INVESTIGATION OF ORTHOGONAL POLYNOMIAL KERNELS AS SIMILARITY FUNCTIONS FOR PATTERN CLASSIFICATION BY SUPPORT VECTOR MACHINES

MOSTAFA EL-SAYED MOHAMED ABDEL-ALEEM

A thesis submitted in partial fulfilment of the requirement of Staffordshire University for the degree of Doctor of Philosophy

February 2018

To My Family . . .

Acknowledgement

It has been a bumpy ride and the longest and toughest journey I have ever had in my life.

I am deeply indebted to my principal supervisor Prof. Claude Chibelushi for his help, valuable advice, continuing support, and endless patience throughout the preparation of this work. No words can express my gratitude for his great efforts and regular follow-ups in closely checking and revising all the stages of this research as he was very keen on producing an outstanding piece of work.

I am also so grateful to my second supervisor Prof. Mansour Moniri for his precious inputs, tremendous assistance, and his consistent expert guidance throughout all the stages of this research work.

I would also like to thank Staffordshire University for providing all the physical and financial resources required for my research. I also owe special thanks to my colleagues for their support, inspiration, and encouragement.

I must never forget to express my endless appreciation and deep thankfulness to my beloved family, especially my parents, without whom I would have never been here in the first place. They have been always providing me with their constant love, understanding, encouragement, patience, and prayers. Their emotional and financial support made such a difference that even when I feel I am stuck in the middle of nowhere, I still have the conviction that I shall find my own way and that someone will be waiting for me on the other side. I hope they find in my current achievement a reward for their care and love.

Abstract

A kernel function is an important component in the support vector machine (SVM) kernel-based classifier. This is due to the elegant mathematical characteristics of a kernel, which amount to the mapping of non-linearly separable classes to an implicit higher-dimensional feature space where they can become linearly separable, and hence easier to classify. Such characteristics are those prescribed by the underpinning positive semi-definite (PSD) property. The properties of this feature space can, however, be difficult to interpret, to customize or select an appropriate kernel for the classification task at hand. Moreover, the high-dimensionality of the feature space does not usually provide apparent and intuitive information about the natural representations of the data in the input space, as the construction of this feature space is only implicit. On the other hand, SVM kernels have also been regarded as *similarity* functions in many contexts to measure the resemblance between two patterns, which can be from the same or different classes. However, despite the elegant theory of PSD kernels, and its remarkable implications on the performance of many learning algorithms, limited research efforts seem to have studied kernels from this *similarity* perspective. Given that patterns from the same class share more similar characteristics than those belonging to different classes, this similarity perspective can therefore provide more tangible means to craft or select appropriate kernels than the properties of the implicit high-dimensional feature spaces that one might not even be able to calculate.

This thesis therefore aims to: (i) investigate the similarity-based properties, which can be exploited to characterise kernels (with focus on the so-called "orthogonal polynomial kernels") when used as similarity functions, and (ii) assess the influence of these properties on the performance of the SVM classifier. An appropriate similarity-based model is therefore defined in the thesis based on how the *shape* of an SVM kernel should ideally look like when used to measure the similarity between its two inputs. The model proposes that the similarity curve should be maximized when the two kernel inputs are identical, and it should decay monotonically as they differ more and more from each other. Motivated by the pictorial characteristics of the Chebyshev kernels reported in the literature, the thesis adopts this kernel-shape perspective to also study some other orthogonal polynomial kernels (such as the Legendre kernels and Hermite kernels), to underpin the assessment of the proposed ideal shape of the similarity curve for kernel-based pattern classification by SVMs.

The analysis of these polynomial kernels revealed that they are naturally constructed from smaller kernel building blocks, which are combined by summation and multiplication operations. A novel similarity fusion framework is therefore developed in this thesis to investigate the effect of these fusion operations on the shape characteristics of the kernels and on their classification performance. This framework is developed in three stages, where Stage 1 kernels are those building blocks constructed from only the polynomial order n (the highest order under consideration), whereas Stage 2 kernels combine all the Stage 1 kernel blocks (from order 0 to n) using a summation fusion operation. The Stage 3 kernels finally combine Stage 2 kernels with another kernel via a multiplication fusion operation. The analysis of the shape characteristics of these three-stage polynomial kernels revealed that their inherent fusion operations are synergistic in nature, as they bring their shapes closer to the ideal similarity function model, and hence enable the calculation of more accurate similarity measures, and accordingly score better classification performance. Experimental results showed

that these summative and multiplicative fusion operations improved the classification accuracy by average factors of 17.35% and 19.16%, respectively, depending on the dataset and the polynomial function employed.

On the other hand, the shapes of the Stage 2 polynomial kernels have also been shown to oscillate after a certain threshold within the standard normalized input space of [-1,1]. A simple adaptive data normalization approach is therefore proposed to confine the data to the threshold window where these kernels exhibit the sought after ideal shape characteristics, hence eliminate the possibility of any data point to be located outside the range where these oscillations are observed. The implementation of the adaptive data normalization approach accordingly leads to a more accurate calculation of similarity measures and improves the classification performance. When compared to the standard normalized input space, experimental results (performed on the Stage 2 kernels) demonstrate the effectiveness of the proposed adaptive data normalization approach, with an average accuracy improvement factor of 11.772%, depending on the dataset and the polynomial function utilized.

Finally, a new perspective is also introduced whereby the utilization of orthogonal polynomials is perceived as a way of transforming the input space to another vector space, of the same dimensionality as the input space, prior to the kernel calculation step. Based on this perspective, a novel processing approach, based on vector concatenation, is proposed which, unlike the previous approaches, ensures that the quantities processed by each polynomial order are always formulated in vector form. This way, the attributes embedded in the structure of the original vectors are maintained intact. The proposed concatenated processing approach can also be used with any polynomial function, regardless of the parity combination of its monomials, whether they are only odd, only even, or a combination of both. Moreover, the Gaussian kernel is also proposed to be evaluated on vectors processed by the polynomial kernels (instead of the linear kernel used in the previous approaches), due to the more accurate similarity shape characteristics of the Gaussian kernel, as well as its renowned ability to implicitly map the input space to a feature space of higher dimensionality. Experimental results demonstrate the superiority of the concatenated approach for all the three polynomial-kernel stages of the developed similarity fusion framework and for all the polynomial functions under investigation. When the Gaussian kernel is evaluated on the vectors processed using the concatenated approach, the observed results show a statistically significant improvement in the average classification accuracy of 22.269%, compared to when the linear kernel is evaluated on the vectors processed using the previously proposed approaches.

Contents

List of Figures

List of Tables

Acronyms

ANN	Artificial Neural Networks
KPCA	Kernel Principal Component Analysis
MKL	Multiple Kernel Learning
OPK	Orthogonal Polynomial Kernel
PSD	Positive Semi-Definite
RKHS	Reproducing Kernel Hilbert Space
SVM	Support Vector Machine

Chapter 1
Introduction

This chapter presents the context and rationale of the research work undertaken in this thesis. It also highlights the research problem addressed, the aim and objectives, the research methodology, and the novel contributions to knowledge introduced by the conducted research.

The chapter starts with a generic background on pattern recognition systems and kernel methods, in general and as applied to the Support Vector Machine (SVM) kernel-based classification algorithm in particular. The research problem and motivations of this work are then clarified, followed by the aim, objectives, and the methodology followed by the conducted research. Finally, the main contributions of the work are highlighted, followed by the outline of the thesis.

1.1 Background Concepts

1.1.1 Pattern Analysis and Recognition

Pattern analysis deals with the automatic search, detection, or recognition of patterns in data, and plays a central role in many modern artificial intelligence (AI) and computer science problems [1]. Patterns are defined as any characterising relations, regularities, or structure in the data. Therefore, by detecting a pattern in some available data, a system can expect to make predictions about new (unseen) data coming from the same source which produced the data that was originally available. This way, the system is said to have acquired some ‘generalization’ power by ‘learning’ something about the source generating the data. Generalization is therefore defined as the activity of inferring, from specific examples, a general rule which can also be applied to new examples [2]. As such, building a model capable of generalizing requires detecting and exploiting regularities (or patterns) in the data.

The general task of pattern analysis or recognition is therefore to find and study general types of relations or regularities depending on the task at hand (e.g., clusters, rankings, principal components, correlations, classifications) in general types of data (e.g., text documents, sets of points, vectors, graphs, images, videos, etc.). There are many complex

problems in AI and computer science that can currently be solved using only this approach, as it is often difficult or even impossible to specify an explicit standard theoretical or exact closed form solution for them (e.g., recognising genes in DNA sequences or recognising a face in a photo). Learning systems, therefore, can offer an alternative methodology by exploiting the knowledge extracted from a sample of data to infer an approximate solution for tackling such problems. In general, this learning approach is usually referred to as the "*data driven*" approach, in contrast to the "*theory driven*" approach that adopts precise specifications for an explicitly defined problem. Examples of problems that can only be tackled using this data driven approach are: text categorization, email filtering, gene detection, protein homology detection, image classification, handwriting recognition, etc. [1].

A pattern recognition system is therefore a system that adopts this data driven learning methodology to discover patterns in the data, and hence gain the ability to make predictions when presented with new data. Pattern recognition is the study of how to implement such systems into machines to get them to observe the environment, learn to distinguish areas or aspects of interest, and make reasonable decisions about the different categories observed [3, 4]. Overall, the best pattern recognizers that are currently available are humans, yet it is still not fully understood how humans recognize patterns.

1.1.2 Pattern Classification

The aim of many pattern recognition systems is classification, in which an observation is assigned to one of known predefined pattern classes. In statistical learning methods of pattern recognition, where it is mostly assumed that the data is represented in vector form, the relations (or patterns) in the data can be represented as classification rules, regression functions, or cluster structures. Many of these methods are usually referred to as 'statistical pattern recognition'. On the other hand, 'syntactical' or 'structural pattern recognition' represents an alternative approach that aims to detect classification rules among for example strings, which are often represented in the form of grammars or equivalent abstractions [1, 3]. A pattern is therefore understood as any relation present in the data, whether it is exact or approximate, depending on the nature of the recognition problem at hand.The basic design of an operating (machined) pattern classification system usually consists of the following main components, as shown in Figure 1.1: sensor or data acquisition device, pre-processing, feature extraction and selection, classifier, and post processing.

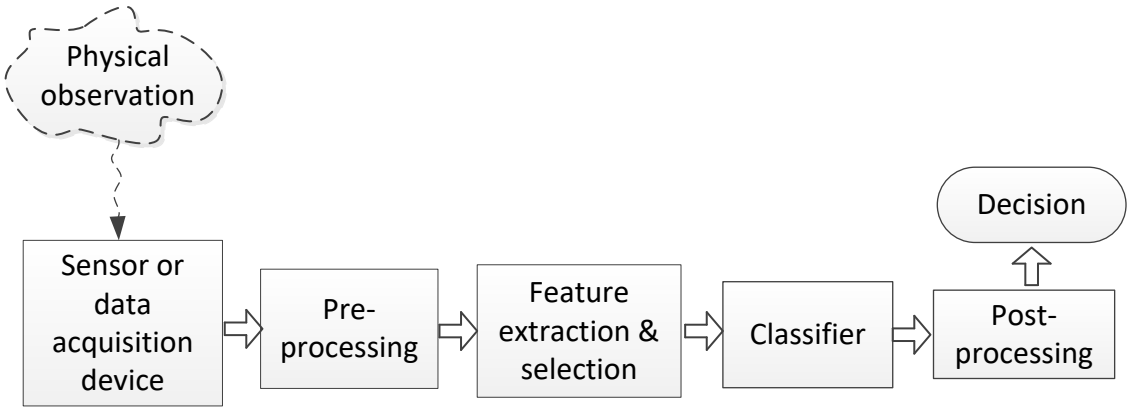


Figure 1.1 Main components of a pattern classification system.

The sensor or the data acquisition device (e.g., a camera) aims to gather raw information about the surrounding environment or the physical observation of interest, where patterns are sought. Some pre-processing is usually carried out on this raw data to assist in the better representation of the acquired information and hence facilitate the detection of patterns. Examples of these processes include, but are not limited to, separating (within an image) the areas of interest from the background or any insignificant information, removing noise, and normalizing the data. As the name implies, the feature extraction and selection step then aims to extract and select the most appropriate features that significantly discriminate the potential classes within the pattern sought and represent them in a feature space suitable for the employed classifier. In statistical pattern classification, for example, this feature space is an abstract space, where the data is usually represented in a vector form, whereby each data input is represented by a set of m features, or attributes. This is viewed as an m-dimensional feature vector in a multivariate (or multi-dimensional) space, where the number of dimensions in this space is equal to the number of extracted and selected features used to describe the input data.

The classification process then comes at the heart of the pattern recognition system, where each input data is recognised as belonging to one of many predetermined classes. As the number of classes and the similarity between these classes increase, the pattern recognition task becomes more challenging. This is known as the multi-class classification. Binary classification, on the other hand, is the case where the feature space contains only two classes, and it is a much easier classification task to solve than its multi-class counterpart.

Some recognition systems also require some sort of post-processing before a final decision about the class label of the input data can be made. For example, when more than one

classifier is used to solve a multi-class classification problem, a post-processing unit is required to combine the decisions of individual classifiers.

A number of commercial products are already available for use, based on the utilization of the above described statistical pattern classification approach, such as: machine printed character recognition, speech recognition systems, and automatic number plate recognition systems [3]. The process of constructing such operational systems, however, includes many challenging tasks, where residing at the core of the classifier is the utilization of a learning algorithm that produces the classifier used in the categorization process. This is commonly referred to as the training or the learning step.

In this step, the learning algorithm usually utilizes some available data from the same operating environment, known as the training set, to partition the selected feature space into class-labelled decision regions, such that examples from the same region share similar feature values, and examples from different regions have dissimilar feature values. The classification of a new input data includes finding out which decision region this input data belongs to and assigning it to that class. As such, this partitioning process involves the construction of an abstract function, sometimes referred to as a decision or hypothesis function, which separates each of the recognised classes. Statistical learning theory models this process as a function estimation problem [5], and the produced function is actually the classifier that will eventually be used to classify future data points.

The input feature space for the learning algorithm should be the same as the one used in the classification or operational process, and hence one of the critical steps of the training process is also to determine beforehand the best pre-processing, feature extraction, and feature selection procedures to be used in a given environment or pattern classification problem.

Following the training step, the classifier is then tested using a separate dataset, referred to as the test dataset, whose data points are not present in the original training dataset. This test method primarily determines the generalisation ability of the classifier and provides an indication about its performance in classifying future data points which were not used during the training stage. If the results are not satisfactory (according to a predefined performance measure), a feedback strategy might be used to optimize not only the model parameters of the learning algorithm, but also the pre-processing and feature extraction or selection strategies, if need be, to produce a better classification perfomance. This optimization process is usually repeated several times before the overall pattern classification system can have a robust performance for a given problem.

As such, each of the components of the pattern classification system bears its own responsibility towards determining the overall performance of the system. Besides the nature and complexity of the environment or the classification problem itself, there are also many other system factors and parameters that collectively contribute towards determining such a performance. For example, a degradation of the performance could be attributed to the misclassification of some, or all of, the new unseen input test samples. As the classification of such test samples is carried out using the classifier or decision rule constructed during the training stage, this therefore means that such a decision rule may not have been constructed properly to separate the different classes in the pattern classification problem under consideration.

The improper construction of a classifier or decision rule, which eventually leads to poor generalization ability, can be attributed to many factors. These include, but are not limited to [3]: (1) insufficient number of available training samples, (2) noisy or random values of the training samples themselves, (3) the number of features is too large compared to the number of training samples (the curse of dimensionality problem), (4) the classifier is too intensively optimized on the training set (the overtraining or overfitting problem), (5) large number of unknown classifier model parameters compared to the number of training samples.

There is an ample amount of research approaches in the literature that try to address each of the above-mentioned system problems (which are outside the scope of this thesis). In many real-life applications, however, it is usually the complex structure of the patterns, which are hidden in the data, which really makes the classification task itself a quite challenging problem to solve. In pattern classification terms, this is usually due to overlapped or non-linearly separable classes.

1.1.3 Kernel Methods and SVM

Classifying patterns that are linearly separable represents the first evolution of automated algorithms (e.g., the Perceptron) for pattern recognition in the 1960s [1, 2]. Nowadays, linearly separable classes represent a relatively easy and a well-founded classification task. Non-linearly separable classes, on the other hand, require the construction of more complex classifiers to enable their separation in the input feature space, which is a lot more challenging than the linear case. Several reasons can lead to the classes being non-linearly separable, such as: the actual classes themselves are indeed non-linearly separable, noise

intrusion (e.g., during the data acquisition process), poor extraction or selection of input features, etc.

In the mid 1980s, the field of machine learning underwent a 'non-linear revolution' to tackle this problem with the introduction of an ample of pattern recognition and learning algorithms. These include, but are not limited to: neural network (NN) algorithms (e.g., feed forward NNs, Multi-layer perceptron NN, Recurrent NN, Back propagation NN, etc.), decision tree predictive models (e.g., the iterative Dichotomiser 3 (ID3) and the C4.5 algorithms), instance-based learning algorithms (e.g., the *k*-nearest neighbour algorithm and RBF networks), and Bayesian classifiers. Such approaches for the first time made it possible to efficiently learn non-linear decision rules, and hence deeply influenced the evolution of many fields and paved the way for the creation of entire fields such as data mining and bioinformatics [2]. However, despite the huge amount of research attracted by the powerful setting of the non-linear rules devised by such algorithms, yet, most of them were based, to a great extent, on gradient decent and greedy heuristics, as opposed to the attractive theoretical elegance and practical convenience of the linear systems of the 1960s. As such, most of them frequently suffered from such problems as local minima and overfitting [1, 2].

It was not until the mid 1990s when the emergence of an innovative class of algorithms, known as *kernel-based learning methods* or *kernel methods*, managed to introduce a more systematic approach to address the problem of non-linearly separable classes. These kernel-based learning algorithms utilize techniques from optimization, statistics, and functional analysis to achieve maximal generality, flexibility, and performance, both in terms of generalization and computational cost [2]. They are explicitly based on a theoretical model of learning rather than on loose analogies with natural learning systems or other heuristics. Such methods enabled researchers to classify non-linearly separable patterns with the efficiency, foundational concept, and computational convenience that had previously been a characteristic for linear algorithms [1]. They were also able to overcome the problems of local minima and overfitting that were typical to neural networks and decision trees, as their training amounts to convex optimization. Moreover, these kernel-based algorithms have also proved to be effective for a wide variety of data types ranging from strings, text documents and images to more complex data types, for example those associated with biosequences, graphs, and grammars [1].

Kernel methods are now well renowned for tackling several machine learning tasks, as their ease of use, theoretical appeal, and remarkable performance have made them a system

of choice for many machine learning problems [2]. Research efforts have shown that they have been successfully applied to solve a wide range of practical applications from text categorization through handwriting recognition, to classification of gene expressions. A wide range of algorithms (e.g., kernel principal component analysis (KPCA), and kernel Fisher discriminant analysis) have also appeared in the machine learning community as successful examples of implementing the kernel-based learning methods [6-8]. The first and most popular implementation of the kernel-based approach, however, was the support vector machine (SVM), a supervised learning algorithm for pattern classification; which is the focus of the research work conducted in this thesis, as will be explained in Section 1.2.

The principal idea behind this kernel-based class of algorithms emerged from the observation that complex (or non-linear) patterns can be mapped to a different structure of regularities by changing the representation of the data such that they become easier to discover [1, 9]. This mapping process, usually to a higher-dimensional feature space, is carried out in the 'hope' that in this higher-dimensional feature space the data could become more easily separated or better structured. When tackling a non-linearly separable classification problem, the kernel-based SVM algorithm, for example, works by utilizing a 'kernel function' that maps the input space to an implicit higher-dimensional feature space, where the classes could become linearly separable, and hence easier to classify. Figure 1.2 illustrates this basic concept using a simple toy binary dataset example. It shows how the relation between the two different non-linearly separable classes becomes linearly separable when the data is represented in 2D instead of 1D.

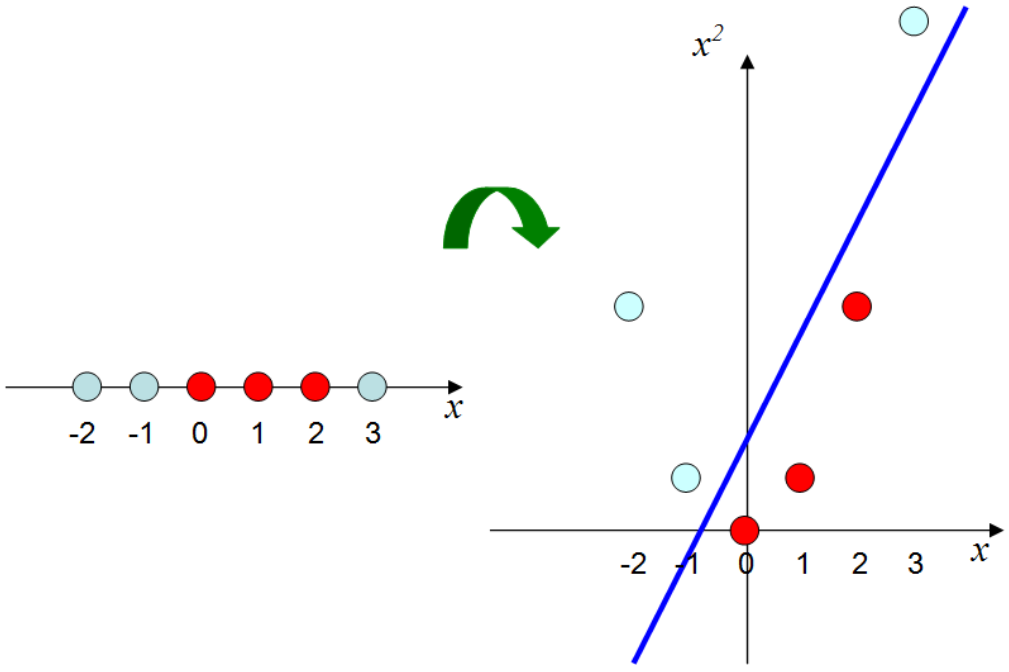


Figure 1.2 Illustrative example of the principal idea of the kernel-based SVM algorithm for pattern classification.

Unlike Artificial Neural Networks (ANNs), which followed a heuristic path, with applications and extensive experimentation preceeding theory, the development of SVMs, on the other hand, involved sound theory first then implementation and experiments. With their introduction in 1995 [10], SVMs marked the beginning of a new era in the learning from examples paradigm. Rooted from the statistical learning theory developed by V. Vapnik [11], SVMs quickly gained attention from the pattern recognition community due to a number of theoretical and computational merits.

For example, a significant advantage of SVMs is that whilst ANNs can suffer from multiple local minima, the solution to an SVM, on the other hand, is global and unique. Moreover, they present a simple geometrical interpretation and optimization of the margin and give a sparse solution. Furthermore, unlike ANNs, the computational complexity of SVMs does not depend on the dimensionality of the input space, and they are also a lot less prone to the problem of overfitting. The biggest attribute of SVMs, however, is perhaps the utilization of kernel functions which amount to mapping the input data to a higher-dimensional feature space, where overlapped classes can be linearly separable, as briefly explained above.

1.2 Research Problem and Motivations

Despite these attractive advantages of SVMs, they, however, have some drawbacks. For example, they can be abysmally slow, both in training and testing phases, when dealing with large datasets, and their employed quadratic programming can sometimes require extensive memory in large-scale tasks. The biggest limitation of SVMs, on the other hand, is perhaps the choice of the kernel function, and how it can be properly selected (together with any associated kernel parameters) to meet the needs of the classification task at hand.

As can be realized from the toy example illustrated in Figure 1.2, pattern classification using SVMs is, therefore, a two-stage approach [1]. The first stage is to map the input data to a higher-dimensional feature space, using a suitable kernel function, such that the classes become linearly separable. This higher-dimensional feature space is constructed implicitly using, what is known as, the ‘kernel trick’ (see Section 2.3.4 for detail). The second stage is then to apply one of the standard linear pattern classification algorithms to the transformed data. As such, the kernel function represents the key core of the SVM kernel-based learning algorithm for pattern classification, and therefore the resulting classification performance largely depends on the characteristics of the chosen kernel [12].

Extensive theoretical research has therefore been conducted in order to specify what characteristics such a function must have [1, 2, 7-9, 13-15]. This is because a valid kernel function is required to provide: (i) the notion of mapping the input space to implicit higher-dimensional feature spaces (so that linear separability between overlapped classes can be improved); (ii) the mathematical guarantee that it calculates the inner product between the mapped image vectors in the higher-dimensional feature space (so that the linear SVM algorithm can be used to establish linear separating hyperplanes in that space). This dual requirement can be expressed in an inner product mathematical form as $k(\mathbf{x},\mathbf{z}) = <\Phi(\mathbf{x}),\Phi(\mathbf{z})>$, where $\Phi(\cdot)$ is the mapping function[1], and **x** and **z** are the input patterns.

To achieve this purpose, the existing literature seems to have reached a generic consensus that for a function to be a valid kernel, it needs to be positive semi-definite (PSD), or equivalently, meet Mercer's condition [1, 2, 7, 9, 13, 16]. Moreover, the positive semi-definiteness property is also said to ensure that the optimization problem employed by the SVM algorithm can be efficiently solved using convex quadratic programming, otherwise, optimum parameters can be too difficult or computationally expensive to obtain [16, 17]. So far, a rich list of kernel functions, or even classes of kernels, have been reported to target the solution of many different problems; the most common of these functions are the Gaussian radial basis function (RBF), Polynomial, and Sigmoidal kernels [1, 9, 18].

At the same time, by calculating the inner-product between the mapped vectors in some high-dimensional feature space, as set by the positive semi-definiteness property shown above, a kernel function has also been defined as a generalised 'similarity measure' tool in manifold contexts [9, 16, 19], to calculate the similarity between its two input arguments. However, despite the elegant former theory of positive semi-definite kernels as being mapping tools to implicit higher-dimensional features spaces, and its remarkable implications on the performance of the SVM algorithm in various applications, yet, it seems to have dominated over the later definition of kernels as similarity measures. In other words, few research efforts seem to have studied SVM kernels from the 'similarity measure' perspective rather than the positive semi-definiteness point of view.

Paying more attention to kernels as tools to measure the similarity can, however, enable a machine learning practitioner (in the design process) to craft SVM kernels using some natural and intuitive 'similarity-based' properties (that describe how the input data belonging to same or different classes resemble or differ from each other). These are more

[1] See Example A.1 in Appendix A for a simple illustrative example.

tangible properties than the 'mathematical' properties of the implicit high-dimensional feature spaces, which one might not even be able to calculate [20-23]. Moreover, restricting kernels to positive semi-definite functions may rule out several natural notions of similarity that arise in many practical applications, which are not positive semi-definite, and yet, can be used successfully for learning to achieve reasonable generalisation levels, as demonstrated by some researchers, such as [24-29].

The research presented in this thesis addresses the problem of investigating what intuitive similarity-based properties can be utilized to characterise SVM kernels when used as similarity functions to measure the degree of resemblance between its two input arguments, rather than the mathematical properties of the implicit high-dimensional feature spaces. Given that a typical similarity function returns a scalar quantity indicative of how its two inputs resemble each other (where the larger the similarity, the more alike the two inputs are, and vice versa) [30], the thesis explores the utilization of these 'similarity-based' properties to define appropriate criteria of how one would intuitively wish a good SVM kernel to perform, and investigates the implications of these properties on the resulting SVM classification performance.

1.3 Aim and Objectives

The aim of the research presented in this thesis is to investigate the utilization of appropriate and intuitive similarity-based properties to characterise kernels when used as similarity functions to measure the degree of resemblance between its two input patterns, and to assess the influence of these properties on the resulting SVM classification performance.

The outreach of this research will therefore open new opportunities to craft or select kernels based on how patterns from the same/different classes are similar to (or different from) each other, which are more tangible quantities than the mathematical properties of the implicit high-dimensional feature spaces that one might not even be able to calculate.

To accomplish this aim, the objectives of the research were as follows:

1- Conduct a detailed study on the properties of kernel functions, together with their role in kernel-based learning algorithms in general and in the SVM for pattern classification in particular.

2- Review the existing open problem of kernel function selection and the previous approaches that utilize the properties of kernels to devise a series of kernel-combination strategies that aim to automatically select the most suitable kernel to the classification task at hand.

3- Investigate the characteristics and role of (dis)similarity functions in metric-based learning algorithms and their relationship to kernel functions, being defined as similarity measures.

4- Devise appropriate 'similarity-based' criteria for SVM kernel functions and investigate their potential impact on achieving correct class-label decisions when implemented within the kernel-based SVM classification algorithm.

5- Analyse the effectiveness of utilizing the devised 'similarity-based' criteria in the calculation of accurate similarity measures between the two input patterns of the underpinning SVM kernel function, with focus on the so-called "orthogonal polynomial kernels".

6- Conduct a comprehensive set of experiments, using suitable benchmark datasets, to evaluate the proposed similarity-based approach on the resulting SVM classification performance.

1.4 Research Methodology

A key outcome of the research work presented in this thesis is the development of intuitive and easily interpretable similarity-based properties, exemplified by the *shape* characteristics of a kernel, to demonstrate how a typical kernel (being defined as a similarity function) should ideally look like when calculating the degree of resemblance between its two input patterns, for SVM classification. Motivated by the pictorial characteristics of the isotropic stationary class of kernels reported by Genton [18] in general, and the Chebyshev kernel plots reported by Ozer et al. [17] in particular, the thesis adopted the previously reported orthogonal polynomial kernels (such as the Chebyshev [31-33], Legendre [34], and Hermite [35] polynomials) to propose a number of novel approaches that investigate how the appropriate utilization of the developed similarity-based shape properties can effectively influence the performance of the SVM classifier.

The systematic methodological customization of the proposed approaches using these orthogonal polynomial kernels is briefly summarised as follows:

- First, a comprehensive study and analysis of the role of kernel functions in kernel-based methods in general, and as applied to SVM classification in particular, has been conducted. The aim of this study was to highlight the identified problem, where the utilization of the kernels' mapping characteristics to implicit high-dimensional feature spaces dominates the study of, and the potential benefits that can be gained from, utilizing their similarity-based properties.

- A detailed analysis is afterwards conducted on the kernels constructed from orthogonal polynomials (e.g., Chebyshev, Legendre, and Hermite) to investigate how their resulting SVM classification performance can be explained by their exhibited shape characteristics.

- A three-stage similarity fusion framework is developed (in Chapter 4), within which the hierarchical construction of these polynomial kernels is formally articulated and their resulting SVM classification performance is evaluated.

- An appropriate method is afterwards implemented (in Chapter 5), using a simple adaptive data normalization approach, to confine the input data to the regions (within the normalized input space), where the polynomial kernels demonstrate the sought after ideal shape characteristics. This normalization approach eliminates the possibility of any data point to be located outside this region, which, consequently, led to more accurate calculation of similarity measures and, hence, improved their resulting classification performance.

- Finally, a new perspective is also proposed (in Chapter 6) whereby the utilization of orthogonal polynomials is perceived as a way of transforming the input space to another vector space, of the same dimensionality, prior to the kernel calculation step. Based on this perspective, a novel processing approach, implemented using vector concatenation, is proposed which, unlike the previous processing approaches, ensures that the polynomial-processed quantities are always formulated in a vector form for any polynomial order, and hence maintains the attributes embedded in the structure of the original vectors intact. It can also be used with any polynomial function, regardless of the parity combination of its monomials, whether they are only odd, only even, or a combination of both. Moreover, the Gaussian kernel is also proposed to be used to evaluate the similarity between the transformed vectors using the concatenated approach (instead of the linear kernel used with the previous approaches), due to its more accurate calculation of similarity measures (in terms of its exhibited shape characteristics), as well as its renowned ability to implicitly map the input space to an

infinite-dimensional feature space. The influence on the SVM classification performance, as a result of implementing these approaches, was also assessed quantitatively using all the three kernel stages of the developed similarity fusion framework and all the polynomial functions under investigation.

As such, the conducted investigations involved the development of theoretical framework models and algorithms, their implementation in software (using MATLAB and a number of relevant toolboxes), and their evaluation and assessment using comprehensive and well-designed experiments, utilizing a number of benchmark datasets, anchored on quantitative performance assessment methods.

