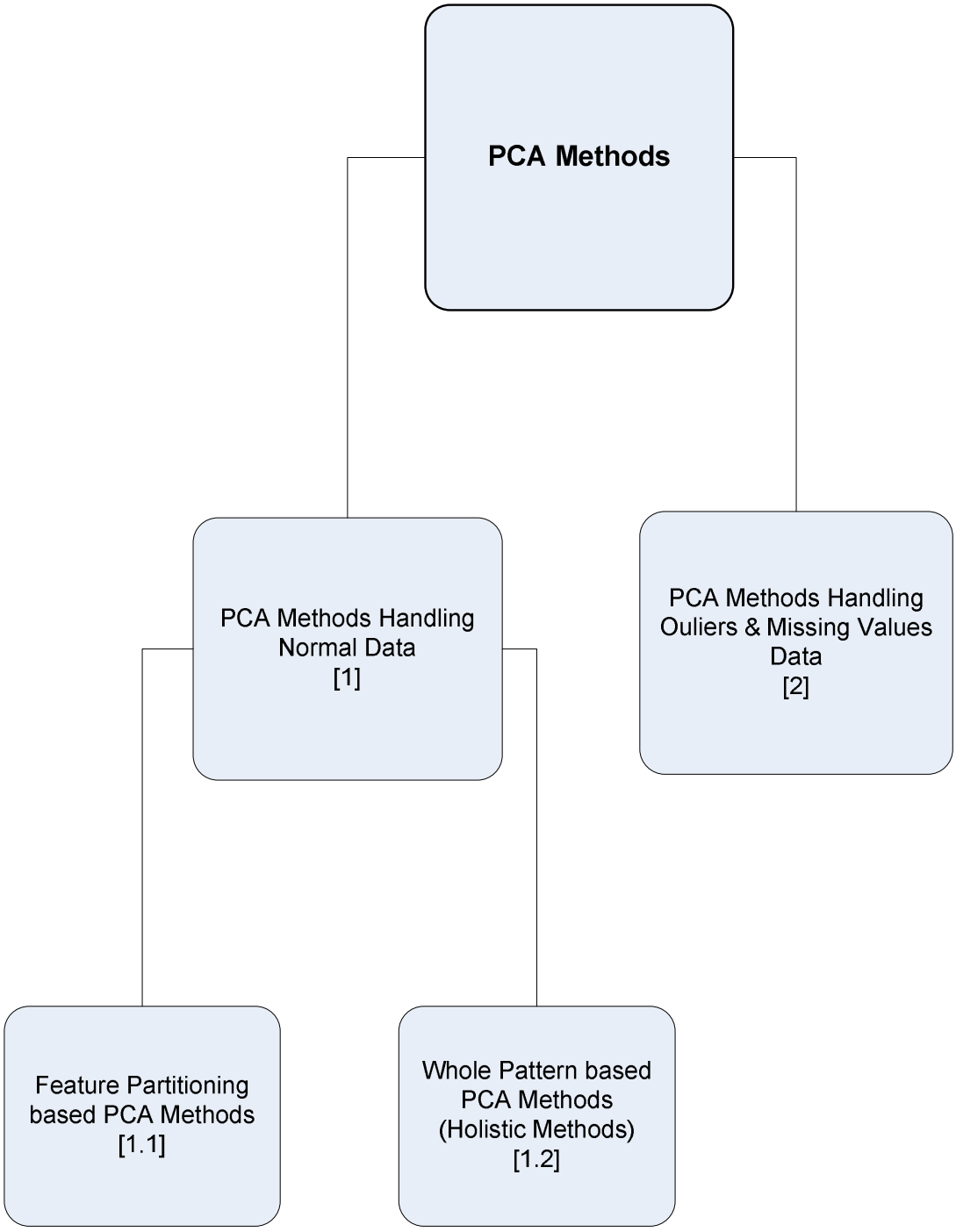


Figure 2.1: Main classification diagram of PCA methods

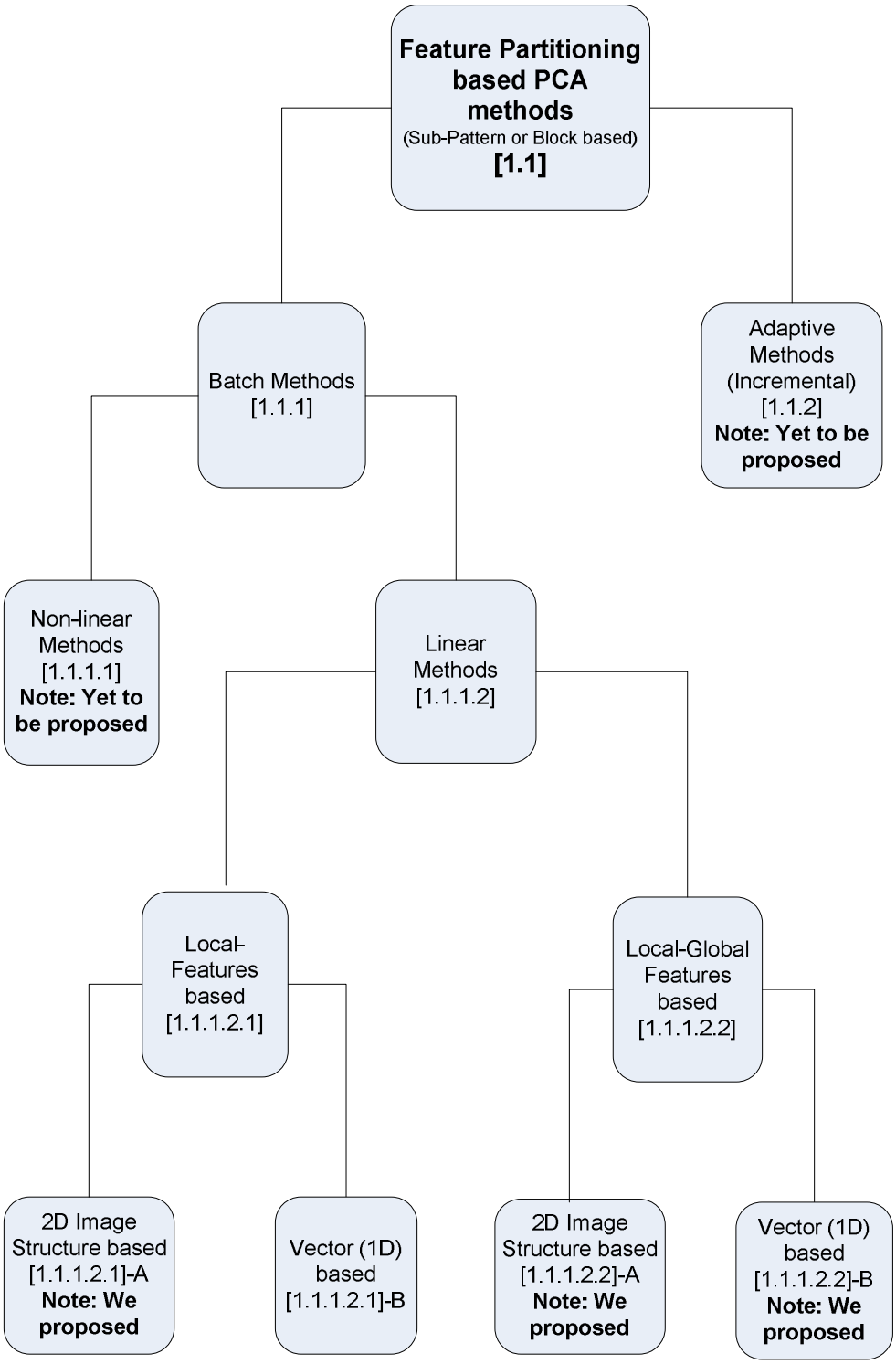


Figure 2.2: Classification of FP-PCA (sub-pattern based PCA) methods

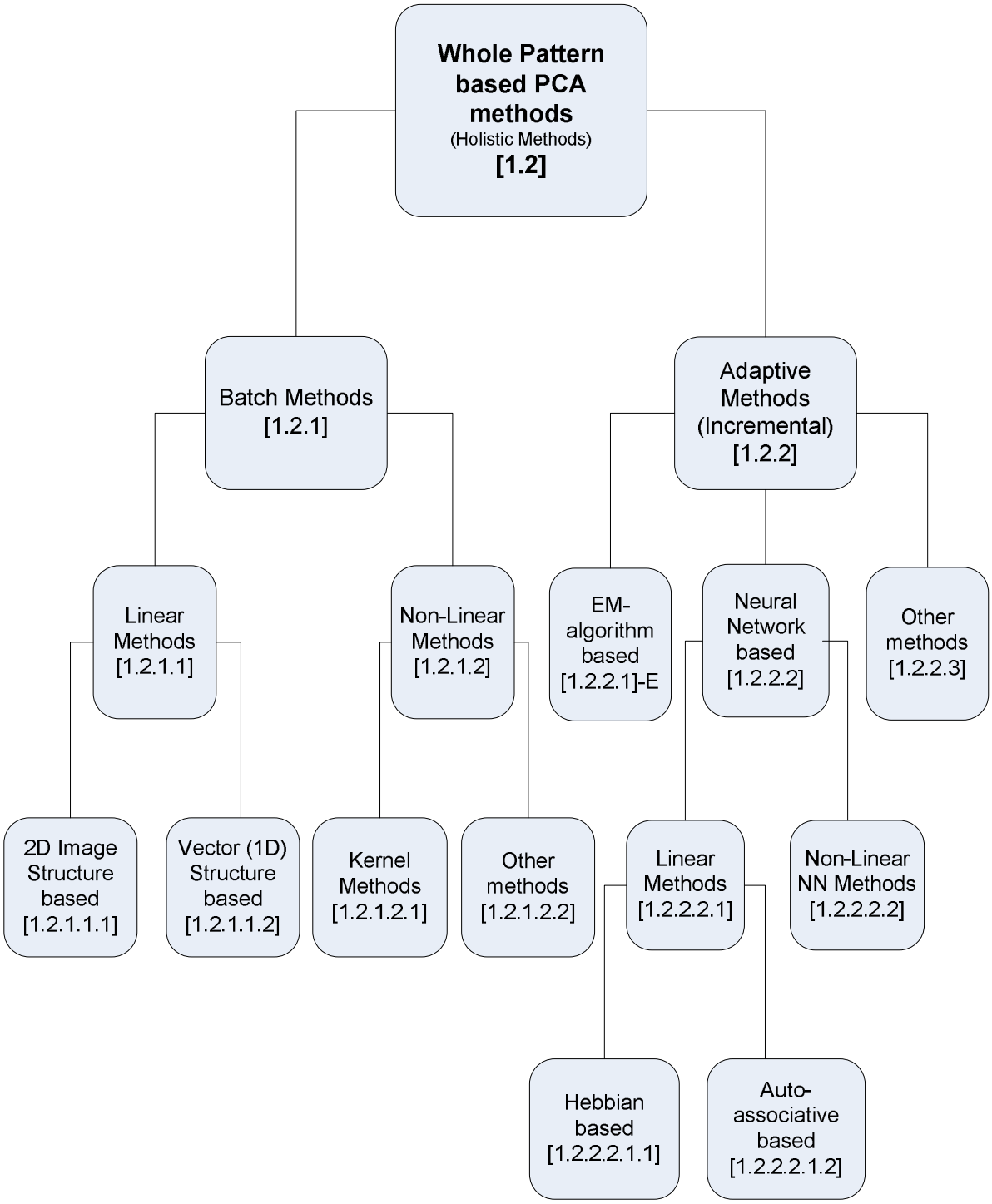


Figure 2.3: Classification of whole-pattern based PCA (global PCA) methods

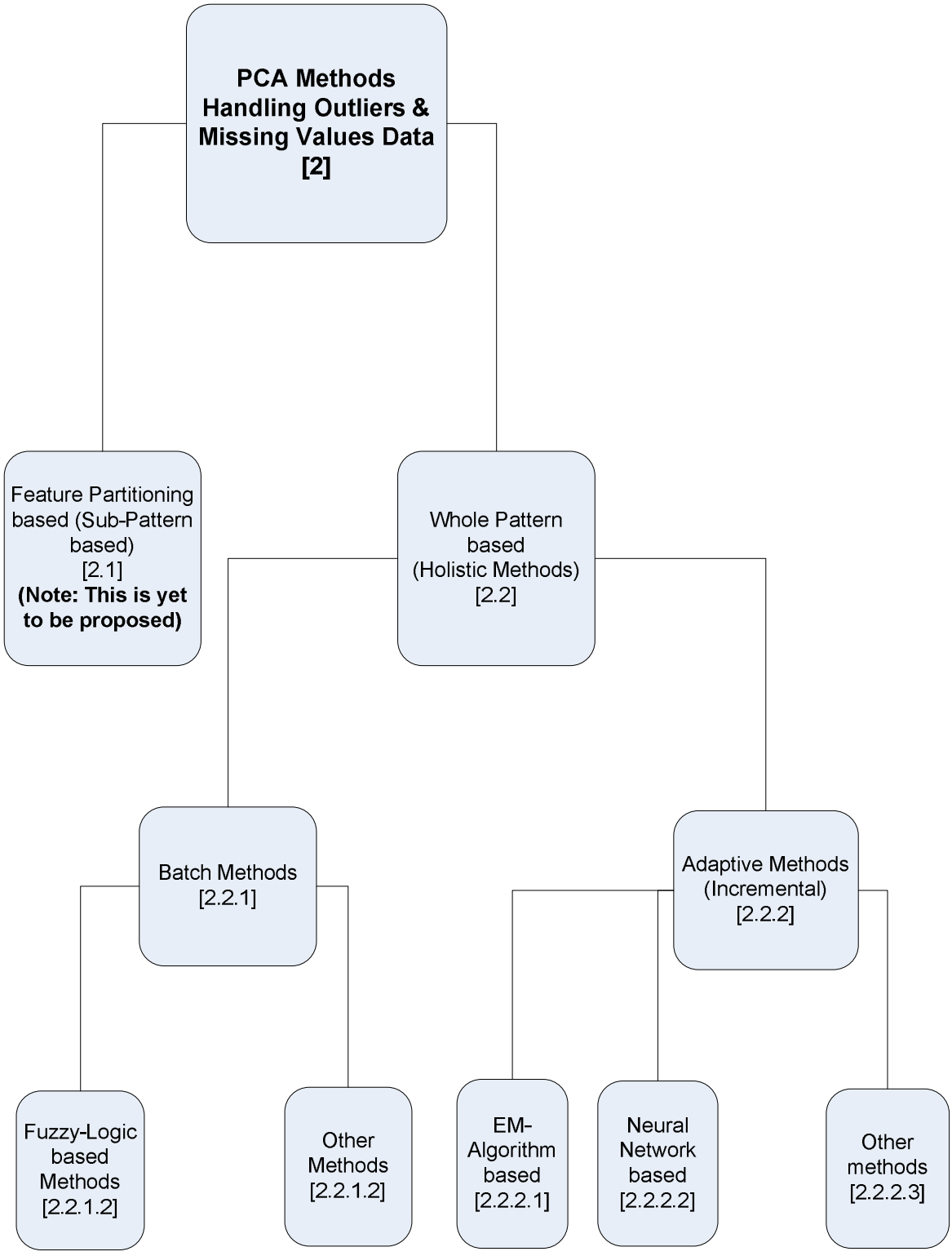


Figure 2.4: Classification of PCA methods for outliers and missing values data

clustering together the target images with respect to their similar sizes and shapes in order to form a group. Each group is further divided into several regions, and a PCA is performed for each region in a particular group to extract feature vectors.

Some improvements in this direction which combine the local features more structurally include Localized PCA [106] [107], Aw-SpPCA [157] and Clustered Block-wise PCA [113]. In localized PCA [106] [107], first they localize faces using skin colour. Erosion and dilation operations from mathematical morphology are used to get rid of small isolated segments. Then a PCA technique is used to localize mouth and eyes. Face is rotated until frontal position is obtained using x and y coordinates of two eyes, followed by contour technique to search for boundaries of the face (i.e. top, left, down, right). Then it is morphed to get a standard face. Next the faces are divided into k regions. Then PCA is applied for each region similar to SubPCA-like methods to extract features. Local features are combined using Probabilistic approach (adding local probability densities). Localized PCA focus more on localizing face image using different image processing techniques and needs a lot of preprocessing of images. Another method, Adaptively Weighted SubPCA (Aw-SpPCA) [157] operates directly on its sub-patterns partitioned from an original whole pattern and separately extracts local features from them. Moreover, Aw-SpPCA can adaptively compute the contributions of each part and then uses them to a classification task. However, Aw-SpPCA has additional burden of computing contributions. Clustered Block-wise PCA [113] uses an algorithm developed by Hall et al [56] to merge a pair of subspaces. For merging subspaces raw data is not required. For each block, distance is computed between its subspace and all other subspaces and store the block number

whose subspace fall below a distance threshold. After listing all the subspaces that are close to each other in terms of subspace distance, each pair is merged one-by-one with the subspace merging algorithm. Clustering and merging the block subspaces results in a reduction in the number of subspaces. When a new set of images are added to the data, once the number of these images becomes equal to the temporal size of the block, one can apply PCA to the blocks within this new data set and merge the subspaces that are close to existing subspaces. In this way, the necessary storage will not increase linearly to the size of the added data and correlation of local visual events can be exploited as new data is added. If an existing subspace is merged with a subspace computed from a new data block, the projection of the existing data block should be updated with the projection in the newly merged subspace. However Clustered Block-wise PCA suffers from the drawbacks: (i) it has quadratic (in terms of number of blocks) time complexity and may be prohibitive if the number of blocks is high and (ii) it has overhead to update projection for each merge.

A slightly different approach, Sub-Holistic PCA (SHPCA) [80] was proposed for face recognition. In this scheme, each face image is not only taken as a whole but four equally-sized sub-images are also formed from the given image. In this scheme, instead of generating a single face space, five face spaces are generated. The image to be tested is also divided into four parts and the complete image with the four sub-parts is projected in their respective face spaces. The results from all five face spaces are obtained and from the five proposed matches one match is found. However, SHPCA suffers from the following drawbacks: (i) It needs to compute original face space, in addition to 4 new face subspaces, which is computationally intensive and (ii)

it divides each face into 4 sub-patterns only, which may not be correct for all faces.

The methods discussed so far in this section have a similarity: *they divide each pattern into sub-patterns and apply classical PCA to each of the sub-patterns separately.* We call these methods as SubPCA-like methods because they perform feature extraction in the similar way as SubPCA method.

In contrast to SubPCA and similar approaches, modular PCA approach (modPCA) [53] divides each pattern into sub-patterns and *apply single PCA to the set of all sub-patterns to find principal eigenvectors, instead of applying single PCA to each of sub-patterns.* Then, the sub-patterns are projected onto the same principal eigenvectors to extract local features. Another approach similar to modPCA, called Eigen-regions method [45], uses segmentation techniques to divide the given image into meaningful regions, and then applies single PCA to all these regions. Eigen-regions method uses down-sampling procedure to reduce the size of a region, so that PCA can be applied to reduce computation.

In contrast to other PCA methods which are based on whole patterns, FP-PCA methods are unique in their approach, which are based on novel idea of *partitioning each pattern into sub-patterns and extract local features.* FP-PCA methods alleviate some of the crucial problems of classical PCA and show (i) Reduced computational complexity, (ii) improved recognition/classification rate by local feature extraction if local variations are prominent, etc. However, FP-PCA methods suffer from the following problems: (i) These methods purely perform local feature extraction, that is feature extraction is limited to a subset of original feature set (or limited to sub-patterns). Therefore, FP-PCA methods may not perform well if there exists global

variations, (ii) Summarization of variance is not good because the entire covariance structure is not utilized which yields more number of local PCs resulting in less dimensionality reduction, (iii) FP-PCA methods do not exploit two-dimensional structure of image data (Section 2.3) because they all use classical PCA in a region or block to extract local features.

2.3 Two Dimensional Image Structure based Methods (2DPCA and Its Variants)

PCA and its disadvantages to image data:

One of the most successful image recognition applications of PCA is the human face recognition. Kirby and Sirovich [85] were the first to employ Karhunen Loeve transform to represent facial images. Their work was followed by the PCA Eigenface technique [164].

PCA (e.g. Eigenface method) [164] considers images as vectors in a high dimensional image space. All the images are projected onto the eigenspace spanned by the leading eigenvectors of the sample covariance matrix of the training images. Although PCA is popular in image feature extraction, it suffers from the following drawbacks: (i) 2D image matrices must be transformed into 1D image vectors. The resulting image vectors of faces usually lead to a high dimensional image vector space, where it is computationally intensive to compute the covariance matrix accurately due to its large size and the relatively small number of training samples. Other methods can be used to avoid covariance matrix computation (e.g. Adaptive methods as dis-

cussed in section 2.4), however the eigenvectors can be evaluated accurately by using covariance matrix only because the eigenvectors are statistically determined by the covariance matrix, irrespective of the method used for obtaining it [189], (ii) PCA does not make use of inherent matrix spatial structure of images, which may lead to low performance, (iii) Generalization ability is limited while extracting local features when variations in local region or a part of patterns are prominent, (iv) Because of the small sample size (SSS) problem, PCA is likely to be over-fitted to the training set.

To overcome the limitations of classical PCA for image data, more recently, Two-dimensional PCA (2DPCA) [189] (also known as image PCA (IMPCA) in the previous paper [187]) was proposed. 2DPCA was proved to be superior in terms of computational cost and classification. Unlike PCA that treats images as vectors, 2DPCA views an image as a matrix of image features. 2DPCA is more suitable for small sample size problems (like face recognition) since its image covariance matrix is quite small.

2DPCA description and algorithm: [189]

First, the covariance matrix, $(\mathbf{M})_{n\times n}$ is computed for the given set of training images, $\mathbf{A} = \{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N\}$ as given by

$$(\mathbf{M})_{n\times n} = \frac{1}{N}.\sum_{i=1}^{N} E[(\mathbf{A}_i - \bar{\mathbf{A}})^T_{n\times m}.(\mathbf{A}_i - \bar{\mathbf{A}})_{m\times n}] \tag{2.1}$$

where $\bar{\mathbf{A}}$ is the mean of images and is given by $\bar{\mathbf{A}} = \frac{1}{N}.\sum_{i=1}^{N} \mathbf{A}_i$. Next, find $r\,(< n)$, eigenvectors of $\mathbf{M}$ corresponding to first r largest eigenvalues. Let $(\mathbf{E})_{n\times r}$ be the matrix of r column eigenvectors (projection vectors) chosen in this step. Finally, the set of images, $\mathbf{A}$, are projected onto $\mathbf{E}$ to get a set of reduced images, $\{\mathbf{IM}_i;$

$i = 1, 2, \ldots, N\}$ and is given by

$$(\mathbf{IM}_i)_{m\times r} = (\mathbf{A}_i)_{m\times n}.(\mathbf{E})_{n\times r} \tag{2.2}$$

Due to its computational superiority, 2DPCA is widely used in various fields especially in face recognition area. Junwei Tao et al [158] applied 2DPCA for palmprint recognition. Xiaoyu Zhang et al [197] used 2DPCA followed by Support Vector Machine (SVM) for face detection. Xiaoyu Zhang et al [197] showed that the method can effectively detect faces under complicated background, and the processing time is shorter than using SVM alone. Yin Hongtao et al [63] proposed a 2DPCA method using wavelets to obtain the feature vector representing a face: First, it uses the wavelet decomposition to extract intrinsic features of face images. As a result of wavelet decomposition, they obtained four sub-images (namely approximation, horizontal, vertical, and diagonal detailed images). It was shown that by decomposing a face image using wavelet transform, the low-frequency face image is less sensitive to the facial expression variations. The authors selected the approximation image as the feature of face image. Second, they performed 2DPCA twice (2DPCA in horizontal direction followed by vertical direction), after which the discriminant information is moved to the upper-left corner of the image and is used to classify the face image. The method showed significant reduction in computational time and slight improvement in recognition as compared to other wavelet methods [100][23].

Improving 2DPCA with different distance measures:

The typical classification measure used in 2DPCA-based face recognition is the sum of the Euclidean distance between two feature vectors in a feature matrix, called distance

measure (DM). However, this measure is not compatible with the high-dimensional geometry theory. So a new classification measure compatible with high-dimensional geometry theory and based on matrix volume is developed by Meng and Zhang for 2DPCA-based face recognition [109]. Distance between two images, $\mathbf{A}_i$, $\mathbf{A}_j$ using Volume measure is given by

$$\mathbf{VM}_{i,j} = \sqrt{det[(\mathbf{A}_i - \mathbf{A}_j)^T.(\mathbf{A}_i - \mathbf{A}_j)]} \tag{2.3}$$

Another method proposed by Zuo et al [201], combines 2DPCA with Assembled matrix distance (AMD) which was proved to be effective for 2DPCA based image recognition. Motivated by the idea of using matrix structure of 2DPCA, Chen et al [22] tried to extract features for any vector pattern by first 'matrixizing' it into a matrix pattern and then applying the matrixized version of PCA, known as MatPCA to the pattern. MatPCA uses a minimization of the reconstructed error for the training samples like PCA to obtain a set of projection vectors. It was observed that the computational burden of extracting features is largely reduced. However, it is to be noted that matrixizing a pattern may not work well with all the data sets because of its creation of artificial spatial relationships.

Some improvements to overcome large number of coefficients in 2DPCA:

Despite its superiority, 2DPCA needs more coefficients (that is more extracted features) for image representation than classical PCA. In an effort to reduce number of coefficients, Junwei Tao et al [158] applied classical PCA (1DPCA) to the coefficients (PCs) obtained by 2DPCA. In their method they selected the PCs that have better balance on maximizing the between-class distance while minimizing the within-

class distance rather than the principal components with highest variance, because such principal components are better for classification. Wen and Pengfei proposed an approach, IPCA [175] which uses 2DPCA to obtain the projective feature image which is processed by 2DPCA again. IPCA shows significant improvement in terms of recognition. Further in this direction of reducing number of coefficients, Zhang and Zhou proposed $(2D)^2$PCA, Two-directional 2DPCA, which computes projection vectors (eigenvectors) for both row and column directions [194]. The same bi-directional concept as shown by $(2D)^2$PCA [194] is re-iterated by different researchers: (i) Sun and Ruan [153] for face expression identification in the form of 2D-2DPCA, (ii) Konga et al [88] for recognition in the form of Bilateral-projection-based 2DPCA (B2DPCA). In the same paper Konga et al proposed a kernel version of 2DPCA called K2DPCA scheme and the relationship between K2DPCA and KPCA is explored and (iii) Anbang Xu in the form of complete PCA [181]. Zuo et al [202] extended assembled matrix distance (AMD) metric to improve Bidirectional PCA (BD-PCA) and an AMD metric is presented to calculate the distance between two feature matrices and then the nearest neighbour and nearest feature line classifiers are used for image recognition. Diagonal Principal Component Analysis (DiaPCA) [195] captures the essence of using the relationships between variations of rows and those of columns of images, using a different approach. In DiaPCA, for each training face image, a diagonal face image is computed as given in [195], then 2DPCA is applied for the set of diagonal face images.

Generalization to nDPCA:

Motivated by 2D forms of the PCA, Yu and Bennamoun further developed and ex-

tended the idea to an arbitrary n-dimensional space. Analogous to 1D- and 2DPCA, the new nD-PCA is applied directly to n-order tensors (n = 3) rather than 1-order tensors (1D vectors) and 2-order tensors (2D matrices). In order to avoid the difficulties faced by tensors computations, nD-PCA algorithm has to exploit a Higher-Order Singular Value Decomposition (HO-SVD) to make it practically feasible.

Discrimination front:

Although PCA ensures that the features extracted have least reconstruction error, it may not be optimal from a discrimination standpoint. To improve feature extraction discrimination point of view, (i) Nhat and Lee [111] proposed a 2DPCA based method which makes use of Laplacian weighting matrix method which considers data labeling, and makes the performance of recognition system better with the complexity nearly same as that of 2DPCA. Another direction to extract discriminant features from 2DPCA is to combine it with LDA technique. Sanguansat et al [142] combined 2DPCA and 2DLDA, which improved dimensionality reduction of the feature matrix in addition to improving classification accuracy. Similarly, Zuo et al [203] combined bidirectional PCA (BDPCA) with LDA (say, BDPCA + LDA), which performs an LDA in the BDPCA subspace. Their experimental results show that BDPCA + LDA needs less computational and memory requirements and has a higher recognition accuracy than PCA + LDA. In a different approach to improve 2DPCA further, Kim and Choi [81] computed window based 2DPCA and 2DLDA methods using image covariance obtained from windowed features of images. A windowed input feature consists of a number of pixels, and the dimension of input space is determined by the number

of windowed features. A $m \times n$ window feature is treated as a $m.n$ feature vector. Each element of an image covariance matrix can be obtained from the inner product of two windowed features. The 2DPCA and 2DLDA methods are then computed using the image covariance matrix of the windowed features. It was observed that 2DLDA performed well as compared to other LDA methods and 2DPCA. Using these window based 2D methods, (i) we can control the dimension of the input space by changing the window size or by overlapping the windows, which consequently solves the small sample size (SSS) problem and (ii) the computational load is significantly reduced.

Link to Block based (Feature Partitioning based) PCA approaches:

The methods based on 2D matrix structure (2DPCA) are proved to be equivalent [170] to special cases of image block based feature extraction (that is, when each row is taken as a block) (Section 2.2). Later Gao [183] proved that the earlier proof by Wang et al [170] is not correct. Gao analyzed that 2DPCA views the rows of images as training samples that constitute m sub-training sets instead of original images (m is equivalent to the rows of images) where as Block based PCA (modPCA) views entire $N.m$ blocks as a single training set.

Note that the approaches discussed in this section (except DWT based PCA) are based on whole patterns and have a common drawback of not exploiting local features. Local features are those extracted from a local region (or sub-pattern) rather than from entire image (or pattern).

2.4 Artificial Neural Network based Principal Component Analysis Methods

Batch methods versus Incremental methods:

Artificial Neural Networks (ANN) are massively parallel interconnections of simple neurons that function as a collective system. PCA computation can be done in two modes: (i) Batch mode (ii) Incremental or Adaptive mode. The batch methods assume that all the data is available beforehand. In batch methods the PCA is performed as follows: (i) Calculate covariance matrix by making use of the training data (patterns), (ii) the covariance matrix is then decomposed to find the principal component directions of the variances (that is eigenvectors corresponding to highest eigenvalues). In practice, usually the covariance matrix is diagonalized using some numerical technique such as Householder-QR technique [127]. In contrast to batch methods, Incremental methods (i) work directly with the data *without computation of covariance matrix* in advance and (ii) they might be implemented adaptively so that the directions of the Principal Components (PCs) are adjusted after a new data is received, without the need of reusing all previous data (patterns). This approach is suitable for real-time applications or for very high dimensional problems where the computational expense and storage requirement is an important consideration. One application area is computer vision, in which all visual filters are incrementally derived from very long on-line real-time video stream, motivated by the development of animal vision systems. On-line development of visual filters requires that the system perform while new sensory signals flow in. An online developing system must

observe an open number of images and the number is larger than the dimension of the observed vectors. There is evidence that biological neural networks use an incremental method to perform various learning, e.g., Hebbian learning [176]. As we all aware of, ANNs are well known for incremental learning. More details of Neural Networks for PCA can be found in [4] [37].

We recollect that PCA has two main properties - (i) It finds the uncorrelated directions of maximum variance in the data space and (ii) it provides the optimal linear projection in the least square sense. According to these two properties, two types of PCA networks can be found in the literature. Hebbian type learning algorithms are based on the variance maximization and uncorrelatedness property, whereas linear Auto-Associative MLPs (AA-MLP) compute the PCA space, because this subspace yields the best linear mean-square approximation [37].

Auto-Associative Neural Networks (AA-NNs) for PCA:

The relationship between PCA and AA-MLPs was first noticed by Bourlard and Kamp [13]. If AA-NN contains hidden layer size less than input layer size, the network works as feature extractor and finds efficient ways of compressing the information contained in the input patterns. The use of such a scheme for information compression and dimensionality reduction was first suggested by Rumelhart et al [139]. It was analyzed formally by Bourlard and Kamp [13] using the concept of singular value decomposition of matrices. Further results were obtained by Baldi and Hornik [4], who provided a complete description of the error surfaces of multilayer linear networks (of which AA-NNs with one hidden layer are a special case). Further PCA using AA-NNs extended to fuzzy data sets by Denoeux and Masson [34].

This method exploits recent results regarding the ability of linear AA-NNs to perform information compression in just the same way as PCA, without explicit matrix diagonalization. Further, Girard and Iovleff [51] proposed auto-associative models to generalize PCA. These AA-NN models have been introduced in data analysis from a geometrical point of view. They are based on the approximation of the observations scatter-plot by a differentiable manifold. In their study those models are interpreted as projection pursuit models adapted to the auto-associative case. The supervised AA-NN algorithms, may end up being trapped into local minima [13], and also their global treatment of information makes it difficult to implement ANNs into efficient hardware [60]. Therefore many researches focussed on the study of unsupervised ANNs, particularly Hebbian type ANNs (after the work of the Canadian neurophysiologist Hebb) [79][116][117][149][141][177]. These methods are based on Oja's earlier work [114].

Hebbian-Type ANNs for PCA (HANN):

The motivation for the popular Hebbian ANNs came from the so called Hebbian Learning Rule. In his seminal work 'The Organization of Behavior' [59], Hebb proposed a simple, yet biologically motivated rule, for adjusting the synaptic weights during a neural network learning process, which is given as *'when neuron N_1 and unit N_2 are simultaneously excited, increase the strength of the connection between them'*. For the case where neurons are modeled as units with continuous output activation this correlation-type rule is given in mathematical form as *'Adjust the strength of the connection between units A and B in proportion to the product of their simultaneous activation'*. Interestingly, this simple rule turns out to be closely related to PCA when

the neural units are linearly modeled. Oja [114] showed that a normalized version of the Hebbian rule applied on a single linear unit extracts the first principal component of the input sequence, i.e., it converges to the principal eigenvector of the input auto-correlation matrix [92] [112]. Recently, Nicole [112] evaluated Subspace Network (SN) [117], Generalized Hebbian Algorithm (GHA) [117][141], Weighted Subspace Algorithm (WSA) [117] and Stochastic Gradient Ascent (SGA) [117] ANNs, in terms of efficiency of extraction of eigenvectors and pattern classification, compression and reported the following facts. It was observed that there is a decrement in the accuracy of computation of the first eigenvector along the number of neurons for all the ANNs, with the relevant exception of WSA. For classification tasks, Hebbian ANNs are unable to distinguish patterns properly as compared to SVD based PCA. It is worth noting the performance of the SN algorithm in terms of compression and reconstruction, when only very few (actually, 4) eigenvectors are considered, is quite comparable to the SVD result. The other ANNs perform grossly worse; their NMSEs (normalized MSE) are by and large an order of magnitude larger than that from SVD. As the number of eigenvectors considered increases, though, the gap between the SVD algorithm's performance and the ANNs' grows and the precision of the reconstruction by the non-adaptive algorithm's results increases. In a nutshell, the results obtained for more demanding tasks suggest that the ANNs (and also WSA) are easily outperformed by other classical numerical algorithms, especially whenever a high precision in the reconstruction is requested.

Kung et al [92][91] proposed the Adaptive Principal-component Extractor (APEX) model which can effectively support a recursive approach for the calculation of the

p^{th} principal component given the first $(p-1)$ ones. The motivation behind such an approach is the need to extract the principal components (PCs) of a given data patterns when the number of required PCs is not known a priory. It is also useful in applications such as speech analysis where the correlation matrix (covariance matrix) of the data might be slowly changing with time. Then the new PC may be added to compensate the change without affecting the previously computed PCs. APEX model has both feed-forward and lateral connections.

The methods based on Oja's original work [114] may not adequately consider automatic selection of learning parameters, thus leading to slow convergence or even divergence if parameters are not properly chosen. The stability and the ways of choosing the values of the learning rate parameters of the Oja's one-unit learning rule and some other gradient type algorithms have been discussed in [118] [78] [31]. Chen and Chang [19] proposed an adaptive learning algorithm (ALA) for PCA, where in, the learning rate parameters can be selected automatically and adaptively according to the eigenvalues of the input covariance matrix that are estimated during the learning process. The simulation results demonstrated that the ALA can converge quickly to the desired targets while the GHA diverges in the large eigenvalue case. Further, based on the work of Oja [118] and Sanger [141], Weng et al [176] proposed a fast converging method, candid covariance-free incremental PCA, to compute the principal components of a sequence of samples incrementally without estimating the covariance. The method is motivated by the concept of statistical efficiency (the estimate has the smallest variance given the observed data). To do this, it keeps the scale of observations and computes the mean of observations incrementally, which is

an efficient estimate for some known distributions such as Gaussian. The method is proposed for real-time applications, and does not allow iterations. It converges very fast for high dimensional image vectors. Chatterjee et al [17] presented adaptive algorithms which are based on an unconstrained objective function, which can be minimized to obtain the principal components. By using this objective function, they derived adaptive algorithms by using: (i) gradient descent, (ii) steepest descent, (iii) conjugate direction and (iv) Newton-Raphson methods for PCA. These methods were shown to converge faster than the traditional gradient descent PCA algorithms due to Oja, Sanger, and Xu [17].

In local Hebbian type learning algorithms the modification of the i^{th} row of the weight matrix between input and output layer depends only on the i^{th} output unit and the input. Due to this locality it has been argued that these algorithms are biologically plausible [174].

The detailed discussion of generalized Neural Network PCA models such as constrained PCA, oriented PCA, asymmetric PCA and other ANN PCA models can be found in [36].

Nonlnear PCA using Neural Networks:

Because of its linearity, PCA is not always suitable, and has redundancy in expressing data. To overcome this problem, some nonlinear PCA methods have been proposed. Whether the nonlinear approach has a significant advantage over the linear approach is highly dependent on the data set. The nonlinear approach is generally not good if the data is short and noisy, or the underlying structure is essentially linear. Nonlinear principal component analysis (NLPCA) using AA-NNs was first introduced by

Kramer [90] in the chemical engineering literature, and is now used by researchers in many fields. The presence of local minima in the cost function renders the NLPCA using ANNs somewhat unstable, as optimizations started from different initial parameters often converge to different minima. Regularization by adding weight penalty terms to the cost function is shown to improve the stability of the NLPCA [65]. Another way to realize non linearity is to have a mixture model [193] by concurrently performing global data partition and local linear PCA. The partition is optimal or near optimal, which is realized by a soft competition algorithm called 'neural gas'. The local PCA type representation is approximated by a neural learning algorithm in a nonlinear auto-encoder network, which is set up on the generalization of the least-squares reconstruction problem leading to the standard PCA. Such a local PCA type representation has a number of numerical advantages, for example, faster convergence and insensitive to local minima. Most of the nonlinear methods have drawbacks, such that the number of principal components must be predetermined, and also the order of the generated principal components is not explicitly given. Ryo Saegusa et al [140] proposed a nonlinear PCA algorithm based on hierarchical MLP neural network model that nonlinearly transforms data into principal components, and at the same time, preserving the order of the principal components. The network composed of a number of independent sub-networks that can extract ordered nonlinear principal components.

Other methods:

A comparative study of derived classification accuracies of a neural network (NN) implementation of Sammon's mapping, an auto-associative NN (AA-NN) and a mul-

tilayer perceptron (MLP) feature extractor and conventional principal component analysis (PCA) was carried out [99]. The study reveals that MLP provides the highest classification accuracy at the cost of deforming the data structure, whereas the linear models preserve the structure but usually with inferior accuracy. Huang [66] used Generalized Hebbian algorithm (GHA) to perform PCA for *Seismic data analysis*. The neural network using an unsupervised GHA is adopted to find the principal eigenvectors of a covariance matrix in different kinds of seismograms. GHA can extract the information of seismic reflection layers and uniform neighbouring traces. The method also provides a significant seismic data compression [66].

From our study, it is understood that, incremental learning based on ANNs for PCA is very useful for applications where the data is received incrementally. These methods also reduce storage requirements and computational requirements as well. However, at times incremental learning methods may suffer from the drawbacks: (i) slow convergence (in this case these methods may be computationally expensive), (ii) not efficient to extract eigenvectors, (iii) may not be efficient for compression and reconstruction and (iv) may not be good for pattern classification view point as compared to classical PCA methods. Batch methods although require relatively more storage and computational requirements, they show their efficiency in applications (where data is available beforehand) in terms of (i) better extraction of eigenvectors, (ii) better feature extraction and construction and (iii) pattern recognition tasks.

2.5 Kernel Principal Component Analysis Methods

The kernel principal component analysis (KPCA) has been applied in numerous machine learning applications and it has exhibited superior performance over previous approaches, such as PCA. Classical PCA is suitable in applications where the underlying structure is linear. The linear PCA either needs more principal components or unsuitable for the data sets where nonlinear structure is present. KPCA [145] introduces a nonlinear form of doing PCA by using *kernel functions*. In KPCA, as a first step we map the input feature space into a kernel space by using a kernel nonlinear mapping, next we perform classical PCA on the transformed kernel space.

$$\phi : \Re^d \rightarrow \mathbf{K} \tag{2.4}$$

where $\mathbf{X}_i \in \Re^d$ is input feature vector and $\mathbf{K}$ is high-dimensional transformed kernel space. In fact, we do not actually compute $\phi - map$ for an input data $\mathbf{X}_i$, instead we use kernel functions in the place of dot products $\phi(\mathbf{X}_i).\phi(\mathbf{X}_j)$. Some kernel functions include (i) polynomial kernel which is given by

$$k(\mathbf{X}_i, \mathbf{X}_j) = (\mathbf{X}_i.\mathbf{X}_j)^n \tag{2.5}$$

(ii) radial basis functions (iii) sigmoid kernels, etc. Compared to other nonlinear methods to PCA, for e.g. Auto Associative MLPs or principal curves, KPCA has the advantage of not needing nonlinear optimization, it needs only solving eigenvalue problem as classical PCA. Thus there is no problem of getting trapped into local minima during learning in case of KPCA. KPCA as a nonlinear feature extractor has

been proved to be a powerful tool in preprocessing for classification tasks. Mika et al [110] tried to emulate KPCA as a natural generalization of linear PCA. They have shown how to use nonlinear features for data compression, reconstruction, and denoising applications common in linear PCA. Please note that as the results provided by KPCA live in some high dimensional feature space and need not have pre-images in input space. Their experiments reveal that reconstruction results of KPCA were comparable with linear PCA and KPCA has shown significantly better results with respect to denoising. More details including derivation on KPCA can be found in [146].

Some improvements to KPCA:

KPCA has high computational cost due to its dense expansions of kernel functions. To overcome this shortcoming, Sparse Kernel Feature Analysis (SKFA) method was proposed by Smola et al [150]. SKFA overcomes the problem by using L_1 norm for feature extraction in coefficient space, instead of kernel Hilbert space in which KPCA is formulated in. The SKFA algorithm was proved to be fast and leads to sparse representations. KPCA is not sparse because the computation of principal components require to compute kernel functions associated with every training patterns. Another approach to overcome this problem is proposed by Tipping [162] which is known as Sparse Kernel Principal Component Analysis (SKPCA). First the SKPCA method approximates covariance matrix in feature space by computing a subset of outer products of feature vectors using maximum likelihood approach based on probabilistic PCA [163]. Next, the KPCA is applied to obtain sparse projections.

The most widely used kernel functions in the literature are polynomial kernels,

Gaussian kernels, and sigmoid kernels. In an effort to improve face recognition performance Chengjun Liu proposed Gabor-based KPCA with new kernel function 'Fractional Power Polynomial Models' [102]. The method integrates Gabor wavelet representation of face images with KPCA using fractional power polynomial models for enhanced face recognition. The feasibility of the Gabor-based KPCA method with fractional power polynomial models has been successfully tested on both frontal and pose-angled face recognition, using two data sets from the FERET database and the CMU PIE database. The superiority of the Gabor-based KPCA method with fractional power polynomial models is shown by the author in terms of both absolute performance indices and comparative performance against PCA, KPCA with polynomial kernels and KPCA with fractional power polynomial models, etc. To avoid the potential problems with standard KPCA, Chin and Suter [24] brought out an incremental computation algorithm for KPCA, where incremental linear PCA is computed in the kernel induced feature space.

Choosing number of dimensions in Kernel based Subspace methods:

One of the traditional approaches to select the dimensions is based on cumulative proportion computed from the kernel matrix for each class. To select number of dimensions systematically, Kim et al [84] proposed a new method selecting optimal or near-optimal subspace dimensions for KNS classifiers using a search strategy and a heuristic function called the 'Overlapping Criterion'. The heuristic criterion that uses critical information about the specific dimensions chosen, called the overlap, between the corresponding subspaces.

Subspace classification using kernel trick:

Peng Zhang et al [196] proposed a kernel-pooled local discriminant subspace method and compared it against KPCA and generalized discriminant analysis (GDA) in classification problems. The method computes a nonlinear pooled local discriminant subspace by using the kernel trick and it makes use of Gaussian kernel.

Some applications of KPCA:

KPCA with polynomial kernel of degree d, is applied to face recognition by Yang et al [190] and it was observed that KPCA with kernel of degree 3 has given a lower error rate in both Yale (Eigenface: 28.49%; KPCA ($d = 3$) : 24.24%)) and AT&T (Eigenface: 2.75% KPCA ($d = 3$) : 2%) data sets as compared to traditional Eigenface method. They carried out experiments using leave-one-out strategy. KPCA is also used by Kim et al [83] for face recognition application using ORL face data set. It was showed that KPCA with polynomial kernel of degree 4 (error rate is : 2.5%) is significantly lower error rate than traditional PCA (error rate: 10%). In natural language domain, KPCA is used by Dekai Wu et al [178] for word sense disambiguation. The method outperformed other methods such as naive Bayes method, maximum entropy method and SVM model as well. KPCA is also used in remote sensing domain [156] and scene analysis for mobile robot based on multi sonar ranger data [171].

2.6 EM Algorithms for PCA

Expectation Maximization (EM) algorithm can be used for learning the principal components of a data set. EM algorithms do not require computing the sample covariance matrix and deal with high dimensional data more efficiently than classical PCA. Tipping and Bishop [163] proposed probability model for PCA. PCA can be

viewed as a limiting case of a particular class of linear Gaussian models. Gaussian models assume that $\mathbf{x}$ is produced as a linear transformation of some r-dimensional latent variable $\mathbf{y}$ plus additive Gaussian noise, $\mathbf{v}$. Based on the probability model, EM algorithm is used to learn principal component directions [138]. An application of Probabilistic PCA for rapid speaker adaptation may be found in [82]. Other EM algorithms include, EM algorithm for integrated-squared-error minimization [1], EM algorithm for high-dimensional spaces [38]. The EM learning algorithm for PCA is an iterative procedure for finding the subspace spanned by the leading r eigenvectors without explicit computation of the sample covariance. It is attractive for small values of r, because its complexity is limited by $O(r.N.d)$ per iteration and depends only linearly on both the dimensionality of the data (d) and the number of points (N). Methods that explicitly compute the sample covariance matrix have complexities limited by $O(N.d^2)$.

2.7 Hybrid Methods

In this section we review the techniques which are combination of PCA and other methods such as LDA, Rough Sets theory.

PCA and LDA:

Zhao et al applied LDA for principal components obtained from PCA technique to improve the generalization ability of LDA when few samples are available. The combined PCA+LDA linear classifier shows significant improvement over pure LDA linear classifier [199]. It is well-known that LDA does not work well with nonlinear applications. To improve LDA for nonlinear case, Yang et al [186] [185] investigated Kernel

Fisher Discriminant (FLD) and then proposed two-step method: first KPCA is applied and then LDA is applied in the KPCA-transformed space. Rajagopalan et al [130] proposed a face recognition method that combines information acquired from global and local features of the face for improving performance. They have considered the 3 complementary features, (i) grayscale image of the face, (ii) the edginess image of the face (which is robust to different illuminations), and (iii) the eyes (which are robust against occlusions and facial expressions). Dimensionality of each of 3 original feature spaces is reduced by applying PCA followed by fisher analysis. Recognition is done by probabilistically fusing the confidence weights derived from each feature space. This method was proved to be quite good as compared to other methods which use only single feature (i.e gray-scale image only or eyes or edginess image only). However, the problem with this approach is computational overhead because it needs to compute 3 different feature spaces. Another interesting combined classifier (called MPL) is proposed by Alok Sharma et al [147]. MPL classifier is a combination of Minimum Distance (MDL), class-dependent PCA and LDA methods.

PCA and Rough Sets Theory:

Pawlak first advocated the rough set theory (RS) as an approach to automatic knowledge acquisition in 1982 [121]. Using RS theory one can find attribute dependencies in database-like information systems, for e.g. a decision table. The basic idea is to compute decision or classification rules through data attribute and attribute-value reductions. RS constrains the data reductions by keeping the discernibility relations among data objects in the table unchanged. Swiniarski and Skowron [154] proposed a method for feature selection that uses Rough Set (RS) theory to the PCA result.

The approach projects the original d-dimensional patterns data, $\mathbf{X}$ into the reduced r-dimensional patterns ($r < d$), $\mathbf{Y}$ in the principal component space. It then makes the reduced and projected data set with real valued attributes discrete and then computes a attribute reduct which comprises of projected features. It is known that, there exists cause-effect relationship between condition and decision attributes in a decision table. The attribute reduct that represents the condition attributes with a larger contribution to cause better than the attribute reduct with smaller contribution. The rule set derived from attribute set with maximal contribution is expected to be more resistant to noise and stronger in its generalization capabilities. To find the contribution of attributes, PCA is one of the well known methods in the pattern recognition literature. Zeng et al [192] proposed an approach KA-RSPCA which is based on PCA in combination with Rough Sets Theory. They used PCA to rank importance of condition attribute by using Cumulative Correlation Coefficient (CCC).

2.8 Methods to Choose Number of Principal Components

PCA is able to retain meaningful information in the early axes whereas variation associated to experimental error, measurement inaccuracy, and/or rounding is summarized in later axes [50]. PCA is able to identify relationships by computing principal components (which are linear combinations of variables) showing common trends of variation can contribute substantially to the recognition or classification of patterns in the data. However, the issue of determining whether or not a given

axis (i.e. principal component) summarizes meaningful variation is not clear in many cases. Please note that when the correct number of non-trivial principal components is not retained for subsequent analysis, either relevant information is lost or noise is included, causing a distortion in underlying patterns of variation/covariation [41][95]. Determining the number of non-trivial principal components remains one of the greatest challenges in providing a meaningful interpretation of multivariate data and has been a long-standing issue in both biological and statistical literature [76].

Methods based on confidence intervals:

Parallel Analysis (PA)[44] involves a Monte Carlo approach to generate a large number of eigenvalues based on simulated data sets that are equivalent in size to the observed data set of interest, and comprises of independent normally distributed variables. These eigenvalues are then used to build confidence intervals for each axis. If observed values exceed the critical value, then we reject the null hypothesis according to the pre-specified significance level. Parallel analysis is based on independent normally distributed data and is parametric. Other methods which are distribution-free include randomization and bootstrap that may give a more robust assessment for non-normal distributions.

Randomization methods based on eigenvalues follow the protocol: (i) randomize the values within variables in the data matrix, (ii) conduct a PCA on the reshuffled data matrix, and (iii) repeat steps (i) and (ii) a total of 999 times. In each randomization, we can evaluate test statistics based on the eigenvalues such as (i) the observed eigenvalue for an axis [159], (ii) a Pseudo-F-ratio is calculated as each eigenvalue divided by the sum of the remaining (or smaller) eigenvalues [160]. Similarly one can

have *randomization methods based on eigenvectors.*

Bootstrap methods based on eigenvalues. Bootstrap confidence intervals for eigenvalues [70] are computed based on resampling the original data with replacement so that the bootstrapped sample is consistent with the original dimensions of the data matrix. 1000 bootstrapped samples are drawn and a PCA is conducted on each of them. There are a number of methods for estimating confidence intervals and one of them is the percentile method [105]. In the similar way, *bootstrap methods based on eigenvectors* compute bootstrap confidence intervals for loadings instead of eigenvalues [122].

Correlation critical values for eigenvectors method tests loadings against the critical values for parametric correlation from standard statistical tables. Any particular axis with at least two significant eigenvector loadings is retained. Other methods include Bartlett‘s test for the first principal component [5], Lawley‘s test for the second principal component [94]. Recently Chen [20] proposed a confidence interval using a stepwise selection procedure for the number of important principal components in PCA. An i^{th} principal component important if λ_i/λ_1 is closer to 1, where λ_1 is the largest eigenvalue.

Methods based on average test statistic values:

Rules based on average values assess whether an observed test statistic based on eigenvalues or eigenvectors is larger than the average value expected under the null hypothesis of no association between variables. According to *Kaiser-Guttman method* [54], when using correlation matrices, population components having eigenvalues larger than 1.0 should be retained. In *Broken-stick* method one assumes that the total

variance in a multivariate data set is divided at random amongst all components, the expected distribution of the eigenvalues can be assumed to follow a broken-stick distribution [98]. The idea underlying the model is that if a stick is randomly broken into p pieces, b_1 would be the average size of the largest piece in each set of broken sticks, b_2 would be the average size of the second largest piece, and so on. If the k^{th} component has an eigenvalue larger than b_k, then the component is retained. *Random average under permutation* method is based on the average eigenvalue obtained under a randomization of the data matrix. If the observed value exceeds the average random value, that particular axis is to be retained. Peres-Neto et al [123] conducted a comparative study 20 stopping rules (to find number of PCs) and found that (i) Random average under parallel analysis, (ii) Random average under permutation, (ii) Parallel Analysis, (iii) $Rnd-Lambda$, (iv) $Rnd-F$ or (v) Minimum Average partial correlation [167] methods perform well as compared to other methods. They also proposed a two-step approach: First, a Bartlett's test is used to test the significance of the first principal component, indicating whether or not at least two variables share common variation in the entire data set. If first PC is significant, a number of different rules can be applied to estimate the number of non-trivial components to be retained. However, the relative merits of these methods depend on whether data contain strongly correlated or uncorrelated variables.

Other methods:

The interpretation of the principal components is generally based on the respective magnitudes of the loadings assigned to the variables. The simplification of the principal components is a great concern for experts in many areas. Vines [169] proposed

a procedure that achieves a simplification of the principal components by seeking approximate components that can be represented by integers. Vigneau and Qannari [168] discussed a strategy of simplification based on cluster analysis of variables. On the similar lines to Jeffers [74], Ledauphin et al [96] proposed a method of hypothesis testing to ascertain the significance of principal components and the variable contributions to the determination of the principal components. If a variable contribution turns out to be non-significant then the loading associated with this variable is set to zero. This process simplifies interpretation of principal components. Hypothesis testing is based on a procedure of simulation by permutations of the rows (each row corresponds to a variable) of the data matrix at hand.

Cadima et al [14] discussed computational aspects of several algorithms for the optimization problems resulting from three different criteria (RM, RV and GCD criteria) in the context of PCA. They found that the local search methods (e.g. local improvement, simulated annealing and genetic algorithms) performed significantly better than the greedy-type algorithms (e.g. forward selection, backward elimination and stepwise algorithms with a default forward or backward direction).

2.9 Comparison of PCA with Other Feature Extraction Methods

PCA versus LDA:

In the context of the appearance-based paradigm for object recognition, it is generally understood that LDA based algorithms are superior to those of PCA based algorithms

because LDA deals with class discrimination. However, empirical evidence by Martonez and Kak [108] shows that this is not always the case and PCA can outperform LDA when the training data set per class is small. It is also showed that PCA is less sensitive to different training data sets. Beveridge et al [10] compared Nearest Neighbor classifiers using principal component and linear discriminant subspaces using different choices of distance metric. They computed probability distributions for algorithm recognition rates and pairwise differences in recognition rates using a permutation methodology. They found that the principal component subspace with Mahalanobis distance performed well and next better performance is using $L2$ distance metric. Linear discriminant subspace is less sensitive to the choice of distance metric, and its performance is always worse than the principal components classifier using either Mahalanobis or L1 distance. Probability distributions for recognition rates and differences in recognition rates relative to different choices of gallery and probe images have been created using a Monte Carlo sampling method. Other comparisons between PCA and LDA may be found in [7] [9].

PCA versus ICA:

Yang et al [188] investigated the two architectures of ICA for image representation and found that ICA Architecture-I involves a PCA process by vertically centering (PCA-I), while ICA Architecture-II involves a whitened PCA process by horizontally centering (PCA-II). These two PCA versions are used as baseline algorithms to evaluate the ICA-based face recognition systems. They found through their experimentation on FERET face data that there is no significant performance differences between ICA Architecture-I (II) and PCA-I (II), and ICA Architecture-II significantly

outperforms the standard PCA. Also the recognition performance of ICA, whether using Architecture-I or II, strongly depends on its involved PCA process (PCA-I or II). The pure ICA projection seems to have little effect on the performance of face recognition. However, Baek et al [2] have tested three different distance metrics L1 norm, L2 norm, and cosine angle - for both PCA and ICA. Baek et al found, contrary to previous reports in the literature, that PCA significantly outperforms ICA when the best performing distance metric (L1 norm in this case) is used for each method. In another development, Fortuna et al [42] compared PCA, ICA, KPCA and FLD with respect to accuracy of visual position measurement and they examined to see the ability of the methods to discriminate positions in a 2D visual subspace. The comparison is done both constant and varying illumination and random occlusion. It was shown that PCA provides overall good performance compared with more sophisticated techniques such as ICA, FLD, and KPCA at a reduced computational complexity. Connie et al [27] performed comparison of PCA, ICA, LDA and Wavelet method on palmprint data and found that application of LDA on wavelet sub-band is able to yield low FAR and FRR rates.

2.10 Some Applications of PCA

Face tracking and recognition:

Turk and Pentland [164] presented an eigenfaces approach to develop a near-real-time face recognition system which tracks subject's head and recognizes the person by comparing the face of the person with those known individuals. Face images are projected onto face space, that best encodes the variation among the faces. The face

space is given by eigenvectors of the training set of faces. A statistical assessment of subject factors was done by Givens et al [52] in PCA recognition of human faces. This study considered 11 factors that might make recognition easy or difficult for 1072 human subjects in the FERET dataset. The factors considered include race (white, Asian, African-American, or other), gender, age (young or old), glasses (present or absent), facial hair (present or absent), etc,. An ANOVA is used to determine the relationship between these subject covariates and the distance between pairs of images of the same subject in a standard *eigenfaces subspace*. Some outcomes of their study include (i) the distance between pairs of images for subjects decreases for people who consistently wear glasses, so wearing glasses makes subjects more recognizable, (ii) Pair-wise distance also decreases for people who are either Asian or African-American rather than white.

Astronomical applications:

Discrimination of Giant and Dwarf Spectra in K-stars. Ibata et al [67], used a variant of PCA for discrimination problems in astronomy. They have presented the problem of discrimination between K-giant and K-dwarf stars from intermediate resolution spectra near the Mg 'b' feature. For the highest S/N spectra, the automated classification agrees very well (at the $90-95\%$ level) with the visual classification.

PCA of the Lick indices of galactic globular clusters. Strader and Brodie [152] applied PCA of high-quality Lick/IDS absorption-line measurements for 11 indices in the wavelength range $4100-5400A$ for 39 galactic globular clusters (GCs). Only the first principal component appears to be physically significant. It was found that there is a tight linear relationship between this first component (PC1) and GC metallicity

over a wide range in $[m/H]$ ($-1.8 \leq [m/H] \leq 0$), suggesting that PC1 can be used to accurately estimate metallicities for old extra galactic GCs from their integrated spectra. It was found that little evidence for substantial differences in broad abundance patterns among galactic GCs.

PCA of speech spectrogram images:

The sound spectrogram is a commonly used three-dimensional (time-frequency-intensity) representation of an acoustic signal. Fourier descriptors (FDs) have been proved very useful for characterizing the boundary of segmented isolated words containing the English semi-vowels /w/, /y/, /l/, and /r/. Pinkowski [125] investigated the relevance of 16 32-point FDs combined with 17 other general features (to characterize non-shape parameters) for classifying objects contained in binary spectrogram images. PCA is used for reducing dimensions on a speaker-dependent data set consisting of 80 sounds representing 20 speaker-dependent words containing English semi-vowels. With only eight features, including four 32-point FDs and four general features obtained from PCA, a 97.5% recognition rate is obtained. The appropriateness of shape descriptors alone for classifying spectrogram objects may be enhanced if they are combined with other features, particularly those containing information on orientation (principal axes). Orientation features can be obtained from the eigenvector and chain code representations on binary objects.

Pattern classification using PCA and fuzzy rule bases:

Ravi et al [134] have used PCA to get principal components, which are subsequently fed in fuzzy rule based classifier as new set of features. This process has given a very high classification rate of 100% in leave-one-out technique with a few rules in the case

of some aggregators. The chosen principal components accounted for only 89% and 92% of the total variance in Wine and Breast cancer data sets respectively. Using this study as an evidence, PCA can be used as a useful alternative to other methods of feature selection existing in the literature while solving classification problems of higher dimensions using fuzzy rule based classifiers.

PCA-based branch and bound search algorithms for computing k nearest neighbours: Searching the k nearest neighbors in a multi-dimensional vector space is a very common phenomenon in pattern recognition. Recently, several branch and bound search algorithms were proposed that use a decomposition method based on PCA. These algorithms search the nearest neighbors in a vector space where the dissimilarity between two vectors is expressed by the Euclidean distance. It was shown that these algorithms have a linear space complexity, an average number of distance computations bounded by a constant term and a time complexity that is very close to logarithmic for a small number of dimensions. The most important aspects that influence the efficiency of the search algorithm are: (i) the decomposition method, (ii) the elimination rule, (iii) the traversal order and (iv) the level of decomposition. A theoretical derivation of an efficient decomposition method based on PCA is given by Dohaes et al [55]. Then, different elimination rules and traversal orders are combined resulting in different search algorithms.

PCA to facilitate fast detection of transient-evoked otoacoustic emissions: Transient-evoked otoacoustic emissions (TEOAE) are acoustic signals coming from the inner ear (outer hair cells of the cochlea) after acoustic stimulation by clicks and tone-bursts. These responses can be recorded from the ear canal of all normal adults,

children, and neonates shortly after birth, and are used as a clinical test to assess the integrity of the peripheral organ in TEOAE based newborn hearing screening programs. Some of their potential applications (e.g., their use as a tool in newborn hearing screening programs) are deeply related to the duration of each recording session. This duration can be strongly reduced by applying PCA approach to a set of TEOAE recorded from the same ear at different stimulus levels averaging only a few sweeps (a maximum of 100 versus the classical 260). The PCA approach used here is able to enhance the signal-to-noise ratio and, in turn, to allow a correct detection of the responses. The application of the PCA approach to a set of TEOAE recorded at different stimulus levels reduces on average the acquisition time of TEOAE to about one fourth of the time with the classical procedure. The comparison between the Similitude values provided statistical evidence that the PCA approach produces no loss of information in the set of data in terms of similarity between the rapidly acquired PCA-processed set and the GS set. The use of the PCA approach statistically improves the reproducibility of the set of data both for 60- and 100-sweep averaged data and the PCA approach improves dramatically the identification of the response in these conditions [133].

Machine Defect Classification:

Sensor-based machine condition monitoring has gained increasing attention from the research community world-wide. The goal of machine condition monitoring is to obtain operational status of the machines and use the information to (i) identify potential machine faults and failure before they occur, thus reducing unexpected and costly machine downtime, and (ii) better control the quality of products, which is

closely related to the condition of the machine. The information gathered from the monitoring sensors ultimately provide insight into the manufacturing process itself, enabling effective high-level decision-making for quality production at a lower cost. The PCA-based feature selection scheme for machine condition monitoring is based on the understanding that the amplitude of vibration signals of defective machine components increases as the severity of the defect increases. The issue of feature selection from a contending feature set arises, because of the stochastic nature of the defect propagation in machinery. Generally, as the defect severity increases, an overall increasing vibration trend is superimposed by local variations of smaller magnitudes. The goal of feature selection is therefore to select features that allow for an accurate description of the defect condition, and subsequently, reliable defect classification, diagnosis, and prognosis. The PCA approach was developed to reduce the dimensionality of the input features for both supervised and unsupervised classification purposes [104].

PCA to improve fault detection and classification (FDC) performance:

Yue et al [191] proposed sample-wise weighted PCA and variable-wise weighted PCA. Sample-wise weighted PCA is used to address issues with model updating. By adapting models with process changes, the long-term validity of a PCA model can be maintained. Variable-wise weighted PCA is used to incorporate process and sensor knowledge. PCA models built this way require less maintenance and result in better FDC performance. Yue et al [191] performed comparison studies on plasma etcher FDC and had shown that weighted PCA can adapt to process drift, reduce the occurrence of false alarms and make the models easy to maintain.

Computer-aided drug design:

Molecular similarity, as an important tool of computer-aided drug design has developed rapidly. Its calculation has also been developed from planar, rigid, 2D molecules to steric, flexible, 3D molecules. However, 3D molecular similarity calculation is easy to fall into local optima and the calculation is always time-consuming. Xian et al [179] proposed a method of flexible 3D molecular similarity calculation through the evaluation of molecular electrostatic potentials (MEP) with PCA, genetic algorithm (GA) and tabu search (TS). PCA is used to preprocess, GA is used to align two molecules and TS is used to decrease the probability of falling into local optima. The authors calculated the molecular similarities of benzene and its derivatives, a group of insecticides and a series of acetylcholinesterase inhibitors.

Discrimination of varieties of tea using near infrared spectroscopy by PCA and BP model:

Visible/near-infrared (Vis/NIR) spectroscopy , with the characteristics of high speed, non-destructiveness, high precision and reliable detection data, etc., is a pollution-free, rapid, quantitative and qualitative analysis method. A new approach for discrimination of varieties of tea by means of Vis/NIR spectroscopy $(325-1075nm)$ was developed by Yong He et al [58]. In this approach, the spectral data is compressed by the wavelet transform (WT). The features from WT can be visualized in principal component (PC) space, which can lead to discovery of structures correlative with the different class of spectra samples. It appears to provide a reasonable clustering of the varieties of tea. The scores of the first eight principal components computed by PCA had been applied as inputs to a back propagation neural network with one hidden

layer. The 200 samples of eight varieties were selected randomly to build BP-ANN model. This model is used to predict the varieties of 40 unknown samples. The recognition rate of 100% is achieved.

PCA-based web page watermarking:

The tamper-proof of web pages is of great importance. Some watermarking schemes have been reported to solve this problem. However these watermarking schemes and the traditional hash methods have a problem of increasing file size. Zhao and Lu [198] proposed a novel watermarking scheme for the tamper-proof of web pages, which is free of this embarrassment. For a web page, the proposed scheme generates watermarks based on PCA technique. PCA is applied on a matrix produced from a web page and a secret key. The watermarks are then embedded into the web page through the upper and lower cases of letters in HTML tags. When a watermarked web page is tampered, the extracted watermarks can detect the modifications to the web page, thus we can keep the tampered one from being published.

Remote Sensing Applications:

Aerosols are liquid and solid particles suspended in the air from natural or man-made sources. They can affect human health, visibility, and climate. Aerosol particles affect climate directly by reflecting and absorbing solar and terrestrial radiation, and indirectly by their influence on cloud micro-physics. Zubko et al [200] applied PCA to estimate how much information about atmospheric aerosols could be retrieved from solar-reflected radiance observed over oceans by a satellite sensor as a function of the number of wavelength bands, viewing angles, and stokes parameters. The following quantities are used to vary: aerosol optical thickness, single-scattering albedo (SSA)

of aerosol particles, height of the aerosol layer, aerosol model (includes size distribution parameters and optical properties), and wind speed. The real refractive index is kept constant and, therefore, is not part of the analysis. To calculate the number of significant principal components (PCs), the cumulative percent variance rule is used, which takes into account anticipated errors of measurements. The reported results predict how much additional information can be retrieved from observations by adding more wavelength, angle, and polarization channels. For example, for the moderate resolution Imaging Spectro-Radiometer instruments, the number of significant PCs is 2 to 3; for multi-angle Imaging Spectro-Radiometer, 3 to 5, etc. The calculations show that the observations should be most sensitive to the aerosol model followed in decreasing order by optical thickness, SSA, and aerosol height. It is found that there is no systematic increase in the information about aerosol starting from $10 - 15$ view angles for unpolarized observations and 30 view angles for those with linear polarization. It is achievable with modern detectors to retrieve up to 10 and 16 significant PCs from unpolarized and polarized observations respectively. The methodology and results of PCA can be useful for estimating the reliability of aerosol parameters retrieved from existing and future satellite observations.

A discussion of PCA in remote sensing may be found in [6].

2.11 How do the Existing PCA Methods Address the Problems of Classical PCA?

Here we briefly review certain problems and issues faced by classical PCA (Section 1.3 of Chapter 1). We see how the several methods we have surveyed in the literature attempt to address these problems.

1. *Addressing high computational complexity of PCA.* Neural Network based methods (Section 2.4) are suitable for high-dimensional data because they compute principal component directions incrementally *without computation of covariance matrix*. EM algorithms also learn eigenvectors adaptively without computation of covariance matrix (Section 2.6). Other methods which do not compute covariance matrix include Covariance-free Incremental PCA [176] and Simple PCA [120]. More recently, 2DPCA methods (Section 2.3) have become popular for image data. 2DPCA methods compute more compact covariance matrix which needs less computations than traditional computation. More interestingly, Feature partitioning based PCA (FP-PCA) methods (Section 2.2) consume less computational requirements by computing sub-covariance matrices using sub-patterns.

2. *Addressing poor performance with data of prominent local variations (local feature extraction).* Most of the existing PCA methods (except FP-PCA methods) are based on extracting features from whole patterns, which form the basis for global feature extraction. These methods may work well in some situations (where variations spread across entire pattern), however, may not perform well

when the variations are limited to a part or block of patterns. FP-PCA methods (Section 2.2) were proved to be superior over classical PCA methods when the variations are restricted to a part or block of a pattern (i.e when local variations are prominent). FP-PCA methods divide each pattern into sub-patterns or blocks and extract local principal components from these blocks. FP-PCA methods form the basis for local feature extraction.

3. *Addressing Small Sample Size (SSS) problem.* If the number of samples are small as compared to dimensionality of the patterns, the feature extraction itself may not be effective. Some of the methods which reduce the SSS problems include (i) FP-PCA methods (Section 2.2)– these methods divide each pattern into sub-patterns and feature extraction is done from these sub-patterns (blocks) instead of whole pattens. It is clear that sub-pattern dimensionality (u) is less than pattern dimensionality (d), therefore reducing SSS problem, (ii) 2DPCA methods (Section 2.3) reduce SSS problem by computing more compact $n \times n$ covariance matrix instead of huge $m.n \times m.n$ matrix (m, n are dimensions of an image matrix). Due to compact covariance matrix, the method requires to compute less number (i.e. n instead of traditional $m.n$) of principal component vectors.

4. *Addressing the problem of handling missing data and outliers.* Many robust methods such as Fuzzy logic based, Neural Network based methods, etc, are proposed to handle outliers data while computing principal components [143] [30] [68] [32] [61] [77] [18] [151]. For addressing the data with missing values many methods were proposed [62] [101] [18] [148].

5. *Addressing the issue of non-linear data.* PCA methods with *kernel trick* can effectively capture non-linearity (Section 2.5). Non Linear Neural Networks (NLNN) are also proved to be useful to capture non-linearity in the patterns (Section 2.4).

6. *Addressing the issue of choosing number of Principal Components.* Right choice of principal components influence the classifier performance as well as total amount of variance (structure) in the reduced data. Many methods are proposed (Section 2.8) in the literature to choose number of PCs.

2.12 What is the Problem We are Solving?

In our work, we investigate FP-PCA methods as reviewed in section 2.2. The motivation for our study on FP-PCA methods is due to their interesting and novel characteristics. In contrast to other PCA methods which are based on whole patterns, FP-PCA methods are unique in their approach, which are based on novel idea of *partitioning each pattern into sub-patterns (blocks) and extract local features.*

FP-PCA methods alleviate some of the crucial problems of classical PCA (i.e. whole-pattern based (global) PCA). FP-PCA methods have the following advantages over classical PCA: (i) Reduced computational complexity, (ii) Improved recognition/classification rate by local feature extraction if local variations are prominent, (iii) Reduced small sample size problem. However, FP-PCA methods as seen in the literature suffer from the following problems:

1. They do not retain the original essence (or merits) of classical PCA.

2. These methods purely perform local feature extraction (i.e. feature extraction is limited to a subset of original feature set (or limited to sub-patterns)). Therefore, FP-PCA methods may not perform well if there exists global variations (or strong correlations between local features extracted from the sub-patterns). Please note that classical PCA (global PCA or whole-pattern based PCA method) works well in this case.

3. Summarization of variance is not good because the entire covariance structure is not utilized which yields more number of local PCs resulting in less dimensionality reduction. Please note that classical PCA (global PCA or whole-pattern based PCA method) has good summarization of variance, because it makes use of entire covariance structure.

4. FP-PCA methods do not exploit two-dimensional structure of image data (Section 2.3) because they all use classical PCA in a region or block to extract local features. In classical PCA, each sub-image is formed as a feature vector, thus collapsing the two dimensional matrix structure of image.

From our analysis, we understand that FP-PCA methods and classical PCA methods (i.e. global PCA or whole-pattern based PCA methods) are complementary approaches. Put it in other way, FP-PCA methods work well in some cases, in which case classical PCA (global or whole-pattern base PCA) may not perform well and vice versa.

The *objectives* of our investigation presented in this thesis are given as follows.

2.12.1 Objectives of Our Investigation in this Thesis

1. From our study, we understand that there is no conceptual study of FP-PCA methods. Thus there is no conceptual basis to understand the nuances of these methods like (i) What are the issues that arise due to partitioning of the patterns, (ii) Is there any impact on dimensionality reduction because of partitioning of the patterns?.

   (a) Can we perform rigorous investigation on FP-PCA methods systematically and propose a generalized framework and issues?

   These concerns are addressed in *Chapter 3*.

2. To propose a novel FP-PCA method which alleviates the crucial problems of both (i) classical PCA and (ii) FP-PCA methods and exploits the strengths of both the methods.

   (a) Less computational complexity (as with FP-PCA methods)

   (b) Reducing SSS problem (as with FP-PCA methods)

   (c) Good classification performance (Good generalization ability) when local variations are dominant (as with FP-PCA methods)

   (d) Good classification performance (Good generalization ability) when global variations are dominant (as with classical PCA methods)

   (e) Less number of coefficients or components, that is good summarization of variation or high dimensionality reduction (as with classical PCA methods)

   These concerns are addressed in *Chapter 4*.

3. To propose some novel FP-PCA methods exclusively for image data (by taking ideas of using feature partitioning and matrix structure of image data), which show

   (a) Less computational complexity (as with FP-PCA methods and better than 2DPCA)

   (b) Reducing SSS problem (as with FP-PCA methods and better than 2DPCA method)

   (c) Good classification performance (Good generalization ability) when local variations are dominant (as with FP-PCA methods)

   (d) Good classification performance (Good generalization ability) when global variations are dominant

   These concerns are addressed in *Chapter 5*.

4. No theoretical study on FP-PCA methods is found in the literature. Can we perform a theoretical analysis of FP-PCA methods which brings out

   (a) Deeper insight of FP-PCA methods and establish links to classical PCA (global PCA)

   (b) General properties of FP-PCA methods

   These concerns are addressed in *Chapter 6*.

5. We are aware of that FP-PCA methods show their superiority as compared to classical PCA methods. Cluster analysis is a well established tool for data analysis in pattern recognition and data mining.

(a) Can we extend the ideas of feature partitioning to improve Cluster Analysis?

These concerns are addressed in *Chapter 7*.

6. We know that subspace classification is one of the traditional approaches in pattern recognition and integrates feature extraction and classification in a seamless fashion.

   (a) Can we extend the ideas of feature partitioning to improve subspace classification?

   These concerns are addressed in *Chapter 8*.

In this thesis, we do not address the following issues: (i) The issue of non-linear data, (ii) Choosing number of principal components and (iii) Handling missing values data and outliers.

2.13 Summary

In this section, we reviewed the state-of-the-art of PCA literature, which include FP-PCA methods, Two-dimensional PCA methods, Neural Network based PCA methods, KPCA methods, etc. Subsequently, we discussed how these PCA methods address the problems faced by classical PCA. In this work, we focus on study of FP-PCA methods. In subsequent Chapters (i) we propose a common framework for FP-PCA methods and identify issues to be addressed, (ii) we bring out some novel FP-PCA methods, which alleviate the problems faced by both the existing FP-PCA

methods and classical PCA (global PCA) methods, (iii) we establish general properties of FP-PCA methods by performing a theoretical analysis and (iv) we extend our feature partitioning ideas to cluster analysis and subspace classification.

In the next Chapter, we start our journey by proposing a framework which brings the existing FP-PCA methods under a common framework and identify the issues need to be addressed in this framework. This framework forms the basis for our work.

Chapter 3

Generalized Feature Partitioning Framework and Issues

3.1 Introduction

Note: The work in this chapter has been published partly in *Pattern Recognition Journal (Elsevier Science)*[1] and in *proceedings of IVCNZ 2005 Conference*[2].

Many feature partitioning based PCA (FP-PCA) techniques were proposed in the literature. However, there is no work (to the best of author's knowledge) which has focussed on study of FP-PCA techniques in general. In this chapter, we unify the existing FP-PCA approaches under a common framework known as, 'generalized feature

[1]Kadappagari Vijaya Kumar and Atul Negi, "SubXPCA and a generalized feature partitioning approach to principal component analysis", *Pattern Recognition*, Vol. 41, No. 4, Apr. 2008, pp. 1398-1409.

[2]Atul Negi and Kadappagari Vijaya Kumar, "An experimental study of sub-pattern based principal component analysis and cross-subpattern-correlation based principal component analysis (SubXPCA)", *In Proceedings of Image and Vision Computing Conference (IVCNZ-2005)*, New Zealand, pp. 20-25, Nov. 28^{th}-29^{th} 2005.

partitioning framework to PCA'. Using the framework, we understand the existing FP-PCA approaches more comfortably. Further we bring out various fundamental issues such as

1. How to partition a pattern?
2. How many sub-patterns (blocks)?
3. How to group the sub-patterns?
4. Which PCA variation is to be used to extract local features from groups of sub-patterns?
5. How to combine local features extracted from sub-patterns?
6. How to address the loss of dependency information (correlation) due to partitioning? etc.

We analyze the various FP-PCA techniques addressing each of the issue proposed here. We elaborate a generalized feature partitioning framework and the issues to be addressed related to the framework in subsequent sections. This chapter provides an abstract generalization of the concepts for which instances are taken up in subsequent chapters.

The rest of the chapter is organized as follows. In section 3.2 we present a generalized framework to feature partitioning based PCA approaches. We bring out various feature partitioning issues in section 3.3.

3.2 Generalized Feature Partitioning Framework

In this section we explain the concept of a feature partitioning framework to PCA computation in detail. We bring out the various feature partitioning issues such as Partitioning procedure, Choice of number of blocks (block size), Deciding to have overlapping blocks, Grouping blocks, Choice of local feature extraction method, Combining local features, Feature order dependency, Loss of inter-sub-pattern dependencies (Inter-sub-pattern correlations), Truncation/padding up of features and Selection of principal components. We also discuss how the existing feature partitioning methods address these issues. More importantly, the framework we discuss here may be applied to any feature extraction method, however, we restrict it to PCA computation in this thesis.

3.2.1 Unified Framework Idea

We present the general feature partitioning idea as follows.

Step 0: Pre-processing patterns

This step is optional. However, it may be used to do pre-processing of patterns such as cropping image patterns, enhancing images, warping images, localizing images etc.

Step 1: Partitioning (dividing) patterns

In this step, we divide each pattern into $k\,(\geq 2)$ sub-patterns (blocks). We may explore many possibilities to divide a pattern. This step focuses on producing a subset

of original features, which finally form the basis for constituting sub-patterns. Please note that a j^{th} sub-pattern of every pattern contains the same feature variables, only feature values may differ.

Definition 1 A Sub-pattern or a Block.

A Sub-pattern or a block is a proper subset of set of dimensions of a pattern $\mathbf{X}_i$*. There may be many blocks of a pattern* $\mathbf{X}_i$*, denoted by* $\mathbf{X}_i^1, \mathbf{X}_i^2, \ldots, \mathbf{X}_i^k$*. In other words,* $\mathbf{X}_i^j$ *is a sub-pattern, if* $\mathbf{X}_i^j \subset \mathbf{X}_i$*.*

Examples of sub-patterns of a pattern vector $\begin{bmatrix} 10 \\ 20 \\ 30 \\ 40 \end{bmatrix}$ *are:* $\begin{bmatrix} 10 \\ 20 \end{bmatrix}, \begin{bmatrix} 30 \\ 40 \end{bmatrix}, \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix}$ *etc.*

Step 2: Grouping sub-patterns (blocks)

There may be many ways of grouping blocks which enable feature extraction local to these groups.

Definition 2 Sub-pattern set or Block set.

Let each pattern $\mathbf{X}_i$*;* $i = 1, \ldots, N$ *be divided into* k $(k \geq 2)$ *sub-patterns,* $\mathbf{X}_i^1, \mathbf{X}_i^2, \ldots, \mathbf{X}_i^k$*. A set of sub-patterns is called sub-pattern set. A sub-pattern set may be homogeneous or heterogeneous.*

A homogeneous sub-pattern set *is the one which contains similar sub-patterns from the training patterns. Two sub-patterns are said to be similar if they share same feature variables.*

A j^{th} *homogeneous sub-pattern set,* $\mathbf{P}^j$,$j \in \{1, \ldots, k\}$ *is given as*

$$\mathbf{P}^j = \{\mathbf{X}_1^j, \mathbf{X}_2^j, \ldots, \mathbf{X}_N^j\} \tag{3.1}$$

A heterogeneous sub-pattern set *is the one which contains dissimilar sub-patterns from the given training patterns. Two sub-patterns are said to be dissimilar if they have different feature variables. A heterogeneous sub-pattern set,* $\mathbf{Q}$*, is given as*

$$\mathbf{Q} = \{\mathbf{X}_1^1, \mathbf{X}_2^1, \ldots, \mathbf{X}_N^1, \mathbf{X}_1^2, \mathbf{X}_2^2, \ldots, \mathbf{X}_N^2 \ldots\ldots, \mathbf{X}_1^k, \mathbf{X}_2^k, \ldots, \mathbf{X}_N^k\} \tag{3.2}$$

$$\mathbf{Q} = \mathbf{P}^1 \cup \mathbf{P}^2 \cup \ldots \cup \mathbf{P}^k \tag{3.3}$$

Examples of homogeneous and heterogeneous sub-pattern sets for patterns $\begin{bmatrix} 10 \\ 20 \\ 30 \\ 40 \end{bmatrix}$, $\begin{bmatrix} 1 \\ 3 \\ 5 \\ 7 \end{bmatrix}$, $\begin{bmatrix} 2 \\ 4 \\ 6 \\ 8 \end{bmatrix}$ *are: Let each pattern be divided into* 2 *sub-patterns.*

$$\mathbf{P}^1 = \left(\begin{bmatrix} 10 \\ 20 \end{bmatrix}, \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \begin{bmatrix} 2 \\ 4 \end{bmatrix} \right)$$

$$\mathbf{P}^2 = \left(\begin{bmatrix} 30 \\ 40 \end{bmatrix}, \begin{bmatrix} 5 \\ 7 \end{bmatrix}, \begin{bmatrix} 6 \\ 8 \end{bmatrix} \right)$$

$$\mathbf{Q} = \left(\begin{bmatrix} 30 \\ 40 \end{bmatrix}, \begin{bmatrix} 5 \\ 7 \end{bmatrix}, \begin{bmatrix} 6 \\ 8 \end{bmatrix}, \begin{bmatrix} 10 \\ 20 \end{bmatrix}, \begin{bmatrix} 1 \\ 3 \end{bmatrix}, \begin{bmatrix} 2 \\ 4 \end{bmatrix} \right)$$

Please note that $\mathbf{P}^1$, $\mathbf{P}^2$ *are homogeneous sub-pattern sets and* $\mathbf{Q}$ *is heterogeneous sub-pattern set.*

Definition 3 Intra-block (Intra-sub-pattern) covariance matrix of a homogeneous block set, $\mathbf{P}^j$.

