

A NOVEL CONIC SECTION CLASSIFIER WITH TRACTABLE GEOMETRIC
LEARNING ALGORITHMS

By

SANTHOSH KODIPAKA

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2009

© 2009 Santhosh Kodipaka

To my mother, brother and sister

ACKNOWLEDGMENTS

For the unflinching support and meticulous guidance I received from my advisers Prof. Baba C. Vemuri, and Dr. Arunava Banerjee throughout my long graduate study, I am extremely grateful. I owe it to Dr. Anand Rangarajan for the broad exposure I gained through his course offerings on vision, learning and shape analysis. I would like to thank Dr. Jeffrey Ho for being very forthcoming in sharing his insights when I approached him for new ideas. I also thank Dr. Stephen Blackband not only for agreeing to be on my committee, but also for his valuable inputs before and after my dissertation proposal, and during the Epilepsy diagnosis project.

I express my sincere gratitude to the Department of Computer & Information Sciences & Engineering, University of Florida and the NIH sponsors for generously supporting my graduate studies. This research was in part funded by NIH grant RO1 NS046812 to Prof. Vemuri. I also acknowledge the travel grant from the Graduate Student Council at UF.

At the Center for Vision, Graphics and Medical Imaging (CVGMI), I had the privilege of working in a high spirited research lab environment. I thank former and current CVGMI lab members: Eric Spellman, Hongyu Guo, Fei Wang, Nicholas Lord, Bing Jian, Adrian Peter, Angelos Barmpoutis, O'Neil Smith, Ajit Rajwade, Ozlem Subakan, Ritwik Kumar, and Guang Cheng, for being very approachable in general, and enthusiastic when it comes to discussing stuff at length. For their camaraderie, empathy and support, I thank my friends Venu, Sudhir, Suri, Slash, Bala, Ashish, Shiva, Ashwin, Teja, Tapasvi, Jayendra, Muthu, Vishak, Arun and Kalyan. Special thanks are extended to Ravi Jampani for his invaluable friendship and of course, for proof reading my drafts.

For the exposure to computer vision and graphics at IIIT-Hyderabad, India I gained under the guidance of Dr. P.J. Narayanan and Dr. C.V. Jawahar, I am very thankful to them. It was the vibrant academic environment in the newly setup IIIT, the exciting courses offered by my faculty advisers and the honors projects, that inspired me to pursue Ph.D.

Most importantly, this graduate study would not have been possible without the unconditional support and encouragement I received from my family. In particular, I am grateful to my younger brother Kishore, who took upon himself all the familial responsibilities in India and left me little to worry about. Same goes to my cousins Laxman and Sagar.

TABLE OF CONTENTS

	<u>page</u>
ACKNOWLEDGMENTS	4
LIST OF TABLES	9
LIST OF FIGURES	10
ABSTRACT	11
CHAPTER	
1 INTRODUCTION	12
1.1 Supervised Learning	13
1.1.1 Features for Classification	14
1.1.2 Concept Learning	16
1.1.3 Generalization	17
1.1.4 Model Selection	18
1.1.5 Application	22
1.2 Concept Classes	23
1.2.1 Bayesian learning	23
1.2.2 Fisher Discriminant	24
1.2.3 Support Vector Machines	24
1.2.4 Kernel Methods	26
1.3 Outline of the Dissertation	27
1.4 Summary	29
2 CONIC SECTION CLASSIFIER	30
2.1 Motivation	30
2.2 Synopsis	31
2.3 The Conic Section Concept Class	32
2.4 Learning Overview	37
2.5 Related Work	39
2.6 Conclusions	41
3 THE GEOMETRY OF CONSTRAINTS	42
3.1 Geometric Primitives	42
3.2 Intersections of <i>Hyperspheres</i>	45
3.3 Intersections of <i>Shells</i>	46

4	LEARNING WITH EQUALITY CONSTRAINTS	49
4.1	Learning Algorithm	49
4.1.1	Finding Class-Eccentricities $\langle e_1, e_2 \rangle$	50
4.1.2	Learning Misclassified Points	51
4.1.3	Updating The Focus	52
4.1.4	Updating The Directrix	55
4.1.5	Initialization	56
4.1.6	Discussion	57
4.2	Results	57
4.2.1	Classifiers	58
4.2.2	Synthetic Data	58
4.2.3	Epilepsy Data	58
4.2.4	Colon Tumor	59
4.2.5	Sheffield FaceDB	60
4.2.6	CURET Textures	60
4.3	Summary and Conclusions	61
5	LEARNING WITH LARGE MARGIN PURSUIT	62
5.1	Learning Algorithm Overview	62
5.2	The Margin Computation	63
5.2.1	Overview	64
5.2.2	Spanning the Discriminant Boundary \mathcal{G}	65
5.2.3	Nearest Point on \mathcal{G} with Fixed Directrix Distances	65
5.2.4	Nearest Point on \mathcal{G} with Fixed Focal Distances	67
5.2.5	Large Margin Pursuit	69
5.3	Experiments	71
5.3.1	Evaluating Margin Computation	71
5.3.2	Classification Results	72
5.4	Summary	74
6	LEARNING WITH INEQUALITY CONSTRAINTS	75
6.1	The Learning Algorithm	75
6.1.1	Learning in Eccentricity Space	77
6.1.2	Constructing <i>Null Spaces</i>	80
6.1.3	Learning Misclassified Points	87
6.1.4	Initialization	89
6.1.5	Discussion	90
6.2	Experiments	90
6.2.1	Datasets	91
6.2.2	Classifiers and Methods	92
6.2.3	Conic Section Classifier	93
6.2.4	Classification Results	93
6.3	Summary	95

7	CONCLUSION	97
	APPENDIX: NOTATION	99
	REFERENCES	100
	BIOGRAPHICAL SKETCH	106

LIST OF TABLES

<u>Table</u>	<u>page</u>
3-1 Geometric Objects	43
4-1 Classification comparisons of CSC	61
5-1 Accuracies of large margin pursuits	71
5-2 Details of data used in experiments	72
5-3 Classification comparisons of CSC with large margin pursuit	73
6-1 Details of data used in experiments	91
6-2 Classification comparisons of CSC with inequality constraints	94
6-3 Parameters for results reported in Table 6-2	95

LIST OF FIGURES

<u>Figure</u>	<u>page</u>
1-1 Challenges involved in supervised classification	13
1-2 Connection between learning, generalization, and model complexity	18
2-1 Conic Sections in 2D	33
2-2 Overview of the Conic Section concept class	34
2-3 Discriminants due to different configurations of class conic sections	35
2-4 Discriminant boundaries in \mathbb{R}^2 for $K = 2$	36
2-5 Discriminant boundaries in \mathbb{R}^3 for $K = 2$	36
3-1 Geometric Objects in 1D, and 2D	43
3-2 Fiber Bundle	44
3-3 Intersection of two spheres	45
3-4 Intersection of two <i>shells</i>	47
4-1 Algorithm to Learn the Descriptors due to Equality Constraints	50
4-2 Classification in <i>ecc-Space</i>	51
4-3 Intersection of two <i>hyperspheres</i>	53
4-4 Deformation features in the Epilepsy Data	59
4-5 Texture Data	60
5-1 Linear subspace \mathcal{H} in which directrix distances are constant	66
5-2 The discriminant boundary in \mathcal{H}	68
5-3 Margin Computation Algorithm	70
6-1 Algorithm to Learn the Descriptors due to Inequality Constraints	77
6-2 Classification in <i>ecc-Space</i>	78
6-3 Cross section of two intersecting <i>hyperspheres</i>	82
6-4 Cross-section of two intersecting <i>shells</i>	85
6-5 Cross-section of the donut-like toroidal <i>Null Space</i>	87

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

A NOVEL CONIC SECTION CLASSIFIER WITH TRACTABLE GEOMETRIC
LEARNING ALGORITHMS

By

Santhosh Kodipaka

August 2009

Chair: Baba C. Vemuri
Cochair: Arunava Banerjee
Major: Computer Engineering

Several pattern recognition problems in computer vision and medical diagnosis can be posed in the general framework of supervised learning. However, the high-dimensionality of the samples in these domains makes the direct application of off-the-shelf learning techniques problematic. Moreover, in certain cases the cost of collecting large number of samples can be prohibitive.

In this dissertation, we present a novel concept class that is particularly designed to suit high-dimensional sparse datasets. Each member class in the dataset is assigned a prototype conic section in the feature space, that is parameterized by a focus (point), a directrix (hyperplane) and an eccentricity value. The focus and directrix from each class attribute an eccentricity to any given data point. The data points are assigned to the class to which they are closest in eccentricity value. In a two-class classification problem, the resultant boundary turns out to be a pair of degree 8 polynomial described by merely four times the parameters of a linear discriminant.

The learning algorithm involves arriving at appropriate class conic section descriptors. We describe three geometric learning algorithms that are tractable and preferably pursue simpler discriminants so as to improve their performance on unseen test data. We demonstrate the efficacy of the learning techniques by comparing their classification performance to several state-of-the-art classifiers on multiple public domain datasets.

CHAPTER 1 INTRODUCTION

Seemingly mundane activities that we take for granted like identifying a voice over phone or a colleague's hand-writing, spotting traffic signs while driving and acting accordingly, and recognizing high school batch-mates from an old photo, can be attributed to our mature *pattern recognition* capabilities. These abilities involve *perception* of the surrounding environment and *learning* from a series of stimuli-responses that are almost automatically assimilated into different *categories*. Upon encountering new scenarios, *relevant instances* are instantaneously *recalled* for further interpretation. Machine Learning as a sub-discipline in computer science is devoted to developing algorithms and techniques to mimic this learning behavior in automated systems. Ambitious as it may sound, the resulting applications in automating spoken word recognition (IVR), text categorization, printed or hand-written character recognition, medical diagnosis, data mining, filtering email spam, fraudulent activity detection, etc., are hugely improving our way of life.

There are three principle learning paradigms based on the kind of feedback given to a learning system. In *supervised learning*, the system is presented with a set of input-output examples and is expected to guess desired outputs for new input patterns. *Unsupervised learning* or *clustering* is at the other end of spectrum, in which a system is exposed to a collection of input patterns with out any output labels. Here the task is to automatically group similar patterns and infer relations between them. In the third mode of learning, called *reinforcement learning*, no desired output is supplied, but the system gets a feedback of whether its inference or decision on an input is good or bad.

Our focus is on *supervised classification learning*, in which given a set of labeled samples belonging to two categories, a classifier is trained to discriminate between the classes so that unseen samples, whose category information is not known, can be assigned to their respective classes. In this dissertation, we introduce a new classifier based on conic sections, referred to as the Conic Section Classifier (CSC).

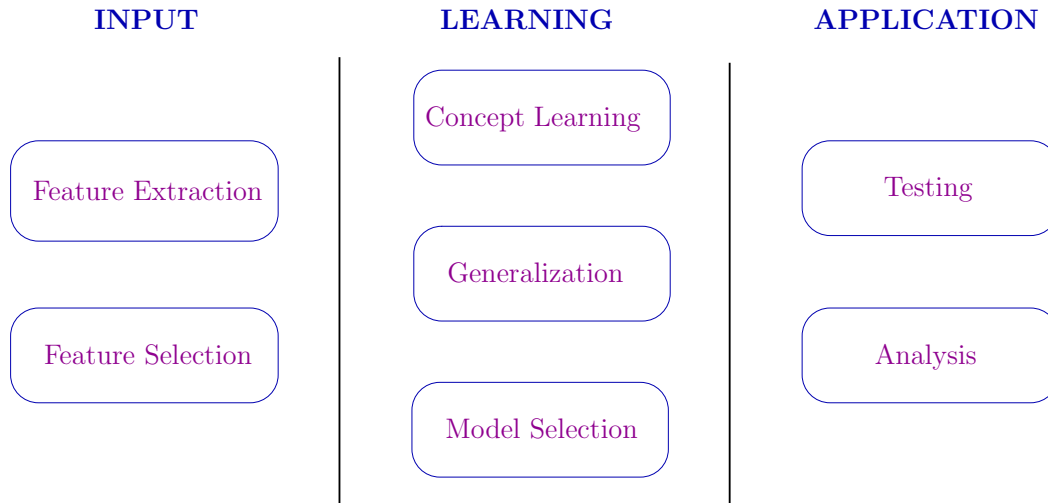


Figure 1-1. Challenges involved in supervised classification

In this chapter, we briefly discuss the various stages and challenges involved in the design and application of a supervised learning technique. We also outline important classification techniques that are used to compare with the CSC.

1.1 Supervised Learning

Many notable problems in computer vision and medical diagnosis can be posed in the general framework of supervised learning theory. Consider, for instance, the diagnosis of disease using DNA micro-array gene expression data [1]. In a typical scenario, several thousands of genes are simultaneously monitored for expression levels either in different tissue types drawn from the same environment or of the same tissue type drawn from different environments. A subset of these genes are then identified based on the relation of their expression levels to the pathology. The learning problem now involves identifying the pathology type of a sample, given its gene-expression levels as the input features and a set of training samples with their known outcomes. More innocuous instances of problems that can be posed in this framework include the automated face recognition [2], diagnosis of various kinds of pathologies, for instance, types of Epilepsy [3] from MRI scans of the brain, and various forms of Cancer [4] from micro-array gene expression data.

The learning application comprises of the following stages: (1) collection of features that describe samples, (2) mapping the features into a structured metric space (like the Euclidean space), (3) identifying which family of functions are better suited for the application domain of interest, (4) learning the best classifier function within a (parametric) family of functions, given the labeled training samples, (5) improving upon the generalizability of the classifier in the learning stage using principles such as the *Occam's razor*, so as to perform better on unseen data, and (6) analyze the classifier's performance to further fine tune it for specific application domains. The various stages involved in a supervised learning application are illustrated in Figure 1-1. In this dissertation, we introduce a novel classifier and hence focus on the learning aspects listed in the figure.

1.1.1 Features for Classification

To begin with, it is very crucial to identify the characteristic features of data for a given application, that have most discriminatory potential. This is not obvious to realize when such a characterization is not explicit. Consider two classes of gray-level images that need to be classified, and have no discernible structure in the images. In such a case, they will be re-sampled to a fixed size and the intensities at corresponding pixels are stacked together in a vector of features. This could increase the difficulty of classification later. In many cases, it is not even meaningful to do so [5]. However, if the distribution of intensities is observed to be different between the classes, one can use histograms or fit parametric distributions to the data. In fact, such features are invariant to image rotation and scale. If we further learn that each image has only one object of interest, it can then be segmented (into foreground and background) and features can be obtained from the object boundary. In this manner, the more discriminable features we extract, the easier it becomes to classify them. A survey of feature extraction methods is presented in [6].

Generally speaking, to extract the best features, we need to characterize the manifold on which the given data lies and employ related metrics and operations. In most cases,

the features are mapped into Euclidean spaces. However, one cannot assign numbers to certain feature types as they are not numerically ordered. Such *nominal* features are encountered in descriptions of the weather conditions, patient profiles, etc. There are a few classification techniques that can handle such data, like the Classification and Regression (Decision) Tree (CART) [7], rule based learning [8] and string-matching methods with edit-distances and kernels defined appropriately for text data.

1.1.1.1 Feature Selection

When it is not feasible to determine the compound features that correlate to the class labels, or when the number of elementary features collected is too large, it is necessary to discard irrelevant features. Doing so, benefits the subsequent learning stages drastically, by mitigating the *curse of dimensionality* and speeding up the overall analysis. This step is crucial for data like the gene-expression arrays, to identify a particular subset of features that are most likely to be correlated to the outcomes. The conventional approaches to accomplish this are (i) pruning based on feature ranking, (ii) variable subset selection, and (iii) dimensionality reduction [9].

Candidate features are typically ranked based on their correlation to the target labels. The measures like Pearson correlation coefficient [10], and mutual information [11], [12] are utilized. A set of features with leading ranks are chosen for classification, while the rest are deemed irrelevant. In bio-medical applications, *p-values* (significance probabilities) are used to perform feature selection despite strong objections [13], [14]. It could be very misleading to use p-values in this context, as they cannot be considered as evidence against the *null* hypothesis [15]. Especially, the notorious $p < .05$ significance level is not a gold standard, but is data dependent.

Certain subsets of features can have more predictive power as a group, as opposed to discarding them based on their individual ranks. The subsets can be scored for their discriminating ability using any given learning machine [16]. Feature subset selection can also be combined into the learning process. For instance, the weight vector in the linear

SVM is used to determine which sets of features do not effect large margins. The other set of approaches reduce dimensionality using linear (PCA) or non-linear projections (kernel PCA), like the pursuit of discriminative subspaces [17], large margin component analysis [18], etc. The criteria for projection are data fidelity and class discriminability.

The Ugly Duckling theorem states that *there are no problem-domain independent best features* [19]. Thus feature selection has to be domain-specific in order to improve performance of the learning algorithm. For recent literature on feature selection, we refer the reader to a special issue on feature selection [20].

1.1.2 Concept Learning

The two-class learning problem may be posed formally as follows: One is given a dataset of N labeled tuples, $\mathbf{Z} = \{\langle X_i, y_i \rangle\}$, $i \in \{1 \dots N\}$, where the X_i 's are the data points represented in some feature space \mathcal{X} , and the y_i 's are their associated class labels. Let \mathcal{Y} be a two element target space $\{-1, +1\}$, and $y_i \in \mathcal{Y}$. Then $\mathbf{Z} \in \mathcal{Z}^N$, where $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$. The feature space \mathcal{X} in which the data points, X_i 's, lie can be any appropriately defined space with some degree of structure. In the majority of cases \mathcal{X} is set to be the Euclidean space \mathbb{R}^M . The goal of learning is to identify a function (also called a *concept*) $f : \mathcal{X} \rightarrow \mathcal{Y}$ that minimizes some notion of classification error. In this section, we use a notation similar to that in [21].

Formulated as above, the learning problem is still ill-posed since there exist uncountably many functions, f 's, that yield zero error. To make this a well posed problem, one further restricts the f 's to a particular family of functions, say \mathcal{F} , known as the *concept class*. The best member of that class, $f^* \in \mathcal{F}$, is identified as the one that results in minimal expected classification error, defined as:

$$\begin{aligned}
 R[f] &\stackrel{\text{def}}{=} \mathbb{E}_{X,y} [\xi(y, f(X))] \\
 f^* &= \underset{f \in \mathcal{F}}{\operatorname{argmin}} R[f]
 \end{aligned}
 \tag{1-1}$$

where ξ is the 0 – 1 loss (indicator) function $\mathbb{I}(y_i = f(X_i))$, and $R[f]$ is the *expected error*. Given N labeled samples, the *empirical* or *training error* is defined as:

$$R_e [f, \mathbf{Z}] \stackrel{\text{def}}{=} \frac{1}{N} \sum_{i=1}^N \xi(y_i, f(X_i)) \quad (1-2)$$

By minimizing the *empirical error* $R_e [f, \mathbf{Z}]$, we can obtain a member classifier, say f_e^* , that could yield zero error while *overfitting* the data. Overfitting implies that the classifier could have learnt noise and might not perform well on unseen data from \mathcal{Z} . Next, we describe some criteria to avoid overfitting.

1.1.3 Generalization

The *generalization error*, given an algorithm $\mathcal{A} : \mathcal{Z}^N \rightarrow \mathcal{F}$, is defined as :

$$R_g [\mathcal{A}, \mathbf{Z}] \stackrel{\text{def}}{=} R_g[\mathcal{A}(\mathbf{Z})] - \inf_{f \in \mathcal{F}} R[f] \quad (1-3)$$

i.e. the deviation in error from that due to the best member classifier f^* . Numerically, stable solutions are obtained by adding a regularization to the involved optimization in the algorithm \mathcal{A} . In accordance with the *Occam's razor* principle, highly non-linear boundaries that have higher capacity to learn need to be penalized. The objective function for \mathcal{A} can now be written as:

$$\mathcal{A}(\mathbf{Z}) \stackrel{\text{def}}{=} \underset{f \in \mathcal{F}}{\text{argmin}} R_e[f, \mathbf{Z}] + \lambda \Omega[f] \quad (1-4)$$

where $\Omega : \mathcal{F} \rightarrow \mathbb{R}^+$, is the regularizer functional that should attempt to quantify the complexity of the classifier. The connection between concept learning, generalization, and complexity of the discriminant, is illustrated in Figure 1-2. Generalizability of the classifier suffers when the learning classifier is either too simple to partition the input space into two class regions, or when it becomes too complex and learns noise as well.

Now, we consider some candidate criteria that could minimize the generalization error. Fisher's criterion, for a two-class cases, is defined as the ratio of distance between the class-means over the sum of variances within each class. The goal is to find a direction, on which the projected data points have the maximal separation between

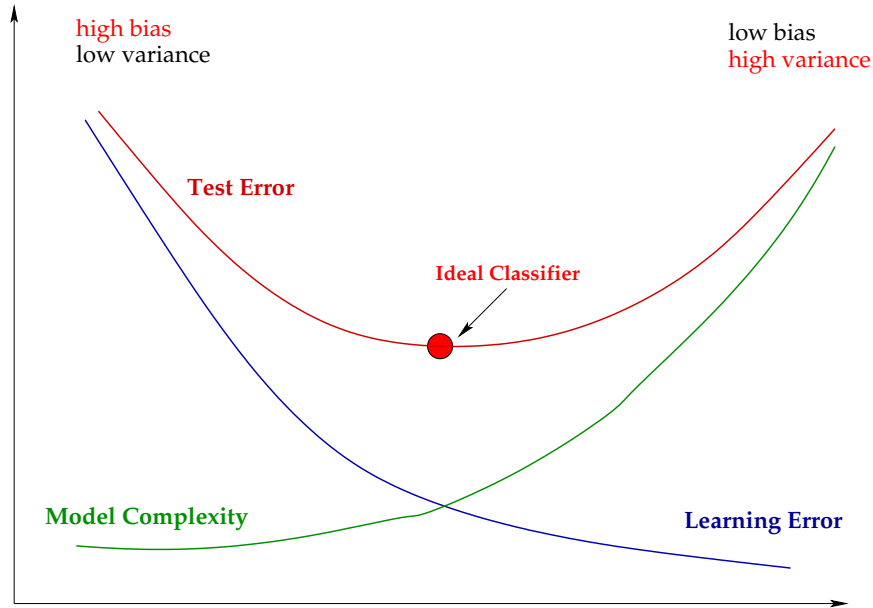


Figure 1-2. Connection between learning, generalization, and model complexity, for member classifiers of any given concept class in general. The x -axis represents members of the *concept class* ordered by the complexity of their discriminants. Generalization (error on unseen data) suffers when the learning model is either too simple to capture the discriminability or when it becomes too complex, thereby learning noise as well. Illustration is adapted from Hastie, Tibshirani, & Friedman [22]

classes and minimal variance within each class. A classifier is considered stable if its label assignment remains unchanged w.r.t. small perturbations in its parameters and/or in the data points. This intuition is one amongst many to justify pursuit of large margins, both functional and geometric so as to find the classifier with the best possible generalization capacity within a concept class. The notion of *shattering* the training set can be used to bound $R_g[f]$, as explained in the Section 1.1.4.1.

1.1.4 Model Selection

The No Free Lunch Theorem, essentially states that *without prior knowledge regarding the nature of the classification task, it is not possible to predict which of a given set of concept classes will yield the smallest generalization error* [23]. Further, *no classifier is superior to (or inferior to) an uniformly random classifier*, since each is gauged by its performance on unseen data. This is true when the performance is uniformly averaged

over all possible labeled samples, excluding the training set \mathbf{Z} . However, in real world applications, not all possible input-out pairs in $\mathcal{X} \times \mathcal{Y}$ are important. So the theorem allows that with particular non-uniform priors, one learning algorithm could be better than the other for a set of problems. Practitioners therefore resort to applying as many classifiers with different concept classes as possible so as to choose the best one.

Model selection deals with the choice between competing concept classes. Minimum Description Length does not fare well for classification problems. A sinusoidal function in \mathbb{R} , with only one parameter, is shown to be capable of learning infinitely many points [24]. In Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), and Stochastic Complexity (SC), the term penalizing complexity is a function of the number of training samples and the number of concept class parameters. In the context of classification, these are not practical either. In this section, we review model selection using VC dimension, and *cross-validation* techniques.

1.1.4.1 VC Dimension

The learning capacity of concept class can be characterized by its *shattering* property. A concept class \mathcal{F} is said to have *shattered* a given set of N samples, if it always has a member that can correctly classify all the samples correctly, for all possible, 2^N , labelings. The Vapnik Chervonenkis (VC) dimension of \mathcal{F} is the largest set of d points, such that there exists at least one configuration of the points that can be *shattered* by \mathcal{F} . *Note that such a shattering is not guaranteed for any configuration in general* [25].

In this dissertation, when we deal with high dimensional sparse data, with $N \ll M$, a hyperplane ($\mathbf{W}^T X + b = 0$) can have a VC dimension of $(M + 1)$. It does not mean that every every configuration of N samples can be *shattered*. A trivial configuration to consider is the XOR configuration of 4 points in \mathbb{R}^2 that cannot be separated by a linear discriminant. The same set of points, when mapped into \mathbb{R}^M under an affine transform, continue to remain non-separable.

In particular, we list two statements of interest in order to clarify upon the *shattering* capacity of a hyperplane, the proofs for which are in [25].

Lemma 1.1.1. *Two sets of points in \mathbb{R}^M may be linearly separable if and only if their convex hulls do not intersect.*

Theorem 1.1.2. *Any configuration of N points in \mathbb{R}^M can be shattered by a hyperplane if and only if those points do not lie in an $(N - 2)$ dimensional linear subspace.*

From these two statements, it can be derived that the VC dimension of a hyperplane in \mathbb{R}^M is $(M + 1)$. Since we can find a configuration of $(M + 1)$ points, such that when one of the points is made the origin, the position vectors of the other M points can be linearly independent. However, such a configuration does not exist for $(M + 2)$ points in \mathbb{R}^M .