The ethical implications of the research presented in this thesis have also been considered and it was concluded that it does not result in any issues, and hence there was no need to apply for any formal ethical approvals throughout the course of the research programme. In accordance with Staffordshire University Ethical Review Policy, this is because:

1- The research does not involve human or animal participants.

2- The research does not present an indirect risk to non-participants (human or animal).

3- The research does not raise ethical issues due to the potential social or environmental implications of the study.

4- The research does not re-use previously collected personal data which is sensitive in nature or enables the identification of individuals.

1.5 Contribution to Knowledge

Having successfully tackled and validated the proposed solutions to the research problem identified in Section 1.2, the main contribution of this thesis is multi-faceted:

1- A new analytical approach is proposed to exploit the potential benefits on the SVM classification performance when utilizing appropriate similarity-based characteristics, as exhibited by the shape of the underpinning kernels (being thought of as a similarity measures), as well as their standard definition as mapping tools to implicit high-dimensional feature spaces, as prescribed by their positive semi-definiteness property.

2- By adopting the previously proposed orthogonal polynomial kernels, a novel three-stage similarity fusion framework has been developed. This framework aims to

demonstrate that the summative and multiplicative fusion operations, inherent in their natural hierarchical structure, synergistically contribute towards the calculation of more accurate similarity measures, as demonstrated by their shape characteristics, and hence, lead to an improved classification performance.

3- A new method is also developed, based on an adaptive data normalization approach, to show that when the shape of the kernel is as close as possible to the characteristics of the ideal similarity function, more accurate similarity measures can be calculated than otherwise; and accordingly, better classification performance can be obtained. This is achieved by confining the data to the regions within the normalized input space where the shape of the polynomial kernels demonstrates the sought after ideal characteristics, and hence eliminate the possibility of any data points to be located outside this region where their shape oscillates in a wavy pattern.

4- Finally, a new perspective has also been proposed whereby the utilization of orthogonal polynomials is viewed as a way of transforming the input data to another vector space, prior to the kernel calculation step. Based on this view, a novel processing approach (Referred to in this thesis as the 'concatenated' processing approach) has also been developed, which, unlike the previous processing approaches, ensures that the polynomial-processed quantities are always formulated in a vector form for any polynomial order, and hence maintains the attributes embedded in the structure of the original vectors intact. The proposed concatenated processing approach can also be used with any polynomial function, regardless of the parity combination of its monomials. Furthermore, the Gaussian kernel is proposed to be used in conjunction with the concatenated approach, instead of the linear kernel, due to its more accurate calculation of similarity measures, as exhibited by its shape characteristics, as well as its renowned ability to map the input space to an implicit high-dimensional feature space.

To the knowledge of the author, no previous work in the literature has identified or addressed the research problem highlighted in this thesis, and no previous work has presented the above solutions or approaches for SVM pattern classification, nor have the experimental investigations conducted in this thesis ever been done before. As such, the comparative analyses, research investigations, and the empirical assessments conducted in this thesis are all considered herein as novel and original contributions to the knowledge.

1.6 Thesis Outline

This thesis is organised as follows:

Chapter 1 presents some generic background concepts about pattern recognition and classification in general, and as applied by the SVM kernel-based classifier in particular. This presentation is aimed at articulating a narrowing down hierarchy that smoothly drives the reader towards the main research problem identified and addressed by the thesis. The aim and objectives of the conducted research are afterwards clarified, followed by the methodology used in the research and the contributions to knowledge achieved. Finally, the thesis organization and outline are herein detailed.

In **Chapter 2**, an analytical review of the SVM classifier is presented to elaborate on the kernel module within which the work of this thesis is focused on. The characteristics and properties of SVM kernel functions are afterwards critically investigated, to highlight the main problem identified and addressed by the thesis.

Chapter 3 then analyses in more detail the identified research problem: namely, the utilization of the similarity-based characteristics of kernels, along side with their standard definition as mapping tools to implicit high-dimensional feature spaces, as prescribed by their positive semi-definiteness property. Based on the definition of a kernel as a measure of similarity between its two input arguments, the chapter then defines some appropriate criteria, based on the shape characteristics of the ideal similarity function, in terms of how one would intuitively wish a kernel function to behave. A detailed comparative analysis is then conducted between such an ideal shape characteristic model and the shape exhibited by the previously proposed orthogonal polynomial kernels, such as the Chebyshev, Legendre, and Hermite kernels, to assess the accuracy of their calculated similarity measures and the expected impact on their resulting SVM classification performance.

The analysis of such polynomial kernels revealed that they are naturally constructed from a mixture of summative and multiplicative base kernel building blocks, which synergistically contribute towards the calculation of more accurate similarity measures, as demonstrated by their shape characteristics. Such behaviour is therefore formally defined within a new three-stage similarity fusion framework developed in **Chapter 4**, whereby the hierarchical composite structure of the polynomial kernels is broken down to their individual kernel components and the effect on their resulting SVM classification accuracy, based on their exhibited shape characteristics as well as their synergistic fusion power, is experimentally explored.

Chapter 5 then proposes a simple method, based on an adaptive data normalization approach, to confine the input data to the regions within the normalized vector space, where the employed polynomial kernels demonstrate the sought after ideal shape characteristics. Hence eliminate the possibility of any data point to be located outside these regions, which could result in the calculation of inaccurate similarity measures, and hence lead to a degraded classification performance. The effectiveness of this approach is also assessed experimentally using the shape characteristics exhibited by the polynomial kernels under investigation.

Due to the extensive study of the polynomial kernels paradigm (such as the Chebyshev, Legendre, and Hermite kernels), **Chapter 6** then introduces a new perspective whereby the utilization of these polynomials to process the input vectors is perceived as a way of transforming the input space to another vector space of the same dimensionality, prior to the kernel calculation step. To appropriately implement such a transformation perspective, the chapter then proposes a new processing approach, based on vector concatenation, to ensure that the processed quantities will always be formulated in a vector form for any polynomial order applied, and hence the transformed image vectors will be good representatives of their original vectors, as no information will be lost during the transformation process. The concatenated processing approach is also designed in a way such that it can also be used with any polynomial function, regardless of the parity combination of its monomials. The chapter also proposes to evaluate the Gaussian kernel on the polynomial-processed vectors, instead of the linear kernel used in previous approaches, due to its more accurate calculation of similarity measures, as exhibited by its shape characteristics, as well as its renowned ability to map the input space to an implicit high-dimensional feature space. The effectiveness of this proposed concatenated approach is afterwards experimentally assessed using all the kernel stages of the developed similarity fusion framework and all the polynomial kernels under investigation.

Finally, **Chapter 7** discusses the conclusions of the work presented in this thesis and offers some directions for future work.

Chapter 2
Literature Review on the Support Vector Machine and the Properties of Kernel Functions

2.1 Introduction

This chapter presents a systematic and critical survey on kernel functions in general and as utilized by the SVM kernel-based classifier in particular. The aim is to conduct a deep study on the role and properties of kernels to set the scene for the underpinning investigations and analytical solutions proposed in this thesis to tackle the identified research problem.

The chapter therefore starts with a quick review of the well established SVM classifier to elicit the kernel module upon which the thesis is focused. The characteristics and properties of kernels are afterwards detailed with a focus on the legitimate operations (and previous research efforts) used to construct more complex kernels from smaller building blocks to enable the solution of more sophisticated classification tasks. Framed by this study, the identified research problem is afterwards elaborated and the rationale for devising its solution is briefly discussed.

2.2 SVM: The kernel-based pattern classification algorithm

Kernel methods first appeared when applied to solve statistical non-linear classification problems [5, 36]. This yielded the Support Vector Machine (SVM) algorithm in 1992 [2, 6, 10], a supervised learning algorithm that was able to overcome the local minima and overfitting difficulties of the previous generation learning algorithms. Since then, the SVM has been extensively studied, greatly generalized, and successfully applied to a number of different pattern recognition problems, mainly classification and regression [7, 14, 37-46]. Due to its elegant characteristics and robust performance, the SVM has attracted the attention of a wide community of researchers from many different disciplines to an extent that made it the best known element in the class of algorithms adopting the kernel-based approach to pattern recognition [13].

2.2.1 SVM for pattern classification

When training a classifier, it is often desirable to maximize the classification performance for the training data [12]. However, if the classifier fits the data too closely, the classification ability for unknown data, i.e., the generalization ability, is degraded. This phenomenon is called *overfitting*. Therefore, there must be a trade-off between the generalization ability and fitting to the training data. In the pattern classification literature, various methods have been proposed to prevent overfitting. One of the main ideas is (as applied in support vector machines) to add a regularization term which controls the generalization ability of the objective function.

For a binary classification problem, a support vector machine is trained so that the decision function maximizes the generalization ability. This is achieved by utilizing a quadratic programming problem that is solved to separate the two classes using an optimal separating hyperplane. Therefore, given a set of training examples, each marked as belonging to one of two classes, the SVM builds a model that assigns new examples into one class or the other. Moreover, the SVM model is built so that the examples of the different classes are divided by a clear gap that is as wide as possible. New examples are then processed using that model and predicted to belong to a class based on which side of the gap they fall on.

The following sub-sections discuss how this process takes place for the different types of training datasets. If the training dataset is linearly separable in the input space, the problem is solved by what is known as the *hard-margin* or *linear* SVM. This is afterwards extended to the *soft-margin* SVM case where the training dataset is non-linearly separable. In this case, the SVM utilizes the so-called *kernel function* to map the input space to a high-dimensional feature space to enhance the linear separability in the feature space. This mapping process is key to the operation of the soft-margin SVM and is an important element behind its remarkable performance in tackling non-linearly separable classification tasks.

2.2.2 Hard-margin support vector machines

Suppose some given data points each belongs to one of two classes, and the aim is to decide which class a new data point will belong to. In SVM binary classification, a data point is viewed as an m-dimensional vector, and we want to know whether we can separate such points in two classes with an (m-1) dimensional hyperplane. This is called a linear

classifier. There are, however, many hyperplanes that might be able to classify the data, as shown in Figure 2.1. One reasonable choice as the best hyperplane, is the one that represents the largest separation, or margin, between the two classes. The SVM finds the hyperplane so that the distance from it to the nearest data point on each side is maximized [47].

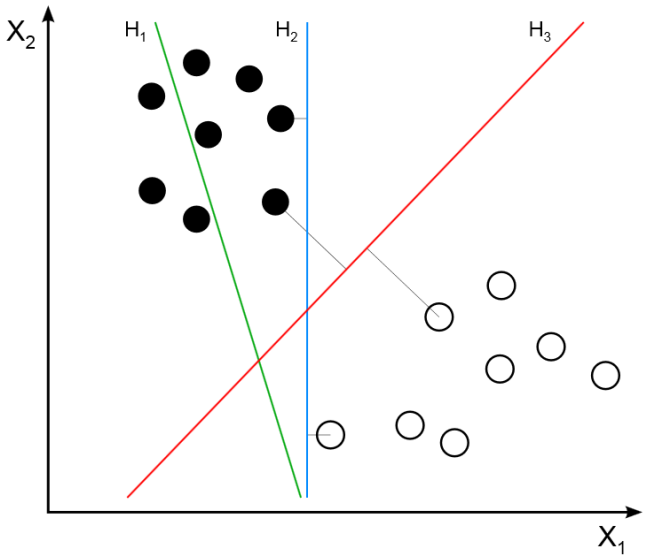


Figure 2.1 H_1 does not separate the classes. H_2 does, but only with a small margin. H_3 separates them with the maximum margin [47].

Therefore, given some *m*-dimensional finite training dataset $S = \{\mathbf{x}_i, y_i\}_{i=1}^{l}$, where $\mathbf{x}_i \in \Re^m$ and the associated labels $y_i = +1$ for Class 1 and $y_i = -1$ for Class 2. Each $\mathbf{x}_i$ is an *m*-dimensional real vector. The aim is to find the maximum-margin hyperplane that divides the points having $y_i = +1$ from those having $y_i = -1$. Any hyperplane can be written as the set of points **x** satisfying

$$\langle \mathbf{w}, \mathbf{x} \rangle - b = 0 \qquad (2.1)$$

where **w** is the normal vector to the hyperplane and *b* is a bias term. The parameter $\frac{b}{\|\mathbf{w}\|}$ determines the offset of the hyperplane from the origin, as shown in Figure 2.2.

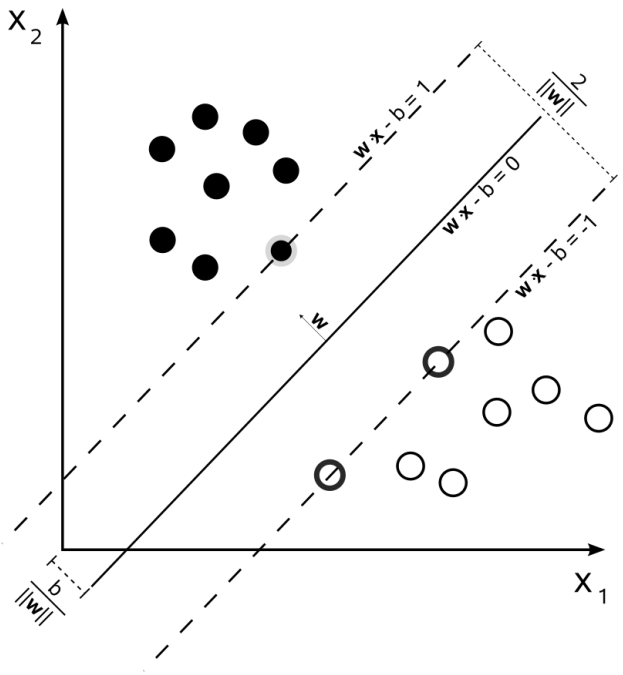


Figure 2.2 Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called the support vectors [47].

If the training data is linearly separable, two hyperplanes can be selected in a way that they separate the data and there are no points between them, and then try to maximize the distance between them. These two hyperplanes can be described by:

$\langle \mathbf{w}, \mathbf{x} \rangle - b = 1$

and

$\langle \mathbf{w}, \mathbf{x} \rangle - b = -1$

where the distance between these two hyperplanes, given by $\frac{2}{\|\mathbf{w}\|}$, is called the *margin*. As the main objective is to maximize this margin, $\|\mathbf{w}\|$ must be minimized. Since no data point from *S* can fall into the margin, the data points need to satisfy the following constraints:

$\langle \mathbf{w}, \mathbf{x}_i \rangle - b \geq 1$ for $\mathbf{x}_i$ of the first class

and

$\langle \mathbf{w}, \mathbf{x}_i \rangle - b \leq -1$ for $\mathbf{x}_i$ of the second class

Collectively, for all the dataset points $i = 1, \ldots, l$, this can be written as

$$y_i\left(\langle \mathbf{w}, \mathbf{x}_i \rangle - b\right) \geq 1 \tag{2.2}$$

Consequently, the main task now is to minimize $\|\mathbf{w}\|$ so that the margin is as large as possible, subject to $y_i\left(\langle \mathbf{w}, \mathbf{x}_i \rangle - b\right) \geq 1$. In the SVM, this is achieved by solving the following quadratic programming convex optimization problem for obtaining both **w** and *b*, and hence, the optimal separating hyperplane can be found:

$$\text{minimize} \quad Q(\mathbf{w}, b) = \frac{1}{2}\|\mathbf{w}\|^2 \tag{2.3}$$

$$\text{subject to} \quad y_i\left(\langle \mathbf{w}, \mathbf{x}_i \rangle - b\right) \geq 1 \quad \text{for all } i = 1, \ldots, l. \tag{2.4}$$

An important characteristic about the SVM that stems from the above optimization problem is the fact that the optimal separating hyperplane is obtained from the two hyperplanes that define the maximum margin. This means that the only data points required to calculate the optimal separating hyperplane are those touching the margins (highlighted in Figure 2.2 as the bold and shaded circles), which are usually a lot less than the total number of examples in the training dataset. This means that we can obtain the same optimal separating hyperplane even if we delete all the other data points that satisfy the strict inequality in (2.4), i.e., those which satisfy only the >1 constraint, and use only the data points that satisfy the equality, i.e., =1, constraint. These points are usually referred to as the *support vectors*. An important outcome of this observation is the fact that the optimal separating hyperplanes established by the SVM algorithm depend only on the number of support vectors, and not on the number of features, whether this be the number of features of the dataset in the input space or, as will be shown later, the number of features of the mapped dataset to a higher-dimensional feature space. This is the main reason why the SVM algorithm is not prone to the problem of the *curse of dimensionality*.

On the other hand, the variables of the convex optimization problem given by (2.3) and (2.4), which are usually referred to as the *primal solution*, are **w** and *b*. Because **w** is a vector of the same dimensionality (*m*) as the input data, the total number of variables is therefore equal to the number of features plus 1: *m*+1 [12]. When the number of input variables is small, the quadratic programming technique can handle solving (2.3) and (2.4). However, as will be discussed later, because we map the input space into a higher-dimensional feature space, it is therefore important to convert (2.3) and (2.4) into their

equivalent *dual representation*[2] whose number of variables is the number of samples in the training data instead [12, 48].

Another important reason for adopting the dual representation is to prepare the algorithm for, as will be shown later, the non-linear mapping process that takes place during the 'kernelization' of the algorithm to enable its application to solve non-linear classification problems. In other words, it is the dual representation of the convex optimization problem that will formalize the solution in terms of inner products of the training examples.

This can be achieved by applying the Lagrangian multipliers method that reduces the optimization problem to the following dual form:

$$\text{maximize} \qquad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,\, j=1}^{l} \alpha_i \alpha_j y_i y_j \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle \tag{2.5}$$

$$\text{subject to} \qquad \sum_{i=1}^{l} y_i \alpha_i = 0 \text{, and } \alpha_i \geq 0 \text{ for } i = 1, \ldots, l. \tag{2.6}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_l)$ and α_i are the non-negative Lagrangian multipliers.

This formulated dual form of the optimization problem is known as the *hard-margin* SVM. Finally, the decision function can be given by

$$f(\mathbf{x}) = \text{sign} \left(\sum_{i \in \text{SV}} \alpha_i y_i \left\langle \mathbf{x}_i, \mathbf{x} \right\rangle + b \right) \tag{2.7}$$

where SV is the set of support vector indices and b is averaged over all of the support vectors and can be defined as [12]

$$b = \frac{1}{|\text{SV}|} \sum_{i \in \text{SV}} \left(y_i - \left\langle \mathbf{w}, \mathbf{x}_i \right\rangle \right). \tag{2.8}$$

The SVM then uses (2.7) to find the label of a new test data **x** as:

$$\begin{aligned} &\text{Class 1} \qquad \text{if } f(\mathbf{x}) > 0, \\ &\text{Class 2} \qquad \text{if } f(\mathbf{x}) < 0. \end{aligned} \tag{2.9}$$

[2] In mathematical optimization theory, **duality** means that optimization problems may be viewed from either of two perspectives, the primal (minimization) problem or the dual problem (the **duality principle**). However, in general for quadratic programming convex optimization problems, the values of the primal and dual problems coincide (i.e., are equal) at the optimal solutions. This is called the *zero-duality gap*.

If $f(\mathbf{x}) = 0$, then $\mathbf{x}$ is on the boundary and therefore is unclassified. Accordingly, if the training data is linearly separable, the region of $\mathbf{x}$ where $1 > f(\mathbf{x}) > -1$ is the generalization region [12].

2.2.3 Soft-margin support vector machines

The hard margin SVM described in the previous sub-section assumes that the training data are linearly separable; i.e., there exists a solution to (2.5). If, however, the data are non-linearly separable, there is no feasible solution, and the hard-margin SVM is unsolvable.

In 1995, however, Corinna Cortes and Vladimir N. Vapnik suggested a modified maximum margin idea that allows for some examples to be misclassified when non-linearly separable data are separated by a hyperplane [10]. In simple terms, this idea states that: if there exists no hyperplane that can explicitly split the Class 1 and Class 2 examples, then the *Soft-margin* SVM method will choose a hyperplane that splits the examples as cleanly as possible, while still maximizing the distance to the nearest cleanly split examples. A large margin in this sense would mean how optimum or how clean the hyperplane would separate the two classes, as some examples will now have to be allowed to fall into the margin.

This is achieved by introducing what is referred to as non-negative *slack variables*, $\xi_i \geq 0$, which measure the degree of misclassification of data point $\mathbf{x}_i$. Using these slack variables, feasible solutions always exist. For the training data $\mathbf{x}_i$, if $0 < \xi_i < 1$, then the data do not have the maximum margin but are still correctly classified. However, if $\xi_i \geq 1$, then the data are misclassified by the optimal hyperplane. To obtain the optimal hyperplane, in which the number of training data that do not have the maximum margin (i.e., the number of training data that are misclassified) is minimum, the optimization problem is then reformulated to introduce a trade-off between a large margin and a small error penalty. This is given in the primal form as

$$\text{minimize} \qquad Q(\mathbf{w}, b, \xi) = \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{p}\sum_{i=1}^{l}\xi_i^p \tag{2.10}$$

$$\text{subject to} \qquad y_i\left(\langle \mathbf{w}, \mathbf{x}_i \rangle - b\right) \geq 1 - \xi_i \ , \ \xi_i \geq 0 \quad \text{for all } i = 1, \ldots, l, \tag{2.11}$$

where C is the margin parameter, known as the penalization parameter, that determines the trade-off between the maximization of the margin and the minimization of the

classification error. The value of the constant p is selected as either 1 (for linear penalty function) or 2 (for quadratic penalty function). When $p = 1$, the support vector machine is known as L1 soft-margin support vector machine (L1 SVM), and when $p = 2$ it is known as the L2 soft-margin support vector machine (L2 SVM). As the L1 SVM is the most commonly used in the machine learning literature, the L2 SVM will not be discussed further herein.

Similar to the hard-margin case, the primal form of the optimization problem in (2.10) is converted to its dual form counterpart using the non-negative Lagrange multipliers, which then reduces the optimization problem to:

$$\text{maximize} \qquad Q(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i,\, j=1}^{l} \alpha_i \alpha_j y_i y_j \left\langle \mathbf{x}_i, \mathbf{x}_j \right\rangle \tag{2.12}$$

$$\text{subject to} \qquad \sum_{i=1}^{l} y_i \alpha_i = 0, \text{ and } C \geq \alpha_i \geq 0 \text{ for } i = 1, \ldots, l. \tag{2.13}$$

where $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_l)$ and α_i are the non-negative Lagrangian multipliers. The only difference then between this L1 SVM and the hard margin SVM is that α_i cannot exceed C. Also notice that the advantage of using a linear penalty function (i.e., L1 SVM with $p = 1$ as mentioned above) is that the slack variables vanish from the dual problem leaving only the constant C as an additional constraint on the Lagrange multipliers.

Similarly, the decision function is also the same as the hard-margin SVM and is given by:

$$f(\mathbf{x}) = \text{sign} \left(\sum_{i \in \text{SV}} \alpha_i y_i \left\langle \mathbf{x}_i, \mathbf{x} \right\rangle + b \right) \tag{2.14}$$

where SV is the set of support vector indices and b is averaged over all of the support vectors as explained before. An unknown data sample $\mathbf{x}$ is then classified as:

$$\begin{aligned} &\text{Class 1} \qquad \text{if } f(\mathbf{x}) > 0, \\ &\text{Class 2} \qquad \text{if } f(\mathbf{x}) < 0. \end{aligned} \tag{2.15}$$

If $f(\mathbf{x}) = 0$, then $\mathbf{x}$ is on the boundary and is therefore unclassified, which is the same as the hard-margin SVM.

For this soft-margin formulation and its huge impact in practice, Cortes and Vapnik received the 2008 Association for Computing Machinery (ACM) Paris Kanellakis Award [47].

2.2.4 Kernelized SVM for non-linear pattern classification

Although the soft-margin SVM in (2.12) and (2.13) obtains an optimal hyperplane, yet it may not have a high generalization ability because the training data is not linearly separable. In other words, it is still a linear classifier that is producing a hyperplane that is splitting non-linearly separable data. However, having already prepared the algorithm to accept only the inner-products in the input space (by representing the optimization problem in the dual form as explained earlier in Section 2.2.2), it is now ready to be converted to its non-linear classifier version counterpart. Here is where the kernel methods approach is implemented in pattern classification as described earlier in this chapter, such that the SVM can now be a non-linear classifier instead.

This idea was first proposed, in 1992, by Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik [6] who suggested a way to create non-linear classifiers by applying what is known as the 'kernel trick' to linear classifiers which produce maximum margin hyperplanes. This idea was triggered by the observation that the overlapped classes in the input space can become linearly separable (or at least linear separability between them can be improved) if they are mapped to a higher-dimensional feature space via a kernel function that should still be able to calculate the inner-products between the mapped image vectors in this high-dimensional feature space instead of the original vectors in the input space. To do this, the inner-product in the linear algorithm is simply replaced by a kernel function with specific characteristics. This will then have the effect of fitting a maximum-margin hyperplane in the transformed high-dimensional feature space rather than the input space. As such, the constructed linear hyperplane classifier in the high-dimensional feature space is equivalent to the sought after non-linear classifier in the input space, and this way, non-linearly separable data can now be classified using the robust generalization power of the 'kernelized' SVM.

Therefore, as long as the kernel is a symmetric function that satisfies

$$\sum_{i,j=1}^{l} h_i\, h_j\, k\left(\mathbf{x}_i, \mathbf{x}_j\right) \geq 0 \quad \text{, for all } l, \mathbf{x}_i, \text{ and } h_i \quad , \tag{2.16}$$

where l takes on a natural number and h_i take on real numbers, then this means that there exists a mapping function, $\Phi(\mathbf{x})$, that maps $\mathbf{x}$ into the inner-product feature space, and also satisfies

$$k(\mathbf{x}_i, \mathbf{x}_j) = \Phi^{\mathrm{T}}(\mathbf{x}_i)\, \Phi(\mathbf{x}_j). \tag{2.17}$$

If (2.17) is satisfied, then the condition in (2.16) is also satisfied as follows:

$$\begin{aligned}\sum_{i,j=1}^{l} h_i\, h_j\, k(\mathbf{x}_i, \mathbf{x}_j) &= \sum_{i,j=1}^{l} h_i\, h_j\, \Phi^{\mathrm{T}}(\mathbf{x}_i)\Phi(\mathbf{x}_j) \\ &= \left(\sum_{i=1}^{l} h_i\ \Phi^{\mathrm{T}}(\mathbf{x}_i)\right)\left(\sum_{j=1}^{l} h_j\ \Phi(\mathbf{x}_j)\right) \\ &= \left\|\sum_{i=1}^{l} h_i\, \Phi(\mathbf{x}_i)\right\|^2 \geq 0\end{aligned} \tag{2.18}$$

The condition in (2.18) is important to guarantee that $k(\mathbf{x}_i,\mathbf{x}_j)$ is a valid kernel; i.e., it does indeed equate to the inner-product between the image vectors in some high-dimensional feature space and hence the optimization problem solved by the SVM classifier will be convex and the solution found will be unique. This condition is commonly known as the 'Mercer's condition' and a function that satisfies it is usually referred to as the a 'positive semi-definite' (PSD) kernel [12].

By utilizing the kernel trick, we do not need to treat the high-dimensional feature space explicitly, and therefore, the kernel function $k(\mathbf{x}_i, \mathbf{x}_j)$ is applied directly to the input space both in the training and testing phases. Therefore, using kernels, a non-linear SVM classifier can then be created by reformulating the dual form of the optimization problem as:

maximize $$Q(\boldsymbol{\alpha}) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{l} \alpha_i \alpha_j y_i y_j\, k(\mathbf{x}_i, \mathbf{x}_j) \tag{2.19}$$

subject to $$\sum_{i=1}^{l} y_i \alpha_i = 0, \quad \text{and } 0 \leq \alpha_i \leq C \quad \text{for } i = 1, \ldots, l\,. \tag{2.20}$$

As mentioned earlier, because $k(\mathbf{x}_i, \mathbf{x}_j)$ is a PSD kernel, then the optimization problem is a concave quadratic programming problem. And because $\alpha = 0$ is a feasible solution, then the problem has a global optimum solution. This is one of the advantages of SVM over neural networks, which can have local minima [12].

As such, the decision function can be given as:

$$f(\mathbf{x}) = \text{sign}\left(\sum_{i \in \text{SV}} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) + b\right) \tag{2.21}$$

and unknown data are classified using the decision function as:

$$\mathbf{x} \in \begin{cases} \text{Class 1} & \text{if} \quad f(\mathbf{x}) > 0 \\ \text{Class 2} & \text{if} \quad f(\mathbf{x}) < 0 \end{cases}. \tag{2.22}$$

If $f(\mathbf{x}) = 0$, then $\mathbf{x}$ is unclassifiable.

It is important to note here that for binary classification, the output of the SVM algorithm is one of three possibilities (Class 1, Class 2, and unclassifiable), and not only the two binary classes possibilities, as sometimes mistakenly assumed. There are various variants of support vector machines available in the literature; and if the word *regular support vector machine* or *standard support vector machine* is used, it denotes the above non-linear version of the L1 soft-margin SVM in (2.19) and (2.21). As this is the most commonly used type of SVM (and to be consistent with similar work in the literature), it will be the one used in this thesis to develop the solutions proposed for the identified research problem.

To successfully tackle non-linearly separable pattern classification tasks, the process of mapping the input space via the utilization of an appropriately constructed kernel function is critical to the performance of the SVM classifier. As the work presented in this thesis is focused on the kernel module, and to enable the formulation of robust solutions to the identified research problem, it was necessary to conduct careful investigation and detailed study on the properties of such kernels. This investigation constitutes an important part of the work presented in this thesis and is summarised in the following sub-sections.

2.3 Definition and role of SVM kernel functions

If the candidate function is a valid Mercer's kernel, this means that the condition in (2.17) is also satisfied. This is another important property of kernels that enables their indirect calculation of the inner products between two vectors in the feature space which are the images of their counterpart original input data points pair, as shown in Figure 2.3. In the general setting of kernel-based learning algorithms, if the adopted algorithm is adapted to use only the inner products between the input data points, then it can be combined with a kernel function that indirectly calculates the inner product between two data points in the feature space which are the images of their counterpart original input data points pair. If this process is conducted successfully, the algorithm is then said to have been 'kernelized'. As for the SVM, it has been shown earlier in (2.12) how it was adapted to accept only inner products. It was therefore possible to 'kernelize' it, in the way shown in (2.19), by

replacing the inner-product with the kernel function, being also represented in an inner-product form, though in a different feature space.

Given a training dataset, the values of these pairwise inner products, computed directly from the original data points via the evaluation of the kernel function *k*, eventually make up what is referred to as the *kernel matrix* which contains all the information required by the SVM algorithm to formulate a hyperplane function in the high dimensional feature space, during the training phase. It is this hyperplane function that the SVM uses afterwards in the testing phase to classify unseen examples that were not originally present in the training dataset.

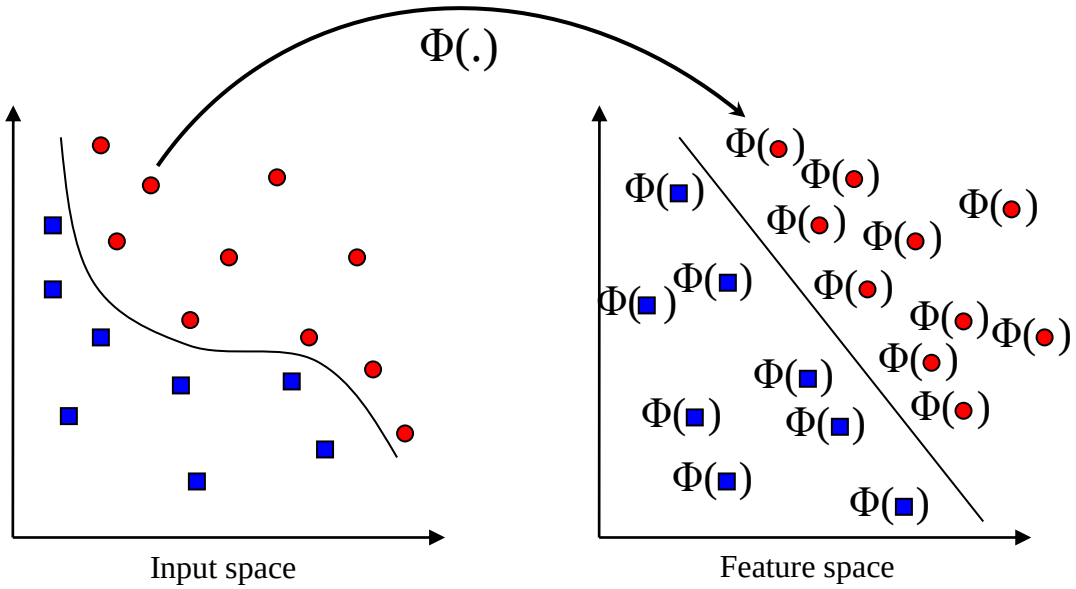


Figure 2.3 Embedding the input data into a feature space, using a mapping function Φ, thereby solving the non-linearly separable classification problem by mapping the input data into a higher-dimensional feature space in which it becomes linearly separable. The kernel function *k* computes the inner products of the mapped data points in the feature space directly from the original input data points. The two classes of the training points shown are indicated by the red circles and the blue squares.