Assume that each sub-pattern contains u dimensions. Intra-block covariance matrix of a homogeneous block set, $\mathbf{P}^j$*, is given by*

$$(\mathbf{C}^j)_{u\times u} = \frac{1}{N}.\sum_{i=1}^{N}(\mathbf{X}_i^j - \mu^j)_{u\times 1}.(\mathbf{X}_i^j - \mu^j)_{1\times u}^T \tag{3.4}$$

where μ^j *is the mean of* $\mathbf{P}^j$.

Intra-block (Intra-sub-pattern) covariance matrix of a heterogeneous block set, $\mathbf{Q}$ *is given by*

$$(\mathbf{C_Q})_{u\times u} = \frac{1}{N.k}.\sum_{j=1}^{k}\sum_{i=1}^{N}(\mathbf{X}_i^j - \mu)_{u\times 1}.(\mathbf{X}_i^j - \mu)_{1\times u}^T \tag{3.5}$$

where μ *is the mean of* $\mathbf{Q}$.

Definition 4 Inter-block (Inter-sub-pattern) covariance matrix.

Inter-block covariance matrix of two block sets, $\mathbf{P}^q, \mathbf{P}^t$*;* $q \neq t$ *is given by*

$$(\mathbf{C}^{q,t})_{u\times u} = \frac{1}{N}.\sum_{i=1}^{N}(\mathbf{X}_i^q - \mu^q)_{u\times 1}.(\mathbf{X}_i^t - \mu^t)_{1\times u}^T \tag{3.6}$$

where $\mu^{\mathbf{q}}$, $\mu^{\mathbf{t}}$ *are the means of* $\mathbf{P}^q$, $\mathbf{P}^t$ *respectively.*

Step 3: Extraction of local features

Local features are to be extracted from each group of blocks (sub-pattern set) using some variation upon the Principal Component Analysis method. This step extracts more informative, less noisy features within each group of blocks (sub-pattern set). In principle, we can use any Principal Component Analysis variation for local feature extraction.

Definition 5 Local features.

Local features are those features which are extracted using an intra-block covariance matrix of a homogeneous block set, $\mathbf{P}^j$ *or a heterogeneous block set,* $\mathbf{Q}$.

In other words, *Local features* are those features extracted from sub-patterns (blocks), not from whole patterns.

Step 4: Unifying the local features

Finally we need to combine the local features using some procedure which forms reduced patterns. These reduced patterns may be used for subsequent tasks such as classification, transmission, clustering, etc.

The block diagram of the proposed feature partitioning framework is shown in Fig. 3.1. Now we discuss the issues to be addressed in each of these steps.

3.3 Feature Partitioning Issues

In this section, we explain the issues related to feature partitioning framework in detail. Further, we analyze the existing FP-PCA approaches to understand how they address the proposed issues.

3.3.1 Partitioning a Given Pattern

Partitioning a pattern should minimize the loss of information due to partitioning, and/or improve the subsequent classification. The issue here is to choose a procedure to divide a given pattern. Some methods include (i) dividing a pattern by choosing features (equal or different number of features) contiguously in the order of

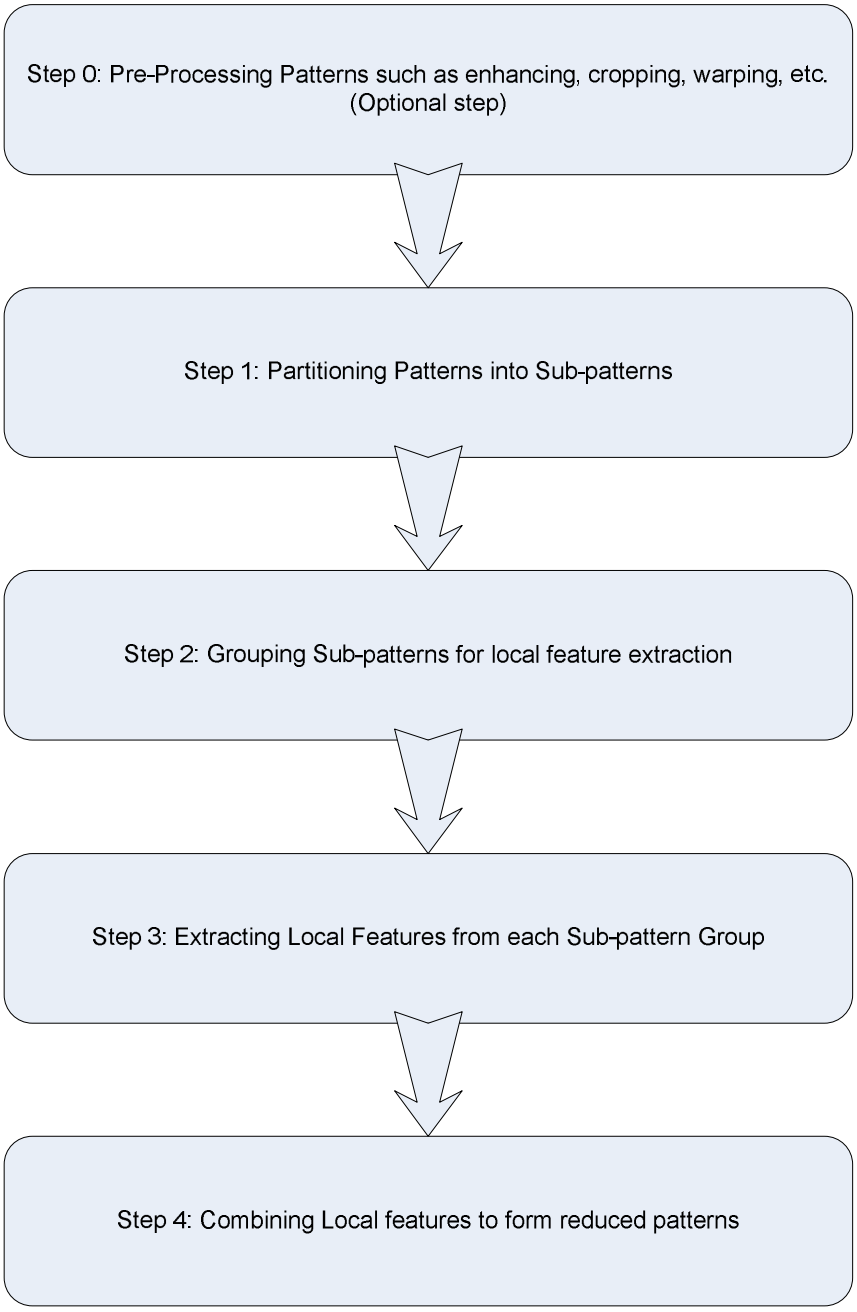


Figure 3.1: Steps in generalized feature partitioning framework.

appearance, e.g. SubPCA [21] (Fig. 3.2), (ii) dividing a pattern by choosing features randomly (Fig. 3.3), (iii) if pattern is an image, we may divide it in different ways such as vertical division or horizontal division or both, e.g. Modular PCA [53], Sub-Holisitic PCA [80], Multi-Block PCA [128], Adaptively weighted SubPCA [157], (iv) another method is to use image segmentation techniques to have meaningful blocks (regions), which do not have any predefined shape, e.g. Eigen-regions method [45] (Fig. 3.4(b)). However, these techniques may have additional computational overhead for segmentation and (v) another option is to divide patterns (e.g. faces) using an elliptical curve, e.g. Localized PCA [106] [107].

3.3.2 Selection of Block Size or Number of Blocks

While partitioning, a question arises about what should be the sub-pattern (block) size or equivalently, how many blocks to have?. A simple method is to have all k sub-patterns the same fixed arbitrary size. If $k = 1$, then FP-PCA approaches and classical PCA (global PCA) methods are identical. Some judgement is required to select appropriate sub-pattern size since its size determines the kind of correlations (dependencies) that can be observed. Choosing an appropriate sub-pattern size which gives optimal performance is an open issue in feature partitioning framework. Also, we need to decide whether to consider all the blocks or to discard some blocks. Most of the existing FP-PCA methods use some adhoc methods for this task. Sub-Holistic PCA [80] uses the option of dividing an image (face) into fixed 4 equally-sized parts. Modular PCA [53] crops the original image into $m \times m$ size, where m is a power of 2 and divides into equally-sized blocks. Eigen-regions method [45] divides every

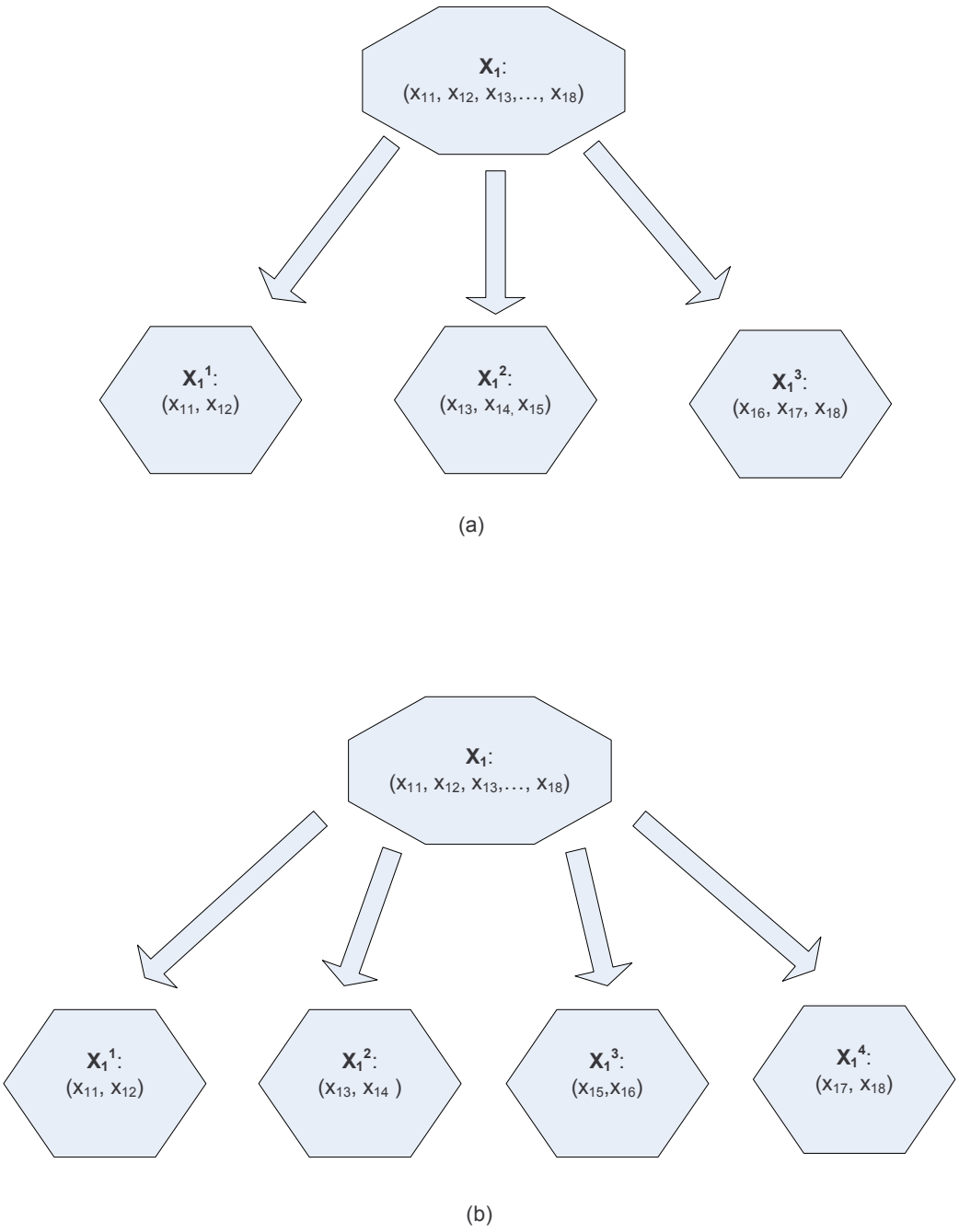


Figure 3.2: *Partitioning by contiguous selection of features.* (a) Partitioning a pattern, $\mathbf{X}_1$ into sub-patterns (blocks), $\mathbf{X}_1^1$, $\mathbf{X}_1^2$, $\mathbf{X}_1^3$ of different sizes. (b) Partitioning a pattern, $\mathbf{X}_1$ into sub-patterns (blocks), $\mathbf{X}_1^1$, $\mathbf{X}_1^2$, $\mathbf{X}_1^3$, $\mathbf{X}_1^4$, of same size.

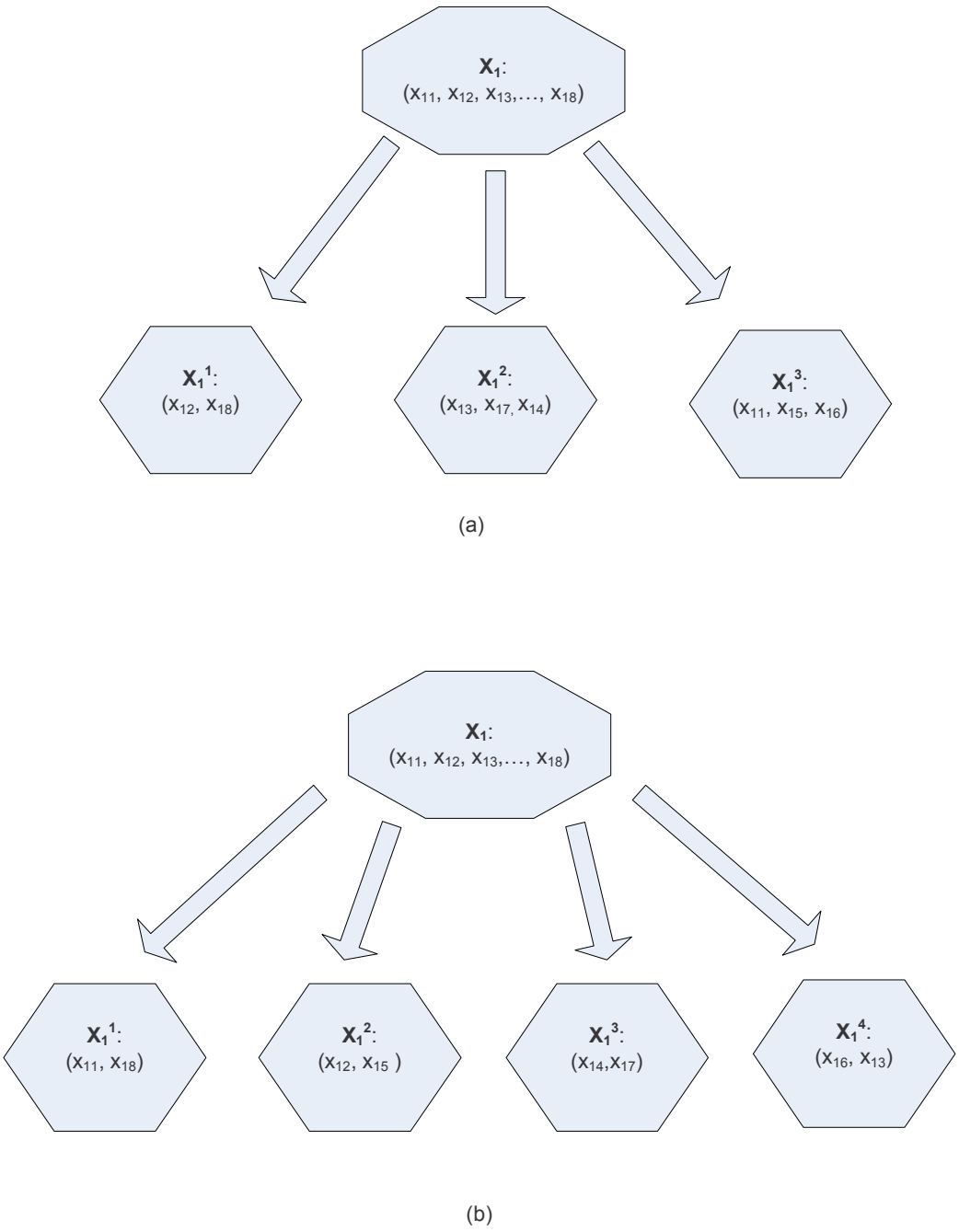


Figure 3.3: *Partitioning by random selection of features.* (a) Partitioning a pattern, $\mathbf{X}_1$ into sub-patterns (blocks), $\mathbf{X}_1^1$, $\mathbf{X}_1^2$, $\mathbf{X}_1^3$, of different sizes. (b) Partitioning a pattern into sub-patterns (blocks), $\mathbf{X}_1^1$, $\mathbf{X}_1^2$, $\mathbf{X}_1^3$, $\mathbf{X}_1^4$, of same size.

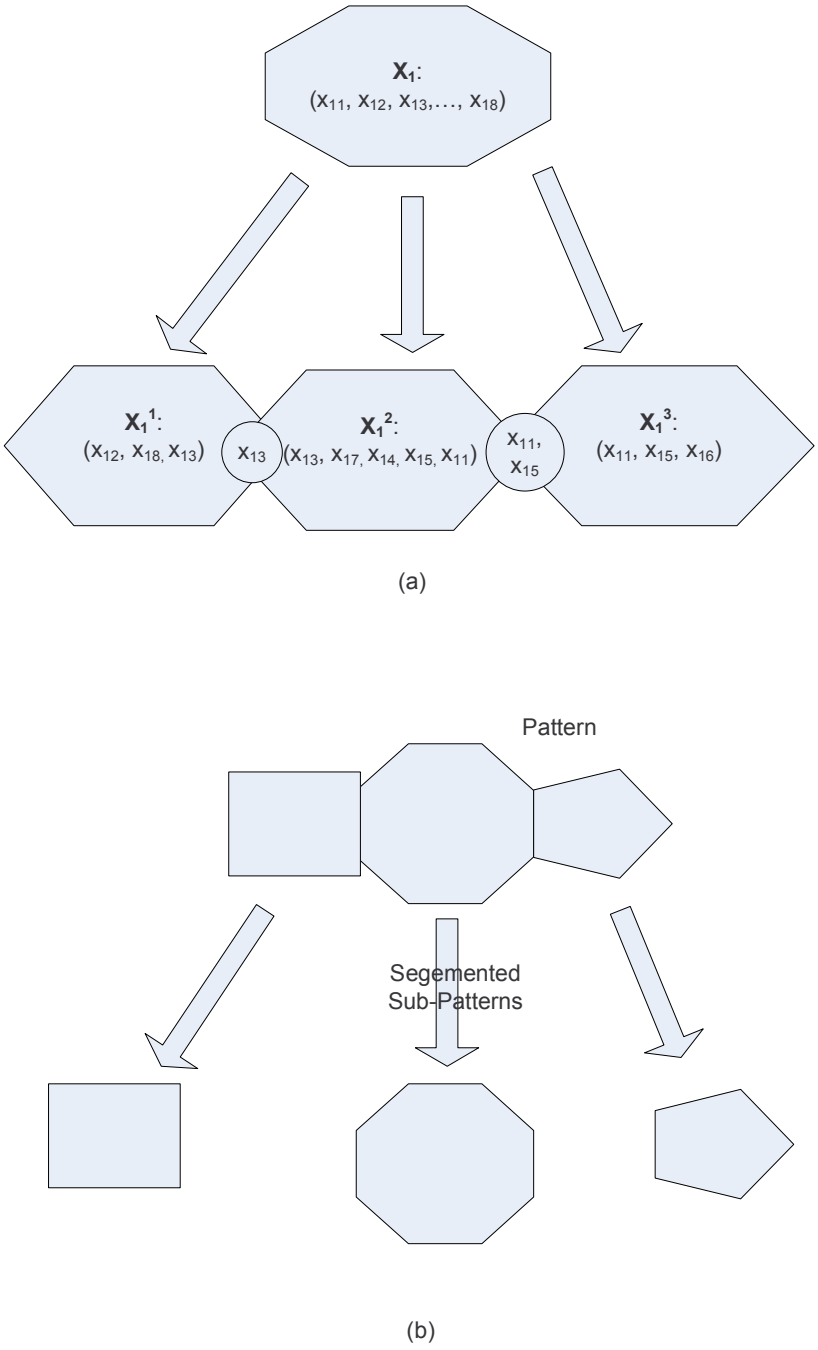


Figure 3.4: (a) *Partitioning with common (overlapping) features.* A pattern, $\mathbf{X}_1$ is partitioned into sub-patterns (blocks), $\mathbf{X}_1^1$, $\mathbf{X}_1^2$, $\mathbf{X}_1^3$. Overlapping features between sub-patterns are indicated within a circle. (b) *Partitioning by using a segmentation technique.* The sub-patterns (blocks) may have arbitrary shapes.

image into regions of no predefined shape using segmentation techniques and these regions are further down-sampled to a fixed block size (e.g. They used 5×5 block). Region based PCA [136] divides each image (target) into 5 regions of different shapes in their experiment. Clustered Block-wise PCA [113] gives some hints to choose block size: similar (high or low) frequency variances in images should be in the same block. Localized PCA [106] [107] divides each pattern into 6 parts of elliptical shape. No other method clearly specifies as to how to choose block size.

Sub-pattern (block) size is chosen to be less than the number of training patterns. This option is useful to reduce (control) Small Sample Size (SSS) problem. The sub-pattern size can also be used to control computational requirements.

3.3.3 Overlap between Sub-Patterns (Blocks)

This issue relates to whether the division of a pattern into blocks forms a proper partition (without overlap, no common features), e.g. SubPCA [21], Modular PCA [53], Eigen-regions method [45], Sub-Holistic PCA [80], Multi-Block PCA [128], Adaptively weighted SubPCA [157] methods form a partition of sub-patterns. The alternative is to divide the features with certain features common between blocks, e.g. Localized PCA [106] [107]. In this case, we need to decide how to create overlapping sub-patterns. An example of overlapping sub-patterns is shown in Fig. 3.4(a).

3.3.4 Grouping of Sub-Patterns (Blocks)

After patterns are divided into blocks, one has to decide how to group these blocks. Depending on how we group the blocks, we obtain different local features.

For example, (i) all the blocks are grouped into a single group, e.g. Modular PCA [53] and Eigen-regions method [45] (Fig. 3.5 and Defn. 2) and (ii) form k groups in such a way that j^{th} block moves into j^{th} group (Fig. 3.6 and Defn. 2). Most of the FP-PCA methods except Modular PCA [53] and Eigen-regions method [45] use the option (ii).

3.3.5 Local Feature Extraction Method

After blocks are grouped, another issue arises is, "which PCA method to use for feature extraction within these groups". Here we call features extracted 'local', because the scope of feature extraction is limited to each of the groups, not across groups of sub-patterns. Local feature extraction is proved to be effective when discriminative information is present in local variations limited to sub-pattern groups, rather than across all patterns. One may opt for (i) classical PCA method, e.g. most of the existing FP-PCA methods use this option, (ii) other variation of PCA such as 2DPCA, (iii) one may use Hebbian neural network based PCA (Section 2.4 of Chapter 2) and (iv) one can also use Kernel PCA or other Non linear PCA methods to capture non-linearity among patterns (Sections 2.5 & 2.4 of Chapter 2). No FP-PCA method is found to be using options (ii) to (iv).

3.3.6 Selection of Principal Components (PCs)

Since patterns are divided into blocks, there is an option to choose PCs locally from each block based on local criteria or to apply global selection criteria across all sub-pattens. For example, (i) we fix a local threshold (δ) on the eigenvalues of a sub-

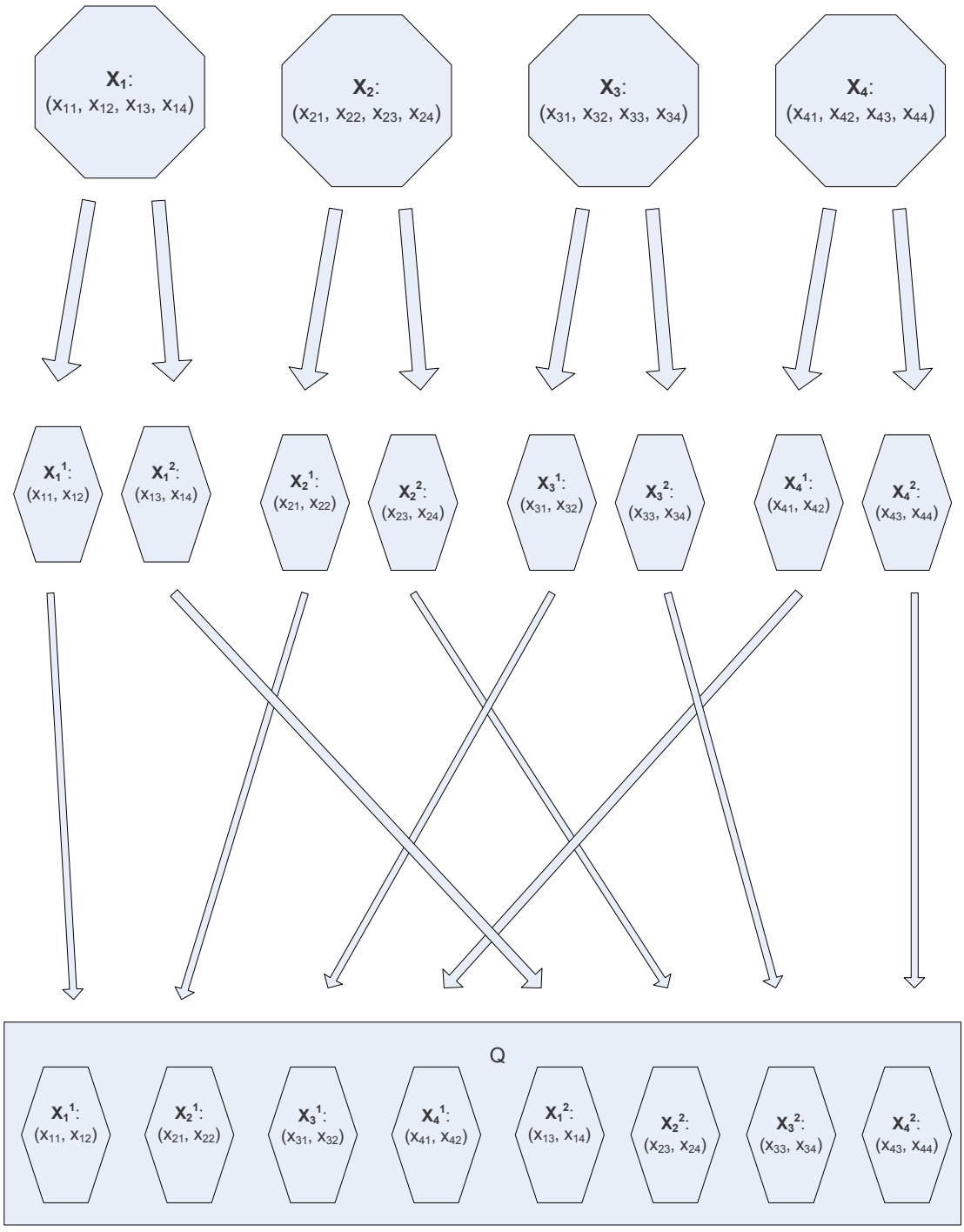


Figure 3.5: Grouping sub-patterns of patterns, $\mathbf{X}_1$, $\mathbf{X}_2$, $\mathbf{X}_3$, $\mathbf{X}_4$, into single sub-pattern set, $\mathbf{Q}$.

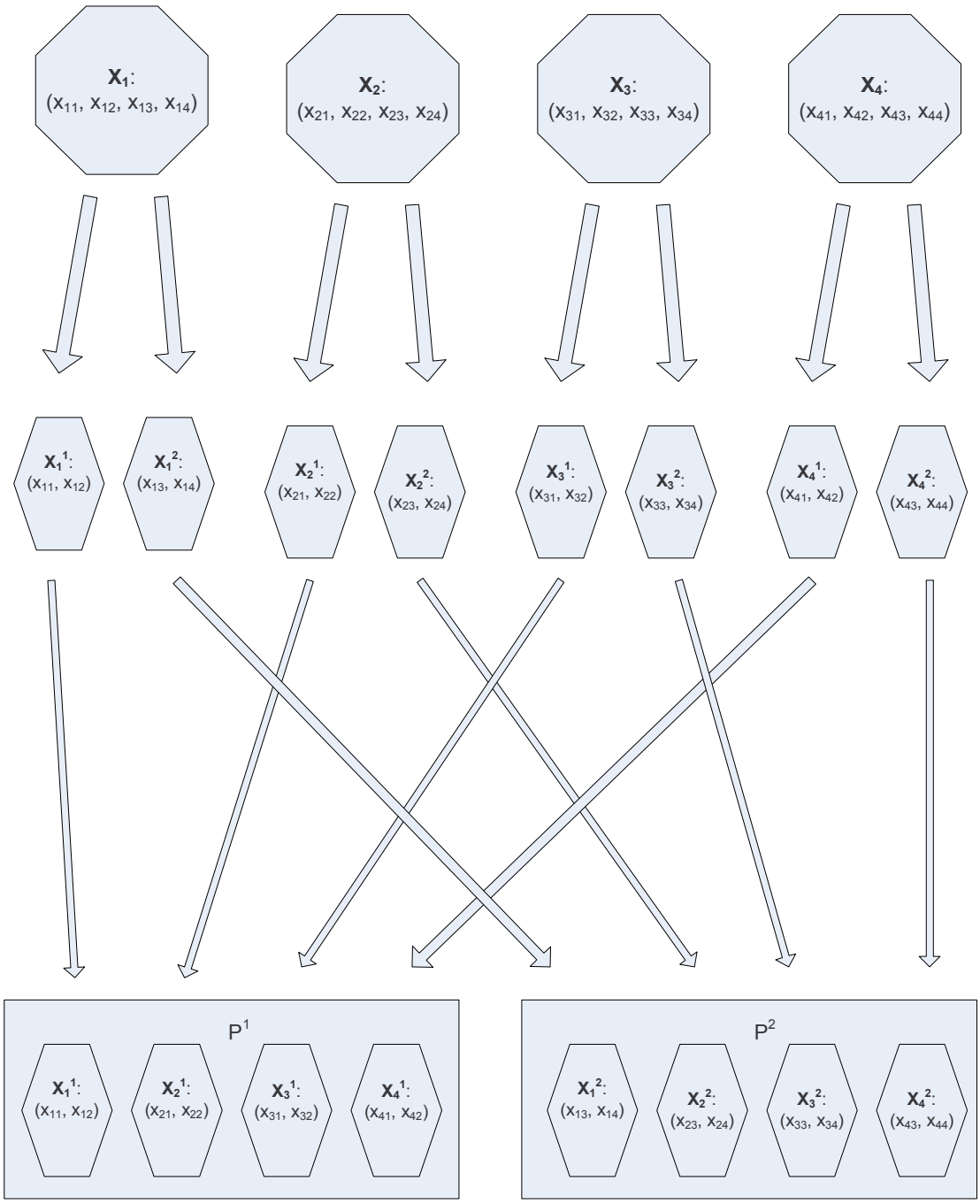


Figure 3.6: Grouping sub-patterns of patterns, $\mathbf{X}_1$, $\mathbf{X}_2$, $\mathbf{X}_3$, $\mathbf{X}_4$, into multiple sub-pattern sets. $\mathbf{P}^1$, $\mathbf{P}^2$.

pattern group or (ii) fix a global threshold on eigenvalues across sub-pattern groups and use it to select PCs. In other words, if $(\frac{\lambda_i}{\sum_i \lambda_i}) \geq \delta$ then choose eigenvector, $\mathbf{e}_i$, corresponding to eigenvalue λ_i or (iii) another approach would be to select a fixed number (r) of eigenvectors (thus r local features or PCs) from each of k blocks and select $k.r$ number of eigenvectors in total. Other methods used for general Principal Component (PC) selection may also be used [123][29] (Section 2.8 of Chapter 2). Region based PCA [136] uses 98% of local variance as a threshold to select principal components. SubPCA [21] use the option of choosing fixed number of eigenvectors from each block (sub-pattern). No other method seems to take up specifically the issue of PC selection.

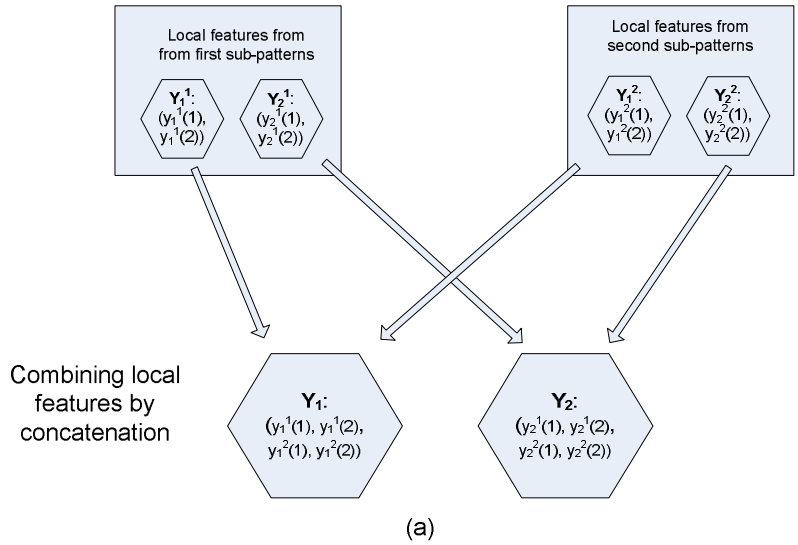
3.3.7 Combining Locally-Extracted Features

After extracting local features from sub-patterns, we need a method to combine them to form reduced patterns. Some methods include (i) simple concatenation of these features, e.g. SubPCA [21], Modular PCA [53] (Fig. 3.7(a)), (ii) exploiting redundancy across blocks (inter-block correlations, see subsection 3.3.8) (Fig. 3.7(b)), (iii) other methods include finding classification based contributions from each block, e.g. Adaptively weighted SubPCA [157] or using probabilistic mixture model, Localized PCA [106] [107] or use subspace distance to merge local subspaces, e.g. Clustered Block-wise PCA [113], (iv) another option is not to combine at all, in other words, local features of a block are used independently for subsequent tasks, e.g. Sub-Holistic PCA [80], Eigen-regions method [45], Region based PCA [136], (v) yet another way is to combine specific to an application, e.g. Multi-Block PCA [128] combines all first

PCs to form newly constructed image, next all second PCs and so on. This method of combination is useful to find changes in the images (patterns).

3.3.8 Loss of Inter-Sub-Pattern Correlations (Inter-Block Correlations or Dependencies)

Another interesting issue is how to exploit correlation or covariance structure or dependencies across sub-patterns (blocks), lost due to partitioning (Figs. 3.8 & 3.9). The features extracted from different blocks, $\mathbf{X}_i^j \; ; j = 1, 2, \ldots, k$, may be correlated. We call such correlations as 'inter-block correlations' or 'inter-sub-pattern correlations or dependencies'. A good PCA approach should not neglect such inter-block correlations, because the core idea of classical PCA is to utilize the entire covariance (correlation) structure to extract more informative and salient features. These correlations are crucial and help very much in dimensionality reduction and improving summarization of variance. It is clear that a greater number of blocks (sub-patterns) leads to more loss of such covariances (or correlations) between the features (Figs. 3.8 and 3.9). Thus the study of inter-block correlations is not a trivial step. A few inter-sub-pattern correlations are illustrated in figures 3.10 and 3.11, for the data sets of UCI repositories [165] (See Section 4.4 of Chapter 4 for details). First, let us study the inter-sub-pattern correlations in waveform data (Fig. 3.11). Here the first PC from each of 3 sub-pattern sets of waveform training data is extracted and are plotted as shown in Fig. 3.11(a,b,c). As shown in the Fig. 3.11(a) almost entire variance of 2 first PCs of sub-pattern sets is approximated by a single line, hence inter-sub-pattern correlation is obvious. Similar kind of inter-sub-pattern correlations



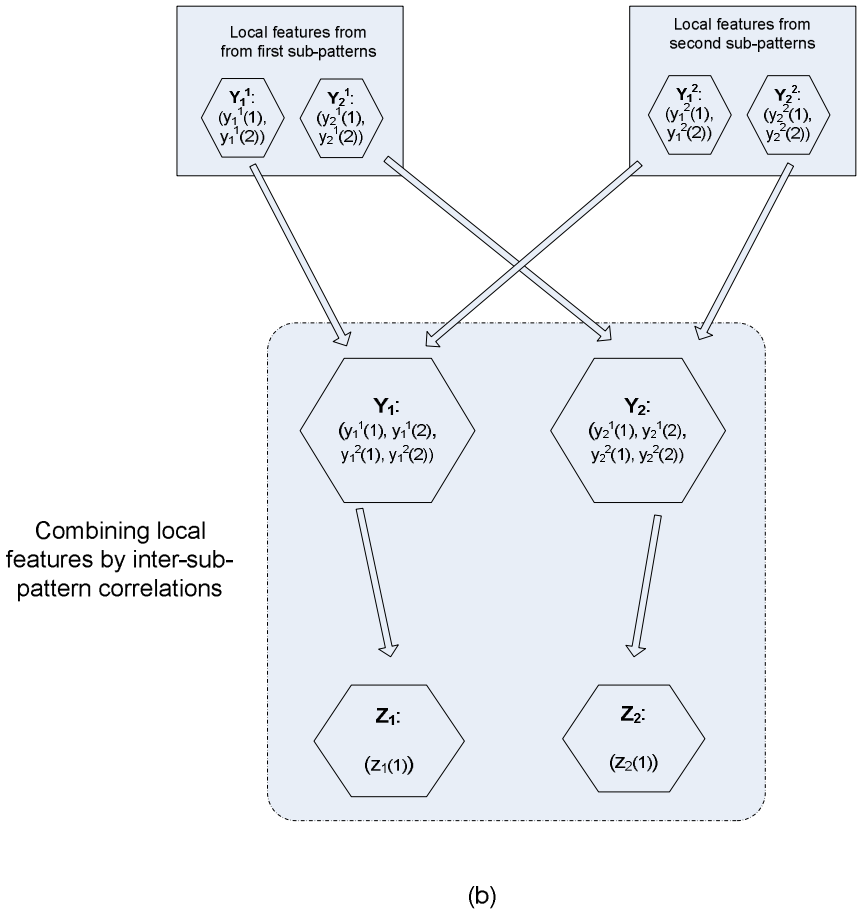


Figure 3.7: (a) Combining local features obtained from sub-patterns *by concatenation* which forms reduced patterns. (b) Combining local features obtained from sub-patterns *by exploiting inter-block correlations or dependencies* (Subsection 3.3.8) to form reduced patterns.

are observed between first PCs of second and third sub-pattern sets (Fig. 3.11(b)) and between first and third sub-pattern sets (Fig. 3.11(c)). Another instance of inter-sub-pattern correlations is seen in musk data (Fig. 3.10). As shown in Fig. 3.10 almost entire variance is approximated by a single line showing the presence of inter-sub-pattern correlations and thus a possibility of dimensionality reduction from 2 PCs (features) to 1 feature for this (musk) data set. Thus, we see that inter-sub-pattern correlations (dependencies) are quite common and can be exploited for better summarization of variance (which leads to better dimensionality reduction). The existing FP-PCA methods in the literature (Section 2.2 of Chapter 2) ignore such inter-block dependencies (correlations).

3.3.9 Feature Order Dependency

Consider the selection of $t\,(\geq 2)$ features contiguously, in their order of appearance to form sub-patterns, while dividing the given pattern into $k\,(\geq 2)$ blocks. That is, each pattern, $\mathbf{X}_i$ can be represented by $\mathbf{X}_i = [\{x_{i_1}, \ldots, x_{i_t}\}, \ldots, \{x_{i_{(k-1).t+1}}, \ldots, x_{i_{k.t}}\}]$, where $\{x_{i_1}, \ldots, x_{i_t}\}$ is the first sub-pattern, $\mathbf{X}_i^1$ and $\{x_{i_{(k-1).t+1}}, \ldots, x_{i_{k.t}}\}$ is the k^{th} sub-pattern, $\mathbf{X}_i^k$. For any two feature orders (arrangements), $\mathbf{F}_1$ and $\mathbf{F}_2$, we get two different partitions (That is different set of sub-patterns for each feature order), which may lead to different covariance (correlation) structures. Therefore, for any two feature orders, $\mathbf{F}_i$ and $\mathbf{F}_j$; $i \neq j$, there is possibility of having two different summarizations of variance (also different local features). This makes evident that different feature orders of the same patterns affect dimensionality reduction and subsequent classification as well. Feature order is a permutation of feature values of a pattern

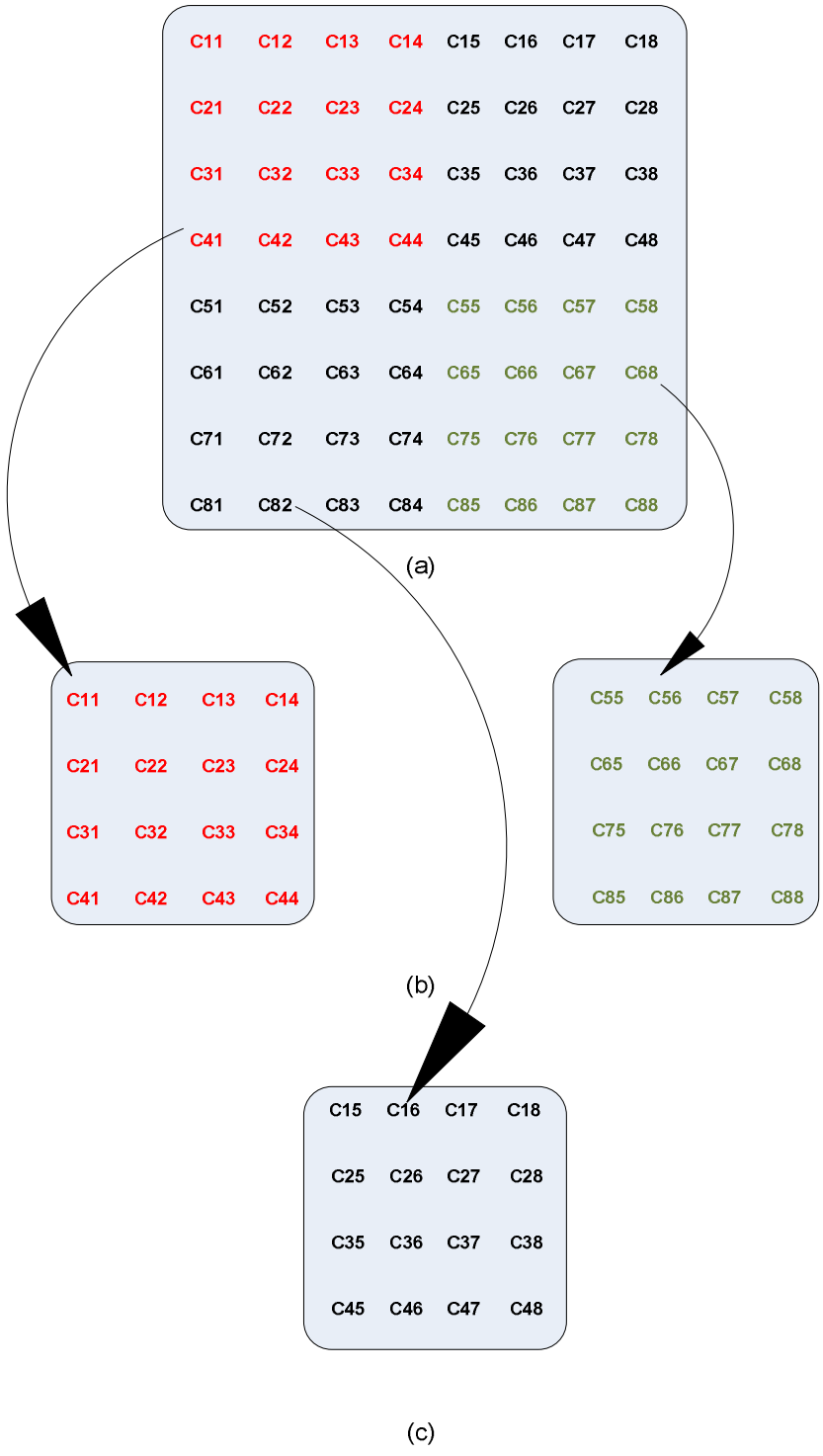


Figure 3.8: *Loss of covariance structure with 2 sub-patterns (blocks).* (a) Covariance structure with 8 features before partitioning. Please note that $c_{ij} = c_{ji}$, (b) Covariance structure after partitioning into 2 equally-sized blocks, (c) The covariances lost due to partitioning, which indicates missing of some dependency information.

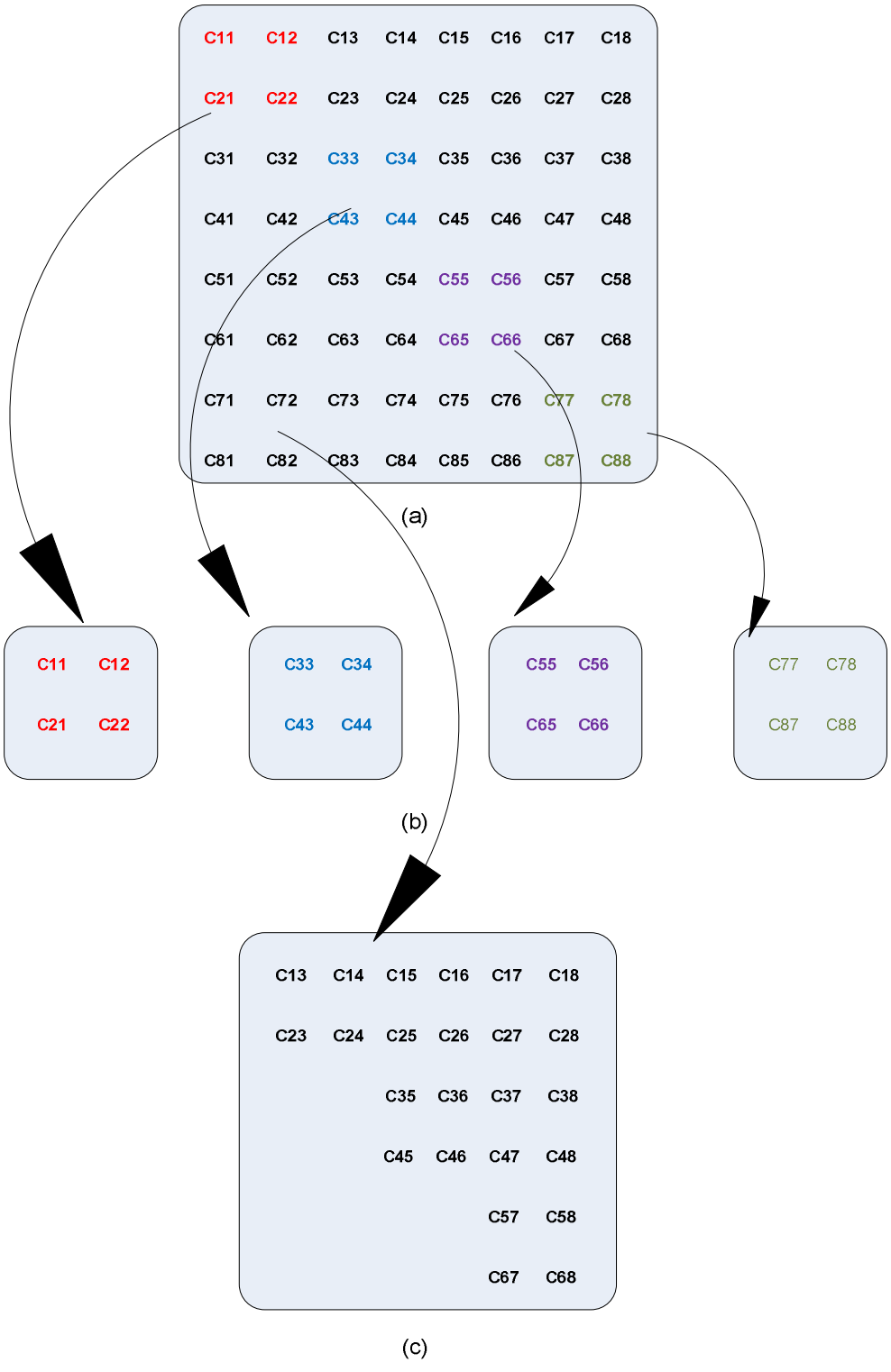


Figure 3.9: *Loss of covariance structure with 4 sub-patterns (blocks).* (a) Covariance structure with 8 features before partitioning. Please note that $c_{ij} = c_{ji}$, (b) Covariance structure after partitioning into 4 equally-sized blocks, (c) The covariances lost due to the partitioning, which indicates missing of some dependency information. Comparing with Fig. 3.8, it is clear that more covariances are lost with more number of blocks.

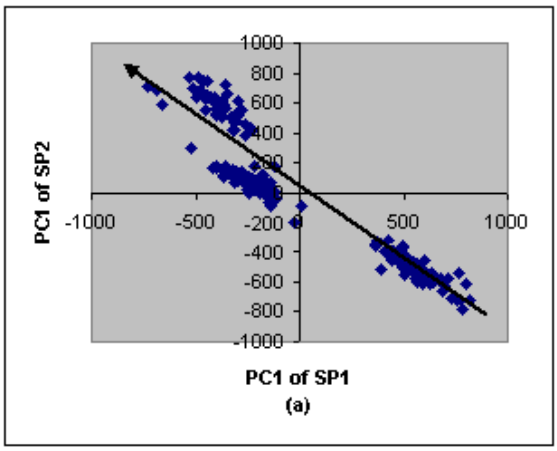


Figure 3.10: Inter-sub-pattern correlations in *Musk Data* [165].

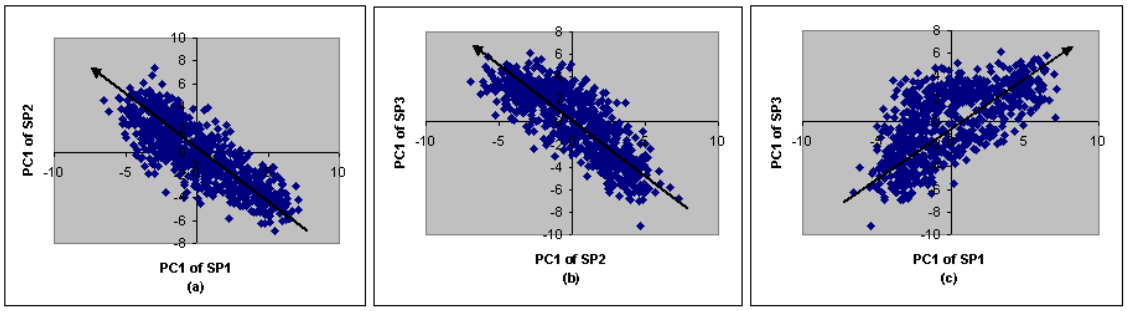


Figure 3.11: Inter-sub-pattern correlations in *Waveform Data* [165]..

(Fig. 3.12). Optimal performance based upon feature ordering in sub-patterns is an open issue. We can reduce the feature order dependency by using a systematic way to combine locally-extracted features, by exploiting inter-block correlations (Fig. 3.7(b)). No FP-PCA method in the literature discussed about this issue.

3.3.10 Truncation of Features

If we consider equally-sized blocks and if the pattern size d is not a multiple of block size (u), then the size of the last sub-pattern (say, d_t) is different from the size of all other blocks. While selecting sub-pattern size, one has to decide, whether d_t features in the last block are to be truncated. The truncation option may cause loss of information, if the truncated features have vital discriminative information. One way to solve this issue is to process the last block separately with a different block size, d_t. This issue does not arise for the case of unequal sub-pattern sizes. The same issue extended to image pattern also. Some methods which use truncation option include SubPCA [21], however they may be easily modified to avoid truncation of last features.

3.4 Summary

We have presented here a general framework for FP-PCA methods and brought out fundamental issues (such as partitioning procedure of given patterns, loss of covariance information, sub-pattern size, feature order dependency, etc) to be addressed in the context of partitioning of patterns. Using the framework, we comprehend FP-PCA methods with ease and have analyzed and framed the issues in a more formal manner.

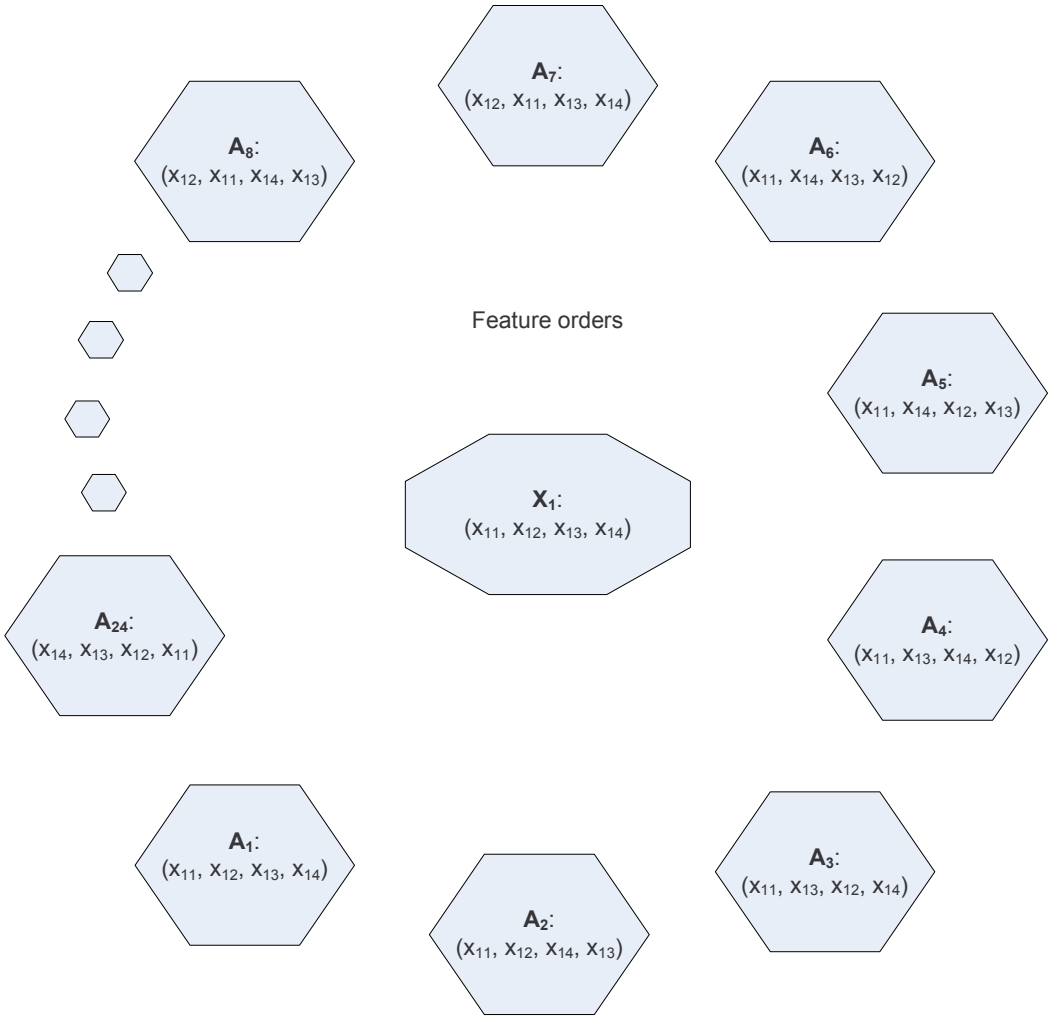


Figure 3.12: 24 Feature arrangements (orders) for a pattern of 4 features.

Next, we take a vital issue 'loss of inter-block correlations', which is crucial for dimensionality reduction and classification. This issue is addressed in the next Chapter, by proposing a novel feature partitioning algorithm which takes up a specific instance from the general concepts proposed here. We also discuss addressing of other issues 'Feature order dependency' and 'overlapping sub-patterns' in the next Chapter.

Chapter 4

SubXPCA: A Feature Partitioning Approach to Principal Component Analysis

4.1 Introduction

Note: An initial version of the work in this chapter has been published in *Pattern Recognition Journal (Elsevier Science)*[1] and in *proceedings of IVCNZ 2005 Conference*[2].

Principal Component Analysis (PCA) is concerned with summarizing the *variance-*

[1]Kadappagari Vijaya Kumar and Atul Negi, "SubXPCA and a generalized feature partitioning approach to principal component analysis", *Pattern Recognition*, Vol. 41, No. 4, Apr. 2008, pp. 1398-1409.

[2]Atul Negi and Kadappagari Vijaya Kumar, "An experimental study of sub-pattern based principal component analysis and cross-subpattern-correlation based principal component analysis (SubXPCA)", *In Proceedings of Image and Vision Computing Conference (IVCNZ-2005)*, New Zealand, pp. 20-25, Nov. 28^{th}-29^{th} 2005.

covariance structure using a few linear combinations of the original set of d variables (features). The usefulness and hence popularity of PCA comes from its properties – it is an optimal linear scheme, in terms of mean squared error for reducing data to a lower dimensionality and uses only matrix multiplication operations for reduction and reconstruction. However, classical PCA suffers from large time complexity ($O(N.d^2)$, N is number of training patterns) just to calculate the covariance matrix for high dimensional data. Reduction of time complexity is essential especially for the algorithms, where PCA is used fundamentally and is computed several times, for example, clusters of correlation connected objects [12]. More discussion on PCA may be found in section 1.3 of Chapter 1. As discussed in Chapter 2, many approaches such as Neural Network based PCA methods, other incremental methods, 2DPCA based methods, etc, have reduced computational complexity as compared to classical PCA methods. However, these methods are based on whole-patterns, which are suitable for global feature extraction like classical PCA, may not perform well if local variations are prominent.

Speeding-up of computation of principal components (PCs) is not only the concern when studying advanced PCA computation methods. Additionally we need to see how to balance local and global feature properties while computing PCs, to improve classification in different scenarios. It is well known that classical PCA performs global feature extraction which yields global features. Here 'global features' are those extracted by considering all covariances (correlations) possible between every pair of original features. Classical PCA may perform well when global variations are more predominant. However, global features often may not work well when varia-

tions among patterns are prominently visible within a part of patterns (i.e. local variations), rather than across whole patterns. In this context, feature partitioning based PCA (FP-PCA) methods such as SubPCA, Region-based PCA, Modular PCA, EigenRegions, etc (Section 2.2 of Chapter 2) are proposed in the literature and are proved to be efficient. FP-PCA methods have advantages over classical PCA: (i) These methods have reduced time complexity, (ii) perform well when local variations are dominant and (iii) reduce small sample size problem. However, the existing FP-PCA methods may not perform well when global variations of patterns are dominant. In addition, these FP-PCA methods need more locally-extracted features (i.e. more local PCs) to retain most of the variance, because these methods do not make use of complete covariance information of the patterns (Section 3.3.8 of Chapter 3).

In this chapter, we propose a novel FP-PCA approach, called as SubXPCA. SubXPCA overcomes the problems faced by global and local feature extraction methods to PCA, keeping the merits of both intact. SubXPCA does so, (i) by extracting local features from sub-patterns (blocks) (local feature extraction) and (ii) by exploiting inter-sub-pattern correlations (cross-sub-pattern correlations) among those locally-extracted features (global feature extraction). We further show that SubXPCA is a generalization of PCA and can be derived as a special case of SubXPCA. SubXPCA balances the global PC computation of classical PCA against the local viewpoint of SubPCA and similar methods. We also prove the computational superiority of SubXPCA over PCA. Comprehensive experimentation shows the superiority of SubXPCA on UCI [165] repository data, the well known Yale [184], CMU [25] and ORL [119] face data sets.

Please note that SubXPCA is an instance of 'Generalized Feature Partitioning Framework to PCA' (Chapter 3), which addresses vital issues (i) Loss of inter-sub-pattern correlations (Section 3.3.8 of Chapter 3) and (ii) feature order dependency (Section 3.3.9 of Chapter 3).

The rest of the chapter is organized as follows. In section 4.2, we give a formal presentation of the proposed SubXPCA method. Time complexity analysis is performed in detail in section 4.3. We show experimental results in section 4.4 and we present a discussion in section 4.5.

4.2 Cross-Sub-Pattern Correlation based PCA (SubXPCA)

In this section, we formally present our technique, SubXPCA, which is based upon the feature partitioning framework (Chapter 3) and specifically deals with the issues 'Loss of cross(inter)-sub-pattern correlations' and 'Feature order dependency' (Section 3.3 of Chapter 3).

4.2.1 SubXPCA Algorithm

The algorithm is given below. For easier understanding please see Fig. 4.1 and Fig. 4.2. Here we assume that all the patterns are mean-subtracted.

1. *Partitioning step (Step-1 in Fig. 4.1):*

Divide every d-dimensional pattern, $\mathbf{X}_i$; $i = 1, 2, \ldots, N$ into k (≥ 2) equally-sized sub-patterns, $\{\mathbf{X}_i^1, \mathbf{X}_i^2, \ldots, \mathbf{X}_i^k\}$. Each sub-pattern is of size u, where $u = \lfloor \frac{d}{k} \rfloor$. We consider

equally-sized sub-pattern option for simplicity. Sub-pattern size is chosen to minimize the loss of last features. However, it is not mandatory to consider equally-sized sub-patterns (Fig. 3.3 of Chapter 3). Sub-patterns are formed by choosing features contiguously as they appear in the pattern. In other words, for a given pattern, $\mathbf{X}_i$=$(x_{i_1}, x_{i_2}, \ldots, x_{i_d})^T$, first sub-pattern, $\mathbf{X}_i^1$ contains features, $x_{i_1}, x_{i_2}, \ldots, x_{i_u}$ and j^{th} sub-pattern, $\mathbf{X}_i^j$ contains features given by

$$(\mathbf{X}_i^j)_{u\times 1} = (x_{i_l}, x_{i_{(l+1)}}, \ldots, x_{i_{(l+u-1)}})^T \tag{4.1}$$

where $l = (j-1).u + 1$, $1 \leq i \leq N$, $1 \leq j \leq k$.

2. *Grouping step (Step-2 in Fig. 4.1):*

We pick-up j^{th} sub-pattern, corresponding to every pattern, $\mathbf{X}_i$; $i = 1, 2, \ldots, N$, and form j^{th} sub-pattern set (or j^{th} sub-pattern group), $\mathbf{P}^j$, (Fig. 3.6 and Defn. 2 of Chapter 3), which is given by

$$(\mathbf{P}^j)_{N\times u} = [\mathbf{X}_1^j\, \mathbf{X}_2^j\, \ldots\, \mathbf{X}_N^j]^T \tag{4.2}$$

Here we use the option of grouping of homogeneous sub-patterns (Defn. 2 of Chapter 3).

3. *Local feature extraction step (Step-3 in Fig. 4.1):*

For every sub-pattern set, $\mathbf{P}^j$, where $j = 1, 2, \ldots, k$, repeat the following steps (a)-(d).

(a) Compute local covariance matrix, $(\mathbf{C}^j)_{u\times u}$ as given by

$$(\mathbf{C}^j)_{u\times u} = \frac{1}{N}.\sum_{i=1}^{N}[\mathbf{X}_i^j]_{u\times 1}.[\mathbf{X}_i^j]_{1\times u}^T \tag{4.3}$$

(b) Compute eigenvalues (λ_p^j) and corresponding eigenvectors $(\mathbf{e}_p^j)$, where $p = 1, 2, \ldots, u$, using eigenvalue decomposition (EVD) of $\mathbf{C}^j$ given by

$$\mathbf{C}^j.\mathbf{e}_p^j = \mathbf{e}_p^j.\lambda_p^j \tag{4.4}$$

(c) Select r ($\leq u$) eigenvectors corresponding to the first r largest eigenvalues obtained in the preceding step. Let $\mathbf{E}^j$ be the set of r eigenvectors (column vectors) selected in this step and is given as follows.

$$(\mathbf{E}^j)_{u\times r} = [\mathbf{e}_1^j\, \mathbf{e}_2^j\, \ldots\, \mathbf{e}_r^j]_{u\times r} \tag{4.5}$$

(d) Extract r local features (local PCs) by projecting $\mathbf{P}^j$ onto $\mathbf{E}^j$ as follows. Let $\mathbf{R}^j$ be the reduced data in this step and is given as follows.

$$(\mathbf{R}^j)_{N\times r} = (\mathbf{P}^j)_{N\times u}.(\mathbf{E}^j)_{u\times r} \tag{4.6}$$

$$(\mathbf{R}^j)_{N\times r} = \begin{bmatrix} (\mathbf{Y}_1^j)^T \\ (\mathbf{Y}_2^j)^T \\ \vdots \\ (\mathbf{Y}_N^j)^T \end{bmatrix} = \begin{bmatrix} y_1^j(1) & y_1^j(2) & \ldots & y_1^j(r) \\ y_2^j(1) & y_2^j(2) & \ldots & y_2^j(r) \\ \vdots & \vdots & \vdots & \vdots \\ y_N^j(1) & y_N^j(2) & \ldots & y_N^j(r) \end{bmatrix} \tag{4.7}$$

where $\mathbf{Y}_i^j$ is the locally-reduced version of $\mathbf{X}_i^j$, the j^{th} sub-pattern of $\mathbf{X}_i$.

4. *Combining locally-extracted features step (Step-4 in Fig. 4.2):*

(a) Form locally-reduced pattern, $\mathbf{Y}_i$ by concatenating locally-reduced sub-patterns (local features), $(\mathbf{Y}_i^j)_{r\times 1}$, $\forall j = 1, 2, \ldots, k$, as shown below.

$$(\mathbf{Y}_i)_{k.r\times 1} = [(\mathbf{Y}_i^1)^T, (\mathbf{Y}_i^2)^T, \ldots\ldots, (\mathbf{Y}_i^k)^T]_{k.r\times 1}^T \tag{4.8}$$

(b) Perform global feature extraction using cross(inter)-sub-pattern correlations (co-variances) of $\mathbf{Y} = \{\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_N\}$, obtained in the preceding step as given in (i)-(iv).

(i) Compute global covariance matrix, $(\mathbf{C}^g)_{(k.r)\times(k.r)}$ for the data $\mathbf{Y}$ as follows.

$$(\mathbf{C}^g)_{(k.r)\times(k.r)} = \frac{1}{N}.\sum_{i=1}^{N}[\mathbf{Y}_i].[\mathbf{Y}_i]^T \tag{4.9}$$

(ii) Compute eigenvalues (λ_s) and corresponding eigenvectors ($\mathbf{e}_s$), where $s = 1, 2, \ldots, (k.r)$ using eigenvalue decomposition given by

$$\mathbf{C}^g.\mathbf{e}_s = \mathbf{e}_s.\lambda_s \tag{4.10}$$

(iii) Select w ($\leq k.r$) eigenvectors corresponding to first w largest eigenvalues obtained in the preceding step. Let $\mathbf{E}^g$ be the set of w eigenvectors selected in this step and is given by

$$(\mathbf{E}^g)_{k.r \times w} = [\mathbf{e}_1\, \mathbf{e}_2\, \ldots\, \mathbf{e}_w]_{k.r \times w} \tag{4.11}$$

(iv) Extract w global features (global PCs) by projecting $\mathbf{Y}$ (obtained in Step-4(a)) onto $\mathbf{E}^g$. Let $\mathbf{Z}$ be the data obtained after projection in this step and is given as

$$(\mathbf{Z})_{N \times w} = (\mathbf{Y})_{N \times k.r}.(\mathbf{E}^g)_{k.r \times w} \tag{4.12}$$

We finally obtained $(\mathbf{Z})_{N \times w}$ which is the reduced form of $(\mathbf{X})_{N \times d}$ and $\mathbf{Z}$ is further used for subsequent tasks such as classification, recognition, clustering, etc.

Theorem 1 *PCA is a special case of SubXPCA.*

Proof 1 *PCA finds covariances (correlations) between every pair of d features. Here in steps 3(c)-3(d) of SubXPCA we need to set $r = u$ (i.e. all features of every sub-pattern are chosen and $u = \frac{d}{k}$). Therefore in Step-4, SubXPCA finds correlations between $k.r = k.u = k.(\frac{d}{k}) = d$ features. Hence the theorem is proved.*

4.3 Time Complexity Analysis

Consider $\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\}$, the set of N patterns of dimensionality d. Time complexities to (i) compute covariance matrix of size $d \times d$, T_{cov}, (ii) compute eigen-

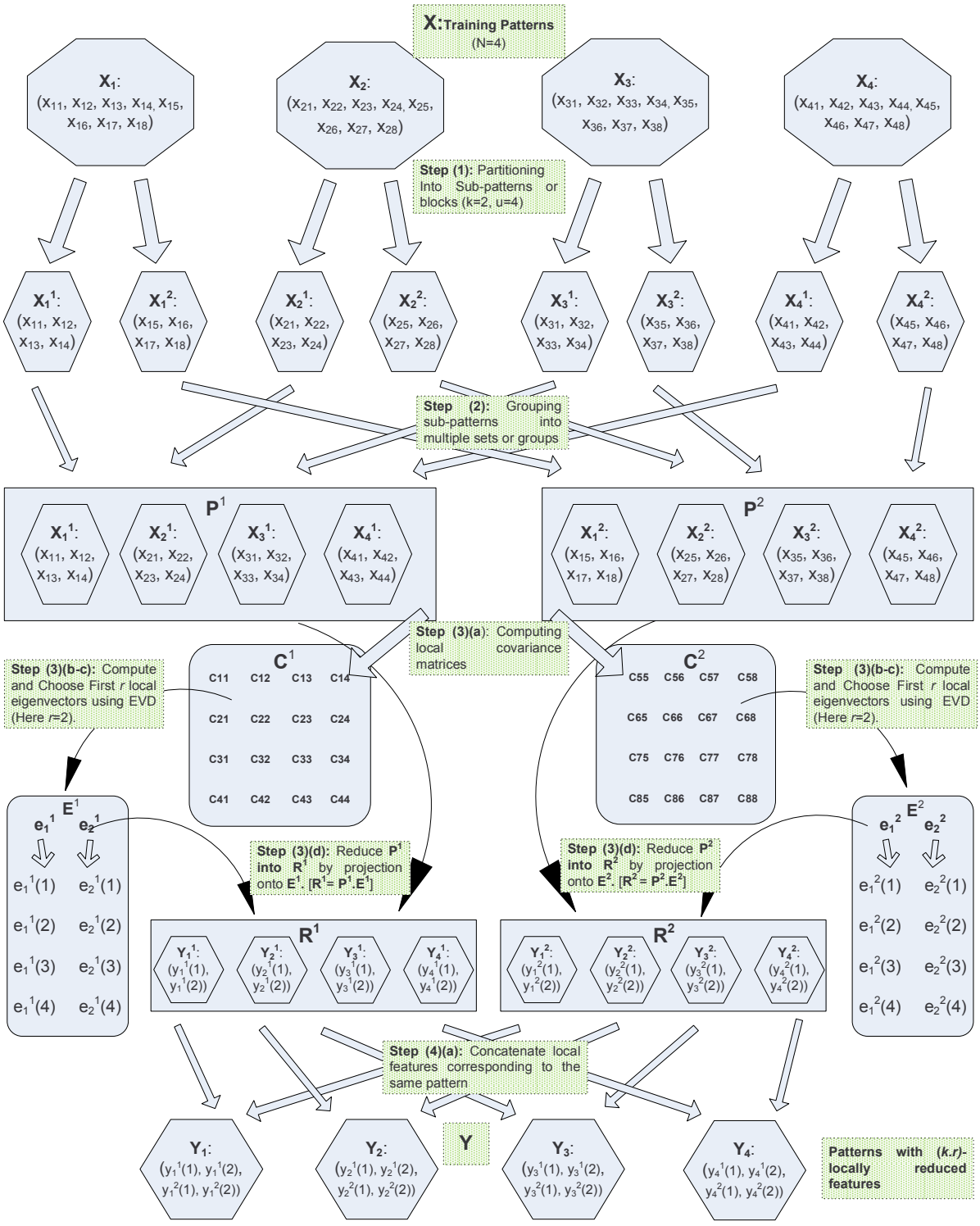


Figure 4.1: Visualizing SubXPCA method Part-I

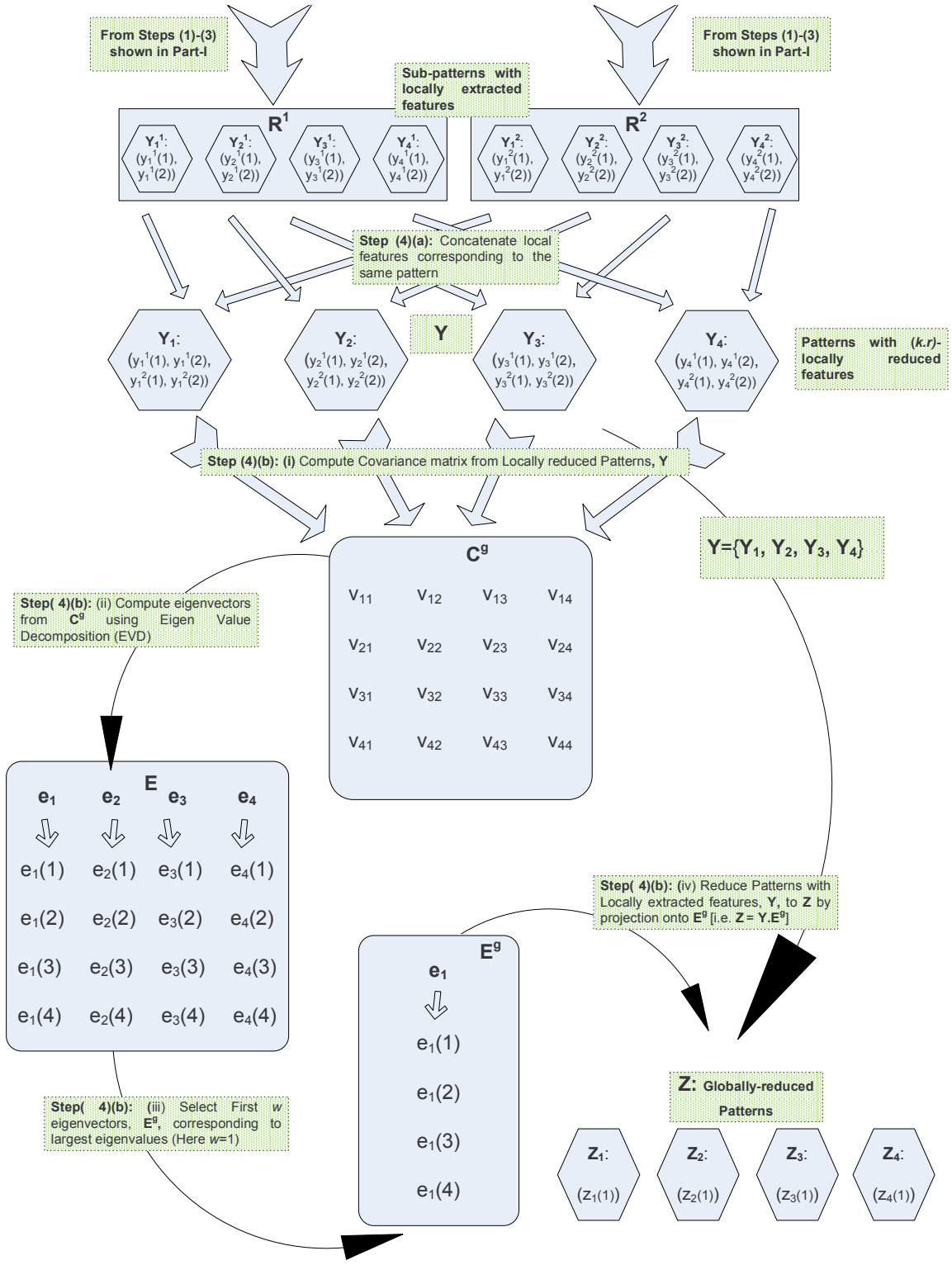


Figure 4.2: Visualizing SubXPCA method Part-II

values and eigenvectors using eigenvalue decomposition[3], T_{evd} are given as follows.

$$T_{cov} = O(N.d^2) \tag{4.13}$$

$$T_{evd} = O(d^3) \tag{4.14}$$

Thus, the total time complexity of classical PCA method is given by

$$T_C = O(N.d^2 + d^3) \tag{4.15}$$

On the similar lines, the time complexity of SubPCA method for one sub-pattern (block) is given by

$$T_b = O(N.u^2 + u^3) \tag{4.16}$$

Therefore, the total time complexity of SubPCA method for all k sub-patterns (blocks) is given by

$$T_S = k.T_b = O[k.(N.u^2 + u^3)] \tag{4.17}$$

Now, we focus on computing the time complexity of SubXPCA, T_F, which is a sum of (i) time complexity of processing all k sub-patterns (same as SubPCA) (Steps 1-3 of section 4.2) and (ii) time complexity of extracting features using inter-sub-pattern correlations (Step-4 of section 4.2).

The total time complexity of SubXPCA, T_F, is given as

$$T_F = T_S + O[N.(k.r)^2 + (k.r)^3] \tag{4.18}$$

Theorem 2 *$T_S < \frac{1}{k}.T_C$. In other words, The time complexity of SubPCA, T_S is less than $\frac{1}{k}$ times that of the time complexity of classical PCA, T_C. Here k is the number of sub-patterns (blocks) (See section 4.2 for terminology).*

[3]We use Householder Transform to find trigonal form and then QL algorithm with implicit shifts [127]

Proof 2 *From eq. (4.17),* $T_S = O[k.(N.u^2 + u^3)]$

$\Rightarrow T_S = O[k.u^2.(N + u)]$

$\Rightarrow T_S = O[k.\frac{d^2}{k^2}.(N + \frac{d}{k})]$ *(because* $u = \frac{d}{k}$*)*

$\Rightarrow T_S = O[\frac{d^2}{k}.(N + \frac{d}{k})]$

$\Rightarrow T_S = O[\frac{1}{k}.(N.d^2 + \frac{d^3}{k})]$

$\Rightarrow T_S < O[\frac{1}{k}.(N.d^2 + d^3)]$ *(because* $\frac{d^3}{k} < d^3$*)*

$\Rightarrow T_S < \frac{1}{k}.T_C$ *(from eq. (4.15))*

Hence the theorem follows.

Theorem 3 $T_F < T_C, \forall r < u.\sqrt{\frac{k-1}{k}}$*, where* $2 \leq k \leq \frac{d}{2}$*, is the number of sub-patterns (blocks) per pattern,* d *is the pattern size,* r *is the number of chosen eigenvectors per sub-pattern set,* $\mathbf{P}^j$ *and* u *is the sub-pattern size (See section 4.2 for terminology).*

Proof 3 *From eq. (4.18),* $T_F = T_S + O[N.(k.r)^2 + (k.r)^3]$

$\Rightarrow T_F = O[k.N.u^2 + k.u^3] + O[N.(k.r)^2 + (k.r)^3]$ *(from eq.* (4.17)*)*

$\Rightarrow T_F = O[k.N.\frac{d^2}{k^2} + k.\frac{d^3}{k^3}] + O[N.\frac{d^2}{u^2}.r^2 + \frac{d^3}{u^3}.r^3]$ *(because* $u = \frac{d}{k}$*;* $k = \frac{d}{u}$*)*

$\Rightarrow T_F = O[N.d^2.(\frac{1}{k} + \frac{r^2}{u^2}) + d^3.(\frac{1}{k^2} + \frac{r^3}{u^3})]$

$\Rightarrow T_F < O[N.d^2 + d^3]$ *provided if (i)* $(\frac{1}{k} + \frac{r^2}{u^2}) < 1$ *and (ii)* $(\frac{1}{k^2} + \frac{r^3}{u^3}) < 1$

$\Rightarrow T_F < O[N.d^2 + d^3]$ *provided if (i)* $\frac{r^2}{u^2} < (1 - \frac{1}{k})$ *and (ii)* $\frac{r^3}{u^3} < (1 - \frac{1}{k^2})$

$\Rightarrow T_F < O[N.d^2 + d^3]$ *provided if (i)* $r^2 < u^2.(1 - \frac{1}{k})$ *and (ii)* $r^3 < u^3.(1 - \frac{1}{k^2})$

$\Rightarrow T_F < O[N.d^2 + d^3]$ *provided if (i)* $r < u.(1 - \frac{1}{k})^{\frac{1}{2}}$ *and (ii)* $r < u.(1 - \frac{1}{k^2})^{\frac{1}{3}}$

$\Rightarrow T_F < O[N.d^2 + d^3]$ *provided if* $r < MIN[(u.(1 - \frac{1}{k})^{\frac{1}{2}}), (u.(1 - \frac{1}{k^2})^{\frac{1}{3}})]$ *(*$MIN(a, b)$ *indicates minimum of* a *and* b*)*

$\Rightarrow T_F < O[N.d^2 + d^3]$ *provided if* $r < u.(1 - \frac{1}{k})^{\frac{1}{2}}$ *(because* $(1 - \frac{1}{k})^{\frac{1}{2}} < (1 - \frac{1}{k^2})^{\frac{1}{3}}$*)*

$\Rightarrow T_F < T_C$ *provided if* $r < u.\sqrt{\frac{k-1}{k}}$ *(from eq. (4.15))*

Hence the theorem follows.