VC dimension, d , has been shown to bound the generalization error $R[f]$, which cannot be computed in general. With probability $(1 - \eta)$, the following holds [24]:

$$R_g[f] \leq R_e[f] + \sqrt{\left(\frac{h(\log(2N/d) + 1) - \log(\eta/4)}{N}\right)} \quad (1-5)$$

where $\eta \in [0, 1]$. However this is a conservative bound. Using the structural risk minimization principle, one should prefer to search for best classifiers starting from simpler classifiers like SVMs with lower degree polynomial kernels, etc. When $N \ll M$, the VC dimension of even a hyperplane is large enough to suffer from poor generalization capacity. A bound that is much more practical is the one that relates bounded margins and VC dimension.

Theorem 1.1.3. *Let all $\{X_1, X_2, \dots, X_N\}$ be points contained within a sphere of radius r . The VC dimension of linear discriminants with a margin ρ is:*

$$d \leq \min\left(\left\lceil \frac{4r^2}{\rho^2} \right\rceil, n\right) + 1 \quad (1-6)$$

The intuition is outlined in [26], [24], wherein the margin ρ can be maximized by placing the points on a regular simplex whose vertices lie on the sphere. The actual proof is presented in [27]. This makes the VC-dimension of the large margin hyperplane almost

independent of the input space dimension, as pursued in the linear support vector machine (SVM). Hence, a large margin pursuit can also be used to determine the best member within a concept class.

1.1.4.2 Cross-Validation

In a cross-validation (CV) protocol, the dataset is randomly partitioned into a training set and a test set. The classifier is trained on the training set and evaluated on the test set. This process is repeated several times and the resultant error averaged over such partitions, is then reported as the estimate of generalization error. In a k -fold CV, the data is partitioned into k sets, called *folds*. Each set is tested once using a classifier that is trained on the other sets. When the number of partitions is equal to the number of samples, it is called the *leave-one-out* cross validation. Typically it yields classification errors with low bias and high variance, since the classifiers in each fold-experiment could have learnt noise also. It may be better suited for continuous error functions, rather than the 0 – 1 classification error functions. As the number of folds decrease the variance drops, but the bias might increase. In a *stratified* cross validation, the proportion of labels within each partition is similar to that of the entire dataset.

In *bootstrapping*, random subsamples with substitution are generated from a given dataset instead of partitioning it. In order to mimic the real world case of the data, the sampling is weighted to generate samples that haven't been chosen earlier. These error estimation protocols are evaluated in [28]. A *stratified 10-fold CV* is recommended, especially for high-dimensional sparse datasets, as it ensures every sample is tested once, and could result in reasonable compromise between error bias and variance. The best parameters for a concept class are chosen as those that yield the least estimated generalization error.

1.1.4.3 Discussion

Although Statistical Learning Theory [29] does provide formal bounds for the *generalization error* of a classifier (i.e., the expected error on as yet unseen data) as a

function of the classifier's empirical error and a formalization of the complexity of the classifier's concept class, such bounds are often weak. In practice therefore, one performs internal cross-validation within the training set, to find the best member within a concept class. Both the cross-validation and boot-strapping are used for model-selection in this manner. Among competing concept classes, the model with the best estimated generalization error can be used for purposes of future classification.

Note that the *No Free Lunch* theorem applies to cross-validation as well, as its inefficiency is demonstrated on a simple problem in [30]. However, learning procedures that use cross-validation tend to outperform random guessing [31], as is the case in several successful applications. The counter examples often tend to be degenerate cases, which do not represent the samples generally seen in learning applications.

1.1.5 Application

In applications like USPS character recognition, spam filtering, finger-print matching, face-detection, the best candidate classifier or a combination of such are employed to classify the test data that the classifier has generally been not exposed to. As new labeled data is added, it is desirable to prefer learning algorithms that can learn online. Online learning involves updating the classifier upon being exposed to the new data alone, without further access to the previous used training data from which it has been learnt.

The confusion matrix (2×2), that contains the true/false positives and negatives, can be used to better interpret the classifier's performance. In many applications the penalty on false positive and false negative classification is not necessarily the same. The requirements of particular applications on preferred sensitivity (false positive or Type-I error, [10]) and specificity (false negative or Type-II error) rates can be accommodated by either selecting appropriate thresholds for classification, or by modifying the learning algorithm accordingly. A classifier function usually involves a threshold that separates two classes on a continuous valued output of a discriminant function. For instance in medical diagnosis and security applications, lower false negative (patient cleared as healthy) rates

are sought while in spam filtering, false positive (genuine mail classified as spam) rate needs to be reduced.

The Receiver operating statistic (ROC) [32] curve is a plot of the true positive rate versus the false positive rate, obtained by varying the threshold for a classifier. It could be used to achieved desired preference between the two types of error noted above.

1.2 Concept Classes

In this section, we review the concept classes that we compared against the classifier introduced in this dissertation. They differ from each other in their learning approaches as well as the criteria that address the generalization capacity.

1.2.1 Bayesian learning

Bayesian learning [8] is a statistical approach in which we compute the probability of an unseen pattern belonging to a particular class (*posterior*), having observed some labeled data. Let $P(\beta_1), P(\beta_2)$ be *a priori* probability mass functions of two classes to which a continuous random sample X belongs to. Given the class-conditional probability densities or *likelihoods* $p(X|\beta_i)$, *a posteriori* probability of X is given by the Bayes formula

$$P(\beta_i|X) = \frac{p(X|\beta_i)P(\beta_i)}{p(X)} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}} \quad (1-7)$$

$$\text{where } p(X) = \sum_{i=1,2} p(X|\beta_i)P(\beta_i)$$

A new sample is assigned the label of class that has the maximum *a posteriori* probability. This classifier is referred to as the *naive Bayes classifier*. The model parameters can be estimated using Maximum Likelihood Estimation (MLE) given a set of labeled samples. In many computer vision applications explicit priors, say on particular shapes of interest, can be used to construct generative models using hierarchical probability models [33] if necessary. The modes of variation within the observed training sample assist in defining appropriate statistical models [34]. In such scenarios, a naive Bayes classifier is reliable enough. The likelihoods are usually modeled as Normal distributions in simpler cases, parameterized by the class sample mean and covariance matrix. The resultant

discriminants are linear if variance is uniform. The discriminants are hyperquadrics for general covariance matrices.

1.2.2 Fisher Discriminant

The Fisher approach [35] is based on projecting data in \mathbb{R}^N onto a line, on which class separation is maximal and variance within each class is minimal. Given two sets of features $\{\chi_1, \chi_2\}$, the vector \underline{w} , along which separation is optimal, can be found by maximizing the *Fisher criterion*:

$$J(\underline{w}) = \frac{\underline{w}^T S_B \underline{w}}{\underline{w}^T S_W \underline{w}} \quad (1-8)$$

With the class means $m_i = \frac{1}{l_i} \sum_{j=1}^{l_i} x_j^i$, the *between-class* and *within-class* covariance matrices S_B and S_W can be defined as follows

$$S_B = (m_1 - m_2)(m_1 - m_2)^T, \quad S_W = \sum_{i=1,2} \sum_{x \in \chi_i} (x - m_i)(x - m_i)^T \quad (1-9)$$

Then, the optimal direction of projection is $\underline{w} = S_W^{-1}(m_1 - m_2)$. All that remains to be determined is a threshold. This is called the *Linear Fisher Discriminant* (LFD) [35]. This can also be interpreted as the optimal Bayes classifier, when the *likelihoods* $p(X/\beta_i)$ in Eq. 1-8 are modeled as multivariate Normal densities with each having the mean m_i and a common covariant matrix as S_W .

1.2.3 Support Vector Machines

Support Vector Machines (SVM) arrive at a linear discriminant driven by *maximum-margin* criterion [36], whose VC dimension provides a loose bound on generalization error [29] (Eq. 1-5). The connection between the margin and VC dimension is given in Eq. 1-6. Given a training set of pattern-label pairs $\langle X_i, y_i \rangle$ where $i \in \{1..N\}$, $X_i \in \mathbb{R}^D$ and $y_i \in \{-1, +1\}$, a separating (canonical) hyperplane is sought such that $y_i(W^T X_i + b) \geq 1 \forall i$ and the margin, $\rho = 1/\|W\|$, is maximal. Margin here is the orthogonal distance between the nearest samples in each class from the hyperplane, assuming that the training data is linearly separable. Since this margin is inversely proportional to $\|W\|_2$ [29], we have to

minimize the Lagrange function :

$$L(W, b, \alpha) = \frac{W^T W}{2} - \sum_{i=1}^N \alpha_i (y_i [X_i^T W + b] - 1) \quad (1-10)$$

where $\alpha_i \geq 0$ are the Lagrange multipliers. This is a quadratic programming problem with a unique minimum when the data is linearly separable. The α_i values reduce to zero for all samples beyond the margin. Hence, the discriminant hyperplane depends only upon the samples, on margin, called the *support vectors*, which is interesting for applications with large samples. A faster numerical technique was proposed by Platt [37] that involves Sequential Minimal Optimization (SMO). Our comparison results are done with the *libSVM* [38] implementation of SMO for the SVM.

The sparsity of the weight vector, W , is used to prune out probable irrelevant features in the feature selection phase of learning applications. Recently, this notion has been used in medical image analysis to identify discriminative locations (voxels) across multiple subjects from fMRI data [39], [40]. For high dimensional sparse data ($N \ll M$), all the samples end up being the support vectors, which does not bode well for the generalizability of the SVM.

We briefly reproduce the recent comments on SVMs, by Bousquet and Scholkopf [41], that address the popular misconceptions persistent in the community of practitioners at large. Statistical learning theory contains analysis of machine learning which is independent of the distribution underlying the data. As a consequence of the *No Free Lunch* theorem, this analysis alone cannot guarantee a priori that SVMs work well in general. A much more explicit assertion would be that *the large margin hyperplanes cannot be proven to perform better than other kinds of hyperplanes independently of the data distribution*. The reasons that make SVM special are the following: (i) the use of *positive definite kernels*, (ii) regularization using the norm in the associated kernel Hilbert space, and (iii) the use of *convex loss* function that is minimized by a classifier.

1.2.4 Kernel Methods

Here, we outline the kernel methods, that make pursuit of non-linear discriminants easier, by simply reducing it to that of the linear-discriminant pursuits. Assume that the two classes of data such that one lies inside the circle and the other lies outside. This data can be separated by a linear discriminant if we add a new dimension defined as the distance of samples from the circle center. It is not trivial to find such maps for a labeled data in general.

Let the higher dimensional feature space be ϕ , in which it is easier to find a linear discriminant for a given data than to find a non-linear boundary for it in the input space. Owing to the connection between the *generalization error*, the VC dimension (Eq. 1-5) and the hyperplane margin (Eq. 1-6), the VC dimension of a large margin hyperplane in the feature space ϕ becomes independent of its dimensionality and could improve upon *generalizability*. However, it is often not possible to explicitly define such a (non-linear) map between the input space and the feature space. Even if it is, the computational aspects might be affected by the *dimensionality curse*.

If all the mathematical operations involved in the learning algorithm, or any task in general, depend only on the inner products in that feature space, it is sufficient to define a valid inner product function, such that $K(x, y) \equiv \phi(x)^T \phi(y)$. In most of the above learning techniques like the SVMs and the Fisher discriminants, an inner product in the Euclidean space \mathbb{R}^D can be replaced by any valid kernel function, $K(x, y)$, and is referred to as the *kernel trick*.

Proposition 1.2.1. (Mercer Kernels [21]) *The function $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ is a Mercer kernel, if and only if for each natural number N and a set of points $\{X_i\}_{1 \dots N}$, the $N \times N$ kernel matrix \mathbf{K} , defined as $\mathbf{K}_{ij} = K(X_i, X_j)$, is positive semidefinite.*

Linear discriminants in the feature space ϕ are now equivalent to rich non-linear boundaries in the original input space \mathbb{R}^D . This is achieved with little or no additional cost. Some of the basic kernels like polynomial (Eq. 1-11), sigmoid (Eq. 1-12) and radial

basis functions (RBF) (Eq. 1-13) are listed below, in which $X, X_j \in \mathcal{X}$.

$$K(X, X_j) = (X^T X_j + c)^d \quad (1-11)$$

$$K(X, X_j) = \tanh((X^T X_j + c)/\rho), \quad \rho > 0 \quad (1-12)$$

$$K(X, X_j) = \exp(-\|X - X_j\|^2/\rho), \quad \rho > 0 \quad (1-13)$$

where ρ, c and the degree d are scalar valued kernel parameters. Typically, the unknown hyperplane orientation (the weight vector W) of the linear discriminant in ϕ can be expressed as a linear combination (Eq.1-14) of mapped input patterns.

$$W = \sum_{i=1}^N \alpha_i \phi(X_i), \quad \alpha_i \in \mathbb{R} \quad (1-14)$$

$$g(X) = b + \sum_{i=1}^N \alpha_i y_i K(X, X_i) \quad (1-15)$$

The resultant non-linear discriminant $g(X)$ is given in Eq. 1-15, where α is a vector of unknown weights to be solved for. The formulations in Sections 1.2.3, 1.2.2 can be transparently adapted to solve for such a linear combination, optimizing their respective criteria.

Now, we have a *model-selection* problem given a host of kernels along with their parameters, each of which is a concept class by itself. The Gaussian RBF kernel is widely successful, especially in applications where the Euclidean distance in input space is meaningful locally [41]. With regards to the polynomial kernel, it is highly probable that the resultant boundary as the degree is increased will become more non-linear, even though they subsume lower degree boundaries in theory. In particular, this could be the case for data with $N \ll M$, since the resultant boundary is a linear combination of degree- d polynomials constructed from each sample point, as $K(X, X_j)$.

1.3 Outline of the Dissertation

In Chapter 2, we present a novel *classification* technique in which each class is assigned a conic section described by a focus (point), a directrix (plane) and an

eccentricity value. Learning involves updating these descriptors for each class. In general, it turned out to be a non-convex non-linear optimization problem. Since the discriminant boundary is a function of distances to foci and directrix hyperplanes, the constraints have interesting geometry that can be exploited to derive tractable learning algorithms. We explain the geometry of learning constraints involved, in Chapter 3.

Assuming, we update one descriptor at a time, the learning constraints are defined such that the value of the discriminant function is fixed at all points except at one misclassified point. This implies that the labels of the other points do not change. So the algorithm either learns a misclassified point or works towards it. Hence, the learning accuracy is non-decreasing. This approach leads to a set of quadratic equality constraints on the descriptors. In Chapter 4, we construct the feasible space in which these constraints are satisfied. We then pick a solution in this space that can learn the misclassified point.

In order to improve upon the generalizability of the CSC, we describe a method in Chapter 5 to estimate geometric margin to the non-linear boundaries due to the CSC. The core idea is to estimate distance of a point to non-linear boundary, by solving for the radius of the smallest hypersphere that intersects the boundary. We found that particular sections of the boundary are at constant distances to each of the class descriptors like a focus or a directrix. This fact is used to estimate the margin by computing exact distances to these sections in an alternating manner.

In Chapter 6, we relax the learning constraints to pursue larger feasible spaces on the class descriptors. First, we can limit the learning constraints to the correctly classified points in a given iteration, instead of all but one misclassified point. A data point is assigned to a class, among two classes, based on the sign of the discriminant function evaluated at that point. Next, instead of constraining the discriminant function to be fixed in value at correctly classified points, we constrain it not to change its sign. This results in quadratic *inequality* constraints on the descriptors. Again, these descriptor constraints have interesting geometry that can be employed to represent sub-regions of the feasible

space with very few parameters. Chapter 7 includes a summary of our contributions with a brief discussion of our future work. A list of symbols used in this dissertation are given in the Appendix.

1.4 Summary

In this chapter, we reviewed the stages involved in an end-to-end supervised classification learning application. We also discussed the desirable characteristics of a concept class and its generalization criterion. Since we introduced a new concept class in this dissertation, we focused on the connection between margins and generalizability in this chapter. It is also clearly established that no method is superior than the other, or a random decider, when no prior information about the data distribution is known. This applies to concept learning, generalization capacity and model selection. So every new concept class is a valid asset as long as it can be shown to perform well on a certain kind of applications in practice. Some concept classes of interest are also briefly discussed. For further reading on machine learning, we refer the reader to the texts by Mitchell [42], Bishop [43], Haykin [44], and Duda, Hart, & Stork [8]. The statistical learning theory and its related concepts are dealt with in detail, in the texts by Vapnik [29], [24], and Hastie, Tibshirani, & Friedman [22].

CHAPTER 2 CONIC SECTION CLASSIFIER

In this chapter, we introduce a novel classifier based on conic sections. Each class is assigned a conic section described by a focus point, a directrix hyperplane and a shape parameter called the eccentricity. First, we motivate the need for a new family of classifier functions, referred to as a concept class in general. Next, we describe the concept class, give an overview of the learning algorithm, and then compare it to other state-of-the-art concept classes.

2.1 Motivation

Without detailed prior knowledge regarding the nature of a dataset, it is not possible in principle to predict which of a given set of concept classes will yield the smallest generalization error. Practitioners therefore resort to applying as many classifiers with different concept classes as possible, before choosing the one that yields the least generalization error. *Every new concept class with a corresponding tractable learning algorithm is consequently a potential asset to a practitioner since it expands the set of classifiers that can be applied to a dataset.*

The learning task becomes remarkably difficult when the number of training samples available is far fewer than the number of features used to represent each sample. We encounter such high dimensional sparse datasets in several applications like the diagnosis of Epilepsy based on brain MRI scans [3], the diagnosis of various types of Cancer from micro-array gene expression data [1], spoken letter recognition [45], and object recognition from images [46], to name only a few. The supervised learning problem is severely under constrained when one is given N labeled data points that lie in \mathbb{R}^M where $N \ll M$. This situation arises whenever the “natural” description of a data point in the problem domain is very large, and the cost of collecting large number of labeled data points is prohibitive.

In such scenarios, learning even a simple classifier such as a linear discriminant is under-constrained because one has to solve for $M+1$ parameters given only N constraints.

Additional objectives, such as maximizing the functional margin of the discriminant, are usually introduced to fully constrain the problem. The learning problem becomes progressively difficult as the concept class gets richer, since such concepts require larger number of parameters to be solved for, given the same number of constraints. This often leads to over-fitting, and the generalization capacity of the classifier suffers.

There are two kinds of traditional solutions to this quandary. In the first approach, the classifier is restricted to the simplest of concept classes like the linear Fisher Discriminant [35], the linear Support Vector Machine (SVM), etc. In the second approach, the dimensionality of the dataset is reduced either by a prior feature selection [47],[48] or by projecting the data onto discriminative subspaces [17]. The criterion for projection may or may not incorporate discriminability of the data, such as in PCA versus Large Margin Component Analysis [18], respectively. The assumption underlying the second approach is that there is a smaller set of compound features that is sufficient for the purpose of classification. Our principal contribution in this dissertation, expands the power of the first approach noted above, by presenting a novel concept class along with a tractable learning algorithm, well suited for high-dimensional sparse data.

2.2 Synopsis

We introduce a novel concept class based on conic sections in Section 2.3. Each member class in the dataset is assigned a conic section in the input space, parameterized by its focus point, directrix plane, and a scalar valued eccentricity. The eccentricity of a point is defined as the ratio between its distance to a fixed focus, and to a fixed directrix plane. The focus, and directrix descriptors of each class attribute eccentricities to all points in the input space \mathbb{R}^M . A data point is assigned to the class to which it is closest in eccentricity value. The concept class is illustrated in Figure 2-2. The resultant discriminant boundary for two-class classification turns out to be a pair of polynomial surfaces of at most degree 8 in \mathbb{R}^M , and thus has finite VC dimension [25]. Yet, it can represent these highly non-linear discriminant boundaries with merely four times the

number of parameters as a linear discriminant. In comparison, a general polynomial boundary of degree d requires $O(M^d)$ parameters, where M is the dimensionality of the input space.

Given a labeled dataset, learning involves arriving at appropriate pair of conic sections (i.e., their directrices, foci, and eccentricities) for the classes, that reduces empirical risk, and results in a discriminant that is simpler, and hence more generalizable. The equivalent numerical formulation turns out to be a severely under-constrained nonlinear non-convex optimization. In Chapters 4, 5, 6, we present tractable geometric algorithms for binary classification that update the class descriptors in an alternating manner. This chapter explains the novel conic section concept class, introduced in [49].

2.3 The Conic Section Concept Class

A conic section in \mathbb{R}^2 is defined as the locus of points whose distance from a given point (the *focus*), and that from a given line (the *directrix*), form a constant ratio (the *eccentricity*). Different kinds of conic sections such as ellipse, parabola, and hyperbola, are obtained by fixing the value of the eccentricity to < 1 , $= 1$, and > 1 , respectively. Conic sections can be defined in higher dimensions by making the directrix a hyperplane of co-dimension 1. Together, the focus point, and the directrix hyperplane generate an *eccentricity function* (Eq. 2-1) that attributes to each point $X \in \mathbb{R}^M$ a scalar valued eccentricity defined as:

$$\varepsilon(X) = \frac{\|X - F\|}{b + Q^T X} ; \text{ where } \|Q\| = 1 \quad (2-1)$$

Hereafter, we use $\|\cdot\|$ to denote the Euclidean \mathbb{L}_2 norm. $F \in \mathbb{R}^M$ is the focus point, and $(b + Q^T X)$ is the orthogonal distance of X to the directrix represented as $\{b, Q\}$, where $b \in \mathbb{R}$ is the offset of the directrix from the origin, and $Q \in \mathbb{R}^M$ is the unit vector that is normal to the directrix. Setting $\varepsilon(X) = e$ yields an axially symmetric conic section in \mathbb{R}^M . In Figure 2-1, we display various conic sections for the following eccentricities: [0.5, 0.8, 1.0, 2.0, 6.0, -10, -3, -2 -1.7]. At $e = 0$, a conic section collapses to the focus point. As $|e| \rightarrow \infty$, it becomes the directrix plane itself.