As such, kernel functions constitute an important concept throughout the kernel methods literature [1, 9, 18, 49]. They have been extensively researched to examine their properties, investigate what algorithms can take advantage of them, and their use in general pattern recognition applications. The main reason for such large attention to kernel functions is because they make possible the use of feature spaces with an exponential or even infinite number of dimensions, something that would seem impossible if a reasonable efficiency requirement is to be satisfied [1].

2.3.1 Kernels as similarity measures

The notion of inner products alluded to in the above section constitutes an important aspect in the identification of kernels. To formalize the presentation of SVM kernels, consider the problem of supervised learning where the input space is denoted by X and the output target domain is denoted by Y. If X is a vector space, it is usually a subset of $\Re^m$, a non-empty set from which the input data is taken; i.e.,

$$X \subseteq \Re^m$$

where the input vectors $\mathbf{x}_i$ are given as m-dimensional row vectors, each denoted as

$$\mathbf{x}_i = (x_1, x_2, \ldots, x_m),$$

where m is the number of features. Each row vector in the input space X represents one of the input instances (also sometimes referred to as cases, inputs, observations, or patterns[3]).

For a supervised learning task, the training set is usually denoted by

$$S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_l, y_l)\} \subseteq (X \times Y)^l, \tag{2.23}$$

where l is the number of the training examples $\mathbf{x}_i$ and y_i are the class labels or targets. For unsupervised learning, this simplifies to

$$S = \{\mathbf{x}_1, \ldots, \mathbf{x}_l\} \subseteq X^l. \tag{2.24}$$

The problem of supervised learning aims to produce a function

$$f : X \rightarrow Y,$$

given the dataset S that can be used to predict a value y given an $\mathbf{x} \in X$. In statistical terms, this function f is usually referred to as the *estimator*, *hypothesis*, or more explicitly the *classifier*. Without delving into any further statistical analysis, this is usually approached in two steps. First, a set of possible candidate hypotheses f is defined. This set is usually referred to as the *hypothesis space* $\mathcal{F}$. Afterwards, using a defined error criterion, the best or optimum candidate is selected from within the hypothesis space $\mathcal{F}$.

[3] A large part of the existing literature uses the term 'pattern' to refer to the individual observations. Others, however, reserve it to refer to the sought after regularities that could be present in the data. The former might be the more commonly used in the field of machine learning, however, the latter is probably closer to the meaning of the term. For this reason, this thesis will therefore endeavor to stick to the latter meaning.

In the simple case of binary classification, there are only two classes and the targets of the training set can simply be represented as $y_i = \{\pm 1\}$. The problem of pattern classification, however, requires the generalization to unseen data points. This means that given a new instance $\mathbf{x} \in X$, the aim is to predict the corresponding label *y*. Loosely speaking, this means that *y* should be chosen such that both **x** and its corresponding label *y* are in some sense *similar* to the data points and their corresponding labels of one of the classes in the training examples [9].

It is therefore important to formalise some notions of 'similarity' both in the input *X* and in the output *Y*. Characterizing similarity for the outputs $y_i = \{\pm 1\}$ is usually an easy task: in binary classification, for example, only two situations can occur: two labels can either be identical or different. The choice of a similarity measure for the inputs, however, is a rather challenging task and lies at the core of the kernel methods field.

In the context of statistical pattern recognition, for example, where the input data domain *X* is formulated as a vector space of the input observations, a '*similarity measure*' would typically be characterized as a function that, given two instances **x** and **z**, returns a real number characterizing how they resemble each other [9]. That is

$k : X \times X \rightarrow \Re$

$(\mathbf{x},\mathbf{z}) \rightarrow k(\mathbf{x},\mathbf{z})$.

Based on such a definition, probably the simplest similarity measure in this case would be the linear *dot-product* between the two input vectors **x** and **z** given by [9]:

$$\langle \mathbf{x}, \mathbf{z} \rangle = \mathbf{x}^T \mathbf{z} = \sum_{i=1}^{n} x_i \, z_i \qquad (2.25)$$

The dot-product here means the projection of **x** onto **z**, multiplied by the magnitude of **z**. In other words, how much overlap do **x** and **z** have in their feature space; i.e., how similar they are to each other.

Recall that the key role of a kernel function is to calculate the pairwise dot-products of the embedded data points in the transformed high-dimensional feature space (as per Definition 2.1 below), therefore, intuitively, kernels indeed also do arise as generalised similarity measure functions because of their definition as dot-products in the so-called feature space [9, 12, 49].

Definition 2.1 [1, 9] A kernel k is a function that for all vectors $\mathbf{x}, \mathbf{z} \in X$ satisfies

$$k(\mathbf{x},\mathbf{z}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{z}) \rangle \tag{2.26}$$

where Φ is a mapping function from the m-dimensional input space X to a higher N-dimensional feature space F, denoted as

$$\Phi : \mathbf{x} \in \Re^m \mapsto \Phi(\mathbf{x}) \in F \subseteq \Re^N \tag{2.27}$$

where the choice of the map Φ aims to convert the non-linear relations into linear ones.

Formally speaking, a kernel k is a function that takes two inputs $\mathbf{x}$ and $\mathbf{z} \in \Re^m$ and produces a real number indicative of how similar they are; i.e., k: $\Re^m \times \Re^m \rightarrow \Re$. As such, if one is looking to solve a particular pattern recognition task, say image classification, then a typical kernel should assign a high score to a pair of images that contain the same object, and a low score to a pair of images with different objects. Similarly, in text processing tasks, for instance, a good kernel would assign a high score to a pair of similar strings and a low score to a pair of dissimilar strings. This is basically what a kernel function is, however, in order to be used in most common machine learning algorithms, such as the SVM under study, there is an additional mathematical condition (explained in Section 2.4.2) that also needs to be met by the kernel so that these algorithms can work properly. There are a number of equivalent methodologies that can be used to check whether or not this condition is met, and, hence, whether a given candidate function is a valid kernel; i.e., it corresponds to an inner-product in some high-dimensional feature space, as illustrated in the above definition.

2.3.2 Distances and similarity measures in learning algorithms

2.3.2.1 Why the need for distance and similarity measures?

A brief review of distance and similarity measures in data analysis has been recently conducted by Abou-Moustafa [30]. The author presented a good systematic high-level explanation about the need for *learning algorithms* that can process *digitally recorded data*, in whichever form these could be (e.g., text, speech, images, video, etc.). That is, to automatically (i.e., using machines) extract some knowledge and meaningful information

from such data that could have been otherwise very difficult, or even impossible, to be extracted by humans due to its size and/or complexity. Weather prediction, finger print recognition, analysis of genomic data are just few examples of such learning tasks that cannot be performed by humans without the help of computing machines that can learn from experience.

To enable such a 'machined' learning approach, various learning algorithms have historically relied on some notion of distance (or similarity) between the objects of the data set, due to their robust ability in revealing the natural groupings, structure, and patterns hidden in the data that are typical sought after 'goals' by the underpinning learning algorithm. The appropriate selection, construction or customization of a faithful distance (or similarity) measure for the learning task at hand can improve the effectiveness and performance of the employed learning algorithm. As such, when using such learning algorithms for a particular problem, the algorithm designer, or the practitioner, is usually faced with the question of: What could be a suitable faithful distance (or similarity) measure for the dataset at hand? A major part of the research in this area is dedicated to answer this question.

2.3.2.2 What are distance and similarity measures?

These two terms ('distance' and 'similarity' measures) carry common notions regarding how two different objects are 'distinct from' or 'resemble' each other. As defined by Abou-Moustafa [30], a *distance function* is a measure of the difference (or dissimilarity) between two objects from the same dataset. The larger the distance, the more different the two objects are (see illustrative simple example shown in Figure 2.4). As such, one would expect a suitable distance function to be minimized (ideally to be zero) when the two objects are identical, and increased as they get different (i.e., further apart) from each other.

A *similarity function*, on the other hand, measures the resemblance between two objects from the same dataset [30]. The larger the similarity, the more alike the two objects are. To reflect this meaning (which is converse to its 'distance function' counterpart), one would expect a good similarity function to be maximized when the two objects are identical and decreased (monotonically) when they differ from each other.

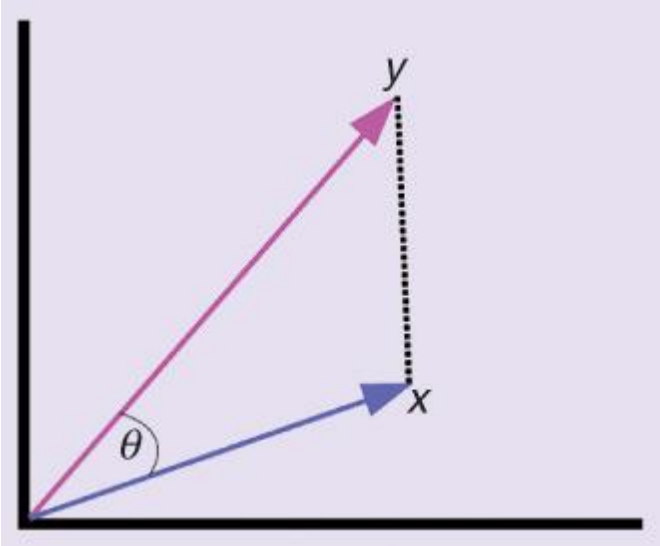


Figure 2.4 Distance versus similarity. An illustrative example of two data vectors *x* and *y*. The *distance* between them can be measured using the Euclidean distance (the dotted line), whereas the similarity can be measured using the dot-product operation [30].

2.3.2.3 Off-the-shelf distance and similarity measures

If the data are in the form of vectors or matrices, or have been transformed by feature extraction methods to such structures, then various off-the-shelf distance and similarity measures can be used [30, 50]. For example: Minkowski distances, the Mahalanobis distance, Matsushita distances, the chi-squared distance, the Hamming distance, cosine similarity, dot-products, kernels, angles between subspaces, and the Grassmann distance.

2.3.2.4 Adaptive distance functions

Although such off-the-shelf pre-defined distance and similarity functions have demonstrated reasonable success for various applications to date, they are not expected to be suitable for all data types. In other words, a pre-defined distance (or similarity) function will need to be adapted to the dataset at hand so that it can best capture its intrinsic structure and any patterns hidden in it [30]. A number of data-specific and data-dependent approaches have emerged (usually known as adaptive distance functions) to tailor the employed distance or similarity function to the data under consideration.

Examples of such adaptive distance functions approaches are the dynamic programming-based distances, such as the dynamic time wrapping (DTW) distance for time series and sequential data, developed originally for speech recognition [30]. A number of methods have also been proposed to adapt the metric rule of the *k*-nearest neighbour pattern classification algorithm, such as the flexible metric method by Friedman [51], the

discriminant adaptive method developed by Hastie and Tibshirani [52], the locally adaptive metric method by Domeniconi et al. [53], and the extremely simple adaptive distance measure by Wang et al. [54].

2.3.2.5 Metric learning algorithms

Handcrafting a good distance or similarity measure for a specific problem is generally a difficult task. This has led to the emergence of an alternative approach, commonly known as 'metric learning' that aims at automatically learning a metric from the data and has attracted the attention of many researchers from within the machine learning (and other related fields) community.

Metric learning directly addresses the problem of learning the appropriate distance or similarity metric from the dataset at hand rather than adapting a pre-defined metric for it. A good survey about these modern algorithms can be found in [55-57]. Within the kernel methods community, the concept of kernel-alignment (originally introduced by [58]) and the multiple kernel learning (MKL) technique [59], explained later in Section 2.6.3, can be categorized under this metric learning umbrella.

2.3.3 SVM: a similarity-based classifier using kernels

Referring back to the dual optimization problem in (2.12), one can now realize that the optimum value for α will certainly depend on the similarity measures calculated by the dot-products of all the pairs of the training data points $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$, which consequently affects the decision being made in (2.14) during the test phase, where the similarities between the test sample $\mathbf{x}$ and the set of labelled support vectors are also being measured in $\langle \mathbf{x}, \mathbf{x}_i \rangle$. Replacing the dot-product by the kernel in the kernelized SVM in (2.19) and (2.21), to implicitly transform the input space to a higher-dimensional feature space, so that linear separability between overlapped classes can be improved, will certainly give us a lot more choices (than the crude dot-product), by which the similarity between the inputs can be measured, hence enable us to address a lot more of the challenging and complex classification tasks that were before very difficult or even impossible to solve using the traditional linear approaches. However, within the context of its definition as a similarity measure, as explained earlier, a good kernel choice should still produce a high score, when its two inputs are similar, and a low score, when they are different.

As described by Chen et al. [19], the SVM classifier is therefore said to estimate the class label of a test sample based on:

1- The pairwise similarities between the training samples calculated by the underpinning kernel, which is a key factor in determining the optimum model parameters when solving the dual optimization problem in (2.19) during the training phase, and

2- The similarities between the test sample and the labelled set of support vectors in (2.21), calculated by the same kernel that was used in the training phase.

In fact, similarity-based classification has been shown to be useful in solving a variety of problems in computer vision, bioinformatics, information retrieval, natural language processing, amongst others [19]. A brief review of how similarity functions were adopted within similarity-based classification can be found in [19]. Crafting an appropriate SVM kernel, based on some intuitive and accurate similarity-based measure criteria, is however discussed in more detail in Chapter 3 together with the consequent implication on its resulting SVM classification performance.

2.3.4 Non-linear feature mapping using the kernel trick

The idea behind the kernel trick is that the inner products between the mapped vectors in the feature space can be computed more efficiently (i.e., a lot cheaper in terms of computation complexity) as a direct function from the input space [7, 9, 60]. This means that it is not even required to explicitly compute the mapping Φ. This is what is basically achieved by applying a kernel function in the form shown in (2.26). In other words, the step of explicitly constructing a feature vector space representation can be by-passed using kernel functions. Appendix A1 shows a simple toy example (reproduced from [9]), which illustrates how the inner products in the feature space can be effectively calculated directly from the input space by means of a suitably chosen kernel. It also shows how the complexity of the kernel function can be a lot less than the dimension of its corresponding feature space H.

In fact, the range of valid kernels is very large. Some are given in closed forms; others can only be computed by means of a recursion or other algorithms. Surprisingly, in most cases, the actual feature mapping corresponding to a given kernel function is even not known; only a guarantee that the data can be embedded in some feature space that gives rise to the chosen kernel. Therefore, provided the function can be evaluated efficiently and it

corresponds to computing the inner products of suitable images of its two arguments, it constitutes a potentially useful kernel [1, 9]. Selecting the best kernel from amongst this extensive range of possibilities is a critical stage in applying kernel-based algorithms (such as SVM) in practice. In most cases, such selection usually relies on our prior knowledge about the data and the types of patterns that we can expect to identify [1].

Hence, if a learning algorithm can be adapted to use only inner products between inputs, it can therefore be combined with a kernel function (sometimes referred to as the algorithm being 'kernelized') that calculates the inner product between the projections of two inputs in a feature space. Hence, making it possible to implement the algorithm in a high-dimensional feature space. This is how the SVM took advantage of this approach as it has been previously explained in Section 2.2.4.

2.4 Properties of Kernels

Within the generic setting of kernel-based algorithms, kernel functions provide a powerful and principled way of classifying non-linearly separable patterns using a theoretically well understood linear algorithms in an appropriate feature space. This section provides a brief study about the fundamental properties that characterise kernel functions. However, as indicated earlier, it should be noted that the prior knowledge and experience also play an important role in kernel-based learning machines; i.e., kernels must be chosen for the problem at hand with a view to capture our prior belief of the potential regularities/patterns in the data, showing that a universal machine is not (yet) possible.

2.4.1 Inner products and the Hilbert space

The kernel-based approach to pattern recognition embeds the data into a high dimensional feature space whereby patterns can become linearly separable and hyperplane functions are equivalent to non-linear functions in the input space. The use of kernels enables this technique to be applied without paying the computational penalty implicit in the number of dimensions of the feature space, since it is possible to evaluate the inner product between the projections of two inputs in a feature space without explicitly computing their coordinates.

This means that pattern recognition algorithms can be applied to the projections of the training data in the feature space through the indirect evaluation of the inner products. A

function that returns the inner product between the images of two inputs in some feature space is the kernel function. It is therefore obvious that the notion and properties of inner products play an important part in the characterization of kernel functions and in the verification that a potential candidate function can indeed be a valid kernel.

Definition 2.2 An *inner product space* can be defined as a vector space X over the reals $\Re$ if there exists a real-valued symmetric bilinear map $\langle \cdot, \cdot \rangle$ that satisfies [1]:

$\langle \mathbf{x}, \mathbf{x} \rangle \geq 0$.

This bilinear map is known as the inner product; and for the vector space $\Re^m$, the standard inner product is given by:

$$\langle \mathbf{x}, \mathbf{z} \rangle = \sum_{i=1}^{m} x_i z_i \tag{2.28}$$

Inner product spaces are also known as L_2 spaces, where p=2 in the following generic norm form:

$$\|\mathbf{x}\|_p = \left[\sum_{i=1}^{n} x_i^2 \right]^{\frac{1}{p}} \tag{2.29}$$

If, however, $p \neq 2$, then the space is a normed space, but not an inner product space, because this norm does not satisfy the parallelogram equality required of a norm to have an inner product associated with it.

Definition 2.3 An inner product space is usually arbitrary referred to as a *Hilbert space H*, although strictly speaking a Hilbert space requires the additional statistical properties of being *complete* (defined by the Cauchy sequence property) and *separable* (which ensures that the minimum difference between the space elements is always non-negative) [1].

Without delving into unnecessary analytical explanation of these properties, the important point to consider here is that these two properties ensure that the transformed inner product feature space can be given a coordinate system [1]. However, as the kernel implicitly defines such a space, there will be no need to construct the feature vectors.

2.4.2 Characterisation of kernel functions

As there is no need to explicitly construct the feature space (via the utilization of the kernel trick) to compute the inner products between the images of the mapped data, there is a necessary requirement for creating kernels without explicitly constructing this feature space. In other words, we need to have some sort of guarantee that if we used a certain function to directly process the input data, then this will indeed be equivalent to their inner products in some high-dimensional feature space. The traditional way that has been shown so far to verify that a candidate function is a kernel, is that to construct a feature space for which the function corresponds to first performing the feature mapping and then computing the inner product between the two images [1].

However, there are few other alternatives available in the literature of kernel methods [1, 9]. These are equivalent methods of demonstrating that a candidate function is a valid kernel. Such methods not only can be used to verify that a function is a kernel, but they can also provide solid theoretical tools to create new kernels for complex pattern recognition applications. Some of these methods are briefly described in the following sub-sections.

2.4.2.1 Gram and kernel matrix

In linear algebra, given a non-empty set of vectors, $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l\}$ in an inner product space, the *Gram matrix* **G** is defined as the $l \times l$ Hermitian matrix of inner products whose entries are $\mathbf{G}_{ij} = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$ [1], given by:

$$\mathbf{G} = \begin{bmatrix} \langle \mathbf{x}_1, \mathbf{x}_1 \rangle & \langle \mathbf{x}_1, \mathbf{x}_2 \rangle & \cdots & \langle \mathbf{x}_1, \mathbf{x}_l \rangle \\ \langle \mathbf{x}_2, \mathbf{x}_1 \rangle & \langle \mathbf{x}_2, \mathbf{x}_2 \rangle & \cdots & \langle \mathbf{x}_2, \mathbf{x}_l \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \mathbf{x}_l, \mathbf{x}_1 \rangle & \langle \mathbf{x}_l, \mathbf{x}_2 \rangle & \cdots & \langle \mathbf{x}_l, \mathbf{x}_l \rangle \end{bmatrix}$$

In the kernel methods context, where a kernel function k is used to evaluate the inner products in a feature space with the feature map Φ, the entries of the Gram matrix will be the evaluation of the kernel function on each pair of vectors in the dataset of the input space as:

$$\mathbf{G}_{ij} = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle = k(\mathbf{x}_i, \mathbf{x}_j). \tag{2.30}$$

In this case, the matrix is referred to as the *kernel matrix* denoted by **K** as follows

$\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$, for $i, j = 1, 2, \ldots, l$.

$$\mathbf{K}_{ij} = \begin{bmatrix} \langle \Phi(\mathbf{x}_1), \Phi(\mathbf{x}_1) \rangle & \langle \Phi(\mathbf{x}_1), \Phi(\mathbf{x}_2) \rangle & \cdots & \langle \Phi(\mathbf{x}_1), \Phi(\mathbf{x}_l) \rangle \\ \langle \Phi(\mathbf{x}_2), \Phi(\mathbf{x}_1) \rangle & \langle \Phi(\mathbf{x}_2), \Phi(\mathbf{x}_2) \rangle & \cdots & \langle \Phi(\mathbf{x}_2), \Phi(\mathbf{x}_l) \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \Phi(\mathbf{x}_l), \Phi(\mathbf{x}_1) \rangle & \langle \Phi(\mathbf{x}_l), \Phi(\mathbf{x}_2) \rangle & \cdots & \langle \Phi(\mathbf{x}_l), \Phi(\mathbf{x}_l) \rangle \end{bmatrix}$$

$$\mathbf{K}_{ij} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & \cdots & k(\mathbf{x}_1, \mathbf{x}_l) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & \cdots & k(\mathbf{x}_2, \mathbf{x}_l) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_l, \mathbf{x}_1) & k(\mathbf{x}_l, \mathbf{x}_2) & \cdots & k(\mathbf{x}_l, \mathbf{x}_l) \end{bmatrix} \tag{2.31}$$

Appendix A2 demonstrates a numerical example to illustrate how such a kernel is calculated using a small artificial dataset.

As illustrated by (2.31), the kernel matrix contains all the information available in order to perform the learning step, with the sole exception of the output labels in the case of supervised learning [1]. In other words, all the information the pattern recognition algorithm can glean about the training data and chosen feature space is contained in the kernel matrix together with any labelling information available. It is therefore important to bear in mind that it is only through the kernel matrix that the learning algorithm obtains information about the feature space and the input training data itself.

In this view, it is perhaps not surprising that some properties of this matrix can be used to assess the generalisation performance of the learning system. These properties, however, vary according to the type of the learning task and the subtlety of the pattern recognition process. However, in general, it plays a central role both in the derivation of generalisation bounds and their evaluation in practical applications [1].

The kernel matrix is, therefore, viewed as the information bottleneck that must transmit enough information about the data for the algorithm to be able to perform its task. Accordingly, it is natural to analyse the properties of these matrices, how they are created, how they can be adapted, and how well they are matched to the task being addressed.

2.4.2.2 Kernel matrix and the finitely positive semi-definite property

The evaluation of the kernel function on the pairs of vectors of an input training dataset of size l produces an $l \times l$ kernel matrix as shown in (2.31). The kernel matrix is a *square* matrix of the same size l as the number of examples in the dataset.

The kernel matrix is also *symmetric* since $\mathbf{K}_{ij} = \mathbf{K}_{ji}$, that is $\mathbf{K}^{\mathrm{T}} = \mathbf{K}$ where the (i, j) entry equals the (j, i) entry, for all i and j, where $\mathbf{K}^{\mathrm{T}}$ is the transpose of $\mathbf{K}$. Another important property that underpins the characterisation of both kernel matrices and kernel functions is what is known as the *finitely positive semi-definite* (PSD) property. A matrix $\mathbf{K}$ can be shown to be PSD in a number of ways [1]; however, below we only highlight the most common.

Definition 2.4 Positive semi-definite matrices [1, 9]

A real symmetric $l \times l$ matrix $\mathbf{K}$ is *positive semi-definite* if and only if its eigenvalues λ's are all non-negative (i.e., greater than or equal 0). This condition can only be achieved if and only if

$$\mathbf{a}^{\mathrm{T}} \mathbf{K} \mathbf{a} \geq 0 \tag{2.32}$$

for all vectors $\mathbf{a} \in \Re^m$.

Similarly, a real symmetric $l \times l$ matrix $\mathbf{K}$ is *positive definite*, if its eigenvalues are positive, or equivalently,

$$\mathbf{a}^{\mathrm{T}} \mathbf{K} \mathbf{a} > 0 \tag{2.33}$$

for $\mathbf{a} \neq 0$.

As such, a matrix can be checked for this property using either the former (eigenvalues) condition or the latter (inequality) condition, whichever is easier to apply to the problem at hand.

Definition 2.5 Eigenvalues and eigenvectors [1]

Given a matrix $\mathbf{K}$, the real number λ and the vector $\mathbf{x}$ are the eigenvalue and the corresponding eigenvector of $\mathbf{K}$ if

$$\mathbf{K} \mathbf{x} = \lambda \mathbf{x} \tag{2.34}$$

The eigenvalues of $\mathbf{K}$ are determined by solving the determinant equation:

$$\det(\mathbf{K} - \lambda \mathbf{I}) = 0 \tag{2.35}$$

where **I** is the $l \times l$ identity matrix

$$\mathbf{I}_l = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \tag{2.36}$$

Investigating this positive semi-definite property of symmetric matrices into kernel matrices has revealed the following important proposition.

Proposition 2.1 Gram and kernel matrices are positive semi-definite. This can be shown as follows [9]:

Recalling that

$\mathbf{K}_{ij} = k(\mathbf{x}_i,\mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$, for $i, j = 1, \ldots, l$,

Therefore, for any vector $\mathbf{a} = \{a_1, a_2, \ldots, a_m\}$ we have:

$$\begin{aligned} \mathbf{a}^{\mathrm{T}}\,\mathbf{K}\,\mathbf{a} &= \sum_{i,j=1}^{m} a_i\, a_j\, \mathbf{K}_{ij} \\ &= \sum_{i,j=1}^{m} a_i\, a_j \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle \\ &= \left\langle \sum_{i=1}^{m} a_i\, \Phi(\mathbf{x}_i), \sum_{j=1}^{l} a_j\, \Phi(\mathbf{x}_j) \right\rangle \\ &= \left\| \sum_{i=1}^{m} a_i\, \Phi(\mathbf{x}_i) \right\|^2 \geq 0 \end{aligned}$$

Because of the great analytical appeal that this proposition had on the theory of kernels, Appendix A3 shows a simple example that aims to clarify how it could be applied.

2.4.2.3 Finitely positive semi-definite functions

The pairwise evaluation of a valid kernel function on a finite set of points gives rise to a positive semi-definite matrix, as shown in the above sub-section. It can therefore be clearly observed how the kernel function and its corresponding kernel matrix are intimately related. Such a relation is what enables the formalization of an alternative method of

characterising and verifying that a candidate function is a valid kernel. In other words, the kernel matrix formed by evaluating a valid kernel on all pairs of any set of points is positive semi-definite [1]. This can be formally summarised as follows.

Proposition 2.2 Finitely positive semi-definite functions [1]

A function

$$k : X \times X \rightarrow \Re$$

satisfies the finitely positive semi-definite property if it is a symmetric function for which the matrices formed by evaluating it on any finite subset $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l\}$ of the space X are positive semi-definite [1].

It is therefore important to ensure that the chosen candidate function is characterised by the positive semi-definite property, for it to be a valid kernel. As such, if the function is PSD, it can be used to calculate dot-products in the feature space H. This is what can briefly be summarised in the following proposition.

Proposition 2.3 Characterisation of kernels [1]

A function

$$k : X \times X \rightarrow \Re$$

which is either continuous or has a finite domain, can be decomposed into dot products

$k(\mathbf{x}, \mathbf{z}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{z}) \rangle$

using an implicitly defined feature map Φ within a feature space H, applied to both of its arguments followed by the evaluation of the inner products in H, if and only if it satisfies the finitely positive semi-definite property [1].

Appendix A4 demonstrates a simple example that shows how to apply this proposition to verify that a candidate function is a valid kernel.

It can therefore be summarised that the relation

$k(\mathbf{x}, \mathbf{z}) = \langle \Phi(\mathbf{x}), \Phi(\mathbf{z}) \rangle$

means that, given a PSD function k, there exists a mapping function Φ such that the evaluation of the kernel on vectors $\mathbf{x}$ and $\mathbf{z}$ is equivalent to calculating the dot-product between $\Phi(\mathbf{x})$ and $\Phi(\mathbf{z})$ in some (perhaps unknown) Hilbert space.

In the literature of kernel methods, however, a number of additional (but equivalent) analytical methods have emerged to verify that a candidate function meets the PSD property and therefore is a valid kernel. The most common of which are known as the *Reproducing Kernel Map* and the *Mercer condition* [1, 9, 61-64]. These are briefly summarised in the following sub-sections.

2.4.2.4 The reproducing kernel map

This method works by a reverse engineering approach. It assumes that k satisfies the PSD property and proceeds to explicitly construct its corresponding space of functions $\boldsymbol{F}_k$, which then should be an inner product space in which its elements satisfy the real-valued symmetric and bilinear properties [1, 9].

In this sense, given a function k that satisfies the PSD property, its corresponding function space $\boldsymbol{F}_k$ is usually referred to as the Reproducing Kernel Hilbert Space (RKHS) H and the function that produced it is the Reproducing kernel. Formally speaking, if a symmetric function $k(\cdot,\cdot)$ satisfies the reproducing property in a Hilbert space of functions $\boldsymbol{F}$

$$\left\langle k(\mathbf{x},\cdot), f(\cdot)\right\rangle_F = f(\mathbf{x}),$$

then k satisfies the finitely positive semi-definite property.

2.4.2.5 Mercer's condition

The statistical analysis of the reproducing map described in the previous sub-section shows that any PSD kernel can be represented as an inner product in a linear space by explicitly constructing a (Hilbert) space that does the job. This sub-section, however, briefly describes another tool, based on the Mercer's theorem, which essentially constructs another Hilbert space for a valid kernel that is in fact defined by a one-to-one relation to the RKHS [65]. This tool has played a crucial role in the understanding of many kernel-based learning algorithms, such as SVMs, and provides valuable insight into the geometry of feature spaces. Instead of delving into intensive analytical explanation of the construction of the Mercer's map and its corresponding Hilbert space, this section will just confine itself to stating the theorem and explaining how it can be used to construct useful kernels.

The Mercer's theorem itself dates back to 1909 when Mercer defined the general form of inner products in Hilbert spaces [5]. This theorem states that the general form of the inner product in a Hilbert space is defined by the symmetric positive definite function $k(\mathbf{x}, \mathbf{z})$ that satisfies the following:

$$\iint k(\mathbf{x},\mathbf{z})\, f(\mathbf{x})\, f(\mathbf{z})\, d\mathbf{x}\, d\mathbf{z} \geq 0\,, \tag{2.37}$$

for all functions $f(\mathbf{x})$ and $f(\mathbf{z})$ satisfying the inequality [5]

$$\int f^{2}(\mathbf{x})\, d\mathbf{x} \leq \infty\,.$$

To be a valid SVM kernel, for any finite function $f(\mathbf{x})$, the condition in (2.37) should always be satisfied for the given kernel function $k(\mathbf{x}, \mathbf{z})$ [9, 36] . If the kernel does not satisfy this Mercer condition, the SVM quadratic programming optimization problem may not find optimal parameters, but rather it may find suboptimal parameters [17]. Also, if the Mercer condition is not satisfied, the kernel matrix in turn may also not be PSD.