Corollary 1 *(i)* $T_F < T_C$, $\forall r < \frac{35.35}{100}.d$ *when the number of sub-patterns is minimum i.e.* $k = 2$ *(ii)* $T_F < T_C$, $\forall r = 1$ *when the number of sub-patterns is maximum i.e.* $k = \frac{d}{2}$*; where* d *is the size of original pattern and* r *is the number of chosen eigenvectors per sub-pattern set,* $\mathbf{P}^j$*.*

Proof 1 *(i) Consider* $T_F < T_C$, $\forall r < u.\sqrt{\frac{k-1}{k}})$ *from Theorem 3 and substitute* $k = 2$*.*

$\Rightarrow T_F < T_C$, $\forall r < u.\sqrt{\frac{2-1}{2}})$

$\Rightarrow T_F < T_C$, $\forall r < u.(0.707)$

$\Rightarrow T_F < T_C$, $\forall r < \frac{70.7}{100}.(\frac{d}{2})$ *(because* $u = \frac{d}{k}$*)*

$\Rightarrow T_F < T_C$, $\forall r < \frac{35.35}{100}.d$

Therefore, one can choose local eigenvectors upto 35% *of original features in this case.*

(ii) Consider $T_F < T_C$, $\forall r < u.\sqrt{\frac{k-1}{k}})$ *from Theorem 3 and substitute* $k = \frac{d}{2}$*.*

$\Rightarrow T_F < T_C$, $\forall r < u.\sqrt{1 - \frac{1}{d/2}})$

$\Rightarrow T_F < T_C$, $\forall r < 2.\sqrt{1 - \frac{2}{d}})$ *(because* $u = \frac{d}{k} = \frac{d}{d/2} = 2$*)*

$\Rightarrow T_F < T_C$, $\forall r < 2$ *(because* $\sqrt{1 - \frac{2}{d}} < 1$, $\forall d \geq 3$*)*

$\Rightarrow T_F < T_C$, $\forall r = 1$.

Hence the corollary follows.

Theorem 4 $\lim_{r \to 1, k \to 2}[T_F \approx (\frac{1}{k}).T_C]$*; where* $2 \leq k \leq \frac{d}{2}$ *is the number of sub-patterns (blocks) per pattern;* d *is the pattern size;* r *is the number of chosen eigenvectors per sub-pattern set,* $\mathbf{P}^j$ *(See section 4.2 for terminology).*

Proof 4 *From eq. (4.18),* $T_F = T_S + O[N.(k.r)^2 + (k.r)^3]$

From Theorem 2, first part of preceding equation, that is T_S *is less than* $(\frac{1}{k}).T_C$

Consider second part of preceding eq. T_F*, i.e.* $O[N.(k.r)^2 + (k.r)^3]$ *and prove that it is minimum for* $k = 2$ *and* $r = 1$*.*

$O[N.(k.r)^2 + (k.r)^3]$ *is minimum provided if (i)* $(k^2.r^2)$ *and (ii)* $(k^3.r^3)$ *are minimum.*

$\Rightarrow O[N.(k.r)^2 + (k.r)^3]$ *is minimum provided if* k *and* r *are minimum.*

$\Rightarrow O[N.(k.r)^2 + (k.r)^3]$ *is minimum provided if* $k = 2$ *and* $r = 1$ *(because* $2 \leq k \leq \frac{d}{2}$ *and* $1 \leq r \leq u$*).*

It is clear that $O[N.(k.r)^2 + (k.r)^3]$ *becomes insignificant for* $k = 2$ *and* $r = 1$*.*

Therefore, T_F *is approximately nearer to* $(\frac{1}{k}).T_C$ *as* k *and* r *approaches their minimum values.*

By Theorem 4, $T_F \approx (\frac{1}{k})T_C$ is true for smaller values of k and r. However, in practice, r may not be chosen as 1 (i.e. smallest possible value), especially when k is small, since the recognition rate may get reduced due to less number of eigenvectors (r). If k is too small, the computational complexity of SubXPCA and SubPCA may not be significantly reduced as compared to PCA. Hence r and k are required to be chosen carefully to achieve good recognition rate and time efficiency.

4.4 Experimental Results and Analysis

In this section, we present the results of our SubXPCA implementation on certain standard data sets and compare the results with SubPCA [21] (an existing FP-PCA method) and classical PCA (global PCA) methods.

4.4.1 UCI Data Sets

We considered 5 publicly available databases from UCI repository of Machine Learning [165] for our experiments. (1) Waveform data (21 features, 3 classes with labels (0,1,2), 5000 Patterns, 250 patterns each class, for training, rest of them for testing). (2) Musk data (166 features, 2 classes with labels (0,1), 6598 patterns, 100 patterns each class, for training, rest of them for testing). (3) Wine data (13 features, 3 classes with labels (1,2,3), 178 patterns, 17 patterns each class, for training, rest of them for testing). (4) Forest data (54 features, 7 classes with labels (1,2,3,4,5,6,7), 7000 patterns (extracted 1000 per class from 581012 patterns), 400 patterns each class, for training, rest of them for testing). (5) Breast cancer (wdbc) data (30 features, 2 classes with labels (1 replaced for M, 2 replaced for B), 569 patterns, 30 patterns each class, for training, rest of them for testing.)

4.4.2 Face Data Sets

(i) ORL face data set [119] contains face images of 40 persons, 10 images per person amounting to 400 images in total. Each image is of dimension, 112×92 (PGM format). Images are with variation in lighting, facial expressions and with/without glasses. We used 5 images per person generated randomly for training and rest of them for testing. (ii) CMU face data set [25] contains face images of 20 persons and a total of 640 images, out of these 16 bad images are not considered. Each image is of dimension, 120×128 (PGM format). The face images were taken with varying pose, expression, eyes (wearing sunglasses or not) and size. We used 10 images per person generated randomly for training and rest of them for testing. (iii) Yale face

data set [184] consists of gray-scale images of 15 persons under 11 different conditions (including different lighting, facial expressions and occlusion effects) amounting to 165 images in total. Each image is of dimension, 243×320 (PGM format). We used 5 images per person generated randomly for training and rest of them for testing.

4.4.3 Experimental Setup

We conducted 10 experiments using different training and test data sets for UCI repository. Each experiment is conducted as follows: We have chosen the number of sub-patterns, k, to minimize the truncation of last features as far as possible. For different data sets, k is chosen as follows: For waveform data, $k = 3$; for musk data, $k = 2$; for wine data, $k = 2$; for forest data $k = 6$; for breast cancer data, $k = 3$; for ORL face data, $k = 8$; for Yale face data $k = 10$ and for CMU face data set, $k = 2$. In case of PCA $k = 1$ for all the data sets. Dimensionality reduction of the data is done using SubXPCA algorithm (Section 4.2). For SubPCA and PCA Step-2 of SubXPCA algorithm is omitted. The classification is done based on Nearest Neighbour rule using reduced training data for the reduced test data sets.

The average of 10 classification results obtained for SubXPCA, SubPCA and PCA are shown in tables 4.1 to 4.3 and Figs. 4.7 to 4.10. The corresponding total execution time is specified in parentheses.

We conducted an experiment by using a more efficient implementation of classical PCA [103] for each of the face data sets and results are shown in Figs. 4.11 to 4.16. We observed that the original implementation of PCA takes enormous amount of time for image data, which is not used here. We generated training and testing

data sets randomly as specified in the preceding subsection. We used Pentium 4 based system with 2.4 GHz CPU clock speed, 256 MB RAM and Fedora Core 5 Linux running on it, to obtain experimental results. We used C language built-in time functions for recording time and procedures, viz. $tqli, tredt, eigensrt$, to find eigenvectors, eigenvalues and for sorting them from [127].

4.4.4 Experiments on Feature-Order Dependence and Overlapping Sub-Patterns

We generated 20 different feature orders randomly for a training set of Musk and Wine data and obtained results for SubPCA and SubXPCA techniques. The variance among these 20 classification rates is plotted in Figs. 4.19 and 4.20 for musk and wine data sets. From Figs. 4.19 and 4.20, it is clear that SubXPCA is relatively more robust against different feature orders as compared to SubPCA. We used the technique of overlapping features between pair of successive sub-patterns to see the change in performance. We used 3 overlapping features for waveform data and 10 overlapping features for forest data. The obtained results are plotted and compared with non-overlapping case in Figs. 4.17 and 4.18. From Figs. 4.17 and 4.18 the following facts are evident: For waveform data, SubPCA improves its classification rate slightly with overlapping of sub-patterns, however SubXPCA with non-overlapping sub-patterns option outperforms all other cases. For Forest data, both SubPCA and SubXPCA coincide with respect to classification for both overlapping and non-overlapping cases. SubPCA with overlapping sub-patterns shows poor performance as compared to both non-overlapping sub-patterns with either SubPCA or SubXPCA.

4.4.5 Summarization of Variance by SubXPCA, SubPCA and PCA

Summarization of most of the variance of the data in first few PCs enables high dimensionality reduction. A glimpse of summarization of variance by SubXPCA, SubPCA and PCA on UCI databases is given in Figs. 4.3 to 4.6. As shown in Figs. 4.3 to 4.6, both SubXPCA and PCA summarize most of the variation in first few PCs in comparison to SubPCA. In waveform data, SubXPCA and PCA summarizes 45.5% of total variance in first PC, whereas SubPCA summarizes only 19.1 % in the first PC. In musk data, SubXPCA and PCA summarises 31.37% of total variance in the first PC, whereas SubPCA summarizes only 16.14 % in the first PC. In wine data, all the three techniques show almost the same summarization of variance in the first PC, this is due to absence of inter-sub-pattern correlations in the data. In Forest data, SubXPCA and PCA summarize 73.71% of total variance in the first PC, whereas SubPCA summarizes only 57.45% in first PC. In the breast cancer data, SubXPCA and PCA summarize 99.11% of total variance in the first PC (indicating high inter-sub-pattern correlations), whereas SubPCA summarizes only 75.95 % in the first PC.

In a nutshell, SubXPCA shows good performance in terms of summarization of variance like PCA, where as SubPCA fails to show the good summarization of variance like PCA and SubXPCA.

4.4.6 Discussion of Experimental Results

The following facts are revealed by our experimentation on various data sets. SubXPCA shows superiority over SubPCA (i) in terms of summarization of variance, which leads to better dimensionality reduction i.e. SubXPCA uses less number of projection vectors (PVs), that is less number of eigenvectors (Columns 3 & 4 in tables 4.1 to 4.3 and Figs. 4.3 to 4.6), (ii) in terms of execution time for less number of PVs (eigenvectors) (Columns 5 & 6 in tables 4.1 to 4.3), (iii) in terms of classification rates (Figs. 4.7, 4.9, 4.10, 4.11, 4.13 & 4.15), (iv) in terms of robustness against different feature orders (Figs. 4.19 and 4.20) and (v) when overlapping is done among sub-patterns (Figs. 4.17 and 4.18).

SubXPCA also shows its superiority over PCA (i) in terms of execution time (Figs. 4.8, 4.12, 4.14 & 4.16), (ii) in terms of classification rate (Figs. 4.7, 4.9, 4.10, 4.11, 4.13 & 4.15). Results in Figs. 4.3 to 4.6 show that SubXPCA and PCA are significantly the same and are better than SubPCA in terms of summarization of variance in the first few PCs. However, It is to be noted that (i) SubPCA is superior in terms of time complexity as compared to PCA and (ii) SubPCA may outperform PCA in terms of classification for the cases where *local structure* has better discriminative information (See Yale face results in Fig. 4.15). It is to be noted that SubPCA may also show relatively less classification rates with varying number of PCs in comparison to PCA (Figs. 4.7, 4.9, 4.10, 4.11, 4.13) for the cases where *global structure* has vital discriminative information.

In a nutshell, by observing (i) execution time (which includes computation of covariance matrices, finding eigenvalues and eigenvectors, classification time, etc.)

(ii) classification accuracies, (iii) summarization of variance and (iv) feature-order dependency, one can appreciate the superior performance of SubXPCA over SubPCA and PCA. Interestingly, SubXPCA shows excellent recognition when local structure has better information (See Yale faces in Fig. 4.15) and also when global structure has better information (See CMU faces in Fig. 4.13). *Please note that SubXPCA is able to perform well even when either PCA or SubPCA does not perform well.*

4.5 Discussion: Why is SubXPCA Better than SubPCA and PCA?

In this section, we explain the possible reasons for superior performance of our method, SubXPCA, over PCA and SubPCA methods in various aspects.

4.5.1 SubXPCA Versus SubPCA

PCA is able to retain meaningful information (variance) in the major axes (eigenvectors corresponding to largest eigenvalues), where as variance associated to experimental error, measurement inaccuracy, and/or rounding (i.e. noise) is summarized in minor axes (eigenvectors corresponding to smaller eigenvalues) [50]. SubPCA considers all $k.r$ local features (extracting $r\,(< u)$ from each of k sub-patterns), which may also include less-expressive, noisy or correlated features. SubPCA ignores correlations among these local features which may result in low dimensionality reduction due to poor summarization of variance. SubXPCA exploits these correlations to summarize the variance among those $k.r$ local features and selects a few salient features,

$w\,(< k.r)$, thus results in better summarization of variance and high dimensionality reduction (Figs. 4.3 to 4.6). Those w features may increase the classification accuracy as well. The same is observed in our experiments (Tables 4.1 to 4.3 and Figs. 4.7 to 4.16). As we observed in our experiments, the overall time complexity of SubXPCA would be lower or competitive to the time complexity of SubPCA, because of two reasons: (i) in most of the cases, computation of additional covariance matrix ($\mathbf{C}^g$) becomes trivial as we choose the first few salient features in each sub-pattern (i.e. r is small) (ii) we choose less number of final features (i.e w is small due to better summarization of variance) in comparison to total features ($k.r$) extracted in SubPCA ($w \ll k.r$) (Step-4(b) in section 4.2), thus reducing time for subsequent tasks such as classification, recognition, etc. SubXPCA takes more time than SubPCA if the number of eigenvectors chosen is equal in both the cases.

From our experimentation, it is observed that SubXPCA is also relatively more feature order independent as compared to SubPCA (Figs. 4.19 and 4.20) and SubXPCA shows better performance as compared to SubPCA with overlapping subpatterns (Figs. 4.17 and 4.18). An interesting characteristic of SubXPCA is that, it is able to do well, when global variations are prominent, in which case SubPCA does not perform well (Fig. 4.13).

4.5.2 SubXPCA Versus PCA

In Theorem 3, we prove that the time complexity of SubXPCA (T_F) is less than the time complexity of PCA (T_C). From Theorem 4, it is evident that the time complexity of SubXPCA could be ideally $\frac{1}{k}$ times that of PCA for suitable values

of k and r. Thus we show that SubXPCA is more efficient than PCA. As proved in Theorem 1, if $r = u$ in Step-3 of SubXPCA, PCA can be derived as a special case of SubXPCA, where r, u are eigenvectors chosen from each sub-pattern set and sub-pattern size respectively. Thus as the number of local eigenvectors (r) in each sub-pattern increases, SubXPCA very closely approaches PCA in terms of summarization of variance (Figs. 4.3 to 4.6). In our experiments, we find that SubXPCA shows superior execution times and better or competitive classification rates than PCA (Figs. 4.7 to 4.16). An interesting characteristic of SubXPCA is that, SubXPCA is able to show better performance when local variations are prominent, in which case PCA does not perform well (Fig. 4.15).

4.5.3 SubPCA Versus PCA

As observed in section 2.2 of Chapter 2, SubPCA [21] does not consider correlations across different sub-patterns which may result in poor summarization of variance (and hence low dimensionality reduction) in comparison to PCA (since PCA considers entire correlation structure of the data). As explained in [50], noisy and less expressive features contain less variance. Such noisy features may reduce classification rate as well. In SubPCA, there is possibility of relatively more (possibly correlated) noisy or less-expressive features which are spread across different sub-patterns, and there is no mechanism to retain them. Thus classification accuracies of SubPCA may not be encouraging, when global variations are predominant in the data. Since PCA exploits the entire correlation structure in the data (hence it is global scheme), it shows better summarization of variance (which implies high dimensionality reduction) and may

lead to better classification rates in comparison to SubPCA, when global variations are prominent. The same is observed in our experiments (Figs. 4.3-4.6, 4.7, 4.9, 4.10, 4.11, 4.13). Interestingly, SubPCA performs better than PCA in terms of classification when the local structure (local features) has good discriminative information (Fig. 4.15 for Yale face data). Another advantage of SubPCA is, it shows much lesser time complexity as compared to PCA (Theorem 2 and Figs. 4.8, 4.12, 4.14 & 4.16).

We conclude that SubXPCA is flexible to extract *local* information like SubPCA (an existing FP-PCA method) and also extracts *global* information like classical PCA (a global PCA method). Thus SubXPCA captures the best characteristics of SubPCA and PCA.

4.6 Summary

In this Chapter, we proposed an FP-PCA approach, SubXPCA technique as a solution to the issues 'loss of inter-sub-pattern correlations' and 'feature order dependency'. SubXPCA improves the summarization of variance, reduces the dimensionality and improves the classification accuracy, thus saving computational and processing costs in subsequent usage of the data as compared to SubPCA. It is found that the overall time complexity of SubXPCA is less than overall time complexity of SubPCA when less number of eigenvectors are used. SubXPCA is more robust to SubPCA with different feature orders. By our analysis we show that PCA is reduced to a special case of SubXPCA. It is also observed that the time complexity of SubXPCA is less than PCA for sizable number of features. Interestingly, SubPCA may outperform PCA in terms of recognition for the cases where local structure has better discrimi-

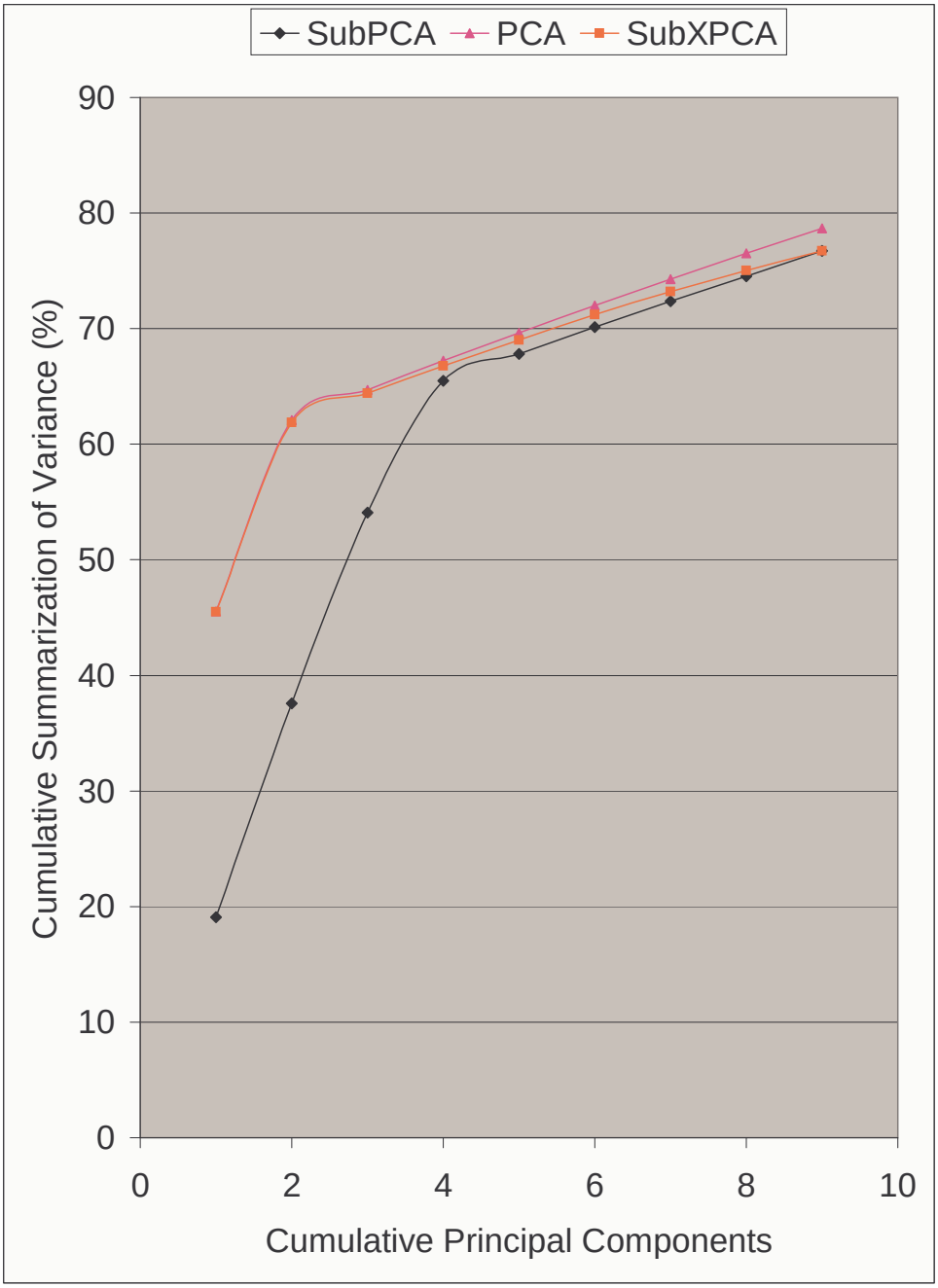


Figure 4.3: *Summarization of variance in PCs for Waveform data.* SubXPCA and PCA show similar values and are superior to SubPCA in terms of summarization of variance. 3 PVs (eigenvectors) per sub-pattern set were used for SubXPCA.

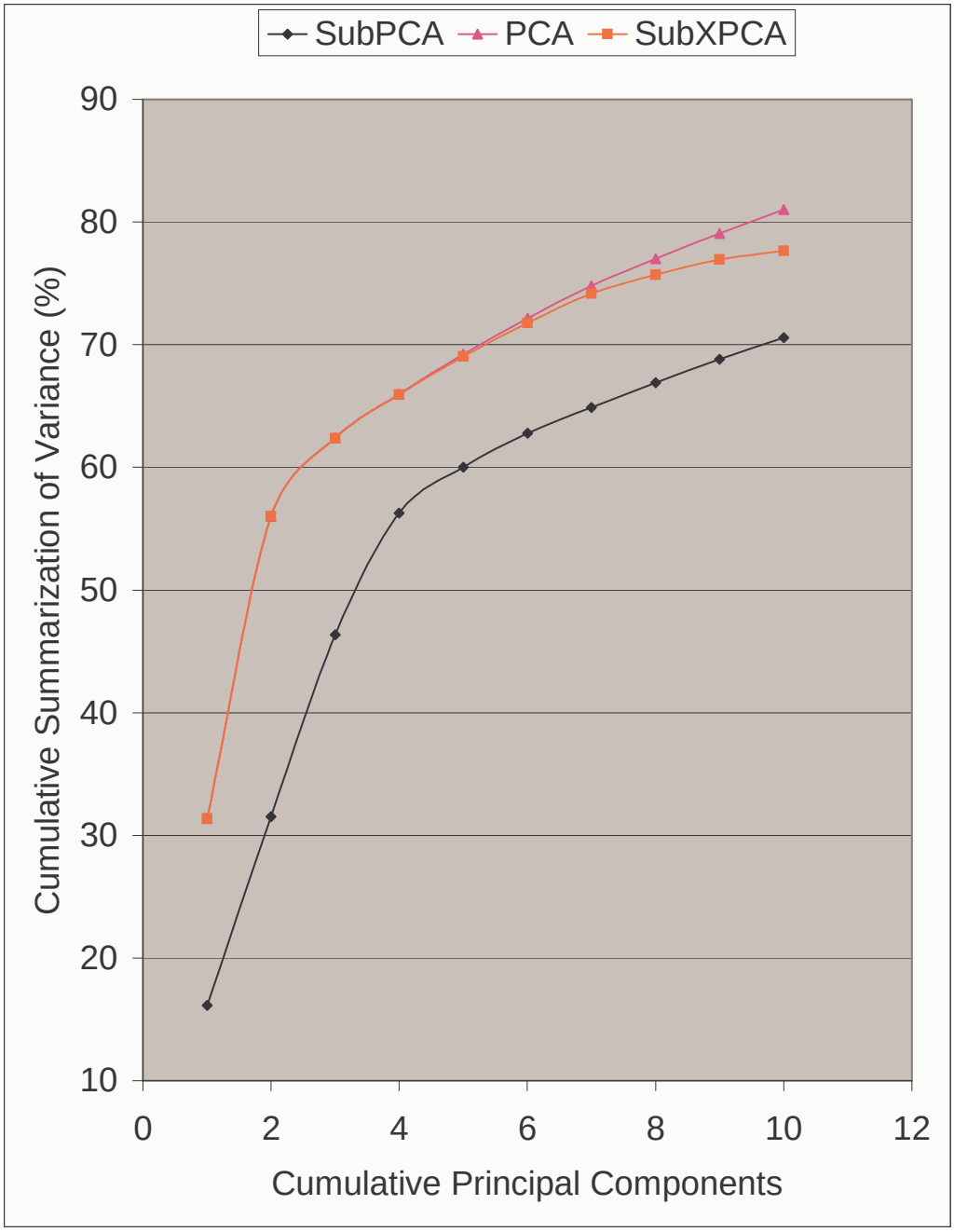


Figure 4.4: *Summarization of variance in PCs for Musk data.* SubXPCA and PCA show similar values and are superior to SubPCA in terms of summarization of variance. 8 PVs (eigenvectors) per sub-pattern set were used for SubXPCA.

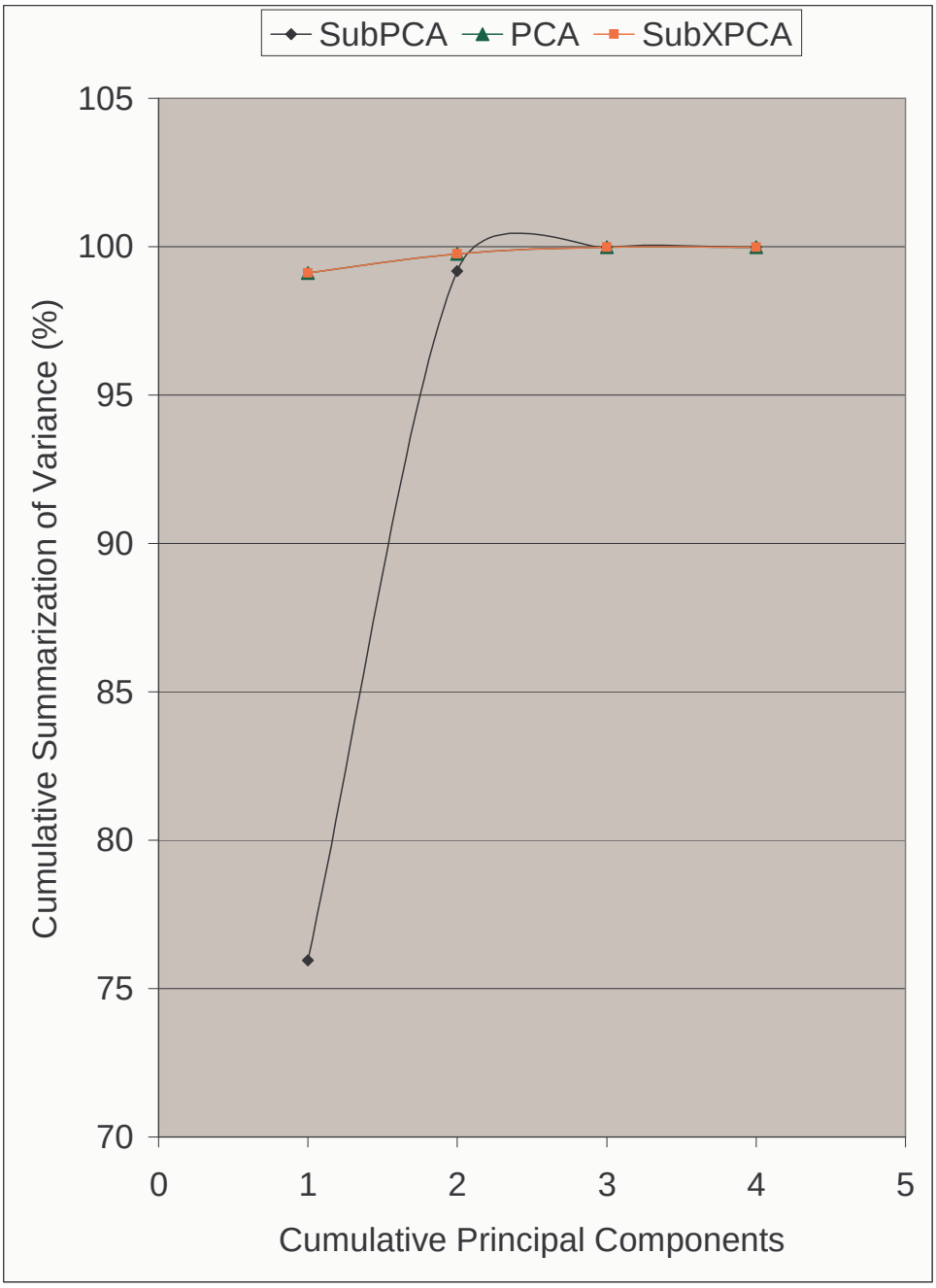


Figure 4.5: *Summarization of variance in PCs for Breast Cancer data.* SubXPCA and PCA show same values and are superior to SubPCA in terms of summarization of variance. 4 PVs (eigenvectors) per sub-pattern set were used for SubXPCA.

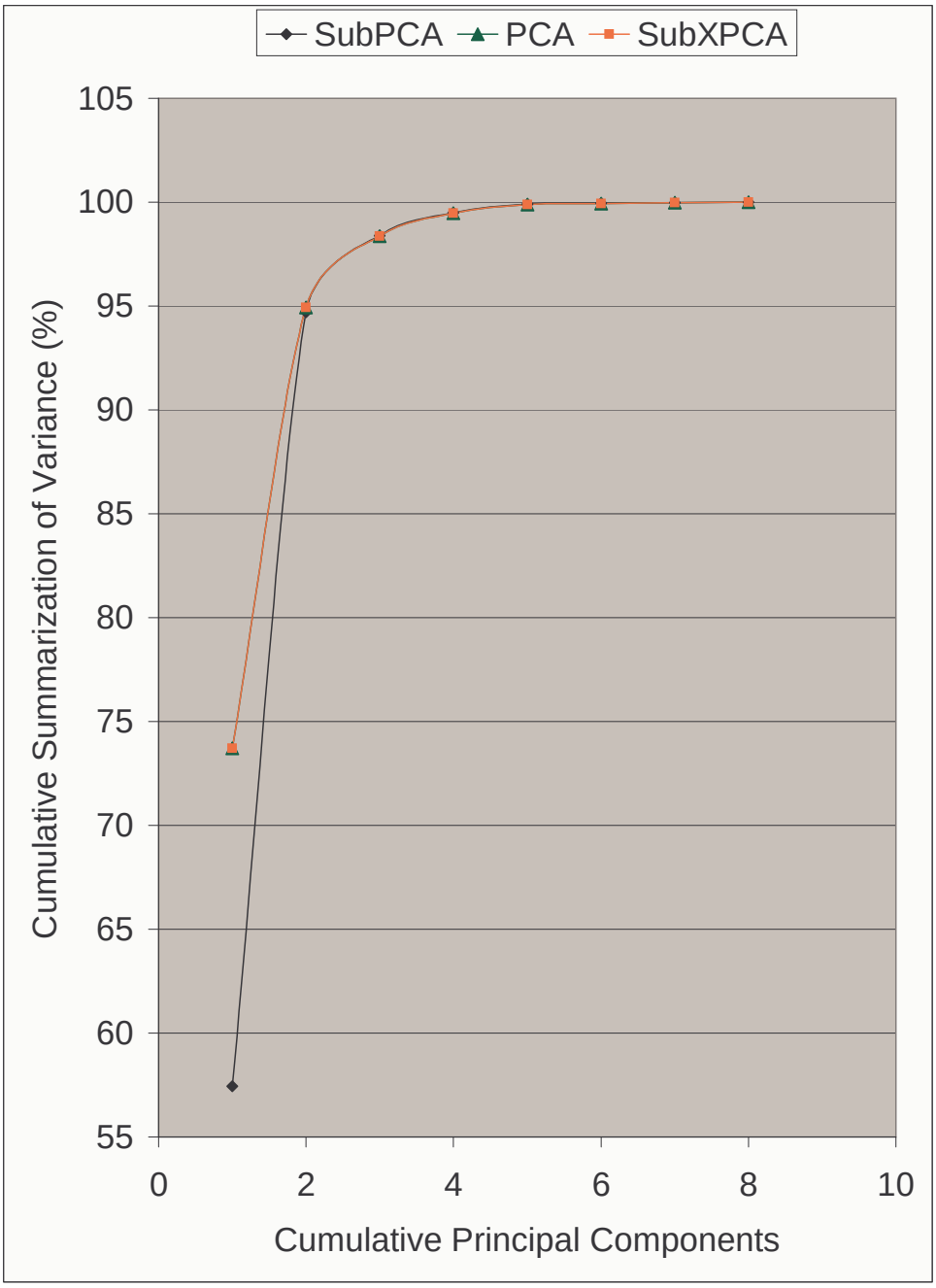


Figure 4.6: *Summarization of variance in PCs for Forest data.* SubXPCA and PCA show same values and are superior to SubPCA in terms of summarization of variance. 7 PVs (eigenvectors) per sub-pattern set were used for SubXPCA.

native information. We observed that SubXPCA is flexible enough (i) to capture the best characteristics of SubPCA (local feature extraction and less time complexity) and (ii) to capture the best characteristics of PCA (extraction of global features and good summarization of variance). The experimentation has shown SubXPCA to be as faithful and reliable as PCA in terms of summarization of variance. The proposed FP-PCA technique, SubXPCA, can be widely used in bio-informatics, data mining applications, palm print recognition, face recognition, etc.

We observed that FP-PCA methods (including SubXPCA) reduce small sample size (SSS) problem because the sub-patten size can be chosen (controlled) to be less than sample size in these FP-PCA methods.

The FP-PCA methods (Section 2.2 of Chapter 2) including SubXPCA and Sub-PCA, use classical PCA as local feature extraction method. Classical PCA treats an image pattern as a vector by collapsing the image feature matrix. This makes PCA not to exploit inherent matrix structure of image patterns and computationally more expensive. Thus the need arises to use a different variation of PCA in feature partitioning framework to extract features from image data. In the next Chapter, we address these concerns by proposing novel FP-PCA methods for image patterns.

Table 4.1: Classification accuracies based on Nearest Neighbour rule: SubPCA versus SubXPCA

Data set	**PVs per sub-pattern set**	**Total no. of PVs**		**Accuracy (%) & time (secs.)**	
		SubPCA	**SubXPCA**	**SubPCA**	**SubXPCA**
	(r)	$(k.r)$	(w)		
Waveform	1	3	2	72.7 (*2.65*)*	73.2 (*2.06*)*
$(21)^a(7)^b$	2	6	2	80.0 (*4.52*)	81.8 (*2.08*)
	3	9	2	79.3 (*6.44*)	81.8 (*2.17*)
	4	12	2	78.3 (*8.36*)	81.6 (*2.21*)
	5	15	2	77.7 (*10.44*)	81.7 (*2.22*)
Musk data	1	2	2	65.3 (*3.71*)	65.3 (*4.28*)
$(166)^a(83)^b$	2	4	4	68.2 (*4.52*)	68.2 (*5.06*)
	3	6	6	70.6 (*5.97*)	70.6 (*6.04*)
	4	8	8	71.7 (*6.94*)	71.7 (*7.05*)
	5	10	10	73.5 (*7.76*)	73.5 (*7.96*)
	7	14	14	74.0 (*9.82*)	74.0 (*9.96*)
	8	16	9	74.5 (*10.59*)	74.6 (*8.98*)
	9	18	10	74.5 (*11.49*)	74.6 (*9.67*)
	10	20	11	74.4 (*12.36*)	74.7 (*10.36*)

[a] The dimension of the original pattern; [b] The dimension of the sub-pattern
PV: Projection Vector, an eigenvector which is chosen for projection
* Figures in parentheses indicate corresponding execution times in seconds

Table 4.2: Classification accuracies based on Nearest Neighbour rule: SubPCA versus SubXPCA contd.

Data set	**PVs per sub-pattern set**	**Total no. of PVs**		**Accuracy (%) & time (secs.)**	
		SubPCA	**SubXPCA**	**SubPCA**	**SubXPCA**
	(r)	$(k.r)$	(w)		
Wine data	1	2	2	65.4 (*0.01*)	65.4 (*0.01*)
$(13)^a(6)^b$	2	4	4	77.8 (*0.01*)	77.8 (*0.01*)
	3	6	6	78.5 (*0.01*)	78.5 (*0.02*)
	4	8	6	79.6 (*0.02*)	80.0 (*0.02*)
	5	10	7	80.0 (*0.04*)	80.1 (*0.02*)
Forest data	1	6	2	36.3 (*26.94*)	36.3 (*17.34*)
$(54)^a(9)^b$	2	12	3	62.1 (*40.29*)	62.1 (*20.63*)
	3	18	4	69.1 (*53.40*)	69.1 (*22.70*)
	4	24	5	71.5 (*67.45*)	71.5 (*25.40*)
	5	30	6	71.7 (*81.28*)	71.7 (*27.66*)
	6	36	7	72.1 (*95.45*)	72.1 (*30.58*)
	7	42	8	72.7 (*109.09*)	72.7 (*33.18*)

[a] The dimension of the original pattern; [b] The dimension of the sub-pattern
PV: Projection Vector, an eigenvector which is chosen for projection
* Figures in parentheses indicate corresponding execution times in seconds

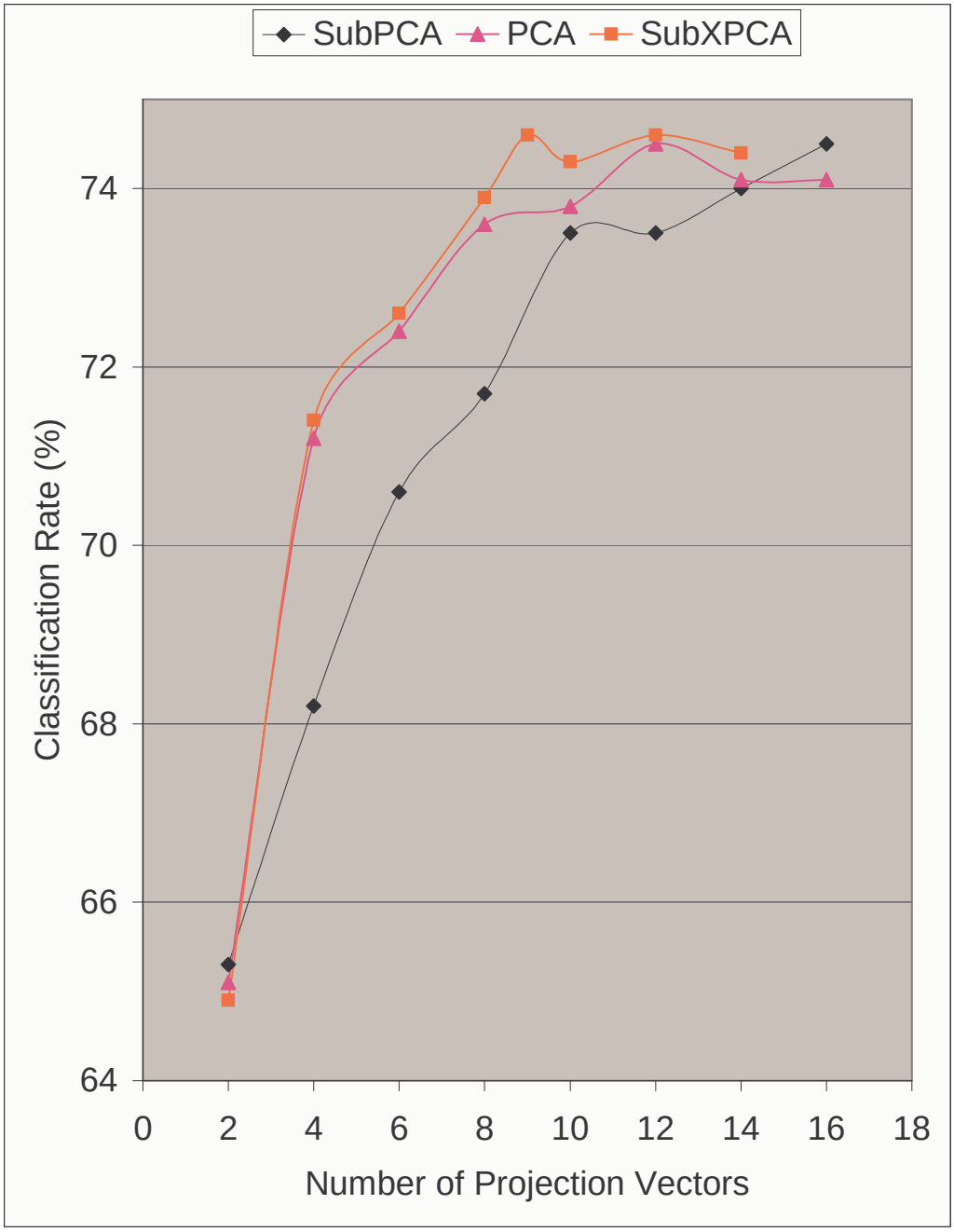


Figure 4.7: *Classification rate for Musk data.* SubXPCA shows relatively better classification rates as compared to both PCA and SubPCA methods. It is clear that SubXPCA shows higher classification rate by using *lesser principal components* as compared to other two methods. We used 8 PVs (eigenvectors) per sub-pattern set for SubXPCA. The average of classification rates out of 10 experiments is shown here.

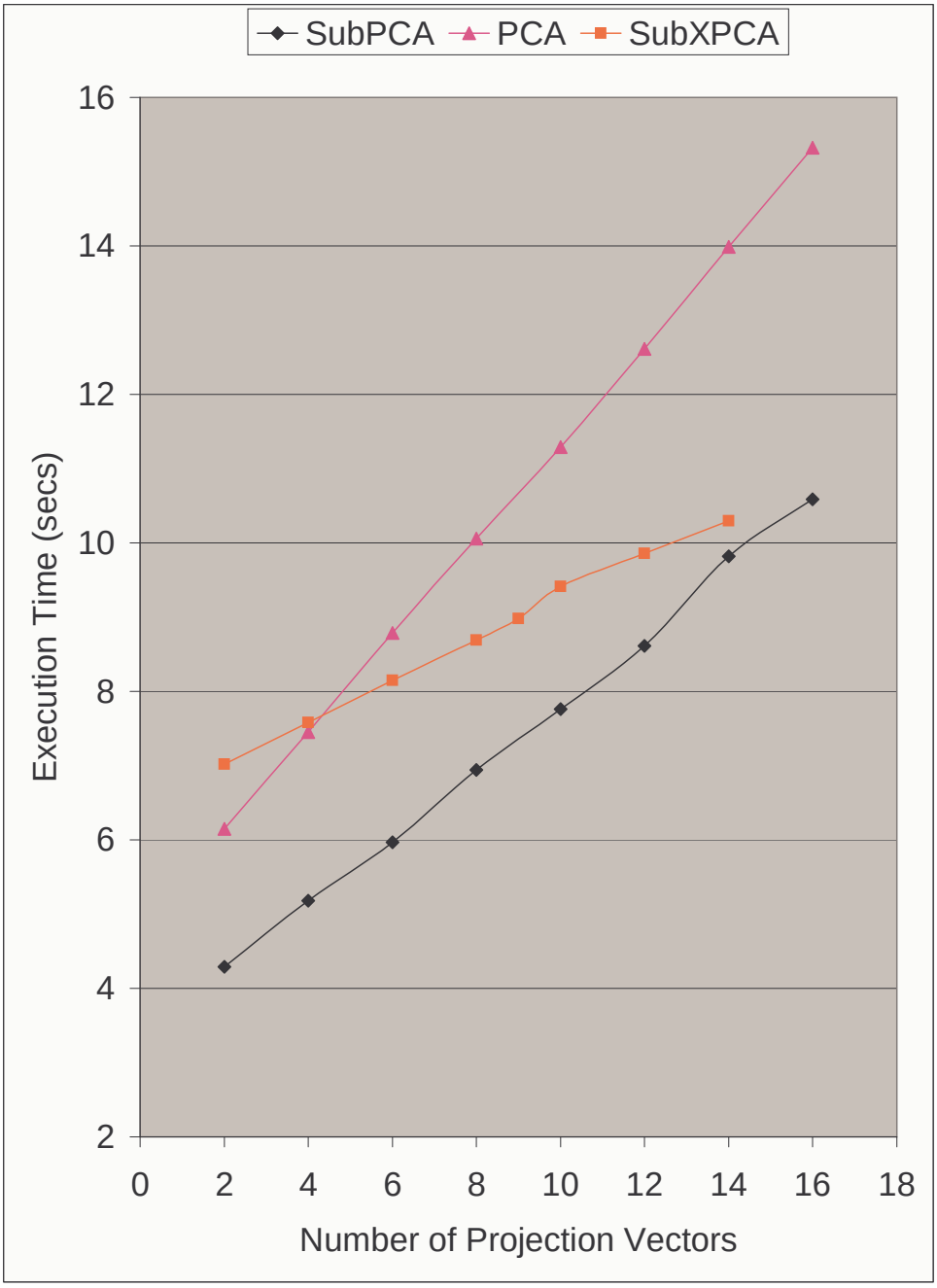


Figure 4.8: *Execution time for Musk data.* SubXPCA is computationally better than PCA and competitive to SubPCA. We used 8 PVs (eigenvectors) per sub-pattern set for SubXPCA.

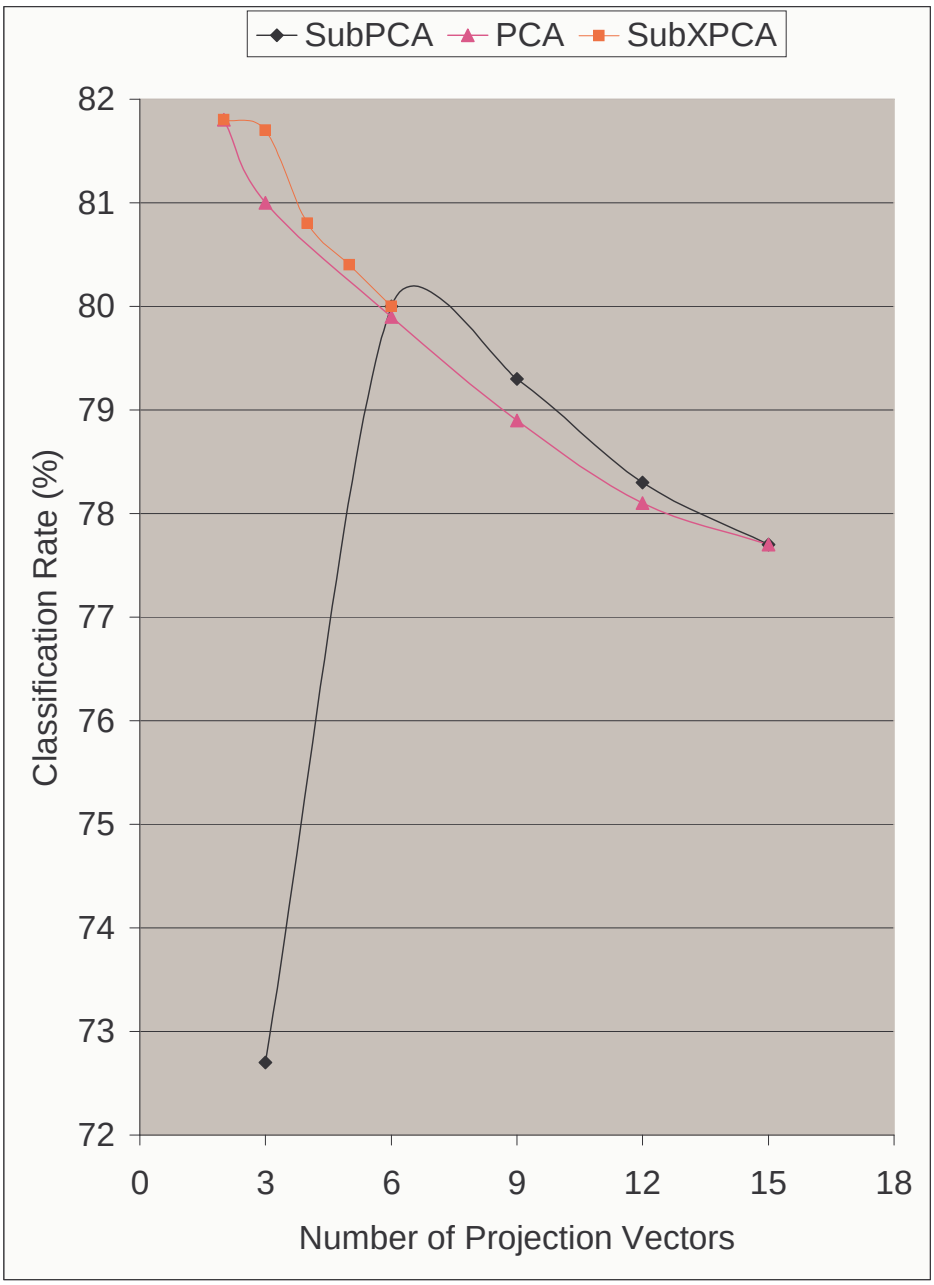


Figure 4.9: *Classification rate for Waveform data.* SubXPCA shows relatively better classification rates with different PVs (eigenvectors) as compared to SubPCA method. It is observed that SubXPCA and PCA show higher classification rate by using *lesser principal components* as compared to SubPCA. We used 2 PVs (eigenvectors) per sub-pattern set for SubXPCA. The average of classification rates out of 10 experiments is shown here.

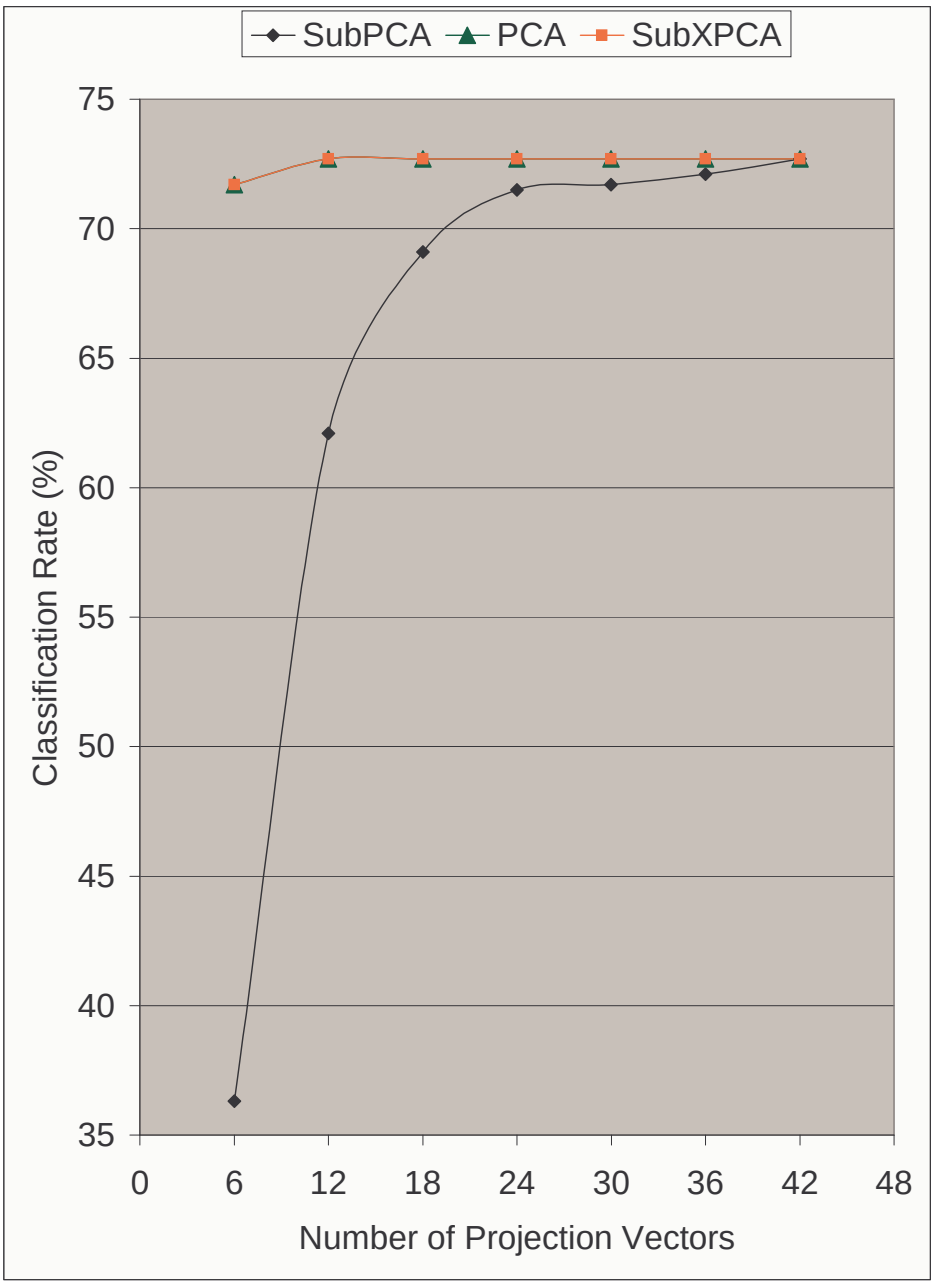


Figure 4.10: *Classification rate for Forest data.* SubXPCA shows better classification rates as compared to SubPCA. SubXPCA coincides with PCA's classification. It is noted that SubXPCA and PCA show higher classification rate by using *lesser principal components* as compared to SubPCA. Hence the curve related to PCA is not clear in the figure. 7 PVs (eigenvectors) per sub-pattern set were used for SubXPCA. The average of classification rates out of 10 experiments is shown here.

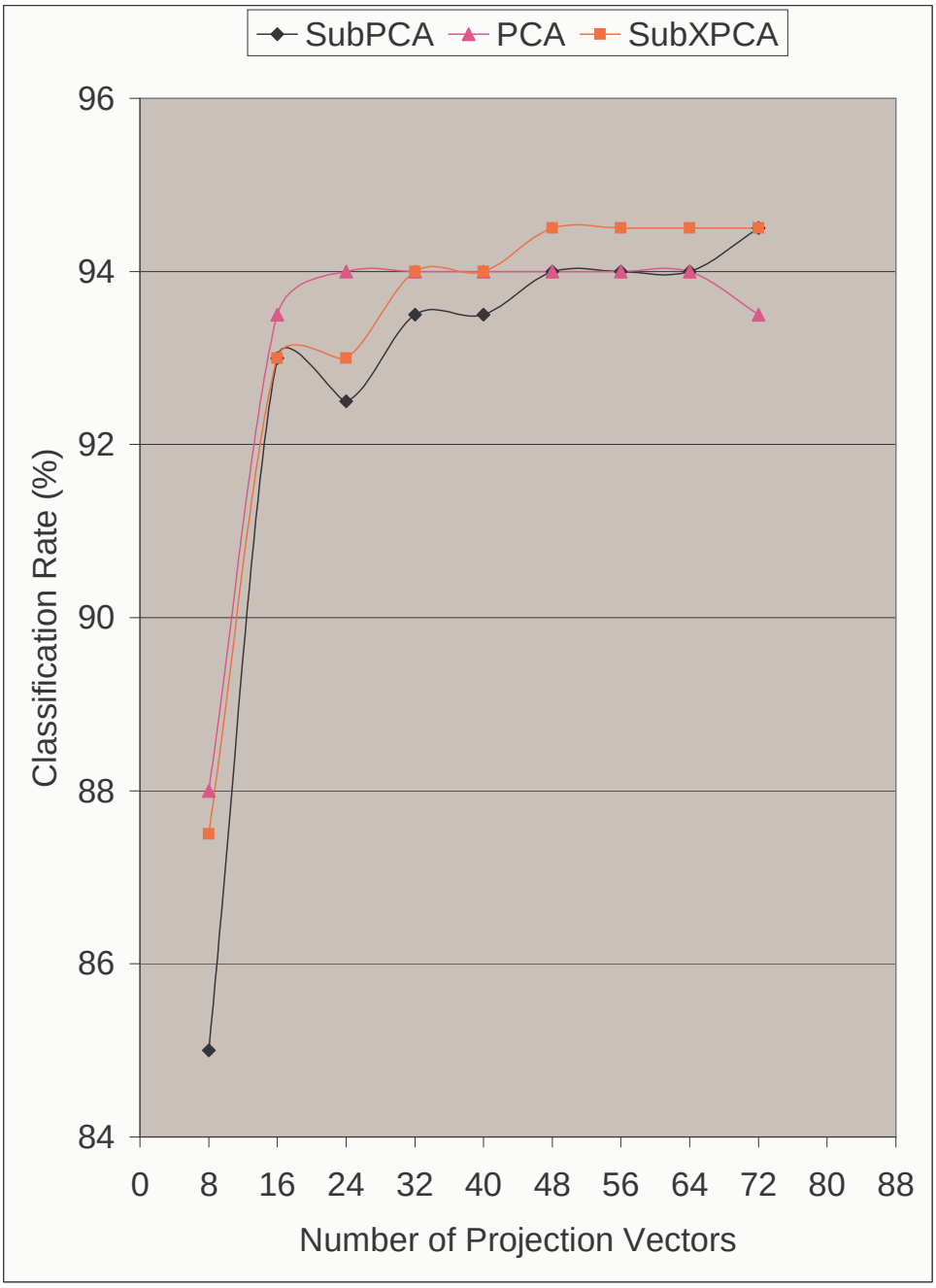


Figure 4.11: *Classification rate for ORL faces.* SubXPCA shows better recognition rate by using *lesser principal components* as compared to SubPCA. SubXPCA also shows its superiority as compared to PCA in terms of maximum recognition rate. We used 9 PVs (eigenvectors) per sub-pattern set for SubXPCA.

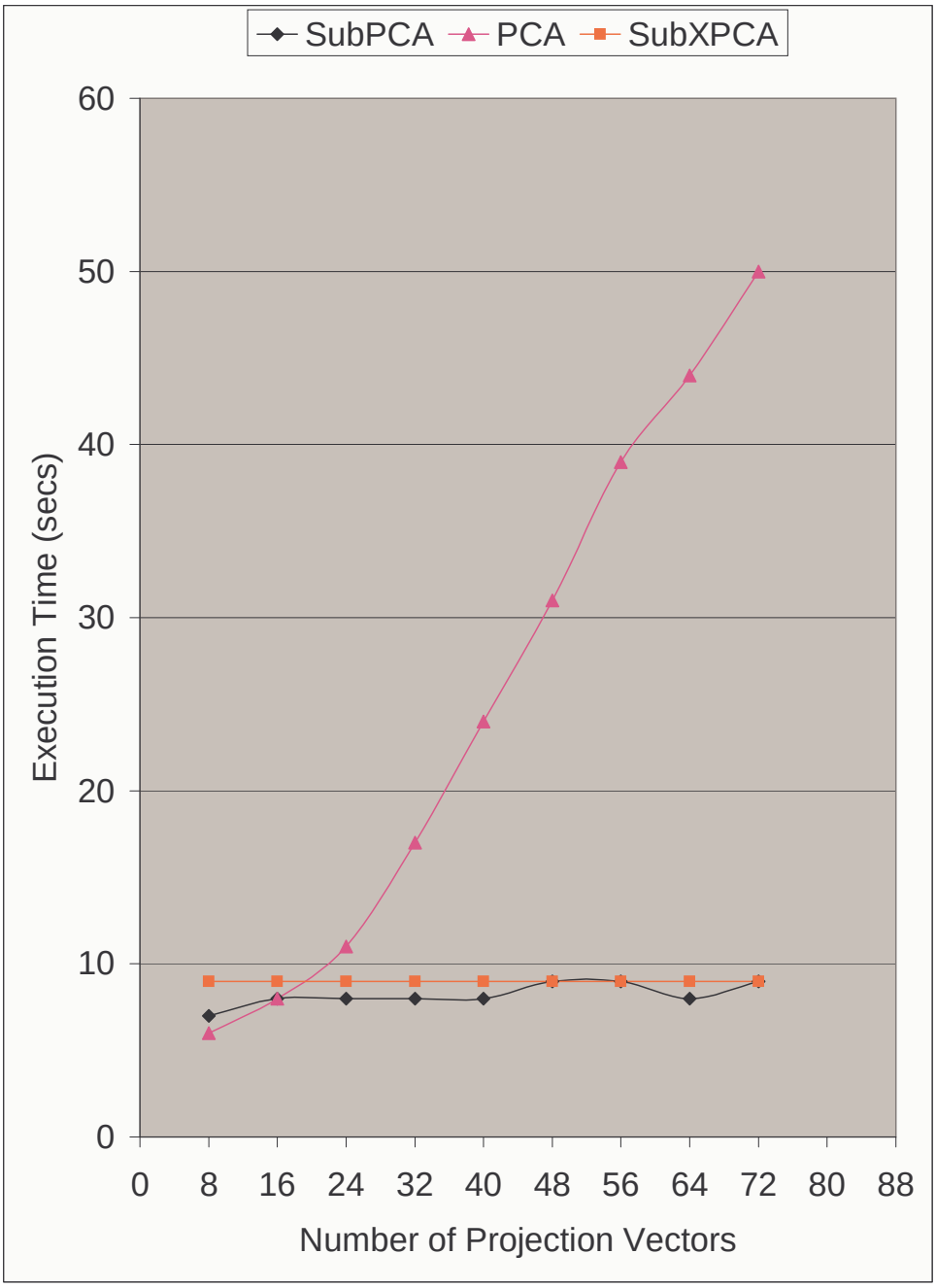


Figure 4.12: *Execution time for ORL faces.* SubXPCA and SubPCA are computationally similar and much superior to PCA. 9 PVs (eigenvectors) per sub-pattern set were used for SubXPCA.

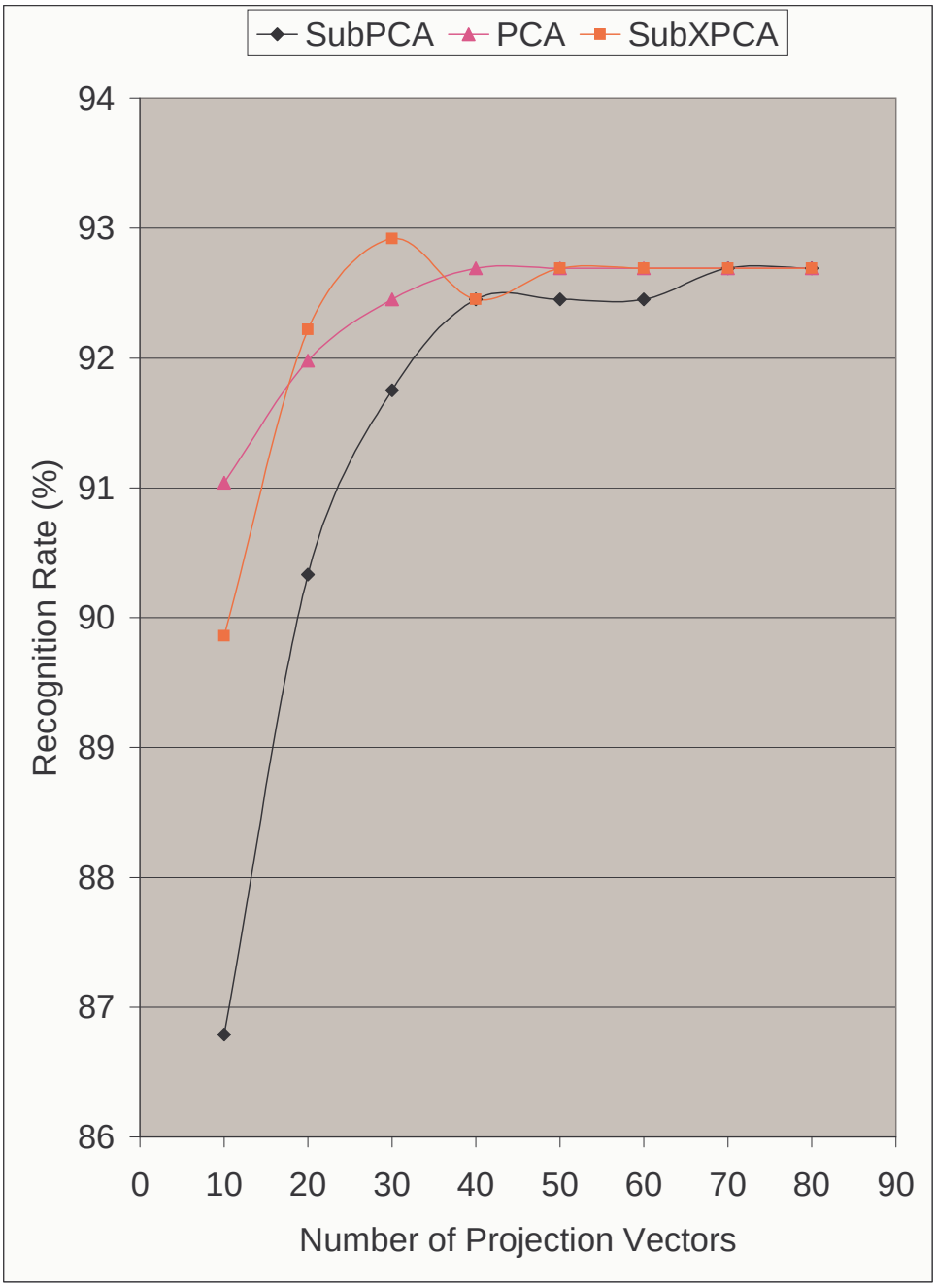


Figure 4.13: *Classification rate for CMU faces.* SubXPCA shows better recognition rate as compared to SubPCA. SubXPCA also shows its superiority as compared to PCA in terms of maximum recognition rate. Please note that SubXPCA shows higher recognition rate by using *lesser principal components* as compared to SubPCA and PCA. We used 40 PVs (eigenvectors) per sub-pattern set for SubXPCA.

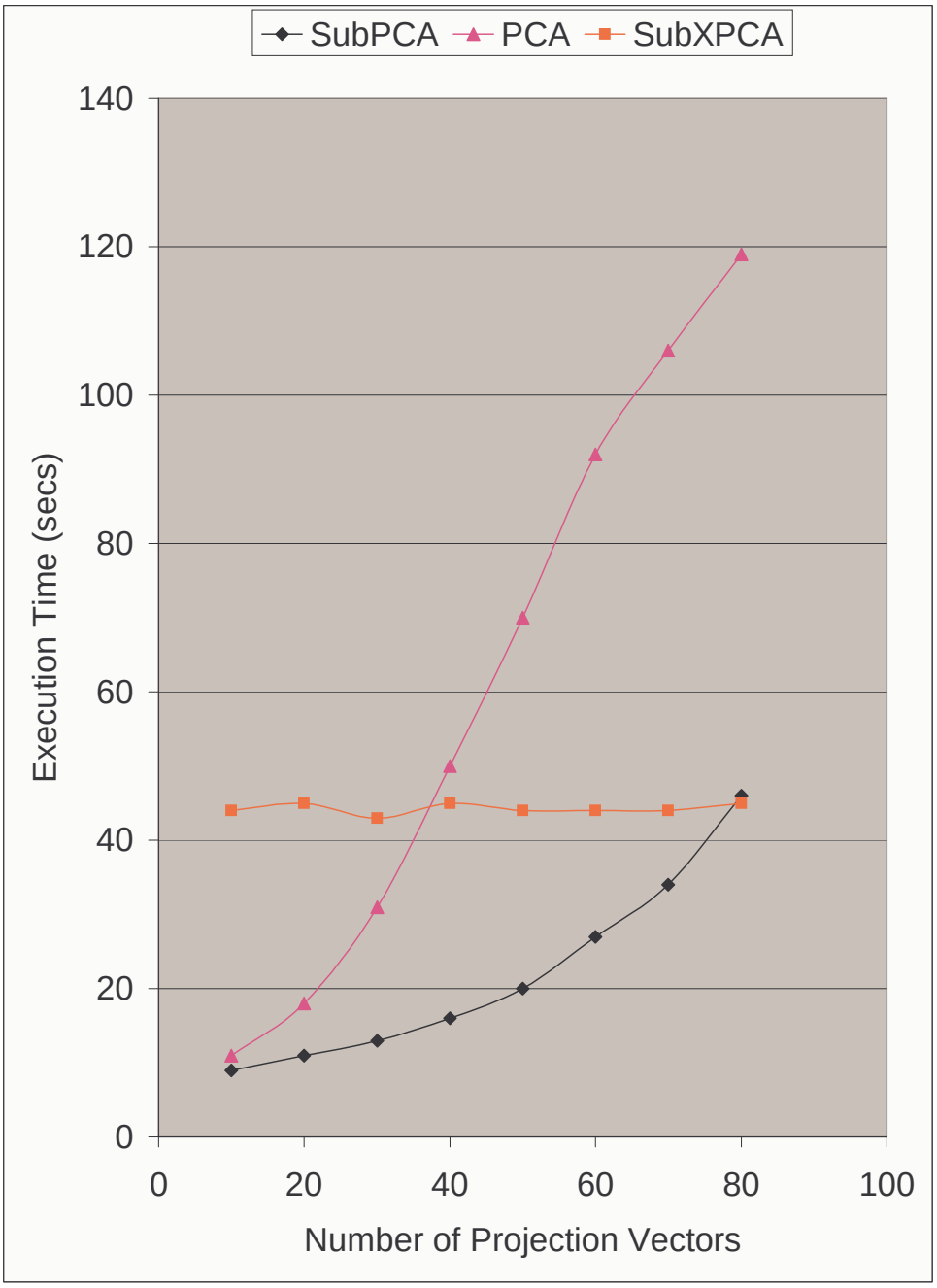


Figure 4.14: *Execution time for CMU faces.* SubXPCA is computationally better than PCA and competitive to SubPCA. 40 PVs (eigenvectors) per sub-pattern set were used for SubXPCA.

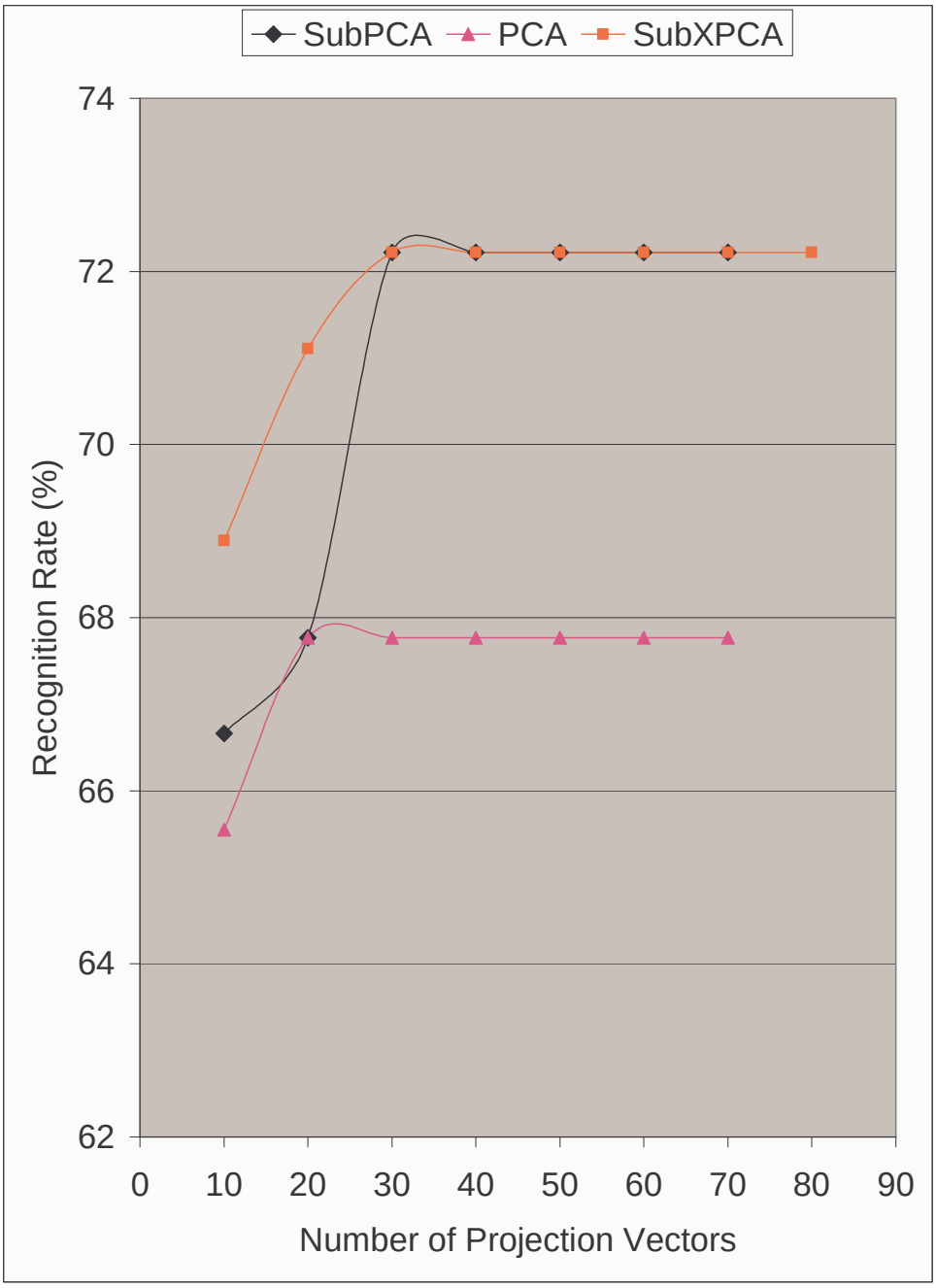


Figure 4.15: *Classification rate for Yale faces.* SubXPCA is better than SubPCA for 10, 20 PVs (eigenvectors); coincides with SubPCA for other PVs (eigenvectors) in terms of recognition. Please note that both SubXPCA and SubPCA outperform PCA in terms of recognition. We used 8 PVs (eigenvectors) per sub-pattern set for SubXPCA.

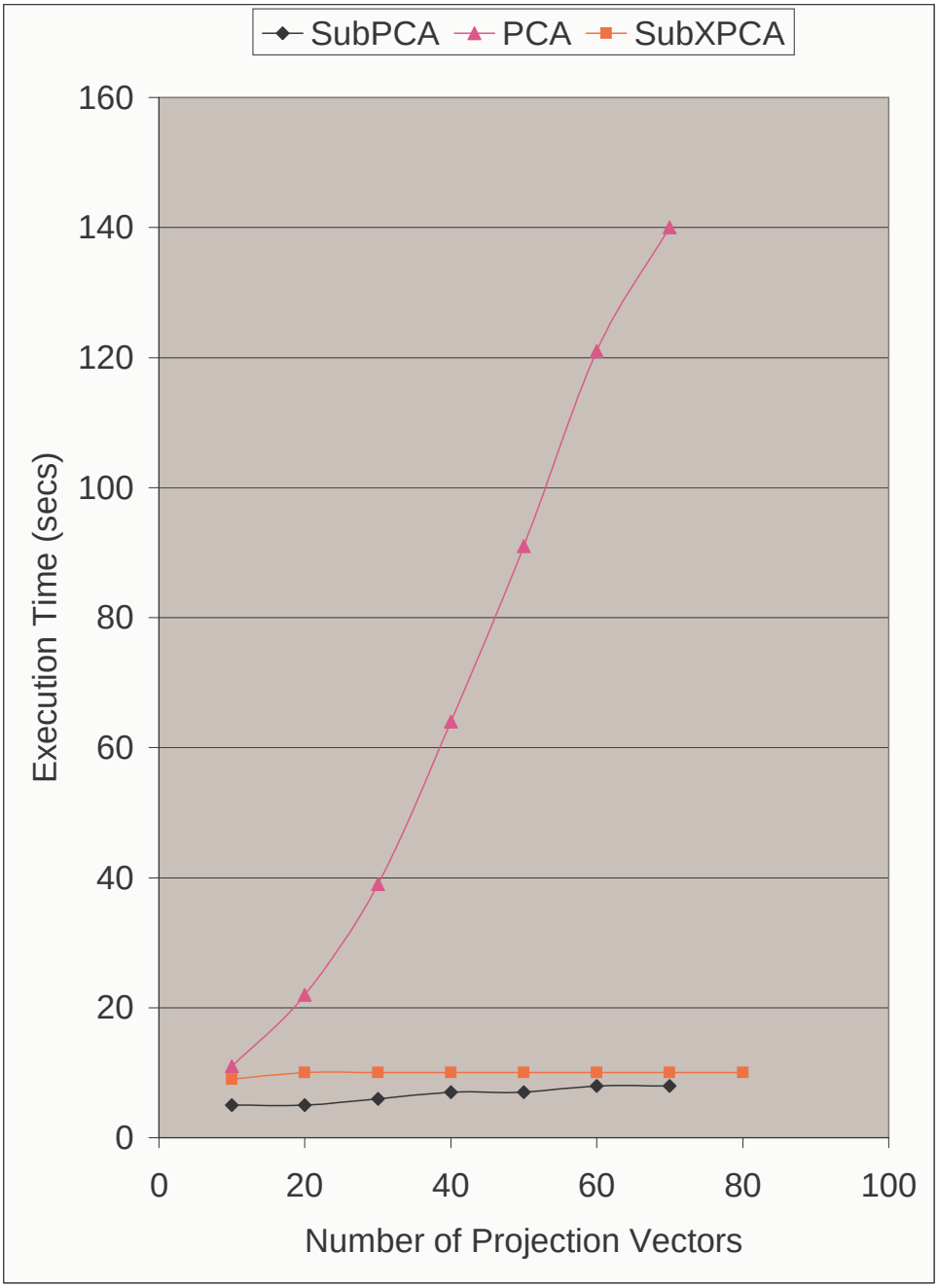


Figure 4.16: *Execution time for Yale faces.* SubXPCA and SubPCA are computationally similar and much superior to PCA. We used 8 PVs (eigenvectors) per sub-pattern set for SubXPCA.

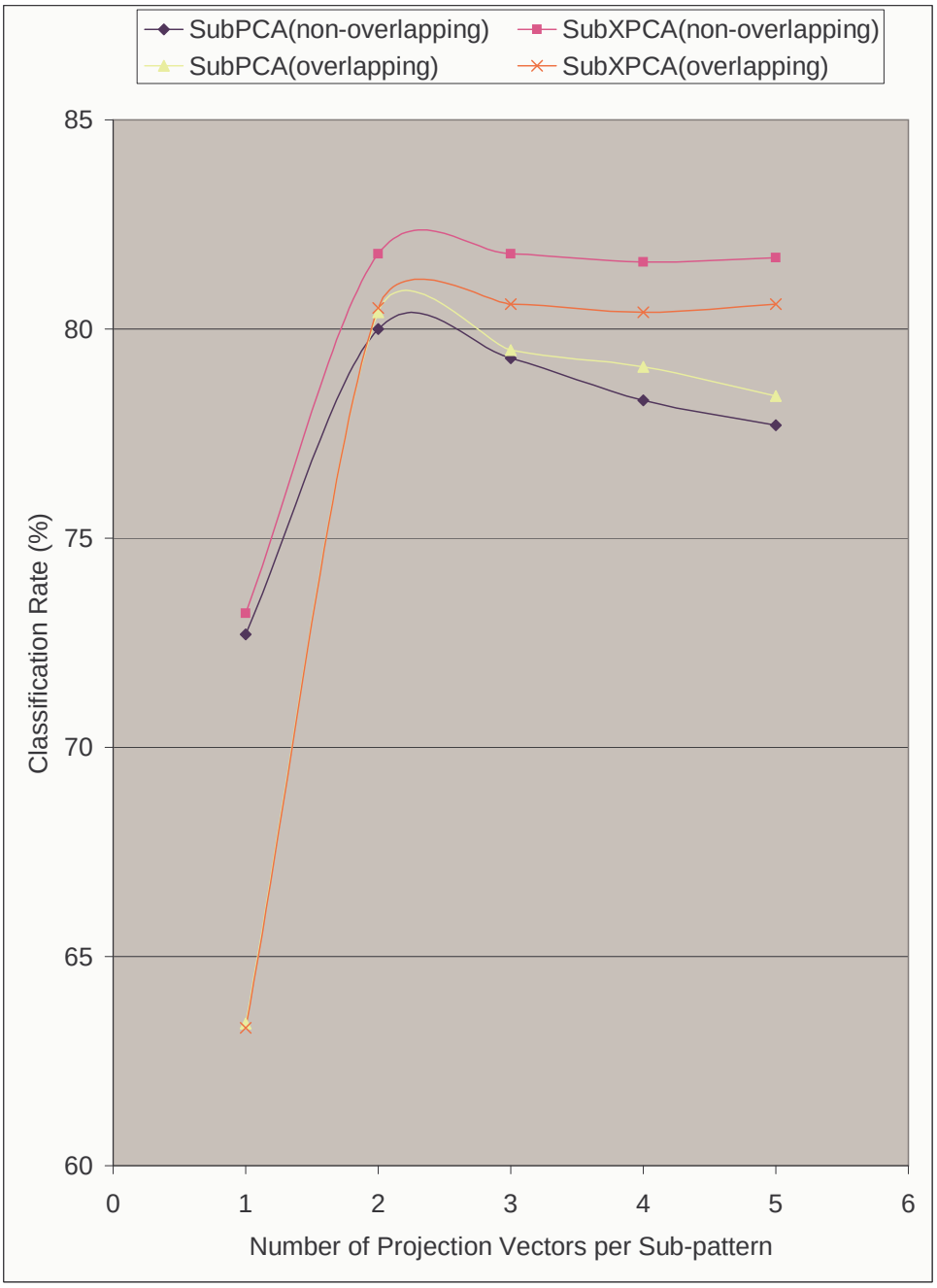


Figure 4.17: *Overlapping versus Non-overlapping sub-patterns for Waveform data.* SubPCA improves its classification rate slightly with overlapping of sub-patterns. However SubXPCA with non-overlapping sub-patterns option outperforms all other methods.