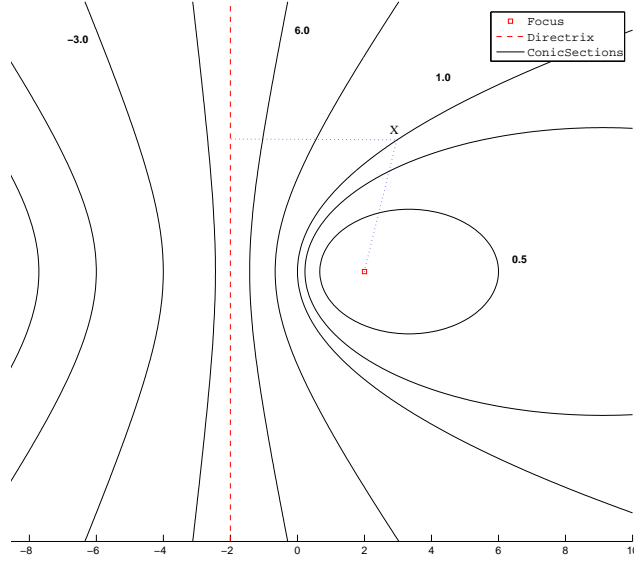


Figure 2-1. Conic sections in \mathbb{R}^2 for various eccentricity values. Eccentricity is the ratio of a point's distance to the focus, F , over to that of the directrix line

We are now in a position to formally define the concept class for binary classification. Each class, $k \in \{1, 2\}$, is represented by a distinct conic section parameterized by the descriptor set: focus, directrix, and eccentricity, as $C_k = \{F_k, (b_k, Q_k), e_k\}$. For any given point X , each class attributes an eccentricity $\varepsilon_k(X)$, as defined in Eq. 2-1, in terms of the descriptor set C_k . We refer $\langle \varepsilon_1(X), \varepsilon_2(X) \rangle$ as the class *attributed eccentricities* of X . We label a point as belonging to that class k , whose class eccentricity e_k is closest to the sample's class attributed eccentricity $\varepsilon_k(X)$, as in Eq. 2-2. The label assignment procedure is illustrated in Figure 2-2.

$$class(X) = \operatorname{argmin}_k (|\varepsilon_k(X) - e_k|) \quad (2-2)$$

$$g(X) = |\varepsilon_1(X) - e_1| - |\varepsilon_2(X) - e_2| \quad (2-3)$$

The resultant discriminant boundary (Eq. 2-3) is the locus of points that are equidistant to the class representative conic sections, in eccentricity. The discriminant boundary is defined as $\mathcal{G} \equiv \{X : g(X) = 0\}$ where $g(X)$ is given by Eq. 2-3. The

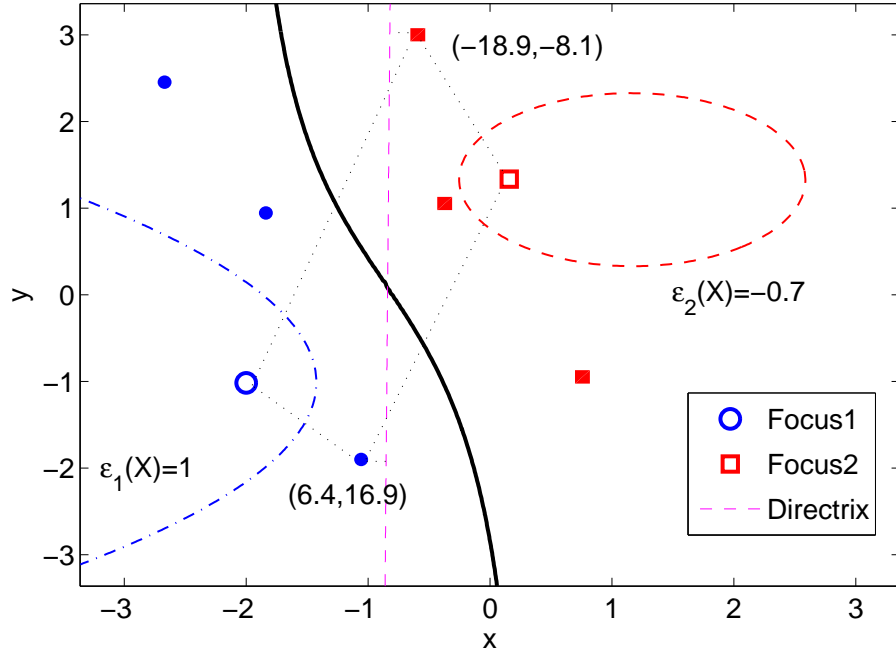


Figure 2-2. Overview of the concept class. Circles and squares are samples from classes 1, & 2. A conic section, $\varepsilon(X) = e$, is the locus of points whose eccentricity is constant. Eccentricity is defined as the ratio between a point's distance to a focus point, and to a directrix line. Each class is assigned a conic section. Both the classes share a common directrix here. For a point in each class, the class attributed eccentricities, $\langle \varepsilon_1(X), \varepsilon_2(X) \rangle$, are shown. A sample is assigned to the class to which it is closest to, in eccentricity, *i.e.* it belongs to class 1 if $|\varepsilon_1(X) - 1| < |\varepsilon_2(X) + 0.7|$. The resultant discriminant is the thick curve.

discriminant boundary equation can be expanded to:

$$\left(\frac{r_1}{h_1} - e_1\right) = \pm \left(\frac{r_2}{h_2} - e_2\right), \quad (2-4)$$

$$\Rightarrow ((r_1 - e_1 h_1) h_2) = \pm ((r_2 - e_2 h_2) h_1), \quad \text{where}$$

$$r_k(X) = \sqrt{(X - F_k)^T (X - F_k)} \quad (2-5)$$

$$h_k(X) = X^T Q_k + b_k \quad (2-6)$$

Upon re-arranging the terms in Eq. 2-4, and squaring them, we obtain a pair of degree-8 polynomial surfaces, in X , as the discriminant boundary.

$$\left((r_1 h_2)^2 + (r_2 h_2)^2 - ((e_1 \mp e_2) h_1 h_2)^2 \right)^2 - (2r_1 r_2 h_1 h_2)^2 = 0 \quad (2-7)$$

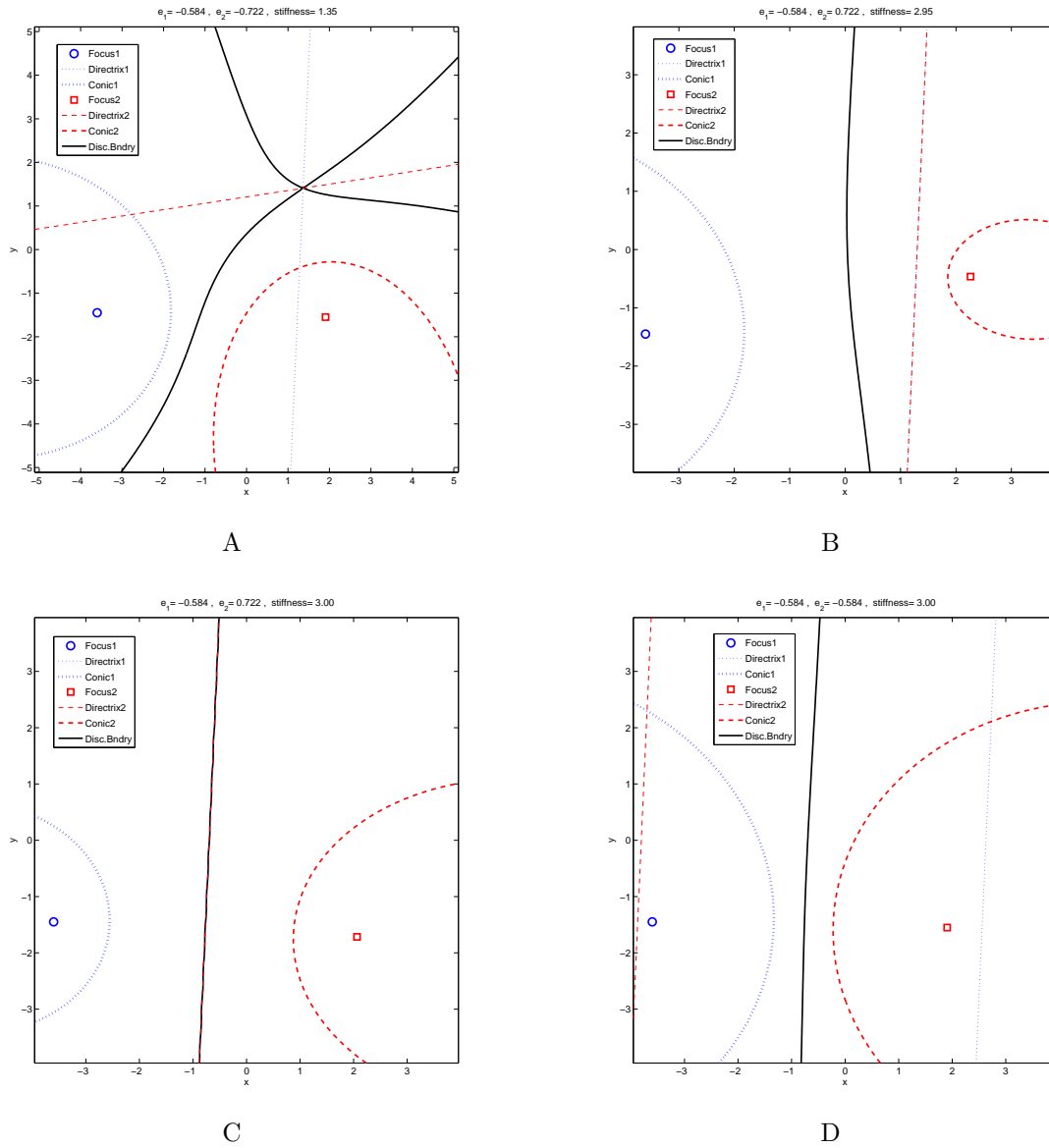


Figure 2-3. Discriminant boundaries for different class conic configurations in \mathbb{R}^2 :
 (A) Non-linear boundary for a random configuration. (B) Simpler boundary: directrices are coincident. Sign of e_2 is flipped to display its conic section.
 (C) Linear boundary: directrices perpendicularly bisect the line joining foci.
 (D) Linear boundary: directrices are parallel and eccentricities are equal.
 (See Section 2.3)

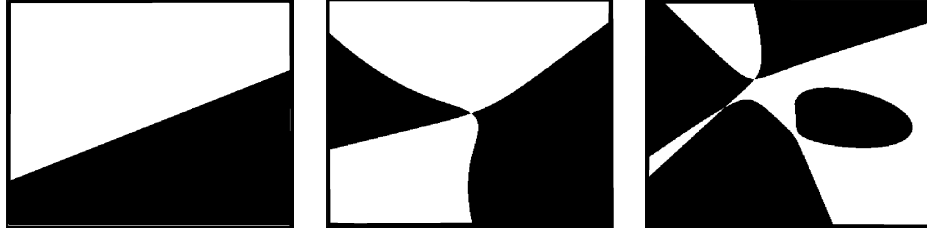


Figure 2-4. Discriminant boundaries in \mathbb{R}^2 for $K = 2$

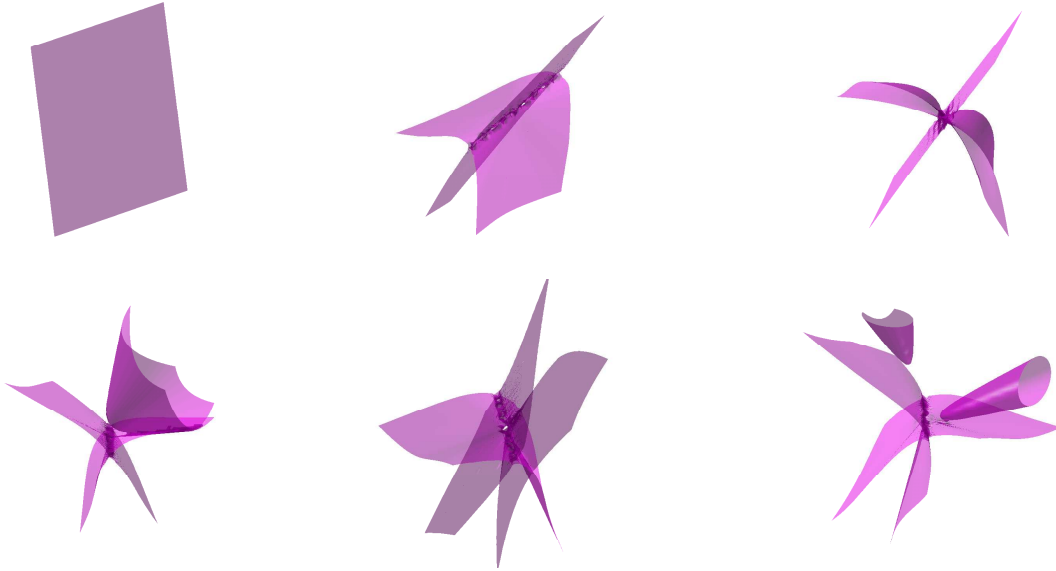


Figure 2-5. Discriminant boundaries in \mathbb{R}^3 for $K = 2$

Depending upon the choice of the conic section descriptors, $\{C_1, C_2\}$, the resultant discriminant can yield lower order polynomials as well. The boundaries due to different class conic configurations in \mathbb{R}^2 , are illustrated in Figure 2-3. When the normals to the directrices, Q_1, Q_2 , are not parallel, the discriminant is highly non-linear (Figure 2-3A). A simpler boundary is obtained when directrices are coincident, as in Figure 2-3B. We obtain linear boundaries when either directrices perpendicularly bisect the line joining foci (Figure 2-3C) or directrices are parallel and eccentricities are equal (Figure 2-3D). Some simple to complex discriminant surfaces in \mathbb{R}^2 , and \mathbb{R}^3 are illustrated in Figures 2-4,2-5 respectively. A list of symbols used in this dissertation are given in the Appendix.

The concept class, referred to as Conic Section classifier (CSC) hereafter, has several notable features. Learning involves arriving at conic descriptors so that the given data samples are well separated. Our learning techniques pursue boundaries that are simple and ensure large functional margins. Our intent is not to fit conic sections to samples from each class, but to learn a generalizable boundary between classes with fewer parameters. Regardless of the dimensionality of the input space, the discriminant is always linear under certain conditions. The linear discriminants can be arrived at, when the directrices for the two classes are identical, the foci lie symmetrically on the two opposite sides of the directrix, the line joining the foci is normal to the directrix, and/or the class eccentricities are equal, and lie in a certain range near zero. The concept class therefore subsumes linear discriminants. We can obtain boundaries ranging from simple to complex, by varying the class conic descriptors. The number of parameters necessary to specify discriminant boundaries due to the conic section concept class is $4 * (M + 1)$. For instance, this is far less than the M^2 parameters required for a general quadratic surface.

2.4 Learning Overview

Given a set of N labeled samples $P = \{\langle X_1, y_1 \rangle, \dots, \langle X_N, y_N \rangle\}$, where $X_i \in \mathbb{R}^M$, and the label $y_i \in \{-1, +1\}$, we assume the data to be sparse in a very high dimensional input space such that $N \ll M$. The empirical risk, L_{err} , is defined as:

$$L_{err} = \frac{1}{N} \sum_i \mathbb{I}(y_i \cdot g(X_i) > 0) \quad (2-8)$$

where \mathbb{I} is the indicator function.

In this dissertation, we present learning algorithms for a two-class classification problem. Learning involves finding the class conic descriptors, that reduce empirical learning risk, and result in a discriminant that is simpler. The equivalent numerical formulation turns out to be a severely under-constrained non-convex optimization. So we alternately update the focus, directrix, and eccentricity descriptors, by exploring the geometry of the learning constraints involved. An $O(N^2)$ algorithm to find the optimal

class eccentricities, $\{e_1, e_2\}$, that minimize learning risk is explained in Chapter 4. We construct a feasible space for each descriptor (focus or directrix) related to the constraint that their labeling of classified points remains unchanged. We represent the feasible space, referred to as the *Null Space*, as a compact geometric object that is built incrementally from the given constraints in the learning phase. We introduce a *stiffness* criterion that captures the extent of non-linearity in the resultant discriminant boundary, given the conic section descriptors. It is used to pursue simpler discriminants by selecting appropriate solutions from the descriptor *Null Spaces*, and thereby improving upon the generalizability of the classifier.

However, finding such optimal foci, and directrices is non-trivial in \mathbb{R}^M . Hence, we introduced learning constraints on these descriptors such that the data points that are correctly classified in an iteration are not misclassified in future iterations. Let Λ^* be a focus or directrix descriptor of a class in a given iteration, that is being updated to Λ . First, we are interested in finding all Λ such that for each correctly classified point X_i :

$$g(X_i, \Lambda) \equiv g(X_i, \Lambda^*) \quad (2-9)$$

$$\Rightarrow \|X_i - F\| = r_i \quad (2-10)$$

$$\Rightarrow b + Q^T X_i = h_i \quad (2-11)$$

When Λ is a focus or directrix, we obtain Eqs.2-10,2-11 respectively. Here, the scalar values r_i , and h_i are the distances to the focus, and directrix descriptors in the previous iteration. We construct the feasible space of Λ that satisfies these constraints, in an incremental manner, and represent it as a tractable geometric object. Any Λ in the feasible space will not mis-classify the points that are correctly classified in a previous iteration. Next, we pick one particular solution from this feasible space that can learn a misclassified point. The learning algorithm in Chapter 4 exploits the constraints listed in Eq. 2-9. In Chapter 6, we pursue larger feasible spaces by finding all Λ such that the sign of the discriminant function remains unchanged at each correctly classified point. This

constraint can be written as :

$$\text{sign}(g(X_i, \Lambda)) \equiv \text{sign}(g(X_i, \Lambda^*)) \quad (2-12)$$

This leads to inequality constraints related to those in Eqs.2-10,2-11. Equivalently, each correctly classified point X_i requires a focus or directrix to lie within a certain interval of distances from itself. The geometry of such constraints on the focus, and directrix descriptors is discussed in Chapter 3.

The generalization capacity of the classifier can be improved by pursuing large geometric margins from the discriminant boundary [27],[50]. In Chapter 5, we estimate distance to the non-linear discriminant boundary due to the Conic Section Classifier (CSC). Each misclassified point results in a desired update for a given descriptor. Among the set of candidate updates, we use a large margin criterion to pick the best one.

2.5 Related Work

Conic sections have been used extensively in several Graphics, and Computer Vision problems like curve fitting [51],[52], and recovering conics from images [53] to infer structure from motion, etc. The principal reasons for this usage is that a large variety of curves can be represented with very few parameters, and that the conic sections are quite common in occurrence. Within the domain of supervised learning, there is one instance in which conic sections were used. One can obtain a conic section by intersecting a cone with a plane at a certain angle. The angle is equivalent to the eccentricity. When the angle is changed, different conic sections can be obtained. This notion was combined with neural networks in [54] to learn such an angle at each node. However, the other descriptors, namely the focus, and directrix are fixed at each node unlike our approach.

Support Vector Machines (SVM), and Kernel Fisher Discriminant (KFD) with polynomial kernel also yield polynomial boundaries like our method does. The set of discriminants due to these classifiers can have a non-empty intersection with those due to the conic section concept class. That is, they do not subsume the boundaries that

result from the latter, and vice-versa. We emphasize here that there is no known kernel equivalent to the conic section concept class for SVM or KFD, and hence the concept class is indeed novel. Next, a detailed comparison to these concept classes is presented.

Comparisons to other Classifiers

We compare CSC to the Support Vector Machines (SVM), and to the Kernel Fisher Discriminants (KFD) [55], with polynomial kernels as they appear to be related to CSC in the type of discriminants represented. For both the classifiers, the discriminant boundary can be defined as:

$$b + \sum_{i=1}^N w_i (X^T X_i + 1)^d = 0 \quad (2-13)$$

where w is a weight vector. Here the decision surface is a linear combination of N degree- d polynomial surfaces defined from each of the data points X_i . The methods differ in their generalization criteria to arrive at the weights w_i . Note that the discriminants due to CSC (Eq. 2-7) cannot be expressed in terms of the boundary due to (Eq. 2-13). There could be a non-empty intersection between the set of the polynomial surfaces represented by CSC, and those due to Eq. 2-13. We point out that there is no kernel which matches this concept class, and therefore, the concept class is novel.

KFD seeks boundaries that maximize Fisher criterion [35], *i.e.* maximize inter-class separation while minimizing within class variance. The learning criterion used in SVM is to pursue large functional margin, resulting in lower VC dimension [27], and thereby improving generalizability. CSC uses similar criterion, and in fact goes a step further. The degree of polynomial kernel is a model-selection parameter in kernel based classifiers like SVM, and Kernel Fisher Discriminants (KFD). As the learning algorithm in CSC will involve arriving at simpler boundaries for the same constraints on the training samples, the degree of polynomial is also being learnt in effect (Figure 2-3).

The advantage of SVM over CSC is that learning in the former involves a convex optimization. The optimization in KFD is reduced to that of matrix inverse problem for binary classification. However, the equivalent numerical formulation for CSC turns out

to be non-convex, and intractable. So we use novel geometric approaches to represent the entire feasible space for constraints on each of the conic descriptors, and then pick a local optimum. In fact, we reduced the feasible space pursuit problem in CSC into that of Gram-Schmidt orthogonalization [56] (Section 6.1.2.2).

When SVM deals with high-dimensional sparse datasets, most of the data points end up being support vectors themselves, leading to about $N * (M + 1)$ parameters, whereas CSC employs only $4 * (M + 1)$ parameters. The boundary due to KFD also involves same number of parameters as SVM. In summary, CSC has some unique benefits over the state-of-the-art techniques that make it worth exploring. These include incorporating quasi model-selection into learning, and shorter description of the discriminant boundary. We also found that CSC out-performed SVM, and KFD with polynomial kernel in many classification experiments.

2.6 Conclusions

In this chapter, we introduced a novel concept class based on conic sections. The concept class has finite VC dimension, can represent highly non-linear boundaries with merely $4 * (M + 1)$ parameters, and subsumes linear discriminants.

CHAPTER 3 THE GEOMETRY OF CONSTRAINTS

In the learning phase, when all the class descriptors except a focus are fixed, the constraints on the unknown focus turn out to be a set of equalities or inequalities on each classified data point's distance to the focus. The locus of all foci that are at a certain distance to a data point is a hypersphere in \mathbb{R}^M . The feasible space of the focus satisfying all the equality constraints due to Eq. 2-9 is the intersection of a set of hyperspheres.

In this chapter, we define the geometric objects of interest like hyperspheres, hyperplanes, and regions between hyperspheres. We will also elaborate upon constituent operations like intersections, projections, and sampling from the feasible spaces. Then, we present geometric intuitions to incrementally construct these feasible spaces for both the focus, and the directrix descriptors.

3.1 Geometric Primitives

Let $X, Q, P \in \mathbb{R}^n$, and $r \in \mathbb{R}^+$. The symbols used from hereon are relevant exclusively in this chapter alone.

Definition. A *hyperplane* H^{n-1} is a linear subspace in \mathbb{R}^n of co-dimension 1. It can be represented as a locus of all points such that

$$H^{n-1} = \{X \in \mathbb{R}^n : X^T Q + b = 0, \|Q\| = 1, b \in \mathbb{R}\} \quad (3-1)$$

From here on, we use $\|\cdot\|$ to represent the Euclidean \mathbb{L}_2 norm. A *hyperplane* is a point in \mathbb{R}^1 , a line in \mathbb{R}^2 , and a plane in \mathbb{R}^3 .

Definition. A *hypersphere* $S^{n-1} \in \mathbb{R}^n$ is the locus of all points at a constant distance from a center P (Eq. 2-10), and defined as

$$S^{(n-1)} = \{X \in \mathbb{R}^n : \|X - P\| = r\} \quad (3-2)$$

A hypersphere is a point-pair in \mathbb{R}^1 , a circle in \mathbb{R}^2 , and a sphere in \mathbb{R}^3 , as illustrated in Figure 3-1.

Table 3-1. Summary of the geometric objects

Object	Hyperplane	Hypersphere	Ball	Shell
Definition	$X^T Q + b = 0$	$\ X - P\ = r$	$\ X - P\ \leq r$	$r_l \leq \ X - P\ \leq r_u$
In \mathbb{R}	H^0 : point	S^0 : two points	B^1 : line segment	two line segments
In \mathbb{R}^2	H^1 : line	S^1 : circle	B^2 : disc	A^2 : annulus
In \mathbb{R}^3	H^2 : plane	S^2 : sphere	B^3 : solid sphere	A^3 : shell
In \mathbb{R}^{n+1}	H^n	S^n	B^{n+1}	$A^{n+1} = S^n \times B^1$
Parameters	(Q, b)	(P, r)	(P, r)	(P, r_l, r_u)

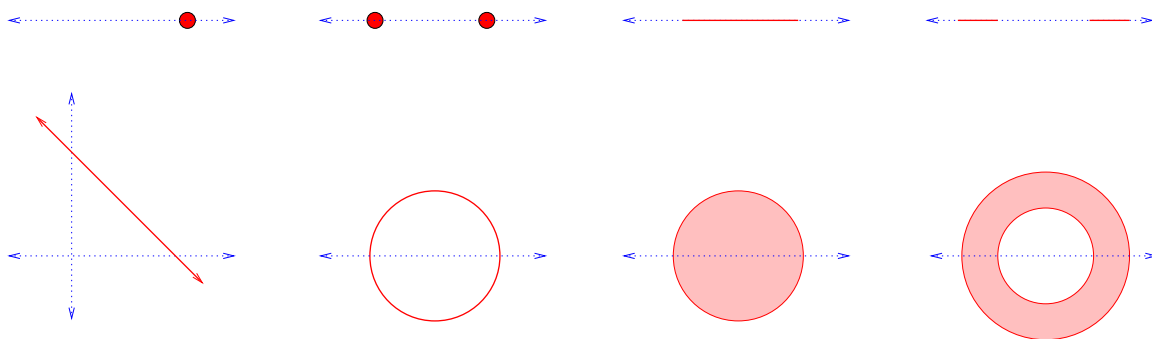


Figure 3-1. The objects: plane, sphere, ball, and shell are illustrated in \mathbb{R} (top row), and \mathbb{R}^2 (bottom row).

Definition. A *ball*, $B^n \in \mathbb{R}^n$, is the region enclosed by a hypersphere.

$$B^n = \{X \in R^n : \|X - P\| \leq r\} \quad (3-3)$$

A ball is a line segment in \mathbb{R}^1 , a disc in \mathbb{R}^2 , and a solid sphere in \mathbb{R}^3 .

Definition. A *shell*, $A^n \in \mathbb{R}^n$, is the region between two concentric hyperspheres. This is the feasible space of an inequality constraint on a focus point, as in Eq. 2-12.

$$A^n = \{X \in R^n : r_l \leq \|X - P\| \leq r_u, \quad r_u \geq r_l \geq 0\} \quad (3-4)$$

The shell in \mathbb{R}^2 is commonly referred to as *annulus* or a ring (Figure 3-1). A^n is its generalization to \mathbb{R}^n .

In Figure 3-1, these geometric objects in \mathbb{R} , and \mathbb{R}^2 are illustrated. Their geometry in \mathbb{R} is important to understand certain intersections of interest. For instance, intersections

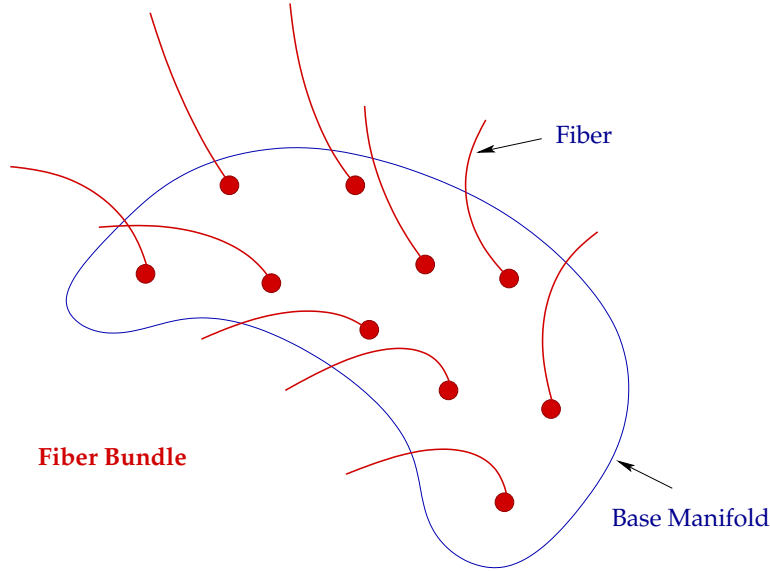


Figure 3-2. A fiber bundle. All the fibers are isomorphic to each other. The bundle can be understood as a product between the base manifold, and a fiber structure. The illustration is inspired by a figure in [57].

of *shells* contain tractable sub-regions that can be expressed as fiber bundle products. We present the definitions for fiber bundle, given in [57].