At the same time, any function $k(\mathbf{x}, \mathbf{z})$ satisfying the above Mercer's condition is a valid kernel and can be used to construct non-linear decision functions in the input space that are equivalent to constructing optimum separating hyperplanes in some linear high-dimensional feature space. Furthermore, it has also been shown that Mercer kernels and positive definite kernels can both be represented as inner products in Hilbert spaces, and therefore can be considered equivalent [9]. This means that the Mercer's theorem is an equivalent formulation of the finitely positive semi-definite property for vector spaces [9].

2.4.3 Legitimate operations on kernels

The analysis provided in Section 2.4.2 showed that the function $k(\mathbf{x}, \mathbf{z})$ is a valid kernel provided that its kernel matrix is positive semi-definite for all training sets S, the so-called finitely semi-definite property [1, 9]. This fact enables the manipulation of kernels without necessarily considering the corresponding feature space. Provided that the finitely positive semi-definite property is maintained, it is guaranteed that we have a valid kernel. That is, there exists a feature space for which k is the corresponding kernel function. The intrinsic modularity of kernel machines also means that any kernel function can be used provided it produces symmetric positive semi-definite kernel matrices. Similarly, any kernel-based algorithm can be applied as long as it can accept as input such a kernel matrix together with any labelling information.

The characterisation of kernel functions and kernel matrices is useful not only for deciding whether a given candidate function is a valid kernel. One of its main consequences is that it can also be used to justify a series of rules for manipulating and combining simple kernels to obtain more complex and useful ones. Such operations on one or more kernels can be shown to preserve the finitely positive semi-definiteness property. These also can include operations on both the kernel functions and the kernel matrix. In other words, as long as we can guarantee that the result of an operation will always be a positive semi-definite symmetric matrix, we will still be embedding the data in some inner product feature space, as required.

An example of creating a new kernel from an existing one is provided by normalizing a kernel. This has the effect of normalizing the feature space, a process that is usually used to take place at the pre-processing step. Given a kernel $k(\mathbf{x},\mathbf{z})$ that corresponds to the feature mapping Φ, the normalized kernel k' is given as:

$$k'(\mathbf{x},\mathbf{z}) = \frac{k(\mathbf{x},\mathbf{z})}{\sqrt{k(\mathbf{x},\mathbf{x})\,k(\mathbf{z},\mathbf{z})}}. \qquad (2.38)$$

Similarly, operations that manipulate the kernel matrix using the finitely positive semi-definite property can also be viewed as an intermediate processing step designed to improve the representation of the data, and, hence, the overall performance of the system, before it is passed to the learning algorithm. One simple example is the addition of a constant to the diagonal of the kernel matrix. This has the effect of introducing a soft margin in classification or equivalently regularisation in regression.

Therefore, using a family of simple operations, usually referred to as the *closure properties* [1], more complicated kernels can be created from simple building block kernels. The approach demonstrates that the new functions are kernels by showing that they are finitely positive semi-definite. This is sufficient to verify that the function is a kernel and to sculpt an appropriate new kernel for a particular application. A full list of the legitimate kernel operations, together with their proofs, can be found in [1] and [9]. Below, however, is just a highlight of the main operations that are utilized in this thesis.

Given that $k_1(\mathbf{x}, \mathbf{z})$ and $k_2(\mathbf{x}, \mathbf{z})$ are two valid kernels over the space $S = X \times X$, where $X \subseteq \Re^m$ and a is a positive real constant $a \in \Re^+$, the following propositions hold true to produce new valid kernel functions [1].

Proposition 2.4 The multiplication of two valid kernels is also a valid kernel; i.e.,

$k(\mathbf{x}, \mathbf{z}) = k_1(\mathbf{x}, \mathbf{z}) \times k_2(\mathbf{x}, \mathbf{z})$

is also a valid kernel. Such combination of kernels is often referred to as the *Schur product* which is obtained by getting the tensor product of each of the kernel functions' matrices, and hence the result is also a positive semi-definite matrix.

Proposition 2.5 The addition of two (or more) valid kernels is also a valid kernel; i.e.,

$k(\mathbf{x}, \mathbf{z}) = k_1(\mathbf{x}, \mathbf{z}) + k_2(\mathbf{x}, \mathbf{z})$

is also a valid kernel, which is probably the simplest and most natural operation one can think of to combine two existing kernels. Similar to the previous proposition, this again stems from the fact that the addition of the respective kernel matrices of the combined kernels also preserves the positive semi-definite property.

Proposition 2.6 The multiplication of a positive real constant by a valid kernel also produces a valid kernel; i.e.,

$k(\mathbf{x}, \mathbf{z}) = a\, k_1(\mathbf{x}, \mathbf{z})$

is also a valid kernel, which also preserves the positive semi-definite property, as long as $a \in \Re^+$.

2.5 Selection of the right kernel

2.5.1 The problem of kernel function selection

The kernel function is a central and crucial element in the design of a kernel-based learning algorithm. It needs to be carefully selected to perform the mapping into the correct feature space, such that the sought after linearly separable classes can be formulated. But how can kernel functions be selected? What function should one choose in order to achieve the best possible representation of the data for a given problem? These questions have brought up the problem of 'kernel function selection', which, to some extent, still remains a mystery.

It has already been demonstrated in Section 2.4.2 that kernel functions need to be positive semi-definite to enable the implicit establishment of a high-dimensional feature space, for which the function computes the corresponding inner products. However, there are many

functions that can be shown to preserve this property, so which one is the best? Is there an 'ideal' kernel that can be used to solve any pattern classification problem?

In fact, the choice of the kernel is an open problem that usually amounts to using our prior expectation about the possible patterns hidden in the data that the algorithm is expected to learn. As it is not always possible to make the right expectations a priori, it is a common practice to search, from within a family of kernels, for the best possible kernel that can achieve the best possible representation of the dataset in the transformed feature space. Ideally, this is usually empirically inferred from the performance of the adopted kernel-based pattern recognition system (such as SVM) on a particular dataset.

The choice of the most appropriate kernel is highly dependent on the problem at hand, since it depends on the nature of the patterns or the kind of information we are expecting to extract from the data. A polynomial kernel, for example, allows us to model feature conjunctions up to the order of the polynomial. The Gaussian kernel function, on the other hand, allows to pick out circles (or hyperspheres), in contrast to the linear kernel, which only allows to pick out hyperplanes or lines [15]. This means that not all kernels have the same data representation power.

As mentioned earlier, the range of valid kernels is very large, especially if we thought about all the different permutations and operations discussed earlier that can be used to create new composite kernels from smaller kernel building blocks. Below is a list of the most commonly used and traditional kernel functions that are available from the existing literature [1, 12, 15].

- **Linear kernel**

This is the simplest kernel function given in a dot-product form as:

$$k(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle . \qquad (2.39)$$

It does not, however, amount to any transformation to higher number of dimensions, and, hence, it does not help with enhancing linear separability between non-linearly separable classes. However, if the classification problem is already linearly separable in the input space, there is no need to map the input space into a high-dimensional space, and in this case the linear kernel can be used [12].

- **Polynomial kernel**

The polynomial kernel is the immediate generalisation of the linear kernel raised to the polynomial order *n* where $n \in Z^+$. It is usually used in either one of two types: either the *homogeneous* polynomial kernel, given by:

$$k(\mathbf{x}, \mathbf{z}) = \left(\langle \mathbf{x}, \mathbf{z} \rangle\right)^n \tag{2.40}$$

or the *inhomogeneous* polynomial kernel, given by:

$$k(\mathbf{x}, \mathbf{z}) = \left(\langle \mathbf{x}, \mathbf{z} \rangle + c\right)^n \tag{2.41}$$

where *c* is a constant usually equal to 1.

- **Gaussian kernel**

This is probably the most commonly used kernel due to its good empirical performance in picking up non-linear relations [1, 15]. The Gaussian kernel is an example of the radial basis function used in a number of applications, such as neural networks. It is given by:

$$k(\mathbf{x}, \mathbf{z}) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{z}\|^2\right), \tag{2.42}$$

where the kernel parameter $\gamma = \frac{1}{2\sigma^2}$ plays an important role in the performance of the kernel and is usually subject to careful tuning to the problem at hand. If σ is overestimated, the exponential will behave almost linearly, and the higher-dimensional projection will start to lose its non-linear power. Conversely, if σ is underestimated, the function will lack regularization and the decision boundary will be highly sensitive to noise in training data [15].

The widespread use of the Gaussian kernel is probably due to the high dimensionality of its equivalent feature space. The process of constructing the Gaussian kernel stems from the use of the Taylor expansion of the exponential function [1]:

$$\exp(x) = \sum_{i=0}^{\infty} \frac{1}{i!} x^i ,$$

As such, the Gaussian kernel is actually equivalent to a polynomial kernel of infinite degree. Hence, its corresponding feature space is of infinite dimensions with all possible monomials of input features with no restriction placed on the degrees.

- **Exponential kernel**

The exponential kernel is the same as the Gaussian kernel but with the square of the norm left out. It is also a radial basis function kernel, given by [66]

$$k(\mathbf{x}, \mathbf{z}) = \exp\left(-\gamma\|\mathbf{x}-\mathbf{z}\|\right), \tag{2.43}$$

where the kernel parameter γ will also still need to be empirically optimized.

- **Tan Sigmoid kernel**

The hyperbolic Tangent Sigmoid kernel is also a popular SVM kernels given by [15]

$$k(\mathbf{x}, \mathbf{z}) = \tanh\left(\alpha\langle\mathbf{x}, \mathbf{z}\rangle + \theta\right), \tag{2.44}$$

whose two adjustable kernel parameters are the slope α and the intercept constant θ. It originates from neural networks whereby it was often used as an activation function for the artificial neurons. Surprisingly, although the Sigmoid kernel is known to not strictly fulfil Mercer's condition (i.e., Mercer's condition is satisfied only for some values of α and θ) [7, 60], yet, it has been reported to perform well in a number of practical applications. A more detailed insight into the Sigmoid kernels for SVMs can be found in [67].

2.5.2 The statistics perspective of kernel classes

Due to the wide diversity of kernels throughout the different machine learning disciplines, Genton [18] conducted an interesting study on positive definite kernels that aimed to categorize them based on how they are expressed in terms of the two input vector examples **x** and **z**. Table 2.1 shows a summary of the kernel categories investigated, namely: anisotropic stationary kernels, isotropic stationary kernels, compactly supported stationary kernels, locally stationary kernels, non-stationary kernels, and separable non-stationary kernels. Based on this categorization, one can easily identify which class would the SVM kernels mentioned in the previous sub-section belong to. For example, since the Gaussian kernel is expressed in terms of the lag vector $\|\mathbf{x}-\mathbf{z}\|$, it would fall under the isotropic stationary kernels category, whereas the homogeneous polynomial kernel, expressed in terms of $<\mathbf{x}, \mathbf{z}>$, would fall under the non-stationary kernels.

Table 2.1 Classification of kernel functions from the statistics perspective, as introduced in [18].

Stationary kernels			**Locally stationary kernels**	**Non-stationary kernels**	**Separable non-stationary kernels**
Anisotropic	**Isotropic**	**Compactly supported**			
The kernel function depends on both the length and direction of the difference vector between the two input vectors $k(\mathbf{x},\mathbf{z}) = k_S(\mathbf{x}-\mathbf{z})$	The kernel function depends only on the magnitude of the lag vector between the two input vectors and not on the direction, and therefore is only function of distance $k(\mathbf{x},\mathbf{z}) = k_I(\lVert\mathbf{x}-\mathbf{z}\rVert)$	These are kernels that vanish whenever the distance between the two vector inputs is larger than a certain cut-off distance; e.g., the spherical kernel is compactly supported because $k_I(\lVert\mathbf{x}-\mathbf{z}\rVert) = 0$ when $\lVert\mathbf{x}-\mathbf{z}\rVert \geq \theta$	These are kernel functions that are usually expressed in the form of $k(\mathbf{x},\mathbf{z}) = k_1\left(\frac{\mathbf{x}+\mathbf{z}}{2}\right)k_2(\mathbf{x}-\mathbf{z})$ Where k_1 is a non-negative function and k_2 is a stationary kernel. Stationary kernels are a special case of locally stationary kernels if k_1 is a positive constant. Other special cases of locally stationary kernels also include: ▪ The exponentially convex kernel $k(\mathbf{x},\mathbf{z}) = k_1(\mathbf{x}+\mathbf{z})$ ▪ The white noise kernel $k(\mathbf{x},\mathbf{z}) = k_1\left(\frac{\mathbf{x}+\mathbf{z}}{2}\right)\delta(\mathbf{x}-\mathbf{z})$ where δ is the positive definite kernel which is equal to 1 if **x**=**z** and 0 otherwise.	These are the most general class of kernels which depend explicitly on the two input vector examples **x** and **z**. For example, the homogeneous polynomial kernel defined by $k(\mathbf{x},\mathbf{z}) = <\mathbf{x},\mathbf{z}>^n$ where n is the degree of the polynomial kernel.	Is a special case of non-stationary kernels expressed in the form of $k(\mathbf{x},\mathbf{z}) = k_1(\mathbf{x})k_2(\mathbf{z})$ where k_1 and k_2 are stationary kernels evaluated separately on the two input vector examples **x** and **z**, respectively.

Genton's study on the classes of kernels was conducted from a statistics perspective whereby he utilized the spectral representation theory to show how positive definite kernels can be constructed in each class. However, he was still not able to advocate which kernel, or classes of kernels, could be useful to certain applications or best suited to target the solution of a particular problem. Of particular interest to the work of this thesis (as will be discussed later in Chapter 3), however, is the study he conducted on the isotropic stationary kernels (which are explicitly defined in terms of the lag vector $\|\mathbf{x}-\mathbf{z}\|$) and the graphical examples he demonstrated for them, such as the circular, spherical, rational quadratic, exponential, Gaussian, and wave kernels. These are replicated in Figure 2.5 and Table 2.2 below, where θ is the kernel parameter.

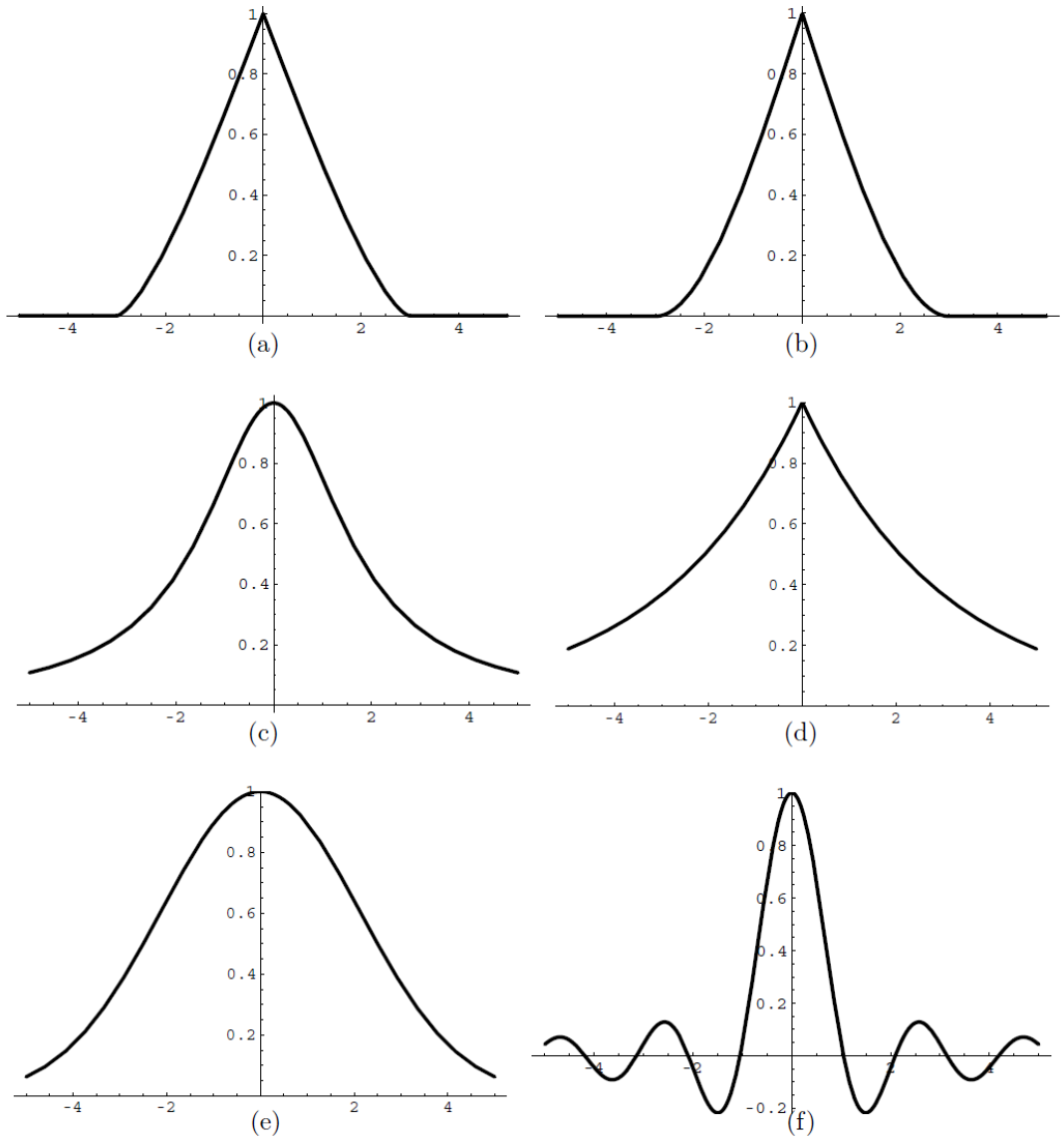


Figure 2.5 Examples of isotropic stationary kernels, as illustrated by [18]: (a) circular; (b) spherical; (c) rational quadratic; (d) exponential; (e) Gaussian; (f) wave.

Table 2.2 Examples of commonly used isotropic stationary kernels, as illustrated by [18].

Name of kernel	$K_I(\|\mathbf{x}-\mathbf{z}\|)/K_I(0)$
(a) **Circular** positive definite in $\mathbb{R}^2$	$\frac{2}{\pi}\arccos\left(\frac{\|\mathbf{x}-\mathbf{z}\|}{\theta}\right) - \frac{2}{\pi}\frac{\|\mathbf{x}-\mathbf{z}\|}{\theta}\sqrt{1-\left(\frac{\|\mathbf{x}-\mathbf{z}\|}{\theta}\right)^2}$ if $\|\mathbf{x}-\mathbf{z}\| < \theta$ zero otherwise
(b) **Spherical** positive definite in $\mathbb{R}^3$	$1 - \frac{3}{2}\frac{\|\mathbf{x}-\mathbf{z}\|}{\theta} + \frac{1}{2}\left(\frac{\|\mathbf{x}-\mathbf{z}\|}{\theta}\right)^3$ if $\|\mathbf{x}-\mathbf{z}\| < \theta$ zero otherwise
(c) **Rational quadratic** positive definite in $\mathbb{R}^d$	$1 - \frac{\|\mathbf{x}-\mathbf{z}\|^2}{\|\mathbf{x}-\mathbf{z}\|^2+\theta}$
(d) **Exponential** positive definite in $\mathbb{R}^d$	$\exp\left(-\frac{\|\mathbf{x}-\mathbf{z}\|}{\theta}\right)$
(e) **Gaussian** positive definite in $\mathbb{R}^d$	$\exp\left(-\frac{\|\mathbf{x}-\mathbf{z}\|^2}{\theta}\right)$
(f) **Wave** positive definite in $\mathbb{R}^3$	$\frac{\theta}{\|\mathbf{x}-\mathbf{z}\|}\sin\left(\frac{\|\mathbf{x}-\mathbf{z}\|}{\theta}\right)$

Given that the norm of the isotropic kernels $\|\mathbf{x}-\mathbf{z}\|$ defines the distance between the two input vectors **x** and **z**, it identifies a straightforward indication as to what extent the two examples are closely related. In statistical pattern classification terms, this provides a measure about how far away these two vectors are from each other and, accordingly, how similar they are to each other, and as such, as to whether or not they belong to the same class. Given that kernels are also considered as measures of similarity, where the higher the similarity, the more alike the two vectors are (and vice versa), as discussed earlier in Section 2.3.1, the shape characteristics (as represented by these isotropic kernels, for example) can be well utilized to investigate what further properties an SVM kernel would also need to exhibit when measuring the similarity between its input patterns, and hence, aid in the decision process as to whether or not they belong to the same class. This subject is not investigated in Genton's work and will be discussed in more details in Chapter 3 within the context of the solutions proposed in this thesis.

2.6 Kernel fusion via hybridization and multiple kernel learning

Despite the long list of kernels within each of the classes mentioned in the previous sub-section, it is still difficult to explicitly determine which kernels are best suited to a particular problem. Moreover, there are many complex pattern classification applications that cannot be handled easily by a single kernel. It was therefore a natural practice for researchers to investigate how would a combination of kernels help to tackle complex pattern classification problems, especially because of the readily available salient algebraic properties, represented by the 'closure properties' discussed earlier in Section 2.4.3, that admit the legitimate combination of kernels while still preserving their semi-definite properties. The 'multiplication' and 'summation' properties (defined by Propositions 2.4 and 2.5, respectively), seem to be the two most commonly used approaches adopted for creating kernel combination strategies in the literature of this specialised field. Accordingly, the two main streams that appeared in this area were mostly the linear and non-linear kernel combination.

For example, Tan et al. [68] were amongst the firsts to utilize the summative and multiplicative closure properties of Mercer kernels to construct a hybrid kernel from the existing common kernels via the non-negative linear combination and the non-linear product of Mercer kernels, and come up with a more flexible and efficient kernel for SVMs. They conducted some experiments on the hybrid SVM kernel constructed from both the linear combination and the non-linear product of the cubic polynomial kernel (n = 3) and the Gaussian kernel. Their reported results revealed that the hybrid kernel is consistently superior in terms of classification performance compared to when each of the individual kernels are used on their own.

A similar work conducted by Song et al. [69] analysed the areas where different kernels could provide complementary information about different aspects of the data, and accordingly defined kernels in two categories: *global kernels* (e.g., the linear kernel), where samples far from each other can affect the value of the kernel function, and hence, are better at extracting global features of data; and *local kernels* (e.g., Gaussian) that only allow samples close to each other to influence the value of the kernel function and hence they are good at extracting local features of data, but short at extracting global features of samples. As such, the authors proposed the construction of an additive compound kernel, named 'CombKer', from both the linear and Gaussian kernels, in the form of:

$$k(\mathbf{x},\mathbf{z}) = (1-\lambda)\langle \mathbf{x},\mathbf{z}\rangle + \lambda e^{\frac{\|\mathbf{x}-\mathbf{z}\|^2}{\sigma^2}}, \qquad (2.45)$$

where $0 \le \lambda \le 1$, $\lambda \in \Re$,

to take advantage of the good prediction ability of the linear kernel, as well as the good learning ability of the Gaussian kernel. Again, their experimental results revealed that the CombKer outperforms the single individual kernels in terms of classification accuracy.

Similar investigations were also reported in [52] and [70], whereby the linear summative combination of two or more commonly used kernels (e.g., Polynomial, Gaussian, and Sigmoid) experimentally shown to consistently outperform the classification performance of the single individual kernels. Non-linear combination techniques of kernels have also shown promise in some other reported work, such as [71] and [72].

The approach of kernel combination or hybridization has probably emerged as a result of being motivated by the information fusion theory, as defined, for example, by Dasarathy [73]: "*Information fusion encompasses the theory, techniques and tools conceived, developed, and employed for exploiting the synergy in the information acquired from multiple sources (sensors, databases, human sources, etc.) such that the resulting decisions or actions are in some sense better (qualitatively or quantitatively, in terms of accuracy, robustness, etc.) than what would be possible if any of these sources are used individually without such synergy exploitation*". Within the context of this definition, researchers all over the world have been making important contributions to this field in a variety of topics and targeting to solve a range of problems in different applications and disciplines. These include, but are not limited to: defence and military applications, such as automatic target detection, tracking, identification, and recognition [74, 75], anti-personnel landmine detection [76], guidance for autonomous vehicles [77]; and non-military and civilian applications, such as industrial measurements and testing [78], remote sensing [79], robotic applications [80, 81], medical imaging and diagnosis [82, 83], etc.

2.6.1 Classifier fusion

Amongst all the previously mentioned applications that benefited from the information fusion concept, the pattern recognition or classification community is also not an exception. Traditionally, pattern recognition systems used to focus on designing only one classifier to achieve the best possible classification performance for the task at hand.

However, having realised the benefits of fusion, a vastly growing literature on modern pattern classification techniques has already adopted the same concept in utilizing a combination of classifiers and fusing their 'opinions' [84, 85]. Experimental investigations revealed that different classifier designs could potentially offer complementary information about the patterns to be classified, and hence, if appropriately integrated or fused together, could improve the performance of the classification task at hand [86]. A bulk of research has also shown that many complex classification problems can only be reliably solved by fusing the experiences of multiple classifier models together. As such, these observations motivated the research on combining classifiers so that not to rely on a single decision-making scheme. Instead, all designs are to be combined or fused together to collectively contribute towards achieving a better consensus decision.

To date, various classifier fusion schemes have been developed and shown to outperform a single best classifier. Coarsely, these schemes could be classified according to whether the fusion process is taking place at the input (i.e., features) or the output (i.e., decisions) of the classifier. Feature fusion strategies deal with the selection and combination of features to remove redundant and irrelevant features that could confuse the classifier and cause it to poorly perform. The resulting set of features is then fused together to obtain a better discriminative feature set, which is then fed to a classifier to obtain the final result [87]. In decision fusion, on the other hand, an ensemble of classifiers (which could be of the same or different type; e.g., *k*-nearest neighbour, neural network, SVM, etc.) is usually used and their outputs merged together by various methods to obtain the final output. A good review of the techniques and algorithms devised within each of these categories can be found in [88].

2.6.2 Kernel fusion

By analogy, the use of different SVM kernel functions amounts to the construction of different classifier models that captures different aspects of the data. As such, similar to the classifier fusion mentioned in the previous sub-section, the use of different kernels can also provide complementary information about the input data, and therefore the performance of the classifier constructed from multiple kernels fused together would be expected to outperform that of a single kernel.

Given that the kernel function is the element that defines how the SVM classifier model will be formulated, combining or fusing kernels is therefore perceived to be analogous to

the approach of combining classifiers, with the obvious difference of combining the heart of the classifiers rather than their peripheral inputs or outputs. Consequently, the kernel methods research community has also started to realize the benefits of combining kernel functions and apply the same concept in kernel-based learning algorithms to fuse or combine them together. An interesting study, conducted by Lee et al. [89], comparing the two approaches (i.e., kernel combination versus classifier combination) has revealed that the former approach is usually preferred over the latter when a dataset has a varying local data distributions. However, classifier combination methods are sometimes more stable with small size training datasets.

A crucial question, however, was how to combine kernels whilst maintaining their positive semi-definite properties intact. Fortunately, a good answer to this question was provided by the salient 'closure properties' of positive semi-definite kernels, as explained earlier in this section, which permits the construction of more complicated kernels from simpler building blocks using a number of methods, without affecting their Mercer's properties [1].

2.6.3 Multiple kernel learning (MKL)

One of the main drawbacks of the SVM is that the selection of the right kernel to solve a certain task has always relied on the experience of the user and his a priori knowledge of the processing data and the problem at hand. In practice, users usually specify a family of kernels and use the training data to select the kernel that scores the best performance, a problem which is known as *learning kernels* [72]. This is, however, very time consuming and researchers started to realize that there is an increasing demand to automate the process of automatically selecting the right kernel for a given application rather than leaving it to the user's choice. Recent advances in kernel combination research have managed to move a step forward towards achieving this goal, by providing extensive theoretical analysis of this problem both in classification and regression.

It has been noticed that the kernels' closure properties also permit the use of weights (as defined by Proposition 2.6), whilst still retaining their semi-definite properties intact [1]. This means that the construction of linear and/or non-linear combinations of kernels could be weighted, and hence assign an 'importance' coefficient to each kernel. Although the choice of the best kernel that suits a certain classification task is not yet a completely solved problem, recent advances in optimization strategies showed that these weights can be automatically learnt from the training data at hand [90]. And hence, in an indirect way,

by calculating the best/optimum weights, one can determine the most important kernel to be used to solve a certain classification task from an ensemble of kernels being 'weighted' combined (linearly and/or non-linearly).

Lanckriet et al. [91] were amongst the firsts to investigate this subject by adopting a semi-definite programming approach to automatically learn the weights of a linear kernel combination in the form of a convex combination of basis kernels, as:

$$k(\mathbf{x},\mathbf{z}) = \sum_{i=1}^{M} \mu_i k_i(\mathbf{x},\mathbf{z}) \tag{2.46}$$

where $\mu_i \geq 0$, $\sum_{i=1}^{M} \mu_i = 1$,

and M is the total number of kernels combined and μ_i are the assigned weights. Lanckriet et al.'s work have paved the way to create a series of similar approaches in a variety of optimization strategies addressing the problem of not only automatically learning the combination of weights from the input training data, but also the hyperparameters of the associated discriminative classifier [71, 92-96]. This work has amalgamated under a relatively new and interesting research topic known as *Multiple Kernel Learning* (MKL). A good review of the several MKL approaches proposed in the literature can also be found in [59]. Targeting the same goal, similar work adopting genetic programming and evolutionary strategies has also showed promising results in [97] and [98].

Whichever MKL methodology is adopted, the calculated weights would depend on the training dataset (i.e., the problem at hand). As such, some of the kernels in the adopted ensemble would be highly important at some datasets and least important at others, which is what the weights can reveal. While the use of hybrid or a combination of kernels demonstrates improved classification performance in many occasions, this is done at the expense of more parameters to be determined by solving a series of additional optimization problems, which results in a higher computation complexity and longer computation time. However, it reduces the problem of kernel selection to the task of determining the weights rather than the kernels; i.e., by automatically learning the best weights that suit the classification task at hand, one is therefore able to automatically determine which kernel is best suited to a specific problem, and hence move a step forward in solving an important open problem.

2.7 Summary

This chapter presented a survey on the SVM classification algorithm, together with a critical analysis and study of the properties of kernel functions which it adopts to tackle non-linearly separable classification problems. The aim was to highlight the main research problem addressed by the thesis, where the use of kernels as similarity functions has been overlooked in previous literature. The chapter also briefly analyzes previous kernel combination strategies, within the context of information fusion; a concept which will play a central role in the solution devised by the thesis (especially in Chapter 4) to address the identified research problem.

As such, if this problem is addressed properly, researchers would therefore be able to craft kernels using some natural and intuitive 'similarity-based' properties (which describe how the examples belonging to the same or different classes resemble (or differ from) each other) which are more tangible quantities than the 'mathematical' positive semi-definite properties of the implicit high-dimensional feature spaces which one might not even be able to calculate. Moreover, restricting kernels to positive semi-definite functions may rule out several natural notions of similarity that arise in many practical applications which are not positive semi-definite, and yet can successfully be used for learning to achieve reasonable generalisation levels.

The research work presented in this thesis therefore addresses this problem by utilizing the SVM kernel as a tool to measure the similarity between its two input vector arguments, alongside with its standard definition as an implicit mapping tool to higher-dimensional feature space, as prescribed by its positive semi-definitness property. As such, the subsequent chapters explore what appropriate similarity-based properties can be exploited to characterise SVM kernels, when used as similarity functions, and to assess the influence of these properties on the SVM classification performance.

Chapter 3
Proposed Similarity-Based Characteristics for SVM Kernel Functions

3.1 Introduction

This chapter aims to highlight the importance of studying kernel functions in terms of their definition as similarity measures alongside with their standard definition as implicit mapping tools to high-dimensional feature spaces using the positive semi-definiteness property (or Mercer's condition). This study will enable a machine learning practitioner to design kernels in terms of some natural and intuitive 'similarity-based' properties which are more tangible quantities than the 'mathematical' properties of the implicit high-dimensional feature spaces that one might not even be able to create.