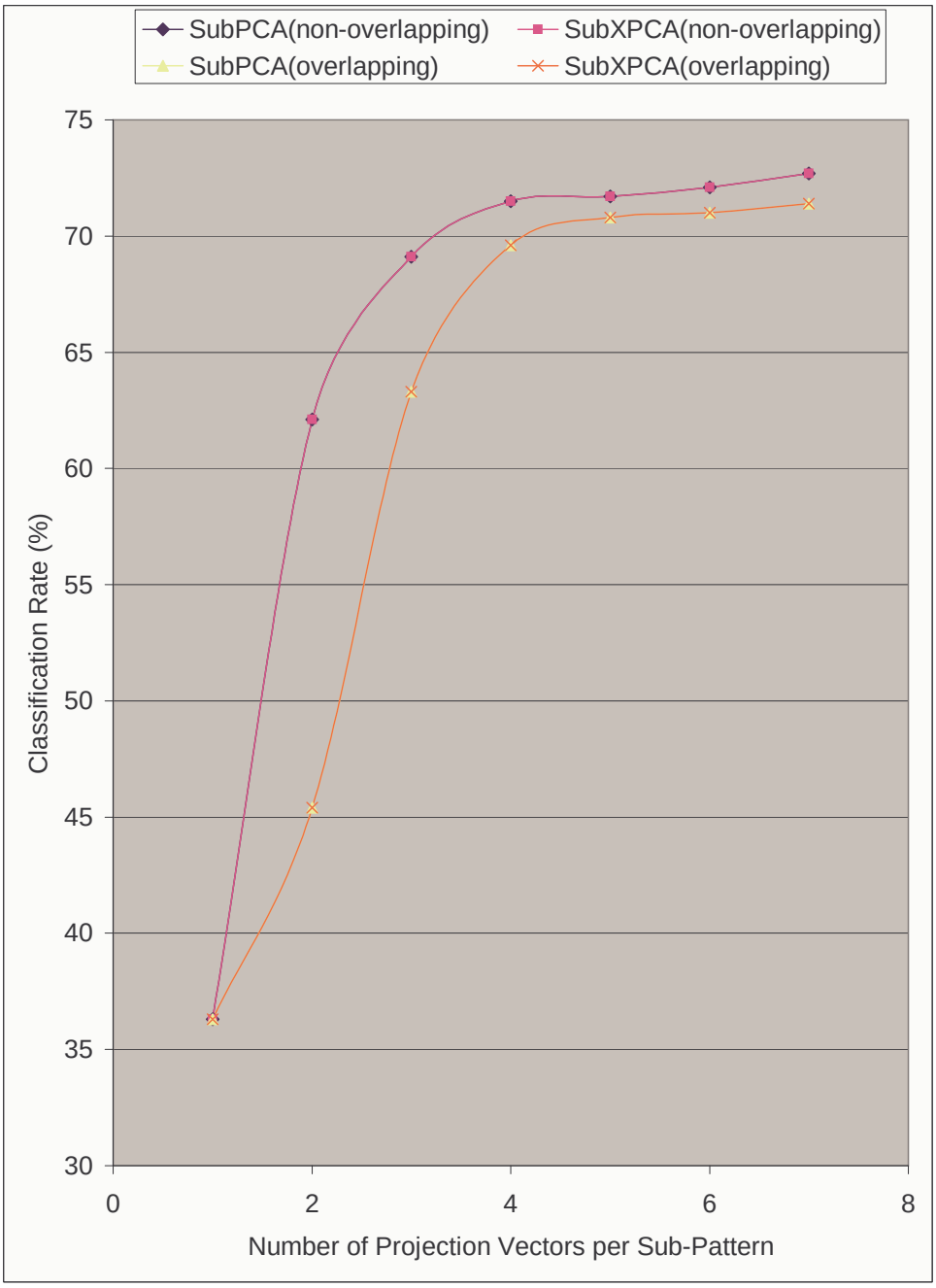


Figure 4.18: *Overlapping versus Non-overlapping sub-patterns for Forest data.* Both SubPCA and SubXPCA coincide with respect to classification for overlapping case and also for non-overlapping case. SubPCA with overlapping sub-patterns shows poor performance as compared to both non-overlapping sub-patterns with either SubPCA or SubXPCA.

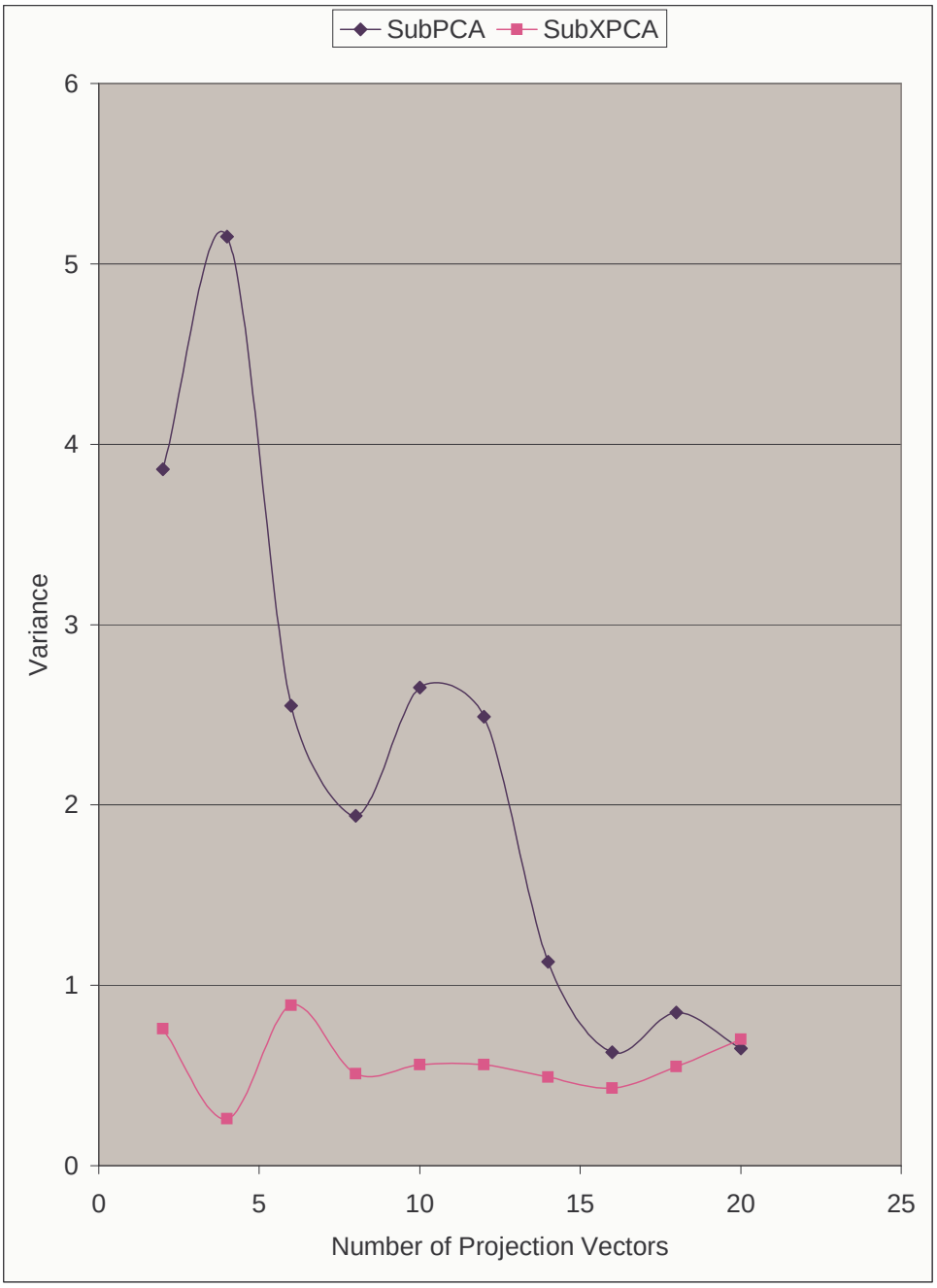


Figure 4.19: *Impact of Feature orders in Musk data.* SubXPCA shows more robustness against different feature orders as compared to SubPCA. SubXPCA uses 11 PVs (eigenvectors) for every sub-pattern set.

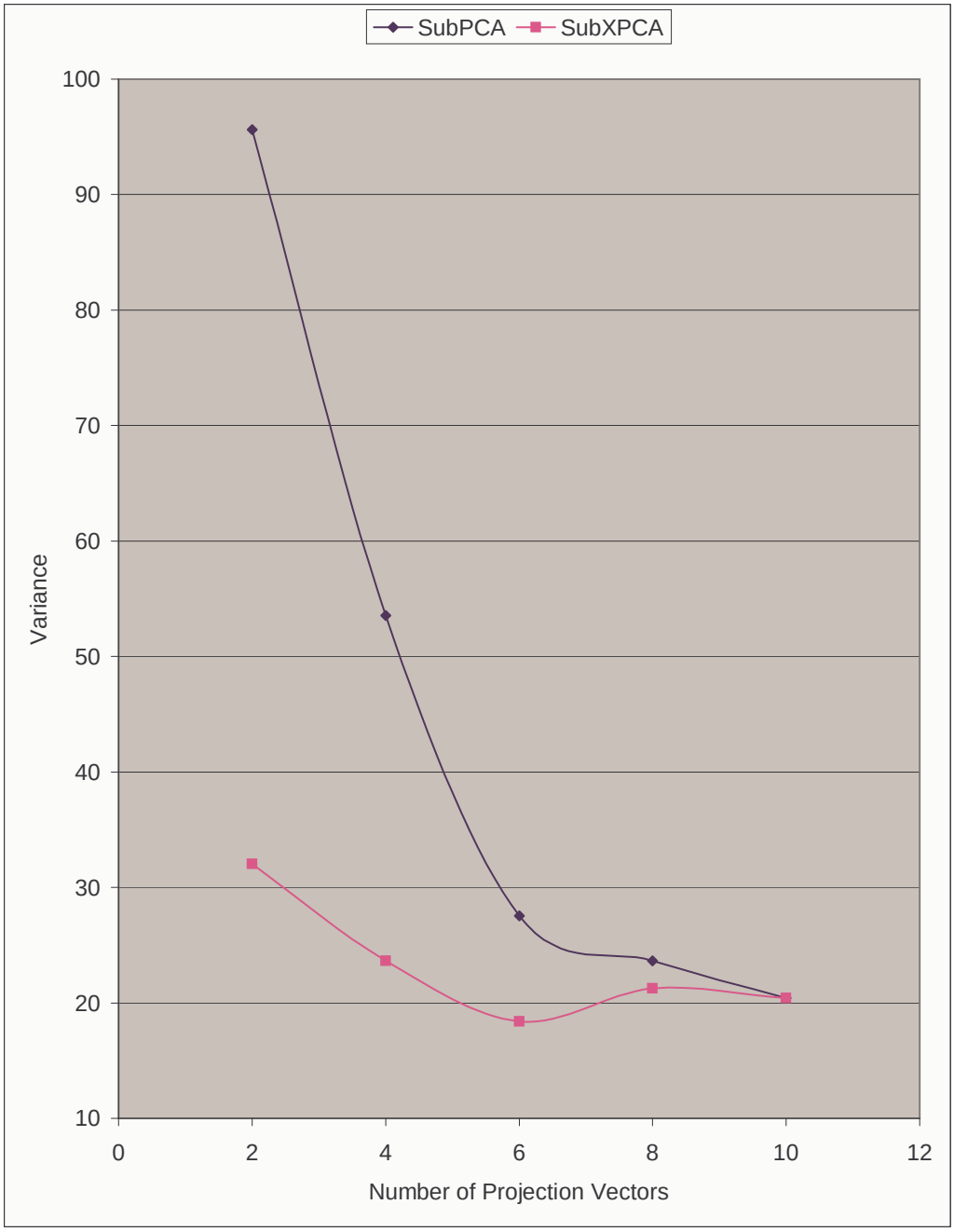


Figure 4.20: *Impact of Feature orders in Wine data.* SubXPCA shows more robustness against different feature orders as compared to SubPCA. SubXPCA uses 5 PVs (eigenvectors) for every sub-pattern set.

Table 4.3: Classification accuracies based on Nearest Neighbour rule: SubPCA versus SubXPCA contd.

Data set	**PVs per sub-pattern set**	**Total no. of PVs**		**Accuracy (%) & time (secs.)**	
		SubPCA	**SubXPCA**	**SubPCA**	**SubXPCA**
	(r)	$(k.r)$	(w)		
Breast cancer	1	3	3	89.3 *(0.07)**	89.3 *(0.07)**
$(30)^a(10)^b$	2	6	3	89.3 *(0.10)*	89.3 *(0.08)*
	3	9	5	89.4 *(0.15)*	89.4 *(0.09)*
	4	12	5	89.4 *(0.15)*	89.4 *(0.11)*
	5	15	5	89.4 *(0.18)*	89.4 *(0.12)*
	6	18	5	89.4 *(0.19)*	89.4 *(0.14)*
	7	21	5	89.4 *(0.23)*	89.4 *(0.14)*
	8	24	5	89.4 *(0.23)*	89.4 *(0.15)*

[a] The dimension of the original pattern; [b] The dimension of the sub-pattern
PV: Projection Vector, an eigenvector which is chosen for projection
* Figures in parentheses indicate corresponding execution times in seconds

Chapter 5

SIMPCA and FLPCA: Feature Partitioning Approaches to PCA for Image Data

5.1 Introduction

Note: The work in this chapter has been published in *Pattern Recognition Letters Journal (Elsevier Science)*[1] and in *proceedings of IAPR Conference on Machine Vision and Applications (IAPR-MVA 2007)*[2].

In the last chapter, the SubXPCA method was proposed which is a novel Fea-

[1]Kadappagari Vijaya Kumar and Atul Negi, "Novel approaches to principal component analysis of image data based on feature partitioning framework", *Pattern Recognition Letters*, Vol. 29 Issue 3, Feb. 2008, pp. 254-264.

[2]Kadappagari Vijaya Kumar and Atul Negi, "A novel approach to eigenpalm features using feature partitioning framework", *In Proceedings of IAPR conference on Machine Vision and Applications*, Japan, pp. 29-32, May 16-18th 2007.

ture Partitioning based PCA (FP-PCA) method. SubXPCA method addressed some of the important feature partitioning issues-(i) Loss of inter-sub-pattern covariances or correlations and (ii) feature order dependency. During the study and motivation of the SubXPCA it was found that, it is not always correct to rely only on *locally* extracted features from feature blocks as done by methods like modPCA [53] and SubPCA [21] (Section 2.2 of Chapter 2). A more sophisticated combination approach with *global features* using inter-block feature correlations is required. It was shown that SubXPCA is relatively more robust against feature orders and less sensitive to overlapping sub-patterns. However, the existing FP-PCA methods including SubXPCA (Section 2.2 of Chapter 2) use classical PCA as their preferred method for local feature extraction. When feature extraction methods like classical PCA are applied on images, we see that classical PCA treats an image pattern of size $m \times n$ as a vector of $m.n$ feature values. In other words, classical PCA does not make use of inherent matrix structure of an image, thus recognition performance and computational performance ($O(N.m^2.n^2)$) of classical PCA may not be encouraging. The same problem lies with existing FP-PCA methods (such as modPCA, EigenRegions method, SubPCA, etc) although better than classical PCA, they do not use matrix structure of images. Please note that matrix structure of image is more appropriate and captures crucial spatial relationships in the image and may improve recognition or classification. Improving the performance on image recognition problems was confirmed through the explicit use of the matrix structure of image data as proposed in the IMPCA (also known as 2DPCA) approach [187][189]. However, IMPCA (2DPCA) approach is not based on a feature-partitioning framework, hence it does not exploit

the strengths of feature-partitioning framework.

Thus, in this chapter, our goal is to apply the feature partitioning framework to PCA computation upon matrix structure of image data. This gives rise to the first of our novel approaches, Sub-IMage based Principal Component Analysis (SIMPCA). A more sophisticated way to combine local features extracted from sub-image blocks (using global inter-block correlations) allows further improvement over SIMPCA and we propose the second novel method, FLexible Image Principal Component Analysis (FLPCA). We prove the computational superiority of our methods by algorithm analysis and confirm practically through comprehensive experimentation on face data sets and palmprint data. The experimentation methodology is the very systematic approach as proposed [135]. In our experimentation we find False Rejection Ratio (FRR), False Acceptance Ratio (FAR) in addition to Total Error Rate (TER). Another important experimentation parameter is the number of sub-images (blocks) used for partitioning, and we analyze the recognition rate stability of the approaches with respect to number of sub-image blocks. This is a unique aspect of our experimentation showing the variation in performance due to partitioning. We categorize the performance of the various approaches studied in this Chapter by plotting their performance in terms of the parameters of computation time and recognition. Our study shows that FLPCA performance is the best according to both of the performance parameters.

The rest of the chapter is organized as follows. In section 5.2 we present formally our proposed approaches upon image data. A detailed study of the time complexity of these approaches is performed in section 5.3. We present the details of experimental

application of these approaches on the standard face databases and palmprint data in section 5.4.

5.2 Feature Partitioning based PCA (FP-PCA) Approaches for Image Data

In this section, we propose the methods for image data, SIMPCA and FLPCA, which take advantage of the appropriate matrix arrangement of images and also improve performance by using feature partitioning framework. SIMPCA extracts features locally from sub-image blocks using 2DPCA (IMPCA) method [189], but performs a simple combination of these local features. FLPCA proposed here improves upon SIMPCA by a more informed combination approach to remove redundant local features using global correlations.

5.2.1 Sub-IMage Principal Component Analysis (SIMPCA)

In this subsection, we formally present the first FP-PCA approach, SIMPCA, exclusively for image data. For a better understanding of the method Fig. 5.1 should be studied. Consider $\mathbf{A} = \{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N\}$, the set of N mean-subtracted images of size $m \times n$, where m, n are the number of rows and columns respectively. Each image, $\mathbf{A}_i$, is treated as a $m \times n$ matrix of image features (pixels).

Step 1: Partitioning of images (Step-1 in Fig. 5.1)

Divide the i^{th} image, $(\mathbf{A}_i)_{m\times n}$, into k, $(2 \leq k \leq \frac{n}{2})$, sub-images, $\{\mathbf{A}_i^j;\, j = 1, 2, \ldots, k\}$, each of size $m \times u$, where $u = \lfloor \frac{n}{k} \rfloor$ and it is clear that $2 \leq u \leq \frac{n}{2}$. If k is not an exact

divisor of n, (i.e. $n \neq k.u$), (i) one option is to truncate last $(n - k.u)$ columns (we used this option for our experimentation), (ii) other option may be to extract features from the last sub-image similar to other sub-images, but with a different sub-image size and so on. We divide each image by taking features of u contiguous columns. The j^{th} sub-image of an image $\mathbf{A}_i$ is given by

$$(\mathbf{A}_i^j)_{m\times u} = \begin{bmatrix} a_{1,(l+1)} & a_{1,(l+2)} & \dots & a_{1,(l+u)} \\ a_{2,(l+1)} & a_{2,(l+2)} & \dots & a_{2,(l+u)} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m,(l+1)} & a_{m,(l+2)} & \dots & a_{m,(l+u)} \end{bmatrix} \tag{5.1}$$

where $l = (j-1).u$ and a_{s_1,s_2} represents an image feature at matrix location (s_1, s_2). Here we used non-overlapping option of sub-images, that is no two sub-images have common features. Although we have shown here to divide an image with respect to columns, n, in principle, we can also divide with respect to rows, m, or with respect to both columns and rows.

Step 2: Grouping of sub-images (Step-2 in Fig. 5.1)

Form $\mathbf{P}^j$ as the set of j^{th} sub-images of images, $\{\mathbf{A}_i,\, i = 1, 2, \dots, N\}$, given by

$$\mathbf{P}^j = \{\mathbf{A}_1^j, \mathbf{A}_2^j, \dots, \mathbf{A}_N^j\} \tag{5.2}$$

Here we use the option of grouping of homogeneous sub-images (sub-patterns) (Defn. 2 of Chapter 3).

Step 3: Local feature extraction from sub-images (Step-3 in Fig. 5.1)

For each sub-image set, $\mathbf{P}^j$, $j \in \{1, 2, \dots, k\}$ perform the following steps (a)-(c).

(a) Compute the local covariance matrix, $(\mathbf{M}^j)_{u\times u}$ for the sub-image set as given by

$$(\mathbf{M}^j)_{u\times u} = \frac{1}{N}.\sum_{i=1}^{N}[\mathbf{A}_i^j]_{u\times m}^T.[\mathbf{A}_i^j]_{m\times u} \tag{5.3}$$

(b) Find $r\,(\leq u)$, eigenvectors of $\mathbf{M}^j$ corresponding to first r largest eigenvalues using eigenvalue decomposition (EVD) as follows.

$$\mathbf{M}^j.\mathbf{e}_p^j = \mathbf{e}_p^j.\lambda_p^j \tag{5.4}$$

where $\mathbf{e}_p^j$ is the p^{th} eigenvector with eigenvalue λ_p^j. Let $(\mathbf{E}^j)_{u\times r}$ be the matrix of r column eigenvectors chosen in this step as given by

$$(\mathbf{E}^j)_{u\times r} = [\mathbf{e}_1^j\ \mathbf{e}_2^j \ldots \mathbf{e}_r^j] = \begin{bmatrix} e_1^j(1) & e_2^j(1) & \ldots & e_r^j(1) \\ e_1^j(2) & e_2^j(2) & \ldots & e_r^j(2) \\ \vdots & \vdots & \ldots & \vdots \\ e_1^j(u) & e_2^j(u) & \ldots & e_r^j(u) \end{bmatrix} \tag{5.5}$$

(c) Project $\mathbf{P}^j = \{\mathbf{A}_1^j, \mathbf{A}_2^j, \ldots, \mathbf{A}_N^j\}$ onto $\mathbf{E}^j$ to get a set of locally-reduced sub-images, $\{\mathbf{B}_i^j;\ i = 1, 2, \ldots, N\}$ and is given by

$$(\mathbf{B}_i^j)_{m\times r} = (\mathbf{A}_i^j)_{m\times u}.(\mathbf{E}^j)_{u\times r} \tag{5.6}$$

$$(\mathbf{B}_i^j)_{m\times r} = \begin{bmatrix} b_{1,(t+1)}^j & b_{1,(t+2)}^j & \ldots & b_{1,(t+r)}^j \\ b_{2,(t+1)}^j & b_{2,(t+2)}^j & \ldots & b_{2,(t+r)}^j \\ \vdots & \vdots & \vdots & \vdots \\ b_{m,(t+1)}^j & b_{m,(t+2)}^j & \ldots & b_{m,(t+r)}^j \end{bmatrix} \tag{5.7}$$

where $t = (j-1).r$ and b_{t_1,t_2}^j is a local feature at matrix location (t_1, t_2).

Step 4: Combining local features of sub-images (Step-4 in Fig. 5.1)

Collate all reduced sub-images corresponding to the image, $\mathbf{A}_i$, to give locally-reduced image matrix, $\mathbf{B}_i$, and is given by

$$(\mathbf{B}_i)_{m\times k.r} = \left[\mathbf{B}_i^1\ \mathbf{B}_i^2 \ldots \mathbf{B}_i^k\right] \tag{5.8}$$

$(\mathbf{B}_i)_{m\times k.r} =$

$$\begin{bmatrix} b^1_{1,1} & b^1_{1,2} & \dots & b^1_{1,r} & b^2_{1,r+1} & b^2_{1,r+2} & \dots & b^2_{1,2.r} & \dots & b^k_{1,(k-1).r+1} & b^k_{1,(k-1).r+2} & \dots & b^k_{1,k.r} \\ b^1_{2,1} & b^1_{2,2} & \dots & b^1_{2,r} & b^2_{2,r+1} & b^2_{2,r+2} & \dots & b^2_{2,2.r} & \dots & b^k_{2,(k-1).r+1} & b^k_{2,(k-1).r+2} & \dots & b^k_{2,k.r} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ b^1_{m,1} & b^1_{m,2} & \dots & b^1_{m,r} & b^2_{m,r+1} & b^2_{m,r+2} & \dots & b^2_{m,2.r} & \dots & b^k_{m,(k-1).r+1} & b^k_{m,(k-1).r+2} & \dots & b^k_{m,k.r} \end{bmatrix} \tag{5.9}$$

Each $(\mathbf{B}_i)_{m\times k.r}$ obtained in *Step-4* is viewed as a $(m.k.r)$-dimensional vector, and is used for subsequent pattern recognition tasks.

In SIMPCA, each image is partitioned into sub-images, then local features are extracted from each sub-image and these local features are used for subsequent tasks. We expect that SIMPCA works better when local structure plays a dominant role and may not perform well when features vary globally (i.e. when global structure is present). To perform well in either case, we propose a flexible method, FLPCA, under the feature partitioning framework in the next sub-section. The flexibility of FLPCA is due to its comprehensive local as well as global capacity for feature extraction.

5.2.2 FLexible Image Principal Component Analysis (FLPCA)

Here, we formally present the second FP-PCA approach, FLPCA. To appreciate and understand our method it is good to study Fig. 5.2.

Step 1, Step 2 and Step 3: Same as SIMPCA method (Steps 1-3 in Fig. 5.1).

Step 4: Combining local features of sub-images (Fig. 5.2):

(A) *Collate locally-reduced sub-images*: Same as *Step-4* of SIMPCA. This step produces locally-reduced images, $\{\mathbf{B}_i;\, i = 1, 2, \dots, N\}$ (*Step-4(A) in Fig. 5.2*).

(B) *Global feature extraction from locally-reduced image matrices,* $\mathbf{B} = \{\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_N\}$ *using inter-block correlations (Step-4(B) in Fig. 5.2):*

(i) Compute image covariance matrix, $(\mathbf{M}^g)_{k.r \times k.r}$ from $\mathbf{B}$ as given by

$$\mathbf{M}^g = \frac{1}{N}.\sum_{i=1}^{N}[\mathbf{B}_i]^T_{k.r \times m}.[\mathbf{B}_i]_{m \times k.r} \tag{5.10}$$

(ii) Compute eigenvectors of $\mathbf{M}^g$ using eigenvalue decomposition (EVD) as given by

$$\mathbf{M}^g.\mathbf{e}_s = \mathbf{e}_s.\lambda_s \tag{5.11}$$

where $\mathbf{e}_s$ is the s^{th} eigenvector with eigenvalue λ_s.

(iii) Select $w\,(\leq k.r)$, eigenvectors of $\mathbf{M}^g$ corresponding to first w largest eigenvalues. Let $(\mathbf{E}^g)_{k.r \times w}$ be the matrix of w column eigenvectors chosen in this step and is given by

$$(\mathbf{E}^g)_{k.r \times w} = [\mathbf{e}_1\ \mathbf{e}_2 \ldots \mathbf{e}_w] = \begin{bmatrix} e_1(1) & e_2(1) & \ldots & e_w(1) \\ e_1(2) & e_2(2) & \ldots & e_w(2) \\ \vdots & \vdots & \ldots & \vdots \\ e_1(k.r) & e_2(k.r) & \ldots & e_w(k.r) \end{bmatrix} \tag{5.12}$$

(iv) Finally project $\mathbf{B} = \{\mathbf{B}_1, \mathbf{B}_2, \ldots, \mathbf{B}_N\}$ onto $\mathbf{E}^g$ to get a set of reduced image matrices, $\mathbf{D} = \{\mathbf{D}_1, \mathbf{D}_2, \ldots, \mathbf{D}_N\}$. $\mathbf{D}_i$ is the reduced image matrix corresponding to the original image, $(\mathbf{A}_i)_{m \times n}$ and is given by

$$(\mathbf{D}_i)_{m \times w} = (\mathbf{B}_i)_{m \times k.r}.(\mathbf{E}^g)_{k.r \times w} \tag{5.13}$$

Each $(\mathbf{D}_i)_{m \times w}$ thus obtained is viewed as a $(m.w)$-dimensional vector, and is used for subsequent pattern recognition tasks.

In the above step, features are extracted based on global variation among the extracted local features (i.e. using inter-block dependencies or correlations as discussed

in section 3.3.8 of Chapter 3), which may further aid in dimensionality reduction and may increase recognition rate as well.

Theorem 5 *IMPCA (2DPCA) is a special case of FLPCA.*

Proof 5 *IMPCA finds correlations between every pair of n column-features. Here in steps 3(b)-3(c) of FLPCA we need to set $r = u$ (i.e. all features of every sub-image are chosen and $u = \frac{n}{k}$). Therefore in Step-4, FLPCA finds correlations between $m \times (k.r) = m \times (k.u) = m \times (k.(\frac{n}{k})) = m \times n$ features, that is FLPCA finds correlations between all column-features. Hence the theorem is proved.*

In the next section, we discuss the time complexities of the various variations of PCA and consequently of the proposed FP-PCA approaches, SIMPCA and FLPCA.

5.3 Time Complexity Analysis

Here we focus mainly on time complexity of computation of covariance matrix since it is the most dominant factor, computationally, in PCA variations. Consider $\mathbf{A} = \{\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_N\}$, the set of N images of size $m \times n$. Each image, $\mathbf{A}_i$, is treated as a $m \times n$ matrix of image features (pixels).

Time complexities of calculating covariance matrix(ces), T_C, T_M, T_o, etc, by various PCA methods are shown in Table 5.1.

Theorem 6 *$T_M = \frac{1}{k.m}.T_C$, where $2 \leq k \leq \frac{n}{2}$ is the number of sub-images per image and m, n are the number of rows and the number of columns of an image matrix respectively.*

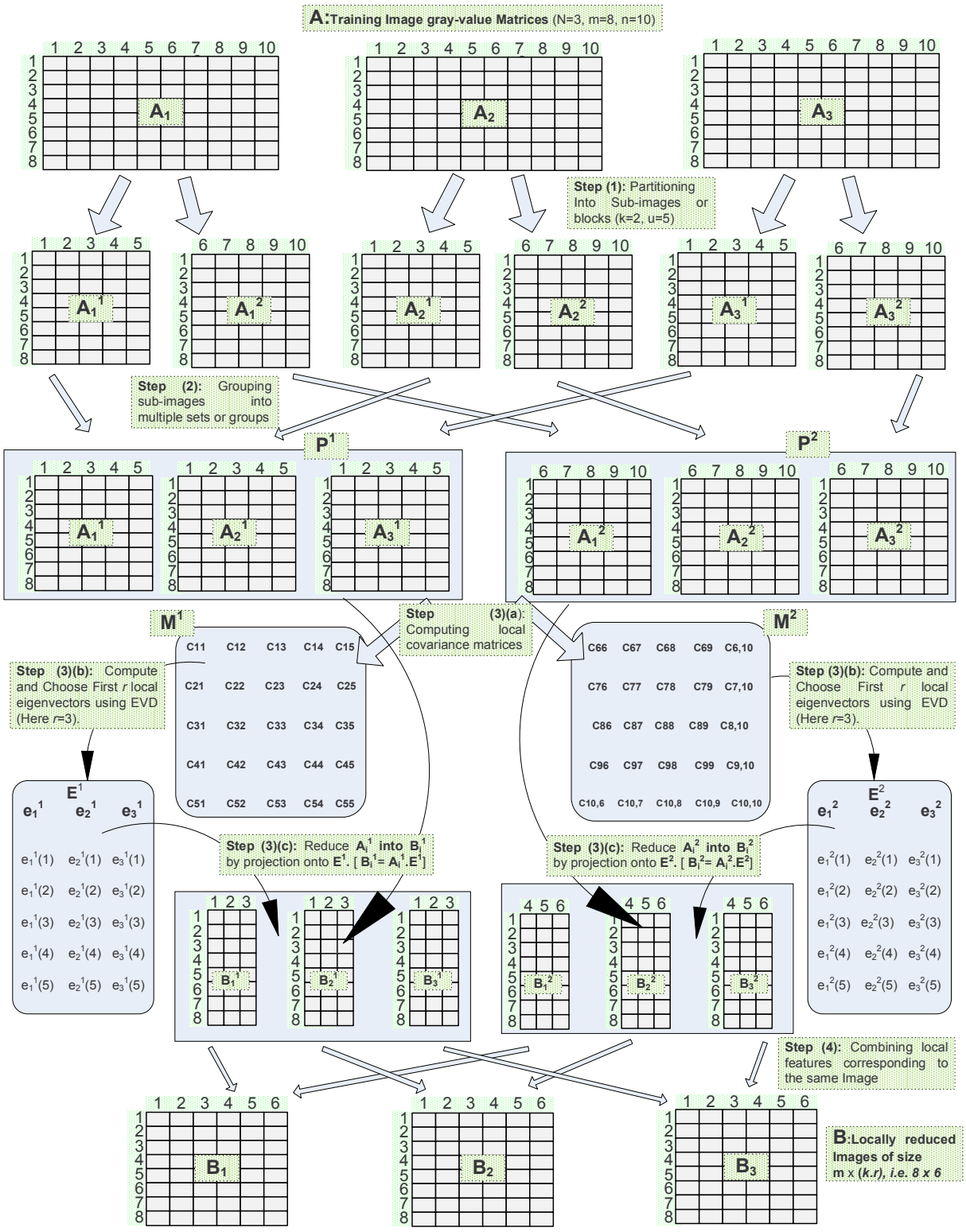


Figure 5.1: Visualizing SIMPCA method

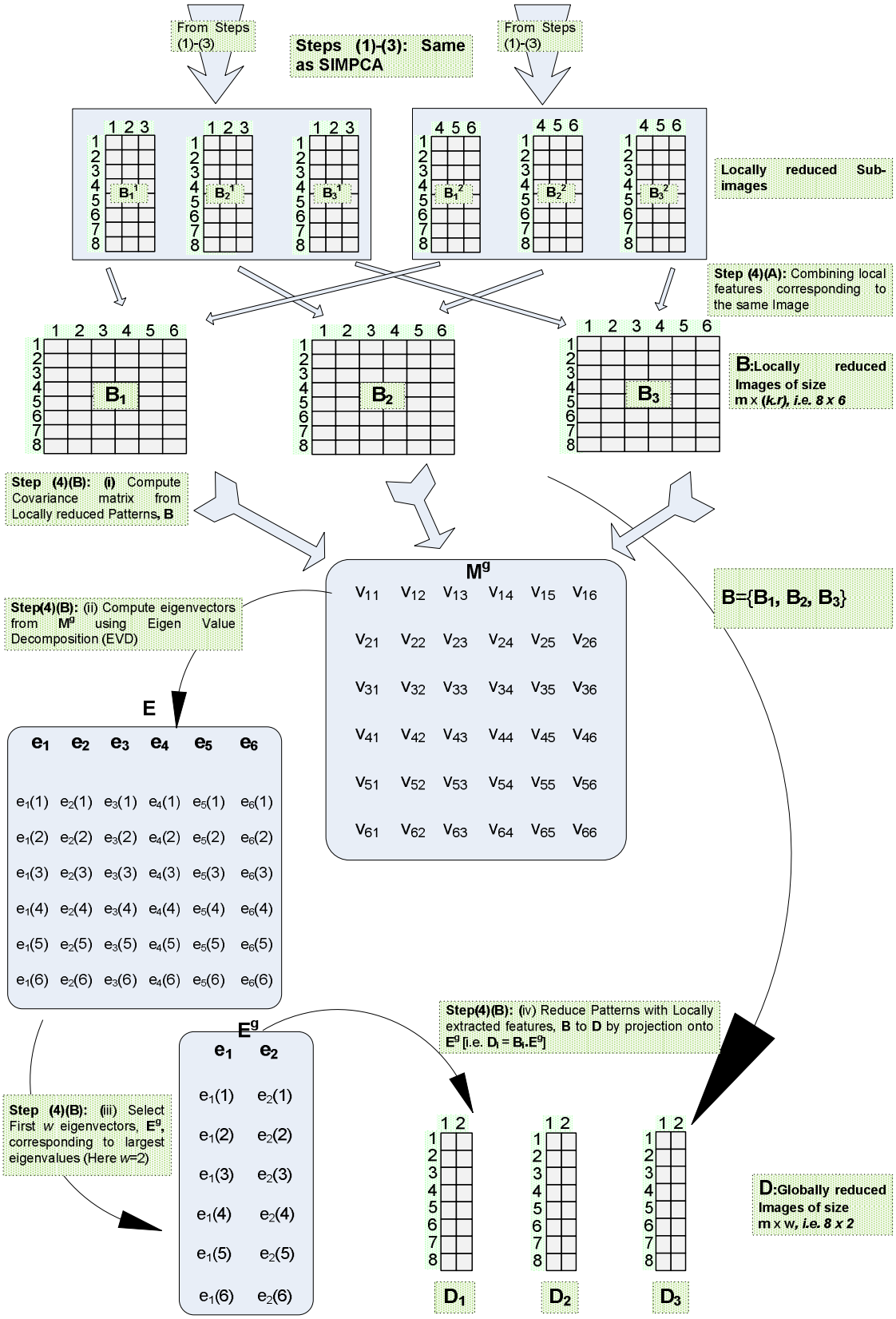


Figure 5.2: Visualizing FLPCA method

Table 5.1: Time complexities of various PCA methods

Method	Time Complexity	Parameter Description
Efficient Classical PCA	$T_E = O(N^2.m.n)$	–
Classical PCA	$T_C = O(N.m^2.n^2)$	–
modPCA	$T_o = O(k.N.u_1^2.u_2^2)$	$u_1 \times u_2$: sub-image size $u_1 = \lfloor \frac{m}{k_1} \rfloor$; $u_2 = \lfloor \frac{n}{k_2} \rfloor$; $k = k_1.k_2$: no. of sub-images
IMPCA (2DPCA)	$T_I = O(N.m.n^2)$	–
SIMPCA	$T_M = O(k.N.m.u^2)$	–
FLPCA	$T_L = O(k.N.m.u^2 + N.m.(k.r)^2)$	$O(k.N.m.u^2)$: to compute k sub-image cov. matrices, $O(N.m.(k.r)^2)$: to compute image cov. matrix, $\mathbf{M}^g$

Proof 6 *From eq. T_M of Table 5.1, $T_M = k.N.m.u^2$*

$\Rightarrow T_M = \frac{1}{k}.N.m.n^2$ *(because $u = \frac{n}{k}$)*

$\Rightarrow T_M = \frac{1}{k}.\frac{1}{m}.N.m^2.n^2$

$\Rightarrow T_M = \frac{1}{k.m}.T_C$ *(from eq. T_C of Table 5.1)*

Hence the theorem follows.

Theorem 7 *$T_L < T_C$, where $2 \leq k \leq \frac{n}{2}$ is the number of sub-images per image, u is the number of columns in sub-image and m, n are the number of rows and the number of columns of an image matrix respectively.*

Proof 7 *From eq. T_L of Table 5.1, $T_L = O(k.N.m.u^2 + N.m.(k.r)^2)$*

$\Rightarrow T_L = T_M + N.m.\frac{n^2}{u^2}.r^2$ *(because $k = \frac{n}{u}$ and from eq. T_M of Table 5.1)*

$\Rightarrow T_L = \frac{1}{k.m}.T_C + \frac{r^2}{u^2}.\frac{1}{m}.N.m^2.n^2$ *(from Theorem 6, $T_M = \frac{1}{k.m}.T_C$)*

$\Rightarrow T_L = \frac{1}{k.m}.T_C + \frac{r^2}{u^2}.\frac{1}{m}.T_C$ *(from eq. T_C of Table 5.1)*

$\Rightarrow T_L = T_C.(\frac{1}{k.m} + \frac{r^2}{u^2}.\frac{1}{m})$

We know that $(\frac{1}{k.m} + \frac{r^2}{u^2}.\frac{1}{m}) < 1$, *because* $k \geq 2, m \geq 2$ *and* $r \leq u$.

$\Rightarrow T_L < T_C$

Hence the theorem follows.

Theorem 8 $T_M < T_E$, $\forall u < N$, *where* u *is the number of columns in a sub-image and* N *is the number of training image patterns.*

Proof 8 *From eq.* T_M *of Table 5.1,* $T_M = k.N.m.u^2$

$\Rightarrow T_M = \frac{1}{k}.N.m.n^2$ *(because* $u = \frac{n}{k}$*)*

$\Rightarrow T_M = N.\frac{n}{k}.m.n$

$\Rightarrow T_M < N.N.m.n$ *if* $\frac{n}{k} < N$

$\Rightarrow T_M < T_E$, $\forall u < N$. *(because* $u = \frac{n}{k}$ *and from eq.* T_E *of Table 5.1)*

Hence the theorem follows.

Theorem 9 $T_L < T_E$, $\forall r < \sqrt{(N-u).\frac{u}{k}}$ *or* $\forall k < (N-u).\frac{u}{r^2}$ *and* $u < N$, *where* u *is the number of columns in a sub-image;* N *is the number of training image patterns;* $2 \leq k \leq \frac{n}{2}$ *is the number of sub-images per image;* m, n *are the number of rows and the number of columns of an image matrix respectively;* r *is the number of projection (eigen)vectors per sub-image set.*

Proof 9 *From eq.* T_L *of Table 5.1,* $T_L = O(k.N.m.u^2 + N.m.(k.r)^2)$

$\Rightarrow T_L = O(\frac{1}{k}.N.m.n^2 + N.m.k.\frac{n}{u}.r^2)$ *(because* $k = \frac{n}{u}$*)*

$\Rightarrow T_L = N.m.n.(\frac{n}{k} + \frac{k}{u}.r^2)$

$\Rightarrow T_L < N.N.m.n$ *if* $(u + \frac{k}{u}.r^2) < N$ *(because* $u = \frac{n}{k}$*)*

$\Rightarrow T_L < N.N.m.n$ *if* $r < \sqrt{(N-u).\frac{u}{k}}$

$\Rightarrow T_L < T_E$, $\forall r < \sqrt{(N-u).\frac{u}{k}}$. *(from eq.* T_E *of Table 5.1)*

$\Rightarrow T_L < T_E,\ \forall k < (N-u).\frac{u}{r^2}$

It is clear that $u < N$, otherwise we get an invalid result.

Hence the theorem follows.

Theorem 10 *$T_M = \frac{1}{k}.T_I$, where $2 \leq k \leq \frac{n}{2}$ is the number of sub-images per image; r is the number of projection (eigen)vectors per sub-image set; $m \times u$ is the sub-image size.*

Proof 10 *From eq. T_M of Table 5.1,*

$T_M = O(k.N.m.u^2)$

$\Rightarrow T_M = k.N.m.\frac{n^2}{k^2}$

$\Rightarrow T_M = \frac{1}{k}.N.m.n^2$

$\Rightarrow T_M = \frac{1}{k}.T_I$ *(from eq. T_I of Table 5.1)*

Hence the theorem follows.

Theorem 11 *$T_L < T_I,\ \forall r < u.\sqrt{\frac{k-1}{k}}$, where $2 \leq k \leq \frac{n}{2}$, is the number of sub-images per image; r is the number of projection (eigen)vectors per sub-image set; $m \times u$ is the sub-image size.*

Proof 11 *From eq. T_L of Table 5.1, $T_L = O(k.N.m.u^2 + N.m.(k.r)^2)$*

$\Rightarrow T_L = \frac{1}{k}.N.m.n^2 + \frac{n^2}{n^2}.N.m.k^2.r^2$

$\Rightarrow T_L = \frac{1}{k}.N.m.n^2 + \frac{k^2}{n^2}.\left[N.m.n^2\right].r^2$

$\Rightarrow T_L = \frac{1}{k}.N.m.n^2 + \frac{r^2}{u^2}.\left[N.m.n^2\right]$ *(because $u = \frac{n}{k}$)*

$\Rightarrow T_L = (\frac{1}{k} + \frac{r^2}{u^2}).\left[N.m.n^2\right]$

$\Rightarrow T_L < N.m.n^2$ *if* $(\frac{1}{k} + \frac{r^2}{u^2}) < 1$

$\Rightarrow T_L < N.m.n^2$ *if* $\frac{r^2}{u^2} < (1 - \frac{1}{k})$

$\Rightarrow T_L < T_I$, $\forall r < u.\sqrt{\frac{(k-1)}{k}}$ *(from eq. T_I of Table 5.1)*

Hence the theorem follows.

Theorem 12 $\lim_{r\to 1,k\to 2}[T_L \approx T_M]$, *where* $2 \leq k \leq \frac{n}{2}$ *is the number of sub-images;* $1 \leq r \leq u$ *is the number of eigenvectors chosen from each sub-image set;* $m \times u$ *is the size of sub-image.*

Proof 12 *Consider the second term of eq. T_L of Table 5.1 that is $O(N.m.(k.r)^2)$*

$O(N.m.(k.r)^2)$ is minimum if $(k.r)^2$ is minimum.

$\Rightarrow O(N.m.(k.r)^2)$ is minimum if (i) k and (ii) r are minimum.

$\Rightarrow O(N.m.(k.r)^2)$ is minimum if (i) $k = 2$ and (ii) $r = 1$ (because $2 \leq k \leq \frac{n}{2}$ and $1 \leq r \leq u$).

Thus $O(N.m.(k.r)^2)$ becomes insignificant for smaller values of k and r. It is to be noted that First term of T_L is equal to T_M.

Hence the theorem follows from eqs. T_M and T_L of Table 5.1.

Theorem 13 $\lim_{r\to 1,k\to 2}[T_L \approx \frac{1}{k}.T_I]$, *where* $2 \leq k \leq \frac{n}{2}$ *is the number of sub-images;* $1 \leq r \leq u$ *is the number of eigenvectors chosen from each sub-image set;* $m \times u$ *is the size of sub-image.*

Proof 13 *From Theorem 10, $T_M = \frac{1}{k}.T_I$.*

From Theorem 12, $T_L \approx T_M$ as $k \to 2$ and $r \to 1$.

Therefore, $T_L \approx \frac{1}{k}.T_I$ as $k \to 2$ and $r \to 1$.

Hence the theorem follows.

Theorem 14 $T_M = \frac{1}{u_1}.T_o$. *Here we assume that both SIMPCA and modular PCA divide each image into sub-images of size* $u_1 \times u_2$ *in the same way, where* $2 \leq u_1 \leq \frac{m}{2}$, *and* $2 \leq u_2 \leq \frac{n}{2}$, *the number of rows and columns of a sub-image respectively.*

Proof 14 *From eq.* T_M *of Table 5.1,* $T_M = O(k.N.m.u^2)$, *for sub-images of size* $m \times u$.

$\Rightarrow T_M = O(k.N.u_1.u_2^2)$, *for sub-images of size* $u_1 \times u_2$.

$\Rightarrow T_M = (\frac{1}{u_1}).k.N.u_1^2.u_2^2$

$\Rightarrow T_M = \frac{1}{u_1}.T_o$ *(from eq.* T_o *of Table 5.1).*

Hence the theorem follows.

Theorem 15 $T_L < T_o$, *if* $k_2 < (u_1 - 1)$. *Here we assume that both FLPCA and modular PCA divide each image into sub-images of size* $u_1 \times u_2$ *in the same way, where* $2 \leq u_1 \leq \frac{m}{2}$, *and* $2 \leq u_2 \leq \frac{n}{2}$, *the number of rows and columns of a sub-image;* $1 \leq r < u_2$ *is the number of eigenvectors chosen from each sub-image set;* $k = k_1.k_2$, *the number of sub-images of an image.*

Proof 15 *From eq.* T_L *of Table 5.1,* $T_L = O(k.N.m.u^2 + N.m.(k.r)^2)$ *for sub-images of size,* $m \times u$.

$\Rightarrow T_L = O(k.N.u_1.u_2^2 + N.m.(k_2.r)^2)$ *for sub-images of size,* $u_1 \times u_2$.

$\Rightarrow T_L = O(k.N.u_1.u_2^2 + N.k_1.u_1.(k_2.r)^2)$ *(because* $u_1 = \frac{m}{k_1}$, $u_2 = \frac{n}{k_2}$ *from Table 5.1)*

$\Rightarrow T_L = \frac{1}{u_1}.T_o + N.(k_1.k_2).u_1.k_2.r^2$ *(from eq.* T_o *of Table 5.1)*

$\Rightarrow T_L = \frac{1}{u_1}.T_o + N.k.u_1.k_2.r^2$ *(because* $k = k_1.k_2$ *from Table 5.1)*

$\Rightarrow T_L = \frac{1}{u_1}.T_o + [\frac{(u_1-1)}{u_1}.\frac{u_1}{(u_1-1)}].N.k.u_1.k_2.r^2$

$\Rightarrow T_L = \frac{1}{u_1}.T_o + \frac{(u_1-1)}{u_1}.N.k.u_1^2.\frac{k_2}{u_1-1}.r^2$

$\Rightarrow T_L = \frac{1}{u_1}.T_o + \frac{(u_1-1)}{u_1}.N.k.u_1^2.u_2^2.[\frac{k_2}{u_1-1}.\frac{r^2}{u_2^2}]$

$\Rightarrow T_L = \frac{1}{u_1}.T_o + \frac{(u_1-1)}{u_1}.T_o.[\frac{k_2}{u_1-1}.\frac{r^2}{u_2^2}]$ *(from eq. T_o from Table 5.1)*

$\Rightarrow T_L < T_o$ *if* $[\frac{k_2}{u_1-1}.\frac{r^2}{u_2^2}] < 1$

It is given in the statement that $r < u_2$*, which implies* $\frac{r^2}{u_2^2} < 1$.

$\Rightarrow T_L < T_o$ *if* $\frac{k_2}{u_1-1} < 1$

$\Rightarrow T_L < T_o$ *if* $k_2 < (u_1 - 1)$

Hence the theorem follows.

From the above theorems we have proved that FP-PCA approaches (SIMPCA and FLPCA) show superior time complexities over Classical PCA, Efficient classical PCA, modular PCA and IMPCA techniques and the same is demonstrated by our experimentation in the next section. From theorems 6, 7 and 12 it is to be noted that both SIMPCA and FLPCA can be ideally $k.m$ times faster than classical PCA. It is also evident from theorems 10-13 that FP-PCA approaches, SIMPCA and FLPCA, can be ideally nearly k times faster than IMPCA. Similarly, from theorems 12, 14 and 15 it is clear that both SIMPCA and FLPCA can be ideally m times faster than modular PCA, where k is the number of sub-images per image and m is the number of rows in an image. It is to be noted that we considered only computation of covariance matrix(ces) [because of its high time complexity as compared to other tasks] to arrive at time complexity of PCA variations considered. In fact, the total time complexity also includes finding eigenvectors, eigenvalues, matrix multiplications, etc,. Therefore, in practice, theoretical and actual time complexities may not match. However, theoretical complexities discussed above give a rough comparison of various PCA methods.

5.4 Experimental Results and Analysis

In this section, we report our experimental results based upon a benchmarking approach [135]. We used test data sets of impostor and clients from different subjects, and the subjects used for testing are not used for training. We explain the experiments conducted using our implementation and compare the results of PCA, IMPCA (2DPCA), modPCA (i.e. an existing FP-PCA approach), SIMPCA and FLPCA. We considered 3 face data sets [119] [166] [184] and PolyU palmprint data [126] for our experiments and we summarize the results in the following subsections.

In our experimentation we observe the performance of these approaches across the number of sub-images (blocks). We want to see the possible improvements due to partitioning and compare the performance of the FP-PCA approaches.

5.4.1 Data Sets

(i) ORL face data set [119] contains face images of 40 persons (subjects), each subject contains 10 images and there are 400 images in total. Each image is of dimension, 112×92 pixels (PGM format). Some images were taken at different times, with variation in lighting, facial expressions (open/closed eyes, smiling/not smiling) and with/without glasses. We selected all 200 images of first 20 subjects for training (i.e. to find principal components), next 13 subjects are taken as client data (legal users). From each client subject, we take first 5 images as templates (total enrollment of 65) and rest of them (65 images) for client testing. The last 7 subjects (total of 70 images) are taken as impostors (illegal users).

(ii) Yale face data [184] contains 165 gray scale GIF images of 15 individuals. There

are 11 images per subject indicating different expression or illumination conditions. We convert all the images into PGM format, of dimension 243×320 pixels. We selected all 77 images of first 7 subjects for training (i.e. to find principal components), next 5 subjects are taken as client data (legal users). From each client subject, we take first 5 images as templates (total enrollment of 25) and rest of them (30 images) for client testing. The last 3 subjects (total of 33 images) are taken as impostors (illegal users).

(iii) UMIST face data set [166] contains images of 20 individuals and a total of 565 images. Each covering a range of poses from profile to frontal views. Subjects cover race, sex, appearance. Each image is of dimension, 112×92 pixels (PGM format). We selected all 255 images of first 10 subjects for training (i.e. to find principal components), next 8 subjects are taken as client data (legal users). From each client subject, we take first 16 images as templates (total enrollment of 128) and rest of them (100 images) for client testing. The last 2 subjects (total of 82 images) are taken as impostors (illegal users).

(iv) PolyU palmprint data. We chose 498 images from first 25 subjects of PolyU palmprint database [126]. The data set contains around twenty samples from each of these subjects collected in two sessions, separated by a collection time interval of two months. The palmprint images of BMP format were converted to PGM format, (284×384 pixels) and were used in our experiments.

5.4.2 Experimental Setup

We have chosen training, clients (for enrollment and testing) and impostor data sets from different subjects without overlapping as described in [135]. First, we find eigenvectors and eigenvalues using training data, then client and impostor data sets are projected on selected eigenvectors to get the data in reduced form. For each reduced client testing and impostor testing data we follow the steps: (i) Find similarity of test data to every client template (i.e. enrolled client) by using Euclidean distance measure, (ii) Next, find the maximum among similarity values found in the previous step, (iii) Accept testing data, if its maximum similarity found in step (ii) is greater than some threshold, $\delta \in (0, 1)$, otherwise reject it. False Rejection Ratio (FRR), False Acceptance Ratio (FAR), Total Error Rate (TER) and Recognition Rates are calculated using the formulae: FAR = Number of Impostor data accepted/ Number of testing data or attempts ; FRR = Number of client data rejected/Number of testing data or attempts; TER = FAR + FRR; Recognition Rate = 100-TER.

We conducted experiments by varying the number of sub-images per image (k). The number of sub-images are varied for different data sets as given: (i) ORL face data: For SIMPCA/FLPCA: $k = 2, 3, 4, 5, 7, 10$; For modPCA: $k_1 \times k_2 : 2 \times 2, 3 \times 3, 4 \times 4, 5 \times 5, 6 \times 6$, (ii) Yale face data: For SIMPCA/FLPCA: $k = 2, 4, 5, 8, 10, 16, 20$; For modPCA: $k_1 \times k_2 : 2 \times 2, 3 \times 3, 4 \times 4, 8 \times 8, 16 \times 16, 32 \times 32$, (iii) UMIST face data: For SIMPCA/FLPCA: $k = 2, 3, 4, 5, 7, 10$; For modPCA: $k_1 \times k_2 : 2 \times 2, 3 \times 3, 4 \times 4, 5 \times 5, 6 \times 6$, (iv) PolyU Palmprint data: For SIMPCA/FLPCA: $k = 4, 8, 9, 16, 25, 32$; For modPCA: $k_1 \times k_2 : 2 \times 2, 3 \times 3, 4 \times 4, 5 \times 5, 10 \times 10, 16 \times 16, 32 \times 32$. For each k value, we find maximum recognition rate by varying number of projection vectors.

Maximum recognition rate and execution times thus obtained are plotted as shown in Figs. 5.3 to 5.10. Overall maximum recognition rates and corresponding FAR, FRR, TER and execution times are shown in Tables 5.2 to 5.5. A novel plot (Fig. 5.11) is created between recognition rate and execution time based on the values shown in Tables 5.2 to 5.4 and the recognition rate is normalized with respect to maximum value for each data set. Similarly, execution time is also normalized.

The C language implementation of FLPCA is also used to compute results of IMPCA (k is set to 1 and Step-4 is omitted) and SIMPCA (Step-4(B) is omitted). Classical PCA is implemented in C language using the efficient procedure [103]. We have not used original implementation of classical PCA because it used to take enormous amount of time for completion. We used a Pentium 4 based system with a CPU clock speed of 2.4 GHz, 256MB RAM and Fedora Core 5 Linux running on it for face data sets. For PolyU palmprint data, we used Pentium D based system with a CPU clock speed of 3.4 GHz, 4GB RAM and openSUSE Linux running on it.

5.4.3 Discussion of Results

The experimental results shown in Tables 5.2 to 5.5 reveal that the proposed FP-PCA approaches (SIMPCA and FLPCA) perform better than the other variations of PCA (including an existing FP-PCA approach, modPCA) based on overall performance criteria of recognition and computational time. For Yale face data, the proposed FP-PCA approaches show much better recognition rates (90%) than PCA (67%). Global structure among different kinds of ORL faces helps PCA to show reasonably good recognition rates (84%), but local structure plays a dominant role which

makes FP-PCA approaches (SIMPCA and FLPCA) to improve their recognition rates enormously (90% and 91% respectively) over PCA. FLPCA shows the highest recognition rate as compared to all the methods. UMIST face data contains images ranging from side to frontal views of images, hence full face is not captured in many images. Hence we expect all the methods to show relatively lower recognition rates, in comparison to other two data sets. In this case the FP-PCA approaches (SIMPCA and FLPCA) show relatively good recognition rate (75%), IMPCA (2DPCA) shows 72%, modPCA shows 73.6% and PCA shows 64%. The proposed FP-PCA approaches showcase their superior time complexities over other methods for all the data sets (Figs. 5.4, 5.6, 5.8 and 5.10). We understand that the original implementation of classical PCA shows huge computational complexity for high dimensional data, hence we use the effiecient implementation of classical PCA [103] for our experimentation. The efficient implemention of PCA shows higher or competitive computational complexity as compared to our proposed FP-PCA approaches.

It is also observed that IMPCA shows better recognition rates for YALE and ORL data sets (90%) in comparison to classical PCA. Similar results are obtained for palmprint data. Modular PCA, although competitive to SIMPCA and FLPCA in terms of maximum recognition rates, but is not consistent in performance across the different number of sub-images as compared with FLPCA and SIMPCA. That is an important result which implies that, to get the best performance of modular PCA, a large amount of experimentation with the k value is required.

To see the effect of local structure on recognition rate and execution time, we took results with varied number of sub-images (k) and the results are plotted in Figs. 5.3

to 5.10. FLPCA and SIMPCA show marginal variation in recognition rate and total error rate (except for UMIST face data) with different number of sub-images on face and palmprint data sets and modPCA shows drastic variation in recognition rate and total error rate with varying number of sub-images. We observed that FLPCA is less sensitive to number of sub-images as compared to modPCA and SIMPCA (Figs. 5.3, 5.5, 5.7 and 5.9).

A novel plot between Execution time and Recognition rate is shown in Fig. 5.11 to group methods based on overall performance. Such a novel plot allows one to conclude as to the method that gives high recognition rate at less execution time. It is clear from the Fig. 5.11 that both the FP-PCA approaches (SIMPCA and FLPCA) form a cluster (top left corner of the figure) of superior overall performance, i.e. superior recognition rates at less execution time as compared to all the other methods.

We observed that FLPCA is flexible to capture the best recognition rates of IMPCA (2DPCA) and SIMPCA. Experimental results confirm the exceptional superiority of both FP-PCA approaches (FLPCA and SIMPCA) as compared to PCA, modPCA and IMPCA in terms of overall performance of recognition and execution time. Interestingly, FLPCA is able to adapt to different scenarios by taking both local and global variations into account. FLPCA does so by exploiting inter-block correlations, to remove redundancy (noise), among local features. In contrast to FLPCA, other two FP-PCA methods (modPCA and SIMPCA) extract only local features and do not exploit inter-block correlations.

5.5 Summary

In this chapter, we have introduced two novel FP-PCA approaches, SIMPCA and FLPCA, which use matrix data arrangement and are more appropriate for image pattern recognition. The approaches use feature partitioning framework providing local feature extraction, superior recognition and lower computational time. Theoretical properties related to the time complexity of the proposed approaches are proved. Experimental results reveal that SIMPCA and FLPCA perform better than IMPCA (2DPCA), modPCA (i.e. an existing FP-PCA approach) and PCA in terms of overall recognition and execution times. Both SIMPCA and FLPCA show better recognition and time complexities by taking local structure into account. FLPCA extracts local features and also combines locally-extracted features globally, to adapt to different image data sets. Our analysis shows that IMPCA (2DPCA) is a special case of FLPCA. FLPCA method is relatively less sensitive to the partitioning effects as compared to the other FP-PCA approaches (SIMPCA and modPCA). The applicability of SIMPCA and FLPCA techniques is demonstrated upon 3 standard face image data sets and palmprint data. The approaches are general and they can be extended to any image data set as well. SIMPCA and FLPCA may be extensively used in face recognition, palmprint recognition, data mining applications to image data and real time image recognition applications.

It is well known that IMPCA (2DPCA) method reduces small sample size (SSS) problem by computing $n \times n$ compact covariance matrix instead of computing $m.n \times m.n$ matrix. Both SIMPCA and FLPCA methods reduce the SSS problem much better by computing more compact $u \times u$ covariance matrices ($u < n$) because they

Table 5.2: Comparison of maximum recognition rates of proposed FP-PCA approaches over other PCA methods for Yale face data

Method	Reco. Rate	FAR	FRR	TER	Time (secs.)	k	u	PVs	Total PVs used
PCA	66.67	30.16	3.17	33.33	175	1	–	–	85
modPCA	90.48	0.0	9.52	9.52	24	1024	7×10	5	5120
IMPCA (2DPCA)	90.48	3.17	6.35	9.52	219	1	–	–	14
SIMPCA	90.48	3.17	6.35	9.52	45	4	80	4	16
FLPCA	90.48	3.17	6.35	9.52	23	8	40	3	12

k: Sub-images per image; For modPCA, u is sub-image size and for other methods, $m \times u$.
PVs: Number of Projection Vectors (eigenvectors chosen for projection) per sub-image set

extract image features by dividing each image of size $m \times n$ into sub-images of size $m \times u$.

In the next chapter, we study theoretical properties of FP-PCA methods including SubXPCA, SIMPCA and FLPCA with respect to summarization of variance.

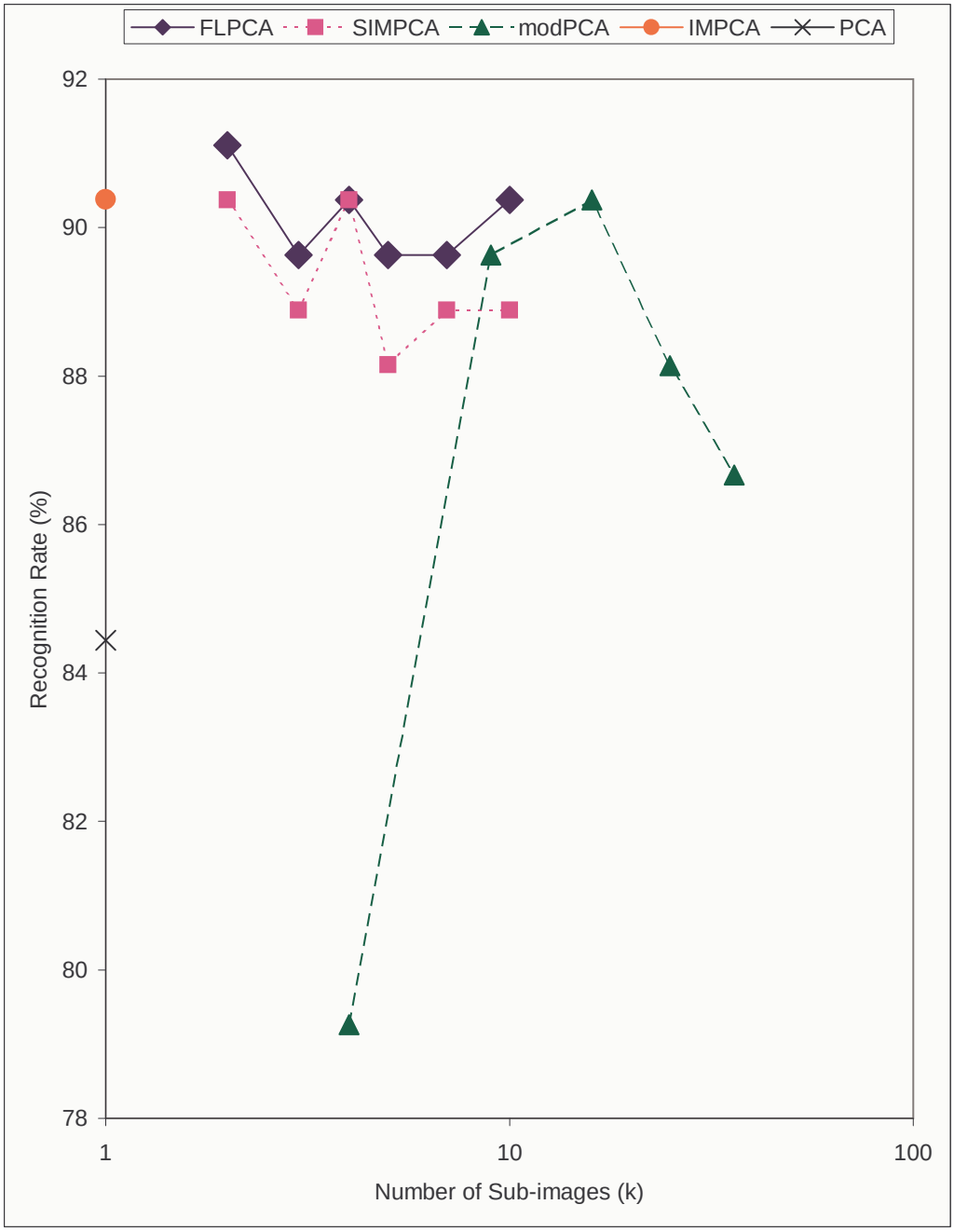


Figure 5.3: *Comparison of recognition rate for ORL face data.* FLPCA shows more consistent performance across the number of sub-images as compared to modPCA and SIMPCA. FLPCA shows highest recognition rate of all the methods. FLPCA and SIMPCA also outperform PCA.

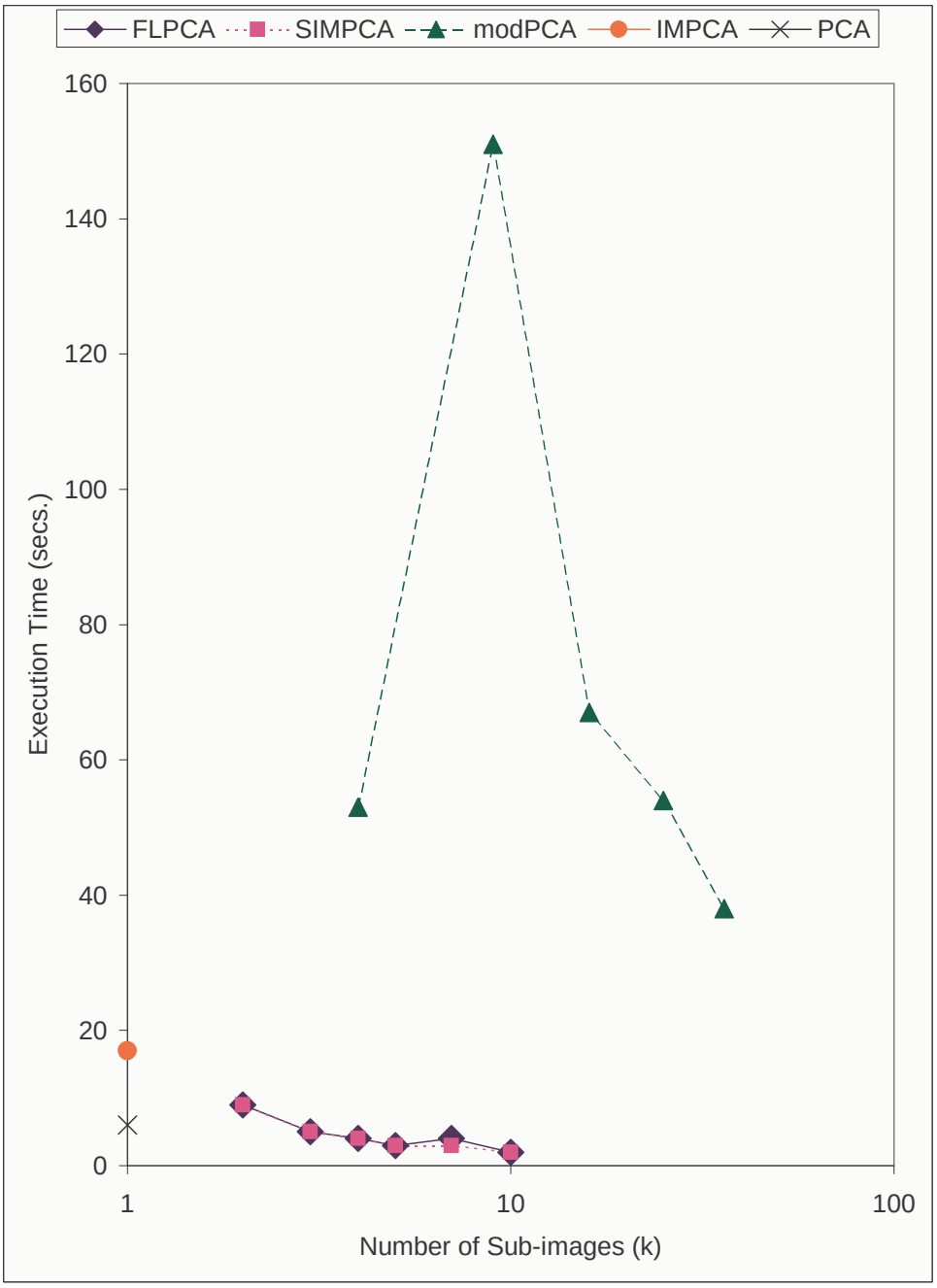


Figure 5.4: *Comparison of computational time for ORL face data.* FLPCA and SIMPCA show better efficiency across the number of sub-images as compared to modPCA. FLPCA and SIMPCA also show less computational time as compared to IMPCA. PCA shows competitive complexity to SIMPCA and FLPCA because we used the efficient implementation [103] instead of the original implementation of PCA.

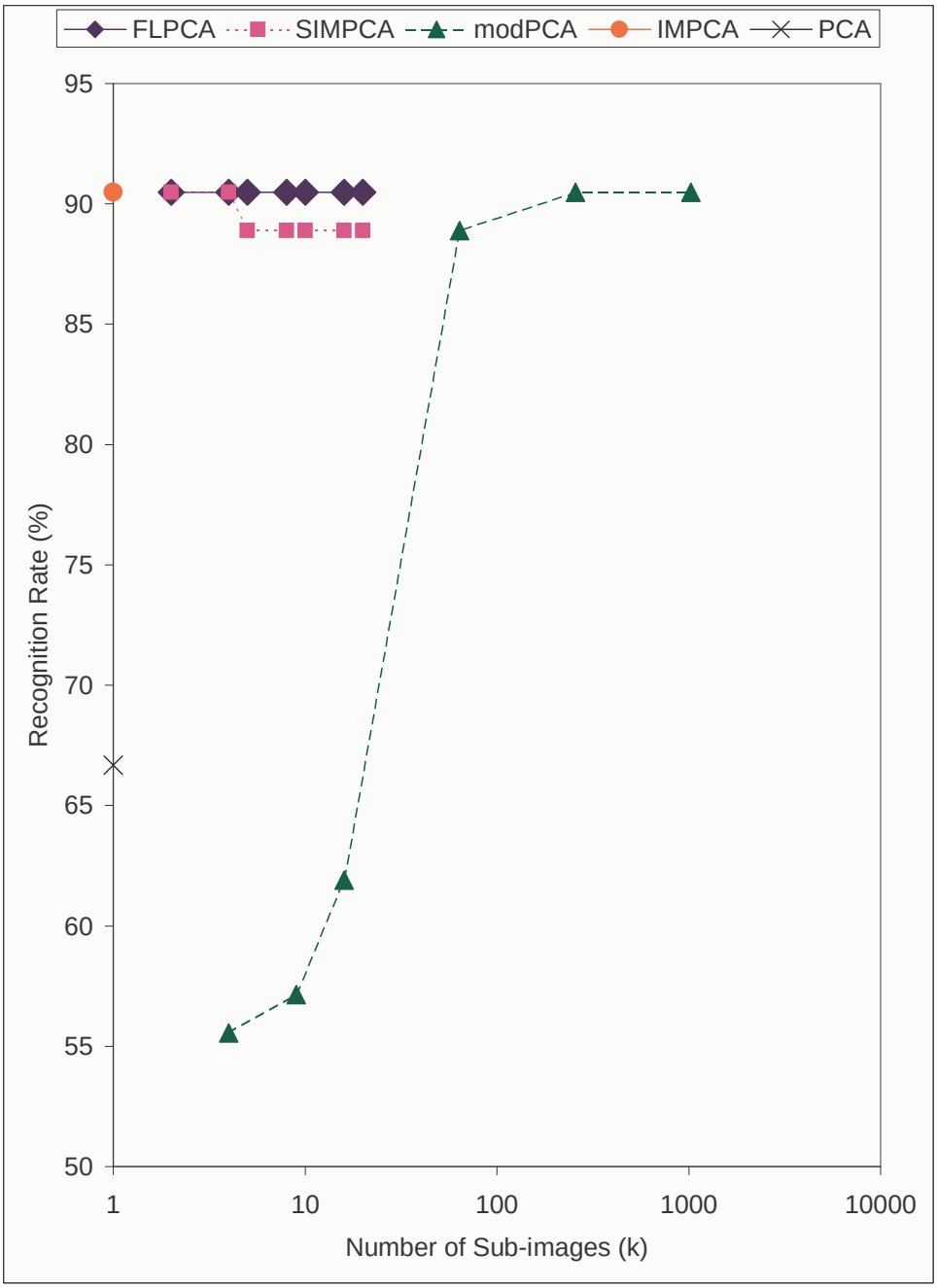


Figure 5.5: *Comparison of recognition rate for Yale face data.* FLPCA shows consistently good performance irrespective of number of sub-images. SIMPCA shows more consistency as compared to modPCA. FLPCA and SIMPCA also outperform PCA.

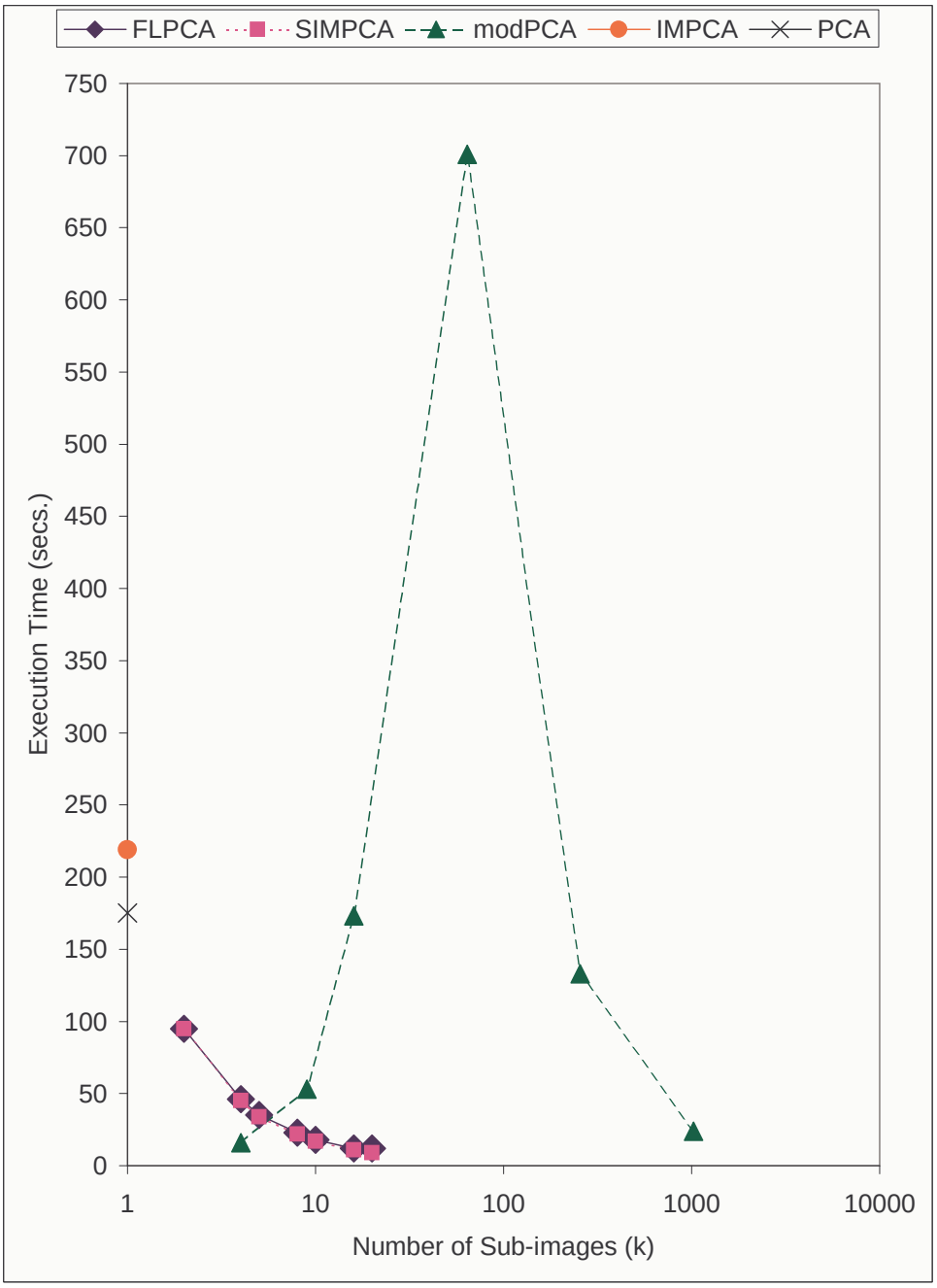


Figure 5.6: *Comparison of computational time for Yale face data.* FLPCA and SIMPCA show better efficiency across various number of sub-images as compared to modPCA. FLPCA and SIMPCA show better computational time as compared to IMPCA (2DPCA) and the efficient implementation of PCA[103].

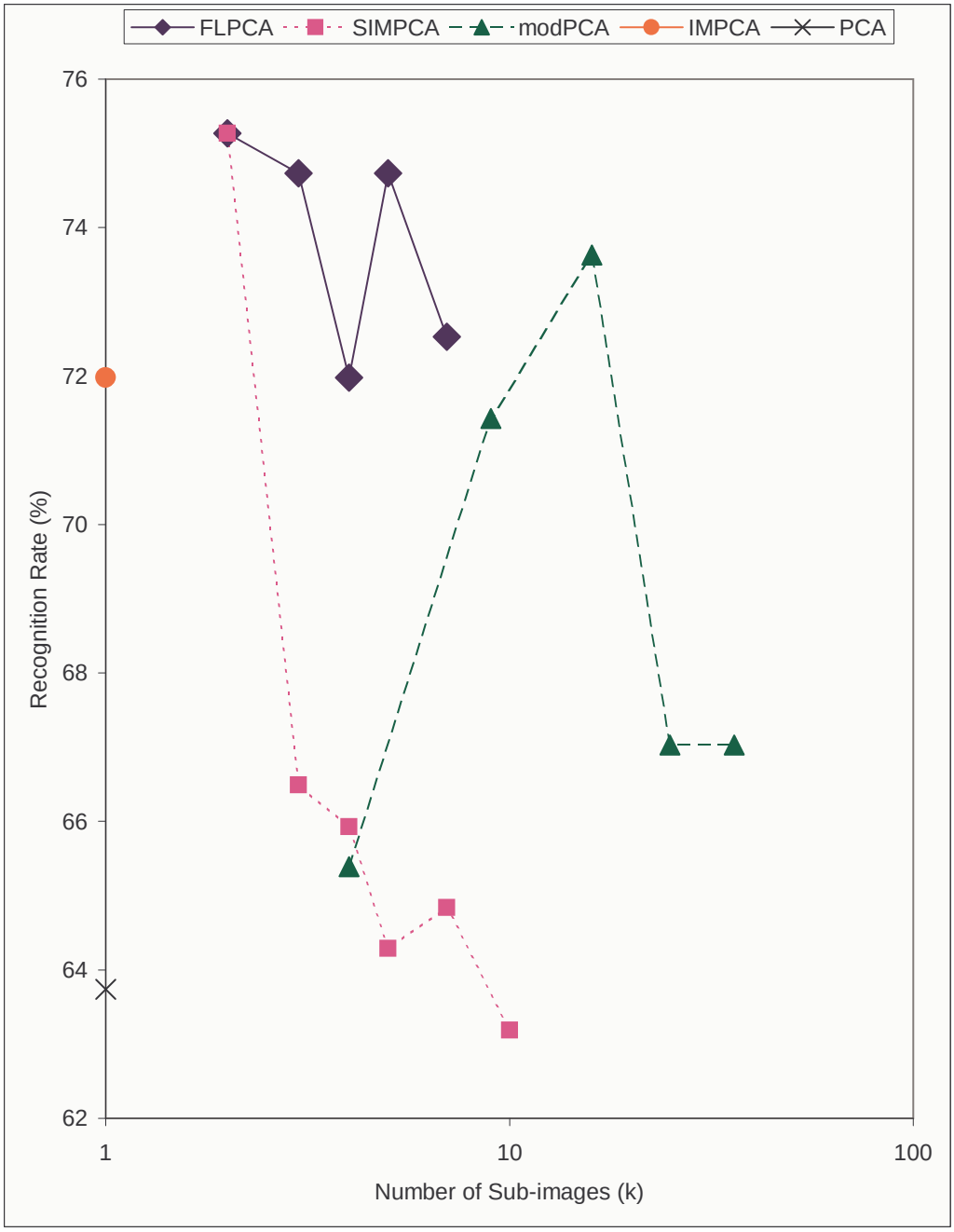


Figure 5.7: *Comparison of recognition rate for UMIST face data.* FLPCA shows better consistency across various number of sub-images as compared to modPCA and SIMPCA. FLPCA and SIMPCA show highest recognition rate as compared to PCA, IMPCA (2DPCA) and modPCA methods.