Definition. Given a map $f : X \rightarrow Y$, a *fiber* is the pre-image of an element $y \in Y$, expressed as :

$$f^{-1}(y) = \{x \in X : f(x) = y\} \quad (3-5)$$

Definition. Let B be a base space, and F be a fiber. A *fiber bundle* with the fiber F is a continuous onto map $\pi : E \rightarrow B$, where E is the total space of the fiber bundle.

A map can be a fiber bundle only if every point in the base space, $b \in B$, has an open neighborhood U such that $\pi^{-1}(U)$ is homeomorphic to $U \times F$.

The total space E will be loosely referred to as the fiber bundle. Locally E looks like a product space $B \times F$. For instance, the tangent bundle, TM , is a fiber bundle where the manifold, M , is the base space, and the fiber is the tangent space $T_x M$ at a point $x \in M$.

Figure 3-2 illustrates the intuition behind a fiber bundle. A *shell* can also be interpreted as

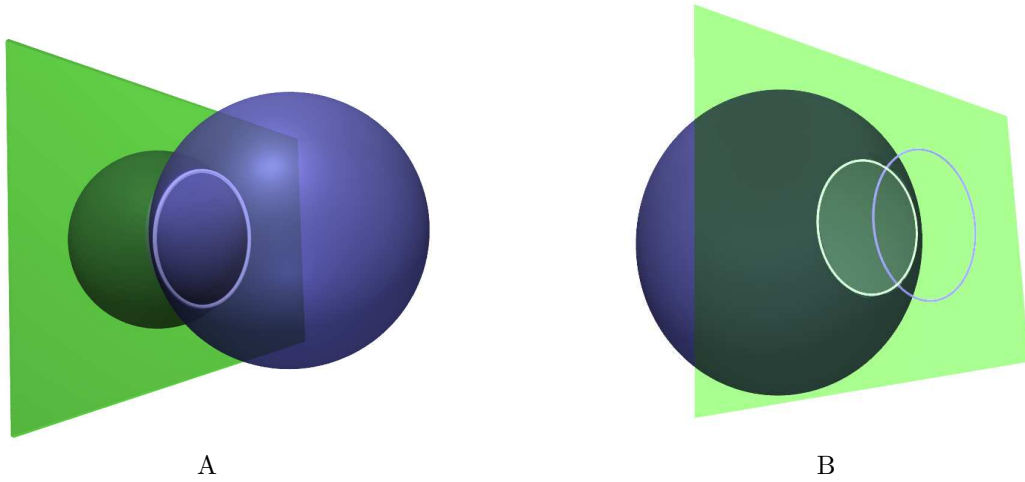


Figure 3-3. (A) Intersection of two spheres is the circle lying the plane of intersection.
 (B) Intersection of a third sphere with the hyperplane. Thus, intersection of three spheres is reduced to that of two circles in the hyperplane.

a fiber bundle of line segments (fibers) over a hypersphere base, *i.e.*

$$A^n = S^{n-1} \times B^1 \quad (3-6)$$

3.2 Intersections of *Hyperspheres*

The feasible space of each equality constraint (Eq. 2-10) on a focus descriptor, when considered separately, turns out to be a hypersphere. Since the focus descriptor in the previous iteration satisfied all the constraints due to correctly classified points, the respective hyperspheres are guaranteed to intersect. In the limiting case, the *Null Space* of all the constraints put together collapses to the previous focus descriptor. We elaborate upon the geometry involved assuming that the intersections always exist.

Proposition 3.2.1. *Intersection of two hyperspheres is a hypersphere of one lower dimension, which lies in the intersection hyperplane of co-dimension 1.*

$$S^n \cap S^n = S^{n-1} \in H^n \quad (3-7)$$

Proposition 3.2.2. *Intersection of a hypersphere, and a hyperplane is a hypersphere of one lower dimension, which lies in the same hyperplane.*

$$S^n \cap H^n = S^{n-1} \in H^n \quad (3-8)$$

Lemma 3.2.3. *Intersection of k hyperspheres in \mathbb{R}^{n+1} can be reduced to that of $(k - 1)$ hyperspheres lying in a hyperplane H^n .*

$$\begin{aligned} S_1^n \cap S_2^n &= S_2^{n-1} \in H^n \\ S_i^n \cap H^n &= S_i^{n-1} \in H^n, \forall i \in \{3, \dots, k\} \\ \Rightarrow \bigcap_{i=1}^k S_i^n &= \bigcap_{i=2}^k S_i^{n-1} \in H^n \end{aligned}$$

Theorem 3.2.4. *Intersection of k hyperspheres in \mathbb{R}^{n+1} is a single hypersphere lying in a hyperplane of co-dimension $(k - 1)$, i.e.*

$$\bigcap_{i=1}^k S_i^n = S^{n-(k-1)} \in H^{(n+1)-(k-1)} \quad (3-9)$$

Proof. Repeat the reduction in Lemma 3.2.3 $(k - 1)$ times to obtain Eq. 3-9. □

Recall that the intersection of k hyperspheres is the *Null Space* within which all the equality constraints on a focus descriptor are satisfied. Now, the *Null Space* can be represented as a single hypersphere in a lower dimensional hyperplane. The analytical formulae to compute the *Null Space* are discussed in Sections 4.1.3.1, 6.1.2.

3.3 Intersections of *Shells*

Intersection of two shells is illustrated in Figure 3-4. The intersection for the former can also be denoted as the fiber bundle $S^0 \times B^2$. Similarly, their intersection in \mathbb{R}^3 is the fiber bundle $S^1 \times B^2$, referred to a toroid. In fact, torus is the fiber bundle $S^1 \times S^1$. We will now generalize these toroidal regions lying within the intersection of shells to \mathbb{R}^n .

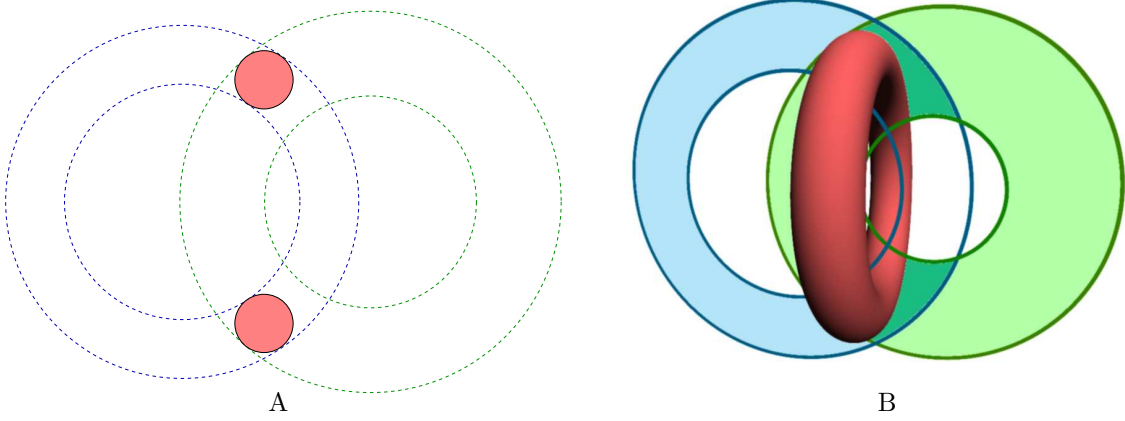


Figure 3-4. A *shell-shell* intersection in \mathbb{R}^2 , and \mathbb{R}^3 . In (A), we can always define two discs that lie within the intersection. In (B), only the cross-section of the shells is shown. Similarly, we define a toroidal region within the intersection on shells.

Definition. A *spherical toroid* $T^{(n-k),k}$ is defined as a fiber bundle of a *ball* fiber B^k over the *hypersphere* base space $S^{(n-k)}$.

$$T^{(n-k),k} = S^{(n-k)} \times B^k \in R^n \quad (3-10)$$

Here, B^k is in the hyperplane spanning the local normal hyperplane to a point on the hypersphere $S^{(n-k)}$, and the center of $S^{(n-k)}$.

Recall that a *shell* $A^n = S^{(n-1)} \times B^1$. From Figure 3-4, we have :

$$T^{(0,2)} = S^0 \times B^2 \subset \{(S^1 \times B^1) \cap (S^1 \times B^1)\}$$

$$T^{(1,2)} = S^1 \times B^2 \subset \{(S^2 \times B^1) \cap (S^2 \times B^1)\}$$

Lemma 3.3.1. *Intersection of two shells in \mathbb{R}^n contains a spherical toroid $T^{(n-2,2)}$, i.e. :*

$$T^{(n-2,2)} = S^{(n-2)} \times B^2 \subset A^n \cap A^n \quad (3-11)$$

Theorem 3.3.2. *Given a $\Lambda^* \in S_i^{(n-1)} \forall i \in \{1 \dots k\}$, there exists a spherical toroid, $T^{(n-k,k)}$ within the intersection of k shells, $A_i^n \supset S_i^{(n-1)}$.*

$$T^{(n-k,k)} = S^{(n-k)} \times B^k \subset \cap_{i=1}^k A_i^n \quad (3-12)$$

Proof. We will prove this by constructing the largest spherical toroid lying within the intersection of shells, whose base hypersphere contains Λ^* .

1. Fix the base space as the hypersphere $S^{n-(k-1)} = \cap_{i=1}^k S_i^n$ from Theorem 3.2.4.
2. The fiber is a ball B^k whose largest radius can be determined from the upper, and lower bounds on the distance of each correctly classified point to the new focus descriptor (Eq. 2-12).

A detailed proof is presented in Section 6.1.2. □

CHAPTER 4 LEARNING WITH EQUALITY CONSTRAINTS

In this chapter, we present a novel incremental algorithm [49] to learn the conic section descriptors, $C_k = \{F_k, \{b_k, Q_k\}, e_k\}$ for $k = 1, 2$, that minimize the empirical error (Eq. 4–1). We assume a set of N labeled samples $P = \{\langle X_1, y_1 \rangle, \dots, \langle X_N, y_N \rangle\}$, where $X_i \in \mathbb{R}^M$ and the label $y_i \in \{-1, +1\}$, and that the data is sparse in a very high dimensional input space, i.e., $N \ll M$. Learning involves arriving at appropriate class descriptors that minimize the empirical risk :

$$L_{err} = \frac{1}{N} \sum_i \mathbb{I}(y_i \cdot g(X_i) > 0) \quad (4-1)$$

where \mathbb{I} is the indicator function, and g is the discriminant function defined as $g(X) = |\varepsilon_1(X) - e_1| - |\varepsilon_2(X) - e_2|$. The learning algorithm should attempt to enhance the generalizability of the resultant discriminants. We observed that the equivalent numerical formulation leads to a highly non-linear non-convex optimization problem which did not perform well. Instead, we exploit the geometry of constraints on the descriptors, explained in Chapter 3, to arrive at a tractable learning algorithm. In our algorithm, the class descriptors are alternately updated to simultaneously learn misclassified points and pursue simpler (near-linear) boundaries with the constraint that the discriminant values, $g(X_i)$, for all the data points except a misclassified point are held fixed.

4.1 Learning Algorithm

The class descriptors are initialized with a linear discriminant based on means of samples from each class among other techniques (Section 4.1.5). The learning process is then comprised of two stages. In the first stage, the foci and directrices of both the classes are held fixed and the class eccentricities, $\langle e_1, e_2 \rangle$, are updated. Each data point X_i is mapped into the space of class attributed eccentricities, referred to as the *ecc-Space*, by computing, $\langle \varepsilon_1(X_i), \varepsilon_2(X_i) \rangle$. The pair of class eccentricities $\langle e_1, e_2 \rangle$ that minimizes the

empirical risk L_{err} is then computed. For each misclassified sample, one can find a desired pair of attributed eccentricities $\langle \varepsilon'_{1i}, \varepsilon'_{2i} \rangle$ that would correctly classify that sample.

Data: Labeled Samples P
Result: Conic Section Descriptors C_1, C_2

- 1: Initialize $\{F_1, b_1, Q_1\}, \{F_2, b_2, Q_2\}$
- 2: Compute $\langle \varepsilon_1(X_i), \varepsilon_2(X_i) \rangle \quad \forall X_i \in P$
- 3: Find *class-eccentricities* $\langle \hat{e}_1, \hat{e}_2 \rangle$
- if** *learning error is not minimal* **then**
 - 4: Compute the desired $\langle \varepsilon'_{1i}, \varepsilon'_{2i} \rangle$
 - 5: Update *foci & directrices* alternately.
 - 6: Goto (2) until convergence of descriptors.
- end**

Figure 4-1: Algorithm to Learn the Descriptors due to Equality Constraints

In the second stage, the foci $\{F_1, F_2\}$ and the directrices $\{\{b_1, Q_1\}, \{b_2, Q_2\}\}$ are updated alternately so as to achieve the desired attributed eccentricities for those misclassified samples, *without affecting the attributed eccentricities for those samples that are already correctly classified*. The process is repeated until the descriptors converge or there can be no further improvement in classification. The learning procedure is listed in Figure 4-1

4.1.1 Finding Class-Eccentricities $\langle e_1, e_2 \rangle$

Note that the dimensionality of *ecc-Space* is the number of classes (2 in our case). Given the labeled data, foci and directrices for both classes, we now present an $O(N^2)$ algorithm to find the optimal *cross-hair*, i.e. $\langle e_1, e_2 \rangle$. The method begins by rotating *ecc-Space* (Figure 4-2A) around the origin by 45° so that any choice of the discriminants will now be parallel to the new axes, as in Figure 4-2B. Each axis is divided into $(N + 1)$ intervals by projecting the points in *ecc-Space* onto that axis. Consequently, *ecc-Space* is partitioned into $(N + 1)^2$ 2D intervals. We now make a crucial observation: *within the confines of a given 2D interval, any choice of a cross-hair classifies the set of samples identically*. (See Figure 4-2B). We can therefore enumerate all the $(N + 1)^2$ intervals

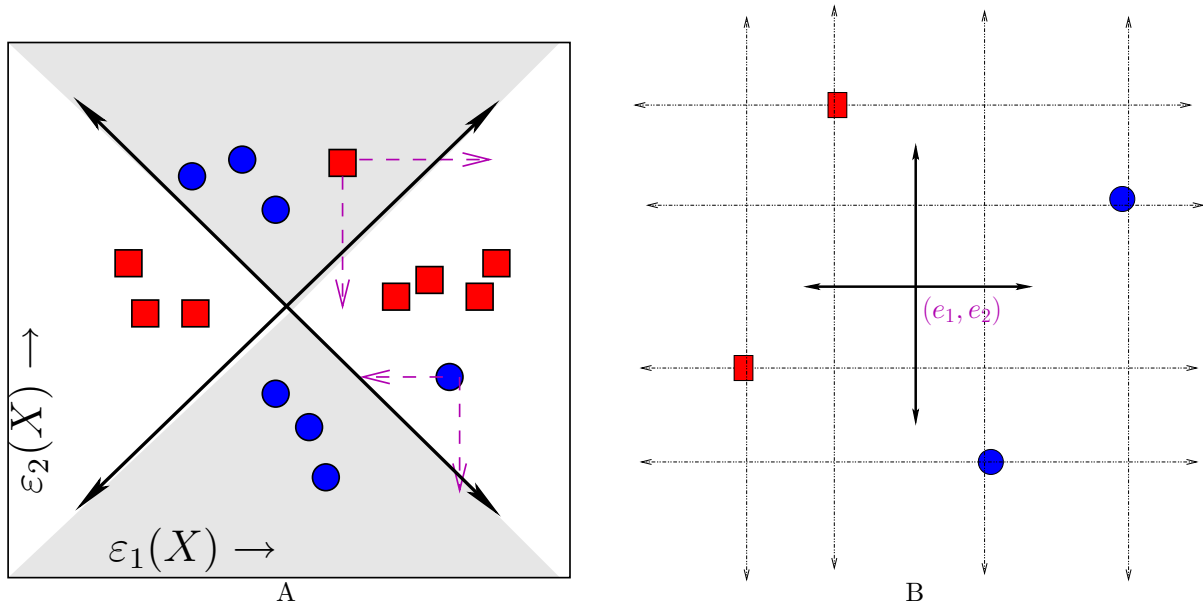


Figure 4-2. (A) Shaded regions belong to one class. The pair of diagonal lines is the discriminant boundary, $|\varepsilon_1(X) - e_1| = |\varepsilon_2(X) - e_2|$. Learning involves shifting misclassified points into desired regions. (B) *ecc-Space* rotated by 45° degrees. When the only unknown is $\langle e_1, e_2 \rangle$, label assignment is invariant within a 2D interval.

and choose the one that gives the smallest classification error. The cross-hair is set at the center of this 2D interval. In cases where there are multiple 2D intervals that give the smallest classification error, the larger one is chosen.

4.1.2 Learning Misclassified Points

Given the attributed eccentricities $\langle \varepsilon_{1i}, \varepsilon_{2i} \rangle$ of a misclassified point, we can compute its desired location $\langle \varepsilon'_{1i}, \varepsilon'_{2i} \rangle$ in *ecc-Space* (see Figure 4-2A) by moving it into the nearest quadrant associated with its class label. This movement can be achieved by updating a focus or directrix in Eq. 2-1. In order to keep the learning process simple, we update only one descriptor of a particular class at each iteration. Hence, we move the misclassified points in *ecc-Space* by changing ε_{1i} or ε_{2i} for the class of the descriptor being updated.

The learning task now reduces to alternately updating the foci and directrices of C_1 and C_2 , so that the misclassified points are mapped into the desired quadrants in *ecc-Space*, while the correctly classified points remain fixed. Note that with such an update,

our learning rate is non-decreasing. We also introduce a *margin* along the discriminant boundary and require the misclassified points to be shifted beyond this margin into the correct quadrant. In most of our experiments the margin was set to 5% of the range of eccentricity values in *ecc-Space*.

4.1.3 Updating The Focus

Our objective here is to achieve the desired attributed eccentricities ε'_{ki} for all the samples by changing the focus F_k . For each correctly classified sample, the desired eccentricity ε'_{ki} is simply its previous value ε_{ki} . From Eq. 2-1 we can conclude that all the ε_{ki} for $k=1$ depend only on the class descriptor C_1 , and likewise for $k=2$. Since we update only one focus at a time, we shall hereafter deal with the case $k = 1$. The update problem may be posed formally as follows. Find a focus F_1 that satisfies the following N quadratic constraints. Let $\|\cdot\|_2$ be the Euclidean \mathbb{L}_2 norm.

$$\|F_1 - X_i\|_2 \begin{cases} = r_{1i}, & \forall X_i \in P_c, \\ \leq \text{or } \geq r_{1i}, & \forall X_i \in P_{mc}, \end{cases} \quad (4-2)$$

$$\text{where, } r_{1i} = \varepsilon'_{1i}(b_1 + Q_1^T X_i) \quad (4-3)$$

In effect, each point X_i desires F_1 to be at a distance r_{1i} from itself, derived from Eq. 4-3. P_c and P_{mc} are the set of classified and misclassified points respectively. The inequalities above imply that the desired location ε'_{1i} can lie in an interval along an axis in *ecc-Space* (See Figure 4-2A). In order to closely control the learning process, we learn one misclassified point at a time, while holding all the others fixed. This leaves us with only one inequality constraint.

We refer to the set of all feasible solutions to the above quadratic constraints as the *Null Space* of F_1 . Further, we have to pick an optimal F_1 in this *Null Space* that maximizes the generalization capacity of the classifier. Although the general Quadratic Programming Problem is known to be NP-hard [58], the above constraints have a nice geometric structure that can be exploited to construct the *Null Space* in $O(N^2M)$ time.

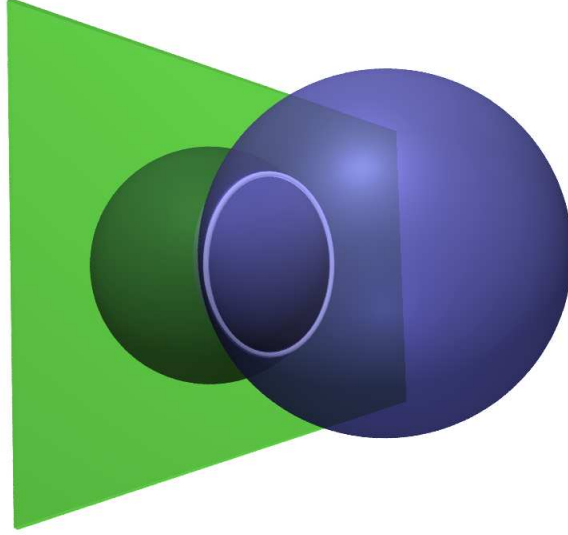


Figure 4-3. Intersection of two hyper-sphere *Null spaces*. The white circle is locus of focal points satisfying current distance constraints.

Note that by assumption, the number of constraints, $N \ll M$. The *Null Space* of F_1 with respect to each equality constraint in Eq. 4-2 is a hyper-sphere in \mathbb{R}^M . Hence, the *Null Space* for all the constraints combined is simply the intersection of all the corresponding hyper-spheres in \mathbb{R}^M with centers $\{X_1, \dots, X_N\}$ and radii $\{r_{11}, \dots, r_{1N}\}$. Let X_N be the single point being updated in *ecc-Space*. Then, r_{1N} can take any value within an interval (r_{min}, r_{max}) corresponding to the range of desired ε'_{1N} . The solution to this case is presented next.

4.1.3.1 The Intersection of Spheres problem

We present an algorithm that builds the *Null Space* incrementally. For ease of readability, we drop the reference to class in this section. The *Null Space* is initialized as the set of feasible solutions for the first equality constraint in Eq. 4-2. It can be parameterized as the hyper-sphere $S_1 = (r_1, X_1)$, where r_1 is the desired distance of a solution from the sample X_1 . At the next step, the second equality constraint is introduced, the *Null Space* for which, considered independently, is the hyper-sphere

$S_2 = (r_2, X_2)$. Hence the combined *Null Space* for the two constraints is the intersection of the two hyper-spheres, $S_1 \cap S_2$.

As illustrated in Figure 4-3, the intersection of two spheres in \mathbb{R}^3 is a circle that lies on the plane of intersection of the two spheres. Our technique is based on a generalization of this setting in \mathbb{R}^M . We make two critical observations: the intersection of two hyper-spheres is a hyper-sphere of one lower dimension, and this hyper-sphere lies on the intersecting hyper-plane of the original hyper-spheres. We re-parameterize the combined *Null Space*, $S_1 \cap S_2$, as a lower-dimensional hyper-sphere $S_{\{1,2\}}$, lying in the hyper-plane of intersection $H_{\{1,2\}}$. Based on the geometry of the problem and the parameterization of S_1 and S_2 , it is trivial to compute, in $O(M)$, the radius and center of the new hyper-sphere $S_{\{1,2\}} = (r_{\{1,2\}}, X_{\{1,2\}})$, as well as the intersecting hyper-plane $H_{\{1,2\}}$ represented as $(b_{\{1,2\}}, Q_{12})$, the first parameter being the displacement of $H_{\{1,2\}}$ from the origin and the second being the unit normal to $H_{\{1,2\}}$. $Q_{\{1,2\}}$ lies along the line joining X_1 and X_2 .

We now solve the remainder of the problem on the hyper-plane $H_{\{1,2\}}$. This is accomplished by intersecting each of the remaining hyper-spheres S_3, \dots, S_N that correspond to the samples X_3, \dots, X_N , with $H_{\{1,2\}}$, in $O(NM)$ time. Once again, based on the geometry of the problem, it is trivial to compute the new radii and centers of the corresponding hyper-spheres. In short, the intersection of the N hyper-spheres problem is converted into the intersection of $(N - 1)$ hyper-spheres and a hyper-plane $H_{\{1,2\}}$ problem.

$$S_1 \cap S_2 \rightarrow S_{\{1,2\}} \in H_{\{1,2\}} \quad (4-4)$$

$$S_i \cap H_{\{1,2\}} \rightarrow S'_i \in H_{\{1,2\}} \quad \forall i = 3, \dots, N \quad (4-5)$$

The problem is now transparently posed in the lower dimensional hyper-plane $H_{\{1,2\}}$ as a problem equivalent to the one that we began with, except with one less hyper-sphere constraint. The end result of this iteration for all the $(N - 1)$ equality constraints is a low dimensional linear subspace in which lies the *Null Space* S^e represented as a single hyper-sphere (parameterized as a radius and a center), computed in $O(N^2M)$ time. It

should be observed that all the intersections thus far are feasible and that the successive *Null Spaces* have non-zero radii since the equality constraints have a feasible solution *a priori*.

Let S_N be the null space for the inequality constraint with $r_N \in (r_{min}, r_{max})$. If S_N intersects with S^e , we chose a radius r_N that maximally shifts the corresponding misclassified point in *ecc-Space*. On the resultant final *Null Space*, we picked a solution that improves the generalization capacity of the classifier. We have found that any choice of the solution that arrives at a configuration close to that in Figure 2-3D, results in a simpler discriminant in the input space. If S^e does not intersect with S_N , this simply means that the chosen misclassified point can not be shifted entirely to the desired location in *ecc-Space*. In such a case, we picked an appropriate solution on S^e that allows the maximum possible shift.