The chapter presents a theoretical investigation on kernels, when used to measure the similarity between its two input instances, for pattern classification by SVMs. It proposes some appropriate and intuitive criteria, based on how the *shape* of a kernel should typically look like, when measuring the similarity between its inputs. Motivated by the previously proposed plots for the Chebyshev kernel, the chapter then adopts the orthogonal polynomial kernels paradigm (such as Chebyshev, Legendre, and Hermite kernels), to inform the underpinning theoretical analysis of the proposed similarity-based shape characteristics, and to assess their effectiveness in aiding the SVM classifier to achieve more correct classification decisions.

3.2 The need for intuitive kernel-design criteria

3.2.1 Why positive semi-definite kernels?

It has been explained in Chapter 2 how a positive semi-definite function k corresponds to the inner product of the images of the input data (under some mapping function Φ) in some high-dimensional feature space, as:

$$k(\mathbf{x},\mathbf{z}) = <\Phi(\mathbf{x}),\Phi(\mathbf{z})>. \qquad (3.1)$$

This observation enabled the 'kernelization' of the soft-margin SVM[4], by directly replacing the inner product in its dual optimization problem by another function[5] that is proven to be positive semi-definite (or equivalently meet Mercer's condition), as positive semi-definite functions are also inner products but in another high-dimensional space.

Although this mapping to the high-dimensional space is never explicitly performed in practice[6], it can enhance the linear separability of the overlapped classes in the input space, when the number of dimensions increases[7]. As such, the implicit mapping to high-dimensional spaces, embedded in the utilization of positive semi-definite kernels, enables solving non-linearly separable classification problems in the input space by calculating linear decision boundaries in the high-dimensional feature space, where the classes can become linearly separable. This is how the kernelized SVM has gained its popular non-linear classification power via the adoption of a positive semi-definite kernel function that has the effect of implicitly mapping the non-linearly separable classes in the input space to a higher-dimensional feature space where they can become linearly separable, and hence easier to classify.

Extensive theoretical and mathematical research have been conducted in order to specify what characteristics such a kernel must have, and the existing rich literature seems to have reached a generic consensus that it does indeed need to be positive semi-definite (or equivalently meet Mercer's condition) to enable the direct replacement of the inner product in the dual optimization problem and, at the same time, provide the notion of implicit mapping to high-dimensional spaces, where classes can become linearly separable. The positive semi-definiteness property also ensures that the SVM can be efficiently solved using convex quadratic programming, otherwise convex solutions can be very challenging to achieve[8] [16].

[4] And many other learning-based algorithms that are configured to accept only inner products between the input data.
[5] See Eqs. (2.12) and (2.19) in Chapter 2.
[6] Because of the kernel trick that enables the kernel function to be applied directly to the input data, as explained in Section 2.3.4.
[7] See for example Figure 2.3 in Chapter 2 and Figure A.1 in Appendix A, and the popular youtube video in: https://www.youtube.com/watch?v=3liCbRZPrZA&list=PLqS2sO7F2t3V0T25Aoj4-ScgHJqJ7aUG0
[8] Although some research, such as [19, 23-29], were actually able to demonstrate that indefinite similarity functions can also achieve reasonable generalisation levels and, hence, can successfully be used for learning, arguing for the need for the kernel to be only restricted to positive semi-definite functions.

3.2.2 Positive semi-definite versus similarity measure kernels

Being defined as inner products in some high-dimensional space, positive semi-definite kernel functions have also been defined as legitimate ‘similarity measure’ tools in many contexts [9, 16, 19]. However, limited efforts seem to have studied kernels from the ‘similarity measure’ perspective rather than the positive semi-definiteness point of view. Amongst the few who actually started to recognize this deficiency, is the line of work of Balcan et al. [20-22] and Kar et al. [23], who realized the drawbacks of limiting the kernel studies to only their implicit mapping theory defined by the positive semi-definite property. For example, some of the reported drawbacks include:

1- The theory does not directly correspond to one’s intuition of what a good kernel should offer as a good ‘measure of similarity’. As such, it may be difficult for a domain expert to use the theory to construct or customize an appropriate kernel (as a similarity measure tool) for the learning task at hand [20, 21, 99].

2- Different types of kernel functions, if viewed as similarity functions, could describe different notions of similarity between objects, which does not correspond to any intuitive or easily interpretable high dimensional representation. This is due to the fact that the underlying high-dimensional feature space is usually not apparent in ‘natural’ representations of the data in the input space, as its construction is only implicit [20, 21, 99].

3- It can sometimes be unsatisfactory and unclear to explain the effectiveness of a learning algorithm in terms of the properties of an implicit high-dimensional feature space that one might not even be able to calculate. There is also a prosaic explanation of what it is that makes a kernel useful for a given learning problem [20].

4- The requirement of positive semi-definiteness may rule out several natural notions of similarity that arise in many practical scenarios that are not positive semi-definite. In other words, many practical similarity functions do not actually satisfy the mathematical properties of an inner product [19], as defined by the positive semi-definiteness property or the Mercer’s condition, and thus, the subsequently produced similarity matrix can be indefinite. For this reason, some methods have been proposed to modify similarities into kernels [19, 100], for example by applying some spectrum transformation procedures (such as Spectrum Clip, Flip, Shift, and Square) to the similarity matrix to change it to a positive semi-definite matrix. However, these

methods can be quite complicated and could require substantial work, possibly reducing the quality of the function, to coerce it into a 'legal' form [20].

For these reasons, Balcan et al. [20-22] developed a theory for learning with similarity functions that addresses a number of these issues. It is more general and in terms of more tangible quantities than the standard theory of kernel functions. It is also based upon some natural and intuitive 'similarity-based properties' that do not require reference to implicit high-dimensional spaces, nor do they require that the similarity function be positive semi-definite or even symmetric. Simply put, their approach is based on developing a mathematical definition of a 'goodness' criterion that describes a notion of what it means for a pairwise function to be a 'good' similarity function and how it should behave. This is basically stemming from the idea that the 'output' of the similarity function should be indicative (as one might intuitively want it to be) about the fact that data examples belonging to the same class are 'expected'[9] to be more similar to each other than those belonging to different classes. In other words, the 'intra-class' similarity should be sufficiently large compared to the 'inter-class' similarity. If a similarity function satisfies such a 'good' criterion, then, given l training examples, there must exist a linear separator that has a specifiable error at some maximum margin. By this way, Balcan et al. [20-22] were able to show why a kernel function under the usual definition can also be a good similarity function under this 'goodness' definition. Such a kernel-to-similarity translation is therefore regarded as the first formal theoretical justification to the standard intuition about kernels and to their good empirical performance.

A direct implication of the theory proposed by Balcan et al. is that one can think (in the design process) of the usefulness of a kernel function in terms of more intuitive 'similarity' properties that can be applied directly to the data in their original input space without needing to refer to implicit mapping to high-dimensional feature spaces. The work presented in this thesis follows some footsteps akin to that of Balcan et al. to investigate some further intuitive characteristics that should naturally be embedded in kernels as similarity functions based on their *shape* properties. It studies how similarity measure kernels should graphically look like and adopts the previously proposed orthogonal polynomial kernels as an example to explore the implications of their shape characteristics upon the resulting SVM classification performance.

[9] Hence the repeated calculations of the probabilistic expectations **E** in their theory.

3.3 Intuitive shape properties of similarity functions

Being regarded as similarity measures, it has been explained in Section 3.2.2 why some researchers, such as Balcan et al. [20], advocate the need for thinking about the usefulness of a kernel function in terms of more intuitive 'similarity-based' properties instead of its 'mathematical' properties as defined by the Mercer's (or positive semi-definite) condition. However, although Balcan et al. [20] developed a number of alternative mathematical conditions, based on one's intuition of what a similarity function is expected to achieve, it was still not very clear what such kernels could be, or how to construct them. There is also a lack of empirical investigations to validate their theory and explore if it actually holds true.

The work presented in this thesis is inspired by such 'similarity-based' intuitive perspective. Unlike previous work, however, the thesis addresses this subject from more tangible quantities stemming from the *shape* characteristics of the kernel as dictated by the definition of similarity measures. In other words, the investigations presented herein are mainly based on how the typical kernel shape should look like, being thought of as a tool to measure similarity between two input vectors. This is thought to provide a more tangible and straightforward resource to readily help the designer of the SVM kernel to intuitively either select or adapt an appropriate kernel for the dataset at hand.

3.3.1 Ideal shape characteristics of kernels as measure of similarity

Kernel-based learning algorithms, such as the SVM for pattern classification (which is the subject of study in this thesis), utilize a kernel function to measure the similarity between two input objects [30]. It can be seen as a generalization for the inner-product operation in vector spaces. As explained in Chapter 2, the function does so by computing the inner-product between the images of two input vectors projected into some high-dimensional feature space. Being considered as a similarity measure between its two input vector arguments $\mathbf{x}$ and $\mathbf{z}$, an SVM kernel is expected to follow the same typical definition of 'similarity functions' mentioned in Section 2.3.2.2; i.e., its value increases, when the two vectors are more similar to each other (and vice versa), and be maximized, when the two vectors are identical to each other. In pattern classification, this would typically mean that it should assign a higher similarity score to any pair of vectors that belong to the same class than it does to any pair of vectors from different classes [101]. This would be the case if the implicit mapping by the kernel function brings similar objects close together and takes dissimilar objects apart from each other in the higher-dimensional feature space.

Assuming that input vectors belonging to the same class usually share more similar characteristics than those belonging to different classes, and given that the pairwise calculation of the SVM kernel is essentially a scalar quantity (reflecting the degree of similarity between its two input vectors), it would be expected that the calculated kernel value is high, when its two input vectors are highly similar (and therefore belong to the same class), and it is maximized when they are identical, and it decreases monotonically[10] as they depart away from each other.

One can depict these intuitive characteristics into a graph of what such an ideal kernel should look like by thinking of it as a bell-like shape curve if plotted against the absolute length (or magnitude) of the lag vector between the two input vector arguments $\|\mathbf{x}-\mathbf{z}\|$. In a typical normalized vector space, one can therefore intuitively interpret the normalized shape of such an ideal kernel (as the one illustrated in Figure 3.1), as being maximized, when the two inputs are identical, and decaying monotonically (i.e., decreasing and never increasing back), when they depart away from each other. If these properties are not strictly satisfied, the subsequently calculated similarity measure could be inaccurate, in the sense that it does not reflect the true resemblance of the kernel's two input vector arguments, a fact, which can have a negative effect on the subsequent classification performance.

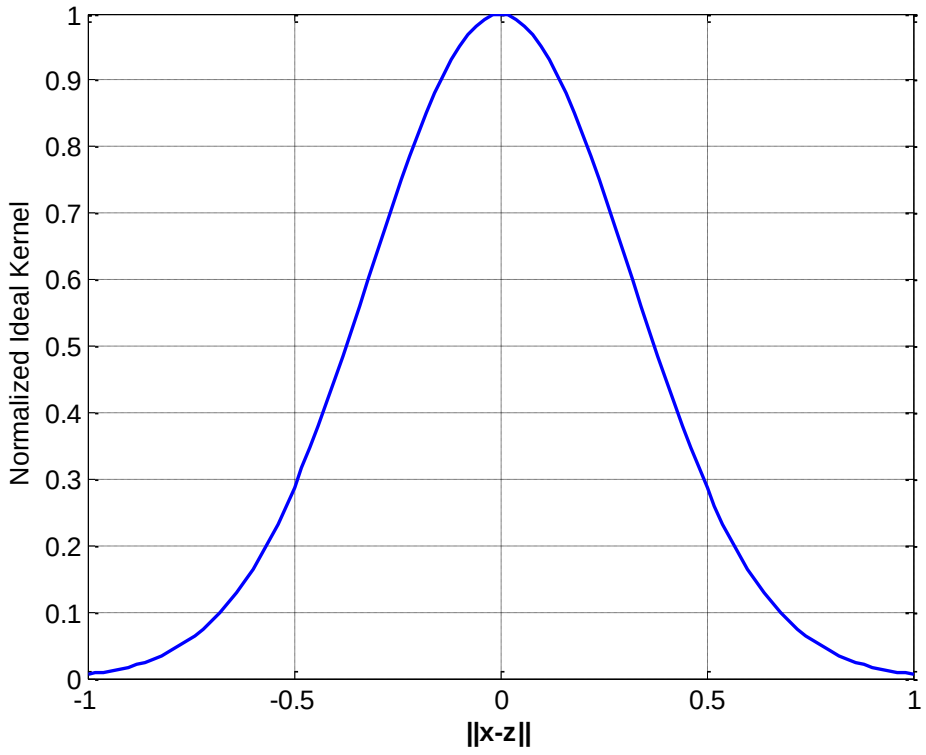


Figure 3.1 A typical example of the shape characteristics of an ideal kernel defined as a similarity function. The kernel value is maximized when the two vectors are identical, and decays monotonically when the distance between them increases.

[10] A monotonically decreasing function is a function that is entirely decreasing; i.e., $f(x)$ is not allowed to increase as x increases.

One can straight away realize that this is actually the shape of the well-known radial basis function defined by the Gaussian kernel:

$$k(\mathbf{x}, \mathbf{z}) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{z}\|^2\right), \tag{3.2}$$

which perhaps explains the reason behind its popularity in SVM classification tasks[11], as it clearly demonstrates the properties of what would be intuitively expected from an ideal kernel being thought of as a similarity measure.

Furthermore, if we refer back to the dual optimization problem of the non-linear kernel-based SVM (in (2.19) - Section 2.2.4) during the training phase, one can notice that the maximisation of $Q(\boldsymbol{\alpha})$ (which is essential to ensure a maximum margin, and, hence, an optimum decision boundary, is obtained) requires that $\alpha_i \alpha_j y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$ has a high negative value, given that $\alpha_i > 0$ and $\alpha_j > 0$. This value will be negative if y_i and y_j have opposite signs (i.e., the two vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ are from different classes), and it will be high if α_i, α_j and $k(\mathbf{x}_i, \mathbf{x}_j)$ are all high. Thus, $Q(\boldsymbol{\alpha})$ is maximised when $\mathbf{x}_i$ and $\mathbf{x}_j$, from different classes, are highly similar (thus the value produced by the kernel $k(\mathbf{x}_i, \mathbf{x}_j)$ is high), and their corresponding α_i and α_j are assigned high values by the optimisation process. In effect, the optimization problem in (2.19) searches for the most similar vectors that belong to different classes; these are the support vectors.

The same kernel is also used during the testing or classification phase, as shown by the decision function in (2.21). It can be seen from the equation that the support vectors most similar to the test vector $\mathbf{x}$ will contribute high values to the sum, and can thus tip the decision of the classifier towards the correct class label of the test vector. The higher the similarity measure calculated by the underpinning kernel between the support vectors most similar to the test vector $\mathbf{x}$, the more likely a correct decision is obtained.

This shows that, during both the training and classification phases of the SVM algorithm, the higher the similarity between two inputs, the stronger their influence on the selection of support vectors, and on the classification decision, respectively. The SVM algorithm thus requires a similarity measure that outputs a high value, when its two input arguments are most similar, and outputs decreasing values, as they become less and less similar. The proposed ideal similarity function model illustrated in Figure 3.1 can therefore well fulfil these similarity-based measure characteristics.

[11] Together with its renowned ability to implicitly map the input space to a feature space of infinite dimensionality.

3.3.2 Shape of isotropic stationary kernels

Clearly the SVM Gaussian kernel is already explicitly expressed in terms of the lag vector between its two input arguments $\|\mathbf{x}-\mathbf{z}\|$, and, hence, it was easy to plot its value against the magnitude of this lag vector straightaway and investigate its shape. Actually, a number of other well-known kernels are also explicitly expressed in terms of the magnitude of this lag vector. A fact which encouraged Genton [18] to easily plot some examples of them, as shown in Figure 2.5 and Table 2.2, and categorized them, from his statistics perspective, as being 'isotropic stationary kernels'.

Although Genton managed to plot some examples of these isotropic stationary kernels, he did not actually reflect upon the usefulness of these plots in characterising the properties of what would be expected from them as being regarded as similarity measures instead of positive definite kernels, and how could this impact upon their performance in pattern classification tasks. His main aim was rather to present a categorization of kernels for machine learning from a statistics perspective.

As can be observed, apart from the wave kernel, all the other examples shown in Figure 2.5, do actually illustrate different variations of the properties of the ideal kernel being thought of as a similarity measure, as explained earlier in Figure 3.1; i.e., the kernel value is maximized when the distance between the two vectors is zero (i.e., when the two vectors are identical) and decreases monotonically as the distance between them increases. The wave kernel illustrated in Figure 2.5 (f) is shown to be maximized, when the two vectors are identical, and decreases, as the distance between them starts to increase; however, after a certain threshold, the kernel increases again and then fluctuates in a wavy pattern. As this is obviously not a monotonically decreasing behaviour, the subsequently calculated measure of similarity could be wrongly evaluated, and, therefore, the classification performance of this kernel could be affected destructively.

3.4 Shape characteristics of orthogonal polynomial kernels

Many other kernels, however, are not explicitly expressed in terms of the lag vector between its two input vector arguments (e.g., the locally stationary and the non-stationary kernels – see Table 2.1), and can be tricky to plot. Motivated by the Chebyshev polynomial kernels plots reported by Ozer et al. [17], this section will explore how these kernels were plotted, and extend the authors' work to investigate and analyse the shape characteristics of SVM kernels that are constructed from some other orthogonal polynomials as well, such as

the Chebyshev [31-33], Legendre [34], and Hermite [35] polynomials. The following subsection will first briefly review previous work on SVM kernels constructed from orthogonal polynomials as reported in the literature.

3.4.1 Construction of SVM kernels from orthogonal polynomials

SVM kernels constructed from orthogonal polynomials have been proposed several times in the literature [11, 17, 31-35, 102-109]. Reported experimental results have shown that they exhibit some salient characteristics, such as they display minimum data redundancy in the feature space and they require less support vectors to construct a discriminative classifier. This means that less memory and execution time are required for solving the quadratic programming problem [110]. Moreover, they have also demonstrated superior classification performance over traditional kernels in some occasions.

Orthogonal polynomials are various families of polynomials, which are useful in solving differential equations arising in physics and engineering. In general, they have many important applications in such areas as mathematical physics, interpolation theory, the theory of random matrices, computer approximations, and many others [102]. The first few orders of some of these orthogonal polynomials are shown in Table 3.1, and more details about them can be found in Appendix A5.

In fact, the potential use of orthogonal polynomials to construct SVM kernels was first pointed out by V. N. Vapnik in 1998 [11]. As an example, he showed that Hermite polynomials can be used to construct one-dimensional Hermite kernels (i.e., for scalar inputs) in the form of:

$$k_{Her}(x,z)=\sum_{i=0}^{\infty} q^i H_i(x)H_i(z) \tag{3.3}$$

where q is a convergence factor that can be chosen in the range of $0 \le q \le 1$.

However, it was not until 2006 when the usefulness of orthogonal polynomials to construct SVM kernels started to attract wider attention by the machine learning community, through the work of Ye et al. [31]. By following a similar approach to Vapnik, Ye et al. proposed to use the Chebyshev polynomials of the first kind to construct SVM kernels. They utilized the approximation equation theory to show that Chebyshev polynomials could be decomposed into an inner product of a series of coefficients and orthogonal radix, and hence satisfy Mercer's condition. Later on, Zhou et al. [102] extended this work to derive a generic formulation of SVM orthogonal polynomial kernels (OPKs) that can be applicable

to any orthogonal polynomial family, and not only the Chebyshev. They adopted an alternative theoretical approach based on the Fourier series of square-integrable functions where the orthogonal polynomials, defined as a positive semi-definite inner product on pairs of polynomials, are used as their basis.

Interestingly, both theories (of Ye et al. [31] and Zhou et al. [102]) agreed that the polynomial element of the SVM kernel constructed from these polynomials (i.e., without being combined with any weighting functions) should be formulated for 'scalar inputs' as:

$$k(x,z) = \sum_{i=0}^{n} P_i(x) P_i(z) \tag{3.4}$$

where $P(\cdot)$ denotes the evaluation of the polynomial (i.e., Chebyshev, Legendre, etc.) on the first and second input kernel arguments x and z, respectively, and n is the highest polynomial order utilized in the kernel. This polynomial element of the kernel, shown in (3.4), is referred to in this thesis as the '*unweighted*' polynomial kernel, for short.

Table 3.1 Comparing the first few orders of some orthogonal polynomials.

Chebyshev	Hermite	Legendre	Laguerre
$T_0(x)=1$	$H_0(x)=1$	$L_0(x)=1$	$P_0(x)=1$
$T_1(x)=x$	$H_1(x)=2x$	$L_1(x)=x$	$P_1(x)=-x+1$
$T_2(x)=2x^2-1$	$H_2(x)=4x^2-2$	$L_2(x)=\frac{1}{2}(3x^2-1)$	$P_2(x)=\frac{1}{2}(x^2-4x+2)$
$T_3(x)=4x^3-3x$	$H_3(x)=8x^3-12x$	$L_3(x)=\frac{1}{2}(5x^3-3x)$	$P_3(x)=\frac{1}{6}(-x^3+9x^2-18x+6)$
$T_4(x)=8x^4-8x^2+1$	$H_4(x)=16x^4-48x^2+12$	$L_4(x)=\frac{1}{8}(35x^4-30x^2+3)$	$P_4(x)=\frac{1}{24}(x^4-16x^3+72x^2-96x+24)$
$T_5(x)=16x^5-20x^3+5x$	$H_5(x)=32x^5-160x^3+120x$	$L_5(x)=\frac{1}{8}(63x^5-70x^3+15x)$	$P_5(x)=\frac{1}{120}(-x^5+25x^4-200x^3+600x^2-600x+120)$
$T_6(x)=32x^6-48x^4+18x^2-1$	$H_6(x)=64x^6-480x^4+720x^2-120$	$L_6(x)=\frac{1}{16}(231x^6-315x^4+105x^2-5)$	$P_5(x)=\frac{1}{720}(x^6-36x^5+450x^4-2400x^3+5400x^2-4320x+720)$
$T_n(x)=2xT_{n-1}(x)-T_{n-2}(x)$	$H_n(x)=2x\,H_{n-1}(x)-2(n-1)\,H_{n-2}(x)$	$L_n(x)=\frac{1}{n}\left[(2n-1)xL_{n-1}(x)-(n-1)L_{n-2}(x)\right]$	$P_n(x)=\frac{1}{n}\left[(2n-1-x)P_{n-1}(x)-(n-1)P_{n-2}(x)\right]$

In most of the pattern recognition applications, however, the input data are in multidimensional vector format. Therefore, to extend the one-dimensional polynomial kernel in (3.4) to multidimensional input vectors, two approaches have been previously proposed. The following sub-sections analyse each of these approaches in detail to investigate how the unweighted polynomial kernel in (3.4) is evaluated on 'vectors input'.

3.4.1.1 The pairwise processing approach

In order to utilize orthogonal polynomials to construct SVM kernels for multidimensional input vectors, Vapnik developed a theorem whereby the evaluations of the one-dimensional kernel on the individual features of the input vectors represent coordinatewise basis functions that should be directly multiplied by each other to formulate the overall multidimensional kernel [11, 108]. In Vapnik's words, this means that "*the kernel that defines the inner product in the m-dimensional basis is the product of m one-dimensional kernels*" [11]. As such, the author proposed to construct the multidimensional OPK in the form of tensor product of one-dimensional kernels as:

$$k(\mathbf{x},\mathbf{z}) = \prod_{j=1}^{m} k_j(x_j, z_j) \tag{3.5}$$

Ye et al. afterwards followed a similar approach to construct multidimensional Chebyshev kernels from the Chebyshev polynomials of the first kind [31]. Their approach is based on a decomposition technique whereby the Chebyshev polynomials first process each corresponding scalar feature pair of the two input vectors **x** and **z** and multiplies them together in a pairwise fashion; and then, to evaluate the overall kernel, these scalar pairwise kernels are afterwards multiplied by each other [31]. Therefore, for *m*-dimensional input vectors **x** and **z** $\in \Re^m$ given by $\mathbf{x} = \{x_1, x_2, \cdots, x_m\}$ and $\mathbf{z} = \{z_1, z_2, \cdots, z_m\}$, Ye et al.'s unweighted Chebyshev kernel, constructed by following this approach, is formulated as:

$$k_{Che}(\mathbf{x},\mathbf{z}) = \prod_{j=1}^{m} k_j(x_j, z_j) = \prod_{j=1}^{m} \sum_{i=0}^{n} T_i(x_j) T_i(z_j), \tag{3.6}$$

where $T(\cdot)$ are the Chebyshev polynomials. Due to the nature of its evaluation on the pairwise features of the input vectors, this processing methodology is referred to in this thesis as the "*pairwise*" processing approach.

Following the pairwise processing approach proposed by [31], Pan et al. [34] have similarly exploited the unity weighting function of the Legendre polynomials, $w(x)=1$, to propose the Legendre kernels for multidimensional input vectors as:

$$k_{Leg}(\mathbf{x},\mathbf{z}) = \prod_{j=1}^{m}\sum_{i=0}^{n} L_i(x_j)L_i(z_j) \tag{3.7}$$

Being constructed without a weight, Pan et al. explained that the Legendre kernel is faster (e.g., compared to the Chebyshev kernel) during classification and saves more time.

Although this pairwise processing approach was the first to propose a methodology by which orthogonal polynomials process input data vectors, and this same methodology has also been followed afterwards by Zhou et al. [102], Pan et al. [34], and Wei and Pan [32] for some other orthogonal polynomials, such as Legendre and Hermite, it suffers from the problem of multiplying the kernels evaluated on the individual features of the input vectors, as shown in (3.5). As explained by Ozer et al. [17], kernels that are constructed in this form via a multiplication operation on the j^{th} elements of the vector pair **x** and **z** can actually lead to poor generalization if it happens that one of the kernels being multiplied is close to zero at the time, when the two vectors **x** and **z** are actually quite similar to each other.

3.4.1.2 The vectorial processing approach

The second approach, by which the orthogonal polynomials process the input data, is the one proposed later on by Ozer et al. [17]. Their main aim was to tackle the multiplication problem identified in the pairwise approach, and hence improve the generalization capabilities of the underpinning OPKs. Again, focused on only the Chebyshev polynomials, Ozer et al. proposed to apply them to the input vectors as a whole, rather than to their individual feature components, and therefore defined the generalized Chebyshev polynomials for vector inputs, as shown in Table 3.2, for the first few orders. By applying the polynomials to the input vectors as a whole, this approach is referred to in this thesis as the "*vectorial*" processing approach.

Of particular importance later in this thesis (in Chapter 6), is to highlight the observation that the odd orders of the generalized Chebyshev polynomials shown in Table 3.2 produce vector quantities, whereas the even orders produce scalar quantities.

Table 3.2 First few orders of the Generalized Chebyshev polynomials, as proposed by Ozer et al. [17].

Order	Generalized Chebyshev polynomials
0	$T_0(\mathbf{x}) = 1$
1	$T_1(\mathbf{x}) = \mathbf{x}$
2	$T_2(\mathbf{x}) = 2\langle\mathbf{x},\mathbf{x}\rangle - 1$
3	$T_3(\mathbf{x}) = \mathbf{x}(4\langle\mathbf{x},\mathbf{x}\rangle - 3)$
4	$T_4(\mathbf{x}) = 8(\langle\mathbf{x},\mathbf{x}\rangle)^2 - 8\langle\mathbf{x},\mathbf{x}\rangle + 1$
5	$T_5(\mathbf{x}) = \mathbf{x}(16(\langle\mathbf{x},\mathbf{x}\rangle)^2 - 20\langle\mathbf{x},\mathbf{x}\rangle + 5)$
6	$T_6(\mathbf{x}) = 32(\langle\mathbf{x},\mathbf{x}\rangle)^4 - 48(\langle\mathbf{x},\mathbf{x}\rangle)^2 + 18\langle\mathbf{x},\mathbf{x}\rangle - 1$

Having applied the Chebyshev polynomials to the input vectors as a whole (in the way shown in Table 3.2), Ozer et al. then defined the 'unweighted' generalized Chebyshev kernel in a dot-product form, as:

$$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} T_i(\mathbf{x}) T_i^T(\mathbf{z}) = \sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle \tag{3.8}$$

and eventually they defined the overall 'composite' generalized Chebyshev kernel (combined by multiplication with its corresponding weighting function) as:

$$k_{G-Che}(\mathbf{x},\mathbf{z}) = \frac{\sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle}{\sqrt{m - \langle \mathbf{x},\mathbf{z} \rangle}} \tag{3.9}$$

where m is the dimension (i.e., number of features) of the input vectors. The experimental results reported by Ozer et al. showed the superiority of their proposed vectorial approach over the pairwise approach, on a number of datasets, though using only the Chebyshev polynomials. They explained that this is believed to be due to the fact that the generalized Chebyshev polynomials are now processing the input vectors as a whole, rather than their individual scalar feature components, and therefore they avoid the multiplication problem that the pairwise approach suffers from. Interestingly, Qu et al. [109] have actually utilized the generalized Chebyshev kernel defined by (3.9) to effectively classify various states of viscoelastic sandwich structures; whereas Zhao et al. [107] introduced a new technique to combine the Chebyshev polynomial kernels of the first and second kinds to propose a new kernel, which they referred to as the 'unified Chebyshev kernel', for support vector regression. Following [17], Tian and Wang [105] have also utilized the vectorial approach to propose the generalised Legendre kernels, based on the generalised Legendre polynomials, as [105, 108]:

$$k_{G-Leg}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle L_i(\mathbf{x}), L_i(\mathbf{z}) \rangle \tag{3.10}$$

On the other hand, by utilizing the kernels' closure property that states that the multiplication of two valid kernels is also a valid kernel (see Section 2.4.3), Ozer et al. also

showed how the generalised Chebyshev kernel can be modified to produce other kernels by replacing the weighting function with another kernel, such as the exponential Gaussian kernel. As such, they proposed the 'Modified Chebyshev kernel' as [17]:

$$k_{Exp-Che}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right) \tag{3.11}$$

where γ is the Gaussian kernel's parameter. This approach has motivated some researchers to investigate how the combination of different kernels with OPKs can constructively influence the performance of the SVM algorithm. For example, Zhang et al. [106] have proposed the modified Legendre kernels by combining the generalised Legendre kernels in (3.9) with the Gaussian kernel as:

$$k_{Exp-Leg}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle L_i(\mathbf{x}), L_i(\mathbf{z}) \rangle \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right) \tag{3.12}$$

Similarly, Jafarzadeh et al. [33] have also proposed two new kernels based on the multiplicative combination of the generalised Chebyshev kernel with the Gaussian and wavelet kernels in the following formats:

$$k(\mathbf{x},\mathbf{z}) = \frac{\sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle}{\sqrt{m - \langle \mathbf{x},\mathbf{z} \rangle}} \times \sum_{i=1}^{m} \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right) \tag{3.13}$$

$$k(\mathbf{x},\mathbf{z}) = \frac{\sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle}{\sqrt{m - \langle \mathbf{x},\mathbf{z} \rangle}} \times \prod_{i=1}^{m} \left(\cos\left(1.75 \frac{x_j - z_j}{a}\right) \exp\left(-\frac{\|\mathbf{x}-\mathbf{z}\|^2}{2a^2}\right)\right) \tag{3.14}$$

where a is the kernel parameter of the wavelet kernel.

Moghaddam and Hamidzadeh [35], on the other hand, decided to adopt the summative closure property instead, to propose a new set of kernels based on the summation of the probabilists' Hermite kernels (denoted by $He(x)$, and formulated using the pairwise approach) with the wavelet, Gaussian, and Chebyshev kernels, which they constructed for scalar inputs respectively as:

$$k(x,z) = \sum_{i=0}^{n} He_i(x) He_i(z) + \prod_{i=1}^{n} \left(\cos\left(1.75 \frac{x_i - z_i}{a}\right) \exp\left(-\frac{(x-z)^2}{2a^2}\right)\right) \tag{3.15}$$

$$k(x,z) = \sum_{i=0}^{n} He_i(x) He_i(z) + \exp\left(-\gamma (x-z)^2\right) \tag{3.16}$$

$$k(x,z) = \sum_{i=0}^{n} He_i(x) He_i(z) + \frac{\sum_{i=0}^{n} T_i(x) T_i(z)}{\sqrt{m - xz}}. \tag{3.17}$$

Recently, Tian and Wang [108] have also combined (by multiplication) the generalised Chebyshev and Legendre kernels (formulated using the vectorial approach) with the

Triangular kenel [111] to propose what they referred to as the 'triangularly modified Chebyshev kernel' and the 'triangularly modified Legendre kernel', which they defined respectively as:

$$k_{Tri-Che}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle \times \left(1 - \frac{\|\mathbf{x}-\mathbf{z}\|}{\lambda}\right)_+ \tag{3.18}$$

$$k_{Tri-Leg}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle L_i(\mathbf{x}), L_i(\mathbf{z}) \rangle \times \left(1 - \frac{\|\mathbf{x}-\mathbf{z}\|}{\lambda}\right)_+ \tag{3.19}$$

where the $(.)_+$ ensures that the kernel is a positive semi-definite function.