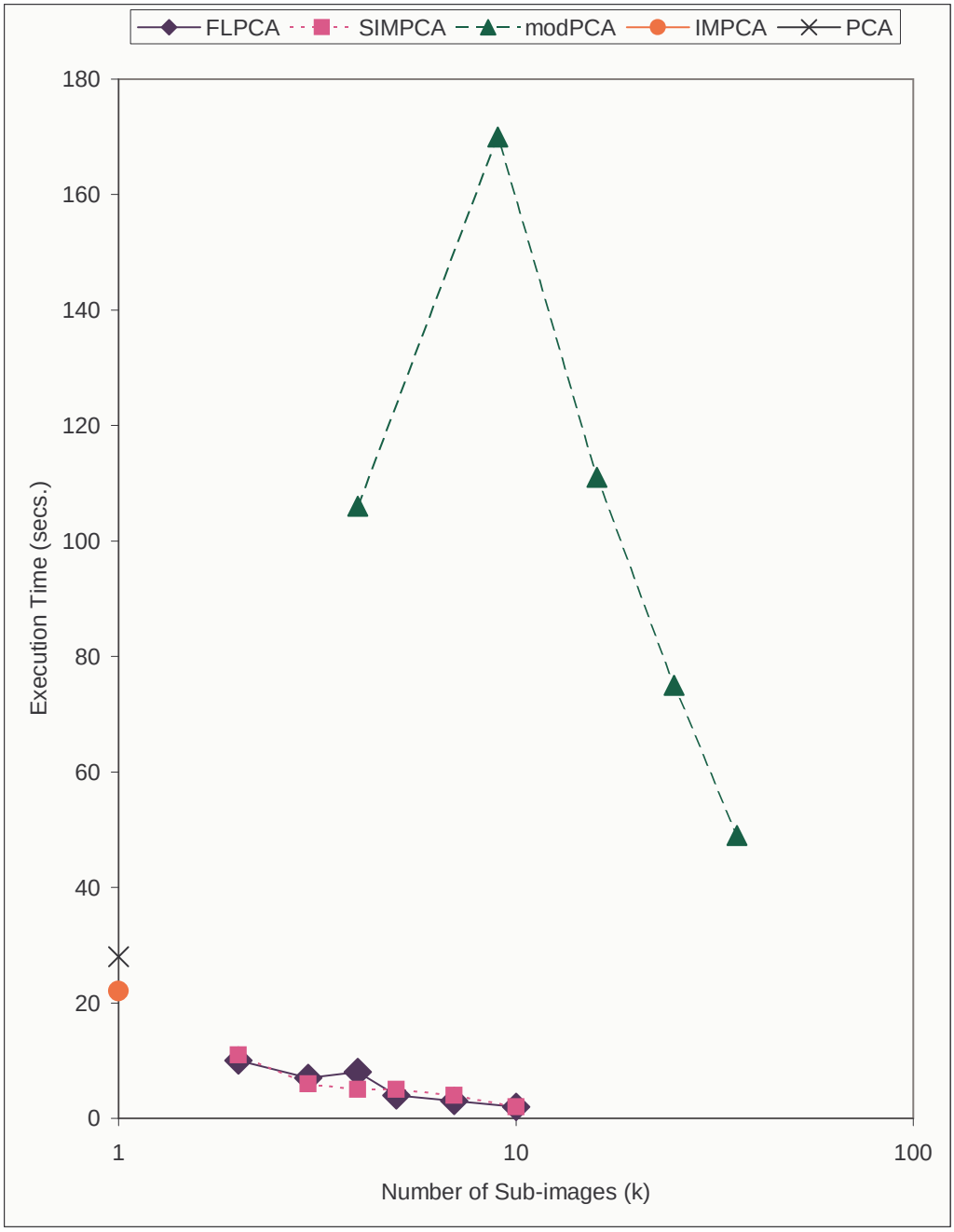


Figure 5.8: *Comparison of computational time for UMIST face data.* FLPCA and SIMPCA show better efficiency across various number of sub-images as compared to modPCA. FLPCA and SIMPCA show better computational time as compared to IMPCA and the efficient implementation of PCA [103].

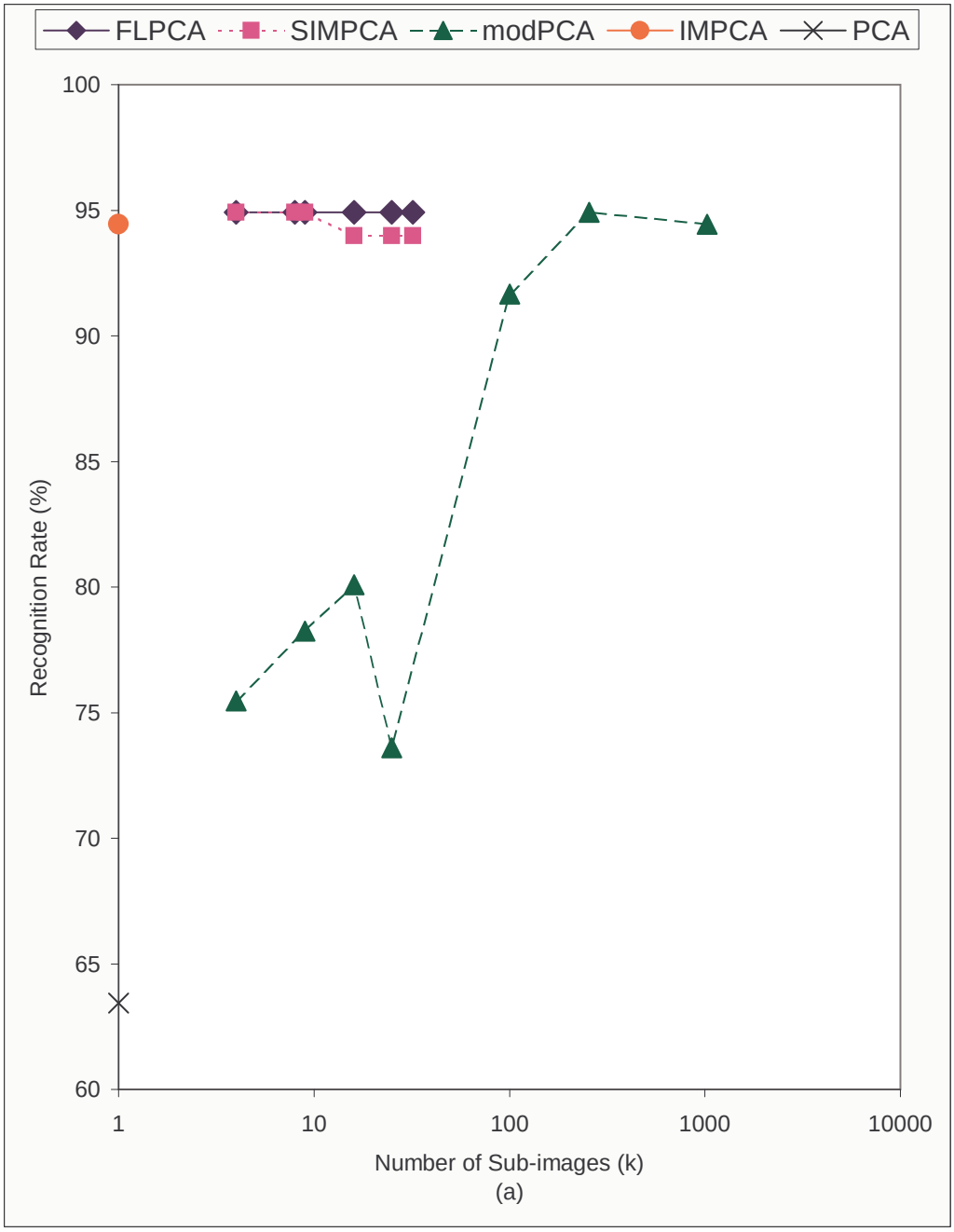


Figure 5.9: *Comparison of recognition rate for PolyU palmprint data.* FLPCA and SIMPCA show better consistency across various number of sub-images as compared to modPCA. FLPCA and SIMPCA also outperform PCA.

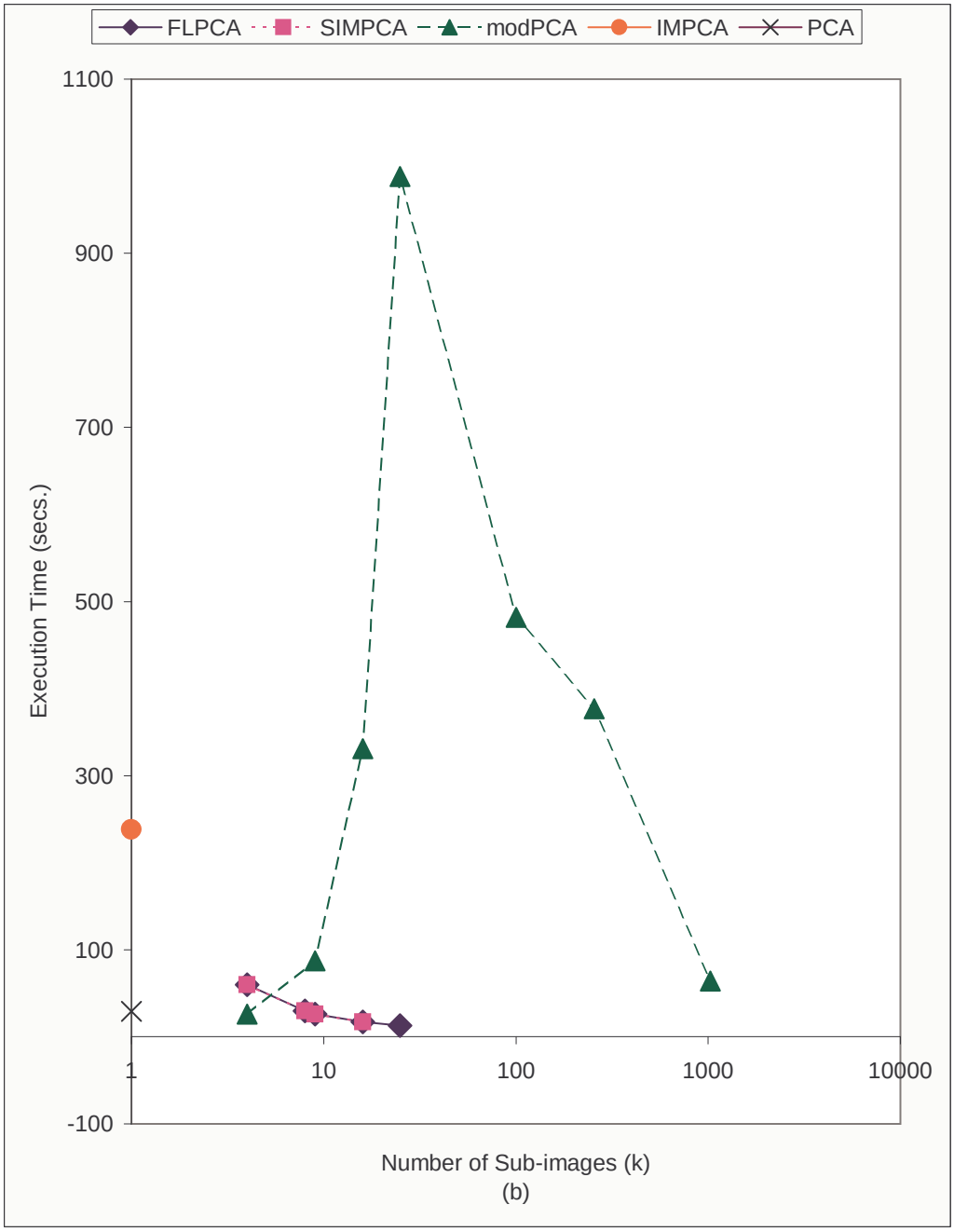


Figure 5.10: *Comparison of computational time for PolyU palmprint data.* FLPCA and SIMPCA show better efficiency across various number of sub-images as compared to modPCA. FLPCA and SIMPCA also show better computational time as compared to IMPCA (2DPCA). PCA shows competitive time complexity to SIMPCA and FLPCA because we used the efficient implementation [103] instead of the original implementation of PCA.

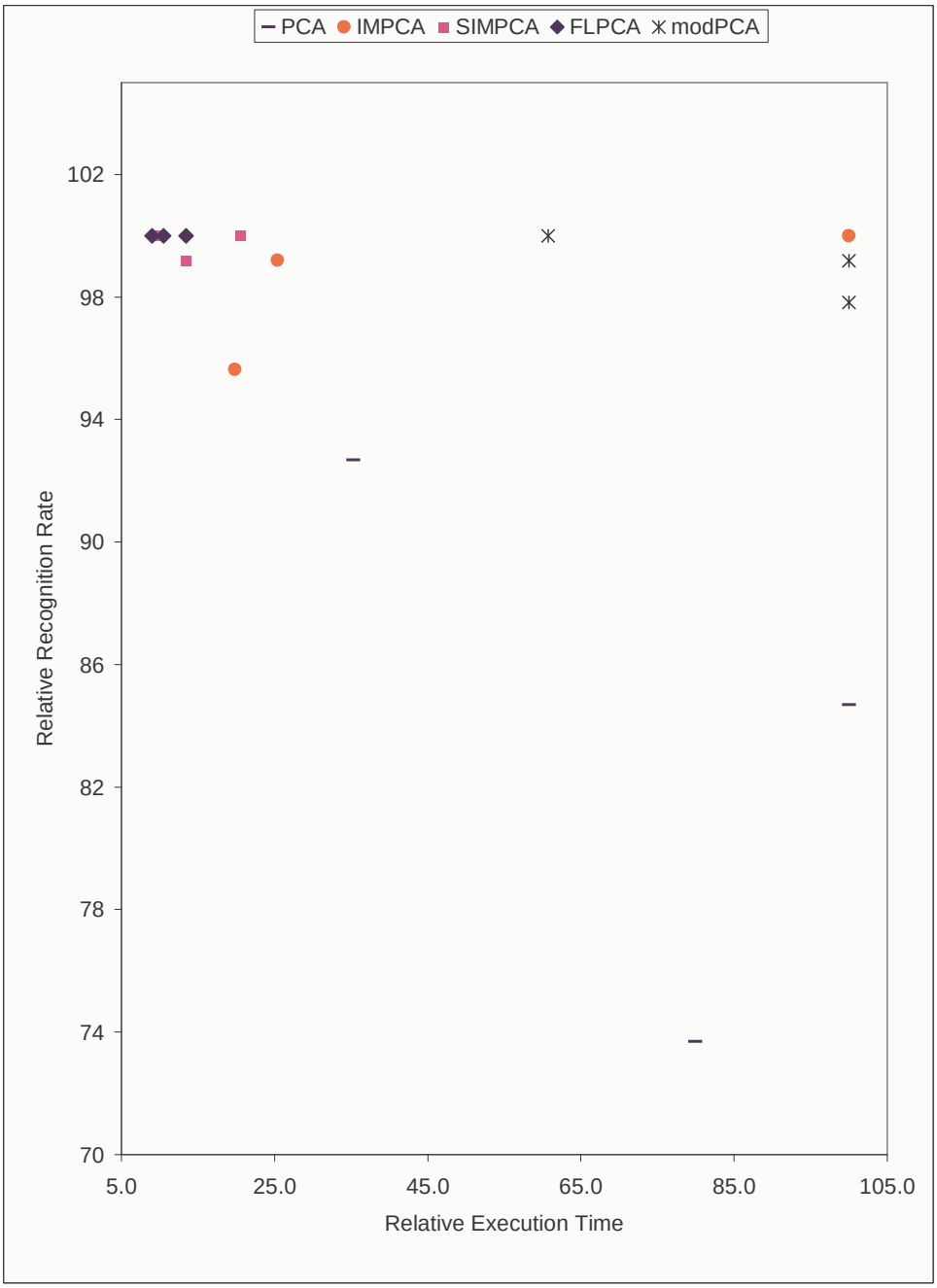


Figure 5.11: *Execution time versus Recognition rate with respect to 3 face data sets (UMIST, ORL, Yale).* FLPCA and SIMPCA points occupy left top corner part of the chart, forming a cluster of superior recognition rate at less computational overhead as compared to other methods.

Table 5.3: Comparison of maximum recognition rates of proposed FP-PCA approaches over other PCA methods for ORL face data

Method	Reco. Rate	FAR	FRR	TER	Time (secs.)	k	u	PVs	Total PVs used
PCA	84.44	3.71	11.85	15.56	6	1	–	–	9
modPCA	90.37	0.0	9.63	9.63	67	16	28×23	7	112
IMPCA (2DPCA)	90.38	2.22	7.40	9.62	17	1	–	–	4
SIMPCA	90.37	0.00	9.63	9.63	9	2	46	4	8
FLPCA	91.11	1.48	7.41	8.89	9	2	46	5	4

k: Sub-images per image; For modPCA, u is sub-image size and for other methods, $m \times u$.
PVs: Number of Projection Vectors (eigenvectors chosen for projection) per sub-image set

Table 5.4: Comparison of maximum recognition rates of proposed FP-PCA approaches over other PCA methods for UMIST face data

Method	Reco. Rate	FAR	FRR	TER	Time (secs.)	k	u	PVs	Total PVs used
PCA	63.74	1.65	34.61	36.26	28	1	–	–	35
modPCA	73.63	0.55	25.82	26.37	111	16	28×23	1	16
IMPCA (2DPCA)	71.98	12.09	15.93	28.02	22	1	–	–	2
SIMPCA	75.27	0.0	24.73	24.73	11	2	46	1	2
FLPCA	75.27	0.0	24.73	24.73	10	2	46	1	2

k: Sub-images per image; For modPCA, u is sub-image size and for other methods, $m \times u$.
PVs: Number of Projection Vectors (eigenvectors chosen for projection) per sub-image set

Table 5.5: Comparison of maximum recognition rates of proposed FP-PCA approaches over other PCA methods for PolyU palmprint data

Method	Reco. Rate	FAR	FRR	TER	Time (secs.)	k	u	PVs	Total PVs used
PCA	63.43	36.11	0.46	36.57	29	1	–	–	15
modPCA	94.91	0.0	5.09	5.09	377	256	17×24	300	76800
IMPCA (2DPCA)	94.45	0.46	5.09	5.55	238	1	–	–	5
SIMPCA	94.91	0.0	5.09	5.09	26	9	42	1	9
FLPCA	94.91	0.0	5.09	5.09	26	9	42	1	5

k: Sub-images per image; For modPCA, u is sub-image size and for other methods, $m \times u$.
PVs: Number of Projection Vectors (eigenvectors chosen for projection) per sub-image set

Chapter 6

Theoretical Analysis of Feature Partitioning based PCA Approaches

6.1 Introduction

Note: The work in this chapter has been submitted to *IEEE Transactions on Pattern Analysis and Machine Intelligence* Journal[1].

In the last three Chapters, that is, Chapters 3-5, we discussed feature partitioning framework and various fundamental feature partitioning issues; then we proposed a novel feature partitioning based PCA (FP-PCA) approach, SubXPCA, which is an instance of proposed feature partitioning framework, which mainly focussed on ad-

[1]Kadappagari Vijaya Kumar and Atul Negi, "A generalized study of feature partitioning methods to principal component analysis", submitted to *IEEE Transactions on Pattern Analysis and Machine Intelligence*, in Feb. 2009.

dressing (i) Loss of inter-sub-pattern correlations, (ii) feature-order dependency. Subsequently, we presented two FP-PCA approaches, SIMPCA and FLPCA, for image data, which are again instances of feature partitioning framework. We demonstrated the applicability of the proposed FP-PCA methods by conducting extensive experiments on various benchmarking data sets. This proved that we obtained not only better computational time but also superior recognition rates. Here we study the FP-PCA approaches to gain a deeper insight into the performance of these methods and the study may form a basis for proposing novel PCA methods in future.

In this Chapter, we focus on theoretical study of *variance-covariance structure* captured by various FP-PCA methods and establish their properties. It is well known that Principal Component Analysis methods reduce the number of dimensions (features) by exploiting *variance-covariance structure* of the given training data and nothing else.

The rest of the Chapter is organized as follows. In the section 6.2, we present some definitions which are essential for the rest of the presentation. Next, we present theoretical study on FP-PCA methods in the form of properties in section 6.3. We demonstrate the theoretical properties through experimentation on UCI waveform data and ORL face data in section 6.4. We summarize in the last section.

6.2 Definitions

In Chapter 3, we have already defined the definitions of (i) a Sub-pattern or a Block (**Definition 1)**, (ii) Sub-pattern Set or Block Set (**Definition 2)**, (iii) Intra-block (Intra-sub-pattern) covariance matrix of a homogeneous or heterogeneous block set

(**Definition 3)**, (iv) Inter-block (Inter-sub-pattern) covariance matrix (**Definition 4)** and (v) Local features (**Definition 5)**. Here we present more definitions which are used subsequently throughout our discussion.

Definition 6 Orthogonal property.

Orthogonal property of an eigenvector matrix, $\mathbf{W} = [\mathbf{e}_1\, \mathbf{e}_2\, \ldots\, \mathbf{e}_d]$*, is given by [75]*

$$\mathbf{W}_{d\times d}.\mathbf{W}_{d\times d}^T = \mathbf{W}_{d\times d}^T.\mathbf{W}_{d\times d} = \mathbf{I}_d \tag{6.1}$$

where $\mathbf{e}_i = [e_{i_1}, e_{i_2}, \ldots, e_{i_d}]^T$ *is the eigenvector corresponding to the eigenvalue* λ_i *and* $\mathbf{I}_d$ *is a* $d \times d$ *Identity matrix.*

From $\mathbf{W}.\mathbf{W}^T = \mathbf{I}_d$ *we get the following.*

$$\sum_{i=1}^{d}(e_{i_q}.e_{i_q}) = 1;\ \ \forall\, q \in \{1, 2, \ldots d\} \tag{6.2}$$

$$\sum_{i=1}^{d}(e_{i_q}.e_{i_t}) = 0;\ \ \forall\, q, t \in \{1, 2, \ldots d\}; q \neq t \tag{6.3}$$

Similarly from $\mathbf{W}^T.\mathbf{W} = \mathbf{I}_d$ *we get the following.*

$$\sum_{q=1}^{d}(e_{i_q}.e_{i_q}) = 1;\ \ \forall\, i \in \{1, 2, \ldots d\} \tag{6.4}$$

$$\sum_{q=1}^{d}(e_{i_q}.e_{j_q}) = 0;\ \ \forall\, i, j \in \{1, 2, \ldots d\}; i \neq j \tag{6.5}$$

Definition 7 Spectral Decomposition *[75].*

Spectral Decomposition of a covariance matrix, $\mathbf{C}_{d\times d}$ *is given as follows.*

$$\mathbf{C}_{d\times d} = \sum_{i=1}^{d}(\lambda_i.\mathbf{e}_i.\mathbf{e}_i^T) = \mathbf{W}_{d\times d}.\mathbf{\Lambda}_{d\times d}.\mathbf{W}_{d\times d}^T \tag{6.6}$$

$$\Rightarrow \mathbf{C}_{d\times d}.\mathbf{W}_{d\times d} = \mathbf{W}_{d\times d}.\mathbf{\Lambda}_{d\times d} \tag{6.7}$$

It is well known that $\mathbf{W}_{d\times d}.\mathbf{W}^T_{d\times d} = \mathbf{W}^T_{d\times d}.\mathbf{W}_{d\times d} = \mathbf{I}_d$, *where* $\mathbf{W} = [\mathbf{e}_1\, \mathbf{e}_2\, \ldots\, \mathbf{e}_d]$ *is a matrix of* d *column eigenvectors and* $\mathbf{e}_i = [e_{i_1}, e_{i_2}, \ldots, e_{i_d}]^T$ *is the eigenvector corresponding to the eigenvalue* λ_i *and* $\mathbf{\Lambda}_{d\times d}$ *is a diagonal matrix, with eigenvalues at the diagonal.*

Definition 8 Feature Partitioning PCA Type-I method (FP-PCA-Type-I).

FP-PCA-Type-I method extracts local features as follows. We assume that the method divides each pattern into $k\,(\geq 2)$ *sub-patterns (blocks).*

(i) For each homogeneous sub-pattern set, $\mathbf{P}^j$,$j \in \{1, 2, \ldots, k\}$*: (a) Compute eigenvectors using eigenvalue decomposition. (b) Project* $\mathbf{P}^j$ *onto first* r *eigenvectors computed to get local features of* $\mathbf{P}^j$.

(ii) For each pattern, $\mathbf{X}_i$*: Concatenate the local features of* k *sub-patterns (obtained in the preceding step) belong to* $\mathbf{X}_i$. *Here we obtain a total of* $k.r$ *local features for each pattern.*

Examples: SubPCA[21], SIMPCA (Chapter 5), Multi-block PCA [128], Region based PCA [136], etc.

Definition 9 Feature Partitioning PCA Type-II method (FP-PCA-Type-II).

FP-PCA-Type-II method extracts local features as follows. We assume that the method divides each pattern into $k\,(\geq 2)$ *sub-patterns (blocks).*

(i) Compute eigenvectors from heterogeneous sub-pattern set, $\mathbf{Q}$.

(ii) For each pattern, $\mathbf{X}_i$*: Extract* r *local features from each of* k *sub-patterns by projecting onto first* r *eigenvectors of* $\mathbf{Q}$ *and concatenate them to get a total of* $(k.r)$ *local features.*

Examples: modPCA [53], EigenRegions method [45]

Definition 10 Feature Partitioning PCA Type-III method (FP-PCA-Type-III).

FP-PCA-Type-III method extracts features as follows. We assume that the method divides each pattern into $k\,(\geq 2)$ *sub-patterns (blocks).*

(i) For each homogeneous sub-pattern set, $\mathbf{P}^j$*,*$j \in \{1, 2, \ldots, k\}$*: (a) Compute eigenvectors using eigenvalue decomposition. (b) Project* $\mathbf{P}^j$ *onto first* r *eigenvectors computed to get local features of* $\mathbf{P}^j$*.*

(ii) For each pattern, $\mathbf{X}_i$*: Concatenate the local features (a total of* $k.r$*) of* k *sub-patterns (obtained in the preceding step) belong to* $\mathbf{X}_i$*.*

(iii) Next, extract global features among patterns of $(k.r)$ *locally-extracted features by using inter-subpattern covariance matrix of these local features.*

Examples: SubXPCA (Chapter 4), FLPCA (Chapter 5). These are novel FP-PCA approaches derived from FP framework proposed (Chapter 3) which was not existing in the literature.

Definition 11 Feature Partitioning PCA Type-IV method (FP-PCA-Type-IV).

FP-PCA-Type-IV method extracts features as follows. We assume that the method divides each pattern into $k\,(\geq 2)$ *sub-patterns (blocks).*

(i) Compute eigenvectors from heterogeneous sub-pattern set, $\mathbf{Q}$*.*

(ii) For each pattern, $\mathbf{X}_i$*: Extract* r *local features from each of* k *sub-patterns by projecting onto first* r *eigenvectors of* $\mathbf{Q}$ *and concatenate them to get a total of* $(k.r)$ *local features.*

(iii) Next, extract global features among patterns of local features by using inter-subpattern covariance matrix of these $(k.r)$ *local features.*

Examples: Not found in the literature.

Definition 12 Holistic PCA *or* Global PCA *or* Whole-Pattern based PCA.
Holistic PCA is some variation of PCA method which extracts features based on whole-patterns and not based on sub-patterns (blocks). Examples: Classical PCA [76], 2DPCA (Section 2.3 of Chapter 2), Kernel PCA (Section 2.5 of Chapter 2), etc.

6.3 Properties

Here we assume that all the sub-patterns of a pattern are equally-sized. Also we assume SubPCA (Section 2.2 of Chapter 2), modPCA (Section 2.2 of Chapter 2), SubXPCA (Chapter 4) and classical PCA as typical representatives of FP-PCA-Type-I, FP-PCA-Type-II, FP-PCA-Type-III and Holistic PCA respectively.

Theorem 16 *FP-PCA-Type-I method is based on* $\mathbf{C}^{j,j}$ *and ignores* $\mathbf{C}^{q,t}$. *Here,* $\mathbf{C}^{j,j}$ *is the intra-block covariance matrix of* j^{th} *block set (sub-pattern set),* $\mathbf{P}^j$; $\mathbf{C}^{q,t}$ *is the matrix of inter-block covariances of block sets,* $\mathbf{P}^q$ *and* $\mathbf{P}^t$, $q \neq t$ *and* $\mathbf{P}^j$ *is the set of* j^{th} *sub-patterns (blocks) of patterns,* $\{\mathbf{X}_i; i = 1, 2, \ldots, N\}$.

Significance : The Theorem says that FP-PCA-Type-I method exploits only covariance structure within each block, which renders it to have more local Principal Components (less dimensionality reduction). However, using the only local covariance structure enable the methods to generalize well if local variations are prominent.

Proof 16 *In a FP-PCA-Type-I method: (i) every pattern of dimension* d, $\mathbf{X}_i$, *is partitioned into* $k\,(2 \leq k \leq \frac{d}{2})$ *sub-patterns (blocks),* $\mathbf{X}_i^1, \mathbf{X}_i^2, \ldots, \mathbf{X}_i^k$. *Without loss of generality, let us assume that first sub-pattern (block) contains first* $u\,(u \geq 2)$ *features*

and second sub-pattern (block) contains next u features and so on and (iii) we extract local features from these sub-patterns (blocks) (Defn. 8).

Outline of Proof: We first consider the covariance structure of entire data, $\mathbf{C}$ *(Step 1), partition it according to blocks and we explore the parts of covariance structure used and not used by FP-PCA-Type-I method (Step 2).*

Step 1: The covariance matrix, $\mathbf{C}_{d\times d}$ *of the original data* $\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\}$ *is given by*

$$\mathbf{C}_{d\times d} = \begin{bmatrix} \sigma_{1,1} & \sigma_{1,2} & \sigma_{1,3} & \sigma_{1,4} & \ldots & \sigma_{1,d} \\ \sigma_{2,1} & \sigma_{2,2} & \sigma_{2,3} & \sigma_{2,4} & \ldots & \sigma_{2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \sigma_{d,1} & \sigma_{d,2} & \sigma_{d,3} & \sigma_{d,4} & \ldots & \sigma_{d,d} \end{bmatrix}$$

where $\sigma_{i,i}$ *is the variance of* i^{th} *dimension and* $\sigma_{i,j}$ *is the covariance between* i^{th} *and* j^{th} *dimensions.*

Step 2: Further, $\mathbf{C}$ *can be partitioned as follows [75].*

$$\mathbf{C}_{d\times d} = \begin{bmatrix} \mathbf{C}^{1,1} & \mathbf{C}^{1,2} & \ldots & \mathbf{C}^{1,k} \\ \mathbf{C}^{2,1} & \mathbf{C}^{2,2} & \ldots & \mathbf{C}^{2,k} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{C}^{k,1} & \mathbf{C}^{k,2} & \ldots & \mathbf{C}^{k,k} \end{bmatrix}$$

where $\mathbf{C}^{j,j}$ *(or* $\mathbf{C}^{j}$ *) is the covariance matrix for the* j^{th} *sub-pattern set (block),* $\mathbf{P}^j$ *and* $\mathbf{C}^{i,j}$ *is the matrix of covariances between features of* $\mathbf{P}^i$ *and* $\mathbf{P}^j$*,* $i \neq j$*. It is clear that* $\mathbf{C}^{i,j} = \mathbf{C}^{j,i}$*;* $\forall i, j \in \{1, 2, \ldots, k\}$.

Further the matrices, $\mathbf{C}^{j,j}$ *and* $\mathbf{C}^{i,j}$*, are given as follows.*

$$(\mathbf{C}^{j,j})_{u\times u} = (\mathbf{C}^{j})_{u\times u} = \begin{bmatrix} \sigma_{q,q} & \sigma_{q,(q+1)} & \cdots & \sigma_{q,q+u-1} \\ \sigma_{(q+1),q} & \sigma_{q+1,q+1} & \cdots & \sigma_{q+1,q+u-1} \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{q+u-1,q} & \sigma_{q+u-1,q+1} & \cdots & \sigma_{q+u-1,q+u-1} \end{bmatrix}$$

where $q = u.(j-1)+1$ *;* $j \in \{1, 2, \ldots, k\}$

$$(\mathbf{C}^{i,j})_{u\times u} = \begin{bmatrix} \sigma_{t,p} & \sigma_{t,p+1} & \cdots & \sigma_{t,p+u-1} \\ \sigma_{t+1,p} & \sigma_{t+1,p+1} & \cdots & \sigma_{t+1,p+u-1} \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{t+u-1,p} & \sigma_{t+u-1,p+1} & \cdots & \sigma_{t+u-1,p+u-1} \end{bmatrix}$$

where $t = u.(i-1)+1$*,* $p = u.(j-1)+1$*;* $i, j \in \{1, 2, \ldots, k\}$*,* $i \neq j$

Step 3: It is clear that $\mathbf{C}^{j,j}$ *(or simply* $\mathbf{C}^{j}$*) indicates intra-block covariance structure of block* $\mathbf{P}^{j}$ *(Defn. 3) and* $\mathbf{C}^{i,j}$ *indicates inter-block covariance structure of blocks* $\mathbf{P}^{i}$ *and* $\mathbf{P}^{j}$*;* $i \neq j$ *(Defn. 4).*

By definition, FP-PCA-Type-I method (Defn. 8) extracts features within sub-pattern (block) set only. Therefore such methods use intra-block covariance structure, $\mathbf{C}^{j,j}$ *of* $\mathbf{P}^{j}$*;* $j \in \{1, 2, \ldots, k\}$ *only.*

Hence the Theorem follows.

Theorem 17 *FP-PCA-Type-III method is based on* $\mathbf{C}^{j,j}$ *and* $\mathbf{C}^{g}_{i,j}$ *as well. Here,* $\mathbf{C}^{j,j}$ *is the intra-block covariance matrix of the block,* $\mathbf{P}^{j}$*;* $\mathbf{C}^{g}_{i,j}$ *is the matrix of inter-block covariances of local features of blocks,* $\mathbf{P}^{i}$ *and* $\mathbf{P}^{j}$*,* $i \neq j$ *and* $\mathbf{P}^{j}$ *is the set of* j^{th} *homogeneous sub-pattern set.*

Significance : The Theorem says that FP-PCA-Type-III method exploits not only covariance structure within each block, also exploits covariance between local features of different blocks, which (i) results in lower number of Principal Components (high dimensionality reduction) and (ii) enables FP-PCA-Type-III method to generalize well in the context of local or global variations.

Proof 17 *In an FP-PCA-Type-III method : (i) Every pattern of dimension d,* $\mathbf{X}_i$*, is partitioned into* $k\,(2 \leq k \leq \frac{d}{2})$ *sub-patterns (blocks),* $\mathbf{X}_i^1, \mathbf{X}_i^2, \ldots, \mathbf{X}_i^k$*. Without loss of generality, let us assume that first sub-pattern (block) contains first* $u\,(u \geq 2)$ *features and second sub-pattern (block) contains next u features and so on. (ii) We extract* $r\,(\leq u)$ *local features from each of* $\mathbf{P}^j\,; j \in \{1, 2, \ldots, k\}$ *and concatenate them to form patterns of locally- extracted features (iii) Finally, we extract* $w\,(\leq k.r)$ *global features by using inter-block covariances among* $(k.r)$ *local features (Defn. 10).*

Outline of Proof: We first consider covariance structure of entire data, $\mathbf{C}$ *(Step 1), partition it according to blocks and we explore the parts of covariance structure used and not used by FP-PCA-Type-III method (Step 2). Next we elaborate how FP-PCA-Type-III method uses inter-block covariance structure among local features (i.e. in locally-projected space) (Step 3).*

Step 1: The covariance matrix, $\mathbf{C}$ *of the original data* $\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\}$ *is given by*

$$\mathbf{C}_{d\times d} = \begin{bmatrix} \sigma_{1,1} & \sigma_{1,2} & \sigma_{1,3} & \sigma_{1,4} & \dots & \sigma_{1,d} \\ \sigma_{2,1} & \sigma_{2,2} & \sigma_{2,3} & \sigma_{2,4} & \dots & \sigma_{2,d} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \sigma_{d,1} & \sigma_{d,2} & \sigma_{d,3} & \sigma_{d,4} & \dots & \sigma_{d,d} \end{bmatrix}$$

where $\sigma_{i,i}$ *is the variance of* i^{th} *feature and* $\sigma_{i,j}$ *is the covariance between* i^{th} *and* j^{th} *features.*

Step 2: Further, $\mathbf{C}$ *can be partitioned as follows [75].*

$$\mathbf{C}_{d\times d} = \begin{bmatrix} \mathbf{C}^{1,1} & \mathbf{C}^{1,2} & \dots & \mathbf{C}^{1,k} \\ \mathbf{C}^{2,1} & \mathbf{C}^{2,2} & \dots & \mathbf{C}^{2,k} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{C}^{k,1} & \mathbf{C}^{k,2} & \dots & \mathbf{C}^{k,k} \end{bmatrix}$$

where $\mathbf{C}^{j,j}$ *(or* $\mathbf{C}^{j}$*) is the covariance matrix for the* j^{th} *sub-pattern set (block),* $\mathbf{P}^{j}$ *and* $\mathbf{C}^{i,j}$ *is the matrix of covariances between features of* $\mathbf{P}^{i}$ *and* $\mathbf{P}^{j}$*. It is clear that* $\mathbf{C}^{i,j} = \mathbf{C}^{j,i}; \forall i, j \in \{1, 2, \dots, k\}$.

Further the matrices, $\mathbf{C}^{j,j}$ *and* $\mathbf{C}^{i,j}$ *are given as follows.*

$$(\mathbf{C}^{j,j})_{u\times u} = (\mathbf{C}^{j})_{u\times u} = \begin{bmatrix} \sigma_{q,q} & \sigma_{q,(q+1)} & \dots & \sigma_{q,q+u-1} \\ \sigma_{(q+1),q} & \sigma_{q+1,q+1} & \dots & \sigma_{q+1,q+u-1} \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{q+u-1,q} & \sigma_{q+u-1,q+1} & \dots & \sigma_{q+u-1,q+u-1} \end{bmatrix}$$

where $q = u.(j-1)+1$ *;* $j \in \{1, 2, \dots, k\}$

$$\mathbf{C}^{i,j} = \begin{bmatrix} \sigma_{t,p} & \sigma_{t,p+1} & \cdots & \sigma_{t,p+u-1} \\ \sigma_{t+1,p} & \sigma_{t+1,p+1} & \cdots & \sigma_{t+1,p+u-1} \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{t+u-1,p} & \sigma_{t+u-1,p+1} & \cdots & \sigma_{t+u-1,p+u-1} \end{bmatrix}$$

where $t = u.(i-1)+1$, $p = u.(j-1)+1$ *;* $i, j \in \{1, 2, \ldots, k\}$, $i \neq j$

Step 3: It is clear that $\mathbf{C}^{j,j}$ *indicates intra-block covariance structure of block* $\mathbf{P}^j$ *(Defn. 3) and* $\mathbf{C}^{i,j}$ *indicates inter-block covariance structure of blocks* $\mathbf{P}^i$ *and* $\mathbf{P}^j$*;* $i \neq j$*, in the original feature space (Defn. 4).*

By definition (Defn. 10), FP-PCA-Type-III method extracts global features by using inter-block covariances among $(k.r)$ *local features,* $(\mathbf{C}^g)_{k.r \times k.r}$ *in the locally-projected feature space (Step-4 of Section 4.2 of Chapter 4) and is given by*

$$(\mathbf{C}^g)_{k.r \times k.r} = \begin{bmatrix} (\mathbf{0})_{r\times r} & (\mathbf{C}^g_{1,2})_{r\times r} & \cdots & (\mathbf{C}^g_{1,k})_{r\times r} \\ (\mathbf{C}^g_{2,1})_{r\times r} & (\mathbf{0})_{r\times r} & \cdots & (\mathbf{C}^g_{2,k})_{r\times r} \\ \vdots & \vdots & \vdots & \vdots \\ (\mathbf{C}^g_{k,1})_{r\times r} & (\mathbf{C}^g_{k,2})_{r\times r} & \cdots & (\mathbf{0})_{r\times r} \end{bmatrix}$$

where $(\mathbf{0})_{r\times r}$ *is the zero matrix,* $(\mathbf{C}^g_{i,j})_{r\times r}$ *is the matrix of covariances between local features of* i^{th} *and* j^{th} *sub-patterns and given as follows.*

$$(\mathbf{C}^g_{i,j})_{r\times r} = \begin{bmatrix} \sigma^g_{t,p} & \sigma^g_{t,p+1} & \cdots & \sigma^g_{t,p+r-1} \\ \sigma^g_{t+1,p} & \sigma^g_{t+1,p+1} & \cdots & \sigma^g_{t+1,p+r-1} \\ \vdots & \vdots & \vdots & \vdots \\ \sigma^g_{t+r-1,p} & \sigma^g_{t+r-1,p+1} & \cdots & \sigma^g_{t+r-1,p+r-1} \end{bmatrix}$$

where $t = r.(i-1)+1$, $p = r.(j-1)+1$ *;* $i, j \in \{1, 2, \ldots, k\}$, $i \neq j$

where $\sigma^g_{q,s}$ *is the covariance between* q^{th} *and* s^{th} *local features of different sub-patterns*

(blocks).

It is clear that FP-PCA-Type-III method utilizes inter-block covariance structure of original feature space, $\mathbf{C}^{i,j}$*, in locally-projected feature space in the form of* $\mathbf{C}^{g}_{i,j}$*.*

Hence the Theorem follows.

Lemma 1 $(\mathbf{Y}_i)_{k.r\times 1} = (\mathbf{V})^T_{k.r\times d}.(\mathbf{X}_i)_{d\times 1}$*. Here,* $\mathbf{Y}_i$ *is the locally-projected pattern of the original pattern* $\mathbf{X}_i$ *of dimension* d*;* $\mathbf{V}$ *is the combined matrix of selected* $k.r$ *local eigenvectors which is given by*

$$(\mathbf{V})_{d\times k.r} = \begin{bmatrix} (\mathbf{E}^1)_{u\times r} & (\mathbf{0})_{u\times r} & \dots & (\mathbf{0})_{u\times r} \\ (\mathbf{0})_{u\times r} & (\mathbf{E}^2)_{u\times r} & \dots & (\mathbf{0})_{u\times r} \\ \vdots & \vdots & \vdots & \vdots \\ (\mathbf{0})_{u\times r} & (\mathbf{0})_{u\times r} & \dots & (\mathbf{E}^k)_{u\times r} \end{bmatrix}_{d\times k.r}$$

where $(\mathbf{E}^j)_{u\times r}$ *is the matrix of* r *local column eigenvectors of* j^{th} *sub-pattern set,* $(\mathbf{P}^j)_{N\times u}$*,* $j \in \{1, 2, \dots, k\}$*;* k *is the number of sub-patterns or blocks (See section 4.2 of Chapter 4 for terminology).*

Proof 1 *The locally-projected pattern,* $(\mathbf{Y}_i)_{k.r\times 1}$ *(Section 4.2 of Chapter 4) is obtained as follows.*

$$(\mathbf{Y}_i)_{k.r\times 1} = \begin{bmatrix} (\mathbf{E}^1)^T_{r\times u}.(\mathbf{X}^1_i)_{u\times 1} \\ (\mathbf{E}^2)^T_{r\times u}.(\mathbf{X}^2_i)_{u\times 1} \\ (\vdots)(\vdots) \\ (\mathbf{E}^k)^T_{r\times u}.(\mathbf{X}^k_i)_{u\times 1} \end{bmatrix}$$

where $\mathbf{X}^j_i$ *is the* j^{th} *sub-pattern of pattern* $\mathbf{X}_i$ *and* $(\mathbf{E}^j)_{u\times r}$ *is the set of* r *local eigenvectors of* $\mathbf{P}^j$*;* $j \in \{1, 2, \dots, k\}$*.*

$$\Rightarrow (\mathbf{Y}_i)_{k.r\times 1} = \begin{bmatrix} (\mathbf{E}^1)_{u\times r} & (\mathbf{0})_{u\times r} & \dots & (\mathbf{0})_{u\times r} \\ (\mathbf{0})_{u\times r} & (\mathbf{E}^2)_{u\times r} & \dots & (\mathbf{0})_{u\times r} \\ \vdots & \vdots & \vdots & \vdots \\ (\mathbf{0})_{u\times r} & (\mathbf{0})_{u\times r} & \dots & (\mathbf{E}^k)_{u\times r} \end{bmatrix}^T_{k.r\times d} . \begin{bmatrix} \mathbf{X}_i^1 \\ \mathbf{X}_i^2 \\ \vdots \\ \mathbf{X}_i^k \end{bmatrix}_{d\times 1}$$

$$\Rightarrow (\mathbf{Y}_i)_{k.r\times 1} = (\mathbf{V})^T_{k.r\times d} . \begin{bmatrix} \mathbf{X}_i^1 \\ \mathbf{X}_i^2 \\ \vdots \\ \mathbf{X}_i^k \end{bmatrix}_{d\times 1}$$

$\Rightarrow (\mathbf{Y}_i)_{k.r\times 1} = (\mathbf{V})^T_{k.r\times d}.(\mathbf{X}_i)_{d\times 1}$ *(because* $(\mathbf{X}_i)_{d\times 1} = \{\mathbf{X}_i^1 \cup \mathbf{X}_i^2 \cup \ldots \cup \mathbf{X}_i^k\}$*)*

Hence the Lemma follows.

Lemma 2 $\mathbf{V}_{d\times d}.\mathbf{V}^T_{d\times d} = \mathbf{I}_d = \mathbf{V}^T_{d\times d}.\mathbf{V}_{d\times d}$. *Here, d is the dimensionality of the original pattern $\mathbf{X}_i$; k is the number of sub-patterns or blocks; $\mathbf{V}$ is the combined matrix of all d local eigenvectors of the sub-pattern sets, $\{\mathbf{P}^j\}$, $j = 1, 2, \ldots, k$ (See section 4.2 of Chapter 4 for terminology), which is given by*

$$(\mathbf{V})_{d\times d} = \begin{bmatrix} (\mathbf{E}^1)_{u\times u} & (\mathbf{0})_{u\times u} & \dots & (\mathbf{0})_{u\times u} \\ (\mathbf{0})_{u\times u} & (\mathbf{E}^2)_{u\times u} & \dots & (\mathbf{0})_{u\times u} \\ \vdots & \vdots & \vdots & \vdots \\ (\mathbf{0})_{u\times u} & (\mathbf{0})_{u\times u} & \dots & (\mathbf{E}^k)_{u\times u} \end{bmatrix}_{d\times d}$$

Proof 2 $\mathbf{V}_{d\times d}.\mathbf{V}^T_{d\times d} =$

$$\begin{bmatrix} (\mathbf{E}^1)_{u\times u} & (\mathbf{0})_{u\times u} & \dots & (\mathbf{0})_{u\times u} \\ (\mathbf{0})_{u\times u} & (\mathbf{E}^2)_{u\times u} & \dots & (\mathbf{0})_{u\times u} \\ \vdots & \vdots & \vdots & \vdots \\ (\mathbf{0})_{u\times u} & (\mathbf{0})_{u\times u} & \dots & (\mathbf{E}^k)_{u\times u} \end{bmatrix}_{d\times d} . \begin{bmatrix} (\mathbf{E}^1)_{u\times u} & (\mathbf{0})_{u\times u} & \dots & (\mathbf{0})_{u\times u} \\ (\mathbf{0})_{u\times u} & (\mathbf{E}^2)_{u\times u} & \dots & (\mathbf{0})_{u\times u} \\ \vdots & \vdots & \vdots & \vdots \\ (\mathbf{0})_{u\times u} & (\mathbf{0})_{u\times u} & \dots & (\mathbf{E}^k)_{u\times u} \end{bmatrix}^T_{d\times d}$$

$$\Rightarrow \mathbf{V}_{d\times d}.\mathbf{V}^T_{d\times d} =$$

$$\begin{bmatrix} (\mathbf{E}^1)_{u\times u}.(\mathbf{E}^1)^T_{u\times u} & (\mathbf{0})_{u\times u} & \dots & (\mathbf{0})_{u\times u} \\ (\mathbf{0})_{u\times u} & (\mathbf{E}^2)_{u\times u}.(\mathbf{E}^2)^T_{u\times u} & \dots & (\mathbf{0})_{u\times u} \\ \vdots & \vdots & \vdots & \vdots \\ (\mathbf{0})_{u\times u} & (\mathbf{0})_{u\times u} & \dots & (\mathbf{E}^k)_{u\times u}.(\mathbf{E}^k)^T_{u\times u} \end{bmatrix}_{d\times d}$$

$$\Rightarrow \mathbf{V}_{d\times d}.\mathbf{V}^T_{d\times d} =$$

$$\begin{bmatrix} (\mathbf{I}_{u\times u}) & (\mathbf{0})_{u\times u} & \dots & (\mathbf{0})_{u\times u} \\ (\mathbf{0})_{u\times u} & (\mathbf{I}_{u\times u}) & \dots & (\mathbf{0})_{u\times u} \\ \vdots & \vdots & \vdots & \vdots \\ (\mathbf{0})_{u\times u} & (\mathbf{0})_{u\times u} & \dots & (\mathbf{I}_{u\times u}) \end{bmatrix}_{d\times d}$$ *(because* $(\mathbf{E}^j)_{u\times u}.(\mathbf{E}^j)^T_{u\times u} = \mathbf{I}_{u\times u}$ *from Defn. 6)*

$$\Rightarrow \mathbf{V}_{d\times d}.\mathbf{V}^T_{d\times d} = \mathbf{I}_{k.u\times k.u} = \mathbf{I}_{k.u}$$

$$\Rightarrow \mathbf{V}_{d\times d}.\mathbf{V}^T_{d\times d} = \mathbf{I}_d$$ *(because* $d = k.u$ *from Step-1 of section 4.2 of Chapter 4)*

Similarly we can prove $\mathbf{V}^T_{d\times d}.\mathbf{V}_{d\times d} = \mathbf{I}_{d\times d}$ *because* $(\mathbf{E}^j)^T_{u\times u}.(\mathbf{E}^j)_{u\times u} = \mathbf{I}_{u\times u}$ *from Defn. 6.*

Hence the Lemma follows.

Lemma 3 $(\mathbf{C}^g)_{k.r\times k.r} = \mathbf{V}^T_{k.r\times d}.\mathbf{C}_{d\times d}.\mathbf{V}_{d\times k.r}$. *Here,* $(\mathbf{C}^g)_{k.r\times k.r}$ *is the inter-block covariance matrix of locally-projected patterns* $(\mathbf{Y})_{N\times k.r} = \{\mathbf{Y}_1, \mathbf{Y}_2, \dots, \mathbf{Y}_N\}$ *used by FP-PCA-Type-III method;* k *is the number of sub-patterns or blocks;* r *is the number of local eigenvectors per block;* $(\mathbf{V})_{d\times k.r}$ *is the combined matrix of selected* $(k.r)$ *local eigenvectors as given in Lemma 1 and* $\mathbf{C}_{d\times d}$ *is the covariance matrix of original*

patterns, $\mathbf{X}_{N\times d} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\}$ *(See section 4.2 of Chapter 4 for terminology).*

Proof 3 *We compute the inter-block covariance matrix,* $\mathbf{C}^g$*, of the locally-projected patterns,* $\mathbf{Y} = \{\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_N\}$ *as follows (Section 4.2 of Chapter 4).*

$(\mathbf{C}^g)_{k.r\times k.r} = \frac{1}{N}.\sum_{i=1}^{N}[(\mathbf{Y}_i)_{k.r\times 1}.(\mathbf{Y}_i^T)_{1\times k.r}]$

$\Rightarrow (\mathbf{C}^g)_{k.r\times k.r} = \frac{1}{N}.\sum_{i=1}^{N}[(\mathbf{V}^T.\mathbf{X}_i).(\mathbf{V}^T.\mathbf{X}_i)^T]$ *(from Lemma 1,* $\mathbf{Y_i} = \mathbf{V^T}.\mathbf{X_i}$*))*

$\Rightarrow (\mathbf{C}^g)_{k.r\times k.r} = \frac{1}{N}.\sum_{i=1}^{N}[(\mathbf{V}^T.\mathbf{X}_i).(\mathbf{X}_i^T.\mathbf{V})]$

$\Rightarrow (\mathbf{C}^g)_{k.r\times k.r} = \mathbf{V}^T.\frac{1}{N}.\sum_{i=1}^{N}[\mathbf{X}_i.\mathbf{X}_i^T].\mathbf{V}$

$\Rightarrow (\mathbf{C}^g)_{k.r\times k.r} = \mathbf{V}^T.\mathbf{C}.\mathbf{V}$ *(where* $\mathbf{C}$ *is the covariance matrix of original data* $\mathbf{X}$*)*

Hence the Lemma follows.

Theorem 18 $\lambda_i(T3) = \lambda_i(H)$*;* $\forall i \in \{1, 2, \ldots, d\}$*, if* $r = u$ *(or equivalently if* $k.r = d$*). In other words, FP-PCA-Type-III method shows the same summarization of variance as its corresponding Holistic PCA method if the number of local eigenvectors selected per block is equal to block (sub-pattern) size (or if the total number of local eigenvectors selected is equal to pattern size). Here,* $\lambda_i(T3)$ *is the* i^{th} *largest eigenvalue obtained by FP-PCA-Type-III method;* $\lambda_i(H)$ *is the* i^{th} *largest eigenvalue obtained by Holistic PCA;* $k.r$ *is the total number of local eigenvectors selected;* d *is the pattern size;* k *is the number of blocks;* r *is the number of local eigenvectors selected per block and* u *is the block size.*

Significance : The Theorem establishes a link (relationship) between FP-PCA-Type-III PCA method (e.g. SubXPCA) and its corresponding Holistic PCA method (e.g. classical PCA).

Proof 18 *It is given that* $k.r = d$ *or* $r = u$.

Please note that all the matrices we use here (i.e. $\mathbf{G}, \mathbf{\Lambda}^g, \mathbf{V}, \mathbf{C}, \mathbf{I}$*) are of size* $d \times d$*. Let* $\mathbf{G}$ *and* $\mathbf{\Lambda}^g$ *represent eigenvector matrix and diagonal eigenvalue matrix of inter-block covariance matrix,* $\mathbf{C}^g$ *respectively. We express the eigenvalue decomposition problem (Defn. 7) of* $\mathbf{C}^g$ *as*

$(\mathbf{C}^g).\mathbf{G} = \mathbf{G}.\mathbf{\Lambda}^g$

$\Rightarrow \mathbf{G}^T.(\mathbf{C}^g).\mathbf{G} = \mathbf{\Lambda}^g$ *(because* $\mathbf{G}$ *is orthogonal (Defn. 6) i.e.* $\mathbf{G}^T.\mathbf{G} = \mathbf{G}.\mathbf{G}^T = \mathbf{I}$*)*

$\Rightarrow \mathbf{G}^T.(\mathbf{V}^T.\mathbf{C}.\mathbf{V}).\mathbf{G} = \mathbf{\Lambda}^g$ *(from Lemma 3,* $\mathbf{C}^g = \mathbf{V}^T.\mathbf{C}.\mathbf{V}$*)*

$\Rightarrow (\mathbf{G}^T.\mathbf{V}^T).\mathbf{C}.(\mathbf{V}.\mathbf{G}) = \mathbf{\Lambda}^g$

$\Rightarrow (\mathbf{V}.\mathbf{G})^T.\mathbf{C}.(\mathbf{V}.\mathbf{G}) = \mathbf{\Lambda}^g$

$\Rightarrow (\mathbf{V}.\mathbf{G}).(\mathbf{V}.\mathbf{G})^T.\mathbf{C}.(\mathbf{V}.\mathbf{G}) = (\mathbf{V}.\mathbf{G}).\mathbf{\Lambda}^g$ *(multiplying both the sides with* $(\mathbf{V}.\mathbf{G})$*)*

$\Rightarrow \mathbf{C}.(\mathbf{V}.\mathbf{G}) = (\mathbf{V}.\mathbf{G}).\mathbf{\Lambda}^g$ *[because* $(\mathbf{V}.\mathbf{G})^T.(\mathbf{V}.\mathbf{G}) = \mathbf{G}^T.(\mathbf{V}^T.\mathbf{V}).\mathbf{G} = \mathbf{G}^T.(\mathbf{I}).\mathbf{G} = \mathbf{I}$ *and* $(\mathbf{V}.\mathbf{G}).(\mathbf{V}.\mathbf{G})^T = \mathbf{V}.(\mathbf{G}.\mathbf{G}^T).\mathbf{V}^T = \mathbf{V}.\mathbf{I}.\mathbf{V}^T = \mathbf{I}$*. From Lemma 2, it is clear that* $\mathbf{V}^T.\mathbf{V} = \mathbf{V}.\mathbf{V}^T = \mathbf{I}$*]*

Now the eigenvalue problem of $\mathbf{C}^g$ *is reduced to eigenvalue problem of original covariance matrix,* $\mathbf{C}$.

Therefore $\mathbf{C}$ *has the same set of eigenvalues as* $\mathbf{C}^g$*. Hence both FP-PCA-Type-III and its corresponding Holistic PCA method show the same summarization of variance in this special case.*

Hence the Theorem follows.

Corollary 2 $(\mathbf{V}.\mathbf{G})_{d \times d}$ *produces a set of eigenvectors of Holistic PCA method. Here,* $(\mathbf{V})_{d \times d}$ *is the combined matrix of* d *local eigenvectors as given in Lemma 2;* $(\mathbf{G})_{d \times d}$ *is the set of global eigenvectors obtained by FP-PCA-Type-III method and* d *is the*

pattern size.

Proof 2 *From Theorem 18,* $\mathbf{C}.(\mathbf{V}.\mathbf{G}) = (\mathbf{V}.\mathbf{G}).\mathbf{\Lambda}^g$ *which is the eigenvalue problem of covariance matrix,* $\mathbf{C}$*, of Holistic PCA method. Therefore* $(\mathbf{V}.\mathbf{G})_{d\times d}$ *represents a set of eigenvectors of Holistic PCA method.*

Lemma 4 $\lim_{r\to u}[\mathbf{V}_{d\times k.r}.\mathbf{V}^T_{k.r\times d} = \mathbf{I}_d]$*. Here,* $\mathbf{V}_{d\times k.r}$ *is the combined matrix of* $k.r\,(\leq d)$ *local column eigenvectors of all blocks as given in Lemma 1;* u *is the block size;* k *is the number of blocks;* r *is the number of local eigenvectors per block and* $\mathbf{I}_d$ *is the* $d \times d$ *Identity matrix (See section 4.2 of Chapter 4 for terminology).*

Proof 4 *Step 1: We study the orthogonal property of* $(\mathbf{V})_{d\times k.r}$*. It is clear that* $\mathbf{V}^T.\mathbf{V} = \mathbf{I}_{k.r}$ *because the dot product of any two eigenvectors is reduced to unity. However,* $\mathbf{V}.\mathbf{V}^T \neq \mathbf{I}_d$ *because* $(\mathbf{V})_{d\times k.r}$ *may not contain all* d *local eigenvectors* $(k.r \leq d)$ *(Defns. 6 and 7) and the same is expressed as follows.*

$$(\mathbf{V}.\mathbf{V}^T)_{d\times d} = \begin{bmatrix} [\mathbf{E}^1.(\mathbf{E}^1)^T]_{u\times u} & (\mathbf{0})_{u\times u} & \dots & (\mathbf{0})_{u\times u} \\ (\mathbf{0})_{u\times u} & [\mathbf{E}^2.(\mathbf{E}^2)^T]_{u\times u} & \dots & (\mathbf{0})_{u\times u} \\ \vdots & \vdots & \vdots & \vdots \\ (\mathbf{0})_{u\times u} & (\mathbf{0})_{u\times u} & \dots & [\mathbf{E}^k.(\mathbf{E}^k)^T]_{u\times u} \end{bmatrix}_{d\times d}$$

where $[(\mathbf{E}^j)_{u\times r}.(\mathbf{E}^j)^T_{r\times u}]$ *is:*

$$[\mathbf{E}^j.(\mathbf{E}^j)^T]_{u\times u} = \begin{bmatrix} (\sum_{i=1}^r [e^j_{i_1}.e^j_{i_1}]) & (\sum_{i=1}^r [e^j_{i_1}.e^j_{i_2}]) & \dots & (\sum_{i=1}^r [e^j_{i_1}.e^j_{i_u}]) \\ (\sum_{i=1}^r [e^j_{i_2}.e^j_{i_1}]) & (\sum_{i=1}^r [e^j_{i_2}.e^j_{i_2}]) & \dots & (\sum_{i=1}^r [e^j_{i_2}.e^j_{i_u}]) \\ \vdots & \vdots & \vdots & \vdots \\ (\sum_{i=1}^r [e^j_{i_u}.e^j_{i_1}]) & (\sum_{i=1}^r [e^j_{i_u}.e^j_{i_2}]) & \dots & (\sum_{i=1}^r [e^j_{i_u}.e^j_{i_u}]) \end{bmatrix}_{u\times u} ; j \in \{1, 2, \dots, k\}$$

Step 2: (i) Consider the first-type expression of $\mathbf{E}^j.(\mathbf{E}^j)^T$ *i.e.* $(\sum_{i=1}^r [e^j_{i_q}.e^j_{i_q}]); q \in \{1, 2, \dots, u\}$ *and substitute* $r = u$ *in the expression.*

After substituting $r = u$, *we get*

$(\sum_{i=1}^{u}[e_{i_q}^j.e_{i_q}^j]); \forall q \in \{1, 2, \ldots, u\}$, *which is equal to* 1 *(from eq. 6.2 of Defn. 6).*

(ii) Next consider the second-type expression of $\mathbf{E}^j.(\mathbf{E}^j)^T$ *i.e.* $(\sum_{i=1}^{r}[e_{i_q}^j.e_{i_l}^j]); q, l \in \{1, 2, \ldots, u\}, q \neq l$ *and substitute* $r = u$ *in the expression.*

After substituting $r = u$, *we get*

$(\sum_{i=1}^{u}[e_{i_q}^j.e_{i_l}^j]); \forall q, l \in \{1, 2, \ldots, u\}, q \neq l$, *which is equal to* 0 *(from eq. 6.3 of Defn. 6)*

Thus, it is evident that the expression $(\sum_{i=1}^{r}[e_{i_q}^j.e_{i_q}^j])$ *moves closer to* 1 *as* r *increases. Similarly* $(\sum_{i=1}^{r}[e_{i_q}^j.e_{i_l}^j])$ *moves closer to* 0 *as* r *increases.*

$\Rightarrow \mathbf{E}^j.[\mathbf{E}^j]^T \rightarrow \mathbf{I}_u$ *as* $r \rightarrow u$.

Hence the Lemma follows.

Theorem 19 $\lim_{r \rightarrow u} \sum_{i=1}^{k.r}[\lambda_i(T3)] = \sum_{i=1}^{k.r}[\lambda_i(H)]$. *In other words, the summarization of variance of 'FP-PCA-Type-III method' tends to the summarization of variance of 'its corresponding Holistic PCA method' as the number of local eigenvectors selected per block tends to block (sub-pattern) size (or as the total number of local eigenvectors selected tends to pattern size). Here,* k *is the number of sub-patterns or blocks;* r *is the number of local eigenvectors per block;* u *is the block size;* $\lambda_i(T3)$ *and* $\lambda_i(H)$ *are the* i^{th} *largest eigenvalues obtained by FP-PCA-Type-III and its corresponding Holistic PCA methods respectively.*

Significance : The Theorem establishes a link (relationship) between FP-PCA-Type-III method (e.g. SubXPCA) and its corresponding Holistic PCA method (e.g. classical PCA).

Proof 19 *Please note that $r \leq u$ (Section 4.2 of Chapter 4).*

Let $(\mathbf{G})_{k.r \times k.r}$ and $(\mathbf{\Lambda}^g)_{k.r \times k.r}$ represent eigenvector matrix and diagonal eigenvalue matrix of $\mathbf{C}^g_{k.r \times k.r}$ respectively . Eigenvalue decomposition problem of $\mathbf{C}^g$ is expressed as follows (Defn. 7).

$(\mathbf{C}^g)_{k.r \times k.r}.(\mathbf{G})_{k.r \times k.r} = (\mathbf{G})_{k.r \times k.r}.(\mathbf{\Lambda^g})_{k.r \times k.r}$

$\Rightarrow (\mathbf{G}^T)_{k.r \times k.r}.(\mathbf{C}^g)_{k.r \times k.r}.\mathbf{G}_{k.r \times k.r} = (\mathbf{\Lambda}^g)_{k.r \times k.r}$*(because $\mathbf{G}$ is orthogonal (Defn. 6) i.e. $\mathbf{G}^T.\mathbf{G} = \mathbf{G}.\mathbf{G}^T = \mathbf{I}$)*

$\Rightarrow (\mathbf{G}^T)_{k.r \times k.r}.(\mathbf{V}^T_{k.r \times d}.\mathbf{C}_{d \times d}.\mathbf{V}_{d \times k.r}).\mathbf{G} = (\mathbf{\Lambda}^g)_{k.r \times k.r}$*(from Lemma 3, $\mathbf{C}^g = \mathbf{V}^T.\mathbf{C}.\mathbf{V}$)*

$\Rightarrow (\mathbf{G}^T.\mathbf{V}^T).\mathbf{C}.(\mathbf{V}.\mathbf{G}) = (\mathbf{\Lambda}^g)_{k.r \times k.r}$

$\Rightarrow (\mathbf{V}.\mathbf{G})^T.\mathbf{C}.(\mathbf{V}.\mathbf{G}) = (\mathbf{\Lambda}^g)_{k.r \times k.r}$

$\Rightarrow (\mathbf{V}.\mathbf{G}).(\mathbf{V}.\mathbf{G})^T.\mathbf{C}.(\mathbf{V}.\mathbf{G}) = (\mathbf{V}.\mathbf{G}).(\mathbf{\Lambda}^g)_{k.r \times k.r}$*(multiplying both the sides with $(\mathbf{V}.\mathbf{G})$)*

$\Rightarrow [\mathbf{V}_{d \times k.r}.\mathbf{V}^T_{k.r \times d}].\mathbf{C}.(\mathbf{V}_{d \times k.r}.(\mathbf{G})_{k.r \times k.r}) = [(\mathbf{V})_{d \times k.r}.(\mathbf{G})_{k.r \times k.r}].(\mathbf{\Lambda}^g)_{k.r \times k.r}$ *(because $\mathbf{G}$ is orthogonal)*

It is noted that $\mathbf{V}_{d \times k.r}.\mathbf{V}^T_{k.r \times d} \neq \mathbf{I}_d$ (because $\mathbf{V}_{d \times k.r}$ does not contain all d local eigenvectors), therefore the eigenvalue problem of $\mathbf{C}^g$ is not reduced to the eigenvalue problem of $\mathbf{C}$.

From Lemma 4, we know that $\mathbf{V}.\mathbf{V}^T$ tends to $\mathbf{I}_d$ as r tends to u (or $k.r$ tends to d). Please note that $k.u = d$ (Section 4.2 of Chapter 4).

$\Rightarrow \mathbf{C}.(\mathbf{V}.\mathbf{G}) \rightarrow (\mathbf{V}.\mathbf{G}).\mathbf{\Lambda}^g$ *as $r \rightarrow u$ (or $k.r \rightarrow d$).*

$\Rightarrow$ *As r tends to u, eigenvalue problem of $\mathbf{C}^g$ approaches to eigenvalue problem of $\mathbf{C}$. Hence the Theorem follows.*

Theorem 20 $\sum_{i=1}^{j}[\lambda_i(T1)] \leq \sum_{i=1}^{j}[\lambda_i(T3)]$ $\forall j \in \{1, 2, \ldots, (k.r)\}$, *where $\lambda_i(T1)$ and $\lambda_i(T3)$ are the variances (eigenvalues) summarized in i^{th} principal component by FP-*

PCA-Type-I and FP-PCA-Type-III methods respectively. In other words, FP-PCA-Type-III method shows better summarization of variance as compared to FP-PCA-Type-I method. Here, k.r is the total number of selected local eigenvectors, arranged in non-increasing order of their eigenvalues; $1 \leq k.r \leq d$*, d is the pattern size.*

Significance : The Theorem establishes the fact that the dimensionality reduction of FP-PCA-Type-I method (e.g. SubPCA) is less as compared to FP-PCA-Type-III method (e.g. SubXPCA).

Proof 20 *Case (i) when all k.r local eigenvectors belong to the same homogeneous sub-pattern (block) set,* $\mathbf{P}^j$*;* $j \in \{1, 2, \ldots, k\}$*, where k is the number of blocks. This case may arise if we select local eigenvectors based on a global threshold (for eigenvalues) across sub-patterns (blocks).*

In this case, there are no two local eigenvectors which belong to two different sub-patterns (blocks), therefore no inter-block covariances exist among them. Thus FP-PCA-Type-III method shows the same summarization of variance as FP-PCA-Type-I method.

Case (ii) when $k.r = d$ *or* $r = u$ *(i.e. when the total number of local features coincides with pattern size).*

From Theorem 18, it follows that FP-PCA-Type-III method shows the same summarization of variance as its corresponding Holistic PCA method when $k.r = d$*.*

From Theorem 16 and Defn. 8, we know that FP-PCA-Type-I method does not use inter block covariances ($\mathbf{C}^{i,j}$*) thus FP-PCA-Type-I method needs more number of*

principal components to summarize most of its variance.

FP-PCA-Type-III method coincides with its corresponding Holistic PCA method which is optimal linear scheme for summarization of variance, thus FP-PCA-Type-III method needs less number of principal components to summarize most of the variance as compared to FP-PCA-Type-I method.

Case (iii) when $2 \leq k.r \leq d-1$ *and out of* $k.r$ *local eigenvectors there exists atleast 2 eigenvectors belong to two different sub-pattern sets (blocks),* $\mathbf{P}^i$ *and* $\mathbf{P}^j$, $i \neq j$.

From the Theorem 19, we understand that FP-PCA-Type-III method tends to its corresponding Holistic PCA as $k.r$ *tends to* d. *This makes it clear that for any* $k.r\,(\geq 2)$, *FP-PCA-Type-III method summarizes most of the variance (spread across* $k.r$ *local features) in first few (*$< k.r$*) principal components, where as FP-PCA-Type-I method needs more (i.e.* $k.r$*) principal components to capture most of the variance due to ignorance of inter-block covariances or correlations (Theorems 16 and 17).*

Hence the Theorem follows.

Theorem 21 $Err(T3) = Err(H) + \Delta$, *where* $\Delta = Var(H) - Var(T3)$; $\Delta \geq 0$ *and* $\lim_{k.r \to d} [Err(T3) = Err(H)]$. *Here* $Err(T3)$ *and* $Err(H)$ *indicate the error of projected data sets (summation of last* $d-w$ *eigenvalues) by FP-PCA-Type-III and Holistic PCA methods respectively;* d *is the pattern size;* $k.r$ *is the total number of local features;* $Var(...)$ *is the summation of first* w *largest eigenvalues of eigenvectors. Here we assume that the number of principal components selected (i.e.* w*) used by FP-PCA-Type-III method and its corresponding Holistic PCA are equal.*

Proof 21 *Total Variance (TV) of the set of original patterns is given by*

$TV = Var(H) + Err(H) = Var(T3) + Err(T3)$

$\Rightarrow Err(T3) = TV - Var(T3)$

$\Rightarrow Err(T3) = Var(H) + Err(H) - Var(T3)$

$\Rightarrow Err(T3) = Err(H) + [Var(H) - Var(T3)]$

$\Rightarrow Err(T3) = Err(H) + \Delta$

$\Rightarrow Err(T3) = Err(H) + \Delta$

From Theorem 19, we conclude that, as $k.r \rightarrow d$ *it clear that* $Var(T3) \rightarrow Var(H)$

$\Rightarrow [Var(T3) - Var(H)] \rightarrow 0$, *as* $k.r \rightarrow d$.

$\Rightarrow \Delta \rightarrow 0$, *as* $k.r \rightarrow d$.

$\Rightarrow Err(T3) \rightarrow Err(H)$, *as* $k.r \rightarrow d$ *(because* $Err(T3) = Err(H) + \Delta$*).*

Hence the Theorem follows.

Theorem 22 $\mathbf{f} = \{f_1, f_2, \ldots, f_d\}$ *represents a set of features of a pattern* $\mathbf{X}_i$*;* $i \in \{1, 2, \ldots, N\}$*. and there are* $d!$ *(! indicates factorial) arrangements (orders) possible for these features for any pattern* $\mathbf{X}_i$*;* $i \in \{1, 2, \ldots, N\}$*. Let* $\mathbf{F} = \{\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_{d!}\}$ *be the set of possible feature orders.*

It follows that $\psi(T3) < \psi(T1)$ $\forall \mathbf{F}_t, \mathbf{F}_q \in \mathbf{F}$; $t \neq q$*, where* $\psi(T3) = |Var^{\mathbf{F}_t}(T3) - Var^{\mathbf{F}_q}(T3)|$ *and* $\psi(T1) = |Var^{\mathbf{F}_t}(T1) - Var^{\mathbf{F}_q}(T1)|$*. In other words, FP-PCA-Type-III method shows relatively less feature order dependence as compared to FP-PCA-Type-I method.* $Var^{\mathbf{F}_v}(...)$ *is the variance summarized by first* w *principal components for the feature order* $\mathbf{F}_v$*.*

Proof 22 *Outline of proof: First we prove that FP-PCA-Type-III method shows the same summarization of variance as its corresponding Holistic PCA method in a special case for different feature orders (Step 1). Next, we show how the effect of feature*

orders on FP-PCA-Type-III method is reduced as the number of local features moves closer to block size (Step 2). Further we show the dependency of FP-PCA-Type-I method on different feature orders (Step 3).

Let each pattern be partitioned into $k\,(\geq 2)$ *sub-patterns (blocks), which makes (i)* $\mathbf{F}_t$ *to split into* $\mathbf{F}_t^1, \mathbf{F}_t^2, \ldots, \mathbf{F}_t^k$ *and (ii)* $\mathbf{F}_q$ *to split into* $\mathbf{F}_q^1, \mathbf{F}_q^2, \ldots, \mathbf{F}_q^k$*. For any feature order,* $\mathbf{F}_t$*, let* $(\mathbf{C}^{\mathbf{F}_t})^g$ *and* $(\mathbf{\Lambda}^{\mathbf{F}_t})^g$ *represent covariance matrix and diagonal eigenvalue matrix respectively obtained by FP-PCA-Type-III method. Similarly,* $(\mathbf{C}^{\mathbf{F}_t})$ *and* $(\mathbf{\Lambda}^{\mathbf{F}_t})$ *represent covariance matrix and diagonal eigenvalue matrix respectively obtained by its corresponding Holistic PCA method.*

Step 1: From Theorem 18, we know that FP-PCA-Type-III method coincides with corresponding Holistic PCA method when the number of local features (r*) is equal to block size (*u*).*

From Theorem 18, the eigenvalue decomposition problem of $(\mathbf{C}^{\mathbf{F}_t})^g$ *is reduced to eigenvalue decomposition of* $\mathbf{C}^{\mathbf{F}_t}$ *when* $r = u$ *and is given by*

$(\alpha^{\mathbf{F}_t})^T.\mathbf{C}^{\mathbf{F}_t}.(\alpha^{\mathbf{F}_t}) = (\mathbf{\Lambda}^{\mathbf{F}_t})^g$ *where* $\alpha^{\mathbf{F}_t} = \mathbf{V}^{\mathbf{F}_t}.\mathbf{G}^{\mathbf{F}_t}$*, the eigenvectors of* $\mathbf{C}^{\mathbf{F}_t}$ *for feature order* $\mathbf{F}_t$*.*

Similarly for any other feature order, $\mathbf{F}_q$*,* $(\alpha^{\mathbf{F}_q})^T.\mathbf{C}^{\mathbf{F}_q}.(\alpha^{\mathbf{F}_q}) = (\mathbf{\Lambda}^{\mathbf{F}_q})^g$*, where* $(\alpha^{\mathbf{F}_q})$ *represent eigenvectors of* $\mathbf{C}^{\mathbf{F}_q}$ *for feature order* $\mathbf{F}_q$*.*

It is well known that Holistic PCA method is independent of feature orders because the covariance structure is same with such different orderings (and same characteristic equation thus same eigenvalues). Therefore for any two feature arrangements (orders), $\mathbf{F}_t$ *and* $\mathbf{F}_q$*,* $t \neq q$

$(\alpha^{\mathbf{F}_t})^T.\mathbf{C}^{\mathbf{F}_t}.(\alpha^{\mathbf{F}_t}) = (\alpha^{\mathbf{F}_q}).\mathbf{C}^{\mathbf{F}_q}.(\alpha^{\mathbf{F}_q})$

$\Rightarrow (\mathbf{\Lambda}^{\mathbf{F}_t})^g = (\mathbf{\Lambda}^{\mathbf{F}_q})^g$

Hence FP-PCA-Type-III method is independent of feature orders in this special case.

Step 2: From Theorem 19 we know that, as r tends to u, FP-PCA-Type-III method tends to its corresponding Holistic PCA method in terms of summarization of variance. It is well known that Holistic PCA method is independent of feature orders. This leads to the conclusion that FP-PCA-Type-III method tends to be less feature order dependent as r tends to u. Next, we show feature order dependency of FP-PCA-Type-I method.

Step 3: case(i): when $\mathbf{F}_t^j = \mathbf{F}_q^j$; $\forall j = 1, 2, \ldots, k; t \neq q$. Here '$=$' indicates set equality.

This is a trivial case. In this case both FP-PCA-Type-I and FP-PCA-Type-III method are independent of feature orders, $\mathbf{F}_t$ and $\mathbf{F}_q$.

case (ii): when $\mathbf{F}_t^s \neq \mathbf{F}_q^s$; $\forall s = 1, 2, \ldots, j$; $j = 1, 2, \ldots, k$; $t \neq q$. Here '$\neq$' is a set operator.

$\Rightarrow \mathbf{Cov}(\mathbf{F}_t^s) \neq \mathbf{Cov}(\mathbf{F}_q^s)$; $\forall s = 1, 2, \ldots, j$; $j = 1, 2, \ldots, k$; $t \neq q$ *(because covariance structure vary with different set of features) where $\mathbf{Cov}(\mathbf{F}_v)$ is the covariance structure of the data with respect to a feature order, $\mathbf{F}_v$.*

$\Rightarrow \mathbf{\Lambda}(\mathbf{F}_t^s) \neq \mathbf{\Lambda}(\mathbf{F}_q^s)$; $\forall s = 1, 2, \ldots, j$; $j = 1, 2, \ldots, k$; $t \neq q$, *where $\mathbf{\Lambda}(\mathbf{F}_v)$ is the eigenvalues obtained from $\mathbf{Cov}(\mathbf{F}_v)$. Hence FP-PCA-Type-I method is dependent of feature orders, where as FP-PCA-Type-III method becomes increasingly independent of feature orders as r tends u.*

Hence the theorem follows.

Theorem 23 *FP-PCA-Type-III method is relatively more independent of sub-pattern (block) size or number of blocks as compared to FP-PCA-Type-I method.*

Proof 23 *From Theorem 18, we know that FP-PCA-Type-III method coincides with its corresponding Holistic PCA method irrespective of sub-pattern (block) size, if the number of local features per block is equal to sub-pattern (block) size, that is* $r = u$. *In addition, from Theorem 19, FP-PCA-Type-III method approaches its Holistic PCA method as the number of local features* (r) *approaches sub-pattern (block) size* (u), *irrespective of sub-pattern (block) size. Therefore, FP-PCA-Type-III method becomes more and more independent of sub-pattern (block) size as* r *approaches* u.
In contrast, FP-PCA-Type-I method gives different local features from sub-patterns with variety of block sizes of the data, because the data with different block sizes (i.e. with different values of u) *gives different subsets of original features altogether. In addition, FP-PCA-Type-I method does not use inter-block covariances or correlations (Theorem 16). Thus FP-PCA-Type-I method gives local features which are more sensitive to block-wise (sub-pattern-wise) original features. In contrast to FP-PCA-Type-I method, FP-PCA-Type-III method combines block-sensitive local features using inter-block dependencies (covariances or correlations) to give global features, which are robust to blocks with varying sizes. Observe that FP-PCA-Type-III method reduces the feature-sensitivity with different block sizes (i.e. with varying number of original features in blocks) as* r *approaches* u *(Theorem 19).*

Theorem 24 *For an FP-PCA-Type-II method,* $Var(\mathbf{Y}_i^1) = Var(\mathbf{Y}_i^2) = \ldots = Var(\mathbf{Y}_i^k) = \sum_{s=1}^{r}[\lambda_s^m]$; $\forall\, i \in \{1, 2, \ldots, N\}$. *Here,* k *is the number of sub-patterns for each pat-*

tern, $\mathbf{X_i}$*; N is the number of patterns;* $\mathbf{Y}_i^j$ *is the locally-reduced form of sub-pattern (block),* $\mathbf{X}_i^j$*;* $Var(\mathbf{Y}_i^j)$ *is the total variance of* j^{th} *locally-reduced sub-pattern;* λ_s^m *is the eigenvalue corresponding to the eigenvector* $\mathbf{e}_s^m$*.*

Proof 24 *From Defn. 9, it is known that FP-PCA-Type-II method projects each sub-pattern (block) onto the same set of first* $r\,(< u)$ *eigenvectors, of heterogeneous sub-pattern set,* $\mathbf{Q}$*. The total variance summarized by these* r *eigenvectors is* $\sum_{s=1}^{r}[\lambda_s^m]$*. where* λ_s^m *is the eigenvalue corresponding to the eigenvector* $\mathbf{e}_s^m$*. Therefore, the same variance equivalent to* $\sum_{s=1}^{r}[\lambda_s^m]$ *is embedded into the projected sub-patterns.*

It is to be noted that FP-PCA-Type-II method embeds the same amount of variance in each of reduced sub-patterns (blocks) and does not make use of inter-subpattern correlations (Theorem 25). Therefore FP-PCA-Type-II method extracts more local features, which are more likely to be correlated. The redundant features of FP-PCA-Type-II method are eliminated by FP-PCA-Type-IV method.

The following properties can be proved in a similar fashion as we proved until now.

Theorem 25 *FP-PCA-Type-II method is based on* $\mathbf{C_Q}$ *and ignores the inter-block covariances. Here,* $\mathbf{C_Q}$ *is the intra-block covariance matrix of all-subpatterns set.*

Proof 25 *The proof is similar to the Theorem 16.*

Theorem 26 *FP-PCA-Type-IV method uses* $\mathbf{C_Q}$ *and makes use of the inter-block covariances as well. Here,* $\mathbf{C_Q}$ *is the intra-block covariance matrix of all-subpatterns set.*

Proof 26 *The proof is similar to the Theorem 17.*

Theorem 27 $\lambda_i(T4) = \lambda_i(H); \forall i \in \{1, 2, \ldots, d\}$, *if* $r = u$ *(or equivalently if* $k.r = d$*). In other words, FP-PCA-Type-IV method shows the same summarization of variance as its corresponding Holistic PCA method if* $r = u$ *(equivalently if* $k.r = d$*). Here,* $\lambda_i(T4)$ *is the* i^{th} *largest eigenvalue obtained by FP-PCA-Type-IV method,* $\lambda_i(H)$ *is the* i^{th} *largest eigenvalue obtained by Holistic PCA;* $k.r$ *is the total number of local eigenvectors;* d *is the pattern size;* r *is the number of local eigenvectors selected per block and* u *is the block size.*

Proof 27 *The proof is similar to the Theorem 18.*

Theorem 28 $\lim_{k.r \to d} \sum_{i=1}^{k.r} [\lambda_i(T4)] = \sum_{i=1}^{k.r} [\lambda_i(H)]$. *In other words, the summarization of variance of 'FP-PCA-Type-IV' tends to the summarization of variance of 'its corresponding Holistic PCA method' as* r *tends to* u *(equivalently as* $k.r$ *tends to* d*). Here,* r *is the number of local eigenvectors selected per block;* u *is the block size;* $\lambda_i(T4)$ *and* $\lambda_i(H)$ *are the* i^{th} *largest eigenvalues obtained by FP-PCA-Type-IV and Holistic PCA methods respectively.*

Proof 28 *The proof is similar to the Theorem 19.*

Theorem 29 $\mathbf{f} = \{f_1, f_2, \ldots, f_d\}$ *represents a set of features of a pattern* $\mathbf{X}_i$*;* $i \in \{1, 2, \ldots, N\}$. *and there are* $d!$ *(! indicates factorial) arrangements (orders) possible for these features for any pattern* $\mathbf{X}_i$*;* $i \in \{1, 2, \ldots, N\}$. *Let* $\mathbf{F} = \{\mathbf{F}_1, \mathbf{F}_2, \ldots, \mathbf{F}_{d!}\}$ *be all possible feature orders.*

It follows that $\psi(T4) < \psi(T2) \, \forall \, \mathbf{F}_t, \mathbf{F}_q \in \mathbf{F}; t \neq q$, *where* $\psi(T4) = |Var^{\mathbf{F}_t}(T4) -$

$Var^{\mathbf{F}_q}(T4)|$ *and* $\psi(T2) = |Var^{\mathbf{F}_t}(T2) - Var^{\mathbf{F}_q}(T2)|$. *In other words, FP-PCA-Type-IV method shows relatively less feature order dependence as compared to FP-PCA-Type-II method.* $Var^{\mathbf{F}_v}(...)$ *is the variance summarized by first* w *principal components for the feature order* $\mathbf{F}_v$.