4.1.4 Updating The Directrix

We demonstrate in this section that the Directrix Update problem is closely related to the Focus Update problem owing to the duality of points and hyper-planes. We begin by once again noting that $\{b_1, Q_1\}$ may be updated independent of $\{b_2, Q_2\}$, and vice versa. The goal here is to update the directrix descriptor $\{b_1, Q_1\}$ for a given F_1 and desired eccentricities ε'_{1i} . Just as in the focus update, each point X_i desires the directrix to be at a certain orthogonal distance v_{1i} , from itself. The problem reduces to finding a *directrix* that satisfies the constraints:

$$b_1 + Q_1^T X_i \begin{cases} = v_{1i} & \forall X_i \in P_c \\ \leq \text{or} \geq v_{1i} & \forall X_i \in P_{mc} \end{cases} \quad (4-6)$$

$$\text{where, } v_{1i} = \| F_1 - X_i \|_2 / \varepsilon'_{1i}, \quad \| Q_1 \|_2 = 1 \quad (4-7)$$

This problem appears simpler at first sight since it is comprised of N linear constraints. However, the quadratic constraint requiring Q_1 to be an unit normal makes the above a Quadratic Programming Problem which is again NP-hard in general. Once

again, we exploit the geometric structure inherent in the problem to arrive at the *Null Space* in $O(N^2M)$ time. We first solve for the scalar b_1 by translating the origin to the first classified point, X_1 , so that $b_1 = v_{11}$. In addition, just as in Section 4.1.3, we learn a single misclassified point, say X_N , in each iteration. With a known b_1 , we translate and scale all remaining points such that the linear constraints become:

$$Q_1^T X_i \begin{cases} = v_{1i} & 2 \leq i < N \\ \leq \text{or} \geq v_{1i} & i = N \end{cases} \quad (4-8)$$

$$\text{where, } X_i = (X_i - X_1) / \|(X_i - X_1)\|_2 \quad (4-9)$$

$$v_{1i} = (v_{1i} - v_{11}) / \|(X_i - X_1)\|_2 \quad (4-10)$$

Now the null space of Q_1 , for each constraint in Eq. 4-8 considered separately, is a hyper-plane $H_i \in \mathbb{R}^M$ represented as $\{-v_{1i}, X_i\}$. The null space corresponding to the quadratic constraint on Q_1 is a unit hyper-sphere, $S_1 \in \mathbb{R}^M$, centered at the new origin. Hence, the final *Null Space* for Q_1 is the intersection of all the H_i 's and S_1 .

We now make two critical observations. The intersection of a hyper-plane with a hyper-sphere is a lower-dimensional hyper-sphere. Same is the case with the intersection of two hyper-spheres. We can therefore convert this hyperplane-hypersphere intersection problem into a hypersphere-hypersphere intersection problem. In effect, we can replace each hyper-plane H_i with a suitable hyper-sphere S_i such that $H_i \cap S_1 = S_i \cap S_1$. Owing to the geometry of the problem, we can compute S_i from H_i and S_1 . The *Null Space* for all the constraints combined is now the intersection of all the hyper-spheres S_1, S_2, \dots, S_N . The problem, now reduced to a hyperspheres-intersection problem, is solved as in Section 4.1.3.1.

4.1.5 Initialization

Given a set of labeled samples, we found that there are several ways of initializing the conic section descriptors that led to a solution. Random initializations converged to different conic descriptors each time leading to inconsistent performance. We observed

that owing to Eq. 2-1, the *Null Spaces* are small or vanishing if the foci or directrices are very close to the samples. We found the following initialization to be consistently effective in our experiments. The foci were first placed at the sample class means and then pushed apart until they were outside the sample clouds. The normals to the directrices were initialized as the line joining the foci. The directrix planes were then positioned at the center of this line or on either sides of the data.

4.1.6 Discussion

One of the core characteristics of our algorithm is that after each update any point that is correctly classified by the earlier descriptors is not subsequently misclassified. This is due to two reasons. First, we begin with an initialization that gives a valid set of assignments for the class attributed eccentricities. This implies that the *Null Space* for the classified points is non-empty. Second, the search for updates in the *Null Space* always guarantees the feasible solution for the constraints related to the correctly classified points.

A key contribution of our technique is the tracking of the set of all feasible solutions as a compact geometric object. From this *Null Space* we pick a solution biased towards a linear discriminant so as to improve upon generalization. The size of margin in *ecc-space* also gives a modicum of control over generalization. The order of samples processed does not affect the final *Null Space*. The convergence of our learning algorithm depends on data and initialization. However, we found that it converged to a local minima typically within 50 iterations of the focus and directrix updates.

4.2 Results

We evaluated the classifier on two synthetic datasets and four real datasets. *viz.* Epilepsy Data [3], Colon Tumor gene-expression data [4], the Sheffield (formerly UMIST) Face Database [59], CURET Texture Database [60]. The results were compared against several state-of-the-art linear and non-linear classifiers. The classification accuracies based on leave-one-out cross-validation are presented in Table 4-1.

4.2.1 Classifiers

We implemented a faster version of the LFD as described in Yu & Yang [61]. This technique exploits the fact that high-dimensional data has singular scatter matrices. They discard the sub-space which carries no discriminative information. Kernel methods are known to be effective non-linear classifiers and the Support Vector Machines (SVM) [36] and Kernel Fisher Discriminants (KFD) [62] broadly represented the non-linear category. Both employ the kernel trick of replacing inner products with Mercer kernels. Among the linear classifiers, we chose the Linear Fisher Discriminant (LFD) [8] and linear SVM. [63] We used the OSU SVM toolbox for MATLAB based on *libSVM* [38]. We considered Polynomial (PLY) and Radial Basis (RBF) Kernels.

The best parameters were empirically explored. Polynomial kernels gave best results with either degree = 1 or 2 and the scale was approximately the sample variance. The RBF kernel performed best when the radius was the sample variance or the mean distance between all sample pairs. The initialization of the descriptors in the learning is described in Sections 4.1.5, 4.1.2 and 4.1.3.1.

4.2.2 Synthetic Data

Synthetic dataset-1 was randomly generated from two well separated Gaussian clusters in \mathbb{R}^{40} . The results in Table 4-1 validate our classifier's effectiveness on simple, linearly separable data. Synthetic dataset-2 was generated by sampling from two intersecting paraboloids (related to the two classes) in \mathbb{R}^3 and placing them in \mathbb{R}^{64} . This instance shows that our classifier favors data lying on paraboloids. It clearly out-performed the other classifiers.

4.2.3 Epilepsy Data

Epilepsy data [3] consists of displacement vector fields between the left and right hippocampi for 31 epilepsy patients. The displacement vectors are computed at 762 discrete mesh points on each of the hippocampal surfaces, in 3D. This vector field representing the non-rigid registration, captures the asymmetry between the left and

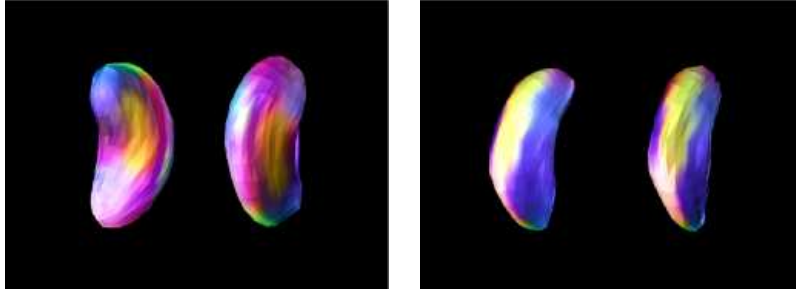


Figure 4-4. Direction of displacement field for epileptics with focus on left and right temporal lobe, respectively. Direction components in (x, y, z) are color coded with Red, Green, Blue components on the hippocampal surface.

right hippocampi. (See Figure 4-4). Hence, it can be used to categorize different classes of epilepsy based on the localization of the focus of epilepsy to either the left (LATL) or right temporal lobe (RATL). The LATL vs. RATL classification is a hard problem. As seen in Table 4-1, our classifier out-performed all the others and with a significant margin, except over SVM-RBF. In fact, our result is better than that reported in [3]. The best RBF kernel parameters for SVM and KFD methods were 600 and 1000, respectively. The best degree for the polynomial kernel was 1 for both of them.

4.2.4 Colon Tumor

The Colon Tumor data [4] comprises of 2000 gene-expression levels for 22 normal and 40 tumor colon tissues. Each gene-expression score is computed from a filtering process explained in [4]. The best parameters are described in Section 4.2.1. The normals to directrix descriptors were initialized with the LFD direction in this case. Our classifier yielded 87% accuracy outperforming the other classifiers. Interestingly, most of the other classifiers could not out-perform LFD, implying that they were learning the noise as well. Terrence, *et al.* [64] were able to correctly classify two more samples with a linear SVM, only after adding a diagonal factor of two to the kernel matrix. We have tested this data with out any further pre-processing.



Figure 4-5. Sand Paper, Rough Paper & Polyester. In the results Table 4-1, we have classification for 6 vs. 12 and 6 vs. 2 respectively.

4.2.5 Sheffield FaceDB

The Sheffield (formerly UMIST) Face Database [59] has 564 pre-cropped face images of 20 individuals with varying pose. Each image has 92 x 112 pixels with 256 gray-levels. Since we only have a binary classifier now, the average classification performance over all possible pairs of subjects is reported. This turned out to be an easier problem. Conic classifier achieved a comparable accuracy of about 98%, while the others were near 100%. Our performance is poorer than LFD here because of the trade off between learning and generalization. illustrated in Figure 2-4.

4.2.6 CURET Textures

CURET database [60] is a collection of 61 texture classes imaged under 205 illumination and viewing conditions. A Varma *et al.* [65] have built a dictionary of 601 textons [66] and proposed ways to generate features for classification based on texton frequencies in a given sample image. The texton frequency histograms obtained from [67], are used as the sample feature vectors for classification. About 47 images were chosen from each class, with out a preferential order so as to demonstrate the efficacy of our classifier for high-dimensional sparse data. We report the results for an easy pair and a relatively tougher pair of textures for classification. The two cases are Sand paper vs. Rough paper (Pair1) and Sand paper vs. Polyester (Pair2), respectively. See Figure 4-5 to note how similar the sand paper and polyester textures are. As seen in Table 4-1, Pair1 turned out to be easier case in deed. KFD out-performed the others for the second pair and our classifier fared comparably.

Table 4-1. Classification accuracies for the Conic Section Classifier, (Linear & Kernel) Fisher Discriminants and SVM.

Samples	Data Size (NxM)	CSC	LFD	KFD PLY	KFD RBF	SVM PLY	SVM RBF
Synthetic Data1	20 x 40	100	100	100	100	100	100
Synthetic Data2	32 x 64	93.75	87.5	75	75	81.25	87.5
Epilepsy	31 x 2286	77.42	67.74	67.74	61.29	67.74	74.19
Colon Tumor	62 x 2000	87.1	85.48	75.81	82.26	82.26	85.48
UMIST FaceDB	575 x 10304	97.74	98.72	99.93	99.91	99.3	99.06
Texture Pair1	95 x 601	100	100	100	100	100	100
Texture Pair2	95 x 601	92.63	98.94	100	100	90.52	82.10

4.3 Summary and Conclusions

In this chapter, we provided a tractable supervised learning algorithm, based on conic sections representing each class, to arrive at appropriate class descriptors given a labeled dataset. For each focus and directrix descriptor, we represented its feasible space in which all descriptor values result in identical labeling of classified points, as a compact geometric object. This rich concept class subsumes linear discriminants. We demonstrated the versatility of the resultant classifier by testing it against several state-of-the-art classifiers on many public domain datasets. Our classifier was able to classify tougher datasets better than others in most cases as validated in Table 4-1.

CHAPTER 5 LEARNING WITH LARGE MARGIN PURSUIT

In this chapter, we present a computational technique to determine the distance of a data point to the discriminant boundary due to the Conic Section Classifier (CSC) [68]. This can then be used to compute classifier margin, given a labeled dataset. The margin computation is included in the learning algorithm so as to pursue large margin CSC and thereby improve upon the generalization capacity of the classifier. First, we briefly describe the incremental Conic Section learning algorithm presented in [49], for the two-class classification problem in the following section.

5.1 Learning Algorithm Overview

Given a set of N labeled samples $\mathcal{Z} = \{\langle X_1, y_1 \rangle, \dots, \langle X_N, y_N \rangle\}$, where $X_i \in \mathbb{R}^M$ and labels $y_i \in \{-1, +1\}$, the objective is to learn the conic section descriptors, C_1, C_2 , that simultaneously minimize the misclassification error and seek a simpler discriminant boundary. The data is assumed to be sparse in a very high dimensional input space, i.e., $N \ll M$.

The process of learning the noted descriptors has two principal stages, played out in the input space \mathbb{R}^M and the *ecc-Space*, respectively. The relationship between these two spaces is given by the map ξ , which depends on the conic section descriptors for each class. In the first stage, given C_1 and C_2 , each X_i is mapped into *ecc-Space* as $\langle \varepsilon_1(X_i), \varepsilon_2(X_i) \rangle$. These values are sufficient to compute a pair of class eccentricities $\langle e_1, e_2 \rangle$ that improves the empirical learning accuracy (Eq. 2–3). For each misclassified sample, one can then find a desired pair of attributed eccentricities, denoted as $\langle \varepsilon'_{1i}, \varepsilon'_{2i} \rangle$, that would correctly classify it.

In the second stage, the *focus* and the *directrix* descriptors are alternately modified via a geometric algorithm to learn the map ξ . The map ξ is constrained to achieve the desired attributed eccentricities for the misclassified samples, *without affecting the eccentricities of those samples that are already classified correctly*. The process is repeated

until the descriptors converge or there can be no further improvement in classification. To exercise a tight control on learning, the descriptors are updated to learn only one misclassified point at a time. Each misclassified point results in a descriptor update. It is here, that we can improve the generalization capacity of the classifier by picking the update that gives the largest margin.

The core components of the learning technique are listed in Figure 4-1. The learning rate is non-decreasing since the algorithm does not misclassify a previously correctly classified point. A notable feature of the technique is its ability to track the entire feasible set for a descriptor, labeled as *Null Space*, that would map the data points to fixed attributed eccentricities, using simple geometric constructs.

5.2 The Margin Computation

The generalizability of a classifier can be improved by pursuing larger margins, as discussed in Section 2.1. We define the margin as the minimum over the distances of all the correctly classified data points, to the discriminant boundary. In order to compute the margin, we first have to find the shortest distance of a point, say $P \in \mathbb{R}^M$, to the discriminant boundary \mathcal{G} (Eq. 2-3). When formulated as an optimization problem (Eq. 5-1) with the non-linear constraint (Eq. 2-3), the distance computation is NP-hard [58] in general.

$$\text{dist}(P, \mathcal{G}) = \min \|P - X\|^2, \quad \text{subject to } g(X) = 0 \quad (5-1)$$

Although the objective function is quadratic in X , the equality constraint is highly non-linear. The numerical solution to Eq. 5-1 therefore works out to be very expensive. Note that we have to compute the shortest distance from the boundary to all the data points to determine the margin. For each competing descriptor update, we compute its margin so as to pick an update that yields the largest margin. This task becomes especially difficult due to two reasons. First, the discriminant boundary is a pair of

polynomial surfaces of degree at most eight. Second, the data lie in very high dimensional space. We now introduce a novel geometric approach to compute the margin.

5.2.1 Overview

The margin computation problem can be posed as finding the smallest hypersphere $\mathcal{S}(P, r)$, centered at P with radius r , that intersects with the discriminant boundary \mathcal{G} . Assuming we can evaluate the existence/lack of $\mathcal{G} \cap \mathcal{S}(P, r)$, the smallest radius is computed by performing a binary search on $r \in (0, r_p)$. The initial radius r_p is obtained by finding a point Z_0 lying on the discriminant boundary \mathcal{G} . Following is a method to find such a point. From the given labeled data set, pick a pair of data points, one from each side of the boundary, (see Eq. 2–3). Existence of such a point pair is guaranteed by our initial discriminant boundary. A binary search on the line joining this pair of data points gives us a point $Z_0 \in \mathcal{G}$ and hence an upper bound, say r_p , on the shortest distance of a given point P to \mathcal{G} .

Consider now sections of the boundary \mathcal{G} for which the distances to either both the directrix planes or both the focal points are held fixed. As a consequence of certain geometric observations, we shall demonstrate that the existence/lack of intersection of the hypersphere \mathcal{S} with any such section of the discriminant boundary can be determined analytically. The shortest distance of a point to these sections of the discriminant boundary can be computed in $O(NM)$ time. We propose an iterative algorithm that alternately updates the shortest distances to these sections so as to find a point on the discriminant boundary, nearest to P . An overview of the technique is presented in Figure 5-3. Next, we present algorithms to evaluate $\mathcal{G} \cap \mathcal{S}(P, r)$.

5.2.2 Spanning the Discriminant Boundary \mathcal{G}

The discriminant boundary (Eq. 2–3) can also be written as the implicit function:

$$((r_1 - e_1 h_1)h_2) \pm ((r_2 - e_2 h_2)h_1) = 0, \quad (5-2)$$

$$r_k(X) = \|X - F_k\|, \quad \forall k \in 1, 2 \quad (5-3)$$

$$h_k(X) = X^T Q_k + b_k, \quad \forall k \in 1, 2 \quad (5-4)$$

where r_k , and h_k are distances to the focus and directrix descriptors respectively. In order to evaluate the intersection, $\mathcal{G} \cap \mathcal{S}(P, r)$, we need to ascertain if $g(X)$ changes sign on the hypersphere, \mathcal{S} . The distances r_k , and h_k are bounded now, since $X \in \mathcal{S}(P, r)$. As a first pass, we can search for such a change in the sign of $g(X)$ by evaluating $g(X)$ at discretized distances in a bounded interval, for each valid combination of $\{h_1, h_2, r_1, r_2\}$.

For any point $X \in \mathcal{S}$, the distance $h_k(X)$ is bounded to be within $[h_k(P) - r_p, h_k(P) + r_p]$ due to Eq. 5–4. Similarly, r_k (Eq. 5–3) is bounded to be within $[|r_k(P) - r_p|, r_k(P) + r_p]$. If we discretize each parameter at $O(N)$ locations, the cost of evaluating an intersection is $O(N^4 M)$, which is expensive. In subsequent sections, we introduce a faster $O(M)$ algorithm to compute the intersection of the hypersphere \mathcal{S} with particular sections of the boundary \mathcal{G} .

5.2.3 Nearest Point on \mathcal{G} with Fixed Directrix Distances

Here, we first determine if the hypersphere \mathcal{S} intersects with the section of the boundary for which the distances to the *directrices* are fixed. Consider a subspace \mathcal{H} in which all the points are at some fixed distances (h_1, h_2) from the two directrices $\{b_k, Q_k\}$ for $k=1,2$. Such a space \mathcal{H} turns out to be a linear subspace of co-dimension 2 in \mathbb{R}^M , *i.e.* the intersection of two hyperplanes. With these constraints, the boundary \mathcal{G} (Eq. 2–3) in \mathcal{H} can also be written as:

$$\|X - F_2\| = m\|X - F_1\| + c \quad (5-5)$$

$$\text{where, } m = \pm \frac{h_2}{h_1}, \quad c = h_2(e_2 \mp e_1),$$

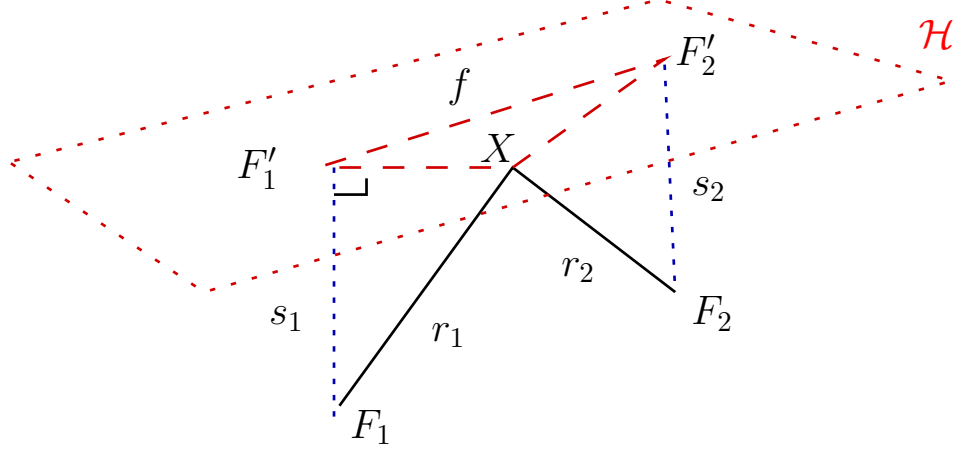


Figure 5-1. $\{X, F'_1, F'_2\} \in \mathcal{H}$, the linear subspace in which distances to directrices are constant (see Eq. 5-6).

Since $\mathcal{H} \equiv \mathbb{R}^{M-2}$, let us track the section of the discriminant boundary, $\mathcal{G} \cap \mathcal{H}$. Let $X \in \mathcal{H}$ and the distance between a focus point F_k and its orthogonal projection F'_k in \mathcal{H} , be $s_k = \|F_k - F'_k\|$. Given (h_1, h_2) , the section of the discriminant boundary, $\mathcal{G} \cap \mathcal{H}$ then becomes

$$\sqrt{\|X - F'_2\|^2 + s_2^2} = m\sqrt{\|X - F'_1\|^2 + s_1^2} + c \quad (5-6)$$

Figure 5-1 illustrates the points $\{X, F'_1, F'_2\} \in \mathcal{H}$, related to Eq. 5-6 for $\mathcal{G} \cap \mathcal{H}$. Any point $P \in \mathbb{R}^M$ can be orthogonally projected into \mathcal{H} with the equations below. The coefficients (u, v) in Eq. 5-7 are obtained by solving the constraints for \mathcal{H} given in Eq. 5-8.

$$P' = P - (uQ_1 + vQ_2), \quad P' \in \mathcal{H} \quad (5-7)$$

$$\text{with, } Q_k^T P' + b_k = h_k, \quad \forall k \in 1, 2 \quad (5-8)$$

Thus, $\mathcal{G} \cap \mathcal{H}$ is fully specified by elements lying in \mathcal{H} . We are now interested in finding out if the hypersphere $\mathcal{S}(P, r)$ intersects with $\mathcal{G} \cap \mathcal{H}$. Further, $\mathcal{S} \cap \mathcal{H}$ can be represented as a hypersphere $\mathcal{S}' \in \mathcal{F}$ centered at the projected data point $P' \in \mathcal{H}$ with radius r' , derived from Eq. 5-7. *Owing to symmetry, the intersection between the section of the boundary, $\mathcal{G} \cap \mathcal{H}$, and the hypersphere \mathcal{S}' needs to be checked only in the plane comprising $\{P', F'_1, F'_2\}$.*

We can deduce from Eq. 5–6 that the discriminant boundary in \mathcal{H} is radially symmetric about the line joining F'_1 and F'_2 . Let α be the length of the component of X along $(F'_2 - F'_1)$ and β be the length of its orthogonal component in the plane defined by $\{P', F'_1, F'_2\}$, as illustrated in Figure 5-2. Let $f = \|F'_2 - F'_1\|$. After translating the projected origin, $O' \in \mathcal{H}$ to F'_1 , the boundary becomes a quartic polynomial in the two variables (α, β) , as in Eq. 5–9. Also, the equation of the hypersphere $\mathcal{S}'(P', r')$ reduces to the circle, Eq. 5–10. Here, (α_p, β_p) are the components of P' along and across the line joining F'_2 and F'_1 .

$$\sqrt{(\alpha - f)^2 + \beta^2 + s_2^2} = m\sqrt{\alpha^2 + \beta^2 + s_1^2} + c \quad (5-9)$$

$$(\alpha - \alpha_p)^2 + (\beta - \beta_p)^2 = (r')^2 \quad (5-10)$$

Upon intersecting the two geometric objects due to Eqs. 5–9, 5–10, we obtain a quartic equation in α after eliminating β . For any quartic polynomial in one variable, we can check for the existence of real roots [69] and compute them explicitly [70], if necessary. Thus, we determine the intersection, $\mathcal{S} \cap (\mathcal{G} \cap \mathcal{H})$, in $O(M)$ time.

Assume that we begin with a hypersphere \mathcal{S} having an initial radius r_o that is guaranteed to intersect with the section of the discriminant boundary in \mathcal{H} , i.e $\mathcal{G} \cap \mathcal{H}$. All that remains to be done is to conduct a binary search on the radius of \mathcal{S} in the interval $(0, r_o]$ to find the shortest distance between P and the discriminant surface in \mathcal{H} . Moreover, we can explicitly determine the nearest point on the section $\mathcal{G} \cap \mathcal{H}$, say Z , from the polynomial roots (α, β) , and the points $\{P', F'_1, F'_2\}$.