Most of these studies on the construction of SVM kernels from orthogonal polynomials are summarised and compared by the recent work of Tian and Wang in [108]. One can notice, however, that the pairwise and vectorial processing approaches only differ from each other when the polynomial basis functions process multidimensional input vectors, whereas when the inputs are scalars, these two processing approaches are mathematically the same.

3.4.2 How to plot a polynomial kernel?

3.4.2.1 Chebyshev polynomial kernels

Although Ye et al. [31] were among the firsts to make use of the Chebyshev polynomials of the first kind to construct SVM kernels for pattern classification in 2006, it was not until 2011 when Ozer et al. [17] provided a clear method to plot them in 2D and briefly reflected upon their shape characteristics. They reported the plots of two versions of their generalised Chebyshev kernel: the first one is the kernel constructed with the sum, as defined in (3.9), which involves all the polynomial orders from 0 up to *n*, and the second one is the kernel constructed from only the polynomial order *n* without summing up any of the preceding orders, defined as:

$$k(\mathbf{x},\mathbf{z}) = \frac{\langle T_n(\mathbf{x}), T_n(\mathbf{z}) \rangle}{\sqrt{m - < \mathbf{x},\mathbf{z} >}} \tag{3.20}$$

It is not very clear though why Ozer et al. reported the second version of the kernel without the sum in (3.20), and the only comment they made is that it cannot be used in SVM for the similarity purpose, without providing much further justification/detail as to why this is the case.

However, for simplicity and to view the shape of the kernel in 2D, Ozer et al. [17] assumed that the data are scalars (i.e., *m*=1) instead of vectors. As such, the dot-product operation

appearing in (3.9) and (3.20) reduce to a simple multiplication, and therefore can be re-written as:

$$k(x,z) = \frac{\sum_{i=0}^{n} T_i(x) \times T_i(z)}{\sqrt{1-xz}} \tag{3.21}$$

$$k(x,z) = \frac{T_n(x) \times T_n(z)}{\sqrt{1-xz}} \tag{3.22}$$

As the kernels in (3.21) and (3.22) are functions of two variables *x* and *z*, to plot them in 2D, Ozer et al. have chosen to affix one of them (namely the *x*-value) and then plot the kernel against the other *z* variable within the normalized range of [-1,+1]. Following this technique, they plotted each of these kernels at three specific *x*-values of: -0.77, 0, +0.77 (Although again it is not very clear in their paper why these *x*-values have been chosen in specific).

This means that the authors plotted the kernel in (3.21) three times as:

$$k(0.77,z) = \frac{\sum_{i=0}^{n} T_i(0.77) \times T_i(z)}{\sqrt{1-0.77z}}, \quad \text{shown in Figure 3.2 (b)} \tag{3.23}$$

$$k(0,z) = \frac{\sum_{i=0}^{n} T_i(0) \times T_i(z)}{\sqrt{1}}, \quad \text{shown in Figure 3.2 (d)} \tag{3.24}$$

$$k(-0.77,z) = \frac{\sum_{i=0}^{n} T_i(-0.77) \times T_i(z)}{\sqrt{1+0.77z}}, \quad \text{shown in Figure 3.2 (f)} \tag{3.25}$$

And similarly, they also plotted the kernel in (3.22) three times as:

$$k(0.77,z) = \frac{T_n(0.77) \times T_n(z)}{\sqrt{1-0.77z}}, \quad \text{shown in Figure 3.2 (a)} \tag{3.26}$$

$$k(0,z) = \frac{T_n(0) \times T_n(z)}{\sqrt{1}}, \quad \text{shown in Figure 3.2 (c)} \tag{3.27}$$

$$k(-0.77,z) = \frac{T_n(-0.77) \times T_n(z)}{\sqrt{1+0.77z}}, \quad \text{shown in Figure 3.2 (e)} \tag{3.28}$$

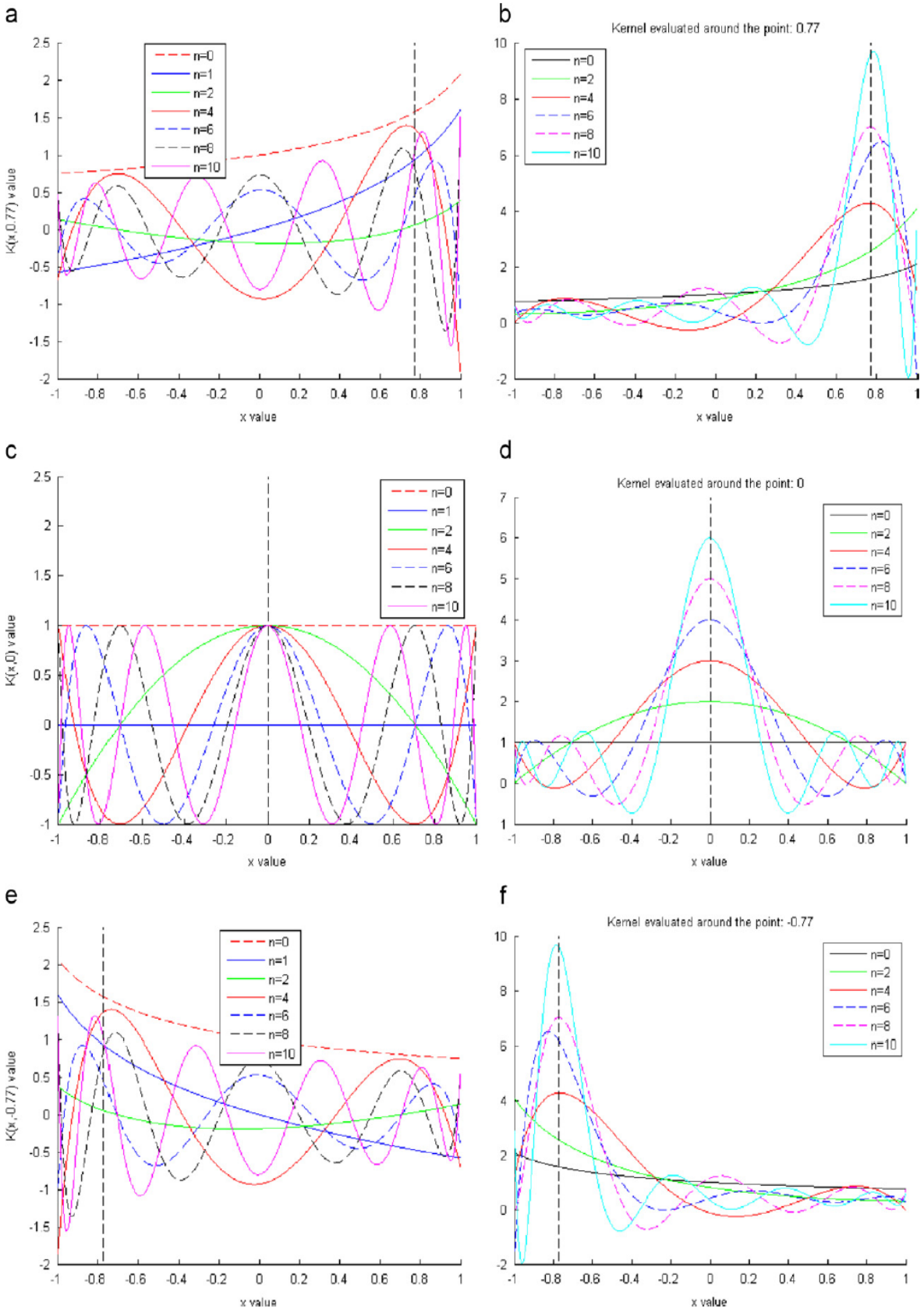


Figure 3.2 Plots of the generalized Chebyshev kernels evaluated with and without sum (b,d,f) as per Eq. (3.21) and (a,c,e) as per Eq. (3.22), respectively, at the three fixed *x*-values of +0.77, 0, -0.77, as reported by Ozer et al. [17].

However, as both of the kernel's input arguments x and z are real-valued scalar quantities that can take any value within the normalized range [-1,+1], it is still possible to investigate how the kernel value changes in 3D shape with respect to both of these two real-valued variables, as reported for example by Qu et al. [109] for the generalized Chebyshev kernel illustrated in Figure 3.3 (although it is not clear what polynomial order this kernel is). At the same time, it is much easier to analyze the shape characteristics of these kernels from their 2D plots in the way illustrated by Ozer et al.

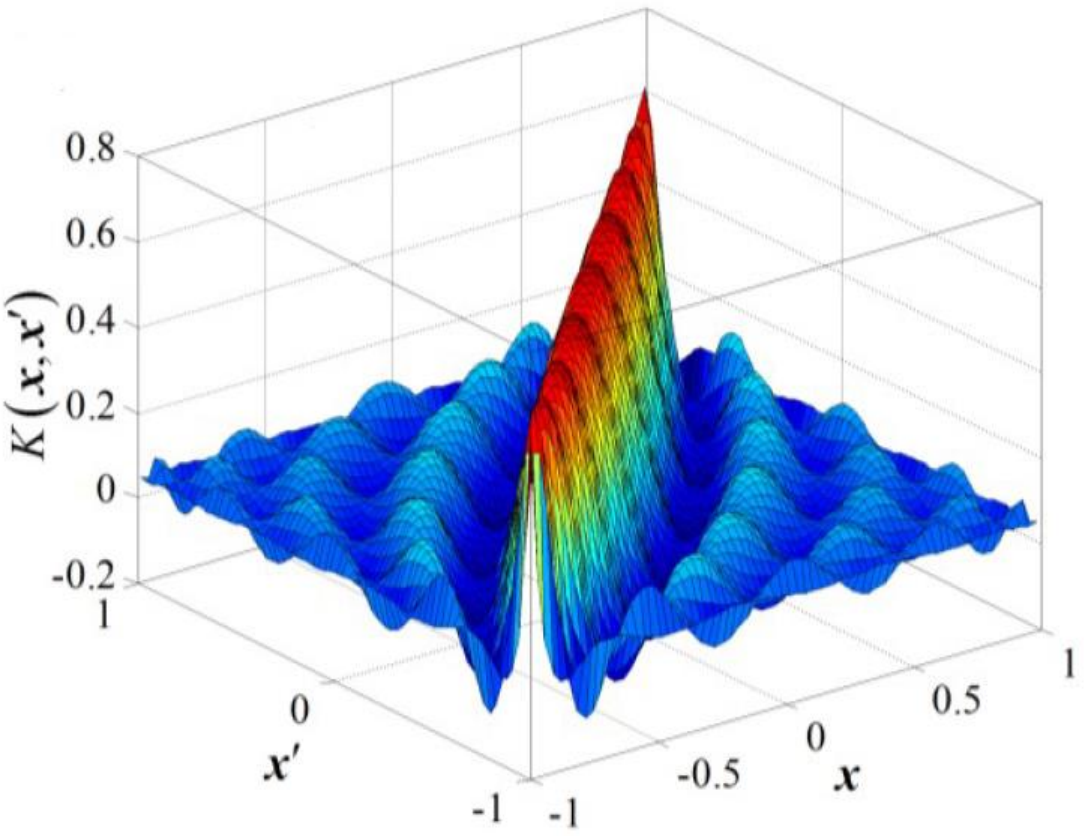
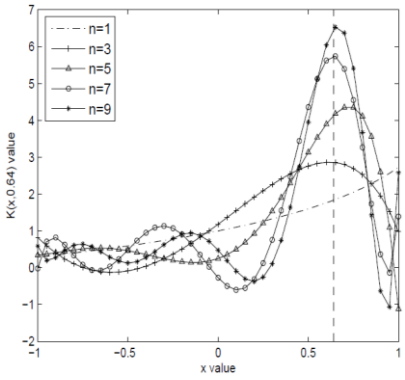
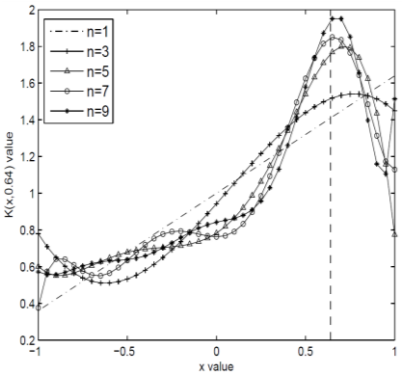


Figure 3.3 3D illustration of the generalized Chebyshev kernel, as reported by Qu et al. [109].

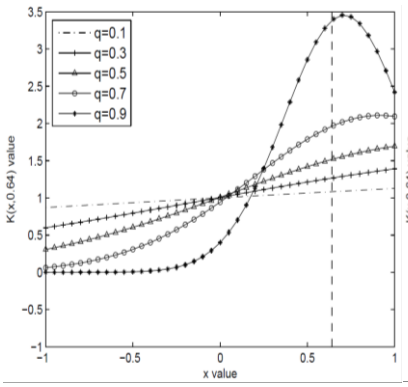
By following the same 2D plotting procedures used in [17] for the generalised Chebyshev kernels, Tian and Wang [108] have also recently extended this work to plot some other one-dimensional orthogonal polynomial kernels (constructed with the sum) that have been previously proposed, as illustrated in Figure 3.4. While the kernels' input arguments are assumed to be scalars, one can therefore notice that the shape of the kernels constructed using the pairwise approach is the same as those constructed using the vectorial approach. For example, this can be easily observed from the shape of the generalised Chebyshev kernels in Figure 3.4 (a) and the Chebyshev kernels in Figure 3.4 (h). Similarly, the generalised Legendre kernels in Figure 3.4 (b) are also the same as the Legendre kernels in Figure 3.4 (i).



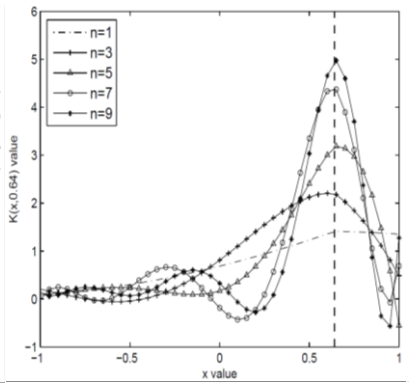
(a) Generalized Chebyshev kernel, $k_{G\text{-}Che}(x,0.64)$, as per (3.9)



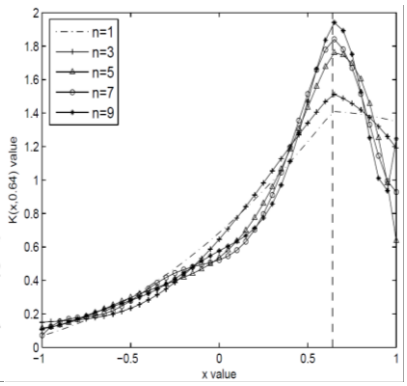
(b) Generalized Legendre kernel, $k_{G\text{-}Leg}(x,0.64)$, as per (3.10)



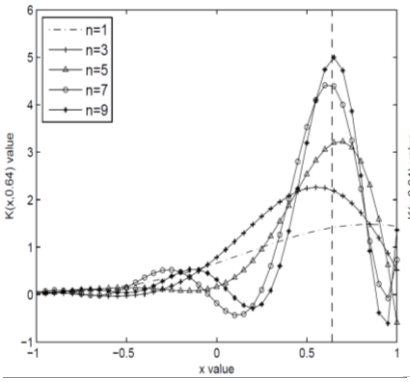
(c) Hermite kernel, $k_{Her}(x,0.64)$, as per (3.3)



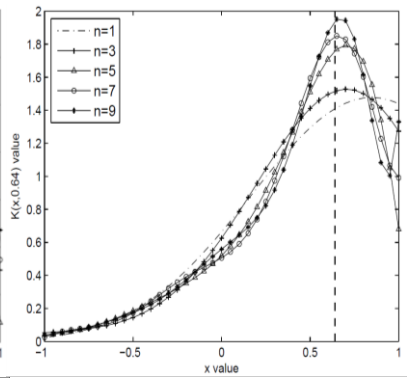
(d) Triangul. modified Chebyshev, $k_{Tri\text{-}Che}(x,0.64)$, as per (3.18)



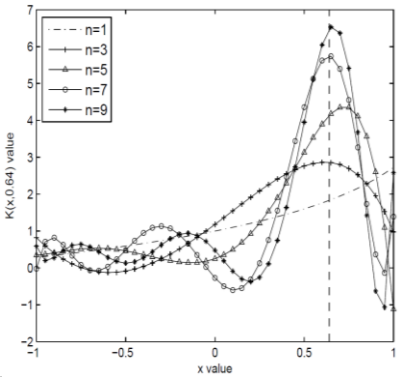
(e) Triangul. modified Legendre, $k_{Tri\text{-}Leg}(x,0.64)$, as per (3.19)



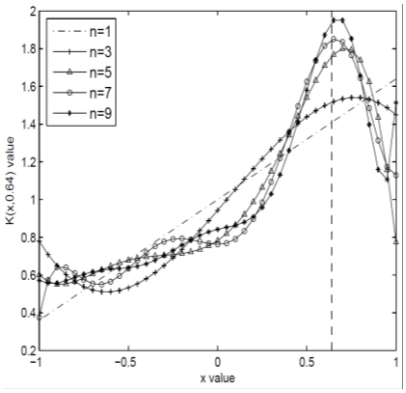
(f) Exp. modified Chebyshev, $k_{Exp\text{-}Che}(x,0.64)$, as per (3.11)



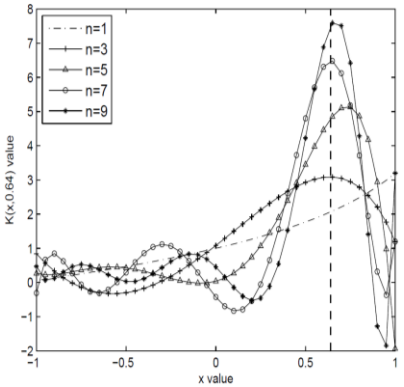
(g) Exp. modified Legendre, $k_{Exp\text{-}Leg}(x,0.64)$, as per (3.12)



(h) Chebyshev kernel, $k_{Che}(x,0.64)$, as per (3.6)



(i) Legendre kernel, $k_{Leg}(x,0.64)$, as per (3.6)



(j) Unified Chebyshev kernel, $k_{U\text{-}Che}(x,0.64)$, as proposed in [107].

Figure 3.4 Comparison of ten different orthogonal polynomial kernels, as reported by [108].

This is basically the technique that Ozer et al. [17] and Tian and Wang [108] have both used to plot in 2D some of the previously proposed OPKs, as illustrated in Figures 3.2 and 3.4. Therefore, this chapter will also follow the same procedure to investigate in more detail the shape charateristics of these polynomials kernels, especially those that are constructed with and without the sum. However, for a fair comparison and analysis of the shape characteristics of the different polynomial kernels, it is important to first eliminate the weighting function from the equation so that all the polynomial kernels are fairly compared to each other under their 'unweighted' status (i.e., without being affected by any combining or corresponding weighting functions).

For example, if we remove the weighting function from the generalized Chebyshev kernels for vector inputs in (3.9) and (3.20), they can then be redefined as:

$$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \left\langle T_i(\mathbf{x}), T_i(\mathbf{z}) \right\rangle \tag{3.29}$$

$$k(\mathbf{x},\mathbf{z}) = \left\langle T_n(\mathbf{x}), T_n(\mathbf{z}) \right\rangle \tag{3.30}$$

And by following the same assumption of scalar inputs above (to enable plotting them in 2D), they can then be defined as:

$$k(x,z) = \sum_{i=0}^{n} T_i(x) \times T_i(z) \tag{3.31}$$

$$k(x,z) = T_n(x) \times T_n(z) \tag{3.32}$$

And again, if we affix the *x*-value to 0, these Chebyshev kernels can be evaluated as:

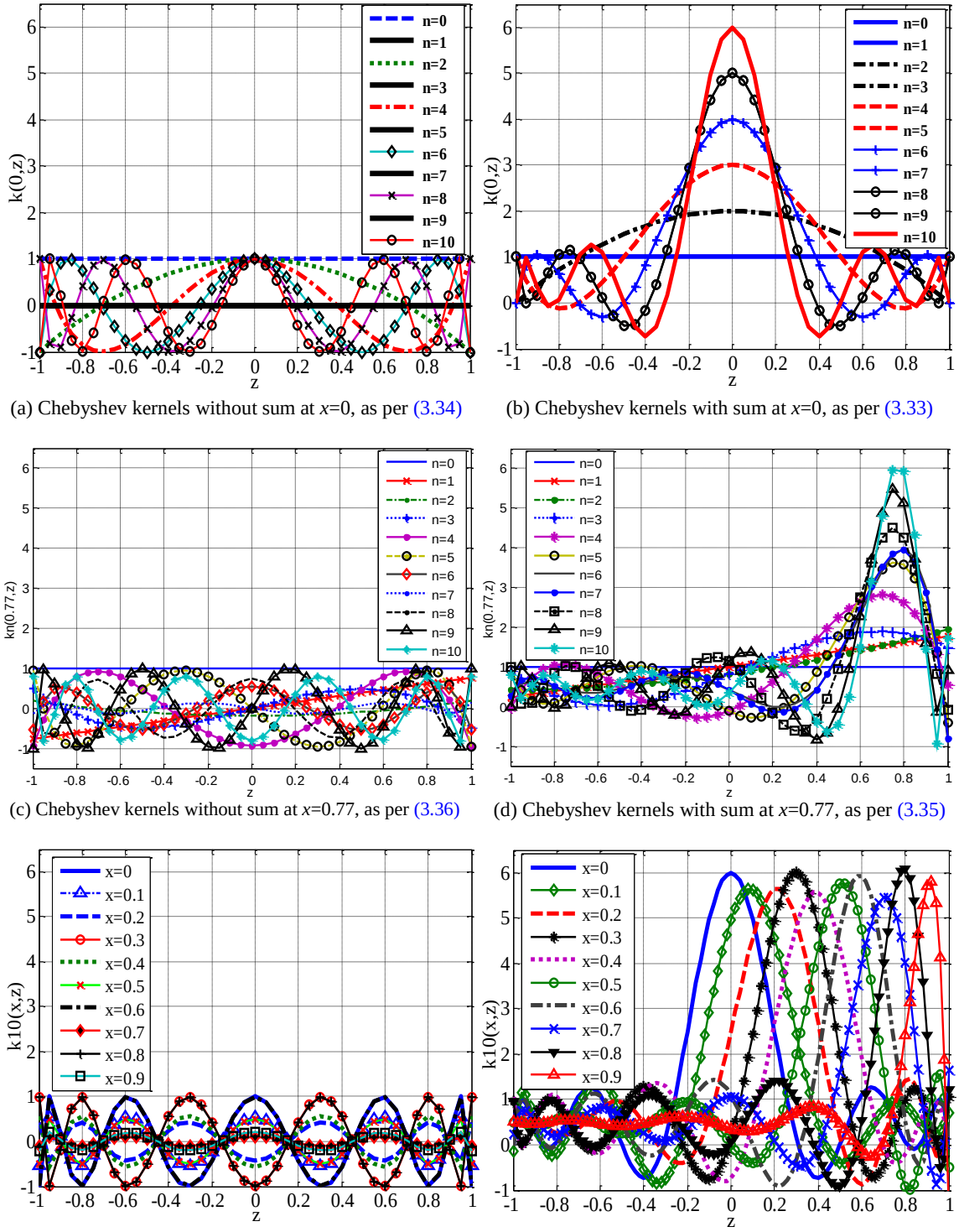
$$k(0,z) = \sum_{i=0}^{n} T_i(0) \times T_i(z), \quad \text{shown in Figure 3.5 (b)} \tag{3.33}$$

$$k(0,z) = T_n(0) \times T_n(z), \quad \text{shown in Figure 3.5 (a)} \tag{3.34}$$

Similarly, if we affix the *x*-value at +0.77, the Chebyshev kernels can be evaluated as:

$$k(0.77,z) = \sum_{i=0}^{n} T_i(0.77) \times T_i(z), \quad \text{shown in Figure 3.5 (d)} \tag{3.35}$$

$$k(0.77,z) = T_n(0.77) \times T_n(z), \quad \text{shown in Figure 3.5 (c)} \tag{3.36}$$



(a) Chebyshev kernels without sum at x=0, as per (3.34)

(b) Chebyshev kernels with sum at x=0, as per (3.33)

(c) Chebyshev kernels without sum at x=0.77, as per (3.36)

(d) Chebyshev kernels with sum at x=0.77, as per (3.35)

(e) 10th order Chebyshev kernel without sum at different x-values, as per (3.38)

(f) 10th order Chebyshev kernel with sum at different x-values, as per (3.37)

Figure 3.5 Illustration of the shape characteristics of the Chebyshev kernels without the weighting function formulated with and without the sum at different polynomial orders and x-values.

For vector inputs, however, it can be seen from the above formulas that the kernel shape will be obtained by adding the shapes of the kernel outputs for each order over a multi-dimensional space. As the axes of this multi-dimensional space are essentially the individual vector components (i.e., the scalar features), the inference is that the kernel shape in the multi-dimensional vector space will also exhibit similar characteritics to those

plotted in the single-dimensional space. To keep the analysis of the kernels' shapes as simple as possible, this thesis assumes that the derived shape characteristics of the kernels plotted in case of single-dimensional vector inputs (i.e., scalars) are also valid kernel characteristics in case of multi-dimensional vector inputs.

Another important point to consider (especially for the work presented in the next chapter) is that the kernels evaluated with and without the sum (i.e., those kernels in (3.33) and (3.34), respectively) should be plotted at the same scale to easily observe the difference between their shape characteristics when compared to each other side by side. Unlike the plots reported by Ozer et al. [17], both of the Chebyshev kernels evaluated with and without the sum, illustrated in Figure 3.5, are plotted at the same scale to achieve this purpose.

As briefly pointed out by Ozer et al. [17], the generalized Chebyshev kernels do alter their shape based on the input values, and perhaps that is the reason why they were interested to show how their shape changes at the x-values of +0.77 and -0.77 from how their shape looked like at x=0. The Chebyshev kernels constructed without the weighting function also tend to follow the same rule; i.e., they alter their shape with the changes in the input x-value. This is demonstrated in Figures 3.5 (c) and (d) when the first few orders of the Chebyshev kernels without the weighting function are plotted at the x-value of +0.77, which clearly demonstrate that this alteration in the shape of the kernel is stemming from the employed Chebyshev polynomials and not from their corresponding weighting function. However, to further elaborate the effect of the variations in the x-value on the shape of the kernel, Figures 3.5 (f) and (e) also plot the Chebyshev kernel of order n=10 (as an illustrative example) with and without sum, respectively, at some other x-values of [0, 0.1, 0.2, …, 0.9], using the equations below:

$$k_{n=10}^{WithSum}(x,z)\Big|_{x=[0,0.1,\ldots,0.9]} = \sum_{i=0}^{10} T_i(x) \times T_i(z), \quad \text{shown in Figure 3.5 (f)} \tag{3.37}$$

$$k_{n=10}^{WithoutSum}(x,z)\Big|_{x=[0,0.1,\ldots,0.9]} = T_{10}(x) \times T_{10}(z), \quad \text{shown in Figure 3.5 (e)} \tag{3.38}$$

Having now focused on the polynomial element of the Chebyshev kernels (by eliminating the weighting function factor) and plotted their two versions evaluated with and without the sum at the same scale, we can now reflect upon their shape characteristics within the context of the definition of kernels as similarity measures explained earlier in Section 3.3.1 via comparing them to the shape of the ideal similarity measure kernel back in Figure 3.1.

First, it is important to clarify that the Chebyshev kernels illustrated in Figure 3.5 are plotted against their second input argument *z*, which is spanning the normalized region of [-1,+1], and at a fixed value of its first argument *x* (0, 0.77, etc.). This means that if the chosen fixed *x*-value coincides with *z* anywhere within the set region [-1,+1], this would then be the instant, where the two input kernel arguments are identical to each other, and accordingly, one should intuitively expect the value of the kernel to be at its maximum peak, based on the definition of the ideal similarity measure kernels explained earlier. Moreover, as the *z*-value departs away from the chosen fixed *x*-value, this means that the two input kernel arguments are becoming less and less similar to each other (and hence are more likely to be belonging to different classes) and accordingly one should also expect the value of the kernel to start decreasing in a monotonic behaviour.

If we analyse closely the shape characteristics of the Chebyshev kernels with sum in Figure 3.5 (b), we can actually observe that they exhibit some of these characteristics already as the order increases at x=0; i.e., they develop a maximum peak at the point, where the two input arguments are identical (i.e., at x=0 in this case), and then decrease as *z* departs from the chosen fixed value of *x*. More importantly, however, is the fact that the Chebyshev kernels with sum continue to develop this maximum peak wherever else in the normalized vector space where their two input arguments happen to be identical, as demonstrated for example in Figure 3.5 (d) and (f), as opposed to their counterparts constructed without sum, shown in Figure 3.5 (a), (c), and (e), which only tend to change their amplitude and polarity as the *x*-value changes, with no dominant maximum value (compared to the other kernel values within the normalized region of [-1,+1]) at the instances, where the two inputs happen to be identical to each other.

This demonstrates that the Chebyshev kernels with sum are better similarity measure tools than their counterparts without the sum, simply because they resemble more the shape characteristics of the ideal similarity measure kernel demonstrated back in Figure 3.1. They are however, not identical to it. For example, one can clearly observe that the Chebyshev kernels with sum in Figure 3.5 (b), (d) and (f) do not actually decrease monotonically as *z* departs away from the chosen fixed *x*-value. Instead, their decrease behaviour only lasts for up to a certain threshold after which the kernel value starts to increase again and then fluctuates in a wavy pattern, similar to the wave kernel shown in Figure 2.5 (f). In other words, the region where the Chebyshev kernels with sum exhibit a complete monotonic decrease behaviour is unfortunately smaller than the normalized data region of [-1,+1], meaning that if the data points happen to be located outside this monotonic window (dictated by the threshold of each kernel), this can result in the kernel calculating an

incorrect similarity measure, which can accordingly misclassify the input data; a problem which will be addressed later on in Chapter 5.

On the other hand, recall that the Gaussian kernel, when the data are vectors, is defined as:

$$k(\mathbf{x},\mathbf{z}) = \exp\left(-\gamma\|\mathbf{x}-\mathbf{z}\|^2\right) \tag{3.39}$$

whereas when the data are scalars, it can be written as:

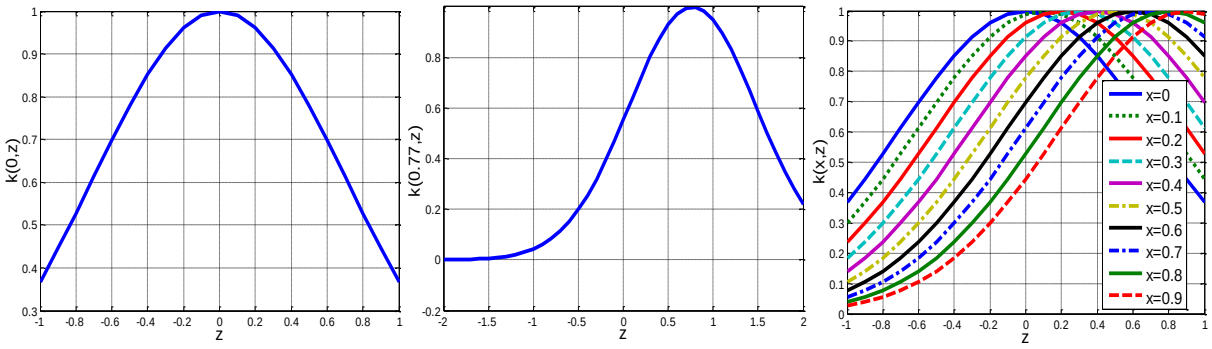
$$k(x,z) = \exp\left(-\gamma(x-z)^2\right) \tag{3.40}$$

where one can now can plot it in 2D in exactly the same way as we did before with the Chebyshev kernels, within the range of z=[-1,+1], to investigate how its shape is affected by the variations in the input x-values, using the equations below which are plotted in Figure 3.6.

$$k(0,z) = \exp\left(-\gamma(0-z)^2\right), \quad \text{shown in Figure 3.6 (a)} \tag{3.41}$$

$$k(0.77,z) = \exp\left(-\gamma(0.77-z)^2\right), \quad \text{shown in Figure 3.6 (b)} \tag{3.42}$$

$$k(x,z)\big|_{x=[0,0.1,..0.9]} = \exp\left(-\gamma(x-z)^2\right), \quad \text{shown in Figure 3.6 (c)} \tag{3.43}$$



(a) Gaussian kernel at x=0, as per (3.41) (b) Gaussian kernel at x=0.77, as per (3.42) (c) Gaussian kernel at x = [0, 0.1,...,0.9], as per (3.43)

Figure 3.6 Effect of the variation in the input x-value on the shape of the Gaussian kernel.