Proof 29 *The proof is similar to the Theorem 22.*

Theorem 30 *FP-PCA-Type-IV method is relatively more independent of sub-pattern (block) size as compared to FP-PCA-Type-II method.*

Proof 30 *The proof is similar to the Theorem 23.*

In a nutshell, we summarize the important properties of FP-PCA methods proposed in this section.

- FP-PCA-Type-I (e.g. SubPCA [21]) and FP-PCA-Type-II (e.g. modPCA[53]) methods make use of covariance structure of features belongs to individual sub-patterns or blocks, but not inter-block covariance structure, which enables the methods to perform local feature extraction. However, the limited use of covariance structure results in *more number of principal components (low dimensionality reduction).*

- FP-PCA-Type-III methods (e.g. SubXPCA (Chapter 4) and FLPCA (Chapter 5)) make use of covariance structure of features belongs to individual sub-patterns or blocks, and also inter-block covariance structure, which enables the methods to perform local feature extraction and also global feature extraction. In addition, the use of more comprehensive covariance structure results in *less number of principal components (high dimensionality reduction).*

- FP-PCA-Type-III methods (e.g. SubXPCA) *move closer* to their corresponding Holistic PCA methods (e.g. classical PCA) as the number of local features are increased and *both coincide if the total number of local features selected is equal to the size of pattern*.

- FP-PCA-Type-III methods show relatively *more feature order independence* as compared to FP-PCA-Type-I and FP-PCA-Type-II methods as the number of local features are increased.

- FP-PCA-Type-III methods show relatively *more block size independence* as compared to FP-PCA-Type-I and FP-PCA-Type-II methods as the number of local features are increased.

- FP-PCA-Type-IV methods share most of the properties of FP-PCA-Type-III methods.

6.4 Experimental Results and Analysis

In this section, we demonstrate some important properties proposed in the previous section through our experimentation on (i) a publicly available database from UCI repository of Machine Learning [165] and (ii) ORL face data set [119]. In our experiments, we take SubPCA [21], SubXPCA (Chapter 4) and classical PCA (Section 1.3 of Chapter 1) as representatives of FP-PCA-Type-I, FP-PCA-Type-III and Holistic PCA respectively.

6.4.1 UCI Waveform Data

Waveform data [165] comprises of 5000 patterns. Each pattern of waveform data is of 21 dimensions, belongs to one of 3 classes with labels $(0, 1, 2)$. We consider randomly generated 250 patterns from each class (a total of 750 patterns), for computing principal components.

We choose the number of blocks (sub-patterns), k, as 3 and 7 for our experiments. We show summarization variance for $k = 3$ in Figs. 6.1-6.7 and for $k = 7$ in Figs. 6.8-6.10.

6.4.2 ORL Face Data

ORL face data set [119] contains face images of 40 persons, 10 images per person amounting to 400 images in total. Each image is of dimension, 112×92 (PGM format). Images are with variation in lighting, facial expressions and with/without glasses. We use 5 images per person generated randomly (a total of 200 images) for computing principal components.

We used C language built-in procedures, viz. $tqli, tredt, eigensrt$, to find eigenvectors, eigenvalues and for sorting them [127].

6.4.3 Discussion

Experiment 1:

(a) For UCI Waveform data: We consider the number of blocks, $k = 3$; block size, $u = 7$. We consider the number of local eigenvectors varying from $r = 1$ to $r = 7 = u$ from each block. For each of the set of local eigenvectors, r, we show the summariza-

tion of variance by SubPCA (FP-PCA-Type-I) and SubXPCA (FP-PCA-Type-III) in Figs. 6.1-6.7. For comparison we also plot PCA's (Holistic PCA) summarization of variance in the figures. From all the figures, it is made clear that summarization of variance by SubPCA (FP-PCA-Type-I) method is quite less as compared to SubXPCA (FP-PCA-Type-III) and classical PCA (Holistic PCA) (**Theorem 20**). Although, SubXPCA (FP-PCA-Type-III) shows low summarization of variance as compared to classical PCA (Holistic PCA) initially (in this case $r = 1$) (Fig. 6.1), also note that SubXPCA (FP-PCA-Type-III) moves closer to classical PCA (Holistic PCA) as r increases (Figs. 6.2-6.6) and SubXPCA (FP-PCA-Type-III) coincides with classical PCA (Holistic PCA) when $r = u = 7$ (Fig. 6.7). The same result was theoretically proved in **Theorem 18** and **Theorem 19**.

(b) For ORL face data set: We consider the number of blocks, $k = 92$; block size, $u = 112$. We consider the number of local eigenvectors $r = 1$, $r = 3$, $r = 5$ and $r = 10$ for each block. For each of the set of local eigenvectors, r, we show the summarization of variances by SubPCA (FP-PCA-Type-I) and SubXPCA (FP-PCA-Type-III) in Figs. 6.14-6.17. We plot the Fig. 6.14 as described: we select a total of 92 local eigenvectors (i.e. one eigenvector from each of 92 blocks) for SubPCA (FP-PCA-Type-I) and SubXPCA (FP-PCA-Type-III) methods and their summarization of variances are plotted. Similarly we plot the Figs. 6.15-6.17 as described below: we select initially 3, 5 and 10 eigenvectors from each of 92 blocks amounting to a total of 276 PCs (i.e. 3×92), 460 PCs (i.e. 5×92) and 920 PCs (i.e. 10×92) respectively. Further, out of these total local eigenvectors, we consider only top 200 eigenvectors and their sum-

marization of variances are plotted in the respective figures. For comparison, we also plot PCA's (Holistic PCA) summarization of variance in the first 200 principal components (PCs). Proportion of variances summarized by SubPCA (FP-PCA-Type-I) and SubXPCA (FP-PCA-Type-III) are computed with respect to the summation of first 200 eigenvalues obtained by classical PCA (Holistic PCA). From these figures, it is clear that the summarization of variance by SubPCA (FP-PCA-Type-I) method is quite less as compared to SubXPCA (FP-PCA-Type-III) and classical PCA (Holistic PCA) (**Theorem 20**). Although, SubXPCA (FP-PCA-Type-III) shows low summarization of variance as compared to classical PCA (Holistic PCA) initially (in this case $r = 1$) (Fig. 6.14), also note that SubXPCA (FP-PCA-Type-III) moves closer to classical PCA (Holistic PCA) as r increases (Fig. 6.17). The same result was theoretically proved in **Theorem 18** and **Theorem 19**.

Experiment 2:

For UCI Waveform data: We consider the number of blocks, $k = 7$; the block size, $u = 3$. We consider the number of local eigenvectors varying from $r = 1$ to $r = 3 = u$. For each of set of local eigenvectors, r, we show the summarization of variance by SubPCA (FP-PCA-Type-I) and SubXPCA (FP-PCA-Type-III) in Figs. 6.8-6.10. For comparison we also plot the PCA's (Holistic PCA) summarization of variance in the figures. From these figures, it is clear that summarization of variance by SubPCA (FP-PCA-Type-I) method is quite less as compared to SubXPCA (FP-PCA-Type-III) and classical PCA (Holistic PCA) (**Theorem 20**). Although, SubXPCA (FP-PCA-Type-III) shows low summarization of variance as compared to classical PCA

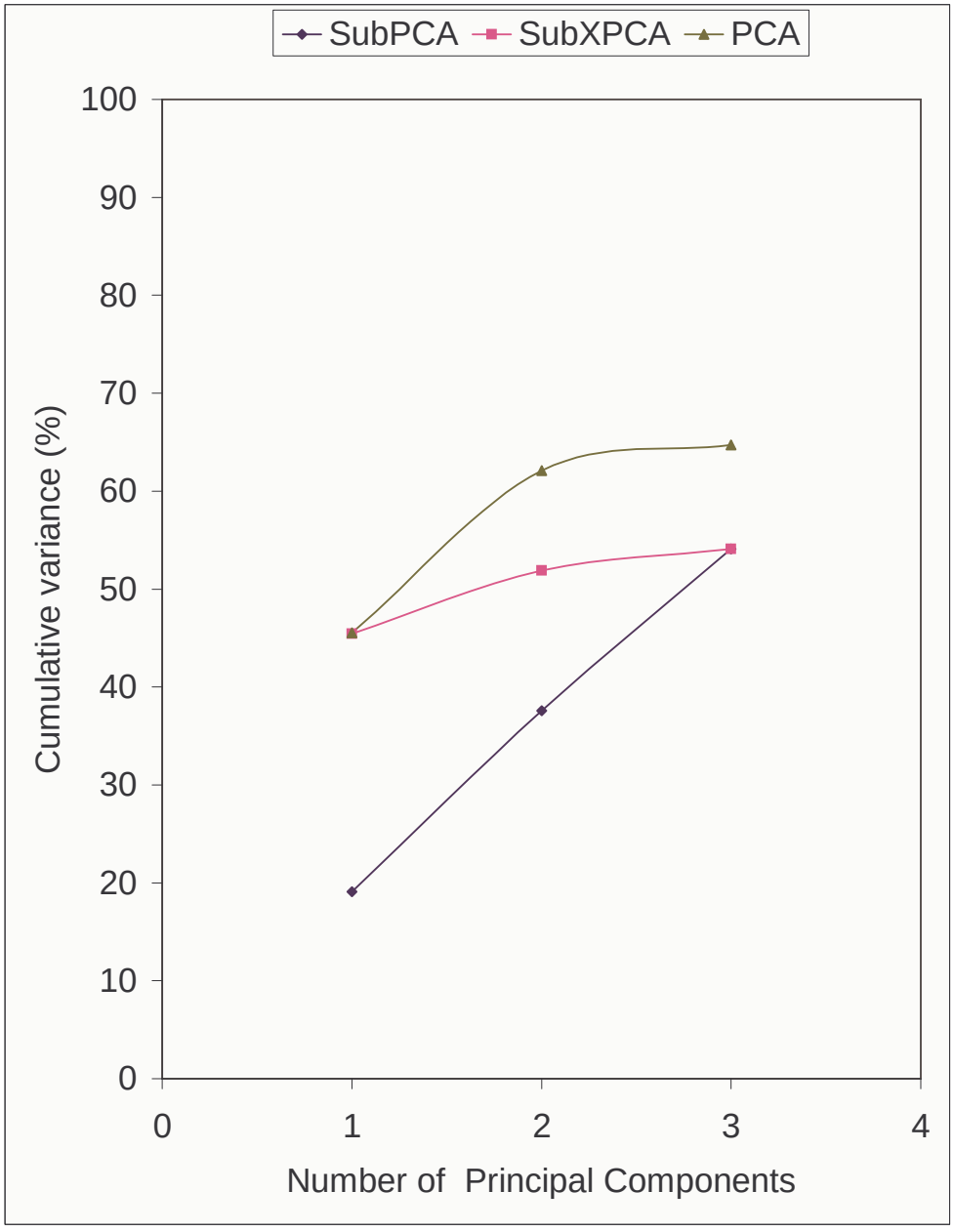


Figure 6.1: *Summarization of variance in first 3 local principal components (i.e. 1 PC per block) for Waveform data.* Each pattern is divided into 3 blocks. Please note that SubXPCA (FP-PCA-Type-III) shows better summarization of variance as compared to the summarization of variance by SubPCA (FP-PCA-Type-I).

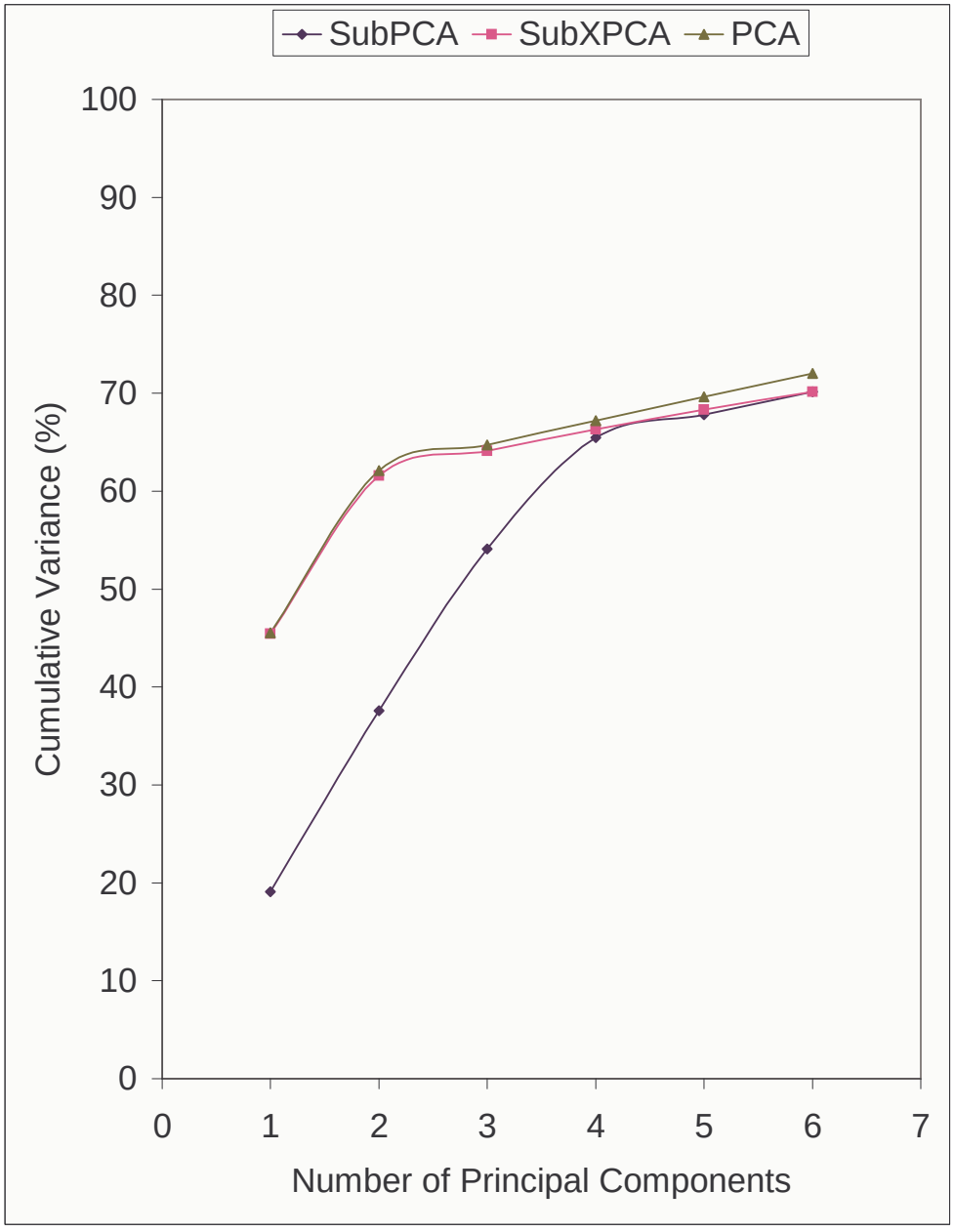


Figure 6.2: *Summarization of variance in first 6 local principal components (i.e. 2 PCs per block) for Waveform data.* Each pattern is divided into 3 blocks. Please note that SubXPCA (FP-PCA-Type-III) moves closer to PCA's (Holistic PCA) summarization of variance as the number of PCs increases (Compare with Fig. 6.1). SubPCA (FP-PCA-Type-I) does not move closer to PCA's (Holistic PCA) summarization of variance.

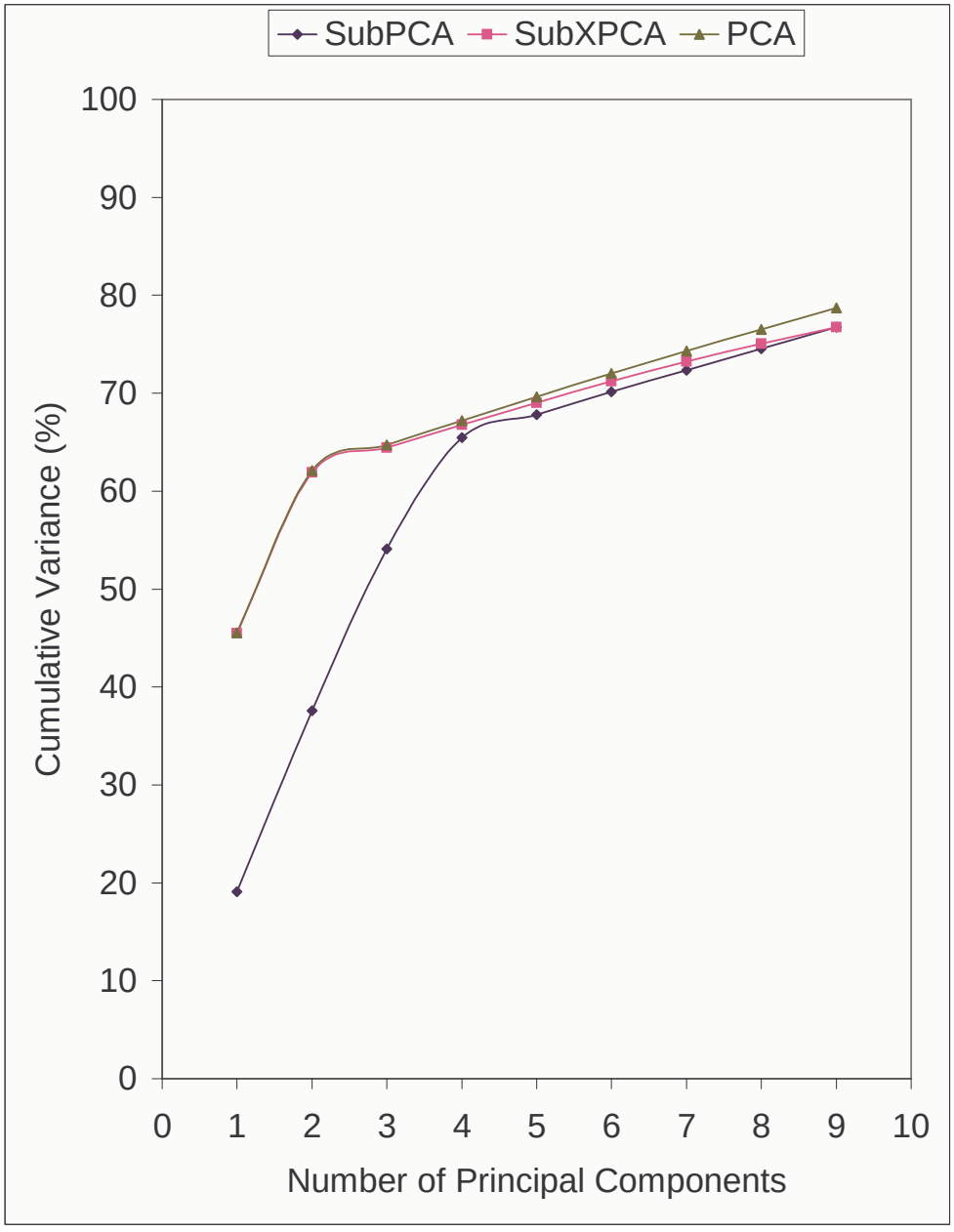


Figure 6.3: *Summarization of variance in first 9 local principal components (i.e. 3 PCs per block) for Waveform data.* Each pattern is divided into 3 blocks. Please note that SubXPCA (FP-PCA-Type-III) moves closer to PCA's (Holistic PCA) summarization of variance as the number of PCs increases (Compare with Figs. 6.1-6.2). SubPCA (FP-PCA-Type-I) does not move closer to PCA's (Holistic PCA) summarization of variance.

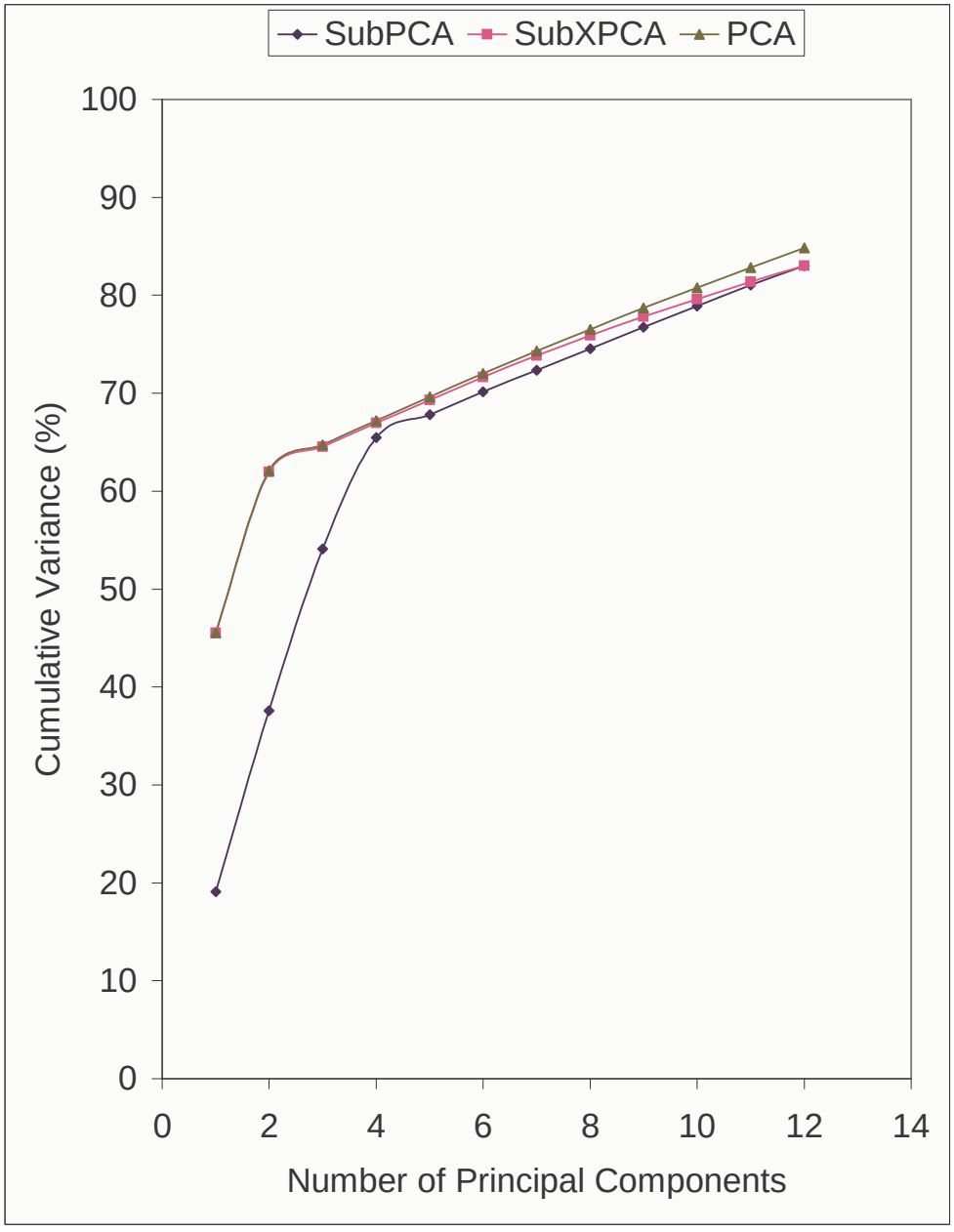


Figure 6.4: *Summarization of variance in first 12 local principal components (i.e. 4 PCs per block) for Waveform data.* Each pattern is divided into 3 blocks. Please note that SubXPCA (FP-PCA-Type-III) moves closer to PCA's (Holistic PCA) summarization of variance as the number of PCs increases (Compare this figure with Figs. 6.1-6.3). SubPCA (FP-PCA-Type-I) does not move closer to PCA's (Holistic PCA) summarization of variance.

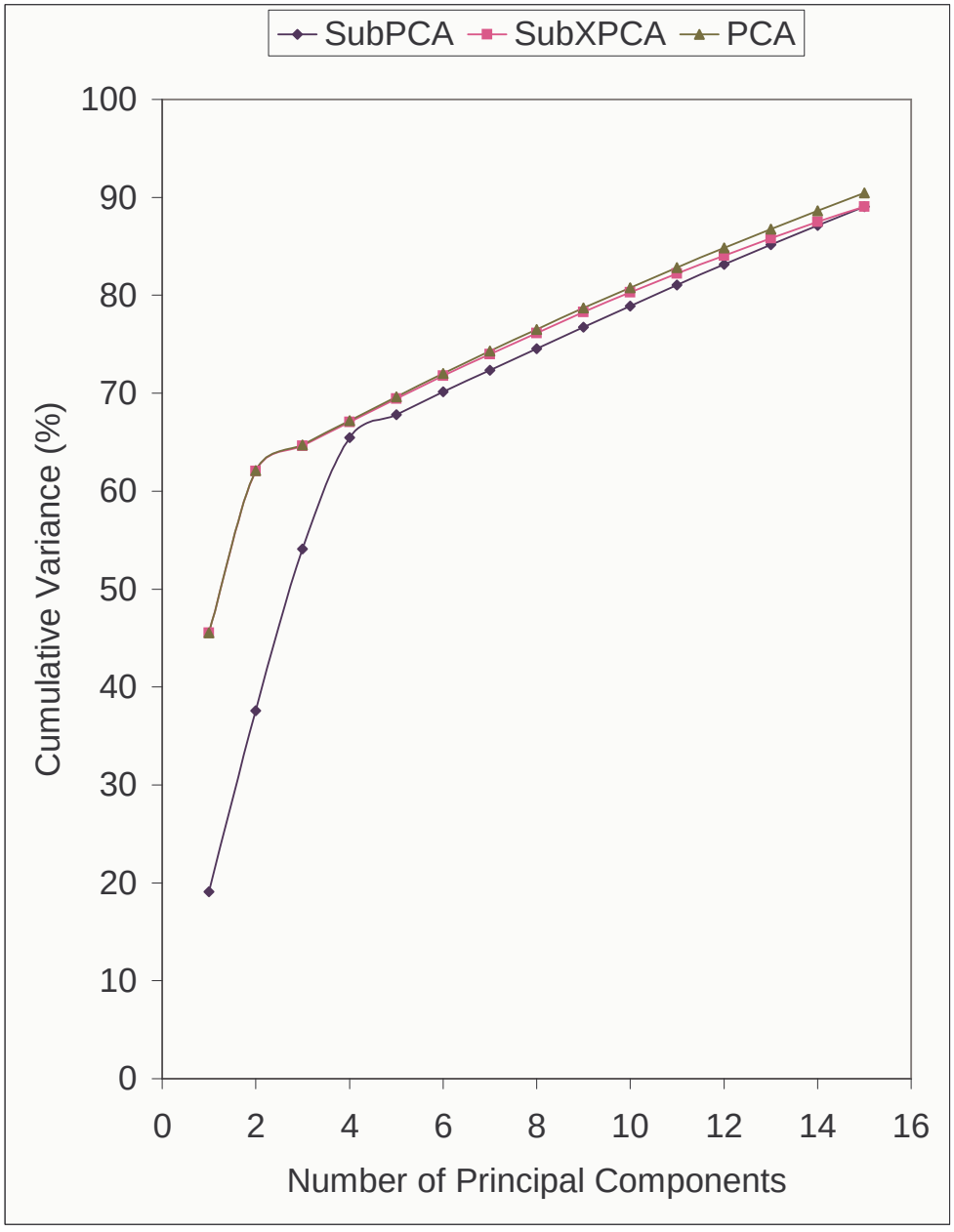


Figure 6.5: *Summarization of variance in first 15 local principal components (i.e. 5 PCs per block) for Waveform data.* Each pattern is divided into 3 blocks. Please note that SubXPCA (FP-PCA-Type-III) moves closer to PCA's (Holistic PCA) summarization of variance as the number of PCs increases (Compare this figure with Figs. 6.1-6.4). SubPCA (FP-PCA-Type-I) does not move closer to PCA's (Holistic PCA) summarization of variance.

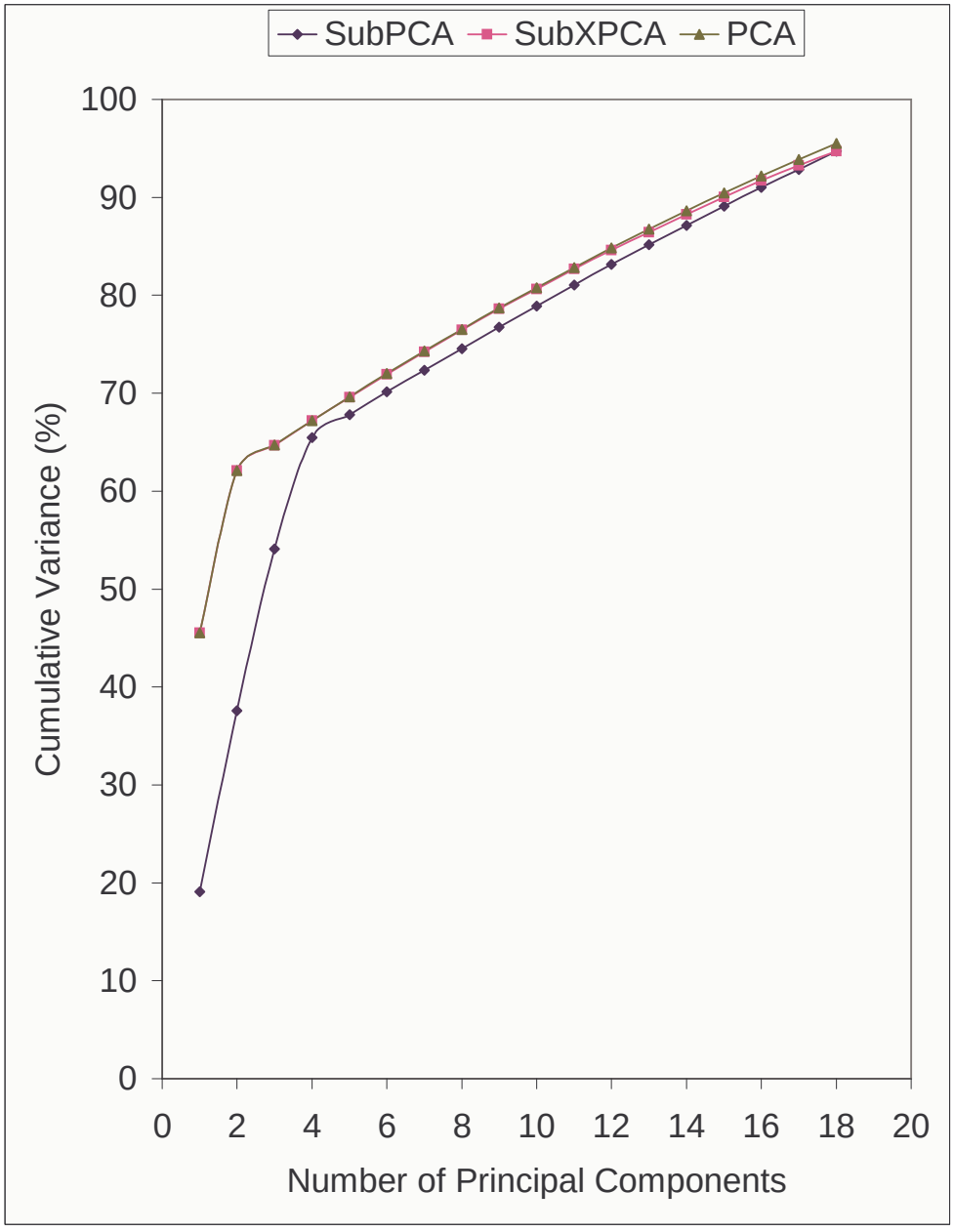


Figure 6.6: *Summarization of variance in first 18 local principal components (i.e. 6 PCs per block) for Waveform data.* Each pattern is divided into 3 blocks. Please note that SubXPCA (FP-PCA-Type-III) moves closer to PCA's (Holistic PCA) summarization of variance as the number of PCs increases (Compare this figure with Figs. 6.1-6.5). SubPCA (FP-PCA-Type-I) does not move closer to PCA's (Holistic PCA) summarization of variance.

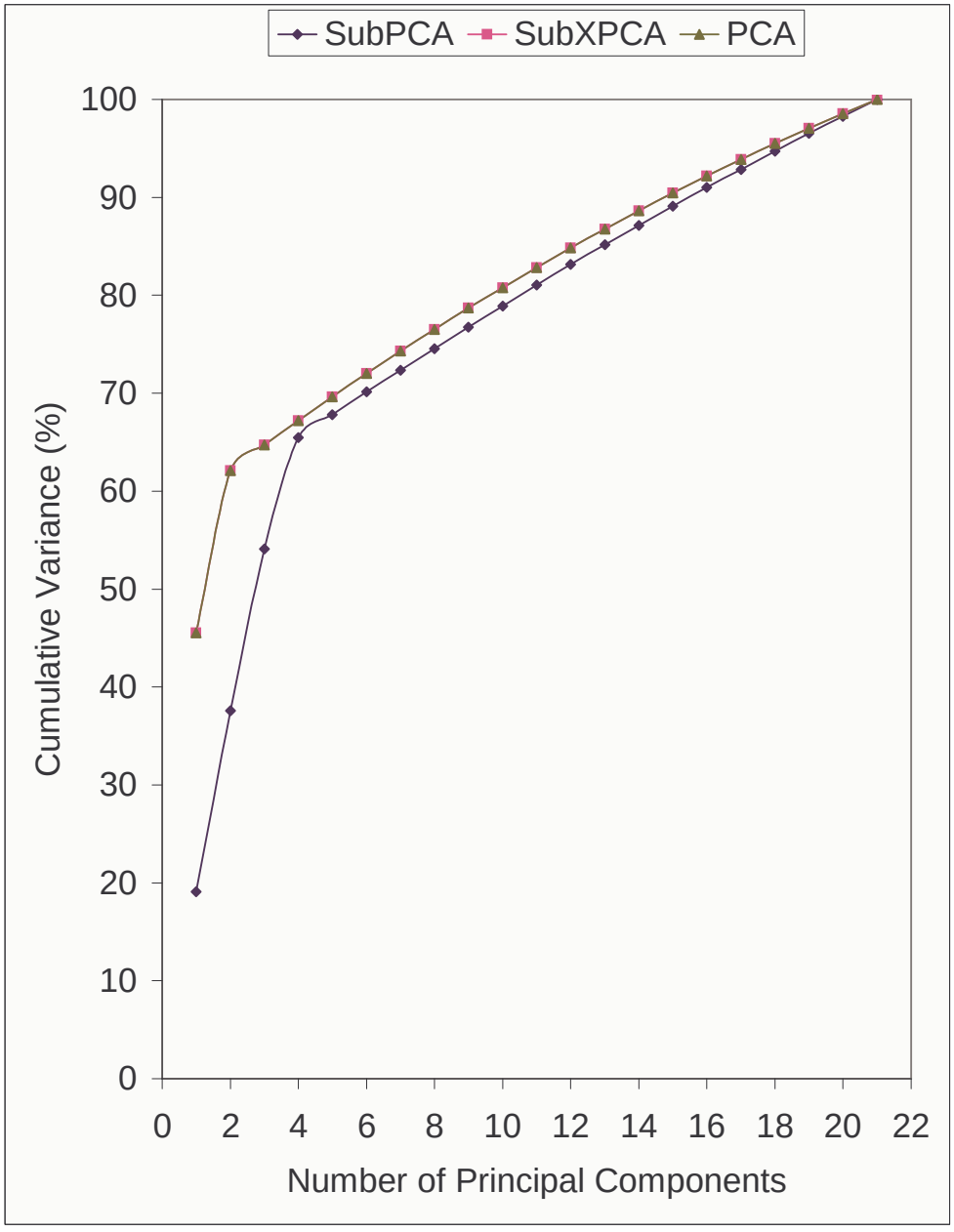


Figure 6.7: *Summarization of variance in first 21 local principal components (i.e. 7 PCs per block) for Waveform data.* Each pattern is divided into 3 blocks. Please note that SubXPCA (FP-PCA-Type-III) coincides with PCA's (Holistic PCA) summarization of variance (Compare this figure with Figs. 6.1-6.6). SubPCA (FP-PCA-Type-I) does not coincide with PCA's (Holistic PCA) summarization of variance.

(Holistic PCA) initially (in this case $r = 1$) (Fig. 6.8), also note that SubXPCA (FP-PCA-Type-III) moves closer to classical PCA (Holistic PCA) as r increases and SubXPCA coincides with classical PCA (Holistic PCA) when $r = u = 3$ (Fig. 6.10). The same result was theoretically proved in **Theorem 18** and **Theorem 19**.

Experiment 3:

For UCI Waveform data: We consider the number of blocks, $k = 3$. We use 5 different feature orders (Original order and 4 randomly generated feature orders). We plot the cumulative variance of principal components of classical PCA (Holistic PCA), SubXPCA (FP-PCA-Type-III) and SubPCA (FP-PCA-Type-I) in Figs. 6.11-6.13 respectively. It is observed from these figures that the variances summarized by classical PCA (Holistic PCA) and SubXPCA (FP-PCA-Type-III) methods are closer to each other for various feature orders. However, SubPCA (FP-PCA-Type-I) shows relatively significant differences among variances summarized for various feature orders as compared to classical PCA (Holistic PCA) and SubXPCA (FP-PCA-Type-III) (Fig. 6.12). One more fact is that SubXPCA (FP-PCA-Type-III) is very close to PCA's (Holistic PCA) summarization of variance for suitable number of principal components (For our experimentation we take 15 PCs). Although theoretically PCA (i.e. Holistic PCA) is expected to be invariant to feature orders, we observe small variations in variances for different feature orders in our experiments perhaps due to round-off errors and machine numerical computations. This experiment demonstrates **Theorem 22** that FP-PCA-Type-III method (SubXPCA) is relatively more independent (robust) of feature orders as compared to FP-PCA-Type-I method (SubPCA).

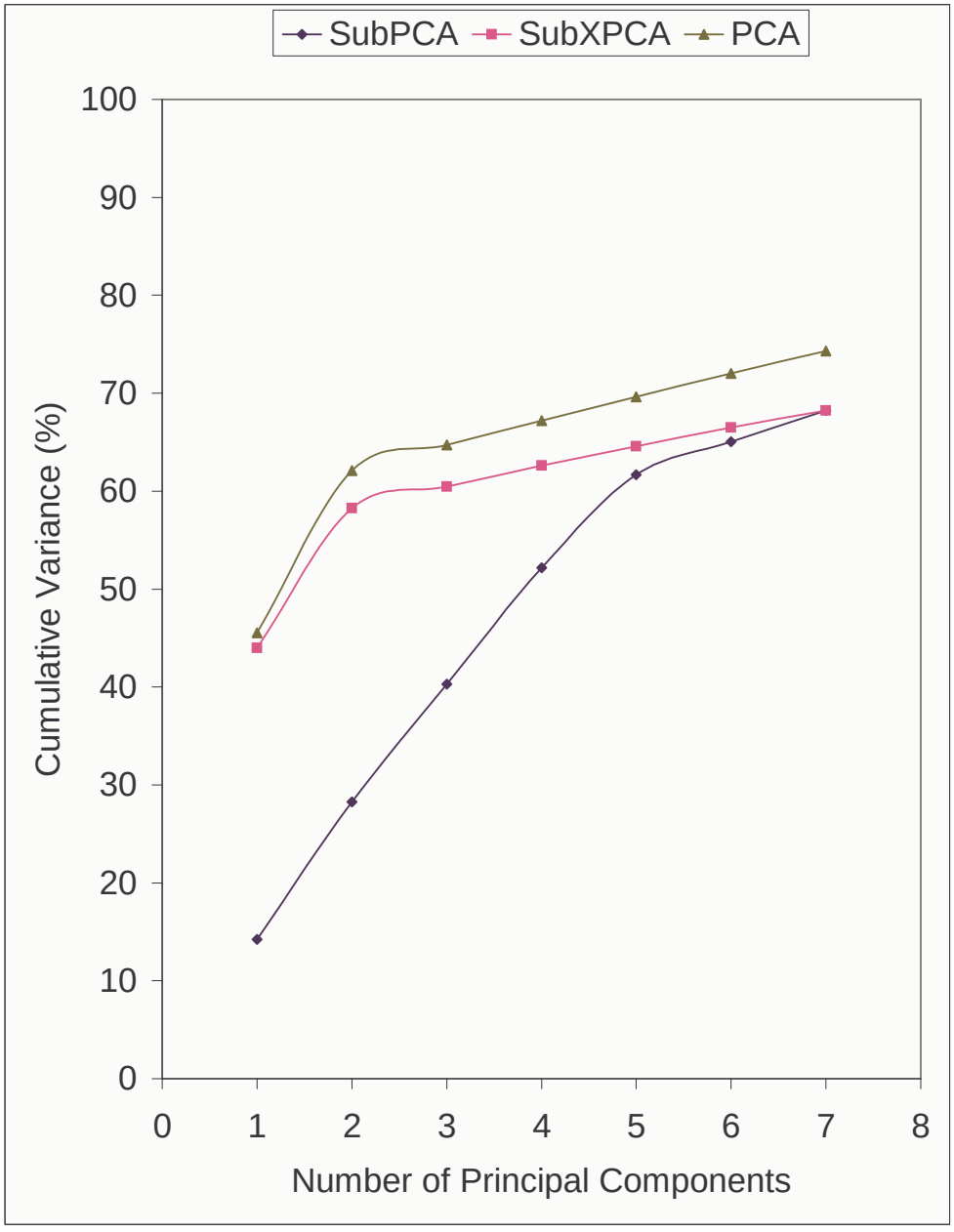


Figure 6.8: *Summarization of variance in first 7 local principal components (i.e. 1 PC per block) for Waveform data with 7 blocks per pattern.* Please note that SubXPCA's (FP-PCA-Type-III) summarization of variance is better than SubPCA's (FP-PCA-Type-I) summarization of variance.

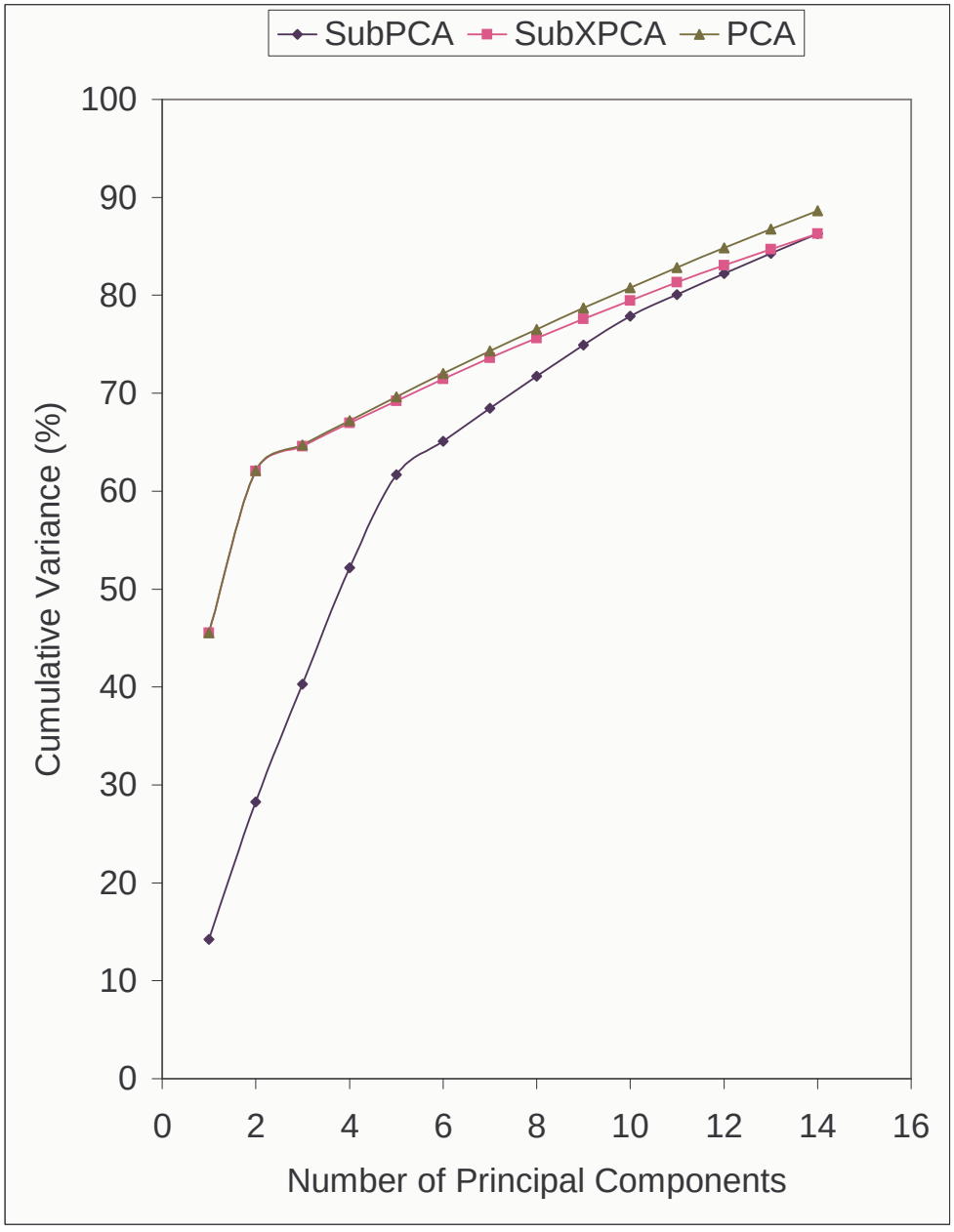


Figure 6.9: *Summarization of variance in first 14 local principal components (i.e. 2 PCs per block) for Waveform data with 7 blocks per pattern.* Please note that SubXPCA (FP-PCA-Type-III) moves closer to PCA's (Holistic PCA) summarization of variance as the number of PCs increases (Compare this figure with Fig. 6.8). SubPCA (FP-PCA-Type-I) does not move closer to PCA's (Holistic PCA) summarization of variance.

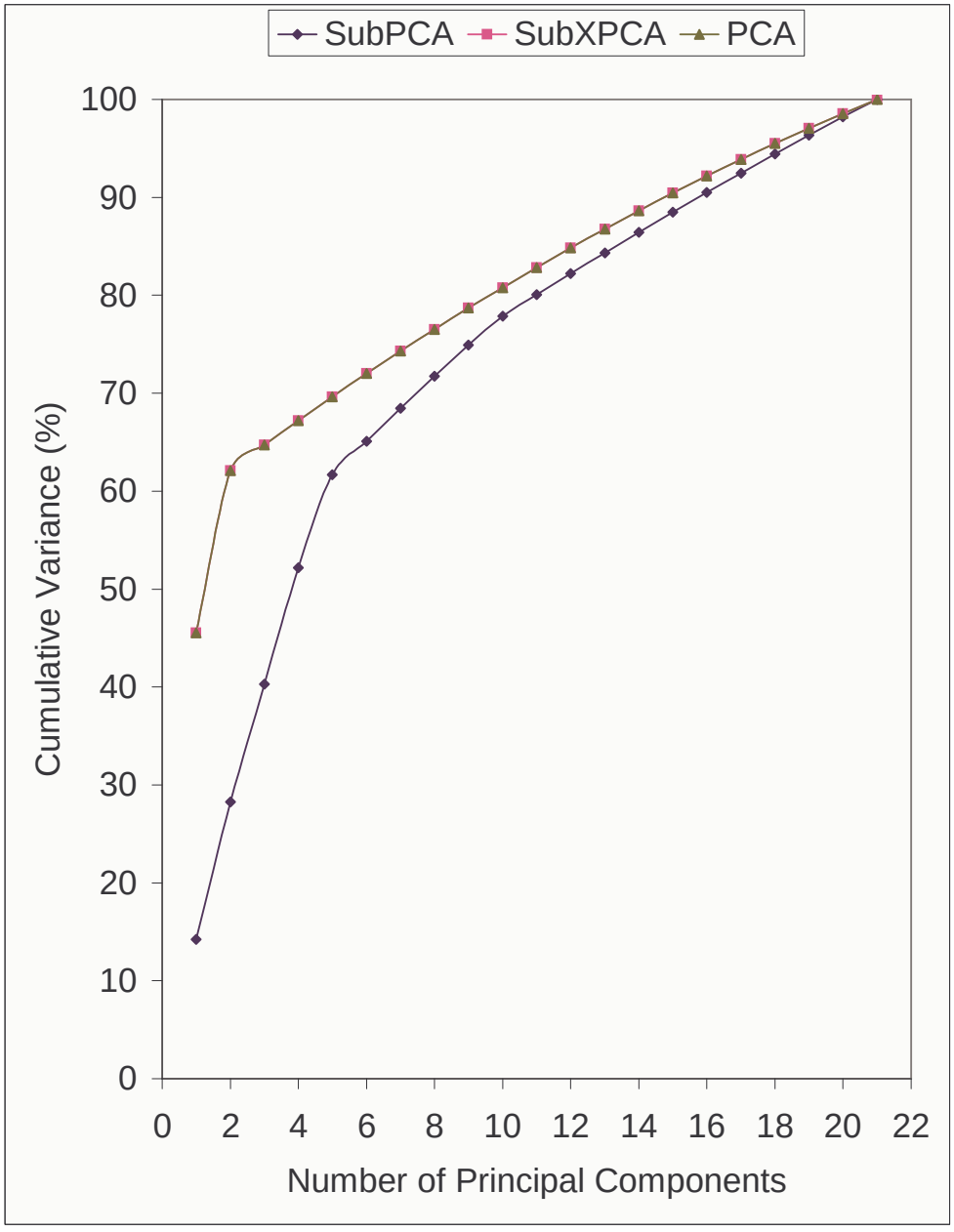


Figure 6.10: *Summarization of variance in first 21 local principal components (i.e. 3 PCs per block) for Waveform data with 7 blocks per pattern.* Please note that SubXPCA (FP-PCA-Type-III) coincides with PCA's (Holistic PCA) summarization of variance (Compare this figure with Figs. 6.8-6.9). It is clear that SubPCA (FP-PCA-Type-I) does not coincide with PCA's (Holistic PCA) summarization of variance.

Experiment 4:

(a) For UCI Waveform data: The objective of the experiment is to observe the impact of block-size (number of blocks) on SubPCA (FP-PCA-Type-I) and SubXPCA (FP-PCA-Type-III) methods. We consider 2 different block sizes, viz, $k = 3$ and $k = 7$. We observe that SubXPCA (FP-PCA-Type-III) shows relatively better block-size independence (i.e. closer variances with different block sizes) as compared to SubPCA (FP-PCA-Type-I) with increased number of principal components (Figs. 6.2-6.7 and Figs. 6.9-6.10), as proved in **Theorem 23**.

(b) For ORL face data set: We consider 3 different block sizes (number of blocks), viz, (i) $u = 112$, $k = 92$, (ii) $u = 56$, $k = 184$ and (iii) $u = 28$, $k = 368$ and the summarization of variance by SubXPCA (FP-PCA-Type-III) and SubPCA (FP-PCA-Type-I) in different principal components is shown in Figs. 6.18 and 6.19. We observe that SubXPCA (FP-PCA-Type-III) shows relatively better block-size independence (i.e. closer variances with different block sizes) as compared to SubPCA (FP-PCA-Type-I) with increased number of local principal components per block (Fig. 6.19), as proved in **Theorem 23**. For comparison, we also plot PCA's (Holistic PCA) summarization of variance in the first 200 principal components (PCs). Proportion of variances summarized by SubPCA (FP-PCA-Type-I) and SubXPCA (FP-PCA-Type-III) are computed with respect to the summation of first 200 eigenvalues obtained by classical PCA (Holistic PCA).

To plot Fig. 6.18, we follow the procedure as described: we choose initially (i) 368 PCs (4 PCs from each of 92 blocks), (ii) 368 PCs (2 PCs from each of 184 blocks) and

(iii) 368 PCs (1 PC from each of 368 blocks) respectively from these 3 cases. Further out of these 368 PCs, we consider only top 200 PCs for each case. Similarly, to plot Fig. 6.19 we follow the procedure as described: we choose initially (i) 460 PCs (5 PCs from each of 92 blocks), (ii) 920 PCs (5 PCs from each of 184 blocks) and 1840 PCs (5 PCs from each of 368 blocks) respectively from these 3 cases. Further out of these 460, 920 and 1840 local PCs, we consider only top 200 PCs for each case.

6.5 Summary

In this Chapter, we performed a theoretical analysis of FP-PCA methods and brought out various properties of the FP-PCA methods. We categorized the FP-PCA methods into FP-PCA-Type-I to FP-PCA-Type-IV classes. FP-PCA-Type-I and FP-PCA-Type-II class of methods do not use complete covariance information, which requires them to use more features, thus leads to lower dimensionality reduction. FP-PCA-Type-III and FP-PCA-Type-IV class of methods make use of more covariance information than FP-PCA-Type-I and FP-PCA-Type-II, thus leads to high dimensionality reduction. In addition, FP-PCA-Type-III and FP-PCA-Type-IV class of methods move closer to corresponding Holistic PCA methods (e.g. classical PCA) as the total number of local features tends to pattern size. More interestingly, FP-PCA-Type-III and FP-PCA-Type-IV methods were proved to be less sensitive to (i) different feature orders and (ii) different block sizes as well.

In the next Chapter, we apply feature partitioning framework to Cluster analysis. Cluster analysis enables us to produce clusters or groups of given set of patterns or objects. For a good clustering technique, similarity between any two patterns or

objects belong to the same cluster is expected to be high, where as similarity between patterns belong to different clusters is expected to be as less as possible.

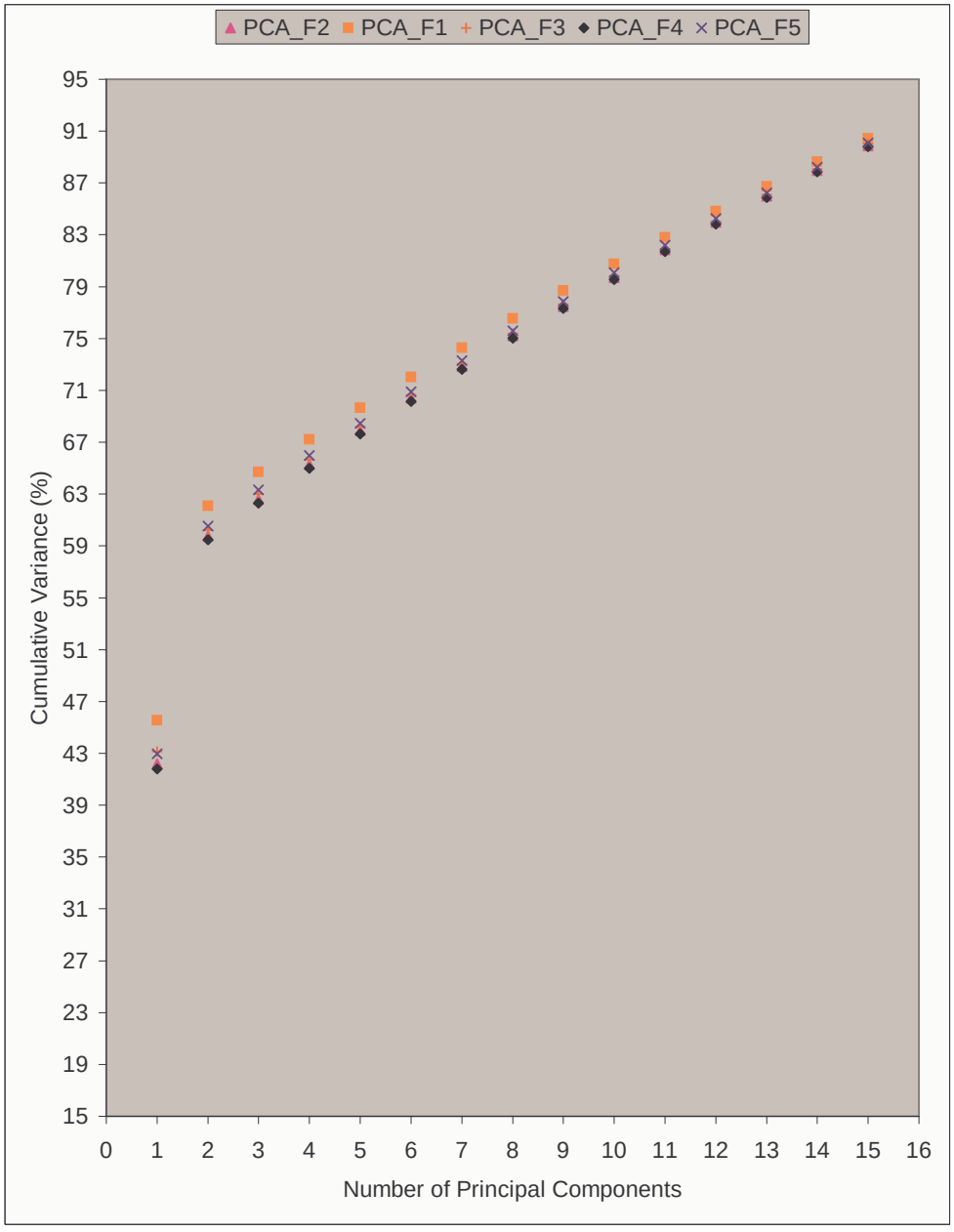


Figure 6.11: *Impact of feature orders on classical PCA (Holistic PCA).* PCA shows closer summarization of variances with varied number of PCs in Waveform data for 5 feature orders (F1, F2,..., F5), which is the indication of PCA's (Holistic PCA) more feature order independence.

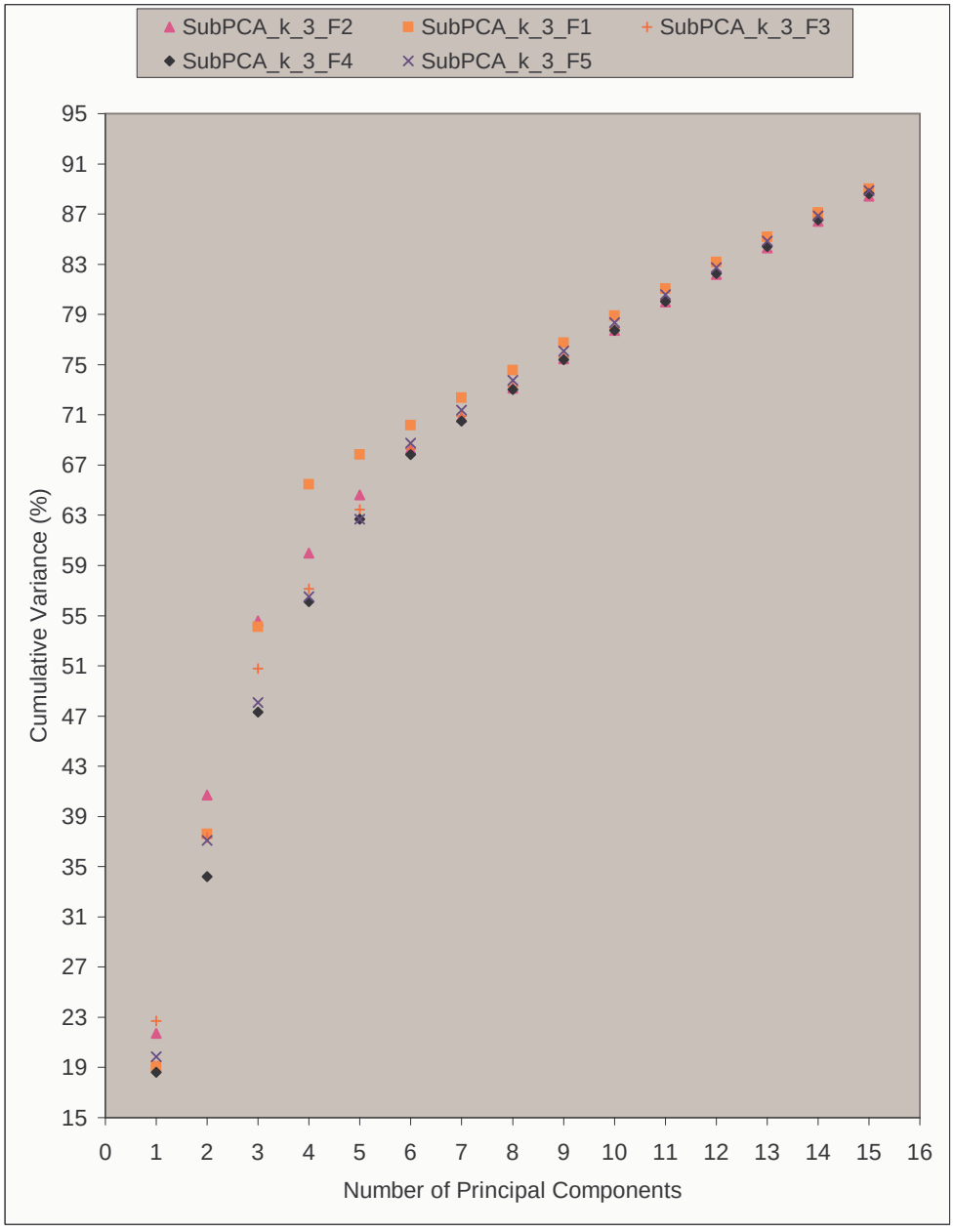


Figure 6.12: *Impact of feature orders on SubPCA (FP-PCA-Type-I).* SubPCA does not show closer summarization of variances with varied number of PCs in Waveform data for 5 feature orders (F1, F2,..., F5), which is the indication of SubPCA's (FP-PCA-Type-I) more feature order dependence. Each pattern is divided into 3 sub-patterns.

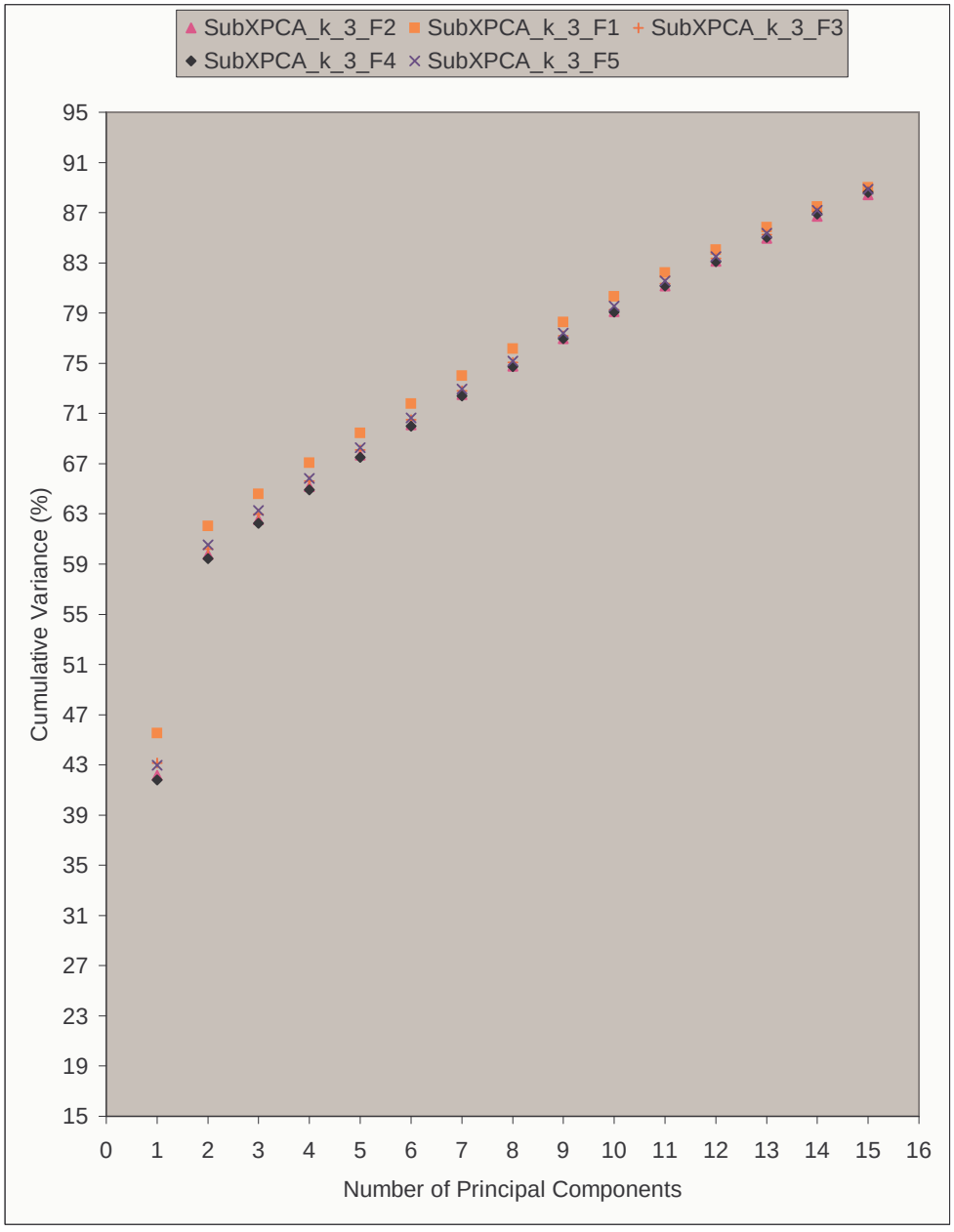


Figure 6.13: *Impact of feature orders on SubXPCA (FP-PCA-Type-III).* SubXPCA shows closer summarization of variances with varied number of PCs in Waveform data for 5 feature orders (F1, F2,..., F5), which is the indication of SubXPCA's (FP-PCA-Type-III) more feature order independence. Each pattern is divided into 3 sub-patterns.

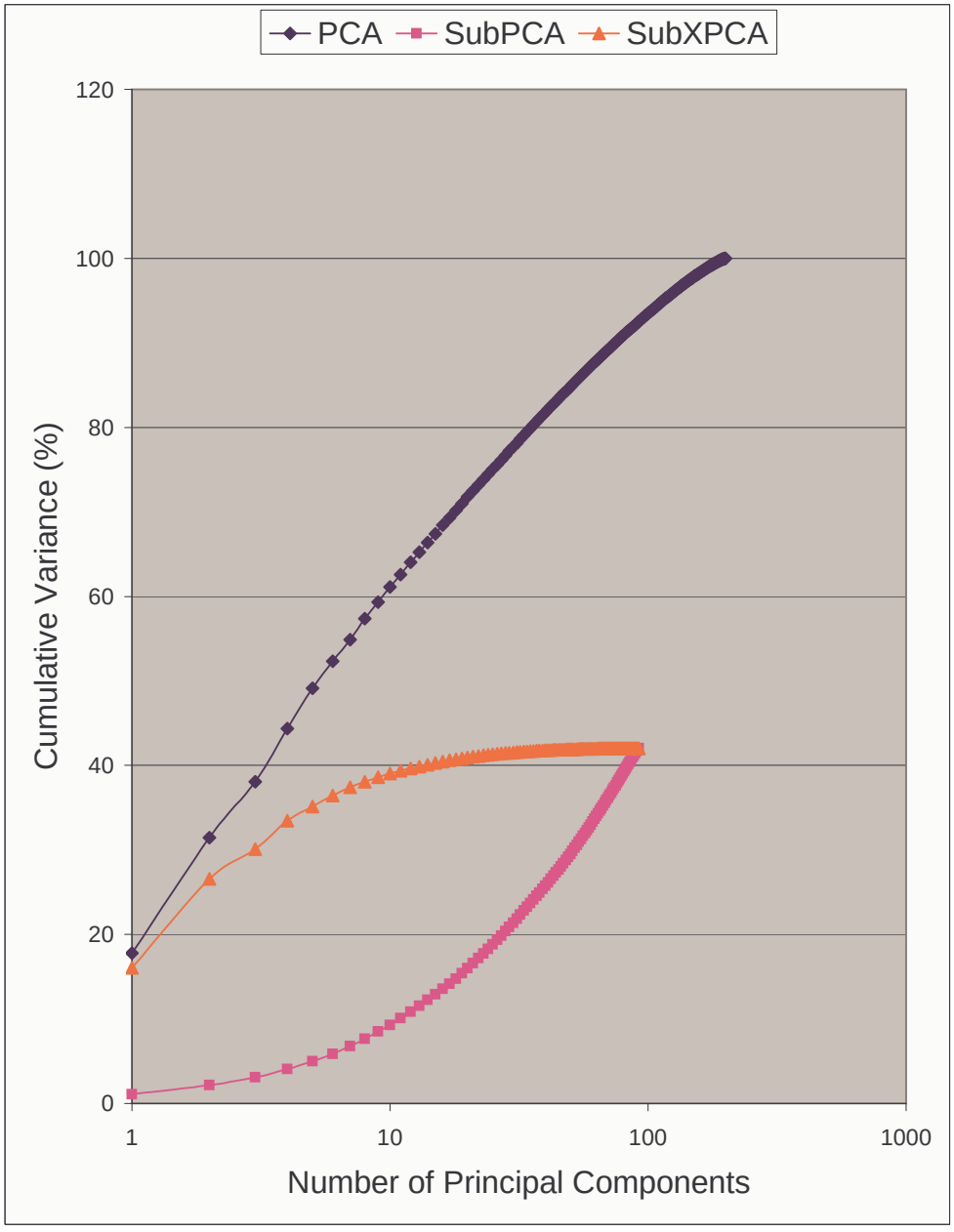


Figure 6.14: *Summarization of variance in first 92 local principal components (i.e. 1 PC per block) for ORL face data with 92 blocks per pattern.* For classical PCA (Holistic PCA), we used first 200 PCs. Please note that SubXPCA's (FP-PCA-Type-III) summarization of variance is better than SubPCA's (FP-PCA-Type-I) summarization of variance.

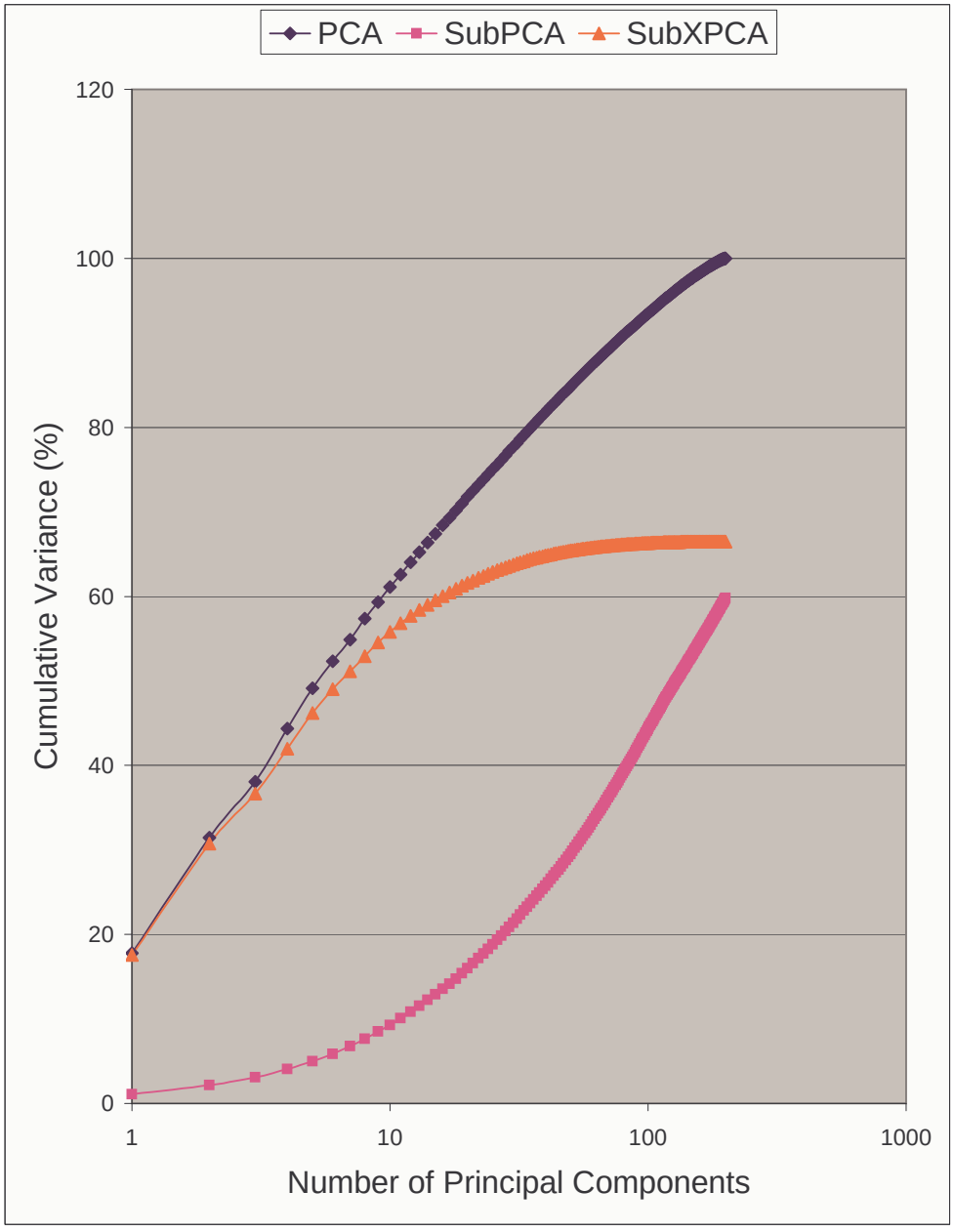


Figure 6.15: *Summarization of variance in first 200 local principal components for ORL face data with 92 blocks per pattern.* We choose initially 276 PCs (3 PCs per block). Further out of these 276 PCs, we consider only top 200 PCs. Please note that SubXPCA (FP-PCA-Type-III) moves closer to PCA's (Holistic PCA) summarization of variance as the number of local PCs per block increases (Compare this figure with Fig. 6.14). SubPCA (FP-PCA-Type-I) does not move closer to PCA's (Holistic PCA) summarization of variance.

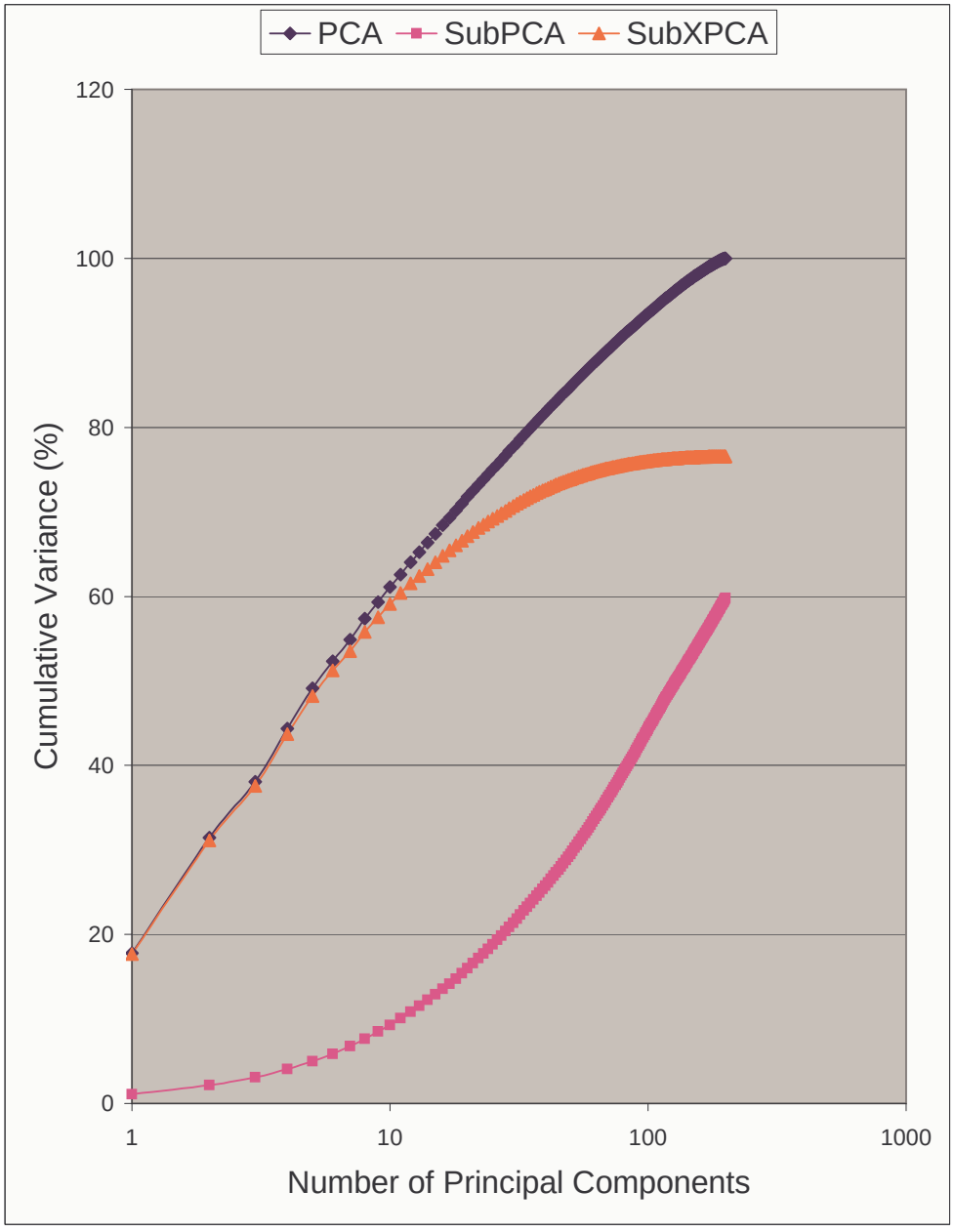


Figure 6.16: *Summarization of variance in first 200 local principal components for ORL face data with 92 blocks per pattern.* We choose initially 460 PCs (5 PCs per block). Further out of these 460 PCs, we consider only top 200 PCs. Please note that SubXPCA (FP-PCA-Type-III) moves closer to PCA's (Holistic PCA) summarization of variance as the number of local PCs per block increases (Compare this figure with Figs. 6.14-6.15). SubPCA (FP-PCA-Type-I) does not move closer to PCA's (Holistic PCA) summarization of variance.

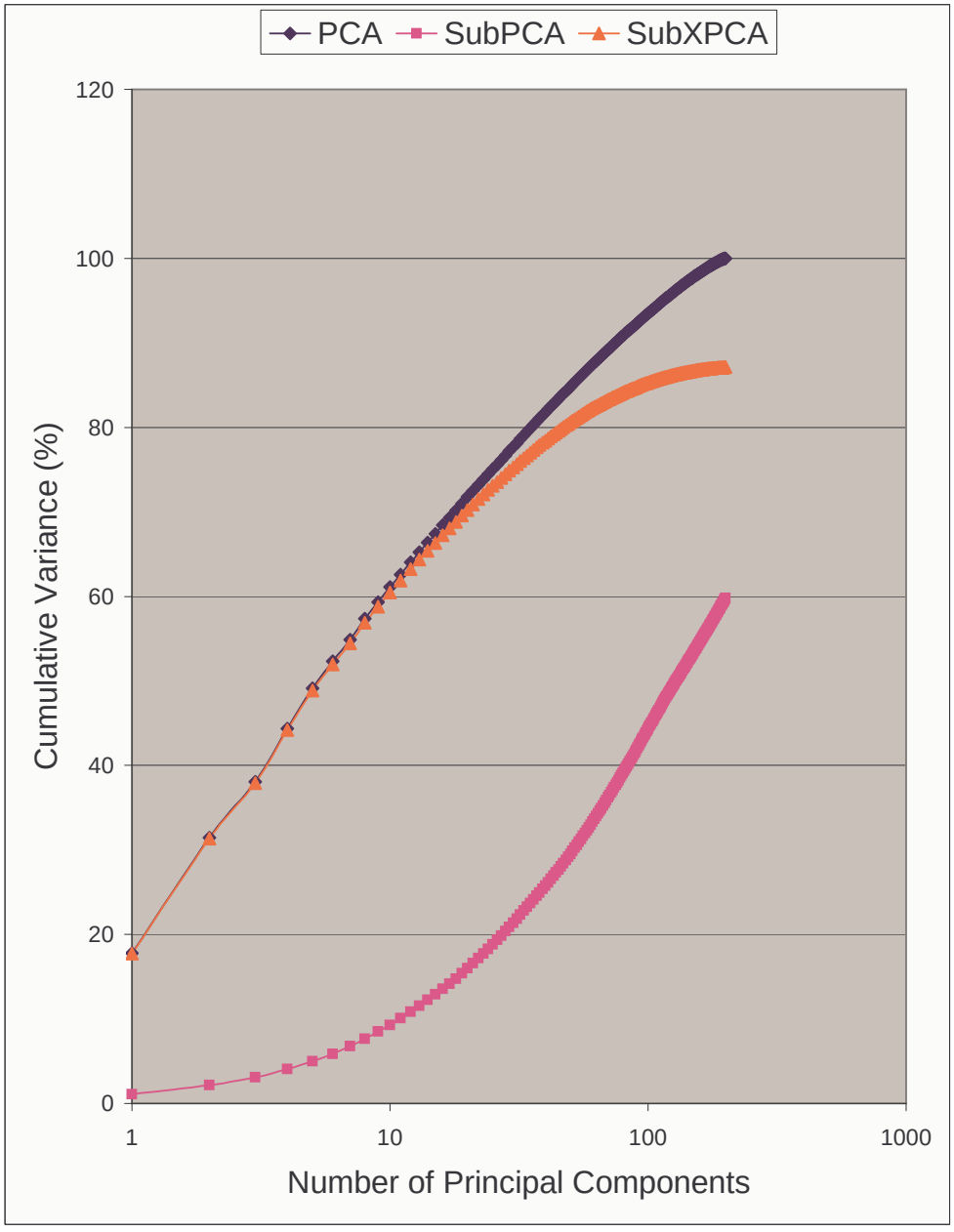


Figure 6.17: *Summarization of variance in first 200 local principal components for ORL face data with 92 blocks per pattern.* We choose initially 920 PCs (10 PCs per block). Further out of these 920 PCs, we consider only top 200 PCs. Please note that SubXPCA (FP-PCA-Type-III) moves closer to PCA's (Holistic PCA) summarization of variance with increased number of local PCs per block (Compare this figure with Figs. 6.14-6.16). SubPCA (FP-PCA-Type-I) does not move closer to PCA's (Holistic PCA) summarization of variance.