5.2.4 Nearest Point on \mathcal{G} with Fixed Focal Distances

In this section, we determine if the hypersphere $\mathcal{S}(P, r)$ intersects with the part of the discriminant boundary in which distances to the *foci* are constant (Eq. 5–2). The locus of all the points that are at fixed distances $\{r_1, r_2, r\}$ from the points $\{F_1, F_2, P\}$ respectively,

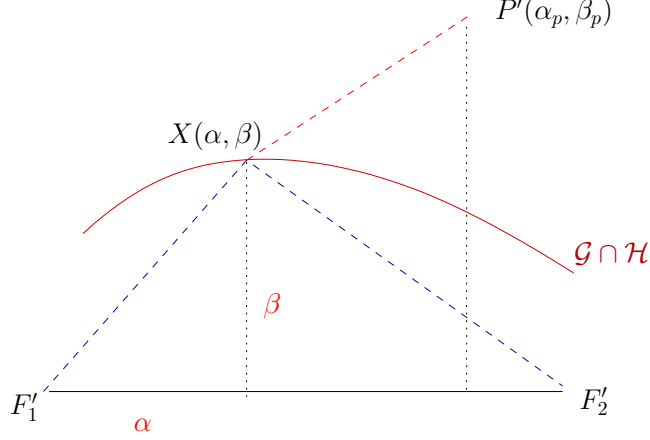


Figure 5-2. The discriminant boundary in the subspace \mathcal{H} becomes a quartic in (α, β) (see Eq. 5-9).

can be constructed by computing the intersection:

$$\mathcal{S}(P, r) \cap \mathcal{S}(F_1, r_1) \cap \mathcal{S}(F_2, r_2) \equiv \mathcal{S}'(C, r') \in \mathcal{F} \quad (5-11)$$

where \mathcal{S}' is a hypersphere in the linear subspace \mathcal{F} of co-dimension 2 in \mathbb{R}^M . This can be understood from an analogue in \mathbb{R}^3 : *The intersection of two spheres in \mathbb{R}^3 is a circle lying within the plane of intersection.* An $O(K^2M)$ algorithm was presented in [49] to compute the intersection of K hyperspheres. We compute the intersection of three spheres here, hence $K = 3$. The problem is now reduced to determining the intersection $\mathcal{G} \cap \mathcal{S}'$. After translating the origin to $C \in \mathcal{F}$, any point X in \mathbb{R}^M can be projected into \mathcal{F} as:

$$X' = X - (X^T U_1) U_1 - (X^T U_2) U_2 \quad (5-12)$$

where, $\{U_1, U_2\}$ are two orthonormal vectors perpendicular to \mathcal{F} . The section of the discriminant boundary \mathcal{G} in \mathcal{F} for a given pair of fixed focal distances (r_1, r_2) , now becomes:

$$\left(\frac{r_1}{h_1(X')} - e_1 \right) \pm \left(\frac{r_2}{h_2(X')} - e_2 \right) = 0 \quad (5-13)$$

which upon re-arrangement of terms results in a quadratic implicit surface equation in X' . Eq. 5-13 represents $\mathcal{G} \cap \mathcal{F}$ lying in \mathcal{F} . The intersection of the directrix hyperplane, denoted

as $(Q_k^T X' + b_k = 0)$, with \mathcal{F} can be equivalently represented as $X^T Q'_k + b_k = 0$. Here, $X \in \mathcal{F}$ and Q'_k is the component of unit normal Q_k in \mathcal{F} , obtained from Eq. 5–12. Now $\mathcal{G} \cap \mathcal{F}$ is fully specified by elements lying in \mathcal{F} .

We make a crucial geometric observation here. *Within \mathcal{F} , the discriminant function (Eq. 5–13) is invariant to translations normal to the plane spanned by $\{Q'_1, Q'_2\}$.* We now exploit the fact that $\mathcal{G} \cap \mathcal{F}$ is a function only of the distances to the directrix planes in the linear subspace \mathcal{F} . Since $\mathcal{S}'(C, r')$ is a hypersphere, *it is sufficient to investigate the intersection of interest, i.e. $\mathcal{S}' \cap (\mathcal{G} \cap \mathcal{F})$, in the plane spanned by $\{Q'_1, Q'_2\}$ and passing through C , the center of the hypersphere \mathcal{S}' .* Any point X in such a plane can be expressed as :

$$X = C + \alpha Q'_1 + \beta Q'_2 \tag{5–14}$$

The section of the boundary $\mathcal{G} \cap \mathcal{F}$ (Eq. 5–13), in this plane, reduces to a quadratic curve in parameters (α, β) . The hypersphere $\mathcal{S}'(C, r')$ in this plane becomes a (quadratic) circle, $\|X(\alpha, \beta) - C\| = r'$, in parameters (α, β) . Again, the intersection of these two geometric objects, obtained by eliminating β , yields a quartic polynomial in α . We can analytically find if real roots exist for a given quartic [69] and compute them exactly [70].

We described an $O(M)$ algorithm to find if the intersection, $\mathcal{S} \cap (\mathcal{G} \cap \mathcal{F})$, exists. The shortest distance of a point P to the section of the boundary \mathcal{G} , in which the focal distances are constant, is computed via a binary search on the radius of the initial hypersphere $\mathcal{S}(P, r)$ within the interval $(0, r_0]$. At the end of the binary search, we also compute the nearest point on the section, say $Z'(\alpha, \beta)$, from Eq. 5–14.

5.2.5 Large Margin Pursuit

As summarized in the algorithm listed in Figure 5–3, we alternately find the shortest distance to sections of the discriminant boundary, \mathcal{G} , with distances to either the foci or directrices fixed. To begin, the fixed distances, (h_1, h_2) , to the directrices are obtained from the initial point Z_0 on \mathcal{G} . We then compute the point Z , that is nearest to P in the

section $\mathcal{G} \cap \mathcal{H}$. The fixed distances, (r_1, r_2) , to the foci are determined by Z for finding the nearest point, Z' in that section of the boundary. In this fashion, the nearest point in each section is used to define the next section.

Data: Data points set \mathcal{Z} , Conic Descriptors C_1, C_2
Result: Margin between point set \mathcal{Z} and boundary \mathcal{G}

- 1: Find a point Z_0 on \mathcal{G}
- for** each Point $P \in \mathcal{Z}$ **do**
 - 2: Initial Distance $m_i = \|P - Z_0\|$, $Z' = Z_0$
 - repeat**
 - 3: Find closest $Z \in \mathcal{G}$ for fixed $\{h_1(Z'), h_2(Z')\}$
 - 4: Find closest $Z' \in \mathcal{G}$ for fixed $\{r_1(Z), r_2(Z)\}$
 - until** Z converges
 - 5: Point Distance $m_i = \|P - Z\|$
- end**
- 6: Margin = $\min\{m_i\}$

Figure 5-3: Margin Computation Algorithm

The alternating process is repeated until either the distance to the boundary converges or an $O(N)$ iteration limit is reached. The number of steps in all the binary searches is limited to $O(N)$. The complexity of computing the shortest distance of a point to the boundary \mathcal{G} in this manner, is $O(N^2M)$. The margin for a set of at most N points and a given conic configuration $\{C_1, C_2\}$ is computed in $O(N^3M)$ time. Similar to the numerical optimization technique (Eq. 5-1), the margin computation could be prone to local minima. However, we observed that the computation times are several orders of magnitude smaller than those for techniques involving either the numerical optimization (Eq. 5-1) or the discretization approach (Section 5.2.2). In the learning phase, among a set of competing updates for a conic section descriptor, we pick the one resulting in the largest margin. We also avoid an update if it reduces the current margin without improving the learning accuracy. We update the descriptors in a constrained manner so that only the correctly classified points do not move in the *ecc-Space*. This approach ensures that the *Null Space* for each descriptor update is larger.

Table 5-1. Given a pair of CSC descriptor sets, the accuracy of picking the one with larger margin is compared for varying boundary types. The last two columns list errors in margin computation.

CSC Boundary Types	Selection Accuracy %	Margin Error	
		μ	σ
Linear only	100.00	-.000870	.001413
Linear + non-linear	96.40	.000011	.004877
Non-linear	78.39	.017060	.022756
Highly non-linear	73.96	.038890	.034711

5.3 Experiments

In this section, we report two sets of experimental results. First, we use the proposed method to compute margins for a dataset and evaluate the accuracy of selecting large margin discriminants. Second, the classification performance of the enhanced classifier on several datasets was compared with various linear and non-linear classifiers.

5.3.1 Evaluating Margin Computation

The margin computation is employed in the learning algorithm to choose a CSC descriptor update that yields larger margin. In this section, we evaluate the accuracy of such a selection and compare margins obtained using our method with the true ones. We considered Colon Tumor data [4] that has 62 samples with 2000 features each, and projected it into \mathbb{R}^5 using a whitening transform so that its covariance matrix is identity. To compute the true margins (Section 5.2.2), we performed brute-force search for change in sign of $g(X)$ on $\mathcal{S}(P, r)$ so as to determine an intersection.

We considered discriminant boundaries of successively higher complexity. In the initial configuration for the results in Table 5-1, the directrices are coincident, the line joining foci is parallel to the directrix normal, say Q , and the class eccentricities are both zeros. This ensures that the discriminant boundary is always linear [49]. Upon making the eccentricities unequal, the boundary turns into a pair of linear and non-linear surfaces. Once the line joining the foci is not parallel to Q , the boundaries become non-linear. In the last case, the directrices are not coincident, resulting in highly non-linear boundaries.

Table 5-2. Details of data used in experiments

Dataset	Features	Samples	Class 1	Class 2
Epilepsy	216	44	19	25
Isolet-BC	617	100	50	50
CNS	7129	60	21	39
Colon Tumor	2000	62	40	22
Leukemia	7129	72	47	25
HWdigits35	649	400	200	200

For each boundary type, 20 CSC descriptors were randomly generated and the errors in margin computation are listed in Table 5-1. It can be seen that our approximations are reliable for simpler boundaries. Given each possible pair of competing descriptors, we verify if our method picked the configuration with larger margins. We observed that for near-linear and simpler boundaries, we selected the desired update in more than 95% of the instances. Since the discriminant turns out to be a collection of non-linear surfaces in general, our method is prone to local minima.

5.3.2 Classification Results

We compared the classification accuracies of CSC with the large margin pursuit (CSC-M) to CSC [49], as well as other state-of-the-art classifiers such as linear and kernel SVMs [29], and Fisher [35],[62] discriminants. We used polynomial kernels for the kernel based classifiers. The degree of the polynomial was empirically determined. Unless otherwise noted, all the classification experiments were performed using a leave-one-out cross validation protocol. The discriminant for CSC-M was initialized with that generated by either the linear SVM [29] or LFD [35]. In the learning phase, the competing conic-section descriptor updates were allowed to increase the margin without losing training classification accuracy. The CSC introduced in [49], pursued simpler boundaries in the learning phase to enhance generalizability. The characteristics of datasets used in our experiments are listed in Table 5-2. The classification results for six high dimensional sparse datasets pertaining to applications in computer vision and medical diagnosis, are listed in Table 5-3.

Table 5-3. Classification accuracies for the Conic Section Classifier with large margin pursuit (CSC-M), CSC, (Linear & Kernel) Fisher Discriminants and SVMs with polynomial (PLY) kernels. The degree of polynomial kernel is in parentheses.

Dataset	CSC-M	CSC	LFD	Lin-SVM	KFD PLY	SVM PLY
Epilepsy	93.18	88.64	77.27	56.18	86.36 (1)	86.36 (6)
Isolet-BC	92.00	84.00	81.00	91.00	91.00 (2)	91.00 (1)
CNS	70.00	73.33	51.67	68.33	65.00 (3)	68.33 (1)
Colon Tumor	87.10	87.10	85.48	80.56	75.81 (1)	82.26 (2)
Leukemia	97.22	98.61	97.22	97.22	98.61 (1)	97.22 (1)
HWdigits35	96.00	96.25	85.25	95.75	97.75 (3)	95.75 (1)

The Epilepsy dataset [71] consists of 3D Histograms of displacement vector fields representing the non-rigid registration between the left and right hippocampi in 3D. The task is to distinguish between Left and Right Anterior Temporal Lobe (RATL) epileptics. The dataset includes features for 19 LATL and 25 RATL epileptic subjects. The feature vector length, which depended on the number of 3D bins used, was empirically determined. All the classifiers performed well with a $6 \times 6 \times 6$ binning of the displacement field, except KFD for which we used a $16 \times 16 \times 16$ binning. As listed in Table 5-3, CSC with margin pursuit (CSC-M) achieved an impressive 93% testing accuracy, clearly outperforming all the other classifiers.

The Isolet-BC dataset is a part of the Isolet Spoken Letter Recognition Database [72]. [45], We considered 50 speech samples each, for alphabets B and C spoken by 25 speakers. Each subject spoke the alphabets B and C twice, resulting in 50 samples for each alphabet. The task here is to discriminate between the spoken alphabets B and C. The results using 10-fold cross-validation testing are reported in Table 5-3. CSC-M classifier outperformed all the other classifiers, with a large improvement over CSC without margin pursuit. The HWdigits35 data is a part of the Hand-written Digits dataset [72]. It has 200 samples of each digit, and around 649 features for each sample. We classified the digits *three* and *five*, as they turned out to be a difficult pair. We report 10-fold cross-validation experiments for these two datasets.

The next three datasets involve gene-expression features. The CNS dataset [73] contains treatment outcomes for central nervous system embryonal tumor on 60 patients, which includes 21 survivors and 39 failures. The expression scores for 7129 genes were obtained for each patient. The Colon Tumor data [4] consists of 2000 gene-expression levels for 22 normal and 40 tumor colon tissue features. Each gene-expression score is computed from a filtering process explained in [4]. In the Leukemia cancer class discovery [1], the task is to discriminate between 25 acute myeloid leukemia (AML) and 47 acute lymphoblastic leukemia (ALL) samples, given 7129 gene-expression probes for each subject.

Overall, the performance of CSC-M improved significantly over that without the margin pursuit, for the *Epilepsy* and *Isolet-BC* datasets. CSC-M almost matched CSC in the remaining four experiments in Table 5-3, as both the methods are susceptible to achieving local optima w.r.t. learning the optimal descriptors. The performance of linear-SVM is comparable to that of CSC-M for Isolet-BC, HWdigits35 and Leukemia datasets, since the large margin discriminant can turn out to be near-linear.

5.4 Summary

In this chapter, we proposed a novel geometric algorithm to compute margin to discriminant boundaries resulting from the Conic Section classifier. This technique was then used to maximize margin while learning conic section descriptors for each class, so as to improve upon the generalizability of the classifier. The classification performance of the classifier did improve on real datasets, due to enhancement of the learning method with a large margin pursuit.

CHAPTER 6 LEARNING WITH INEQUALITY CONSTRAINTS

In our preliminary work [49], we tracked a feasible space for each descriptor (focus or directrix) related to the constraint due to holding the values of the discriminant function at the classified points to be fixed. In this chapter, we construct a much larger feasible space for each descriptor related to the constraint that their labeling of correctly classified points remains unchanged.

We represent the feasible space, referred to as the *Null Space*, as a compact geometric object that is built incrementally in the learning phase, as described in Section 6.1.2. In Section 6.1.3.1, we introduce a stiffness criterion that captures the extent of non-linearity in the resultant discriminant boundary, given the conic section descriptors. It is used to pursue simpler discriminants by selecting appropriate solutions from the descriptor *Null Spaces*, and thereby improving upon the generalizability of the classifier. We demonstrate the efficacy of our technique in Section 6.2, by comparing it to well known classifiers like Linear and Kernel Fisher Discriminants and kernel SVM on several real datasets. Our classifier consistently performed better than LFD, as desired. In the majority of cases, it out-performed state-of-the-art classifiers. We discuss concluding remarks in Section 6.3. A list of symbols used to represent different geometric entities in this dissertation are given in the Appendix.

6.1 The Learning Algorithm

We introduce a novel incremental algorithm for the two-class Conic Section Classifier in this section. We assume a set of N labeled samples $P = \{\langle X_1, y_1 \rangle, \dots, \langle X_N, y_N \rangle\}$, where $X_i \in \mathbb{R}^M$ and the label $y_i \in \{-1, +1\}$, to be sparse in a very high dimensional input space such that $N \ll M$. Learning involves finding the conic section descriptors, $\{C_1, C_2\}$, that can minimize empirical learning risk and simultaneously result in simpler discriminant boundaries. The empirical risk, L_{err} , is defined as:

$$L_{err} = \frac{1}{N} \sum_i \mathbb{I}(y_i \cdot g(X_i) > 0) \quad (6-1)$$

where \mathbb{I} is the indicator function. A brief description of the algorithm is presented next.

In the learning phase, we perform a constrained update to one conic descriptor at a time, holding the other descriptors fixed; *the constraint being that the resultant boundary continues to correctly classify points that are already correctly classified in previous iteration.* The feasible space for each descriptor within which these constraints are satisfied will be referred to as its *Null Space*. We pick a solution from each *Null Space* in a principled manner such that one or more misclassified points are learnt. The sets of descriptors that will be updated alternately are $\{\langle e_1, e_2 \rangle, F_1, F_2, \{b_1, Q_1\}, \{b_2, Q_2\}\}$.

To begin with, we initialize the focus and directrix descriptors for each class such that the *Null Spaces* are large. The initialization phase is explained in detail in Section 6.1.4. The subsequent learning process is comprised of two principal stages. In the first stage, given fixed foci and directrices we compute attributed eccentricities (Eq. 2–1) for each point X_i , denoted as $\langle \varepsilon_{1i}, \varepsilon_{2i} \rangle$. We then compute an optimal pair of class eccentricities $\langle e_1, e_2 \rangle$, that minimizes the empirical risk L_{err} in $O(N^2)$ time, as described in Section 6.1.1.1. For a chosen descriptor (focus or directrix) to be updated, we find feasible intervals of desired attributed eccentricities, such that $y_i \cdot g(X_i) > 0 \forall X_i$, *i.e.* , the samples are correctly classified (Section 6.1.1.2).

In the second stage, we solve for the inverse problem. Given *class-eccentricities*, we seek a descriptor solution that causes attributed eccentricities to lie in the desired feasible intervals. This results in a set of geometric constraints on the descriptor, due to Eq. 2–4, that are dealt with in detail in Section 6.1.2. We compute the entire *Null Space*, defined as the equivalence class of the given descriptor that results in the same label assignment for the data. For each misclassified point, we pick a solution in this *Null Space*, that learns it with a large margin while ensuring a simpler decision boundary in \mathbb{R}^M . In Section 6.1.3.1, we introduce a *stiffness* criterion to quantify the extent of the non-linearity in a discriminant boundary. Among the candidate updates due to each misclassified point, an update is chosen that yields maximum *stiffness*; *i.e.* , minimal

Data: Labeled Samples P
Result: Conic Section Descriptors C_1, C_2

- 1: Initialize the class descriptors $\{F_k, \{b_k, Q_k\}\}, k \in \{1, 2\}$

repeat

- 2: quad Compute $\langle \varepsilon_1(X_i), \varepsilon_2(X_i) \rangle \quad \forall X_i \in P$
- 3: Find the best *class-eccentricities* $\langle e_1, e_2 \rangle$
- for** *each descriptor* in $\{F_1, F_2, \{b_1, Q_1\}, \{b_2, Q_2\}\}$ **do**
 - 4: Determine the classifying range for $\langle \varepsilon_{1i}, \varepsilon_{2i} \rangle$
 - 5: Find its feasible space due to these constraints.
 - for** *each misclassified point* X_{mc} **do**
 - 6: Compute a descriptor update to learn X_{mc}
- end**
- 7: Pick updates with least *empirical* error
- 8: Then pick an update with largest *Stiffness*

end

until *the descriptors* C_1, C_2 *converge*

Figure 6-1: Algorithm to Learn the Descriptors due to Inequality Constraints

non-linearity. The second stage is repeated to update the foci $\{F_1, F_2\}$ and the directrices $\{\{b_1, Q_1\}, \{b_2, Q_2\}\}$, one at a time. The two stages are alternately repeated until either the descriptors converge or there can be no further improvement in classification and *stiffness*. Note that all through the process, *the learning accuracy is non-decreasing since a previously classified point is never misclassified due to subsequent updates*. The learning algorithm is listed in Figure 6-1. We discuss the first phase in detail in the following section.

6.1.1 Learning in Eccentricity Space

The focus and directrix descriptors of both the classes induce a non-linear mapping, $\varepsilon^*(X)$, due to Eq. 2-1, from the input space into \mathbb{R}^2 , as:

$$\varepsilon^*(X) = \langle \varepsilon_1(X), \varepsilon_2(X) \rangle \tag{6-2}$$

This space of attributed eccentricities will be referred to as the eccentricity space (*ecc-Space*). We defined this space so as to determine the best pair of class eccentricities simultaneously. In Figure 6-2A, the x and y axes represent the maps $\varepsilon_1(X)$ and $\varepsilon_2(X)$ respectively, as defined in Eq. 2-1. For any given choice of class eccentricities, e_1 and e_2 ,

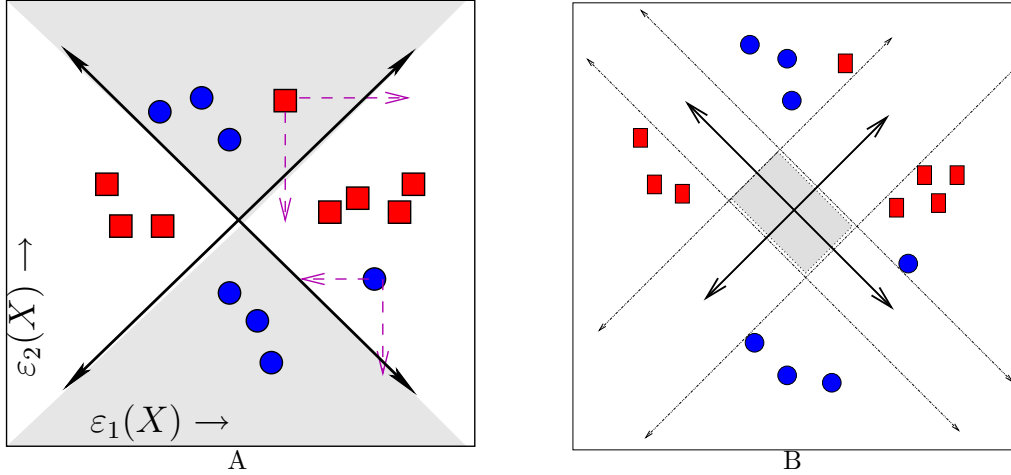


Figure 6-2. (A) Shaded regions in this *ecc-Space* belong to the class with label -1 . The pair of diagonal lines is the discriminant boundary, $|\varepsilon_1(X) - e_1| = |\varepsilon_2(X) - e_2|$. Learning involves shifting misclassified points into desired regions. (B) The discriminant boundary is the pair of thick lines. Within the shaded rectangle, any choice of *class eccentricity* descriptors (the point of intersection of two thick lines) results in identical classification.

the discriminant boundary equivalent in *ecc-Space*, $|\varepsilon_1(X) - e_1| - |\varepsilon_2(X) - e_2| = 0$, becomes a pair of mutually orthogonal lines with slopes $+1, -1$, respectively, as illustrated in the figure. These lines intersect at $\langle e_1, e_2 \rangle$, which is a point in *ecc-Space*. The lines divide *ecc-Space* into four quadrants with opposite pairs belonging to the same class. It should be noted that this discriminant corresponds to an equivalent non-linear decision boundary in the input space \mathbb{R}^M . We use *ecc-Space* only as a means to explain the learning process in the input space. The crucial part of the algorithm is to learn the eccentricity maps, Eq. 2-1 for each class by updating the foci and directrices. In this section, we first present an $O(N^2)$ time algorithm to find the optimal class eccentricities. Next, we determine resultant constraints on the focus and directrix descriptors due to the classified points.

6.1.1.1 Finding Optimal Class-Eccentricities $\langle e_1, e_2 \rangle$

We now present an $O(N^2)$ algorithm to find the optimal pair of class-eccentricities resulting in the least possible empirical error (Eq. 6-1), given fixed foci and directrices. The discriminant boundary (Eq. 2-3) in *ecc-Space* is completely defined by the location of

class eccentricities. Consider a pair of orthogonal lines with slopes $+1$ and -1 respectively, passing through each mapped sample in *ecc-Space*, as illustrated in Figure 6-2B.

Consequently, these pairs of lines partition the *ecc-Space* into $(N + 1)^2$ 2D rectangular intervals. We now make the critical observation that *within the confines of such a 2D interval, any choice of a point that represents class eccentricities results in identical label assignments* (see Figure 6-2B). Therefore, the search is limited to just these $(N + 1)^2$ intervals. The interval that gives the smallest classification error is chosen. The cross-hair is set at the center of the chosen 2D interval to obtain large functional margin. Whenever, there are multiple 2D intervals resulting in the least empirical error, the larger interval is chosen so as to obtain a larger margin.

6.1.1.2 Geometric Constraints on Foci and Directrices

Having fixed the class eccentricities, the learning constraint on the attributed eccentricities for each X_i , with a functional margin $\delta > 0$, is given by Eq. 6-3. Assuming that the descriptors of class 2 are fixed, the constraint on $\varepsilon_1(X_i)$ due to Eq. 6-3 is derived in Eq. 6-4. Now, if we are interested in updating only focus F_1 , the constraints on F_1 in terms of distances to points, X_i , are given by Eq. 6-5.

$$y_i (|\varepsilon_1(X_i) - e_1| - |\varepsilon_2(X_i) - e_2|) > \delta \quad (6-3)$$

$$\Rightarrow y_i (|\varepsilon_1(X_i) - e_1|) > (\delta + y_i \Delta \varepsilon_{2i}) = w_{1i}$$

$$l_{1i} = (y_i e_1 - w_{1i}) > y_i \varepsilon_1(X_i) > (y_i e_1 + w_{1i}) = u_{1i} \quad (6-4)$$

$$l_{1i} > y_i \frac{\|F_1 - X_i\|}{h_{1i}} > u_{1i} \quad (6-5)$$

Here $\Delta \varepsilon_{2i} = |\varepsilon_2(X_i) - e_2|$, and h_{1i} is the distance to the directrix hyperplane of class 1 (Eq. 2-4). In Eq. 6-5, the only unknown variable is F_1 . Similarly, we can obtain constraints for all the other descriptors. Whenever the intervals due to the constraints are unbounded, we apply bounds derived from the range of attributed eccentricities in the previous iteration. The margin, δ , was set to 1% of this range.

In the second stage of the learning algorithm, we employ a novel geometric technique to construct the *Null Space* of say, F_1 , for distance constraints (Eq. 6–5) related to the currently classified points. Next, we pick a solution from the *Null Space* that can learn a misclassified point by satisfying its constraint on F_1 , if such a solution exists. The learning task now reduces to updating the foci and directrices of both the classes alternately, so that the misclassified points are mapped into their desired quadrants in *ecc-Space*, while the correctly classified points remain in their respective quadrants. Note that with such updates, *our learning rate is non-decreasing*. In the next section, we construct *Null Spaces* for the focus and directrix descriptors. In Section 6.1.3 we deal with learning misclassified points.