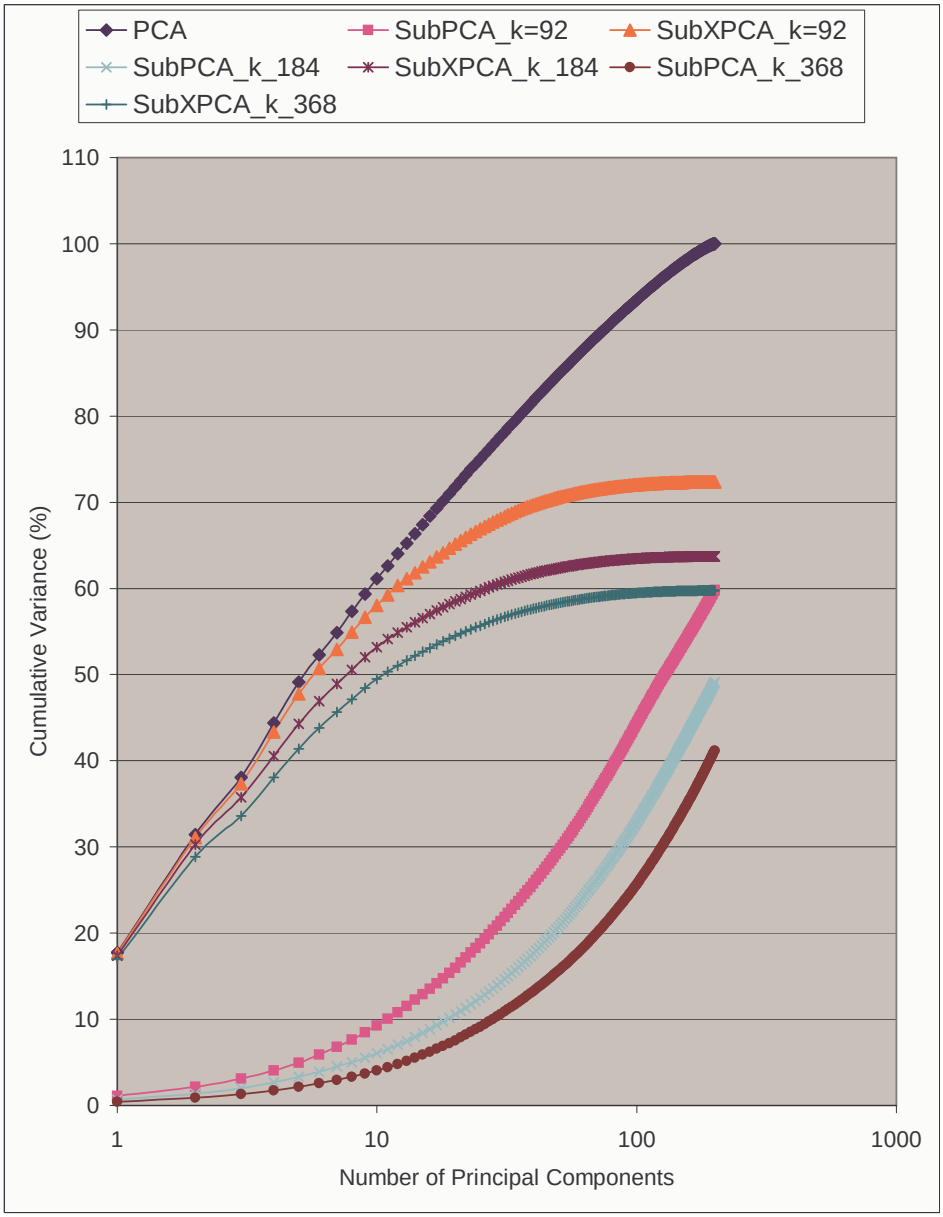


Figure 6.18: *Impact of block size on summarization of variance for ORL face data (with less number of local PCs per block).* Each pattern is divided into (i) 92, (ii) 184 and (iii) 368 blocks. We choose initially (i) 368 PCs (4 PCs from each of 92 blocks), (ii) 368 PCs (2 PCs from each of 184 blocks) and (iii) 368 PCs (1 PC from each of 368 blocks) respectively from these 3 cases. Further out of these 368 PCs, we consider only top 200 PCs for each case. It is clear that SubXPCA (FP-PCA-Type-III) shows better summarization of variance as compared to SubPCA (FP-PCA-Type-I) with different block sizes or number of blocks.

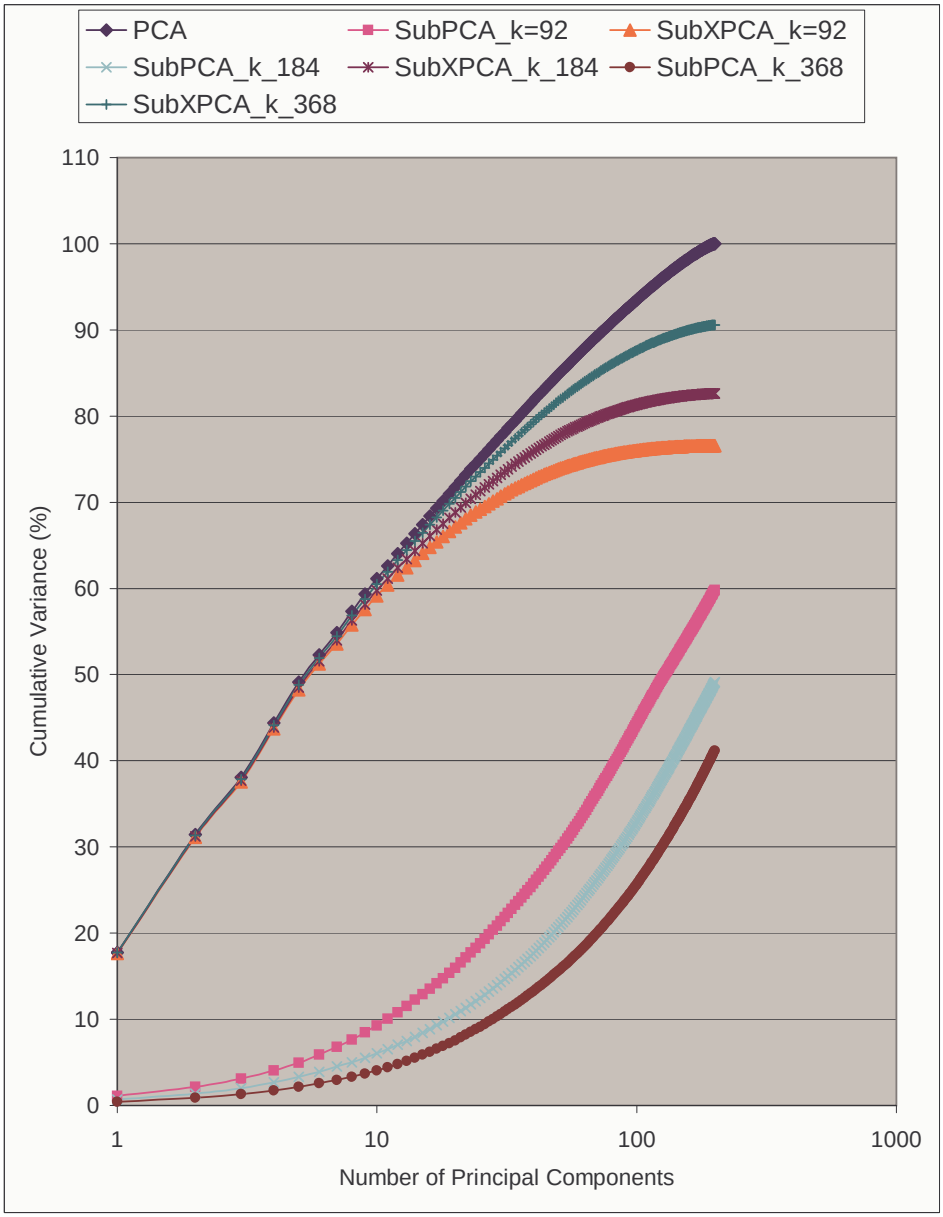


Figure 6.19: *Impact of block size on summarization of variance for ORL face data (with more local PCs per block).* Each pattern is divided into (i) 92, (ii) 184 and (iii) 368 sub-patterns. We choose initially (i) 460 PCs (5 PCs from each of 92 blocks), (ii) 920 PCs (5 PCs from each of 184 blocks) and 1840 PCs (5 PCs from each of 368 blocks) respectively from these 3 cases. Further out of these 460, 920 and 1840 local PCs, we consider only top 200 PCs for each case. SubXPCA (FP-PCA-Type-III) shows relatively better independence of block-size or number of blocks as compared to SubPCA (FP-PCA-Type-I), with increased number of local PCs per block. Compare this figure with Fig. 6.18.

Chapter 7

A Feature Partitioning Approach to Correlation Connected Cluster Analysis

7.1 Introduction

Note: An initial version of the work in this Chapter has been published in *proceedings of ICAPR-2007 international conference*[1].

Due to tremendous amount of data in the information industry, there has been an urgent need to make the voluminous data into useful information (knowledge), which is used for subsequent crucial decision making tasks. To put into words, *We are drowning in data, but starving for knowledge (information).* In this context, data

[1]Kadappagari Vijaya Kumar and Atul Negi, "An attribute partitioning approach to correlation connected clusters", *In Proceedings of International Conference on Advances in Pattern Recognition (ICAPR-2007)*, ISI Kolkata, India, pp. 93-98, Jan. 2-4th 2007.

mining is one of the promising technologies to extract highly useful, novel knowledge (golden nuggets) from heaps of data. Data mining applications include fraud detection (for e.g. in bank industry, health insurance companies), market analysis, production control, educational data mining [137] and scientific applications. More information on data mining can be found in [129] [57].

One of the primary tasks of data mining is cluster analysis. Clustering is the process of grouping the data into groups or classes or clusters in such a way that objects or patterns within a cluster have high similarity in comparison to other patterns, but are dissimilar to objects or patterns in other clusters. Dissimilarity or similarity is computed using the attribute or feature values of the objects. Clustering is an unsupervised technique which is very useful to find how objects are distributed (that is to find natural groupings) in applications where there is no prior class or category information of the objects is available. Each cluster found represents a class or category which is not known a priori. Clustering has useful applications such as market analysis (e.g. customers with similar buying habits), web-log analysis (e.g. browsing patterns of web users), structuring large documents using hierarchical clustering, thematic maps generation from satellite images. As described in a recent survey [182], clustering algorithms are classified into various categories: Hierarchical (Agglomerative, divisive), Squared error based (Vector Quantization), Graph theory based, Fuzzy based, Kernel based, Neural Network based, Large-Scale data based (DBSCAN), High-dimensional data based, etc.

In general, many clustering techniques do not consider correlation analysis while forming clusters. A recently proposed clustering technique, 'Computing Clusters of

Correlation Connected objects' (4C technique) [12] is based on density-based clustering (DBSCAN) [40] and correlation analysis (Principal Component Analysis (PCA) [76]). The 4C method [12] was proved to be useful and has very interesting applications in molecular biology, time sequences, etc. Correlation analysis finds correlations or dependencies among features of the data. One of the most useful methods to find correlations is PCA. Having the information of correlations of features, (i) we can perform dimensionality reduction which may improve data mining performance and (ii) correlations reveal hidden causal relationships in the data (For e.g. age or weight of a person is related to dosage of medicine to be prescribed; expression level of a gene may affect the expression of another gene). 4C method aims at finding correlation connected clusters. A correlation connected cluster is a local subset of data which shows strong correlations and are densely populated with respect to, a given density threshold. In this context, it is to be noted that the strong correlations may be present in some subsets of data (visible locally) because dependency between features can be different for different subgroups of data set. Such strong correlations may not be visible when entire data set is considered (not visible globally). One can visualize such correlations in Fig. 7.2. The objective of 4C method is to find subsets of data which are densely populated with strong correlations. Some applications of 4C method include (i) Recommendations systems or target marketing in Electronic Commerce, where customers of similar behaviour are detected (Positive correlations are useful here), (ii) In gene expression analysis, negative correlations indicate that if one gene shows high expression level, other shows low and vice versa. Such correlations may be hidden only in subsets of data [12].

The 4C method (4C is combination of DBSCAN and PCA) was proved to be superior to clustering methods, DBSCAN, CLIQUE, etc [12] and has interesting applications as already mentioned. However, 4C may not cope up well with *high dimensional* data because both DBSCAN and PCA demands high computational requirements for such data. PCA is a crucial aspect of 4C method. In 4C method, PCA is used to find correlations in the neighbourhood of a core object and PCA plays vital role in 4C in finding correlated clusters. A clustering approach based on subsets of attributes was previously presented by Friedman and Meulman [46]. Following this line of thinking, a novel FP-PCA approach, SubXPCA was proposed (Chapter 4) to improve PCA. SubXPCA finds local features from subsets of attributes and combines them globally like PCA does. More importantly, SubXPCA (Chapter 4) was found to be computationally superior to PCA for high dimensional data. It was also proved that SubXPCA shows improved classification rates as compared to PCA. In this Chapter we attempt to improve 4C method based upon our insight into advantages of FP-PCA approaches (i.e. SubXPCA and similar methods) for high dimensional data and we prove the computational superiority of our approach over 4C method.

The organization of the chapter is as follows. In section 7.2 we discuss concepts related to DBSCAN [40] and we review the salient aspects of 4C [12] in section 7.3. We propose 'A Feature Partitioning approach to Correlation Connected Clusters' (FP-4C) in section 7.4. The time efficiency of FP-4C over 4C is proved in section 7.5.

7.2 Density-Based Spatial Clustering of Applications of Noise (DBSCAN)

In this section, we review concepts and algorithm related to DBSCAN in brief. For details, please see [40][129]. DBSCAN works on a density-based notion of clusters which discovers clusters of arbitrary shape. DBSCAN was proved to be more efficient as compared to CLARANS method [129], by a factor of about 100 [40].

7.2.1 Definitions

In this section we discuss the definitions in brief as presented in [40]. Consider $\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\}$, the set of N points or patterns (objects), each of d dimensionality.

Definition 13 *ϵ-neighbourhood of a point or a pattern, $\mathbf{X}_i$ ($N_\epsilon(\mathbf{X}_i)$): It is given as follows*

$N_\epsilon(\mathbf{X}_i) = \{\mathbf{X}_j \in \mathbf{X} \,|dist(\mathbf{X}_i, \mathbf{X}_j) \leq \epsilon\}$

Definition 14 *Core Point (or Core Object): A point, $\mathbf{X}_i$, is said to be a core point if $N_\epsilon(\mathbf{X}_i) \geq \mu$, where μ indicates the minimum number of points.*

Definition 15 *Direct Density Reachability. A point, $\mathbf{X}_i \in \mathbf{X}$ is directly reachable from another point $\mathbf{X}_j \in \mathbf{X}$ with respect to μ and ϵ, ($DirReach(\mathbf{X}_j, \mathbf{X}_i)$) if the following conditions hold.*

$DirReach(\mathbf{X}_j, \mathbf{X}_i) \Leftrightarrow$ (a) $\mathbf{X}_i \in N_\epsilon(\mathbf{X}_j)$ and (b) $\mathbf{X}_j$ is a core point.

Definition 16 *Density Reachability. A point $\mathbf{O}_1$ is density reachable from a point $\mathbf{O}_2$ with respect to μ and ϵ ($DenReach(\mathbf{O}_2, \mathbf{O}_1)$ if there is a chain of points $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_M$*

$(\mathbf{X}_1 = \mathbf{O}_2, \mathbf{X}_M = \mathbf{O}_1)$ *such that* $\mathbf{X}_{i+1}$ *is directly density reachable from* $\mathbf{X}_i$.

Definition 17 *Density Connected Property. A point,* $\mathbf{X}_i$*, is density-connected to another point,* $\mathbf{X}_j$ *with respect to* ϵ *and* μ *if there is a point* $\mathbf{O}$ *such that both* $\mathbf{X}_i$ *and* $\mathbf{X}_j$ *are density reachable from* $\mathbf{O}$ *with respect to* ϵ *and* μ.

Definition 18 *Cluster. Let* $\mathbf{X}$ *be a database of points or patterns. A cluster* $\mathbf{CL}$ *with respect to* ϵ *and* μ *is a non-empty subset of* $\mathbf{X}$ *satisfying the following conditions: (i)* $\forall\, \mathbf{X}_i, \mathbf{X}_j \in \mathbf{X}$*: if* $\mathbf{X}_i \in \mathbf{CL}$ *and* $\mathbf{X}_j$ *is density reachable from* $\mathbf{X}_i$ *with respect to* ϵ *and* μ*, that is* $DenReach(\mathbf{X}_i, \mathbf{X}_j)$ *then* $\mathbf{X}_j \in \mathbf{CL}$ *(Maximality). (ii)* $\forall\, \mathbf{X}_i, \mathbf{X}_j \in \mathbf{CL}$*:* $\mathbf{X}_i$ *is density-connected to* $\mathbf{X}_j$ *with respect to* ϵ *and* μ *(Connectivity).*

Definition 19 *Noise. Let* $\mathbf{CL}_1, \mathbf{CL}_2, \ldots, \mathbf{CL}_s$ *be the clusters of the database* $\mathbf{X}$ *with respect to parameters* ϵ_i *and* μ_i*,* $i = 1, 2, ..., s$ *respectively. Then we define the noise as the set of points in* $\mathbf{X}$ *that does not belong to any cluster* $\mathbf{CL}_i$*,* $i \in \{1, 2, \ldots, s\}$*, That is noise* $= \{\mathbf{X}_j \in \mathbf{X} | \forall i : \mathbf{X}_j \notin \mathbf{CL}_i\}$.

7.2.2 DBSCAN Algorithm

In this section, we present DBSCAN algorithm in brief and more details can be found in [40][129]. Consider $\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\}$, the set of N patterns (objects or points), each of d dimensionality.

Input: (i) $\mathbf{X}$, pattern (object) set (ii) ϵ, the neighbourhood parameter (iii) μ, the minimum number of points in $\epsilon-$neighbourhood.

Output: Every object in $\mathbf{X}$ is assigned a cluster-id or marked as noise.

Assumption: Each pattern (object) in $\mathbf{X}$ is marked as unclassified initially.

Method:

For each unclassified pattern (object), $\mathbf{X}_i \in \mathbf{X}$ repeat the steps $1-2$.

1. Test whether $\mathbf{X}_i$ is a core object as follows:

1.1. Compute $\epsilon-$neighbourhood of $\mathbf{X}_i$, $N_\epsilon(\mathbf{X}_i)$.

1.2. If Number of objects (patterns) in $N_\epsilon(\mathbf{X}_i) \geq \mu$, then

$\mathbf{X}_i$ is a core object.

else

Mark $\mathbf{X}_i$ as a noise and return.

2. If ($\mathbf{X}_i$ is a core object) then Expand a new cluster as follows:

2.1. Generate new cluster-id and assign the same id to all objects in $N_\epsilon(\mathbf{X}_i)$.

2.2. Insert all $\mathbf{X}_j \in N_\epsilon(\mathbf{X}_i)$ into list of candidate objects, **CD**.

2.3. While (**CD** is not empty) repeat the steps $2.3.1-2.3.3$.

2.3.1. Let **U** be current object in **CD** and Remove it from candidate objects, **CD**.

2.3.2. Compute $\epsilon-$neighbourhood of **U**, that is $N_\epsilon(\mathbf{U})$.

2.3.3. if $|N_\epsilon(\mathbf{U})| \geq \mu$ then repeat steps 2.3.3.1–2.3.3.2.

2.3.3.1. Select all the objects in $N_\epsilon(\mathbf{U})$ which are not yet classified or noise then Assign current cluster-id to them.

2.3.3.2. If an object $\mathbf{X}_j \in N_\epsilon(\mathbf{U})$ is unclassified then

Insert $\mathbf{X}_j$ into list of candidate sets **CD**.

The sample clusters from DBSCAN are shown in Fig. 7.1.

7.3 Computing Clusters of Correlation Connected Objects (4C)

We review briefly the 4C method here and a detailed discussion may be found in Bohm et al's work [12]. Consider $\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\}$, the set of N patterns (objects), each of d dimensionality.

7.3.1 Definitions

We give some of the definitions of correlation core object, correlation dimension, direct correlation-reachability, correlation ϵ-neighbourhood, etc, as defined by Bohm et al [12] for our discussion.

Definition 20 *Correlation ϵ-neighbourhood. The correlation ϵ-neighbourhood of an object (pattern), $\mathbf{O}$ is given by* $N_\epsilon^{\bar{\mathbf{C}}_\mathbf{O}}(\mathbf{O}) = \{\mathbf{X}_i \in \mathbf{X} | \; max\{dist_\mathbf{O}(\mathbf{O}, \mathbf{X}_i), dist_{\mathbf{X}_i}(\mathbf{X}_i, \mathbf{O})\} \leq \epsilon\}$ *where* $dist_\mathbf{O}(\mathbf{O}, \mathbf{X}_i) = \sqrt{(\mathbf{O} - \mathbf{X}_i).\bar{\mathbf{C}}_\mathbf{O}.(\mathbf{O} - \mathbf{X}_i)^T}$, $\bar{\mathbf{C}}_\mathbf{O}$ *is the correlation similarity matrix as defined in [12].*

Definition 21 *Correlation core object. A point, $\mathbf{X}_i$ is a correlation core object with respect to ϵ, μ, δ and θ if (i) its ϵ-neighbourhood is a θ-dimensional linear correlation set and (ii) its correlation ϵ-neighbourhood contains at least μ points.*

Definition 22 *Correlation dimension. Let $\mathbf{S} \subseteq \mathbf{X}$, $\theta \leq d$, $\mathbf{EV} = \{\lambda_1, \lambda_2, \ldots, \lambda_d\}$, the eigenvalues of $\mathbf{S}$ in descending order and $\delta \in \Re\, (\delta \approx 0)$. $\mathbf{S}$ forms a $\theta-$ dimensional*

linear correlation set w.r.t. δ if at least $d - \theta$ eigenvalues of $\mathbf{S}$ are close to zero. Let $\mathbf{S} \subseteq \mathbf{X}$ be a linear correlation set w.r.t. $\delta \in \Re$. The number of eigenvalues with $\lambda_i > \delta$ is called the correlation dimension of $\mathbf{S}$.

Definition 23 *Direct correlation reachability ($DirCorReach^{\delta,\theta}_{\epsilon,\mu}(\mathbf{O}_2, \mathbf{O}_1)$): Let $\epsilon, \delta \in \Re$ and $\mu, \theta \in \mathbf{N}$. A point $\mathbf{O}_1 \in \mathbf{X}$ is direct correlation-reachable from a point $\mathbf{O}_2 \in \mathbf{X}$ with respect to ϵ, μ, δ and θ if (i) $\mathbf{O}_2$ is correlation core object, (ii) the correlation dimension of ϵ-neighbourhood of $\mathbf{O}_1$, $N_\epsilon(\mathbf{O}_1)$ is at most θ and (iii) $\mathbf{O}_1 \in N_\epsilon^{\bar{\mathbf{C}}_{\mathbf{O_2}}}(\mathbf{O}_2)$, where $\bar{\mathbf{C}}_{\mathbf{O_2}}$ is the correlation similarity matrix.*

7.3.2 4C Algorithm

We present 4C algorithm in brief here and more details can be found in [12].

Input: $\mathbf{X}$, pattern (object) set; ϵ, the neighbourhood parameter; μ, the number of points in $\epsilon-$neighbourhood; θ, the upper bound for correlation dimension; δ, the threshold to select correlation dimension.

Output: Every object is assigned a cluster-id or marked as noise.

Assumption: Each pattern (object) in $\mathbf{X}$ is marked as unclassified initially.

Method: For each unclassified pattern (object), $\mathbf{X}_i \in \mathbf{X}$ repeat the steps $1 - 2$.

1. Test whether $\mathbf{X}_i$ is a correlation core object as follows:

1.1. Compute the $\epsilon-$neighbourhood of $\mathbf{X}_i$, $N_\epsilon(\mathbf{X}_i)$.

1.2. If the Number of elements in $N_\epsilon(\mathbf{X}_i) \geq \mu$, then

1.2.1. Compute covariance matrix of patterns in $N_\epsilon(\mathbf{X}_i)$, i.e. $\mathbf{C}_{\mathbf{X}_i}$.

1.2.2. If the Correlation Dimension of $N_\epsilon(\mathbf{X}_i) \leq \theta$ then

1.2.2.1. Compute correlation similarity matrix, $\bar{\mathbf{C}}_{\mathbf{X}_i}$ and $N_\epsilon^{\bar{\mathbf{C}}_{\mathbf{X}_i}}(\mathbf{X}_i)$.

1.2.2.2. If the number of patterns in $N_{\epsilon}^{\bar{\mathbf{C}}_{\mathbf{X}_i}}(\mathbf{X}_i) \geq \mu$ then

$\mathbf{X}_i$ is a correlation core object

else

Mark $\mathbf{X}_i$ as a noise and return.

2. If ($\mathbf{X}_i$ is a correlated core object) then Expand a new cluster as follows:

2.1. Generate a new cluster-id.

2.2. Insert all $\mathbf{X}_j \in N_{\epsilon}^{\bar{\mathbf{C}}_{\mathbf{X}_i}}(\mathbf{X}_i)$ into a queue, **CD**.

2.3. While (**CD** is not empty) repeat the steps $2.3.1 - 2.3.4$.

2.3.1. $\mathbf{U}$ = first object in **CD**.

2.3.2. Compute $\mathbf{DR} = \{\mathbf{X}_j \in \mathbf{X} |\ DirCorReach_{\epsilon,\mu}^{\delta,\theta}(\mathbf{U}, \mathbf{X}_j)\}$,

where $DirCorReach_{\epsilon,\mu}^{\delta,\theta}(\mathbf{U}, \mathbf{X}_j)$ indicates $\mathbf{X}_j$ is direct correlation reachable from correlation core object $\mathbf{U}$.

2.3.3. For each $\mathbf{X}_j \in \mathbf{DR}$ repeat the steps 2.3.3.1-2.3.3.2

2.3.3.1. If $\mathbf{X}_j$ is unclassified or noise then Assign current cluster-id to $\mathbf{X}_j$.

2.3.3.2. If $\mathbf{X}_j$ is unclassified then insert $\mathbf{X}_j$ into queue **CD**.

2.3.4. Remove $\mathbf{U}$ from queue **CD**.

It is better to see an example clustering obtained from DBSCAN and 4C methods as shown in Figs. 7.1 and 7.2 to understand the differences between the two methods. 4C method was found to be very useful to find correlations in subsets of data such as microbiology, e-commerce. 4C makes use of PCA to find correlations in the data set, which is not suitable for high dimensional data, hence 4C consumes a large chunk of

time for such data sets. To counter this problem, we make use of SubXPCA (Chapter 4), a more efficient variation of PCA.

7.4 A Feature Partitioning Approach to Correlation Connected Clusters (FP-4C)

In this section we present our clustering method, FP-4C, which is based on DBSCAN [40] and SubXPCA (Chapter 4). To find the correlation dimension of ϵ-neighbourhood of a core object (Step 1.2.2 of section 7.3.2), 4C method uses classical PCA which is computationally expensive for high-dimensional data. In addition, PCA is a global feature extraction method (that is, it makes use of all covariances between every pair of features) and PCA may not be appropriate when local feature variations are prominent. Here local features are those extracted from sub-patterns with subsets of original features rather than whole patterns with all original features. To ease this problem, FP-4C method uses SubXPCA to find eigenvalues, eigenvectors and to compute correlation dimension. SubXPCA was proved to be more efficient in terms of (i) computational complexity, (ii) classification as compared to classical PCA. SubXPCA is flexible enough to adapt to both global variations and local variations with respect to feature extraction (Chapter 4). The FP-4C algorithm is as follows.

The same definitions of 4C method [12] – core object, correlation core object, correlation dimension, correlation similarity matrix, correlation set, correlation density reachability, etc., are also applicable to FP-4C method. Hence we do not reproduce

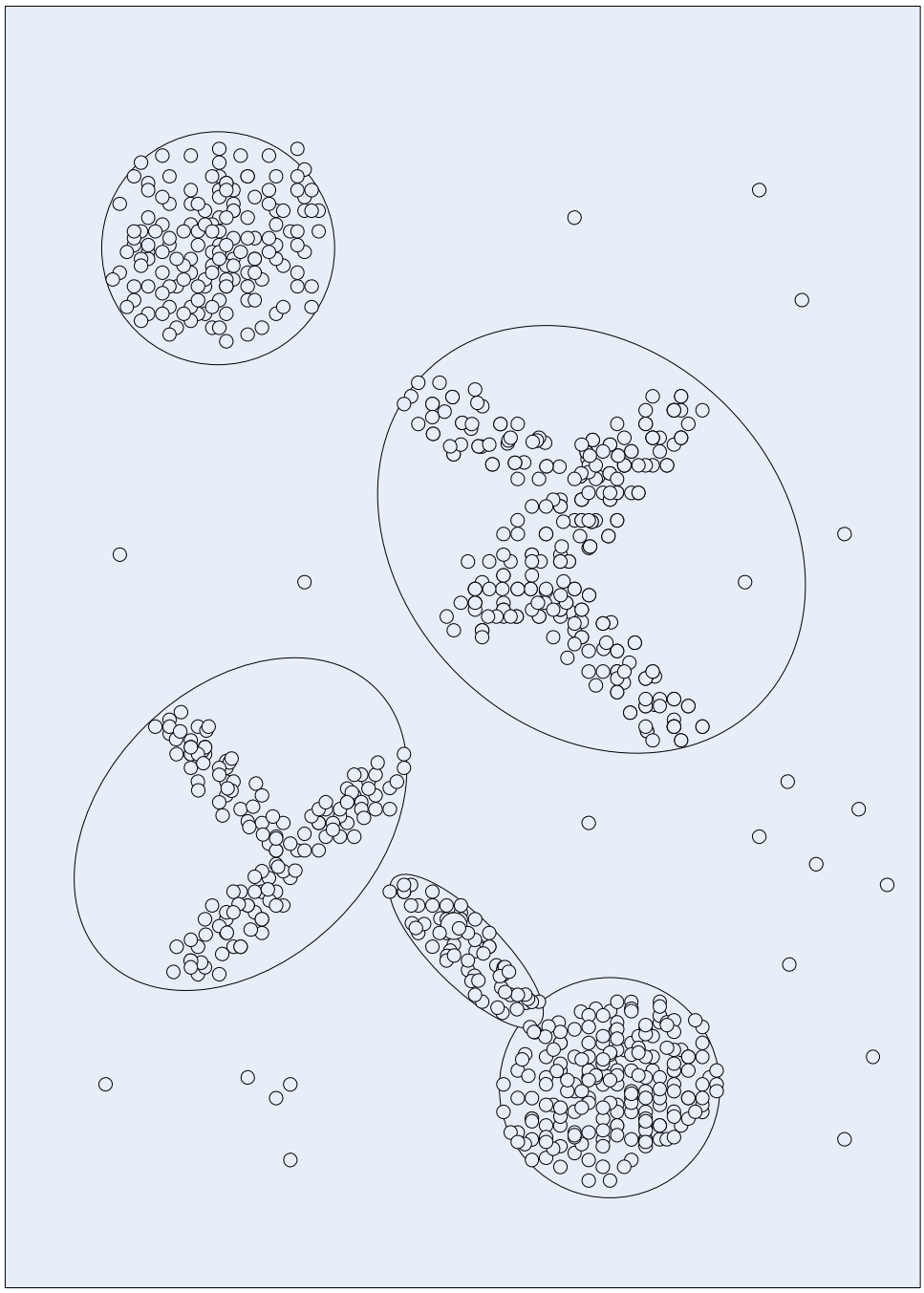


Figure 7.1: An example of clustering produced by DBSCAN

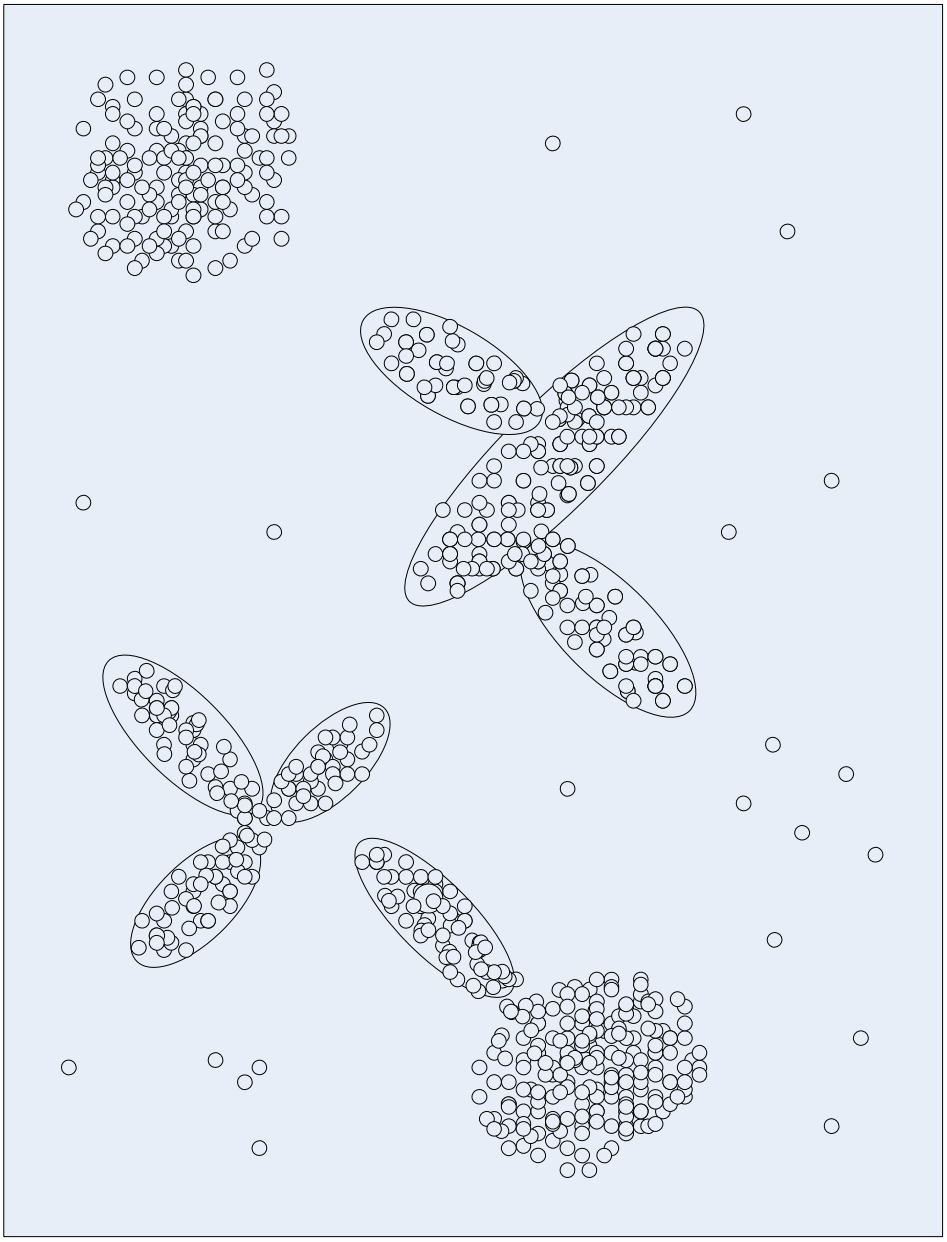


Figure 7.2: An example of clustering produced by 4C or FP-4C

them here.

7.4.1 FP-4C Algorithm

Input: $\mathbf{X}$, pattern (object) set; ϵ, the neighbourhood parameter; μ, the number of points in $\epsilon-$neighbourhood; θ, the upper bound for correlation dimension; δ, the threshold to select correlation dimension.

Output: Every object is assigned a cluster-id or marked as noise.

Assumption: Each pattern (object) in $\mathbf{X}$ is marked as unclassified initially.

Method: For each unclassified pattern (object), $\mathbf{X}_i \in \mathbf{X}$ repeat the steps $1-2$.

1. Test whether $\mathbf{X}_i$ is correlation core object as follows:

1.1. Compute the ϵ-neighbourhood of $\mathbf{X}_i$, $N_\epsilon(\mathbf{X}_i)$.

1.2. If the Number of elements in $N_\epsilon(\mathbf{X}_i) \geq \mu$, then

1.2.1. Find the Correlation Dimension of $N_\epsilon(\mathbf{X}_i)$ using SubXPCA (Chapter 4) as given in steps 1.2.1.1–1.2.1.3 (Figs. 7.3-7.4):

1.2.1.1. (i) Partition each pattern (object) in $N_\epsilon(\mathbf{X}_i)$ of pattern $\mathbf{X}_i$, into $k\,(\geq 2)$ sub-patterns of size u, (ii) find sub-covariance matrices for each of the sub-patterns, (iii) select r eigenvectors corresponding to first highest eigenvalues, (iv) project sub-patterns onto the selected eigenvectors to get locally-reduced sub-patterns, (v) concatenate locally-reduced sub-patterns (of the same pattern) to form locally-reduced patterns and (vi) find final covariance matrix ($\mathbf{C}^g_{\mathbf{X}_i}$) using locally-reduced patterns of the same neighbourhood.

1.2.1.2. Compute eigenvalues of $\mathbf{C}^g_{\mathbf{X}_i}$ obtained in step 1.2.1.1.

1.2.1.3. Count the number of eigenvalues greater than δ to get Correlation Dimension of $N_\epsilon(\mathbf{X}_i)$.

1.2.2. If the Correlation Dimension of $N_\epsilon(\mathbf{X}_i) \leq \theta$ then

1.2.2.1. Compute correlation similarity matrix $\bar{\mathbf{C}}^g_{\mathbf{X}_i}$ and $N_\epsilon^{\bar{\mathbf{C}}^g_{\mathbf{X}_i}}(\mathbf{X}_i)$.

1.2.2.2. If the number of patterns in $N_\epsilon^{\bar{\mathbf{C}}^g_{\mathbf{X}_i}}(\mathbf{X}_i) \geq \mu$ then

$\mathbf{X}_i$ is a correlation core object.

else

Mark $\mathbf{X}_i$ as a noise.

Step 2: Same as 4C method (Section 7.3), except that $\bar{\mathbf{C}}$ is replaced with $\bar{\mathbf{C}}^g$. Hence for brevity we do not reproduce it here.

To give a better conceptual comprehension of our method we summarize it in Figs. 7.3 and 7.4. The FP-4C method produces correlation clusters similar to 4C method as shown in Fig. 7.2.

7.5 Computational Analysis of 4C and FP-4C

It is known that 4C and FP-4C methods use classical PCA and SubXPCA respectively for correlation analysis. In PCA variations, where covariance matrix is computed explicitly, a large amount of time is spent in computing covariance matrix, and relatively an insignificant amount of time for other tasks such as finding eigenvalues, eigenvectors, computing mean-subtracted data, matrix multiplications, etc,. Consider $\mathbf{X} = \{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\}$, the set of N patterns (objects) of size d. From section 4.3 of Chapter 4, we know the time complexities of classical PCA and SubXPCA and are reproduced here.

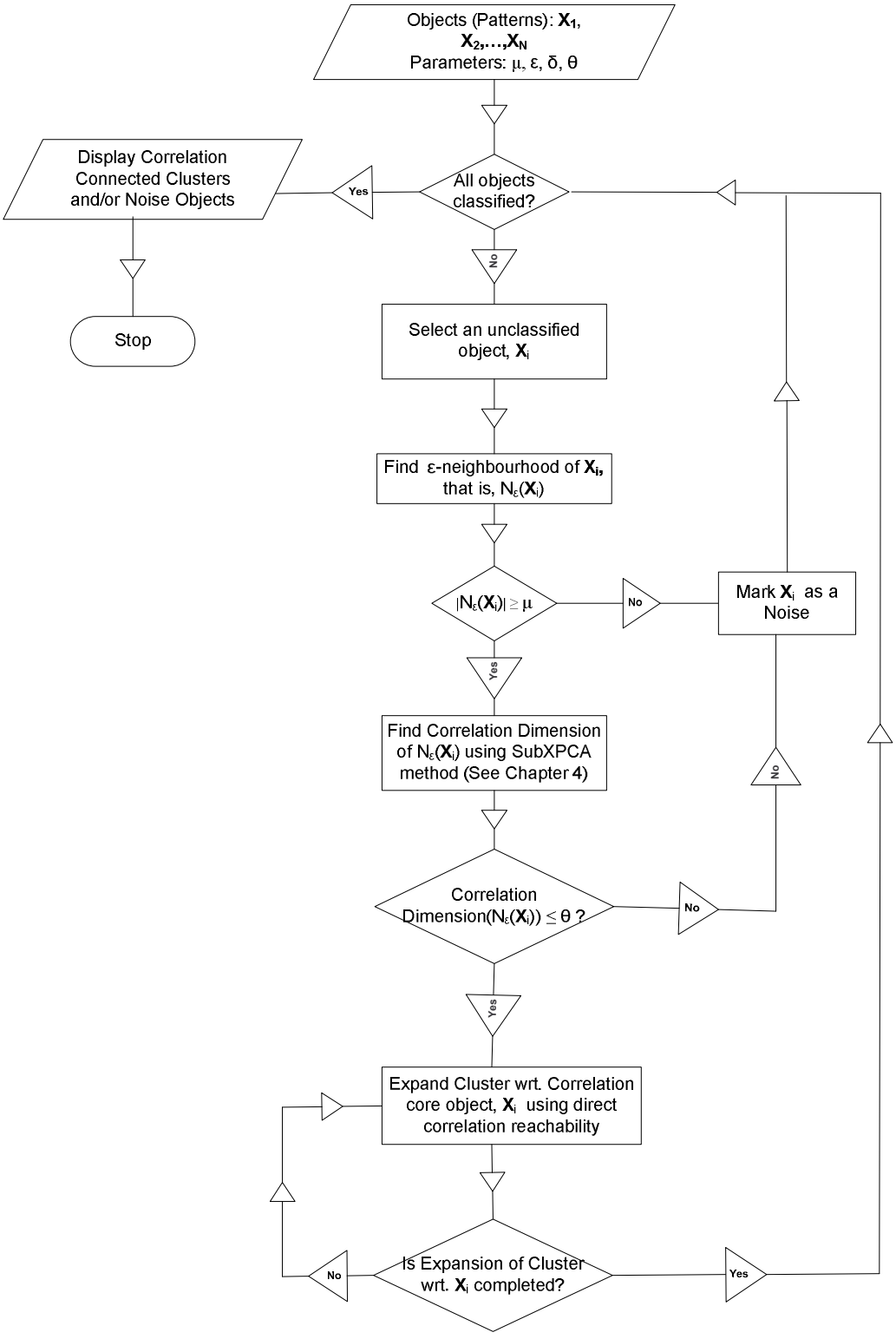


Figure 7.3: The flow chart of proposed FP-4C algorithm-Part I

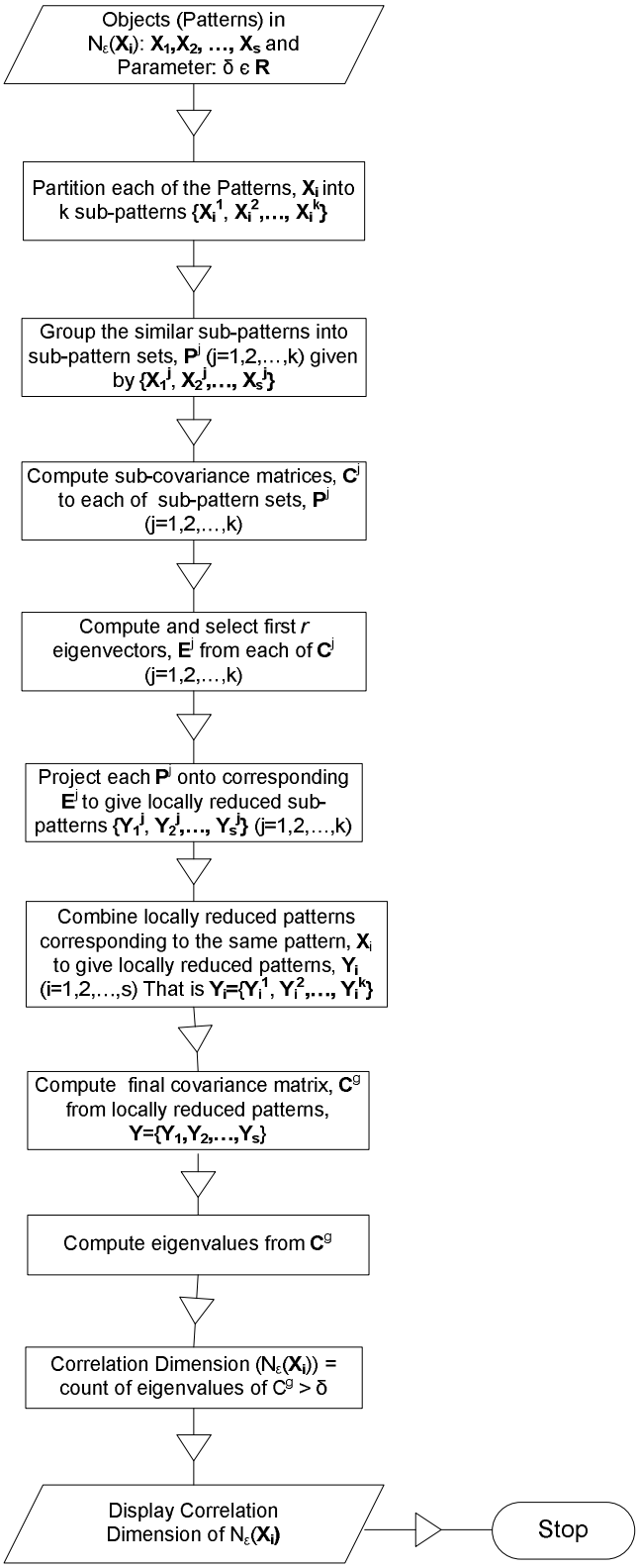


Figure 7.4: The flow chart of proposed FP-4C algorithm-Part II (To find Correlation dimension of $N_\epsilon(\mathbf{X}_i)$ using SubXPCA method)

Table 7.1: Time complexities of 4C and FP-4C clustering methods

To compute	4C method	FP-4C method
ϵ-neighbourhood, $N_\epsilon(\mathbf{O})\ \forall\, \mathbf{O} \in \mathbf{X}$	$O(N^2.d)$	$O(N^2.d)$
Covariance matrix $\forall\, \mathbf{O} \in \mathbf{X}$	$O(N^2.d^2)$	$O[k.(N^2.u^2) + k.N.u^3]$ $+O[N^2.(k.r)^2]$
Correlation Dimension $\forall\, \mathbf{O} \in \mathbf{X}$	$O(N.d^3)$	$O(N.(k.r)^3)$
Correlation similarity matrix $\forall\, \mathbf{O} \in \mathbf{X}$	$O(N.d^3)$	$O[N.(k.r)^3]$
Correlation ϵ-neighbourhood $\forall\, \mathbf{O} \in \mathbf{X}$	$O(N^2.d^2)$	$O(N^2.(k.r)^2)$
Total time complexity	$O(N^2.d^2 + N.d^3)$ $= N.T_C$	$O(k.(N^2.u^2) + N.k.u^3)$ $+O(N^2.(k.r)^2 + N.(k.r)^3)$ $= N.T_F$

The time complexity of PCA, T_C, is given by

$T_C = O(N.d^2 + d^3)$

The time complexity of SubXPCA, T_F, is given by

$T_F = O[k.(N.u^2 + u^3)] + O[N.(k.r)^2 + (k.r)^3]$

The time complexity of 4C method, T_{4C}, is given by (Table 7.1).

$$T_{4C} = O(N.N.d^2 + N.d^3) = O(N.T_C) \tag{7.1}$$

On the same lines, the time complexity of FP-4C method, T_{FP}, is given by (Table 7.1)

$$T_{FP} = O[k.(N^2.u^2) + N.k.u^3 + N^2.(k.r)^2 + N.(k.r)^3] = O(N.T_F) \tag{7.2}$$

where u is the sub-pattern size and k is the number of sub-patterns per pattern (Section 4.2 of Chapter 4).

Theorem 31 *$T_{FP} < T_{4C}$, $\forall r < u.\sqrt{\frac{k-1}{k}}$, where $2 \leq k \leq \frac{d}{2}$, is the number of sub-patterns (blocks) per pattern, d is the pattern size, r is the number of chosen eigen-*

vectors per sub-pattern set, $\mathbf{P}^j$ *and* u *is the sub-pattern size (See Chapter 4 for terminology).*

Proof 31 *From Theorem 3 of Chapter 4, we know that* $T_F < T_C$, $\forall r < u.\sqrt{\frac{k-1}{k}}$, *where* $2 \leq k \leq \frac{d}{2}$.

$\Rightarrow N.T_F < N.T_C$, $\forall r < u.\sqrt{\frac{k-1}{k}})$.

$\Rightarrow T_{FP} < T_{4C}$, $\forall r < u.\sqrt{\frac{k-1}{k}})$ *(from eqs. (7.1)-(7.2)).*

Hence the theorem follows.

7.6 Discussion: Why is FP-4C more Efficient than 4C ?

FP-4C uses SubXPCA to compute eigenvalues which are used to find correlation dimension of ϵ-neighbourhood of a core object (Step 1.2.1 of section 7.4.1). Similarly 4C method uses classical PCA for computing eigenvalues. In PCA variations, where covariance matrix is computed explicitly, most of the time is consumed for the computation of covariance matrix alone. In contrast to classical PCA (where a single large covariance matrix, $\mathbf{C}$, is computed), SubXPCA computes $k\,(\geq 2)$, smaller sub-pattern covariance matrices ($\mathbf{C}^j$), one for each sub-pattern set, $\mathbf{P}^j$, and a final covariance matrix ($\mathbf{C}^g$). By the Theorem 3 of Chapter 4, it is obvious that $T_F < T_C$, $\forall r < u.\sqrt{\frac{(k-1)}{k}}$, where r is the number of eigenvectors selected from each sub-pattern set, $\mathbf{P}^j$, u is the sub-pattern size and k is the number of sub-patterns per pattern. The upper bound for r (i.e. $u.\sqrt{\frac{(k-1)}{k}}$) is reasonably large and in practice, we choose first few salient features (i.e. r is small in general), therefore, the computation of final co-

variance matrix, $\mathbf{C}^g$, becomes trivial. Finally, SubXPCA is faster by nearly k times to PCA (as proved in Theorem 4 of Chapter 4). The concept of partitioning is the basic reason for the lower time complexity of SubXPCA. Since we use SubXPCA, instead of classical PCA, in FP-4C for finding correlation dimension of $\epsilon-$neighbourhood, FP-4C is thus faster than 4C and the same is proved in Theorem 31. It was found that classification results of SubXPCA are better than classical PCA (In some cases, the results were same for both SubXPCA and PCA) (Chapter 4).

7.7 Summary

In this Chapter we have proposed a new and efficient method, FP-4C, for correlation cluster analysis which is suitable for high dimensional data. Theoretical proofs reveal that FP-4C is more efficient than 4C. 4C becomes a special case of FP-4C if (i) the number of local features from each sub-pattern in SubXPCA is taken as equal to sub-pattern size, u. FP-4C may be extensively used in high dimensionality data mining and other pattern recognition tasks.

In the next Chapter, we show how a feature partitioning approach (FP-PCA) can be used for subspace classification.

Chapter 8

A Feature Partitioning Approach to Subspace Classification

8.1 Introduction

Note: An initial version of the work in this chapter has been published in *proceedings of IEEE TenCon 2007 International Conference*[1].

In this Chapter, we explore the applicability of feature partitioning based PCA (FP-PCA) methods such as SubXPCA (Chpater 4), SubPCA (Section 2.2 of Chapter 2) for subspace classification. Subspace classification is one of the widely used methods for pattern recognition tasks, where a linear subspace of the Euclidean sample space is found [115]. The motivation for subspace classifiers originates from compression and optimal reconstruction of multidimensional data with linear principal

[1]Kadappagari Vijaya Kumar and Atul Negi, "A feature partitioning approach to subspace classification", *In Proceedings of IEEE TenCon 2007 Conference*, Taipei, Taiwan, pp. 1-4, 30.10.2007-Nov. 2^{nd} 2007.

components. The use of linear subspaces as class models is based on the assumption that the vector distribution in each class lies approximately on a lower-dimensional subspace of the feature space. The subspaces representing classes are defined in terms of basis vectors that are linear combinations of the sample vectors of each class. Once the basis vectors spanning those subspaces are computed, a test data vector from an unknown class is classified based on the lengths of the projections of that sample onto each of the subspaces or, alternatively, on the distances of the test vector from these subspaces. Even though this linearity assumption may not be valid in all the cases, acceptable classication accuracies can be achieved if the input vector dimensionality is large enough [93].

Subspace methods in data analysis date back to the 1930s by Hotelling [64]. The value of the subspace methods in data compression and optimal reproduction was observed in the 1950s by Kramer and Mathews [89]. Later, Watanabe et al [172] published the first application in pattern classification. Learning subspace methods gained popularity after the pioneering work of Kohonen et al [87] in 1970s. These methods aimed for classication instead of optimal compression or reproduction since their inception. The guiding idea in the learning methods is to modify the bases of the subspaces in order to diminish the number of misclassifications. The nature of the modifications varies in different learning algorithms.

The remainder of the Chapter is organized as follows. In section 8.2 we review some classical PCA based subspace classification methods. We propose a novel Sub-XPCA based Feature Partitioning approach to Subspace Classification (FP-SC) in section 8.3. Time complexity analysis of subspace methods is done in section 8.4. We

demonstrate our approach by using experimentation on UCI repository of Machine Learning data sets in section 8.5.

8.2 Review of Classical Subspace Methods

In this section, we discuss some of classical PCA based subspace methods in brief. A useful review of subspace classification may be found in [93].

8.2.1 Class-Featuring Information Compression(CLAFIC)

Watanabe et al applied Principal Component Analysis (PCA), or the Karhunen-Loeve Transform (KLT), in classification which is known as CLAFIC algorithm [172]. CLAFIC simply forms the base vectors for the classifier subspaces from the eigenvectors of the covariance matrix or correlation matrix of each class. For each class h_q, we compute the covariance matrix $\mathbf{C}_q = E[\mathbf{x}.\mathbf{x}^T|\mathbf{x} \in h_q]$. Then we find first r eigenvectors of $\mathbf{C}_q$, $\{\mathbf{e}_1^q, \mathbf{e}_2^q, \ldots, \mathbf{e}_r^q\}$, in the order of decreasing eigenvalues λ_i^q, $i = 1, 2, \ldots, r$ and used as columns of the basis eigenvector matrix $\mathbf{E}_q$, which is given by

$$\mathbf{E}_q = \{\mathbf{e}_i^q | (\mathbf{C}_q.\mathbf{e}_i^q = \mathbf{e}_i^q.\lambda_i^q;\ \lambda_i^q \geq \lambda_{i+1}^q, i = 1, 2, \ldots, r\} \tag{8.1}$$

A test vector $\mathbf{L}$ is classified according to the maximal similarity value using the function

$$B(\mathbf{L}) = argmax_{q=1,\ldots,c} \parallel [\mathbf{E}_q]^T.\mathbf{L} \parallel^2 \tag{8.2}$$

8.2.2 Multiple Similarity Method (MSM)

One way to generalize the classification function (8.2) is to introduce individual weights for all the basis vectors. Iijima et al [69] have selected to weight each basis vector with the corresponding eigenvalue in their Multiple Similarity Method (MSM).

$$D(\mathbf{L}) = argmax_{q=1,\ldots,c} \sum_{i=1}^{r^q} \frac{\lambda_i^q}{\lambda_1^q} . (\mathbf{L}^T . \mathbf{e}_i^q)^2 \tag{8.3}$$

This emphasizes the effect of the most prominent directions, for which $\frac{\lambda_i^q}{\lambda_1^q} \approx 1$. The selection of the subspace dimension r^q is, therefore, less important because the influence of the less prominent eigenvectors, which have multipliers $\frac{\lambda_i^q}{\lambda_1^q} \approx 0$, becomes trivial. Thus the influence of the eigenvectors which have small eigenvalues and are created by additive noise is thus cancelled out.

One of the problems with classical PCA based subspace methods is large computational complexity especially for high-dimensional data because of high computational time to calculate covariance matrix. The covariance matrix is subsequently used to compute eigenvectors and eigenvalues. Another problem is the classical PCA based subspace methods may not yield good classification especially if local variations are dominant (i.e. variations restricted to a subset of original features). In the next section we propose a promising novel approach to reduce time complexity as well as to improve classification rate, which is based on feature partitioning framework. For information on feature partitioning framework and approaches please see Chapters 3-5.

8.3 Feature Partitioning (SubXPCA based) Approach to Subspace Classification (FP-SC)

In this section, we present our proposed approach to subspace classification based on feature partitioning framework.

8.3.1 FP-SC Algorithm

Consider $\mathbf{X} = \{(\mathbf{X}_1)^1, (\mathbf{X}_2)^1, \ldots, (\mathbf{X}_{N_1})^1, \ldots\ldots, (\mathbf{X}_1)^c, (\mathbf{X}_2)^c, \ldots, (\mathbf{X}_{N_c})^c\}$, the set of $N = N_1 + N_2 + \ldots + N_c$ patterns of size d. Here, $(\mathbf{X}_i)^q$ denotes i^{th} pattern of class h_q, N_q indicates the number of data items which belongs to class h_q, $q \in \{1, 2, \ldots, c\}$ and c is the number of classes.

Step 1: Computing subspace for each class, h_q, using SubXPCA (See Chapter 4 for SubXPCA approach)

(A) Partitioning:

For each class $h_q, q \in \{1, 2, \ldots, c\}$, we divide each pattern, $(\mathbf{X}_i)^q$ into $k\,(\geq 2)$ equally-sized sub-patterns, $\{(\mathbf{X}_i^1)^q, (\mathbf{X}_i^2)^q, \ldots, (\mathbf{X}_i^k)^q\}$. Each sub-pattern is of size u, where $u = \lfloor \frac{d}{k} \rfloor$ and let $\mathbf{P}_j^q$ be the set of j^{th} sub-patterns of $\{(\mathbf{X}_i)^q\}$; $i = 1, 2, \ldots, N_q$ and is given by

$$(\mathbf{P}_j^q)_{N_q \times u} = [(\mathbf{X}_1^j)^q \ (\mathbf{X}_2^j)^q \ \ldots \ (\mathbf{X}_{N_q}^j)^q]^T \tag{8.4}$$

(B) Local feature extraction:

For each sub-pattern set, $\mathbf{P}_j^q$, $j \in \{1, 2, \ldots, k\}$: (i) Compute the sub-covariance matrix $(\mathbf{C}_j^q)_{u \times u}$, then (ii) Choose $r\,(\leq u)$ local column eigenvectors, $(\mathbf{E}_j^q)_{u \times r}$, corresponding to

highest eigenvalues computed from the sub-covariance matrix, $\mathbf{C}_j^q$. (iii) Subsequently extract r local features from $\mathbf{P}_j^q$ by projecting $\mathbf{P}_j^q$ onto $\mathbf{E}_j^q$ as given by

$$(\mathbf{R}_j^q)_{N_q \times r} = (\mathbf{P}_j^q)_{N_q \times u}.(\mathbf{E}_j^q)_{u \times r} = [(\mathbf{Y}_1^j)^q \ (\mathbf{Y}_2^j)^q \ \ldots \ (\mathbf{Y}_{N_q}^j)^q]^T \tag{8.5}$$

$(\mathbf{Y}_i^j)_{r \times 1}^q$ is the set of extracted r local features corresponding to $(\mathbf{X}_i^j)_{u \times 1}^q$ from $\mathbf{P}_j^q$ for class h_q.

(C) Global feature extraction:

(i) Concatenate r local features (extracted in the preceding step) corresponding to the same pattern $(\mathbf{X}_i)^q$ as given by

$$(\mathbf{Y}_i)_{k.r \times 1}^q = [((\mathbf{Y}_i^1)^q)^T, ((\mathbf{Y}_i^2)^q)^T, \ldots, ((\mathbf{Y}_i^k)^q)^T]^T \tag{8.6}$$

(ii) Compute final covariance matrix $(\mathbf{C}^g)_{k.r \times k.r}^q$ from locally-reduced patterns, $(\mathbf{Y}_1)^q$, $(\mathbf{Y}_2)^q$, $\ldots$, $(\mathbf{Y}_{N_q})^q$, then (iii) Compute $w\,(\leq k.r)$ global eigenvectors, $(\mathbf{E}^g)_{k.r \times w}^q$, corresponding to w highest eigenvalues.

Step 2: Classification. For a test pattern, $\mathbf{L}$ of d features : (i) Divide $\mathbf{L}$ into $k\,(\geq 2)$ equally-sized sub-patterns as done in *Step-1*. Each sub-pattern, $\mathbf{L}_j$ $(j = 1, 2, \ldots, k)$ is of size u, where $u = \lfloor \frac{d}{k} \rfloor$. $\mathbf{L}_1$ contains first u features of $\mathbf{L}$, $\mathbf{L}_2$ contains next u features and so on. (ii) Project each $(\mathbf{L}_j)_{u \times 1}$; $j = 1, 2, \ldots, k$ onto $(\mathbf{E}_j^q)_{u \times r}$ of class h_q to get $k.r$ local principal component features, denoted by $(\mathbf{L}^q)_{k.r \times 1}$, as described in the previous step. (iii) Subsequently we classify the test pattern, $\mathbf{L}$ using the following function.

$$F(\mathbf{L}) = argmax_{q=1,2,\ldots,c} \parallel [(\mathbf{E}^g)^q]^T.\mathbf{L}^q \parallel^2 \tag{8.7}$$

where $(\mathbf{E}^g)_{k.r \times w}^q$ is the set of w column global eigenvectors for class h_q obtained in the previous step.

8.4 Time Complexities of Classical and Feature Partitioning based Subspace Classification Methods

We categorize subspace methods such as CLAFIC, MSM, etc as Classical PCA based subspace methods. In PCA based subspace methods, the computation of covariance matrices consumes a large amount of computational time for high-dimensional data in particular, and a relatively insignificant amount of time for other tasks such as finding eigenvalues, etc. Hence, here we focus our study on time complexity of covariance matrices as being computed by classical PCA based subspace methods and FP-SC.

From Chapter 4, we know that the time complexity to calculate a $d \times d$ covariance matrix by classical PCA based subspace methods for a class h_q, T_C^q, is given as

$$T_C^q = O(N_q.d^2 + d^3) \tag{8.8}$$

and the time complexity to calculate all covariance matrices by FP-SC for a class h_q, T_F^q, is given by

$$T_F^q = O(k.N_q.u^2 + k.u^3 + N_q.k^2.r^2 + k^3.r^3) \tag{8.9}$$

The *total time complexity* of a subspace method is the sum of time complexities with respect to all the classes.

Theorem 32 *For a class with a label h_q, $T_F^q < T_C^q$, $\forall r < u.\sqrt{\frac{(k-1)}{k}}$, where $2 \leq k \leq \frac{d}{2}$, is the number of sub-patterns per pattern, r is the number of chosen projection (eigen)vectors (PVs) per sub-pattern set and u is the sub-pattern size.*

Proof 32 *The Theorem directly follows on the similar lines of Theorem 3 of Chapter 4.*

Theorem 33 *$lim_{r\rightarrow 1,\, k\rightarrow 2}[T_F^q \approx (\frac{1}{k}).T_C^q]$, where $2 \leq k \leq \frac{d}{2}$, is the number of sub-patterns per pattern and r is the number of chosen projection (eigen)vectors (PVs) per sub-pattern set.*

Proof 33 *The Theorem directly follows on the similar lines of Theorem 4 of Chapter 4.*

By Theorem 33, $T_F^q \approx (\frac{1}{k}).T_C^q$ is true for smaller values of k and r. However, in practice, r may not be chosen as 1 (i.e. smallest possible value), especially when k is small, since the classification rate may get reduced due to less number of eigenvectors (r). Hence some trade-off between r and k is required to achieve good classification rate and time efficiency.

8.5 Experimental Results and Discussion

In this section, we compare FP-SC method (i.e. SubXPCA based subspace classifier) with PCA based and SubPCA based subspace classifiers. SubPCA [21] is another FP-PCA method and the description on SubPCA method may be found in section 2.2 of Chapter 2.

8.5.1 UCI Data Sets

We considered 2 publicly available databases from UCI repository of Machine Learning [165] for our experiments. (1) Waveform data (21 features, 3 classes with

labels (0,1,2), 5000 Patterns, 50 patterns each class, for training, rest of them for testing). (2) Musk data (166 features, 2 classes with labels (0,1), 6598 patterns, 500 patterns each class, for training, rest of them for testing).

8.5.2 Experimental Setup

For each class, an experiment is conducted as follows: We choose the number of sub-patterns, k, to minimize the truncation of last features as far as possible. For each class, h_q, w projection eigenvectors are found using FP-SC algorithm (Section 8.3.1). For SubPCA based subspace classification, Steps-1(C)(ii)-(iii) are omitted. Each test data is classified into a class for which the norm of projection of the test data item is maximum. The classification is done based on subspace classification rule as given in eq. (8.7) by FP-SC method. For SubPCA based subspace classification, we use $\| \mathbf{L}^q \|^2$ instead of $\| [(\mathbf{E}^g)^q]^T.\mathbf{L}^q \|^2$ in eq. (8.7) of Step-2(iii). The experiment is repeated 10 times, each iteration with different training and testing data sets keeping the same values of k and w. Further (a) the average of these 10 classification rates and (b) best of these 10 classification rates are calculated for the given values of k and w. Now repeat the procedure by varying k and w.

For different data sets, the number of blocks, k is varied as follows: For musk data, we consider $k = 2, 3, 5, 11, 15, 33$ and for waveform data, we consider $k = 2, 3, 4, 5, 7$. In the case of PCA based subspace classification, $k = 1$ for all the data sets.

We plot the results as follows: (i) Among the experiments with varying number of sub-patterns (blocks) we choose the case (i.e. $k = 11$ for SubPCA and SubXPCA with respect to musk data; $k = 2$ for SubPCA and $k = 3$ for SubXPCA with respect

to waveform data) with relative good *average classification performance*. Here we take the average of results of 10 iterations, each iteration with different training and testing data. The average classification results and the corresponding total computational time for 10 iterations, thus obtained are plotted in Figs. 8.1-8.2 for musk data and in Figs. 8.8-8.9 for waveform data.

(ii) Among the experiments with varying number of sub-patterns (blocks) we choose the case (i.e. $k = 5$ for SubPCA and $k = 11$ for SubXPCA with respect to musk data; $k = 2$ for SubPCA and $k = 5$ for SubXPCA with respect to waveform data) with relative *best classification performance*. Here we take the best of classification results of 10 iterations, each iteration with different training and testing data. The best classification results thus obtained are plotted in Fig. 8.3 for musk data and in Fig. 8.10 for waveform data.

The comparison of PCA, SubPCA and SubXPCA based subspace classifiers with respect to both average classification rate and computational time is shown in Fig. 8.4 for musk data.

(iii) The classification performance by *varying the number of sub-patterns (k)* is shown in Figs. 8.5, 8.6, 8.11 and 8.12. For each k (number of sub-patterns): (a) we find maximum average classification rate (of average classification rates obtained with varied number of projection eigenvectors) and is plotted in Fig. 8.5 for musk data and in Fig. 8.11 for waveform data, (b) we find maximum best (of 10 iterations) classification rate (of best classification rates obtained with varied number of projection eigenvectors) and is plotted in Fig. 8.6 for musk data and in Fig. 8.12 for waveform data.

(iv) The comparison of total computational time of 10 iterations for PCA and SubPCA (with different k values) and SubXPCA (with different k values) based subspace classification is shown in Fig. 8.7 for musk data and in Fig. 8.13 for waveform data.

We used Pentium 4 based system with 2.4 GHz CPU clock speed, 256 MB RAM and Fedora Core 5 Linux running on it, to obtain experimental results. We used C language built-in time functions for recording time and procedures, viz. $tqli, tredt, eigensrt$, to find eigenvectors, eigenvalues and for sorting them from [127].

8.5.3 Discussion of Experimental Results

For UCI musk data:

From Fig. 8.1, it is clear that SubXPCA based subspace classifier (FP-SC) outperforms both PCA based and SubPCA-based subspace classifiers. It is observed that SubXPCA based method (FP-SC) shows (i) 7% higher classification rate as compared to PCA based classifier and (ii) 2.5% higher classification rate as compared to SubPCA based classifier. Here SubXPCA based method uses 1 eigenvector from each of sub-pattern sets. Fig. 8.2 shows the computational efficiency of SubXPCA based feature partitioning subspace classifier as compared to PCA and SubPCA based subspace classifiers. A novel plot between average classification rate and computational time (Fig. 8.4) allows one to conclude as to the method that gives good classification at less computational requirements. Such method forms a cluster of its points at top-left corner of the plot. Interestingly, SubXPCA based subspace classifier (FP-SC) forms such a cluster at top-left corner, which shows its superiority over other

methods in terms of classification at less computational time. Please note that other two methods (SubPCA and PCA based subspace classifiers) have their points moved away from top-left corner (except one point of SubPCA), which implies that those methods either show lower classification rate or more computational time or both.

Another way to analyze the subspace methods is in terms of best individual classification rate. From Fig. 8.3, it is seen that SubXPCA based FP-SC method outperforms both PCA and SubPCA based subspace methods. That is, SubXPCA based method (FP-SC) shows (i) 12.4% higher classification rate than PCA based subspace method and (ii) 8.4% higher classification rate than SubPCA based subspace method.

Another interesting analysis is to see the performance of these methods with varying number of sub-patterns or blocks (k). Figs. 8.5 and 8.6 reveal that SubXPCA based subspace classifier (FP-SC) consistently shows better classification as compared to SubPCA based subspace method with varying number of blocks. Note that SubPCA based subspace method shows decreasing performance with increased number of sub-patterns or blocks because more noisy features are added up with increased number of blocks. SubXPCA based method (FP-SC) is able to remove such noisy features effectively by using inter-block correlations or dependencies among these local features. It is also observed that FP-SC shows superior classification as compared to PCA based subspace classifier.

Finally we compare the computational time of all these methods with respect to varying number of sub-patterns or blocks as shown in Fig. 8.7. From the Fig. 8.7 it is clear that both feature partitioning based subspace classifiers (SubPCA and SubXPCA based) show decreasing computational time with increased number of blocks.

However, it is to be noted that PCA shows high computational requirements as compared to feature partitioning based methods (SubXPCA and SubPCA based). Also SubXPCA based subspace classifier shows better computational time as compared to SubPCA based method because SubXPCA is more effective in summarizing most of the variance in less number of principal components.

In a nutshell, our experimentation on UCI musk data shows that, SubXPCA based subspace classifier (FP-SC) outperforms both PCA based and SubPCA based subspace classifies in terms of classification rate and computational time. Please note that SubPCA based subspace classifier shows better classification rate and computational time as compared to PCA based method. This is perhaps due to dominant local variations in waveform data. In this case (where local variations are dominant), SubXPCA shows much better performance as compared to other two methods.

For UCI Waveform data:

From Fig. 8.8 and Fig. 8.10 it is observed that SubXPCA based subspace classifier (FP-SC) shows slightly improved classification rate as compared to SubPCA and PCA based subspace classifiers with respect to maximum classification values. SubXPCA based subspace classifier shows computational superiority as compared to PCA based and SubPCA based subspace classifiers as shown in Figs. 8.9 and 8.13. Please note that PCA based method shows better classification as compared to SubPCA based method, however PCA takes more computational time as compared to SubPCA based method. With varying number of sub-patterns (blocks), SubXPCA based subspace classifier (FP-SC) shows consistently good classification rate as compared to SubPCA based subspace classifier. It is also observed that FP-SC method shows

slightly improved classification as compared to PCA based subspace classifier (Figs. 8.11 and 8.12).

In a nutshell, from our experimentation on UCI waveform data it is clear that SubXPCA based method (FP-SC) shows slightly improved classification rate as compared to PCA based method. SubPCA based subspace classifier shows lower performance as compared to FP-SC method with varying number of blocks. This is perhaps due to more global variations than local variations in the data. However, feature partitioning based subspace classifiers (SubXPCA and SubPCA) remain computationally superior as compared to PCA based subspace classifier.

8.6 Explaining Possible Reason Why FP-SC is better than Other Methods?

From Theorem 32 and Theorem 33, it is clear that FP-SC (SubXPCA based) method can improve its computational time ideally upto $\frac{1}{k}$ times that of classical PCA based subspace methods. From our experimentation it is well known that FP-SC (i.e. SubXPCA based subspace classifier) is more efficient in terms of classification because it considers local structure as well as global structure in computing subspace (Step-1 of algorithm in section 8.3.1). In contrast to FP-SC (SubXPCA based), (i) classical PCA based subspace methods consider only global structure to compute subspace, which may not capture local variations (i.e. variations limited to subset of original features) and (ii) other FP-PCA based subspace classifiers (e.g. SubPCA and similar methods) capture only local structure, but fail to capture global structure.

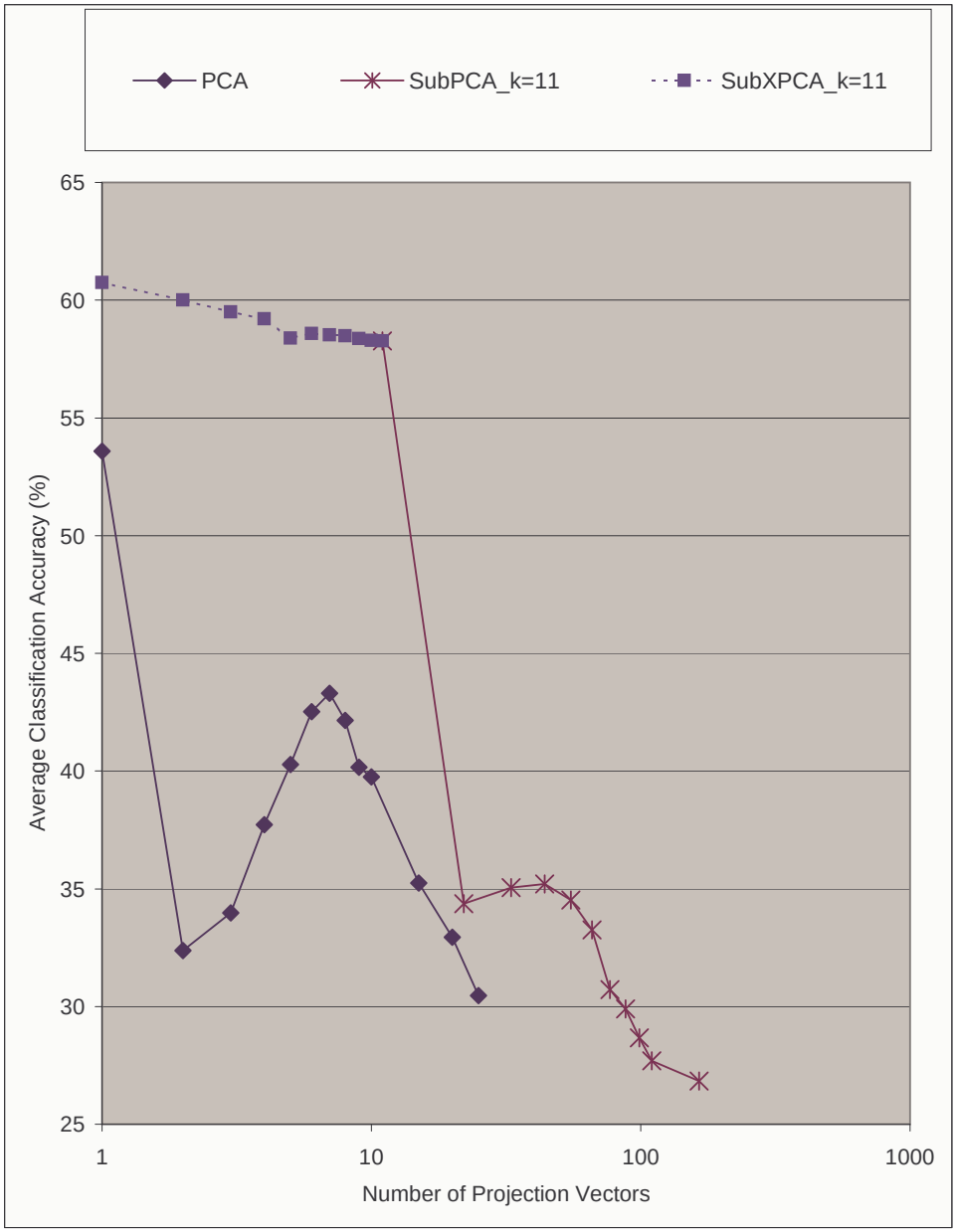


Figure 8.1: *Comparison of average classification rates for UCI Musk data.* SubXPCA based subspace classifier (FP-SC) outperforms both PCA based and SubPCA-based subspace classifiers. It is clear that SubXPCA based method (FP-SC) shows (i) 7% higher classification rate as compared to PCA based subspace classifier and (ii) 2.5% higher classification rate by using less number of projection eigenvectors as compared to SubPCA based subspace classifier. SubXPCA based method uses 1 eigenvector from each of sub-pattern sets.

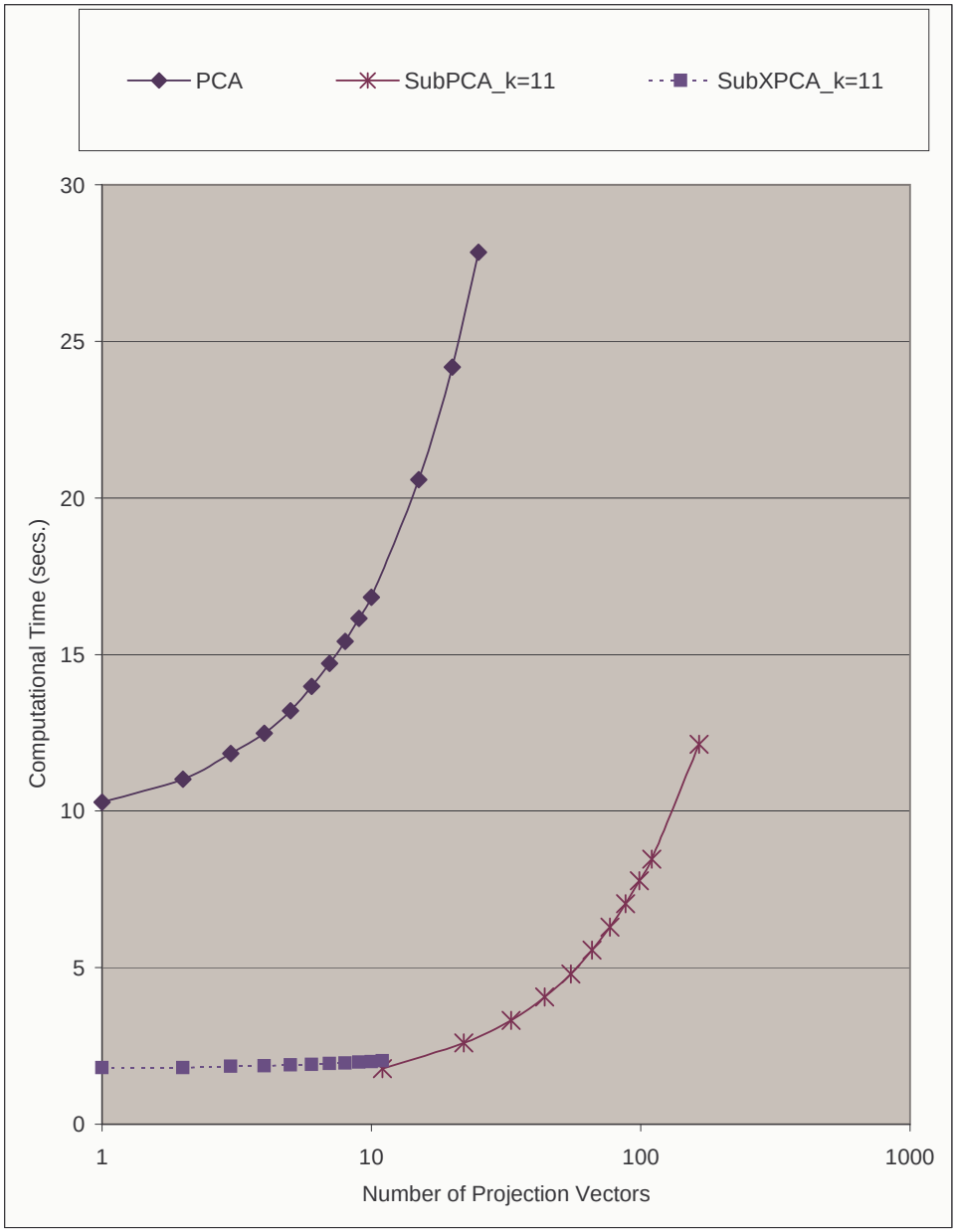


Figure 8.2: *Comparison of computational time for UCI Musk data.* SubXPCA based subspace classifier (FP-SC) shows less computational time as compared to PCA and SubPCA based subspace classifiers.

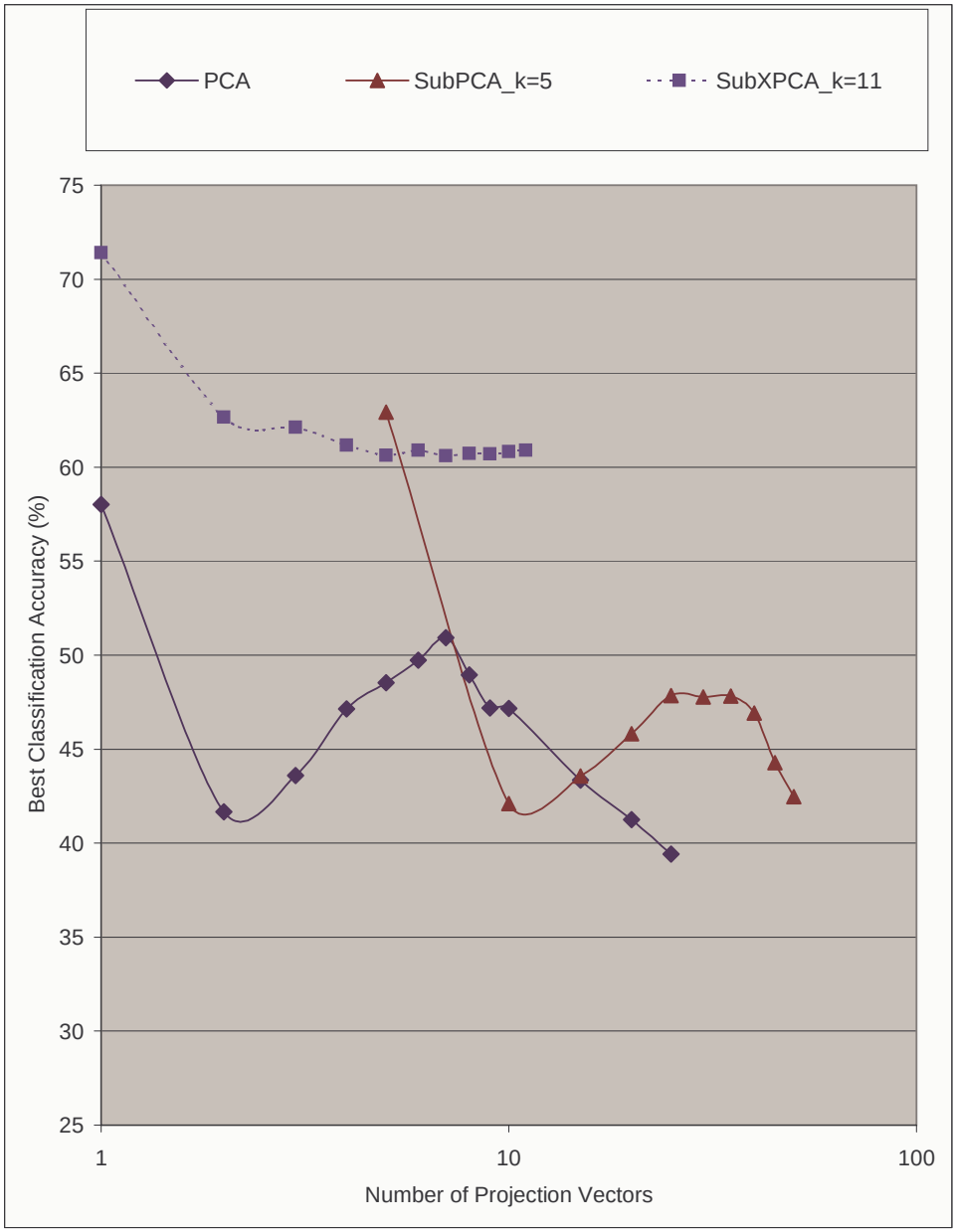


Figure 8.3: *Comparison of best classification rates for UCI Musk data.* SubXPCA based subspace classifier (FP-SC) outperforms both PCA based and SubPCA-based subspace classifiers. It is clear that SubXPCA based method (FP-SC) shows (i) 12.4% higher classification rate as compared to PCA based subspace classifier and (ii) 8.4% higher classification rate by using less number of projection eigenvectors as compared to SubPCA based subspace classifier. SubXPCA based method uses 1 eigenvector from each of sub-pattern sets.

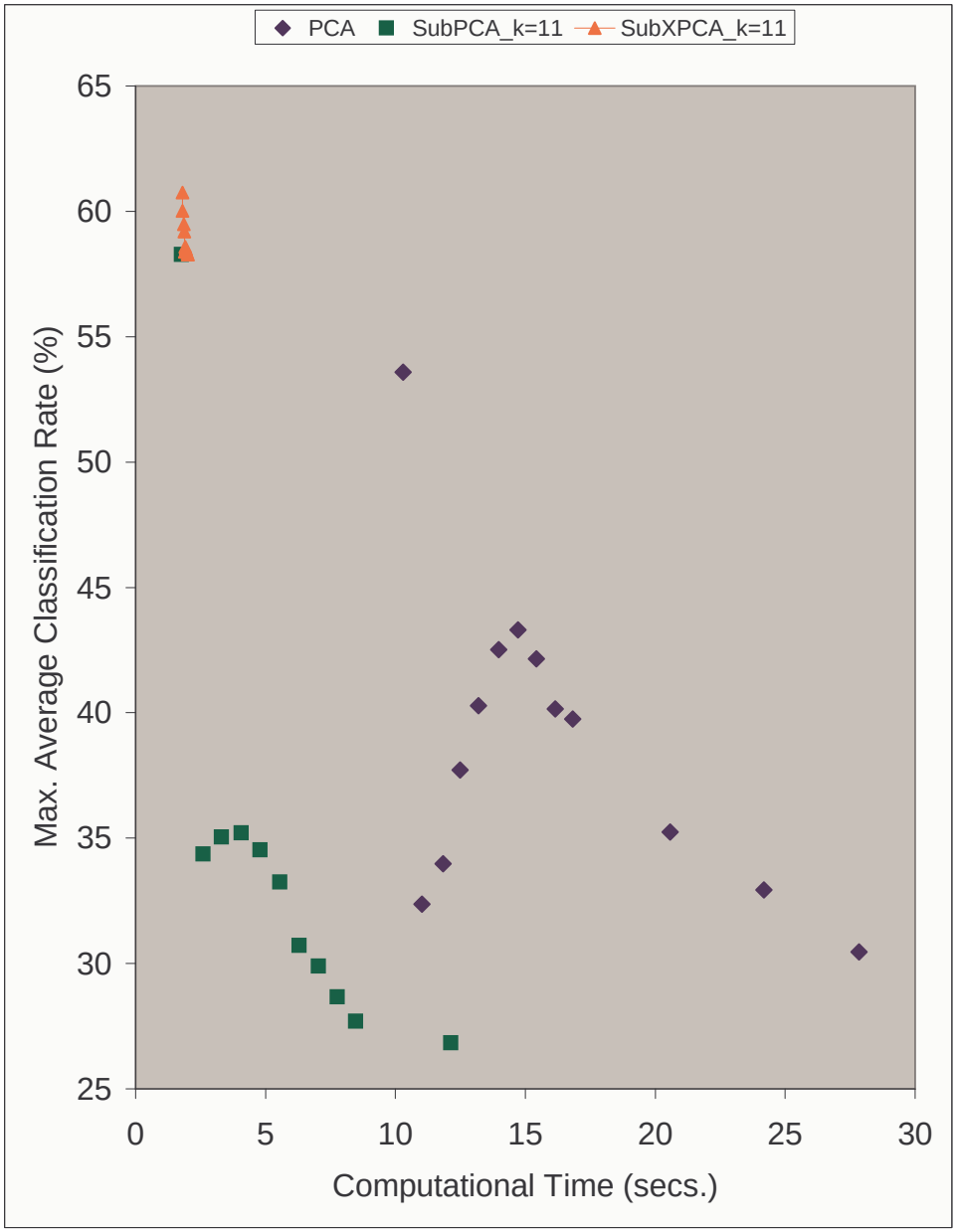


Figure 8.4: *Comparison of PCA, SubPCA and SubXPCA based subspace classifiers with respect to both computational time and classification rate for UCI Musk data.* SubXPCA based method (FP-SC) forms all its points at the top-left corner of the plot, which is the indication of high classification rate at less computational time. Other two methods have the points concentrated away from top-left corner which indicates that both PCA and SubPCA based classifiers either show lower classification rate or high computational time or both.

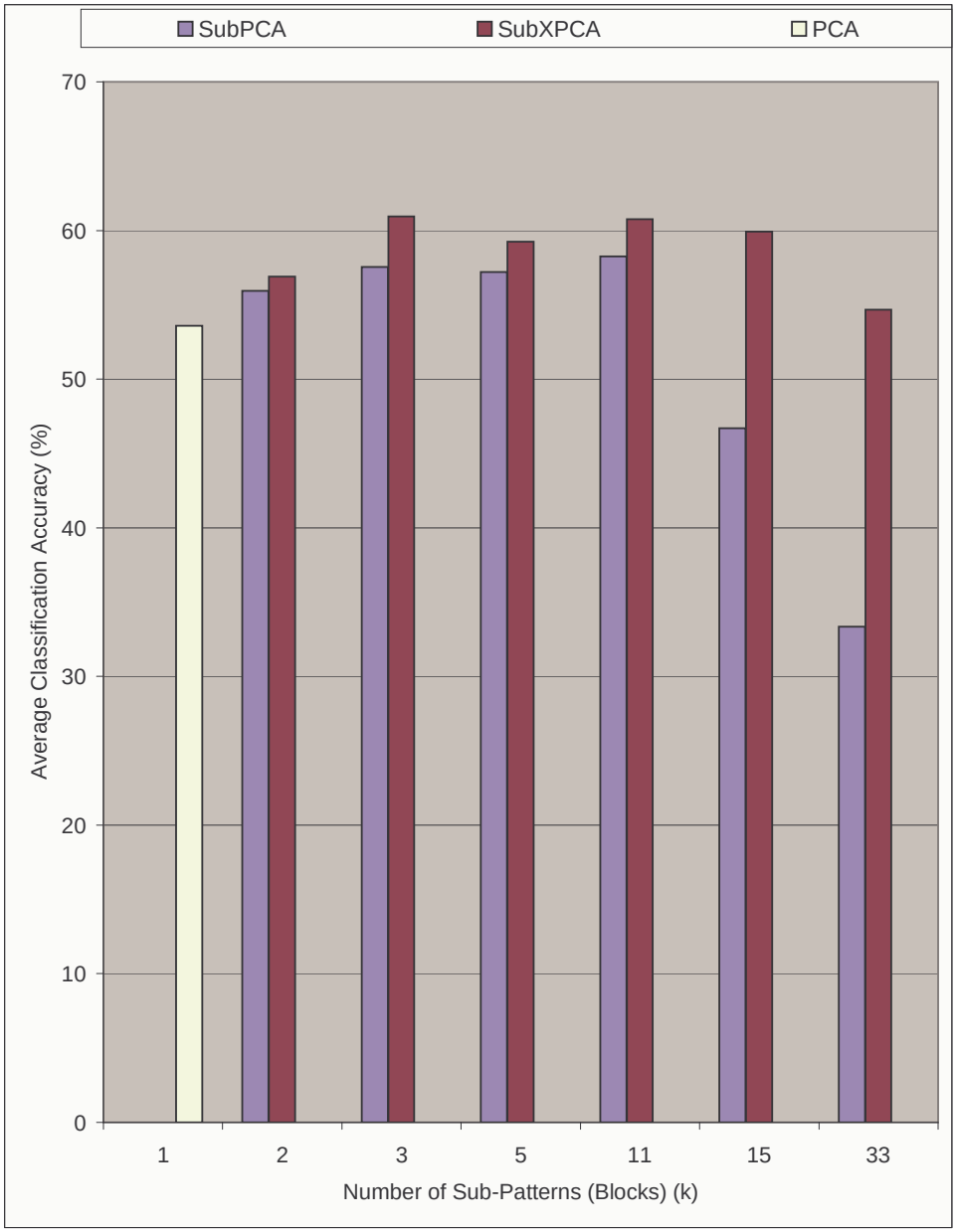


Figure 8.5: *Comparison of average classification rates with varied number of sub-patterns (blocks) for UCI Musk data.* SubXPCA based subspace classifier (FP-SC) consistently shows good performance as compared to SubPCA based subspace classifier with different number of blocks. Please note that PCA based classifier shows lower classification rate as compared to (i) FP-SC classifier (with all k values) and (ii) SubPCA based classifier (except for $k = 15, 33$).

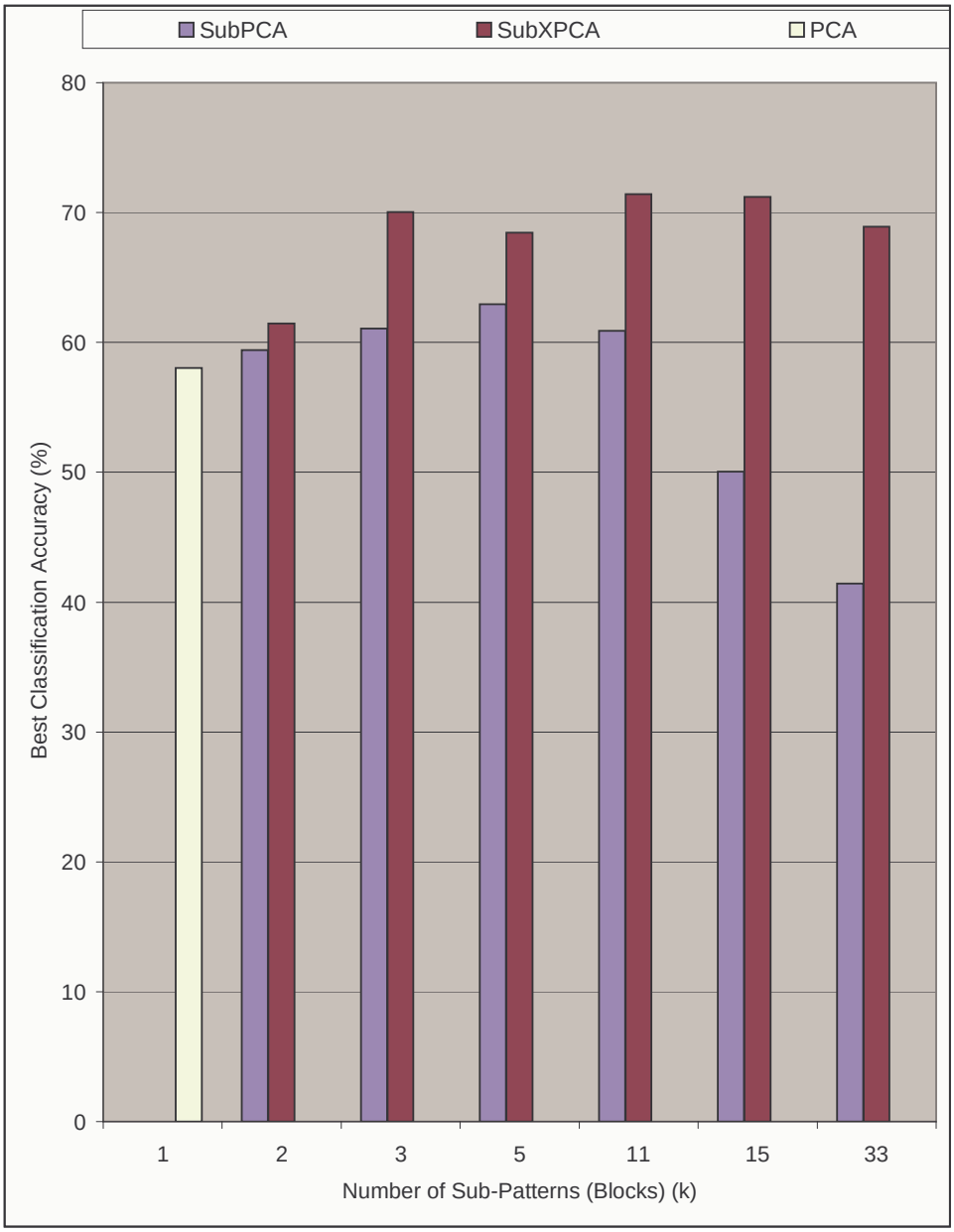


Figure 8.6: *Comparison of best classification rates with varied number of sub-patterns (blocks) for UCI Musk data.* SubXPCA based subspace classifier (FP-SC) consistently shows good performance as compared to SubPCA based subspace classifier with different number of blocks. Please note that PCA based classifier shows lower classification rate as compared to (i) FP-SC classifier (with all k values) and (ii) SubPCA based classifier (except for $k = 15, 33$).

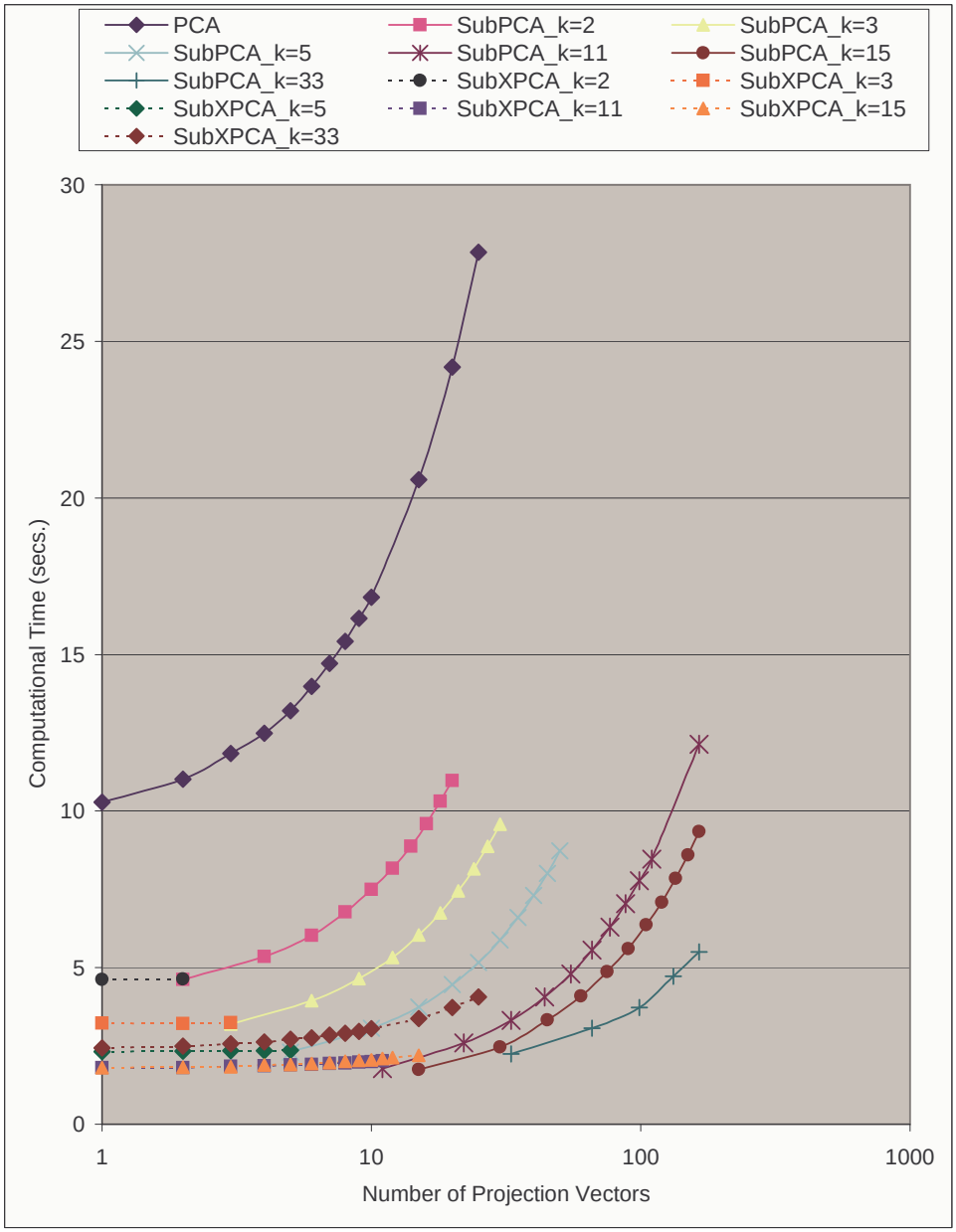


Figure 8.7: *Comparison of computational time with different number of blocks for UCI Musk data.* It is to be noted that SubXPCA based subspace classifier (FP-SC) is computationally more efficient as compared to other two methods. Also SubPCA based method shows less computational time over PCA based subspace classifier.

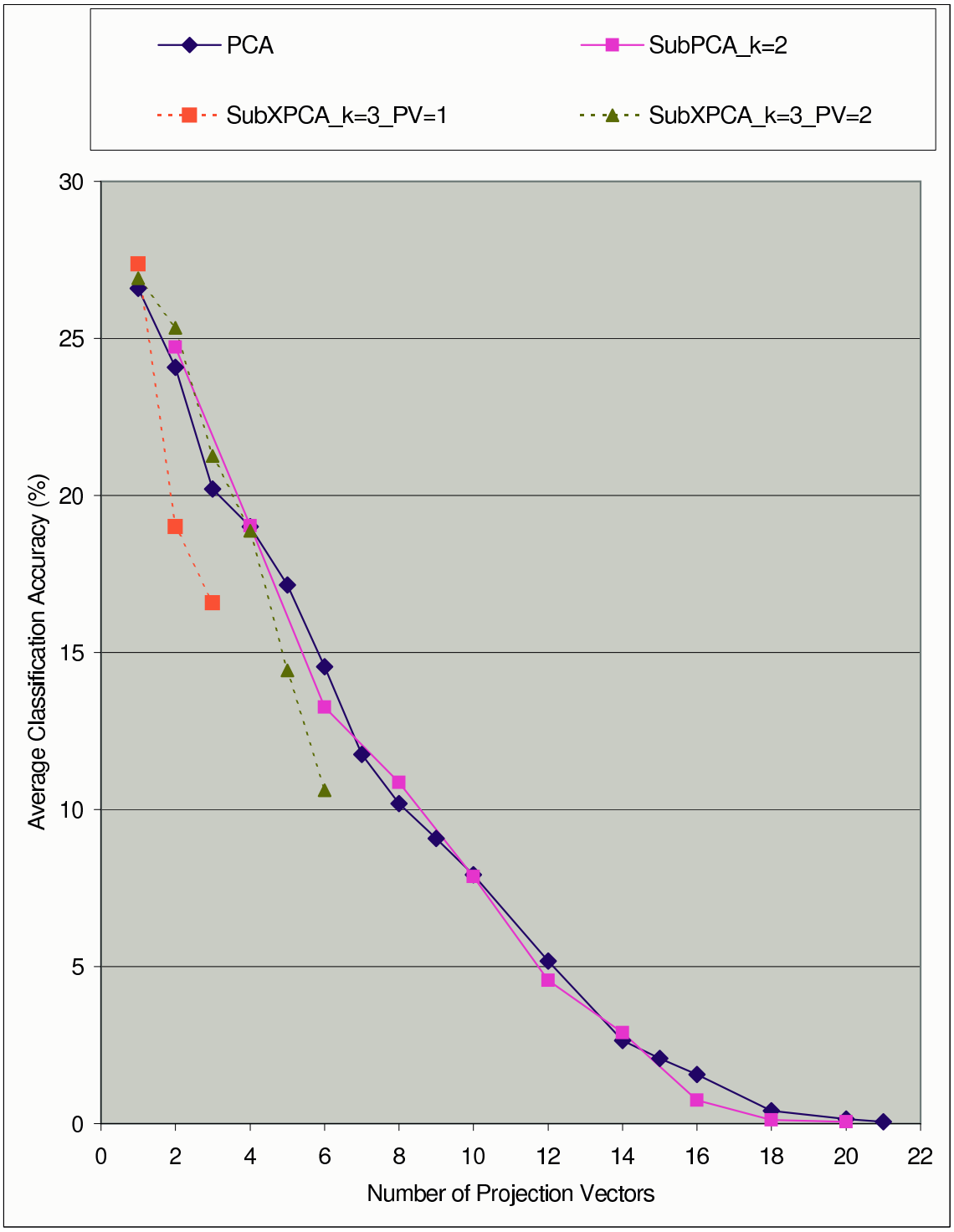


Figure 8.8: *Comparison of average classification rates for UCI Waveform data.* SubXPCA based subspace classifier shows slight improvement over PCA based method with respect to its maximum of plotted classification rates. However, SubXPCA based method shows nearly 2% higher classification as compared to SubPCA based subspace classifier with respect to its maximum of plotted classification rates. SubXPCA uses 1 and 2 projection eigen vectors (PVs) per sub-pattern set.

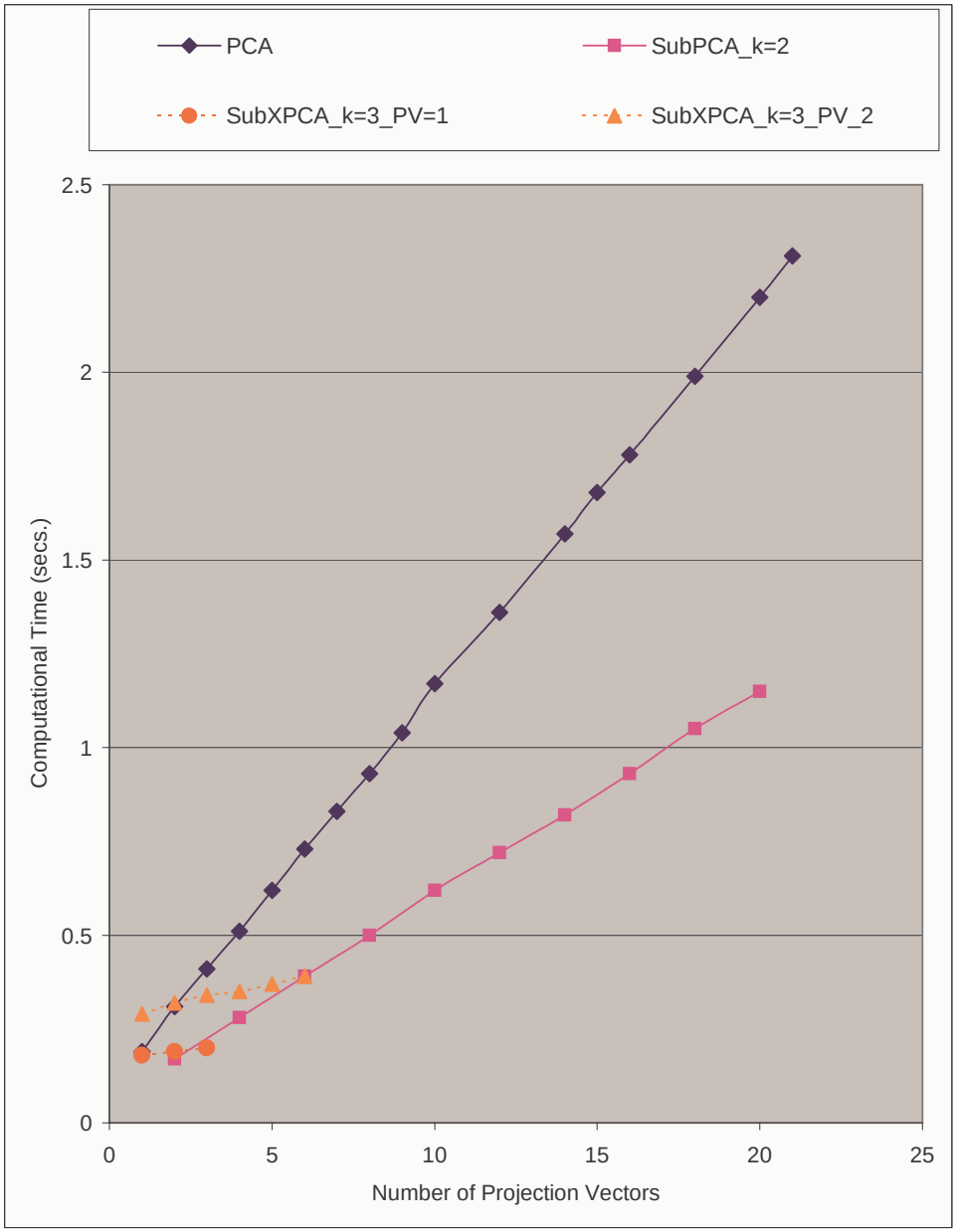


Figure 8.9: *Comparison of computational time for UCI Waveform data.* SubXPCA based subspace classifier (FP-SC) shows less computational time as compared to PCA and SubPCA based subspace classifiers.

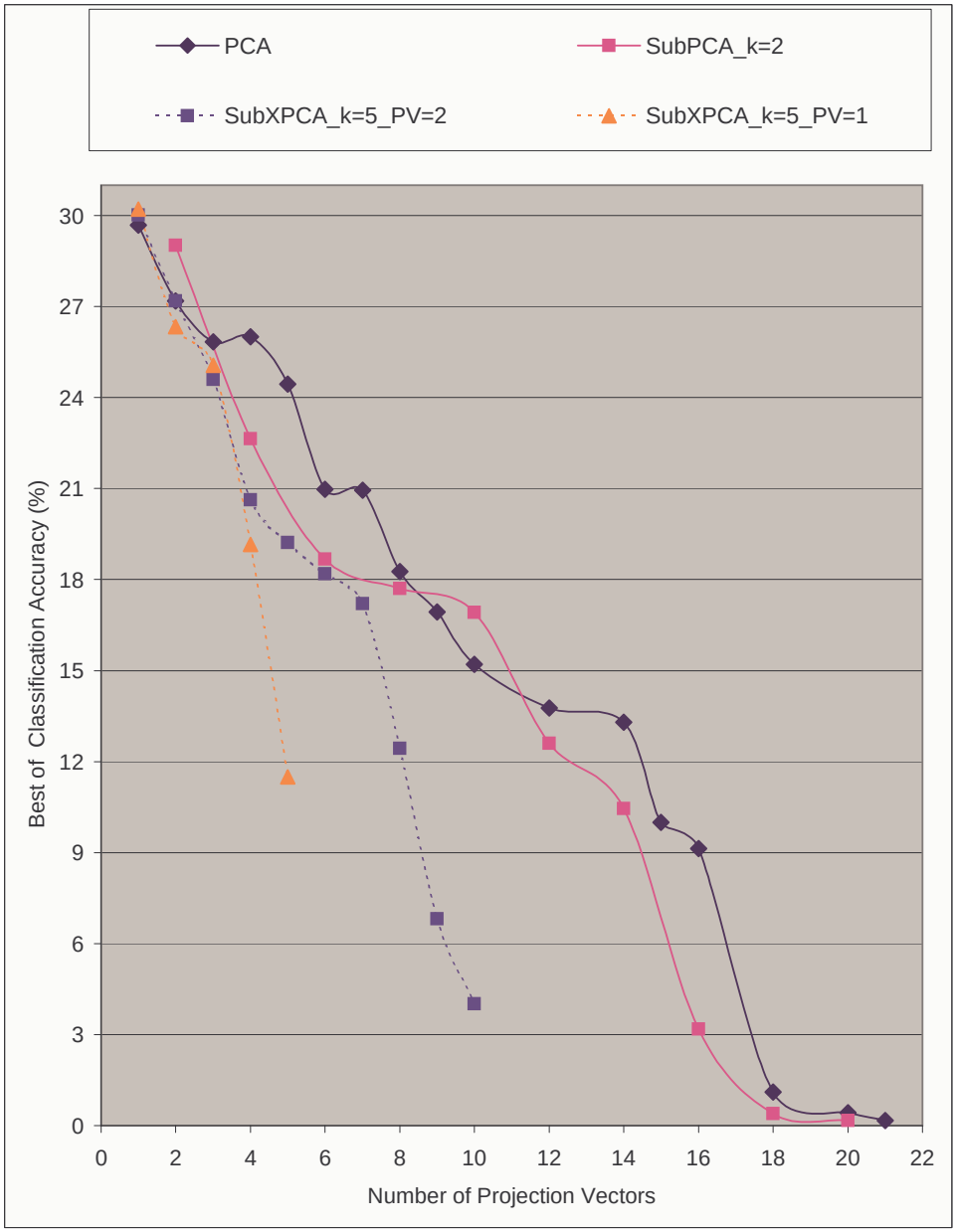


Figure 8.10: *Comparison of best classification rates for UCI Waveform data.* SubXPCA based subspace classifier shows slight improvement over PCA and SubPCA based methods with respect to its maximum of plotted classification rates. SubXPCA uses 1 and 2 projection eigen vectors (PVs) per sub-pattern set.

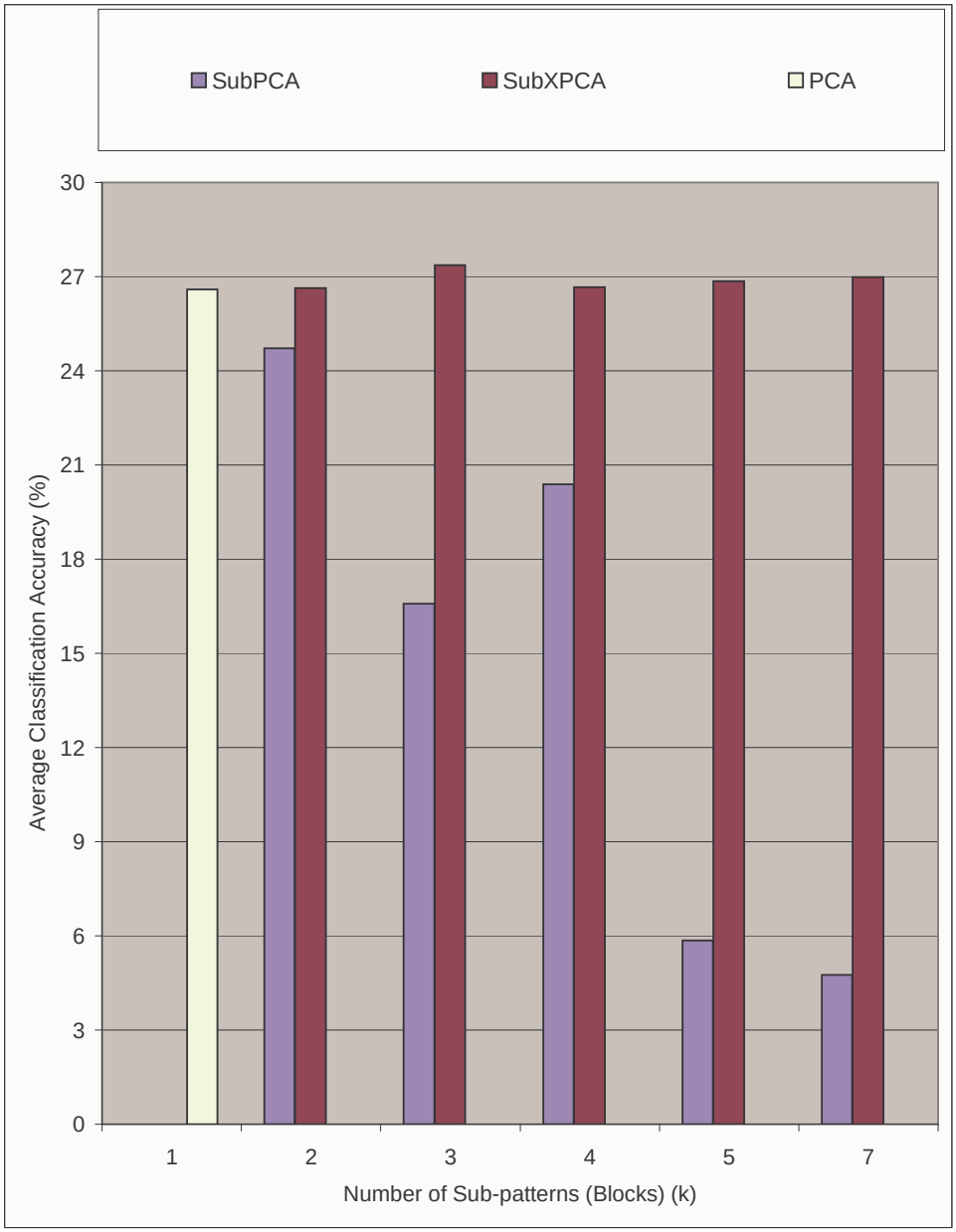


Figure 8.11: *Comparison of average classification rates with varied number of sub-patterns (blocks) for UCI Waveform data.* SubXPCA based subspace classifier (FP-SC) consistently shows good performance as compared to SubPCA based subspace classifier with different number of blocks. SubXPCA based method shows slight improvement over PCA based subspace classifier. It is clear that SubPCA based classifier shows lower performance as compared to other two methods.

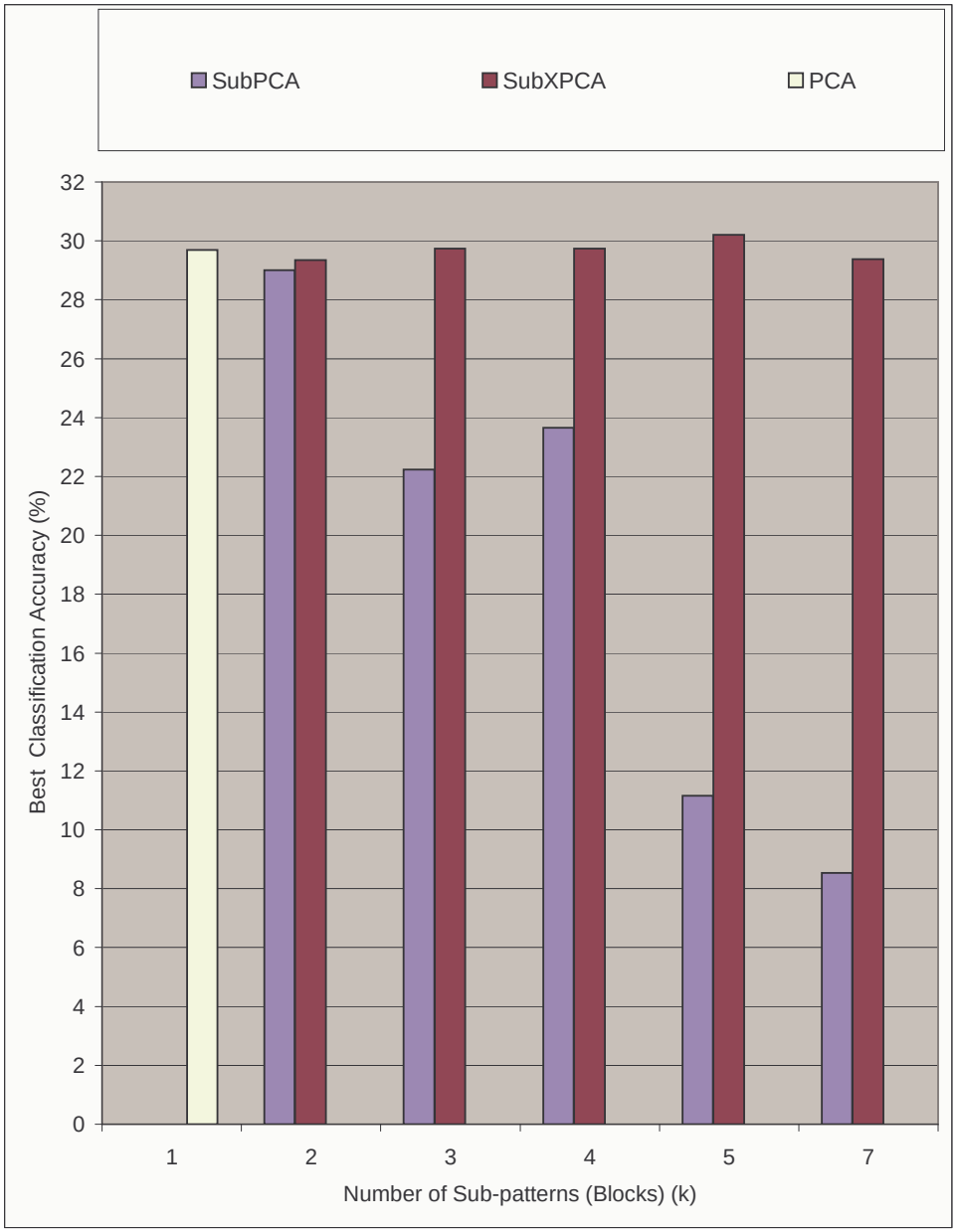


Figure 8.12: *Comparison of best classification rates with varied number of sub-patterns (blocks) for UCI Waveform data.* SubXPCA based subspace classifier (FP-SC) consistently shows good performance as compared to SubPCA based subspace classifier with different number of blocks. SubXPCA based method shows slight improvement over PCA based subspace classifier. It is clear that SubPCA based classifier shows lower performance as compared to other two methods.

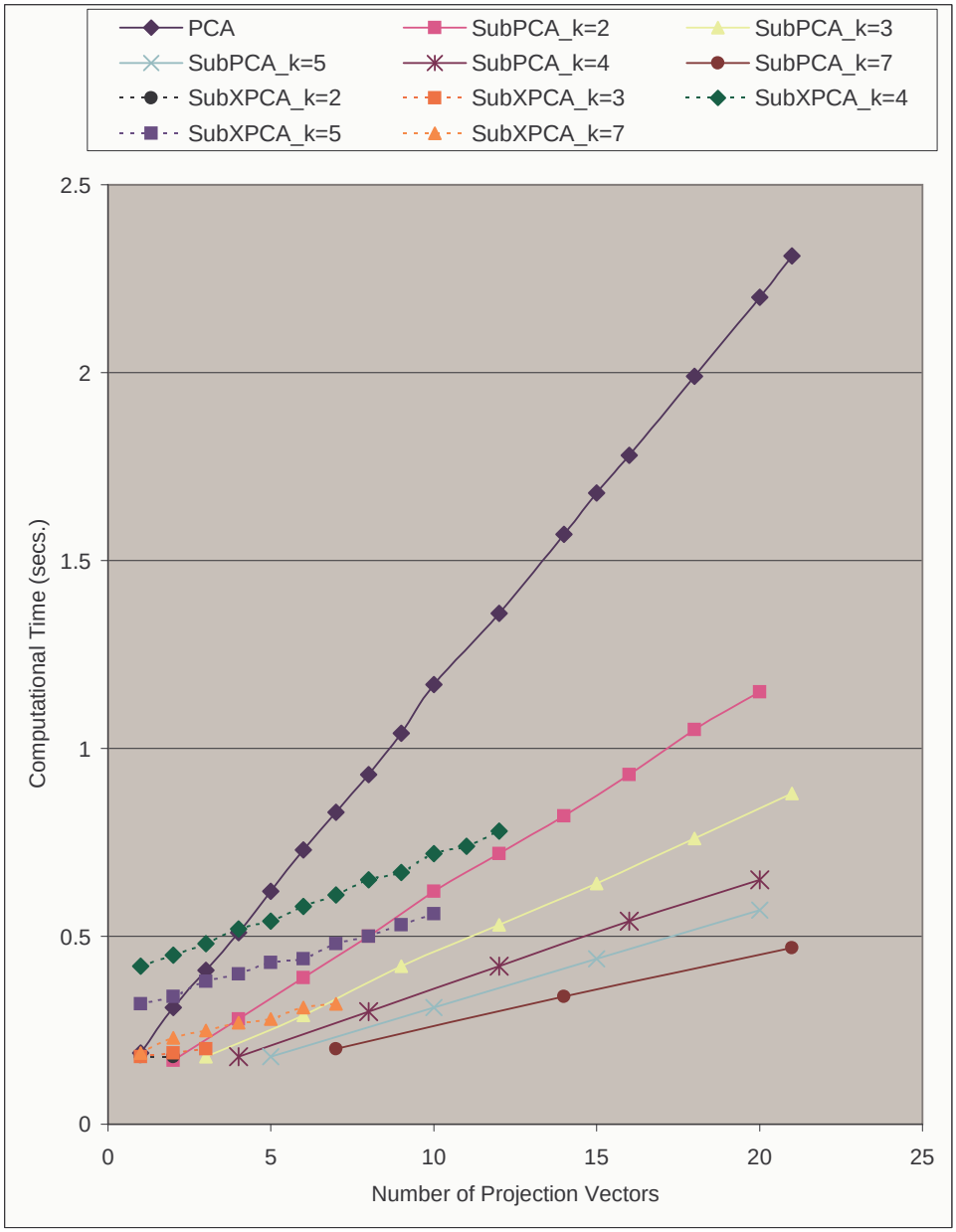


Figure 8.13: Comparison of computational time with different number of blocks for UCI Waveform data. It is to be noted that SubXPCA based subspace classifier (FP-SC) is computationally more efficient as compared to other two methods. Also SubPCA shows less computational time over PCA based subspace classifier.

However, FP-SC (SubXPCA based) overcomes these problems, by capturing local information in addition to global structure. Therefore, SubXPCA based subspace classifier (FP-SC) shows its superiority by taking advantage of merits of both local and global PCA based feature extraction methods.

8.7 Summary

We proposed a feature partitioning approach to subspace classification, FP-SC (SubXPCA based subspace classifier). FP-SC outperforms other classical PCA based subspace methods in terms of time complexity and classification. Also, FP-SC shows superiority in terms of classification as compared to SubPCA (an existing FP-PCA method) based subspace classifier. In addition, FP-SC classifier performs well in terms of computational time as compared to SubPCA based subspace classifier if FP-SC uses less number of principal components than SubPCA based method. Classical PCA based subspace methods use classical PCA to compute subspace, which may take large amount of time if the dimensionality of the data is high. Unlike classical PCA based subspace methods, FP-SC uses feature partitioning approach to compute subspace, where it reduces the time complexity enormously and improves the classification rate as well. The proposed method may be extensively used for face recognition, palmprint recognition, OCR applications, etc.

Chapter 9

Conclusions and Future Work

In this work, we brought out the existing feature partitioning based PCA (FP-PCA) approaches in a common framework, which facilitates to understand these approaches easily. Further basic issues to be addressed in the context of partitioning were identified. To address the loss of covariance structure, impact of feature order dependency, impact of overlapping patterns, we proposed a novel FP-PCA approach, called SubXPCA. SubXPCA method was proved to be superior as compared to classical PCA (a global PCA) and SubPCA (an FP-PCA method). Further, we extend the feature partitioning concept to image data by proposing two approaches SIM-PCA and FLPCA. These approaches reduce computational requirements better than PCA, modPCA (an FP-PCA method), 2DPCA and improves recognition rate. Both SubXPCA and FLPCA were proved to be flexible enough to adapt to local or global variations of patterns.

Subsequently, we performed a theoretical analysis of FP-PCA approaches and established properties of FP-PCA methods. From our study, we understood that FP-

PCA-Type-III (e.g. SubXPCA and FLPCA) and FP-PCA-Type-IV methods move closer to their corresponding Holistic PCA methods (e.g. classical PCA, 2DPCA) in terms of summarization of variance with increasing number of local features. FP-PCA-Type-I (e.g. SubPCA and similar methods) and FP-PCA-Type-II (e.g. mod-PCA and similar methods) show less summarization of variance as compared to Holistic PCA, FP-PCA-Type-III and FP-PCA-Type-IV methods. We applied feature partitioning idea to correlation connected cluster analysis and subspace classification and the proposed approaches were proved to be more efficient than traditional methods. The proposed approaches may be extensively used in Biometrics applications, Computer vision, Data mining, Fault detection applications, Remote sensing, Change detection applications, Real-time surveillance applications, etc.

The work presented in this thesis was presented at several forums and has undergone peer review process. The results are encouraging. However there is always room for improvement. The work may be extended in future to extract non-linearity by using Kernel PCA methods and other non-linear methods. Other possible extension is to investigate incremental versions of FP-PCA methods to exploit characteristics of Artificial neural networks. Another direction is to improve FP-PCA methods to handle outliers and missing values data (for e.g. using fuzzy logic). One may bring out novel FP-PCA methods which address the various feature partitioning issues proposed in this thesis.

Bibliography

[1] Jong-Hoon Ahn, Seungjin Choi, and Jong-Hoon Oh. A new way of PCA: integrated-squared-error and EM algorithms. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 777–780, May 2004.

[2] Kyungim Baek, Bruce A. Draper, J. Ross Beveridge, and Kai She. PCA versus ICA: A comparison on the FERET data sets. *www.cs.colostate.edu/evalfacerec/papers/cvprip02.pdf*, 2002.

[3] P. M. Baggenstoss. Class-specific classifier: Avoiding the curse of dimensionality. *IEEE A & E Systems Magazine Part 2: Tutorials*, 19 No. 1:37–52, Jan. 2004.

[4] P. F. Baldi and K. Hornik. Learning in linear neural networks: A survey. *IEEE Transactions on Neural Networks*, 6:837–858, 1995.

[5] M. S. Bartlett. A note on the multiplying factors for various x^2 approximations. *Journal of Royal Statistical Society Series B*, 16:296–298, 1954.

[6] Morro Bay. Image processing and interpretation. *http://rst.gsfc.nasa.gov/Sect1/Sect1-14.html*, 128:1671–1675, 2004.

[7] J. P. Belhumeur, J. Hespanha, and D. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

[8] Samy Bengio and Yoshua Bengio. Taking on the curse of dimensionality in joint distributions using neural networks. *IEEE Transactions on Neural Networks*, 11, No. 3:550–557, May 2000.

[9] J. Ross Beveridge. The geometry of LDA and PCA classifier illustrated with 3D examples. Technical Report 01-101, Computer Science Department Colorado State University, Colorado, May 2001.

[10] J. Ross Beveridge, Kai She, Bruce A. Draper, and Geof H. Givens. A nonparametric statistical comparison of principal component and linear discriminant subspaces for face recognition. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'01)*, page 535, 2001.

[11] C. M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.

[12] Christian Bohm, Karin Kailing, P. Kroeger, and Arthur Zimek. Computing clusters of correlation connected objects. In *Proceedings of ACM SIGMOD International Conference on Management of Data, France*, pages 455–466, 2004.

[13] H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*, 59 No. 4:291–294, 1988.

[14] Jorge Cadima, J. Orestes Cerdeira, and Manuel Minhoto. Computational aspects of algorithms for variable selection in the context of principal components. *Computational Statistics & Data Analysis*, 47:225–236, 2004.

[15] J. Cardoso. Blind signal separation: Statistical principles. *Proc. IEEE*, 863:2009–2025, 1998.

[16] Miguel A. Carreira-Perpinan. A review of dimension reduction techniques. Technical Report CS 96-09, Dept. of Computer Science, University of Sheffield, UK, 1997.

[17] Chanchal Chatterjee, Zhengjiu Kang, and Vwani P. Roychowdhury. Algorithms for accelerated convergence of adaptive PCA. *IEEE Transactions on Neural Networks*, 11, No. 2:338–355, Mar. 2000.

[18] Haifeng Chen. Principal component analysis with missing data and outliers. *www.caip.rutgers.edu/riul/research/tutorials/tutorialrpca.pdf*, 2002.

[19] Liang-Hwa Chen and Shyang Chang. An adaptive learning algorithm for principal component analysis. *IEEE Transactions on Neural Networks*, 6. No. 5:1255–1263, Sep. 1995.

[20] Pinyuen Chen. A confidence interval for the number of principal components. *Journal of Statistical Planning and Inference*, 136, Issue 8:2630–2639, 2006.

[21] Songcan Chen and Yulian Zhu. Subpattern-based principal component analysis. *Pattern Recognition*, 37:1081–1083, 2004.

[22] Songcan Chen, Yulian Zhu, Daoqiang Zhang, and Jing-Yu Yang. Feature extraction approaches based on matrix pattern: MatPCA and MatFLDA. *Pattern Recognition Letters*, 26:1157–1167, 2005.

[23] C. Wu Chien. Discriminant wavelet faces and nearest feature classifiers for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12):1644–1649, 2002.

[24] Tat-Jun Chin and David Suter. Incremental kernel principal component analysis. *IEEE Transactions on Image Processing*, 16, No. 6:1662–1674, Jun. 2007.

[25] CMU. CMU face data set. *http://kdd.ics.uci.edu/databases/faces/faces.data.html*.

[26] P. Comon. Independent component analysis, a new concept? *Signal Processing*, 36 No. 3:287–314, 1994.

[27] Tee Connie, Andrew Teoh Beng Jin, Michael Goh Kah Ong, and David Ngo Chek Ling. An automated palmprint recognition system. *Image and Vision Computing*, 23:501–515, 2005.

[28] Dietmar Cordes and R. N. Rajesh. Estimation of the intrinsic dimensionality of FMRI data. *NeuroImage*, 29:145–154, 2006.

[29] J. A. Cumming and D. A. Wooff. Dimension reduction via principal variables. *Computational Statistics & Data Analysis*, 52:550–565, 2007.

[30] Thomas R. Cundari, Costel Sarbu, and Horia F. Pop. Robust fuzzy principal component analysis (FPCA)-a comparative study concerning interaction of carbon-hydrogen bonds with molybdenum-oxo bonds. *Journal of Chemical Information and Computer Sciences*, 42:1363–1369, 2002.

[31] C. Darken, J. Chang, and J. Moody. Learning rate schedules for faster stochastic gradient search. In *Proceedings of the IEEE-SP Workshop on Neural Networks for Signal Processing, Denmark*, pages 3–12, Sep. 1992.

[32] R. N. Dave and Sumit Sen. Robust fuzzy clustering of relational data. *IEEE Transactions on Fuzzy Systems*, 10, No. 6:713–727, 2002.

[33] Sampath Deegalla and Henrik Bostrom. Reducing high-dimensional data by principal component analysis vs. random projection for nearest neighbor classification. In *Proceedings of the 5th International Conference on Machine Learning and Applications*, pages 245–250, Dec. 2006.

[34] Thierry Denoeux and Marie-Helene Masson. Principal component analysis of fuzzy data using autoassociative neural networks. *IEEE Transactions on Fuzzy Systems*, 12, No. 3:336–349, Jun. 2004.

[35] Pierre A. Devjver and Josef Kitler. *Pattern Recognition: A Statistical Approach.* Prentice Hall International, Englewood Cliffs, New Jersey, 1982.

[36] K. I. Diamantaras. *Principal component learning networks and applications.* PhD thesis, Princeton University, Greece, 1992.

[37] K. I. Diamantaras and S. Y. Kung. *Principal component neural networks: Theory and applications*. Adaptive and Learning Systems for Signal Processing, Communications, and Control. New York: Wiley, 1996.

[38] Bruce A. Draper, Daniel L. Elliott, Jeremy Hayes, and Kyungim Baek. EM in high-dimensional spaces. *IEEE Transactions on Systems, Man, and Cybernetics Part B*, 35, No. 3:571–577, Jun. 2005.

[39] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification Second Edition*. John Wiley & sons (ASIA) Pte Ltd, 2 Clementi Loop 02-01 Singapore 129809, 2002.

[40] M. Ester, H. P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clustering in large spatial databases. In *Proceedings of International Conference on Knowledge Discovery and Data Mining, Portland*, pages 226–231, Aug. 1996.

[41] L. Ferre. Selection of components in principal component analysis: a comparison of methods. *Computational Statistics and Data Analysis*, 19:669–682, 1995.

[42] J. Fortuna, P. Quick, and D. Capson. A comparison of subspace methods for accurate position measurement. In *Proceedings of 6th IEEE Southwest Symposium on Image Analysis and Interpretation*, pages 16–20, 2004.

[43] D. Fradkin and D. Madigan. Experiments with random projections for machine learning KDD 2003. In *Proceedings of 9th ACM SIGMOD International Conference on Knowledge Discovery and Data mining*, pages 517–522, 2003.

[44] S. B. Franklin, D. J. Gibson, P. A. Robertson, J. T. Pohlmann, and J. S. Fralish. Parallel analysis: a method for determining significant principal components. *Journal of Vegetarian Sciences*, 6:99–106, 1995.

[45] Clement Fredembach, Michael Schroder, and Sabine Susstrunk. Eigenregions for image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26, No. 12:1645–1649, 2004.

[46] J. Friedman and J. Meulman. Clustering objects on subsets of attributes. *Journal of Royal Statistical Society*, 66 No. 4:815–849, 2004.

[47] J. H. Friedman. Exploratory projection pursuit. *Journal of American Statistical Association*, 82:249–266, 1987.

[48] K. S. Fu. *Syntactic Pattern Recognition and Applications*. Prentice Hall, Englewood cliffs, 1982.

[49] Keinosuke Fugunaga. *Introduction to Statistical Pattern Recognition*. Academic Press Inc. (London) Ltd., 24/8 Oval Road, London NW1 7DD, 1972.

[50] Jr. H. G. Gauch. Noise reduction by eigenvector ordination. *Ecology*, 63:1643–1649, 1982.

[51] Stephane Girard and Serge Iovleff. Auto-associative models and generalized principal component analysis. *Journal of Multivariate Analysis*, 93:21–39, 2005.

[52] Geof Givens, J. Ross Beveridge, Bruce A. Draper, and David Bolme. A statistical assessment of subject factors in the PCA recognition of human faces.

In *Proceedings of Statistical Analysis in Computer Vision Workshop*, page 96, 2003.

[53] Rajkiran Gottumukkal and Vijayan K. Asari. An improved face recognition technique based on modular PCA approach. *Pattern Recognition Letters*, 25:429–436, 2004.

[54] L. Guttman. Some necessary conditions for common factor analysis. *Psychometrika*, 19:149–161, 1954.

[55] Wim D. haes, Dirk van Dyck, and Xavier Rodet. PCA-based branch and bound search algorithms for computing k nearest neighbors. *Pattern Recognition Letters*, 24:1437–1451, 2003.

[56] P. Hall, D. Marshall, and R. Martin. Merging and splitting eigenspace models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22, Issue 9:1042–1049, Sep. 2000.

[57] Jiawei Han and Michline Kambler. *Data mining Concepts and Techniques*. Morgan Kaufmann Publishers An imprint of Elsevier Science, 340 Pine Street, Sixth Floor, San Francisco, CA, USA, 2001.

[58] Yong He, Xiaoli Li, and Xunfei Deng. Discrimination of varieties of tea using near infrared spectroscopy by principal component analysis and BP model. *Journal of Food Engineering*, 79:1238–1242, 2007.

[59] D. O. Hebb. *Organization of Behavior*. Wiley, New York, 1949.

[60] J. Hertz, A. Krogh, and R. G. Palmer. *Introduction to the Theory of Neural Computation*. Adaptive and Learning Systems for Signal Processing, Communications, and Control. Addison-Wesley, Redwood City, CA, 1991.

[61] Isao Higuchi and Shinto Eguchi. Robust principal component analysis with adaptive selection for tuning parameters. *Journal of Machine Learning Research*, 5:453–471, 2004.

[62] Katsuhiro Honda and Hidetomo Ichihashi. Linear fuzzy clustering techniques with missing values and their application to local principal component analysis. *IEEE Transactions on Fuzzy Systems*, 12, No. 2:183–193, Apr. 2004.

[63] Yin Hongtao, Fu Ping, and Meng Shengwei. Face recognition with DWT and two-dimensional principal component analysis. In *Proceedings of The Eighth International Conference on Electronic Measurement and Instruments*, pages I–86–I–89, Xian, China, Aug. 2007.

[64] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:498–520, 1933.

[65] William W. Hsieh. Nonlinear principal component analysis by neural networks. *Tellus*, 53A:599–615, 2001.

[66] Kou-Yuan Huang. Neural networks for seismic principal components analysis. *IEEE Transactions on Geoscience and Remote Sensing*, 37, No. 1:297–311, 1999.

[67] Rodrigo A. Ibata and Michael J. Irwin. Discrete classification with principal

component analysis: Discrimination of giant and dwarf spectra in k stars. *The Astronomical Journal*, 113:1865, 1997.

[68] Hidetomo IchihashI and Katsuhiro Honda. Fuzzy robust PCA with intra-sample outlier process. *Proceedings of the Annual Meeting of Biomedical Fuzzy Systems Association*, 17:101–104, 2004.

[69] T. Iijima, H. Genchi, and K. Mori. A theory of character recognition by pattern matching method. In *Proceedings of 1st International Joint Conference on Pattern Recognition*, pages 50–56, Washington, D.C., 1973.

[70] D. A. Jackson. Stopping rules in principal component analysis: a comparison of heuristical and statistical approaches. *Ecology*, 74:2204–2214, 1993.

[71] Anil K. Jain. *Fundamentals of digital image processing.* Prentice-Hall, Englewood Cliffs, NJ, 1989.

[72] Anil K. Jain, Robert P. W. Duin, and Jianchang Mao. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22, No. 1:4–37, 2000.

[73] Anil K. Jain, Jianchang Mao, and K. M. Mohiuddin. Artificial neural networks: A tutorial. *IEEE Computer*, pages 31– 44, 1996.

[74] N. R. Jeffers. Two case studies in the application of principal component analysis. *Applied Statistics*, 16:225 – 236, 1967.

[75] Richard A. Johnson and Dean W. Wichern. *Applied Multivariate Statistical Analysis, Third edition.* Prentice-Hall of India, New Delhi, 2001.

[76] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verelag, New York, 2002.

[77] Hidehiko Kamiya. A class of robust principal component vectors. *Journal of Multivariate Analysis*, 77:239–269, 2001.

[78] J. Karhunen and J. Joutsensalo. Tracking of sinusoidal frequencies by neural network learning algorithms. In *Proceedings of the International Conference on Acoustics, Speech, Signal Process, Toronto*, pages 3177–3180, 1991.

[79] Juha Karhunen and Jyrki Joutsensalo. Generalizations of principal component analysis, optimization problems, and neural networks. *Neural Networks*, 8, No. 4:549–562, 1995.

[80] M. M. Khan, M. Y. Javed, and M. A. Anjum. Face recognition using sub-holistic PCA. In *Proceedings of First International Conference on Information and Communication Technologies*, pages 152–157, Aug. 2005.

[81] Chunghoon Kim and Chong-Ho Choi. Image covariance-based subspace method for face recognition. *Pattern Recognition*, 40:1592–1604, 2007.

[82] Dong Kook Kim and Nam Soo Kim. Rapid speaker adaptation using probabilistic principal component analysis. *IEEE Signal Processing Letters*, 8 No. 6:180–183, Jun. 2001.

[83] Kwang In Kim, Keechul Jung, and Hang Joon Kim. Face recognition using kernel principal component analysis. *IEEE Signal Processing Letters*, 9, No. 2:40–42, Feb. 2002.

[84] Sang-Woon Kim and B. John Oommen. On utilizing search methods to select subspace dimensions for kernel-based nonlinear subspace classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27, No. 1:136–141, 2005.

[85] M. Kirby and L. Sirovich. Application of the KL procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12 No. 1:103–108, 1990.

[86] T. Kohonen. *Self-Organizing Maps*. Springer Series in Information Sciences, Vol. 30, Berlin, 1995.

[87] T. Kohonen, G. N. Meth, K. J. Bry, M. Jalanko, and H. Riittinen. Classification of phonemes by learning subspaces. Technical Report TKK-F-A348, Helsinki University of Technology, Espoo Finland, 1978.

[88] Hui Konga, Lei Wang, Eam Khwang Teoh, Xuchun Li, Jian-Gang Wang, and Ronda Venkateswarlu. Generalized 2D principal component analysis for face image representation and recognition. *Neural Networks*, 18:585–594, 2005.

[89] H. Kramer and M. Mathews. A linear coding for transmitting a set of correlated signals. *IRE Transactions on Information Theory IT-2*, pages 41– 46, 1956.

[90] M. A. Kramer. Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal*, 37:233–243, 1991.

[91] S. Y. Kung and K. I. Diamantaras. Auto aussociative neural network learning algorithm for adaptive principal component extraction (APEX). In *Proceedings*

of ICASSP (Albuquerque, NM), pages 861–864, Washington, DC, USA, Apr. 1990.

[92] S. Y. Kung, K. I. Diamantaras, and J. S. Taur. Adaptive principal component extraction (APEX) and applications. *IEEE Transactions on signal processing*, 42 No. 5:1202–1217, 1994.

[93] Jorma Laaksonen. *Subspace Classifiers in Recognition of Handwritten Digits*. PhD thesis, Helsinki University of Technology, Finland, May 1997.

[94] D. N. Lawley. Tests of the significance for the latent roots of covariance and correlation matrices. *Journal of Royal Statistical Society Ser. B*, 43:128–136, 1956.

[95] F. R. Lawrence and G. R. Hancock. Conditions affecting integrity of a factor solution under varying degrees of overextraction. *Educational and Psychological Measurement*, 59:549–579, 1999.

[96] Stephanie Ledauphin, Mohamed Hanafi, and El Mostafa Qannari. Simplification and signification of principal components. *Chemometrics and Intelligent Laboratory Systems*, 74:277–281, 2004.

[97] T. W. Lee. *Independent Component Analysis*. Kluwer Academic Publishers, Dordrech, 1998.

[98] P. Legendre and L. Legendre. *Numerical Ecology. 2nd English Edition*. Elsevier Science BV, Amsterdam, 1998.

[99] Boaz Lerner, Hugo Guterman, Mayer Aladjem, and Itshak Dinstein. A comparative study of neural network based feature extraction paradigms. *Pattern Recognition Letters*, 20:7–14, 1999.

[100] B. Li and Y. Liu. When eigenfaces are combined with wavelets. *Knowledge-Based Systems*, 15 (5-6):343–347, 2002.

[101] Chee-Peng Lim, Jenn-Hwai Leong, and Mei-Ming Kuan. A hybrid neural network system for pattern classification tasks with missing features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27 No. 4:648–653, 2005.

[102] Chengjun Liu. Gabor-based kernel PCA with fractional power polynomial models for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26, No. 5:572–581, May 2004.

[103] Guangming Lu, David Zhang, and Kuanquan Wang. Palmprint recognition using eigenpalms features. *Pattern Recognition Letters*, 24:1463–1467, 2003.

[104] Arnaz Malhi and Robert X. Gao. PCA-based feature selection scheme for machine defect classification. *IEEE Transactions on Instrumentation and Measurement*, 53, No. 6:1517–1525, Dec. 2004.

[105] B. J. F. Manly. *Randomization, Bootstrap and Monte Carlo Methods in Biology. 2nd Edition.* Chapman and Hall, London, 1997.

[106] Aleix M. Martinez. Recognition of partially occluded and/or imprecisely localized faces using a probabilistic approach. In *Proceedings of IEEE conference on Computer Vision and Pattern Recognition*, pages 712–717, 2000.

[107] Aleix M. Martinez. Recognition of imprecisely localized, partially occluded and expression variant faces from a single sample per class. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24 , Issue 6:748–763, Jun. 2002.

[108] Aleix M. Martonez and Avinash C. Kak. PCA versus LDA. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23, No. 2:228–233, 2001.

[109] Jicheng Meng and Wenbin Zhang. Volume measure in 2DPCA-based face recognition. *Pattern Recognition Letters*, 28:1203–1208, 2007.

[110] Sebastian Mika, Bernhard Scholkopf, Alex Smola, Klaus-Robert Muller, Matthias Scholz, and Gunnar Ratsch. Kernel PCA and de-noising in feature spaces. In *Proceedings of the 1998 conference on Advances in neural information processing systems II*, pages 536–542, 1998.

[111] Vo Dinh Minh Nhat and Sung Young Lee. Two-dimensional weighted PCA algorithm for face recognition. In *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 219–223, Espoo, Finland, Jun. 2005.

[112] Sandro Nicole. Feedforward neural networks for principal components extraction. *Computational Statistics & Data Analysis*, 33:425–437, 2000.

[113] Ko Nishino, Shree K. Nayar, and Tony Jebara. Clustered block wise PCA for representing visual data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27, No. 10:1675–1679, Oct. 2005.

[114] E. Oja. Simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 15:267–273, 1982.

[115] E. Oja. *Subspace Methods of Pattern Recognition*. Research Studies Press, 1983.

[116] E. Oja. Neural networks, principal components, and subspaces. *Journal of Neural Systems*, 1:61–68, 1989.

[117] E. Oja. Principal components, minor components, and linear neural networks. *Neural Networks*, 5:927–935, 1992.

[118] E. Oja and J. Karhunen. On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix. *Journal of Mathematical Analysis and Applications*, 106 No. 1:69–84, Feb. 1985.

[119] ORL. ORL face data set. *www.cam-orl.co.uk/facedatabase.html*.

[120] Matthew Partridge and Rafael Calvo. Fast dimensionality reduction and simple PCA. *Intelligent Data Analysis*, 2 No. 1:203–214, 1998.

[121] Z. Pawlak. Rough sets. *International Journal of Information and Computer Sciences*, 11:341–356, 1982.

[122] P. R. Peres-Neto, D. A. Jackson, and K. M. Somers. Giving meaningful interpretation to ordination axes: assessing loading significance in principal component analysis. *Ecology*, 84:2347–2363, 2003.

[123] Pedro R. Peres-Neto, Donald A. Jackson, and Keith M. Somers. How many principal components? stopping rules for determining the number of non-trivial axes revisited. *Computational Statistics & Data Analysis*, 49:974–997, 2005.

[124] Vladimir Pestov. An axiomatic approach to intrinsic dimension of a data set. *Neural Networks*, 21:204–213, 2008.

[125] Ben Pinkowski. Principal component analysis of speech spectrogram images. *Pattern Recognition*, 30, No. 5:777–787, 1997.

[126] PolyU. PolyU palmprint database, second edition. *http://www.comp.polyu.edu.hk/∼biometrics/*.

[127] W. H. Press, S. A. Teukolsk, W. T. Vetterling, and B. P. Flannery. *Numerical recipes in C, Second Edition*. Cambridge University Press, pp. 469-481, 2002.

[128] Qiu B. Prinet, V. Perrier, and E. Monga. Multi-block PCA method for image change detection. In *Proceedings of 12th International Conference on Image Analysis and Processing*, pages 385–390, Sep. 2003.

[129] Arun K. Pujari. *Data mining Techniques*. Universities Press, India, 2002.

[130] A. N. Rajagopalan, K. Srinivasa Rao, and Y. Anoop Kumar. Face recognition using multiple facial features. *Pattern Recognition Letters*, 28:335–341, 2007.

[131] S. J. Raudys and Anil K. Jain. Small sample size effects in statistical pattern recognition: Recommendations for practitioners. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13 No. 3:252–264, 1991.

[132] S. J. Raudys and V. Pikelis. On dimensionality, sample size, classification error and complexity of classification algorithms in pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2:243–251, 1980.

[133] Paolo Ravazzani, Gabriella Tognola, Marta Parazzini, and Ferdinando Grandori. Principal component analysis as a method to facilitate fast detection of transient-evoked otoacoustic emissions. *IEEE Transactions on Biomedical Engineering*, 50, No. 2:249–252, Feb. 2003.

[134] V. Ravi, P. J. Reddy, and H. J. Zimmermanni. Pattern classification with principal component analysis and fuzzy rule bases. *European Journal of Operational Research*, 126:526–533, 2000.

[135] Slobodan Riboric and Ivan Fratric. A biometric identification system based on eigenpalm and eigenfinger features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27 No. 11:1698–1709, 2005.

[136] Syed A. Rizvi, Tarek N. Saadawi, and Nasser M. Nasrabadi. A clutter rejection technique for FLIR imagery using region based principal component analysis. *Pattern Recognition*, 33:1931–1933, 2000.

[137] C. Romero and S. Ventura. Educational data mining: A survey from 1995 to 2005. *Expert Systems with Applications*, 33:135–146, 2007.

[138] S. Roweis. EM algorithms for PCA and SPCA. *Neural Information Processing Systems*, pages 626–632, 1997.

[139] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning internal representations by error propagation in Parallel Distributed Processing: Explorations in the Microstructure of Cognition, D. E. Rumelhart and J. L. McClelland, Eds, pp. 318-362.* MIT Press, Cambridge, MA, 1988.

[140] Ryo Saegusa, Hitoshi Sakano, and Shuji Hashimoto. Nonlinear principal component analysis to preserve the order of principal components. *Neurocomputing*, 61:57–70, 2004.

[141] T. D. Sanger. Optimal unsupervised learning in a single-layer linear feed forward neural network. *Neural Networks*, 2:459–473, 1989.

[142] P. Sanguansat, W. Asdornwised, S. Jitapunkul, and S. Marukatat. Two-dimensional linear discriminant analysis of principal component vectors for face recognition. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 345–348, Toulouse, May 2006.

[143] C. Sarbu and H. F. Pop. Principal component analysis versus fuzzy principal component analysis: A case study: the quality of danube water (1985-1996). *Talanta*, 65, Issue 5:1215–1220, 2005.

[144] Robert Schalkoff. *Pattern Recognition: Statistical, Structural and Neural approaches*. John Wiley & sons (ASIA) Pte Ltd, 2 Clementi Loop 02-01 Singapore 129809, 2005.

[145] Bernhard Scholkopf, Alexander Smola, and Klaus-Robert Mulle. Kernel principal component analysis. *http://sml.nicta.com.au/Publications/homepublications/publications/papers /1999/SchSmoMul99.pdf*, 1999.

[146] Bernhard Scholkopf, Alexander Smola, and Klaus-Robert Muller. Nonlinear

component analysis as a kernel eigenvalue problem. Technical Report 44, Max-Planck-Institute, Germany, December 1996.

[147] Alok Sharma, Kuldip K. Paliwal, and Godfrey C. Onwubolu. Class-dependent PCA, MDC and LDA: A combined classifier for pattern classification. *Pattern Recognition*, 39:1215–1229, 2006.

[148] Heung-Yeung Shum, Katsushi Ikeuchi, and Raj Reddy. Principal component analysis with missing data and its application to polyhedral object modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17, No. 9:854–867, Sep. 1995.

[149] J. A. Sirat. A fast neural algorithm for principal component analysis and singular value decomposition. *Journal of Neural Systems*, 2:147–155, 1991.

[150] Alexander Smola, Olvi L. Mangasarian, and Bernhard Scholkop. Sparse kernel feature analysis. *http://citeseer.ist.psu.edu/232092.html*, 1999.

[151] Wang Song and Xia Shaowei. Robust PCA based on neural networks. In *Proceedings of the 36th Conference on Decision and Control USA*, pages 503–508, Dec. 1997.

[152] Jay Strader and Jean P. Brodie. A principal components analysis of the lick indices of galactic globular clusters. *The Astronomical Journal*, 128:1671–1675, 2004.

[153] Wenyu Sun and Qiuqi Ruan. Two-dimension PCA for facial expression recogni-

tion. In *Proceedings of The 8th International Conference on Signal Processing*, Beijing, 2006.

[154] R. W. Swiniarski and A. Skowron. Rough set methods in feature selection and recognition. *Pattern Recognition Letters*, 24 No. 6:833–849, 2003.

[155] D. Tahmoush and H. Samet. High-dimensional similarity retrieval using dimensional choice. In *Proceedings of First International Workshop on Similarity Search and Applications (SISAP 2008)*, pages 35–42, 2008.

[156] John Tan, Ruixin Yang, and Menas Kafatos. Kernel PCA analysis for remote sensing data. In *Proceedings of 18th Conference on Climate Variability and Change*, 2006.

[157] Keren Tan and Songcan Chen. Adaptively weighted sub-pattern PCA for face recognition. *Neurocomputing*, 64:505–511, 2005.

[158] Junwei Tao, Wei Jiang, Zan Gao, Shuang Chen, and Chao Wang. Palmprint recognition based on 2-dimension PCA. In *Proceedings of First International Conference on Innovative Computing, Information and Control*, pages 326– 330, Aug. 2006.

[159] C. F. J. ter Braak. Canoco–a fortran program for canonical community ordination by [partial] [detrended] [canonical] correspondence analysis, principal component analysis and redundancy analysis (version 2.1), agricultural mathematic group. Technical Report LWA-88-02, Wageningen.

[160] C. F. J. ter Braak. Update notes: Canoco (version 3.1), agricultural mathematic group. AMS Short Course.

[161] C. W. Therrien. *Discrete random signals and statistical signal processing.* Prentice-Hall, Englewood Cliffs, NJ, 1992.

[162] Michael E. Tipping. Sparse kernel principal component analysis. *www.miketipping.com/papers/skpca-nips.ps.gz*, 2001.

[163] Michael E. Tipping and Christopher M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B*, 61, Part 3:611–622, 2005.

[164] Matthew A. Turk and Alex P. Pentland. Face recognition using eigenfaces. In *Proceedings of IEEE conference on Computer Vision and Pattern Recognition*, pages 586–591, 1991.

[165] UCI. UCI repository of machine learning databases. *www.ics.uci.edu/∼mlearn/MLRepository.html.*

[166] UMIST. UMIST face data set. *http://images.ee.umist.ac.uk/danny/database.html.*

[167] W. F. Velicer. Determining the number of components from the matrix of partial correlations. *Psychometrika*, 41:321–327, 1976.

[168] E. Vigneau and E. M. Qannari. Clustering of variables around latent components, communications in statistics. *Simulation and Computation*, 32:1131 – 1150, 2003.

[169] S. K. Vines. Simple principal components. *Applied Statistics*, 49:441 – 451, 2000.

[170] Liwei Wang, Xiao Wang, Xuerong Zhang, and Jufu Feng. The equivalence of two-dimensional PCA to line-based PCA. *Pattern Recognition Letters*, 26:57–60, 2005.

[171] Xiuqing Wang, Zengguang Hou, Yongqian Zhang, and Min Tan. Scene analysis for mobile robot based on multi-sonar-ranger data. In *Proceedings IEEE International Conference on Information*, pages 365–369, Weihai, Shandong, China, Aug. 2006.

[172] S. Watanabe, P. F. Lambert, C. A. Kulikowski, J. L.. Buxton, and R. Walker. Evaluation and selection of variables in pattern recognition. *Computer and Information Sciences II New York: Academic Press*, 1967.

[173] S. Watnabe. *Pattern Recognition: Human and Mechanical*. Wiley, NY, 1985.

[174] Andreas Weingessel and Kurt Hornik. Local PCA algorithms. *IEEE Transactions on Neural Networks*, 11:1242–1250, 2000.

[175] Ying Wen and Pengfei Shi. Image PCA: A new approach for face recognition. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pages I–1241–I–1244, Honolulu, HI, Apr. 2007.

[176] Juyang Weng, Yilu Zhang, and Wey-Shiuan Hwang. Candid covariance-free incremental principal component analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25, No. 8:1034–1040, 2003.

[177] R. H. Whiter. Competitive hebbian learning: Algorithm and demonstrations. *Neural Networks*, 5:261–275, 1992.

[178] Dekai Wu, Weifeng Su, and Marine Carpuat. A kernel PCA method for superior word sense disambiguation. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics Barcelona, Spain*, 2004.

[179] Bi Xian, Tonghua Li, Gexiao Sun, and Tongcheng Cao. The combination of principal component analysis, genetic algorithm and tabu search in 3D molecular similarity. *Journal of Molecular Structure (Theochem)*, 674:87–97, 2004.

[180] Xiaoping Xie, Zhitong Cao, Xuchu Weng, and Dan Jin. Estimating intrinsic dimensionality of FMRI dataset incorporating an AR(1) noise model with cubic spline interpolation. *Neurocomputing*, doi:10.1016/j.neucom.2008.04.003, 2008.

[181] Anbang Xu, Xin Jin, Yugang Jiang, and Ping Guo. Complete two-dimensional PCA for face recognition. In *Proceedings of 18th International Conference on Pattern Recognition*, pages 481–484, Hong Kong, 2006.

[182] Rui Xu and Donald Wunsch II. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16 No. 3:645–678, 2005.

[183] Quan xue Gao. Is two-dimensional PCA equivalent to a special case of modular PCA? *Pattern Recognition Letters*, 28:1250–1251, 2007.

[184] Yale. Yale face data set. *http://cvc.yale.edu/projects/yalefaces/yalefaces.html*.

[185] Jian Yang, Alejandro F. Frangi, Jing yu Yang, David Zhan, and Zhong Jini. KPCA Plus LDA: A complete kernel fisher discriminant framework for fea-

ture extraction and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27, No. 2:230–244, 2005.

[186] Jian Yang, Zhong Jin, Jing yu Yang, David Zhang, and Alejandro F. Frangi. Essence of kernel fisher discriminant: KPCA plus LDA. *Pattern Recognition*, 37:2097–2100, 2004.

[187] Jian Yang and Jing-Yu Yang. From image vector to matrix: a straight forward image projection technique—IMPCA vs. PCA. *Pattern Recognition*, 35:1997–1999, 2002.

[188] Jian Yang, D. Zhang, and Jing yu Yang. Is ICA significantly better than PCA for face recognition? In *Proceedings of Tenth IEEE International Conference on Computer Vision*, pages 198 – 203, Oct 2005.

[189] Jian Yang, David Zhang, Alejandro F. Frangi, and Jing-Yu Yang. Two-dimensional PCA: A new approach to appearance-based face representation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1):131–137, 2004.

[190] Ming-Hsuan Yang, Narendra Ahuja, and David Kriegman. Face recognition using kernel eigenfaces. In *Proceedings of International Conference on Image Processing*, pages 37–40, 2000.

[191] H. Henry Yue and Masayuki Tomoyasu. Weighted principal component analysis and its applications to improve FDC performance. In *Proceedings of 43rd IEEE Conference on Decision and Control*, pages 4262 – 4267, Dec. 2004.

[192] An Zeng, Dan Pan, Qi-Lun Zheng, and Hong Peng. Knowledge acquisition based on rough set theory and principal component analysis. *IEEE Intelligent Systems*, 21, Issue 2:78–85, Mar. 2006.

[193] Bailing Zhang, Minyue Fu, and Hong Yan. A nonlinear neural network model of mixture of local principal component analysis: application to handwritten digits recognition. *Pattern Recognition*, 34:203–214, 2001.

[194] Daoqiang Zhang and Zhi-Hua Zhou. $(2D)^2$PCA: Two-directional two-dimensional PCA for efficient face representation and recognition. *Neurocomputing*, 69 Issues 1-3:224–231, Dec. 2005.

[195] Daoqiang Zhang, Zhi-Hua Zhou, and Songcan Chen. Diagonal principal component analysis for face recognition. *Pattern Recognition*, 39:140–142, 2006.

[196] Peng Zhang, Jing Peng, and Carlotta Domeniconi. Kernel pooled local subspaces for classification. *IEEE Transactions on Systems, Man and Cybernetics-Part B: Cybernetics*, 35 No. 3:489–502, 2005.

[197] Xiaoyu Zhang, Jiexin Pu, and Xinhan Huang. Face detection based on two dimensional principal component analysis and support vector machine. In *Proceedings of IEEE International Conference on Mechatronics and Automation*, pages 1488–1492, Luoyang, Henan, Jun. 2006.

[198] Qijun Zhao and Hongtao Lu. PCA-based web page watermarking. *Pattern Recognition*, 40:1334–1341, 2007.

[199] W. Zhao, R. Chellapa, and A. Krishnaswamy. Discriminant analysis of principal

components for face recognition. In *Proceedings of Third IEEE International Conference on Automatic Face and Gesture Recognition*, pages 336–341, 1998.

[200] Viktor Zubko, Yoram J. Kaufman, Richard I. Burg, and J. Vanderlei Martins. Principal component analysis of remote sensing of aerosols over oceans. *IEEE Transactions on Geo-science and Remote Sensing*, 45, No. 3:730–745, Mar. 2007.

[201] Wangmeng Zuo, David Zhang, and Kuanquan Wang. An assembled matrix distance metric for 2DPCA-based image recognition. *Pattern Recognition Letters*, 27:210–216, 2006.

[202] Wangmeng Zuo, David Zhang, and Kuanquan Wang. Bidirectional PCA with assembled matrix distance metric for image recognition. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, 36, No. 4:863 – 872, 2006.

[203] Wangmeng Zuo, David Zhang, Jian Yang, and Kuanquan Wang. BDPCA plus LDA: A novel fast feature extraction technique for face recognition. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, 36 No. 4:946–953, 2006.