6.1.2 Constructing *Null Spaces*

Assume that we have N_c classified points and a point X_{mc} that is misclassified. We attempt to modify the discriminant boundary by updating one descriptor at a time in \mathbb{R}^M such that the point X_{mc} is correctly classified. The restriction on the update is that all the classified points remain in their respective class quadrants in *ecc-Space*, *i.e.*, their labels not change. In this section, we construct feasible spaces for each of the focus and directrix descriptors within which the labeling constraint is satisfied.

6.1.2.1 The Focus *Null Space*

Here, we consider updating F_1 , the focus of class 1. For ease of readability, we drop the reference to class from here on, unless necessary. First, we construct the *Null Space* within which F adheres to the restrictions imposed by the following N_c quadratic constraints, due to Eq. 6–5,:

$$r_{li} < \|F - X_i\| < r_{ui} \quad \forall i \in 1, 2, \dots, N_c \quad (6-6)$$

where r_{li} , and r_{ui} are lower and upper bounds on the distance of X_i to F . In effect, each point X_i requires F to be at a certain distance from itself, lying in the interval (r_{li}, r_{ui}) . Next, we need to pick a solution in the *Null Space* that satisfies a similar

constraint on X_{mc} so as to learn it, and improve the generalization capacity of the classifier (Section 6.1.3). While this would otherwise have been an NP-hard [58] problem (like a general QP problem), the geometric structure of these quadratic constraints enables us to construct the *Null Space* in just $O(N^2M)$ time. Note that by assumption, the number of constraints $N \ll M$.

The *Null Space* of F with respect to each constraint in Eq. 6–6 is the space between two concentric hyperspheres in \mathbb{R}^M , referred to as a *shell*. Hence, the *Null Space* for all the constraints put together is the intersection of all the corresponding shells in \mathbb{R}^M . This turns out to be a complicated object. However, we can exploit the fact that the focus in the previous iteration, denoted as F° , satisfies all the constraints in Eq. 6–6 since it resulted in N_c classified points. To that end, we first construct the locus of all focus points, F' , that satisfy the following equality constraints:

$$\|X_i - F'\| = \|X_i - F^\circ\| = r_i, \quad \forall i \in 1 \dots N_c \quad (6-7)$$

Note that such an F' will have the same distances to the classified data points X_i as the previous focus F° , so that the values of the discriminant function at X_i remain unchanged, *i.e.*, $g(X_i, F') \equiv g(X_i, F^\circ)$. Later, we will use the locus of all F' to construct a subspace of the *Null Space* related to Eq. 6–6 that also has a much simpler geometry.

6.1.2.2 Intersection of Hyperspheres

The algorithm we discuss here incrementally builds the *Null Space* for the equality constraints in Eq. 6–7; *i.e.*, the locus of all foci F' that are at distance r_i to the respective classified point X_i . The *Null Space* is initialized as the set of feasible solutions for the first equality constraint in Eq. 6–7. It can be parameterized as the hypersphere $S_1 = (r_1, X_1)$, centered at X_1 with radius r_1 . Next, the second equality constraint is introduced, the *Null Space* for which, considered independently, is the hypersphere $S_2 = (r_2, X_2)$. Then the combined *Null Space* for the two constraints is the intersection of the two hyperspheres, $S_1 \cap S_2$.

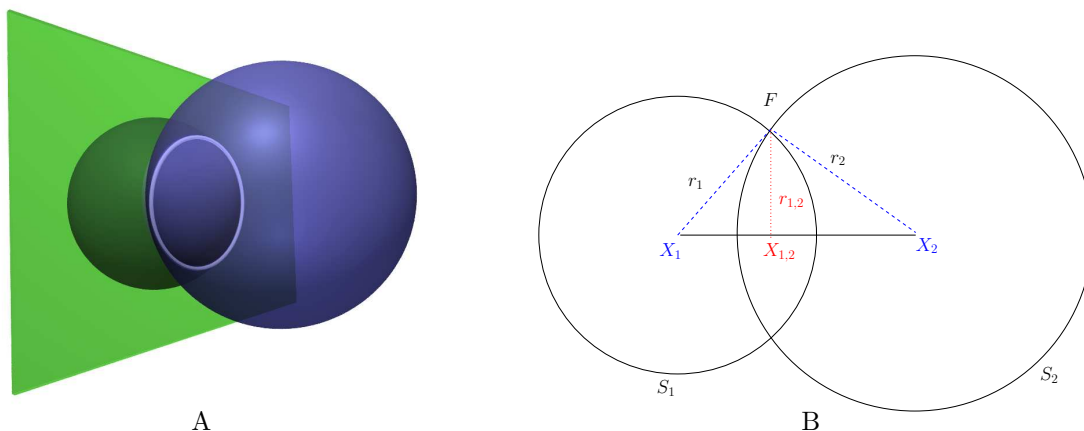


Figure 6-3. (A) Intersection of two hypersphere *Null Spaces*, $S_1(r_1, X_1)$ and $S_2(r_2, X_2)$ lying in a hyperplane. Any point on the new *Null Space* (bright-circle) satisfies both the hypersphere (distance) constraints. (B) $S_1 \cap S_2$ can be parameterized by the hypersphere $S_{\{1,2\}}$ centered at $X_{1,2}$ with radius $R_{1,2}$.

As illustrated in Figure 6-3A, the intersection of two spheres in \mathbb{R}^3 is a circle that lies on the plane of intersection of the two spheres. The following solution is based on the analogue of this fact in \mathbb{R}^M . We make two critical observations: *the intersection of two hyperspheres is a hypersphere of one lower dimension, and this hypersphere lies on the intersecting hyperplane of the original hyperspheres*. Each iteration of the algorithm involves two steps. In the first step, we re-parameterize the combined *Null Space*, $S_1 \cap S_2$, as a hypersphere $S_{\{1,2\}}$ of one lower dimension lying in the hyperplane of intersection $H_{\{1,2\}}$. Based on the geometry of the problem and the parameterization of S_1 and S_2 , shown in Figure 6-3B, we can compute the radius and the center of the new hypersphere $S_{\{1,2\}} = (r_{\{1,2\}}, X_{\{1,2\}})$ in $O(M)$ time, given by Eqs. 6-8, 6-9, 6-10. We can also determine the intersecting hyperplane $H_{\{1,2\}}$ represented as $\{b_{\{1,2\}}, Q_{12}\}$. The first descriptor $b_{\{1,2\}}$ is the displacement of $H_{\{1,2\}}$ from the origin and the other descriptor is the unit vector normal to $H_{\{1,2\}}$. In fact, $Q_{\{1,2\}}$ lies along the line joining X_1 and X_2 . The parameters of

the new hypersphere $S_{\{1,2\}}$ and the hyperplane $H_{\{1,2\}}$ are computed as :

$$Q_{1,2} = (X_2 - X_1) / \|X_2 - X_1\| \quad (6-8)$$

$$X_{1,2} = X_1 + Q_{1,2} Q_{1,2}^T (F^\circ - X_1) \quad (6-9)$$

$$r_{1,2} = \|X_{1,2} - F^\circ\|$$

$$b_{1,2} = -Q_{1,2}^T X_{1,2} \quad (6-10)$$

In the second step, the problem for the remaining equality constraints is reposed on the hyperplane $H_{\{1,2\}}$. This is accomplished by intersecting each of the remaining hyperspheres S_3, \dots, S_{N_c} that correspond to the samples X_3, \dots, X_{N_c} , with $H_{\{1,2\}}$, in $O(NM)$ time. Once again, based on the geometry of the problem, the new centers of the corresponding hyperspheres can be computed using Eq. 6-9 and their radii are given by:

$$r'_i = \sqrt{r_i^2 - ((X_i - F^\circ)^T Q_{1,2})^2}$$

In short, the intersection of the N_c hyperspheres problem is converted into the intersection of $(N_c - 1)$ hyperspheres in the hyperplane $H_{\{1,2\}}$, as summarized below:

$$\begin{aligned} S_1 \cap S_2 &\rightarrow S_{\{1,2\}} \in H_{\{1,2\}} \\ S_i \cap H_{\{1,2\}} &\rightarrow S'_i \in H_{\{1,2\}} \quad \forall i = 3, \dots, N_c \\ S_1 \cap S_2 \dots \cap S_{N_c} &\rightarrow S_{1,2} \cap S'_{3\dots} \cap S'_{N_c} \in H_{1,2} \end{aligned} \quad (6-11)$$

The problem is now transparently posed in the lower dimensional hyperplane $H_{\{1,2\}}$ as a problem equivalent to the one that we began with, except with one less hypersphere constraint. The end result of repeating this process $(N_c - 1)$ times yields a *Null Space* that satisfies all the equality constraints (Eq. 6-7), represented as a single hypersphere lying in a low dimensional linear subspace and computed in $O(N^2M)$ time. It should be observed that all the intersections thus far are feasible and that the successive *Null Spaces* have non-zero radii since the equality constraints have a feasible solution *a priori*, i.e. , F° .

Upon unfolding Eqs. 6–8, 6–9 over iterations, we notice that the computation of the normal to the hyperplane of each intersection and that of the new center can be equivalently posed as a Gram-Schmidt orthogonalization [56]. The process is equivalent to the QR decomposition of the following matrix:

$$\mathbf{A} = \begin{bmatrix} \tilde{X}_2 & \tilde{X}_2 & \dots & \tilde{X}_{N_c} \end{bmatrix} = \mathbf{Q}\mathbf{R} \quad (6-12)$$

where, $\tilde{X}_i = (X_i - X_1)$

$$C_e = \pi(X_1) = X_1 + \mathbf{Q}\mathbf{Q}^T(F^\circ - X_1) \quad (6-13)$$

$$r_e = \|F^\circ - C_f\|$$

The function $\pi(X)$ in Eq. 6–13 projects any point into a low dimensional linear subspace, \mathcal{H} , normal to the unit vectors in \mathbf{Q} (Eq. 6–12) and contains F° . \mathcal{H} can be defined as the linear subspace $\{X \in \mathbb{R}^M : \mathbf{Q}^T(X - F^\circ) = 0\}$. We use existing stable and efficient QR factorization routines to compute \mathcal{S}^e in $O(N^2M)$ time. *It is noteworthy that the final Null Space due to the equality constraints in Eq. 6–7 can be represented as a single hypersphere $\mathcal{S}^e = (r_e, C_e) \subset \mathcal{H}$.* The center and radius of \mathcal{S}^e can be computed using Eq. 6–13. The geometry of \mathcal{S}^e enables us to generate sample focus points that always satisfy the equality constraints. In the next section, we pursue larger regions within which the inequality constraints on the focus due to Eq. 6–6 are satisfied.

6.1.2.3 Intersection of Shells

Consider the intersection of two *shells* centered at X_1, X_2 , in \mathbb{R}^3 , as illustrated in Figure 6-4. We first compute the largest disc centered at the previous focus, F° , which is guaranteed to lie within the intersection. Next, we revolve the disc about the line joining X_1 and X_2 . The resultant object is a doughnut like toroidal region, that can be defined as a product of a circle and a disc in \mathbb{R}^3 , as illustrated in Figure 6-4A. The circle traced by F° is the locus of all foci F that are at the same distances to points X_1, X_2 , as F° . We pursue this idea in \mathbb{R}^M . The final *Null Space*, say \mathcal{S}^f , that satisfies the inequality

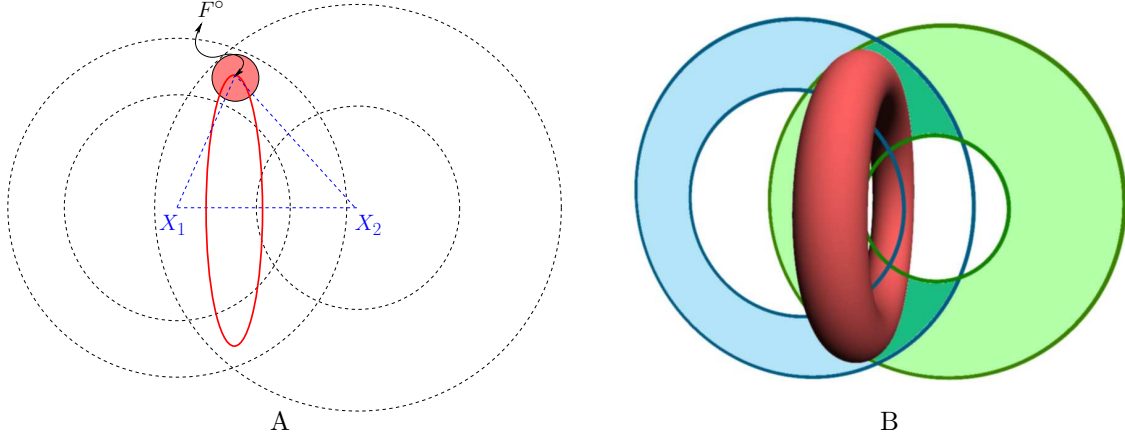


Figure 6-4. Cross-section of *shell-shell* intersection in \mathbb{R}^3 . The red disc in (A) is the largest disc centered at F° that is within the intersection. The thick circle passing through F° and orthogonal to the cross-section plane, is the locus of all foci with same distance constraints as F° . A product of the disc and the thick circle results in a toroid, as in (B), which is a tractable subregion lying within the intersection of *shells*.

constraints on F (Eq. 6-6) can be defined as :

$$\mathcal{S}^f \equiv \{F = F' + \beta U : F' \in \mathcal{S}^e, \|U\| = 1, \beta \in [0, \alpha)\} \quad (6-14)$$

where F' is a point from the feasible space, \mathcal{S}^e , due to the equality constraints on F (Eq. 6-7), and $\alpha > 0$ is the radius of the largest solid hypersphere (ball) at all F' that satisfies the inequality constraints. In this manner, we intend to add a certain α thickness to the locus of F' .

We can compute the radius, α , of the largest disc at F° from Eqs. 6-6, 6-14. Let $r_i = \|X_i - F^\circ\|$, $F = F^\circ + \beta U$ be a point on the ball at F° . Due to triangle inequality between F , F° and any point X_i , $\forall \beta \in [0, \alpha)$ we have:

$$\begin{aligned} |r_i - \beta| &\leq \|(X_i - F^\circ) - \beta U\| \leq r_i + \beta \\ \Rightarrow r_{li} &< |r_i - \beta| \leq \|X_i - F\| \leq r_i + \beta < r_{ui} \\ \Rightarrow \beta &< r_i - r_{li}, \beta < r_{ui} - r_i \\ \Rightarrow \alpha &= \min \{(r_{ui} - r_i), (r_i - r_{li})\}_{i=1 \dots N_c} \end{aligned} \quad (6-15)$$

Here $\beta < r_i$ so as to satisfy Eq. 6-6. We can thus compute α from Eq. 6-15 in just $O(N)$ time given the distance bounds r_{li} , and r_{ui} . With very little computational expense, we can track a larger *Null Space* than that due to the equality constraints (Eq. 6-7).

6.1.2.4 The Directrix Null Space

Here, we present a two-step technique to construct the *Null Space* for a directrix that satisfies the learning constraints in Eq. 6-4. First, we list constraints on the directrix so that the classified points remain in their quadrants when mapped into *ecc-Space*. Second, we reduce the problem of constructing the resultant feasible space into that of a focus *Null Space* computation.

The constraints on the hyperplane descriptor set $\{b, Q\}$, due to those on attributed eccentricities in Eq. 6-4, $\forall i = 1 \dots N_c$, are :

$$h_{li} < (b + Q^T X_i) < h_{ui} \quad \text{with } Q^T Q = 1 \quad (6-16)$$

$$\Rightarrow h_{li} - h_{u1} < Q^T (X_i - X_1) < h_{ui} - h_{l1}$$

$$\Rightarrow \tilde{h}_{li} < Q^T \tilde{X}_i < \tilde{h}_{ui} \quad (6-17)$$

where h_{li} , and h_{ui} are lower and upper bounds on the distance of X_i to the hyperplane $\{b, Q\}$. We assume every other descriptor is fixed except the unknown $\{b, Q\}$. Upon subtracting the constraint on X_1 from the other constraints, we obtain Eq. 6-17, where $\tilde{X}_i = (X_i - X_1)$, $\tilde{h}_{li} = (h_{li} - h_{u1})$, and $\tilde{h}_{ui} = (h_{ui} - h_{l1})$. We can now convert the distances to hyperplane constraints to those of distances to a point, by considering $\|Q - \tilde{X}_i\|$ as in Eq. 6-18. Let $z_i = (1 + \|\tilde{X}_i\|^2)$. The resultant *shell* constraints on Q , $\forall i = 1 \dots N_c$ are in Eq. 6-19

$$\|Q - \tilde{X}_i\|^2 = 1 + \|\tilde{X}_i\|^2 - 2Q^T \tilde{X}_i \quad (6-18)$$

$$\Rightarrow (z_i - 2\tilde{h}_{ui}) < \|Q - \tilde{X}_i\|^2 < (z_i - 2\tilde{h}_{li}) \quad (6-19)$$

Thus given N_c inequality constraints on point distances to Q and given the previous Q° , we use the solution from the focus update problem to construct a *Null Space* for Q . The

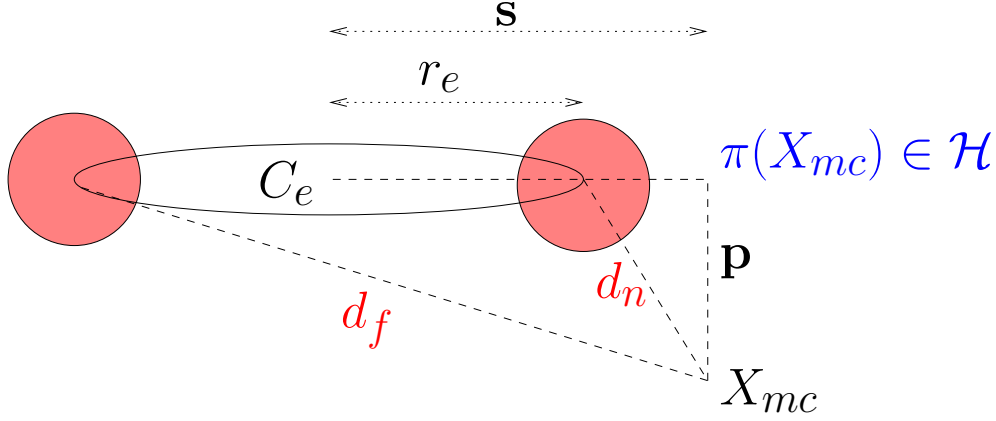


Figure 6-5. Cross-section of the (donut like) toroidal *Null Space*, \mathcal{S}^f , lying in the plane spanned by $\{X_{mc}, \pi(X_{mc}), C_e\}$. The expressions for distances d_n and d_f in terms of p, s , and r_e are given in Eq. 6-20

only unknown left is b which lies in an interval due to X_1 in Eq. 6-16, given Q . We choose b to be the center of that interval, as $b = (h_{u1} + h_{l1})/2 - Q^T X_1$, so that the margin for X_1 is larger in *ecc-Space*.

6.1.3 Learning Misclassified Points

In order to keep the learning process tractable, we learn one misclassified point, X_{mc} , at a time. The final *Null Space*, \mathcal{S}^f , for the inequality constraints on F can be defined as all $F = F' + \beta U$, where $F' \in \mathcal{S}^e \subset \mathcal{H}$, $\beta \in [0, \alpha)$, and U is any normal vector. To determine if \mathcal{S}^f has a solution that can learn X_{mc} , we intersect \mathcal{S}^f with a *shell* corresponding to the inequality constraint (Eq. 6-6) of X_{mc} , say $r_l < \|F - X_{mc}\| < r_u$. This is equivalent to checking if the range of $\|F - X_{mc}\|$, $\forall F \in \mathcal{S}^f$ intersects with the interval (r_l, r_u) . This range can be determined by computing the distance to the nearest and farthest points on \mathcal{S}^e to X_{mc} , as in Eq. 6-20. First, we project X_{mc} into the linear subspace \mathcal{H} , in which \mathcal{S}^e lies, using Eq. 6-13 to obtain $\pi(X_{mc})$. Let $p = \|X_{mc} - \pi(X_{mc})\|$ and $s = \|C_e - \pi(X_{mc})\|$.

We then have :

$$\begin{aligned} d_n &= \sqrt{(s - r_e)^2 + p^2} \\ d_f &= \sqrt{(s + r_e)^2 + p^2} \end{aligned} \quad (6-20)$$

$$d_n - \alpha < \|F - X_{mc}\| < d_f + \alpha \quad (6-21)$$

where (r_e, C_e) are the radius and center of \mathcal{S}^e . See Figure 6-5 to interpret these relations. Now the intersection of \mathcal{S}^f and the *shell* corresponding to X_{mc} is reduced to that of the intervals: $(d_n - \alpha, d_f + \alpha) \cap (r_l, r_u)$. If they don't intersect, we can easily pick a (focus) point on \mathcal{S}^f that is either nearest or farthest to the *shell* related to X_{mc} , so as to maximally learn X_{mc} , *i.e.* , change $r(X_{mc})$ which in turn maximizes $y_{mc} \cdot g(X_{mc})$. If they do intersect, there are only a few cases of intersections possible due to the geometry of \mathcal{S}^f and the *shell*. Each such intersection turns out to be or contains another familiar object like \mathcal{S}^f , a *shell*, or a solid hypersphere (ball). Whenever the intersection exists, we pick a solution in it that maximizes $\sum y_i g(X_i)$ for the other misclassified points, *i.e.* , a solution closer to satisfying all the inequality constraints put together. In this manner, we compute a new update for each misclassified point. Next, we describe a criterion to pick an update, from the set of candidate updates, that can yield simplest possible discriminant.

6.1.3.1 Stiffness Measure

We introduce a *stiffness* measure that quantifies the extent of the non-linearity of the conic section classifier. It is defined as:

$$\Gamma(C_1, C_2) = |Q_1^T Q_2| + |Q_1^T F_{1,2}| + |Q_2^T F_{1,2}| \quad (6-22)$$

where, $F_{1,2}$ is the unit normal along the line joining the foci. The maximum of this measure corresponds to the configuration of conic sections that can yield a linear discriminant as discussed in Section 2.3. The measure is defined for focus and directrix descriptors. It can be used to pick an update yielding the largest *stiffness* so that the resultant discriminant can be most generalizable. We noticed that the non-linearity of

the discriminant boundary is primarily dependent upon the angle between the directrix normals, Q_1 and Q_2 . In Figure 2-3, the stiffness measures for different types of boundaries are listed. The discriminant boundary becomes simpler when the directrix normals are made identical. Moreover, the functional dependencies between the distance constraints which determine the discriminant, in Eq. 2-4, become simpler if the line joining the foci is parallel to the directrix normals. It stems from the observation that every point in the hyperplane that perpendicularly bisects the line joining the foci, is at equal distance from the foci ($r_1(X) \equiv r_2(X)$). Linear discriminants can be guaranteed when the stiffness is maximum and the class eccentricities are either equal in magnitude or near zero. Two configurations that yielded linear discriminants are illustrated in Figures 2-3C, 2-3D. From the final *Null Space* of the descriptor being updated, we determine a descriptor update favored by each misclassified point. Among the competing updates, we choose the update with the least *empirical* error and maximum *stiffness*. Thus, the large *stiffness* pursuit incorporates a quasi model selection into the learning algorithm.

6.1.4 Initialization

Our learning algorithm is equivalent to solving a highly non-linear optimization problem that requires the final solution to perform well on both seen and as yet unseen data. Naturally, the solution depends considerably on the choice of initialization. As expected, random initializations converged to different conic descriptors leading to inconsistent performance. We observed that owing to the eccentricity function (Eq. 2-1), the *Null Spaces* become small (at times, vanishingly small) if the focus or directrix descriptors are placed very close to the samples. We found the following data-driven initializations to be consistently effective in our experiments. The initialization of all descriptors were done so as to start with a linear discriminant obtained from either linear SVM or Fisher [35] classifier or with the hyperplane that bisects the line joining the class means. The class eccentricities were set to $\langle 0, 0 \rangle$. The foci were initialized to lie far from their respective class samples. We initialized normals $Q_1 = Q_2$ with that in the

initial linear discriminant. Now, b_1, b_2 were initialized to be either equal or far from their respective class clusters. If the data was not well separated with the current initialization, the foci and directrices were pushed apart until they were outside the smallest sphere containing the samples.

6.1.5 Discussion

One of the core characteristics of our algorithm is that after each update any point that is correctly classified by the earlier descriptors is not subsequently misclassified. This is due to two reasons. First, we begin with an initialization that gives a valid set of assignments for the class attributed eccentricities. This implies that the *Null Space* for the classified points is non-empty to begin with. Second, any descriptor update chosen from the *Null Space* is a feasible solution that satisfies the distance constraints introduced by the correctly classified points. Moreover, the current descriptor to be updated is also a feasible solution, *i.e.* , in the worst case the *Null Space* collapses to the current descriptor. *Therefore, the learning rate of CSC is non-decreasing.* The radius of the final *Null Space* for each descriptor, monotonically decreases as more points are correctly classified, as can be deduced from Eq. 6–13. Also, the order of classified samples processed does not affect the final *Null Space* .

A key contribution of our learning technique is the tracking of a large set of feasible solutions as a compact geometric object. From this *Null Space* we pick a set of solutions to learn each misclassified points. From this set, we pick a solution biased towards a simpler discriminant using a stiffness criterion so as to improve upon generalization. The size of the margin, δ , in *ecc-Space* also gives a modicum of control over generalization.

6.2 Experiments

We evaluated the classifier on several real datasets concerning cancer class discovery, epilepsy diagnosis, and recognition of objects and spoken alphabets. We begin with a brief overview of the datasets, and the classifiers used for comparison with Conic Section

Table 6-1. Details of data used in experiments

Dataset	Features	Samples	Class 1	Class 2
Epilepsy	216	44	19	25
Colon Tumor	2000	62	40	22
Leukemia	7129	72	47	25
CNS	7129	60	21	39
ETH-Objects	16384	20	10	10
TechTC38	1683	143	74	69
Isolet-BC	617	100	50	50

classifier (CSC). Next, we discuss implementation details of CSC followed by a review of the results.

6.2.1 Datasets

We conducted experiments on a variety of datasets involving medical diagnosis, object recognition, text-categorization and spoken letter recognition. All the datasets considered have the peculiar nature of being high dimensional and sparse. This is due to the fact that either the collection of samples is prohibitive or that the features obtained for each sample can be too many, especially when there is no clear way to derive a reduced set of meaningful compound features.

The dimensionality and feature size details of the datasets used are listed in Table 6-1. Epilepsy data [3] consists of the shape deformation between the left and right hippocampi for 44 epilepsy patients. First, we computed the displacement vector field in 3D representing the non-rigid registration that captures the asymmetry between the left and right hippocampi. We found the joint 3D histogram of the (x, y, z) components of the displacement vector field to be a better feature set for classification. We used a $6 \times 6 \times 6$ binning of the histograms in 3D. The task was to categorize the localization of the focus of epilepsy to either the left (LATL) or right temporal lobe (RATL).

The Colon Tumor [4], the Leukemia [1], and the CNS [73] datasets are all gene array expression datasets. The gene-expression scores computation is explained in their respective references. In the Colon Tumor data the task is to discriminate between normal and tumor tissues. The leukemia data consists of features from two different cancer

classes, namely acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL). The CNS dataset contains treatment outcomes for central nervous system embryonal tumor on 60 patients, which includes 21 survivors and 39 failures.

From the ETH-80 [46] object dataset, we chose 10 profile views of a dog and a horse. The views are binary images of size 128×128 . From the TechTC-38 [74] text-categorization dataset, we classify the documents related to Alabama vs. Michigan localities (id: 38 [74]), given word frequencies as their features. We dropped features (words) that are either very infrequent or too common. The Isolet-BC dataset is a part of the Isolet Spoken Letter Recognition Database [45, 72], which consists of features extracted from speech samples of B and C from 25 speakers.

6.2.2 Classifiers and Methods

We implemented a faster version of the Linear Fisher Discriminant (LFD) [35] as described in Yu & Yang [61]. This technique exploits the fact that high-dimensional data has singular scatter matrices and discards the sub-space that carries no discriminative information. Support Vector Machines (SVM) [36] and Kernel Fisher Discriminants (KFD) [62] broadly represented the non-linear category. Both employ the kernel trick of replacing inner products with Mercer kernels. Among linear classifiers, we chose LFD and linear SVM. We used *libSVM* [75], a C++ implementation of SVM using Sequential Minimal Optimization [37] and our own MATLAB implementation of KFD. Polynomial (PLY) and Radial Basis (RBF) Kernels were considered for SVM and KFD.

We performed *stratified* 10-fold cross validation (CV) [28] in which the samples from each class are randomly split into 10 partitions. Partitions from each class are put into a fold, so that the label distribution of training and testing data is similar. In each run, one fold is withheld for testing while the rest is used for training and the process is repeated 10 times. The average test-error is reported as the generalization error estimate of the classifier. The experimental results are listed in Table 6-2. The best parameters for each classifier were empirically explored using grid search. We searched for the best

degree for polynomial kernels from 1 . . . 5, beyond which the classifiers tend to learn noise. We computed the largest distance between a sample pair as the base-size for selecting the radius of the RBF kernels. The radius was searched in an exponentially spaced grid between 10^{-3} to 10^5 times the base-size.

6.2.3 Conic Section Classifier

We implemented CSC in MATLAB. The computationally intensive aspect in the learning is tracking the *Null Space* for the equality constraints. Since this has been reduced to *QR* decomposition of a matrix composed from the sample vectors, we used the MATLAB `qr` routine which is numerically more stable and faster than having to compute the hypersphere intersections separately. Whenever possible, we cached the *QR* decomposition results to avoid recomputations. We have an $O(N^2)$ algorithm to compute the optimal class-eccentricities, that will be used in the first iteration. This gave us the best learning accuracy for the initial configuration of CSC. This could result in a non-linear boundary to start with. Then the stiffness guided choice of updates ensured that we pursue simpler discriminants without decreasing the learning rate. In subsequent iterations, we searched for better eccentricity descriptors in the immediate 10×10 neighborhood intervals of the previous class eccentricities, in *ecc-Space*. This avoids jumping from the current global optima to the next, after learning the foci and directrix descriptors, thereby making the algorithm more stable. We found that the descriptors converged to a local minima typically within 75 iterations of the learning technique, listed in Figure 6-1. The parameters of CSC involve the choice of initialization, described in Section 6.1.4, and the functional margin δ . The δ value is fixed to be at a certain percentage of the extent of attributed eccentricities, $\varepsilon^*(X)$. We searched for the best percentage among [1%, .1%, .01%].

6.2.4 Classification Results

Classification results are listed in Table 6-2. Conic Section Classifier (CSC) performed significantly better than the other classifiers for the Epilepsy data. In the gene-expression

Table 6-2. Classification results for CSC, (Linear & Kernel) Fisher Discriminants and SVM.

Dataset	CSC	LFD	KFD-PLY	KFD-RBF	SVM-PLY	SVM-RBF
Epilepsy	91.50	78.00	80.50	85.00	84.50	84.50
Colon Tumor	88.33	85.24	78.57	83.57	88.33	83.81
Leukemia	95.71	92.86	98.57	97.32	97.14	95.71
CNS	71.67	50.00	63.33	75.00	65.00	65.00
ETH-Objects	90.00	85.00	90.00	55.00	85.00	90.00
TechTC38	74.14	67.05	71.24	71.90	72.10	60.24
Isolet-BC	95.00	91.00	94.00	94.00	94.00	94.00

datasets, CSC was comparable to others with Leukemia and Colon Tumor data, but not with the CNS data which turned out to be a tougher classification task. CSC fared slightly better than the other classifiers in text-categorization and spoken letter recognition data. In fact, CSC has consistently performed substantially better than the LFD, as it is desirable by design. It is also interesting to note that none of the SVM classifiers actually beat CSC. This empirically proves that CSC is able to represent more generalizable boundaries than SVM for all the data considered.

The parameters used in the experiments are listed in Table 6-3. The column related to Q , lists that the normals to directrices were initialized from those due to either linear SVM, or LFD, or the line joining the means of the samples from the same class. The column related to δ denotes the margin as percentage of the range of class attributed eccentricities. We report the average *stiffness* of CSC over 10-folds in each experiment. Note that this is not a parameter of CSC. We included this in Table 6-3 for comparison with the degree of the polynomial kernels. The stiffness values for all experiments were near its maximum (3.0). The standard deviation of the stiffness was less than 0.02 for all the data, except for CNS (.08), and Isolet-BC (.1) datasets. Hence, *stiffness* does guide descriptor updates towards yielding simpler discriminants for the same or higher learning accuracy. The best degree for other polynomial kernel classifiers also turned out to be 1, in several cases. Except for the Leukemia data, CSC performed better than linear SVM and linear Fisher classifiers in all cases.

Table 6-3. Parameters for results reported in Table 6-2

Dataset	Q	CSC		KFDPLY	KFDRBF	SVMPLY	SVMRBF
		$\delta\%$	Stiffness*	degree	radius	degree	radius
Epilepsy	Means	1	2.9997	4	3	5	.006
Colon Tumor	LFD	.01	2.9955	1	5	2	.5
Leukemia	LFD	1	2.9793	1	70	1	3
CNS	Means	.1	2.9454	5	.07	1	.001
ETH-Objects	LFD	.01	2.9884	3	300	1	2
TechTC	SVM	1	2.9057	1	500	1	300
Isolet-BC	SVM	1	3.0	4	.3	1	.07

*Not a parameter.

We performed experiments on a quad core 3.8 GHz 64-bit Linux machine. Each 10-fold CV experiment on gene-expression datasets took under 4 minutes for all the classifiers including CSC. For the other datasets, run times were less than 2 minutes. Since we used a faster variant of LFD [61], it took under a second for all the datasets. The run times for CSC were better than the other classifiers with RBF kernels. However, the search for the optimal model parameters adds a factor of 5 to 10. From the experimental results in Table 6-2, it is evident that Conic Section classifier out-performed or matched the others in a majority of the datasets.

6.3 Summary

In this chapter, we provided a tractable supervised learning algorithm in which the set of feasible solutions, called the *Null Space*, for a descriptor update is represented as a compact geometric object. The computation of the *Null Space* related to quadratic equality constraints on class descriptors is reduced to that of a Gram-Schmidt [56] orthogonalization. We introduced a *stiffness* criterion that quantifies the extent of the non-linearity in the discriminant boundary. In each descriptor *Null Space*, we pick solutions that learn misclassified points and yield simpler discriminant boundaries, due to the *stiffness* criterion. Thus *stiffness* enables the classifier to perform model selection in the learning phase. As the learning problem is equivalent to a non-linear optimization problem that is not convex, our method is prone to local minima as well.

We tested the resultant classifier against several state-of-the-art classifiers on many public domain datasets. Our classifier was able to classify tougher datasets better than others in most cases, as validated in Table 6-2. The classifier in its present form uses axially symmetric conic sections. The concept class, by definition applies to the multi-class case as well. The learning algorithm needs to incorporate equivalent boundary representation in *ecc-Space*, among other aspects. In future work, we intend to extend this technique to multi-class classification, to conic sections that are not necessarily axially symmetric, and explore pursuit of conics in kernel spaces.

CHAPTER 7 CONCLUSION

We have introduced a novel concept class based on conic section descriptors, that can represent highly non-linear discriminant boundaries with merely $O(M)$ parameters. This rich concept class subsumes linear discriminants. We have provided tractable supervised learning algorithms to learn the class descriptors given a labeled dataset. We have represented the set of feasible solutions, called the *Null Space*, for a descriptor update in each learning algorithm as a compact geometric object with very few parameters. The computation of *Null Space* related to quadratic equality constraints on class descriptors is reduced to that of Gram-Schmidt orthogonalization [56].

We have introduced a *stiffness* criterion that quantifies the extent of non-linearity in the discriminant boundary. In each descriptor *Null Space*, we pick solutions guided by the *stiffness* criterion, that can learn misclassified points as well as yield simpler discriminant boundaries. Thus *stiffness* enables the classifier to perform an ad-hoc model selection in the learning phase. We have also presented a novel geometric algorithm to compute margin to the discriminant boundaries resulting from the Conic Section classifier. This technique was then used to maximize margin while learning conic section descriptors for each class, so as to improve upon the generalizability of the classifier.

We have demonstrated the versatility of the resultant classifier by testing it against several state-of-the-art classifiers on many public domain datasets. Our classifier was able to classify tougher datasets better than others in most cases. As the learning problem is equivalent to a non-linear optimization problem that is not convex, our method is also prone to local minima.

Future Work

The concept class, by definition applies to multi-class case as well. However, the learning algorithms need to be modified to incorporate equivalent boundary representation in *ecc-Space*, among other aspects. The classifier in its present form uses axially

symmetric conic sections. In future work, we intend to extend this technique to multi-class classification, to data in dimensions lower than the number of samples and to conic sections that are not necessarily axially symmetric. We also intend to explore pursuit of conics in kernel spaces.

APPENDIX
NOTATION

N, M	Number of samples and features.
X	A point in \mathbb{R}^M .
F	A focus point.
$\langle y_i, X_i \rangle$	A labeled data sample. $y_i \in \{-1, +1\}$
(b, Q)	Directrix hyperplane descriptors: offset and unit normal.
Hyperplane	Locus of all points $\{X \in \mathbb{R}^M : b + Q^T X = 0, \ Q\ = 1\}$
C_k	Descriptors of class k : $\{F_k, (b_k, Q_k), e_k\}$
e_k	The scalar valued class eccentricity.
$\varepsilon_k(X)$	The eccentricity function given C_k .
\mathcal{G}	$\{X \in \mathbb{R}^M : g(X) \equiv 0\}$ (decision boundary)
$r_k(X)$	Distance to focus : $\ X - F_k\ $
$h_k(X)$	Distance to directrix : $b_k + Q_k^T X$
$S_i(r_i, X_i)$	Hypersphere centered at X_i with radius r_i
$\mathcal{S}^e(r_e, C_e)$	$\cap_{i=1}^{N_c} S_i$ for N_c classified points.
\mathcal{H}	A linear subspace in which \mathcal{S}^e lies.
\mathcal{S}^f	Toroidal <i>Null Space</i> defined in Eq. 6–14.

REFERENCES

- [1] T. Golub, D. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. Mesirov, H. Coller, M. Loh, J. Downing, M. Caligiuri, C. Bloomfield, and E. Lander, “Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring,” *Science*, vol. 286, no. 5439, pp. 531–537, 1999.
- [2] W. Zhao, S. Corporation, A. Rosenfeld, and P. J. Phillips, “Face recognition: A literature survey,” *ACM Computing Surveys*, vol. 35, no. 4, pp. 399–458, 2003.
- [3] N. Vohra, B. C. Vemuri, A. Rangarajan, R. L. Gilmore, S. N. Roper, and C. M. Leonard, “Kernel fisher for shape based classification in epilepsy,” *MICCAI*, pp. 436–443, 2002.
- [4] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and A. Levine, “Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays,” in *Proc. Nat. Acad. Sc . USA*, vol. 96, 1999, pp. 6745–6750.
- [5] A. M. Martinez and M. Zhu, “Where are linear feature extraction methods applicable?” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 12, pp. 1934–1944, 2005.
- [6] Due, A. K. Jain, and T. Taxt, “Feature extraction methods for character recognition-a survey,” *Pattern Recognition*, vol. 29, no. 4, pp. 641–662, April 1996.
- [7] R. J. Quinlan, “Induction of decision trees,” *Machine Learning*, pp. 81–106, 1986.
- [8] R. Duda, P. Hart, and D. Stork, *Pattern Classification*. Wiley Interscience, 2001.
- [9] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.
- [10] G. Casella and R. Berger, *Statistical Inference*. Duxbury Resource Center, June 2001.
- [11] N. Kwak and C.-H. Choi, “Input feature selection by mutual information based on parzen window,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 12, pp. 1667–1671, 2002.
- [12] H. Peng, F. Long, and C. Ding, “Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy.” *IEEE Trans Pattern Anal Mach Intell*, vol. 27, no. 8, pp. 1226–1238, August 2005.
- [13] J. A. Sterne, G. D. Smith, and D. R. Cox, “Sifting the evidence—what’s wrong with significance tests? another comment on the role of statistical methods,” *BMJ*, vol. 322, no. 7280, pp. 226–231, January 2001.

- [14] J. Horra and M. Rodriguez-Bernal, “Posterior predictive p-values: what they are and what they are not,” *TEST: An Official Journal of the Spanish Society of Statistics and Operations Research*, vol. 10, no. 1, pp. 75–86, June 2001.
- [15] T. Hsing, E. Dougherty, P. Sebastiani, I. S. Kohane, and M. F. Ramoni, “Relation between permutation-test p values and classifier error estimates,” in *Machine Learning, Special Issue on Machine Learning in the Genomics*, vol. 52, 2003, p. 1130.
- [16] R. Kohavi and G. H. John, “Wrappers for feature subset selection,” *Artificial Intelligence*, vol. 97, no. 1-2, pp. 273–324, 1997.
- [17] K.-C. Lee, J. Ho, and D. J. Kriegman, “Acquiring linear subspaces for face recognition under variable lighting,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 5, pp. 684–698, 2005.
- [18] L. Torresani and K. chih Lee, “Large margin component analysis,” in *Advances in Neural Information Processing Systems 19*. Cambridge, MA: MIT Press, 2007, pp. 1385–1392.
- [19] S. Watanabe, *Pattern recognition: human and mechanical*. New York, NY, USA: John Wiley & Sons, Inc., 1985.
- [20] JMLR, *Special Issue on Variable and Feature Selection*, 2003, vol. 3.
- [21] R. Herbrich, *Learning Kernel Classifiers: Theory and Algorithms (Adaptive Computation and Machine Learning)*. The MIT Press, December 2001.
- [22] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning*. Springer, August 2001.
- [23] D. H. Wolpert, “The lack of a priori distinctions between learning algorithms,” *Neural Computation*, vol. 8, no. 7, pp. 1341–1390, October 1996.
- [24] V. N. Vapnik, *The Nature of Statistical Learning Theory (Information Science and Statistics)*. Springer, November 1999.
- [25] C. Burges, “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.
- [26] V. N. Vapnik, “Estimation of dependences based on empirical data (translated by s. kottz).” New York: Springer-Verlag, 1982.
- [27] D. Hush and C. Scovel, “On the vc dimension of bounded margin classifiers,” *Machine Learning*, vol. 45, no. 1, pp. 33–44, 2001.
- [28] R. Kohavi, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” in *International Joint Conference on Artificial Intelligence*, 1995, pp. 1137–1145.
- [29] V. Vapnik, *Statistical Learning Theory*. John Wiley and Sons, New York, 1999.

- [30] H. Zhu and R. Rohwer, “No free lunch for cross-validation,” *Neural Computation*, vol. 8, pp. 1421–1426, 1996.
- [31] C. Goutte, “Note on free lunches and cross-validation,” *Neural Computation*, vol. 9, no. 6, pp. 1245–1249, 1997.
- [32] T. Fawcett, “An introduction to roc analysis,” *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, June 2006.
- [33] D. M. Gavrilu, “A bayesian, exemplar-based approach to hierarchical shape matching,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 8, pp. 1408–1421, 2007.
- [34] Y. Zhou, L. Gu, and H.-J. Zhang, “Bayesian tangent shape model: Estimating shape and pose parameters via bayesian inference,” *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 1, p. 109, 2003.
- [35] R. Fisher, “The use of multiple measurements in taxonomic problems,” in *Annals of Eugenics (7)*, 1936, pp. 111–132.
- [36] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.
- [37] J. C. Platt, “Sequential minimal optimization: A fast algorithm for training support vector machines,” in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT Press, Cambridge, MA, USA, 1999, pp. 185–208.
- [38] C.-C. Chang and C.-J. Lin, “Libsvm: a library for support vector machines (version 2.31),” 2001.
- [39] J. Mouro-Miranda, K. J. Friston, and M. Brammer, “Dynamic discrimination analysis: A spatial-temporal svm,” *NeuroImage*, vol. 36, no. 1, pp. 88 – 99, 2007.
- [40] X. Wang, R. Hutchinson, and T. M. Mitchell, “Training fmri classifiers to discriminate cognitive states across multiple subjects,” in *In NIPS*, 2004.
- [41] O. Bousquet and B. Scholkopf, “Comment on ”support vector machines with applications”,” *Statistical Science*, vol. 21, p. 337, 2006.
- [42] T. M. Mitchell, *Machine Learning*. McGraw-Hill, 1997.
- [43] C. M. Bishop, *Neural Networks for Pattern Recognition*. New York, NY, USA: Oxford University Press, Inc., 1995.
- [44] S. Haykin, *Neural Networks: A Comprehensive Foundation (2nd Edition)*, 2nd ed. Prentice Hall, 1998.
- [45] R. Cole, Y. Muthusamy, and M. Fanty, “The ISOLET spoken letter database,” Oregon Graduate Institute, Tech. Rep. CSE 90-004, 1990.

- [46] B. Leibe and B. Schiele, “Analyzing appearance and contour based methods for object categorization,” in *Computer Vision and Pattern Recognition*, vol. 2, 2003, pp. 409–415.
- [47] A. Jain and D. Zongker, “Feature selection: Evaluation, application, and small sample performance,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 2, pp. 153–158, 1997.
- [48] P. Somol, P. Pudil, and J. Kittler, “Fast branch and bound algorithms for optimal feature selection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 7, pp. 900–912, 2004.
- [49] A. Banerjee, S. Kodipaka, and B. C. Vemuri, “A conic section classifier and its application to image datasets,” *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, vol. 1, pp. 103–108, 2006.
- [50] A. J. Smola and P. J. Bartlett, Eds., *Advances in Large Margin Classifiers*. Cambridge, MA, USA: MIT Press, 2000.
- [51] F. L. Bookstein, “Fitting conic sections to scattered data,” *Computer Graphics and Image Processing*, vol. 9, no. 1, pp. 56–71, 1979.
- [52] T. Pavlidis, “Curve fitting with conic splines,” *ACM Transactions on Graphics*, vol. 2, no. 1, pp. 1–31, 1983.
- [53] L. Quan, “Conic reconstruction and correspondence from two views,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 2, pp. 151–160, 1996.
- [54] G. Dorffner, “A unified framework for MLPs and RBNFs: Introducing conic section function networks,” *Cybernetics and Systems*, vol. 25, no. 4, 1994.
- [55] D. DeCoste, M. Burl, A. Hopkins, and N. Lewis, “Support vector machines and kernel fisher discriminants: A case study using electronic nose data,” *Fourth Workshop on Mining Scientific Datasets*, 2001.
- [56] L. N. Trefethen and D. Bau, *Numerical Linear Algebra*. SIAM: Society for Industrial and Applied Mathematics, 1997.
- [57] E. W. Weisstein, *CRC Concise Encyclopedia of Mathematics*, 2nd ed. Boca Raton, FL: CRC Press, 2003.
- [58] K. G. Murty and S. N. Kabadi, “Some np-complete problems in quadratic and nonlinear programming,” *Math. Program.*, vol. 39, no. 2, pp. 117–129, 1987.
- [59] D. Graham and N. Allinson, “Characterizing virtual eigen signatures for general purpose face recognition,” *Face Recognition: From Theory to Applications, NATO ASI Series F, Computer and Systems Sciences*, vol. 163, pp. 446–456, 1998.

- [60] K. J. Dana, B. van Ginneken, S. K. Nayar, and J. J. Koenderink, “Reflectance and texture of real-world surfaces,” *ACM Trans. on Graph.*, vol. 18, no. 1, pp. 1–34, 1999.
- [61] H. Yu and J. Yang, “A direct lda algorithm for high-dimensional data with application to face recognition,” *Pattern Recognition*, vol. 34, no. 12, pp. 2067–2070, 2001.
- [62] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller, “Fisher discriminant analysis with kernels,” *Neural Networks for Signal Processing IX*, pp. 41–48, 1999.
- [63] B. Schölkopf, C. Burges, and A. Smola, *Advances in Kernel Methods-Support Vector Learning*. MIT Press, Cambridge, 1999.
- [64] T. S. Furey, N. Duffy, N. Cristianini, D. Bednarski, M. Schummer, and D. Haussler, “Support vector machine classification and validation of cancer tissue samples using microarray expression data,” *Bioinformatics*, vol. 16, no. 10, pp. 906–914, 2000.
- [65] M. Varma and A. Zisserman, “Texture classification: Are filter banks necessary?” in *Computer Vision and Pattern Recognition*, vol. 2, 2003, pp. 691–698.
- [66] T. Leung and J. Malik, “Recognizing surfaces using three-dimensional textons,” in *Proceedings of the International Conference on Computer Vision (ICCV)*, vol. 2. Washington, DC, USA: IEEE Computer Society, 1999.
- [67] E. Spellman, B. C. Vemuri, and M. Rao, “Using the KL-center for efficient and accurate retrieval of distributions arising from texture images,” *Computer Vision and Pattern Recognition*, pp. 111–116, 2005.
- [68] S. Kodipaka, A. Banerjee, and B. C. Vemuri, “Large margin pursuit for a conic section classifier.” in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2008.
- [69] I. Z. Emiris and E. P. Tsigaridas, “Comparing real algebraic numbers of small degree.” in *European Symposia on Algorithms*, 2004, pp. 652–663.
- [70] S. S. Rao, *Mechanical vibrations*, 4th ed. Addison-Wesley, 2004.
- [71] S. Kodipaka, B. C. Vemuri, A. Rangarajan, C. M. Leonard, I. Schmallfuss, and S. Eisenschenk, “Kernel fisher discriminant for shape-based classification in epilepsy,” *Medical Image Analysis*, vol. 11, no. 2, pp. 79–90, 2007.
- [72] A. Asuncion and D. Newman, “UCI machine learning repository,” 2007.
- [73] S. L. Pomeroy *et al.*, “Prediction of central nervous system embryonal tumour outcome based on gene expression,” *Nature*, vol. 415, no. 6870, pp. 436–442, 2002.
- [74] E. Gabrilovich and S. Markovitch, “Text categorization with many redundant features: using aggressive feature selection to make svms competitive with c4.5,” in *ICML '04: Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004.

- [75] R.-E. Fan, P.-H. Chen, and C.-J. Lin, “Working set selection using second order information for training support vector machines,” *Journal of Machine Learning Research*, vol. 6, pp. 1889–1918, December 2005.

BIOGRAPHICAL SKETCH

Santhosh Kodipaka received Bachelor of Technology in computer science and engineering, with honors in visual information processing from the International Institute of Information Technology, Hyderabad, India in May 2003. He received his Master of Science in computer engineering from the University of Florida in December 2007. He graduated with a Ph.D in computer engineering in August 2009, advised by Prof.Vemuri and Dr.Banerjee. In his dissertation, he introduced a novel Conic Section Classifier along with tractable geometric learning algorithms. During his graduate study, he also worked on diagnosis of epilepsy in humans given 3D hippocampal shapes segmented from MR scans, and classification of longitudinal rat hippocampal scans for seizure incidence from 3D volume deformation tensor based morphometry. His research interests include Machine Learning, Medical Image Analysis, Computer Vision, Graphics and Visualization.