One can notice that the shape of the Gaussian kernel is a typical example of the definition of the ideal similarity function (as explained earlier in Section 3.3.1), because it clearly develops a peak whenever its two input arguments happen to be identical anywhere within the normalized vector space, as well as decreases monotonically when they depart away from each other. Recall also that the Gaussian kernel amounts to mapping the input space to an infinite-dimensional feature space [1, 12], where the linear separability of overlapped classes can be improved. So, possessing these two important characteristics is perhaps the reason behind the good performance and popularity of this kernel in solving a variety of pattern classification tasks. It can also be noticed from the Gaussian shapes in Figure 3.6 that it is not structurally altered by the variations in the input x-values and it is rather just a

transitional change dictated by the lag vector $\|\mathbf{x}-\mathbf{z}\|$. Hence, the only parameter that can structurally alter the shape of the Gaussian kernel is its associated parameter γ which was kept constant at $\gamma=1$ for all the plots illustrated in Figure 3.6, for simplicity.

Using the same procedure, we can also plot some of the other widely used kernels, such as the linear and traditional polynomial kernels, to analyse how good their calculated similarity measures are when compared to the shape characteristics exhibited by either the Gaussian kernel above or the ideal similarity function demonstrated back in Figure 3.1. Starting by the simplest linear kernel, defined by the dot-product as per (2.39), Figure 3.7 demonstrates its shape characteristics at various *x*-values within the normalized vector space. As indicated by its name, one can notice that its shape demonstrates a basic linear behaviour and the variations in the *x*-value only change its slope, with no peaks developed at all at the instances where the two input arguments happen to be identical to each other, as is the case for example with the Gaussian and Chebyshev (with sum) kernels. Moreover, one can also notice that its shape is far from being monotonically decreasing (in fact it can actually increase instead), when the two input arguments depart away from each other.

As such, based on what one would intuitively wish a similarity function to pictorially look like when calculating the similarity between two vectors, we cannot admit that the linear kernel provides a good indication as to how two vectors resemble each other, whether they are identical or not, and therefore would argue the fact that it can actually be regarded as a good similarity measure tool. Furthermore, recall also that it does not amount to any mapping to higher number of dimensions in the feature space, as explained earlier in Section 2.5.1, it is also therefore not a suitable candidate kernel to be used to solve non-linearly separable classification problems.

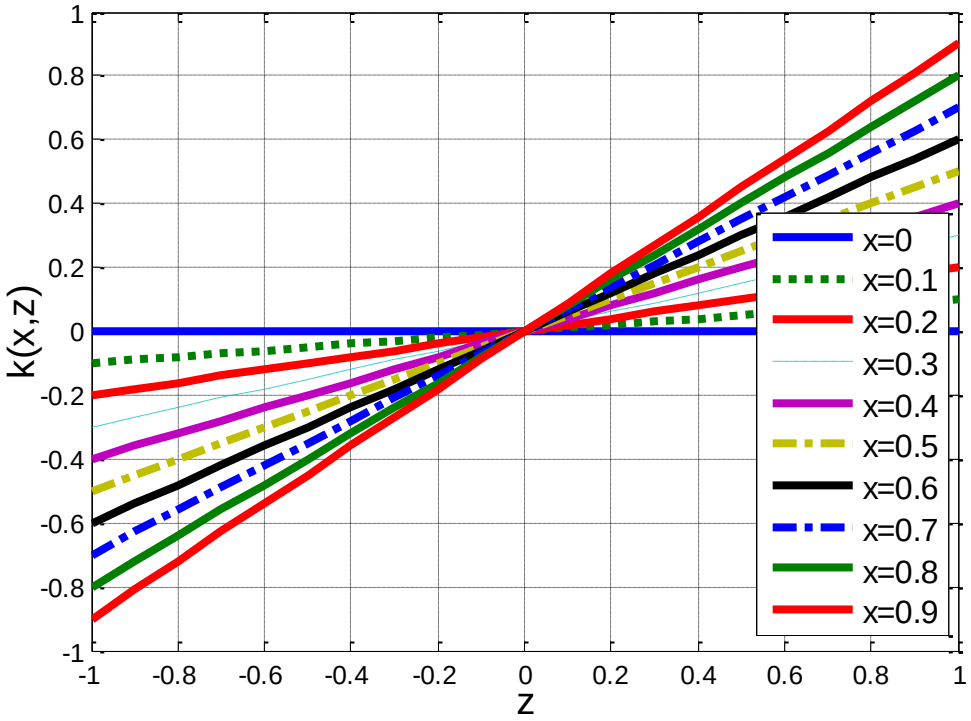
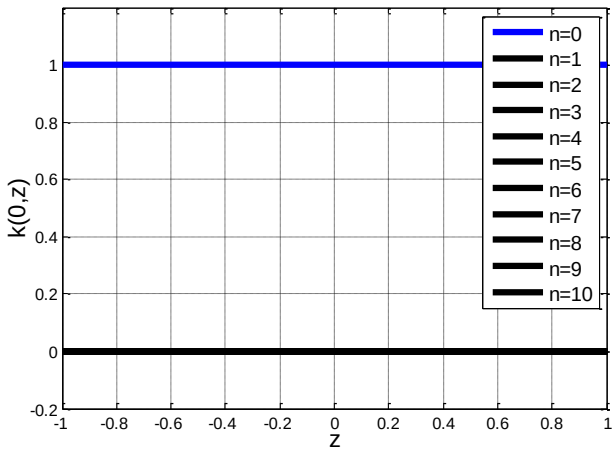
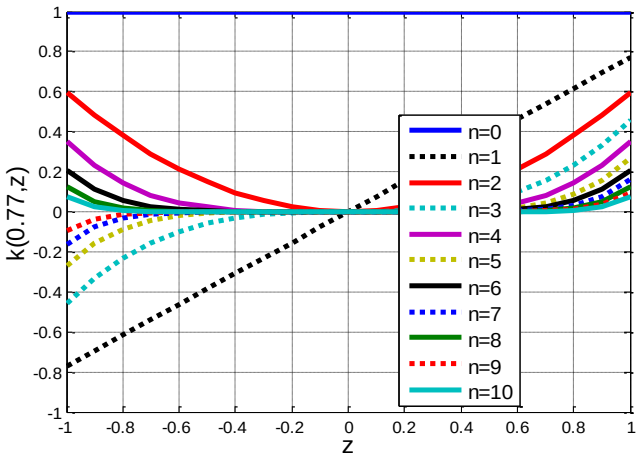


Figure 3.7 Analysis of the shape characteristics of the linear kernel at different *x*-values of *x*=0, 0.1, …, 0.9.

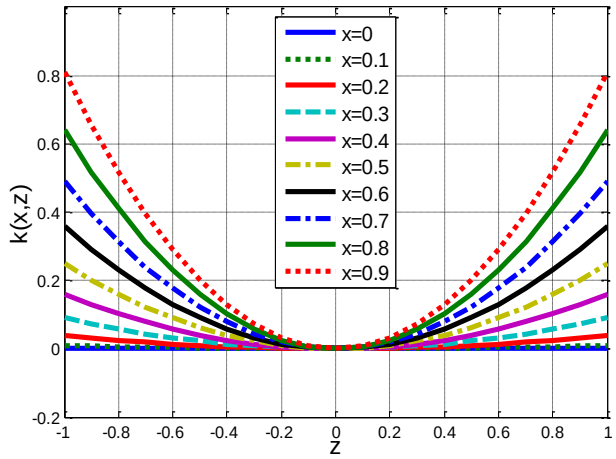
Two other types of traditional polynomial kernels have also been in common use. These are the homogeneous and inhomogeneous polynomial kernels, defined by (2.40) and (2.41), respectively. The shape characteristics of these traditional polynomial kernels have also been studied in this thesis via Figures 3.8 and 3.9, respectively, for the first 10 orders and at various *x*-values. Although the higher orders of such polynomial kernels amount to higher number of dimensions in the feature space [9], which can be high enough to establish a linear separation between classes in some datasets, yet, one can still observe that both of them were also not able to exhibit the sought after shape characteristics of the ideal similarity function, neither when the two input arguments are identical nor when they are different from each other.



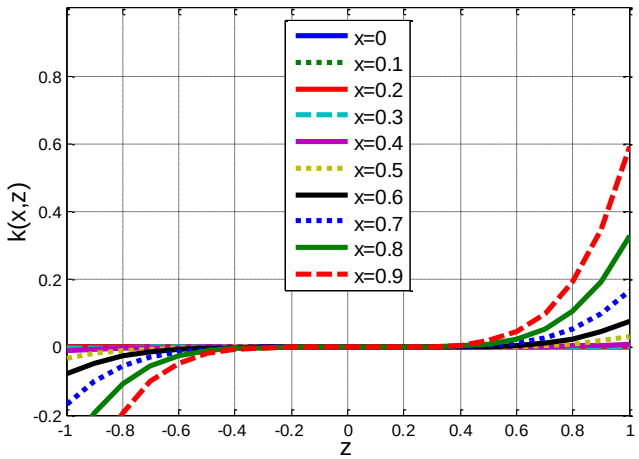
(a) First 10 orders of the homogeneous polynomial kernel, as per (2.40), at x=0.



(b) First 10 orders of the homogeneous polynomial kernel, as per (2.40), at x=0.77.

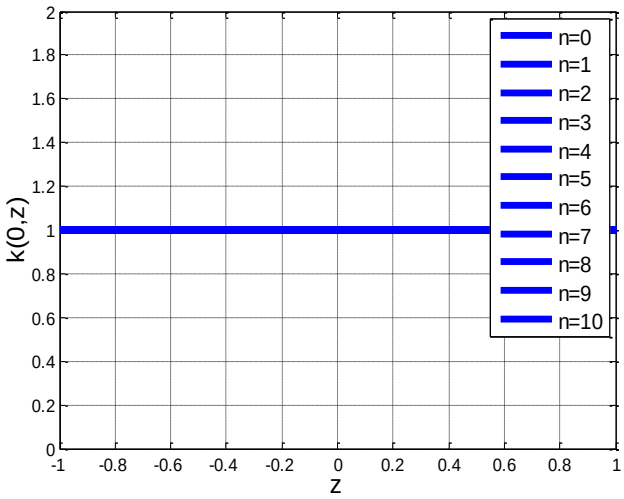


(c) 2nd order homogeneous polynomial kernel, as per (2.40), at various x-values = 0, 0.1, …, 0.9.

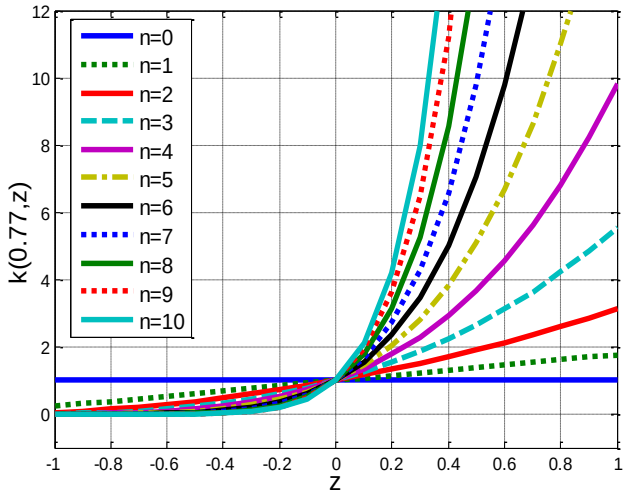


(d) 5th order homogeneous polynomial kernel, as per (2.40), at various x-values = 0, 0.1, …, 0.9.

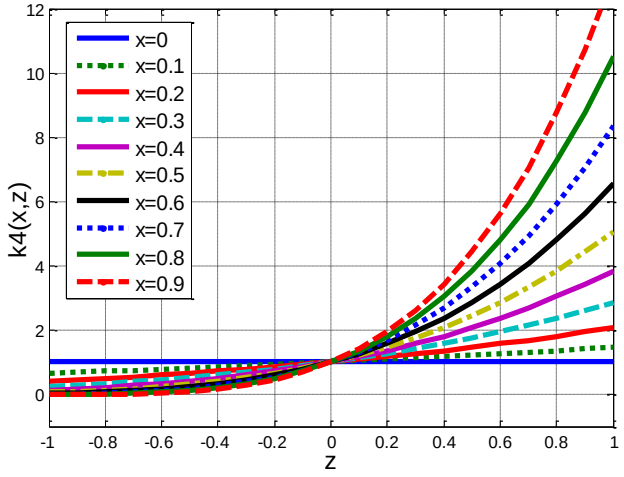
Figure 3.8 Analysis of the shape characteristics of homogeneous polynomial kernels at various orders and x-values.



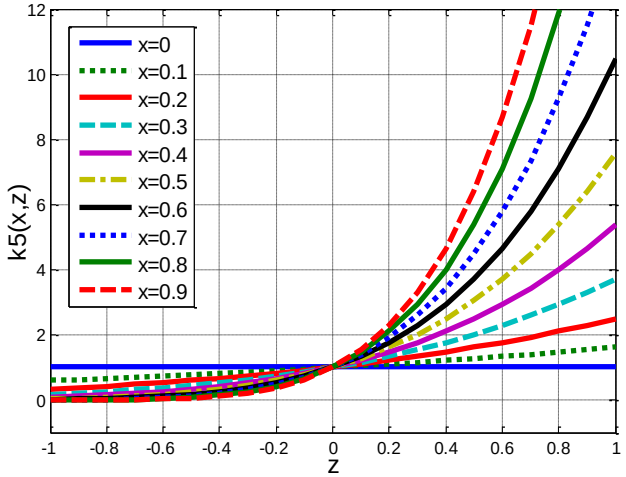
(a) First 10 orders of the inhomogeneous polynomial kernel, as per (2.41), at x=0.



(b) First 10 orders of the inhomogeneous polynomial kernel, as per (2.41), at x=0.77.



(c) 4th order inhomogeneous polynomial kernel, as per (2.41), at various x-values = 0, 0.1, …, 0.9.



(d) 5th order inhomogeneous polynomial kernel, as per (2.41), at various x-values = 0, 0.1, …, 0.9.

Figure 3.9 Analysis of the shape characteristics of the inhomogeneous polynomial kernels at various orders and x-values.

3.4.2.2 Legendre polynomial kernels

To further investigate the shape characteristics of orthogonal polynomial kernels based on their similarity measure properties, it was a natural practice to extend Ozer et al.'s work [17] to conduct a comparative graphical analysis with some other families of orthogonal polynomials, rather than being confined to only the Chebyshev polynomials. This section explores the shape characteristics of the kernels constructed from the Legendre polynomials[12] using the same plotting procedures followed in the previous sub-section for the Chebyshev polynomials, for consistency. Recall that the Legendre polynomials,

[12] Legendre polynomials are named after Adrien Marie Legendre (1752-1833), and in mathematics, they are most commonly used as solutions to Legendre differential equations [112]. They are also frequently encountered in physics and other technical fields, in particular when solving Laplace's equation in spherical coordinates [112].

denoted by $L_n(x)$, are characterised by their unity weighting function $w(x)=1$, and as such, can be simpler to implement when used to construct SVM kernels and computationally faster than their Chebyshev counterparts [34]. Table 3.1 lists the first few orders of the Legendre polynomials, however, as with most other orthogonal polynomials, given the first two polynomials in the family, one can construct a recursive formula from which any other polynomial order can be generated, as follows [112]:

Given that the first two orders of the Legendre polynomials are:

$L_0(x) = 1$, and

$L_1(x) = x$,

one can produce any other Legendre polynomial order using the following recursion formula:

$$nL_n(x) = (2n-1)xL_{n-1}(x) - (n-1)L_{n-2}(x) \tag{3.44}$$

Following the vectorial processing approach proposed by Ozer et al. (and explained earlier in Section 3.4.1.2), one can define the SVM kernels constructed from the Legendre polynomials for vector inputs (with and without sum, respectively) as:

$$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \left\langle L_i(\mathbf{x}), L_i(\mathbf{z}) \right\rangle \tag{3.45}$$
$$k(\mathbf{x},\mathbf{z}) = \left\langle L_n(\mathbf{x}), L_n(\mathbf{z}) \right\rangle \tag{3.46}$$

However, similar to the plotting procedures followed with the Chebyshev kernels, it is also necessary to assume that the inputs are scalars to facilitate the plotting of the Legendre kernels in 2D. As such, (3.45) and (3.46) can then be written for scalar inputs as:

$$k(x,z) = \sum_{i=0}^{n} L_i(x) \times L_i(z) \tag{3.47}$$
$$k(x,z) = L_n(x) \times L_n(z) \tag{3.48}$$

And now we can plot the first few orders of these Legendre kernels at x=0 using:

$$k(0,z) = \sum_{i=0}^{n} L_i(0) \times L_i(z), \quad \text{shown in Figure 3.10 (b)} \tag{3.49}$$
$$k(0,z) = L_n(0) \times L_n(z), \quad \text{shown in Figure 3.10 (a)} \tag{3.50}$$

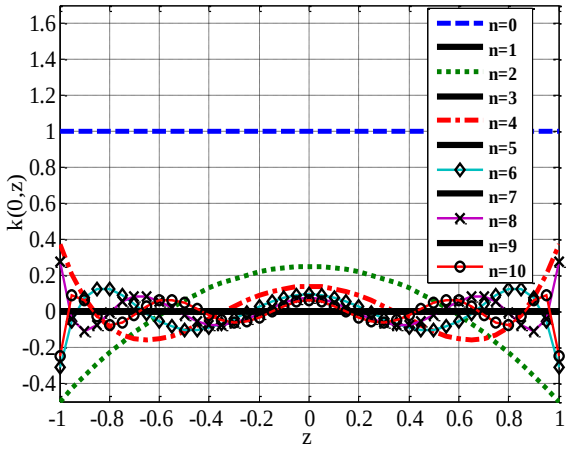
And at x=0.77 using:

$$k(0.77,z) = \sum_{i=0}^{n} L_i(0.77) \times L_i(z), \quad \text{shown in Figure 3.10 (d)} \tag{3.51}$$
$$k(0.77,z) = L_n(0.77) \times L_n(z), \quad \text{shown in Figure 3.10 (c)} \tag{3.52}$$

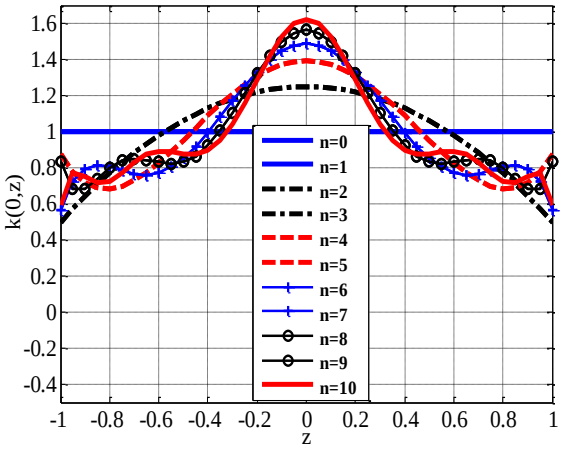
And the 10th order (as an illustrative example) Legendre kernel at various x-values using:

$$k_{n=10}^{WithSum}(x,z)\Big|_{x=[0,0.1,\ldots,0.9]} = \sum_{i=0}^{10} L_i(x) \times L_i(z), \quad \text{shown in Figure 3.10 (f)} \qquad (3.53)$$

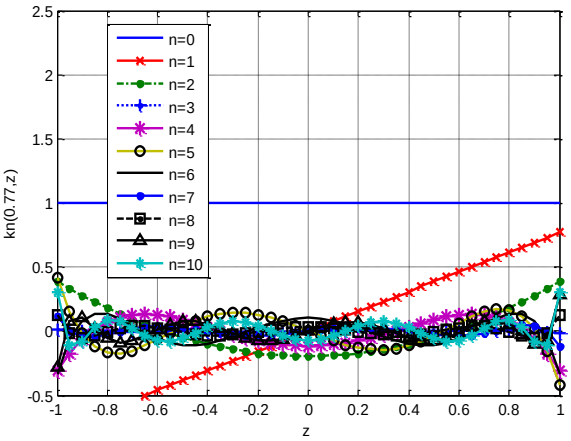
$$k_{n=10}^{WithoutSum}(x,z)\Big|_{x=[0,0.1,\ldots,0.9]} = L_{10}(x) \times L_{10}(z), \quad \text{shown in Figure 3.10 (e)} \qquad (3.54)$$



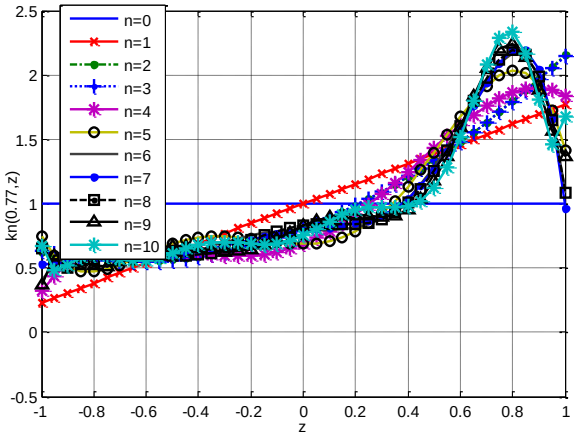
(a) Legendre kernels without sum at x=0, as per (3.50)



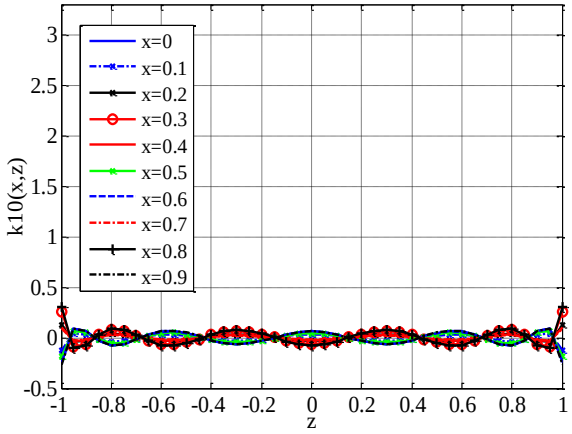
(b) Legendre kernels with sum at x=0, as per (3.49)



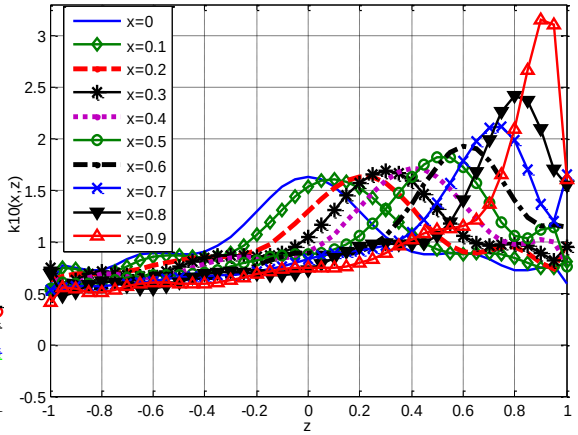
(c) Legendre kernels without sum at x=0.77 as per (3.52)



(d) Legendre kernels with sum at x=0.77 as per (3.51)



(e) 10th order Legendre kernel without sum at different x-values, as per (3.54)



(f) 10th order Legendre kernel with sum at different x-values, as per (3.53)

Figure 3.10 Illustration of the shape characteristics of the Legendre kernels formulated with and without the sum at different polynomial orders and x-values.

On analysing the Legendre kernels illustrated in Figure 3.10, one can observe that they actually share some common shape characteristics with their Chebyshev counterparts

demonstrated in the previous sub-section. For example, the Legendre kernels with sum also develop a maximum peak value anywhere in the normalized region [-1,+1], where the two input arguments happen to be identical, as shown in Figure 3.10 (b), (d) and (f), as opposed to the Legendre kernels without the sum, shown in Figure 3.10 (a), (c) and (e), where they only tend to change their amplitude and polarity with no dominant maximum value within the normalized region of [-1,+1] at the instances where the two inputs happen to be identical.

This means that the Legendre kernels with sum are also expected to be better similarity measure tools than their counterparts without the sum, as their shape characteristics get closer to the ideal similarity function as the polynomial order increases. On the other hand, similar to the Chebyshev kernels, the Legendre kernels with sum have shown not to exhibit a complete monotonic decrease behaviour within the region [-1,+1] as z departs away from the chosen fixed x-value; a problem which will again be dealt with in Chapter 5.

3.4.2.3 Hermite polynomial kernels

This section complements the investigation on the similarity-based properties of orthogonal polynomial kernels via the analysis of the shape characteristics of the physicists' Hermite polynomials. Although some previous work, such as [102] and [35], have studied the construction of SVM kernels from Hermite polynomials in the past, their shape characteristics have not been investigated and analysed before in the way shown in this section. Usually denoted by $H_n(x)$, where n is the polynomial order, the physicists' Hermite polynomials are defined over the domain $x=[-\infty,\infty]$ where they are orthogonal to each other with respect to the Gaussian weighting function $w(x)=e^{-x^2}$. Table 3.1 shows the first few orders of the Hermite polynomials, and they can also be generated from the first two polynomials using the following recursive formula.

Given that

$H_0(x) = 1$, and

$H_1(x) = 2x$,

Any order of Hermite polynomial can be generated using the following recursive formula (reformulated from [113]):

$$H_n(x) = 2x\,H_{n-1}(x) - 2(n-1)\,H_{n-2}(x) \tag{3.55}$$

Again, whilst focusing only on the polynomial element of the kernel, to eliminate the influence of any weighting function, and by following the same vectorial processing approach proposed by Ozer et al. [17], one can define the Hermite polynomial kernels for vector inputs (with and without sum, respectively) as:

$$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle H_i(\mathbf{x}), H_i(\mathbf{z}) \rangle \tag{3.56}$$

$$k(\mathbf{x},\mathbf{z}) = \langle H_n(\mathbf{x}), H_n(\mathbf{z}) \rangle \tag{3.57}$$

And again, to plot them in 2D, (3.56) and (3.57) can be defined for scalar inputs as:

$$k(x,z) = \sum_{i=0}^{n} H_i(x) \times H_i(z) \tag{3.58}$$

$$k(x,z) = H_n(x) \times H_n(z) \tag{3.59}$$

Table 3.3 illustrates the plots of the first 10 orders of these Hermite polynomial kernels at x=0 using:

$$k(0,z) = \sum_{i=0}^{n} H_i(0) \times H_i(z) \tag{3.60}$$

$$k(0,z) = H_n(0) \times H_n(z) \tag{3.61}$$

whereas Table 3.4 illustrates their plots at x=0.77 using:

$$k(0.77,z) = \sum_{i=0}^{n} H_i(0.77) \times H_i(z) \tag{3.62}$$

$$k(0.77,z) = H_n(0.77) \times H_n(z) \tag{3.63}$$

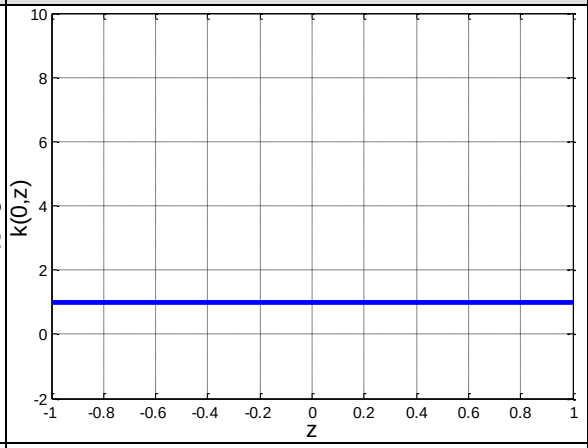
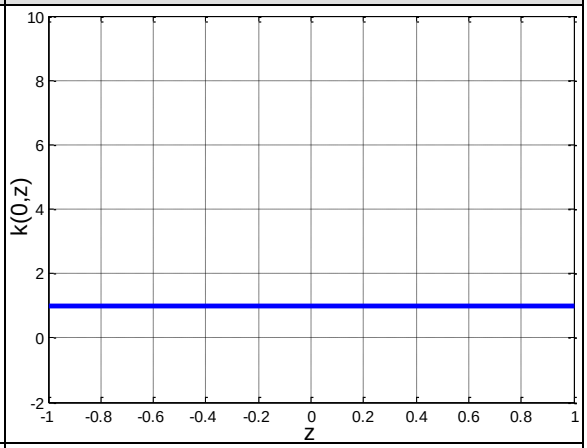
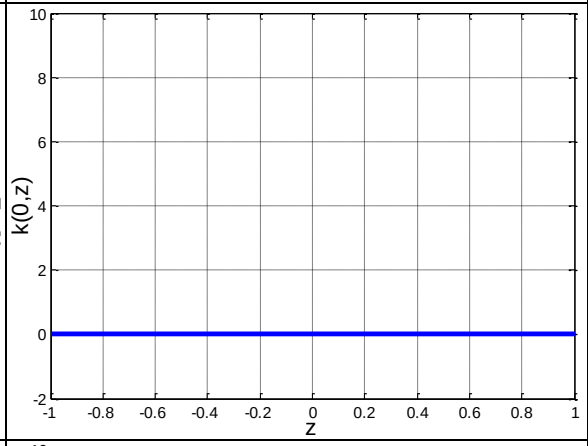
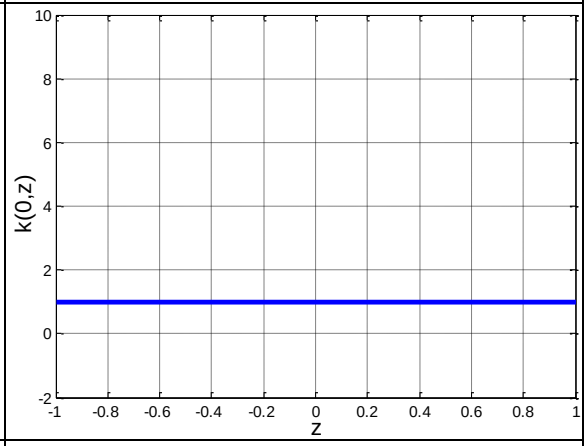
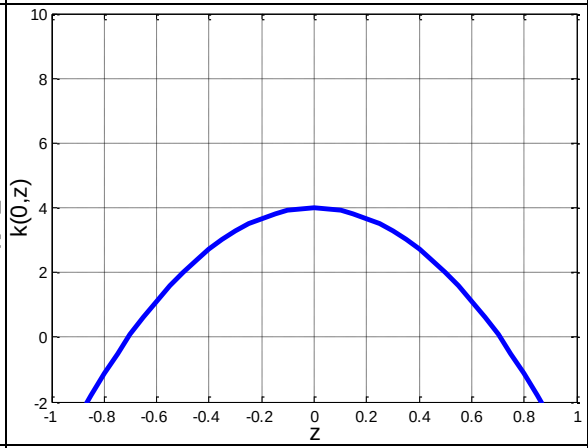
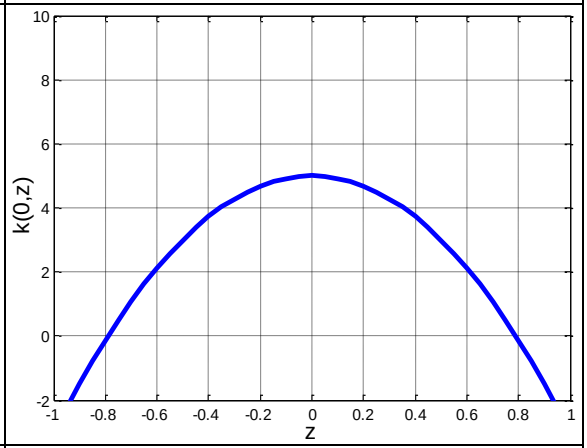
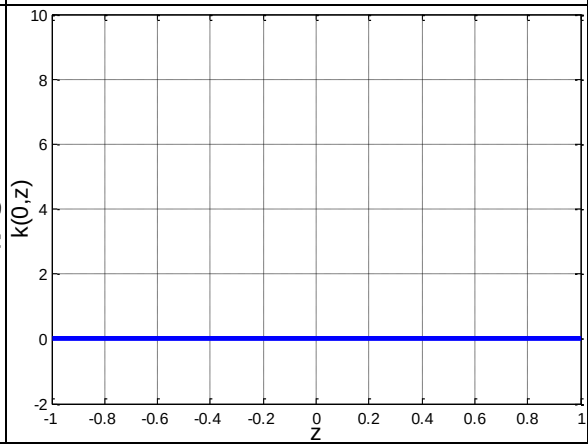
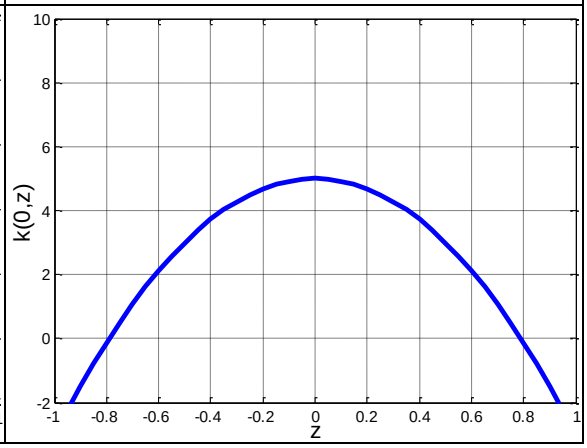
Figure 3.11 demonstrates the effect of the variation in the input x-value on the shape characteristics of the 10th order Hermite kernel (as an illustrative example) with and without sum using:

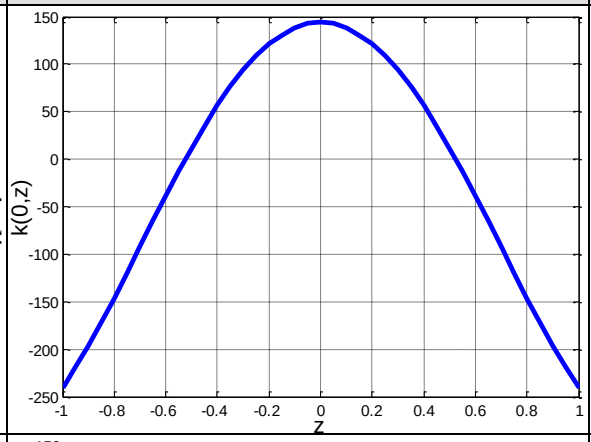
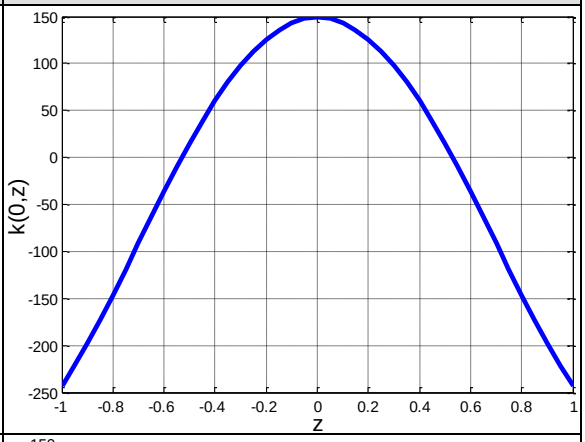
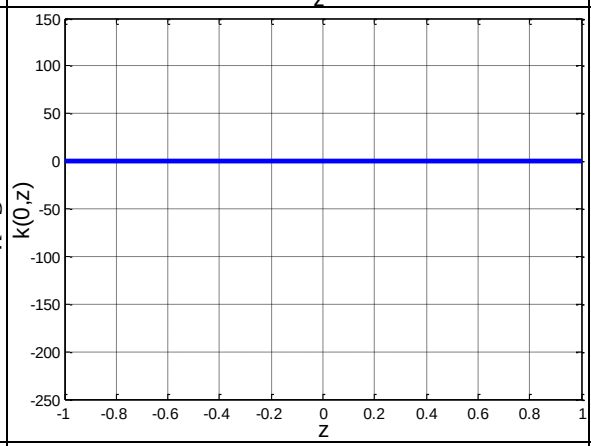
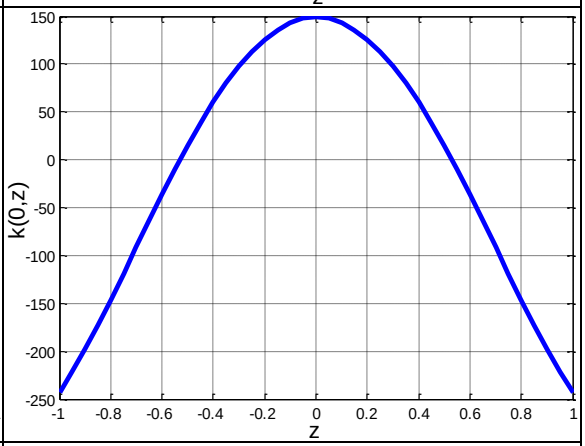
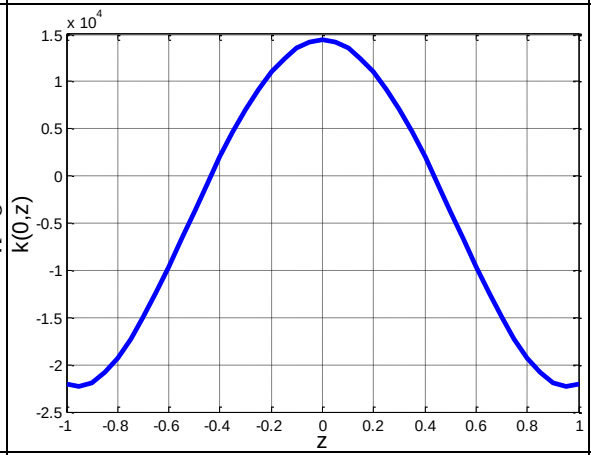
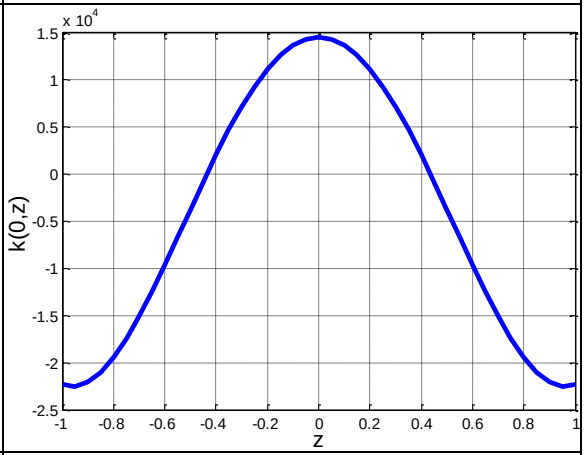
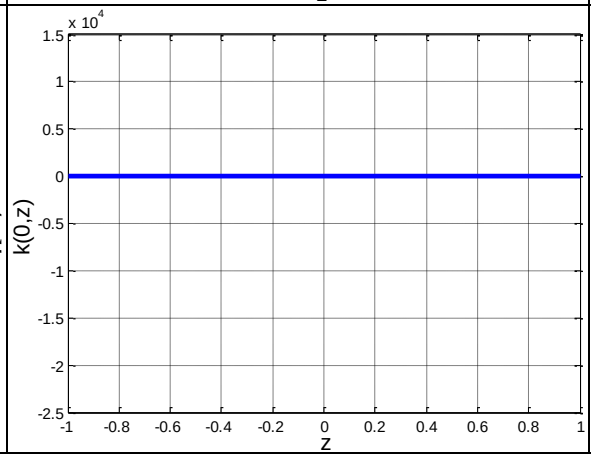
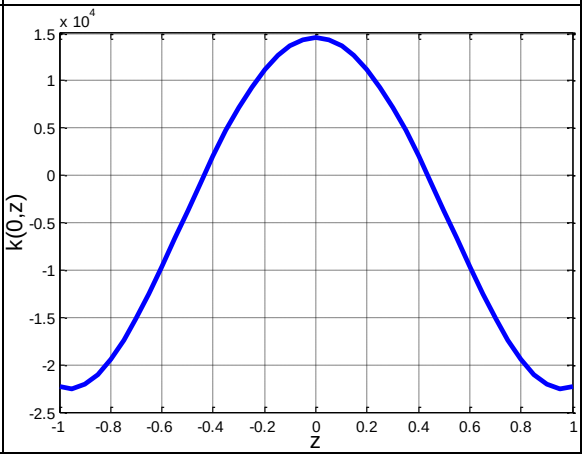
$$k_{n=10}^{WithSum}(x,z)\Big|_{x=[0,0.1,\ldots,0.9]} = \sum_{i=0}^{10} H_i(x) \times H_i(z) \tag{3.64}$$

$$k_{n=10}^{WithoutSum}(x,z)\Big|_{x=[0,0.1,\ldots,0.9]} = H_{10}(x) \times H_{10}(z) \tag{3.65}$$

One can notice, however, that the coefficients of the Hermite polynomials are quite sparse causing the amplitudes of the Hermite kernels to be significantly elevated from one order to another. For this reason, the first 10 orders of Hermite kernels in (3.60-3.63) have been plotted separately in Tables 3.3 and 3.4 to enable a clear analysis of their original shape characteristics, before they are combined in one figure at an absolute logarithmic scale at the bottom of each table.

Table 3.3 Comparison of the shape characteristics of the first 10 orders of the Hermite polynomial kernels with and without sum, as per (3.60) and (3.61) respectively, at x=0.

	Hermite kernels without sum, at x=0	**Hermite kernels with sum, at x=0**
n=0		
n=1		
n=2		
n=3		

	Hermite kernels without sum, at x=0	Hermite kernels with sum, at x=0
n=4		
n=5		
n=6		
n=7		

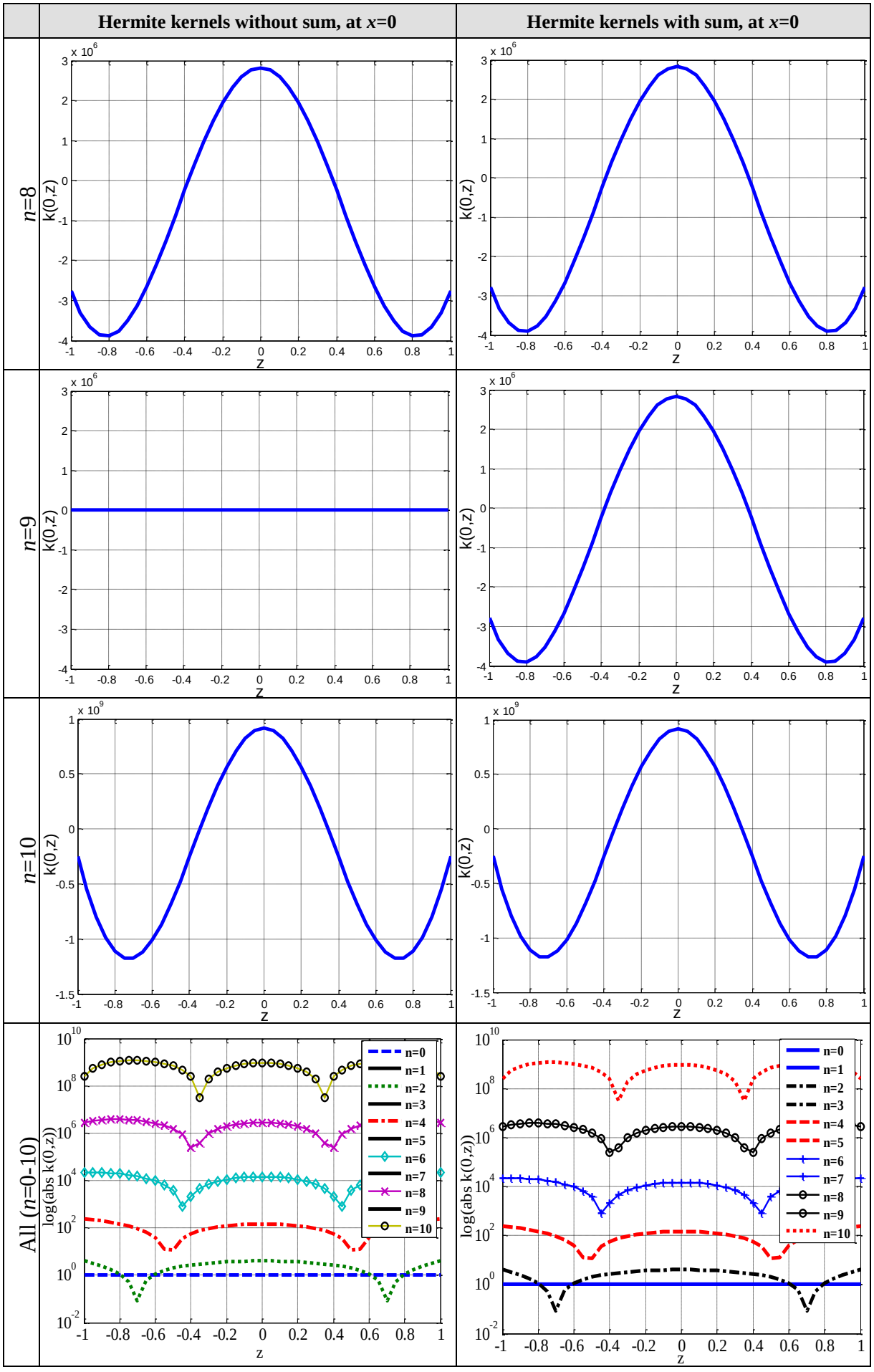
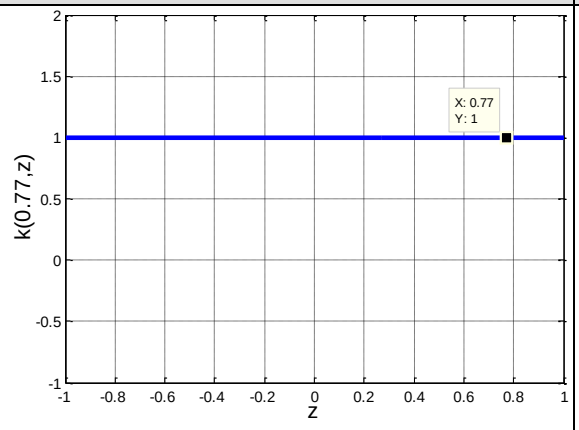
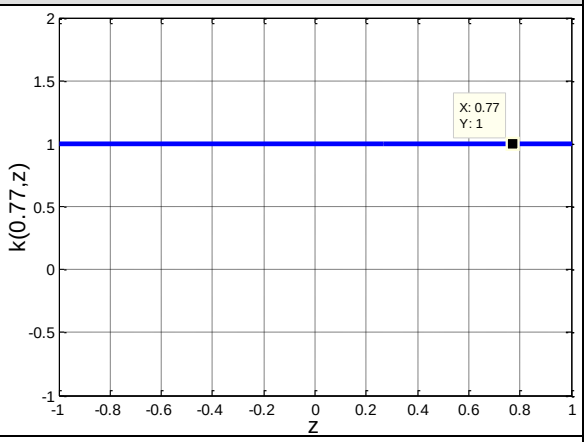
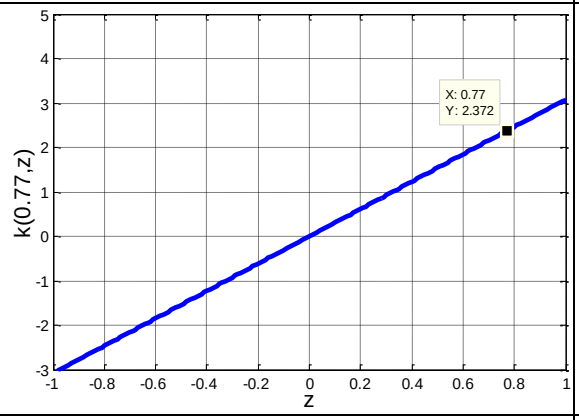
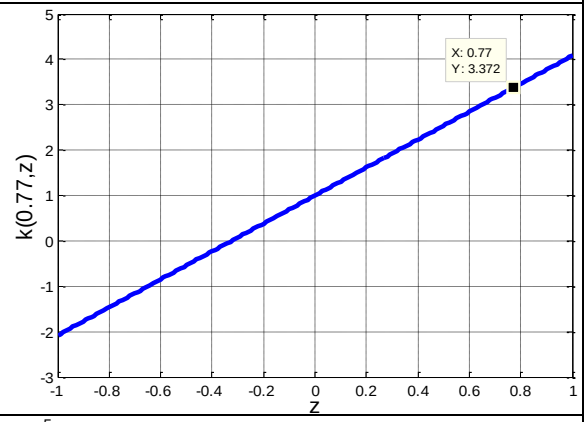
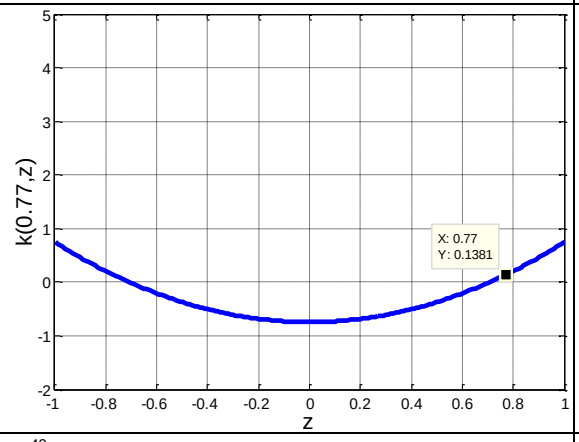
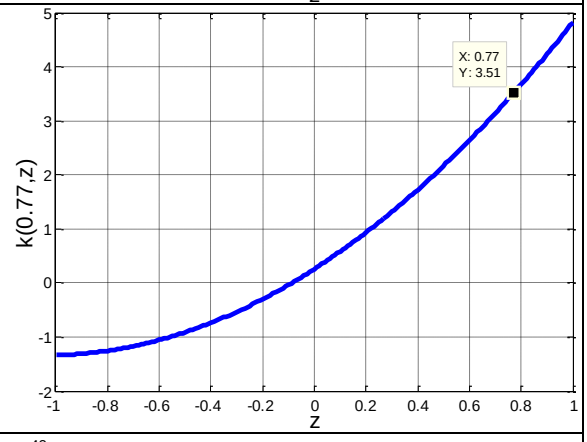
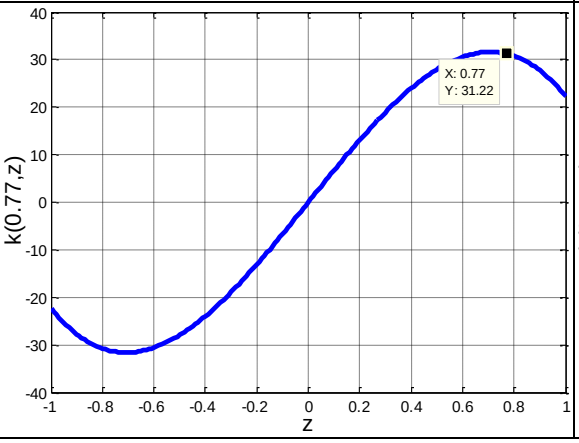
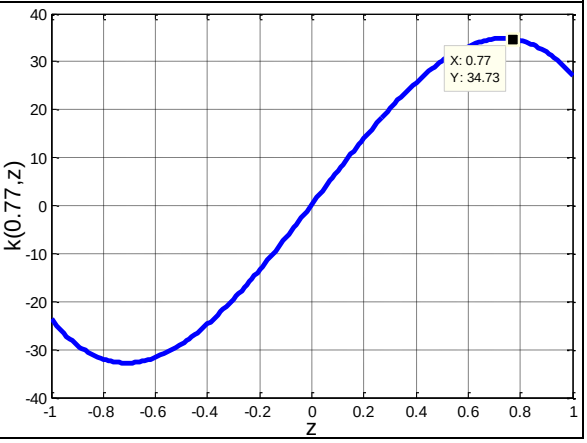
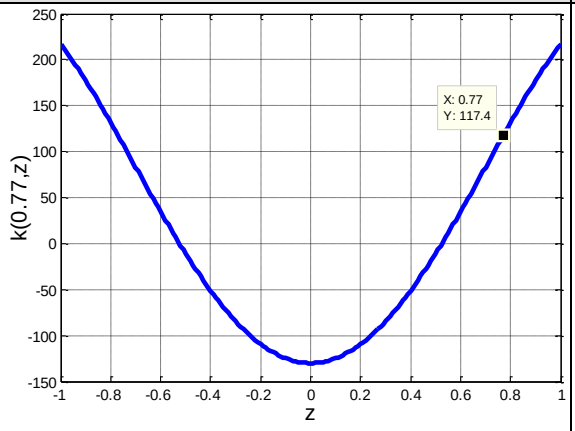
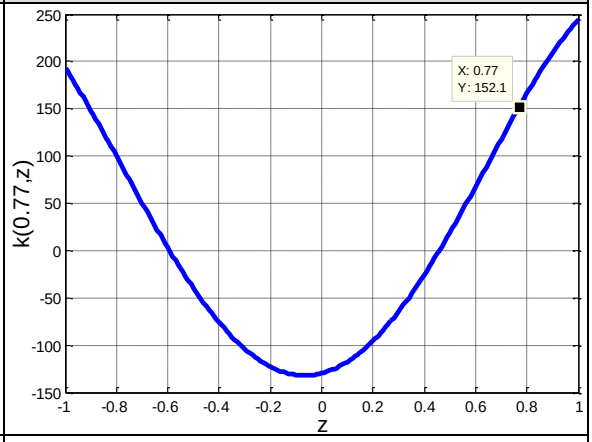
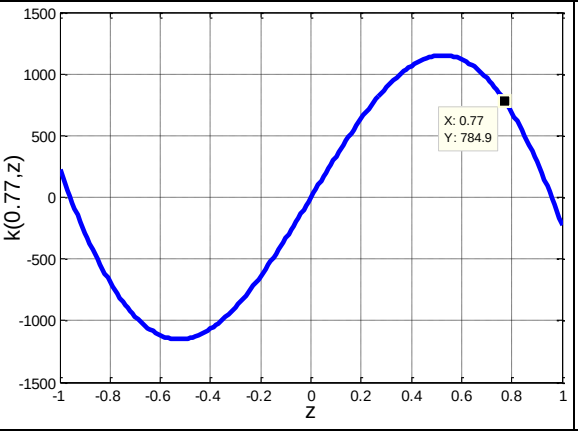
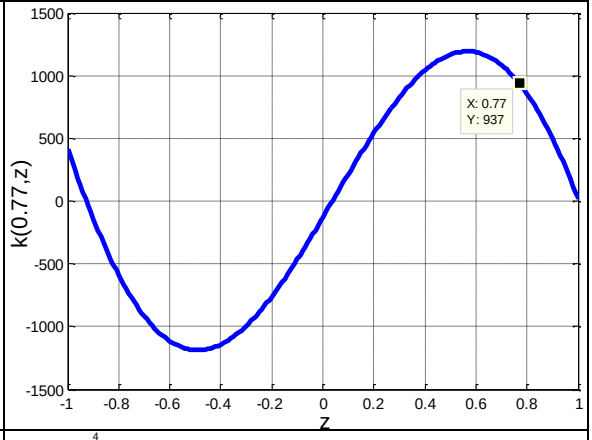
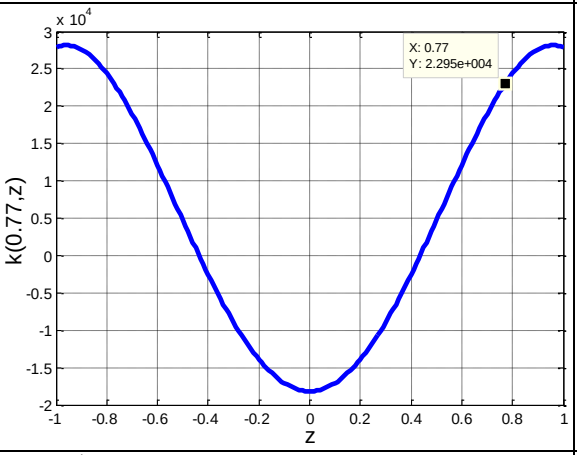
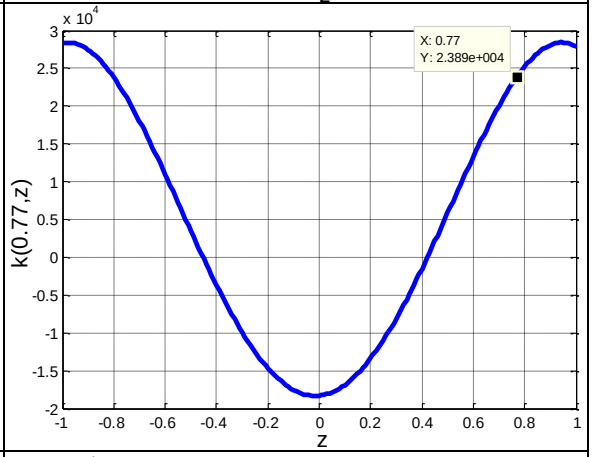
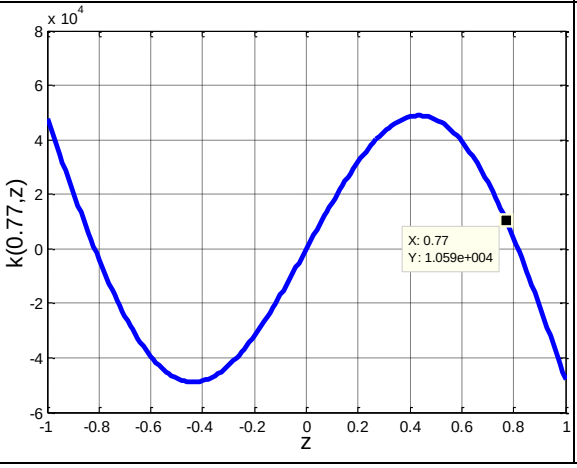
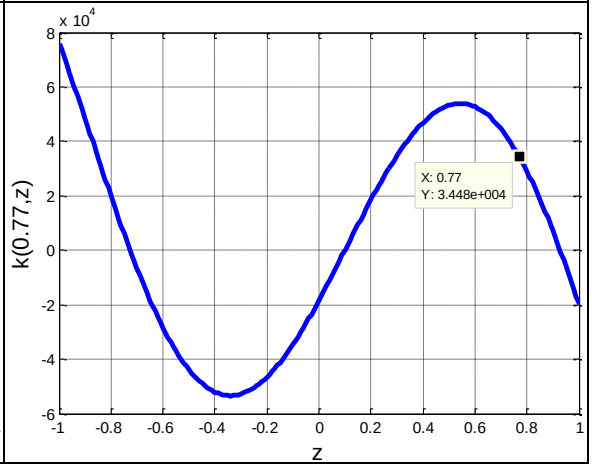
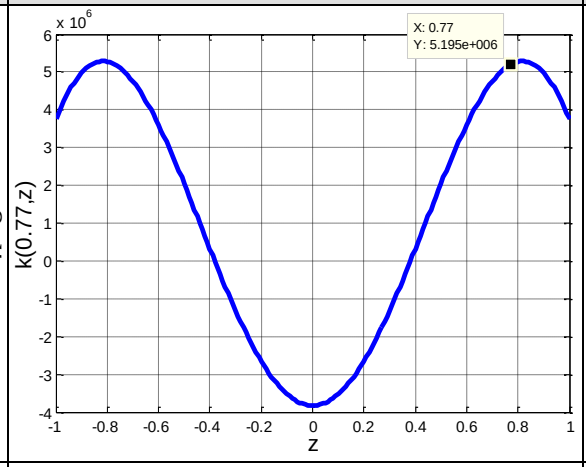
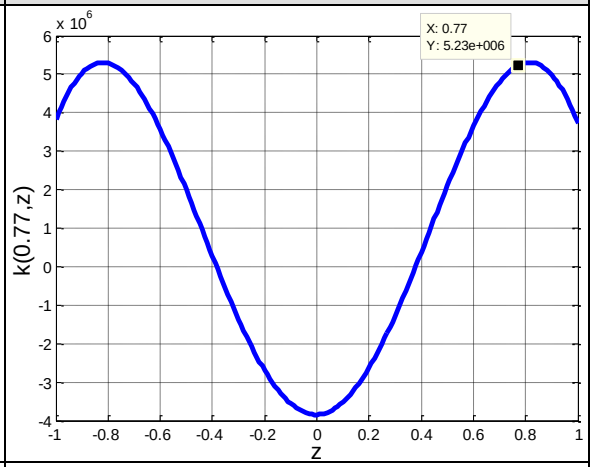
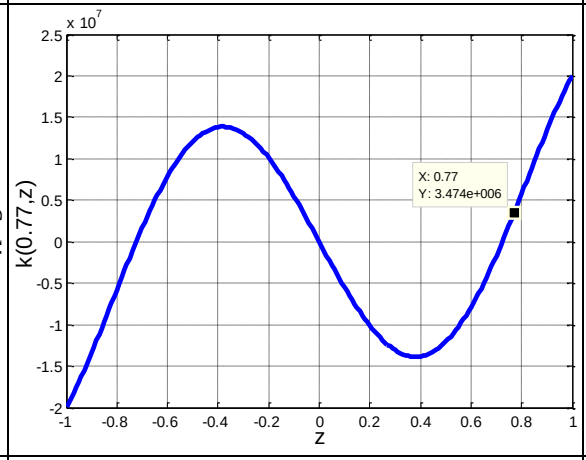
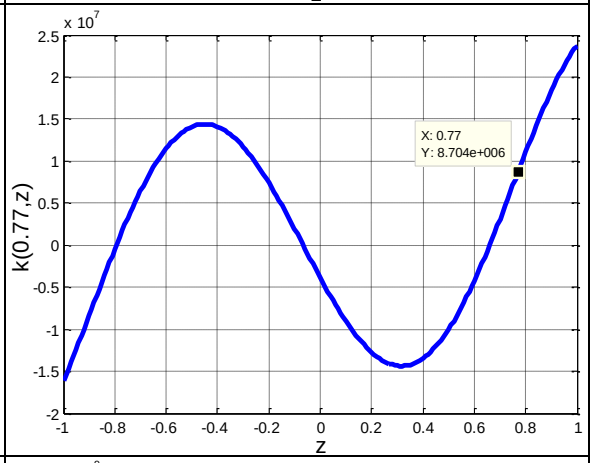
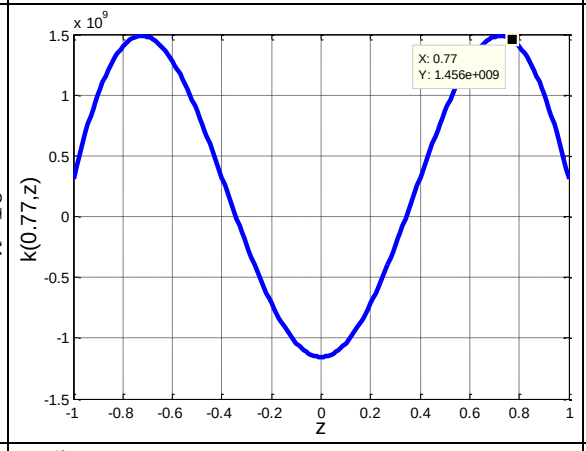
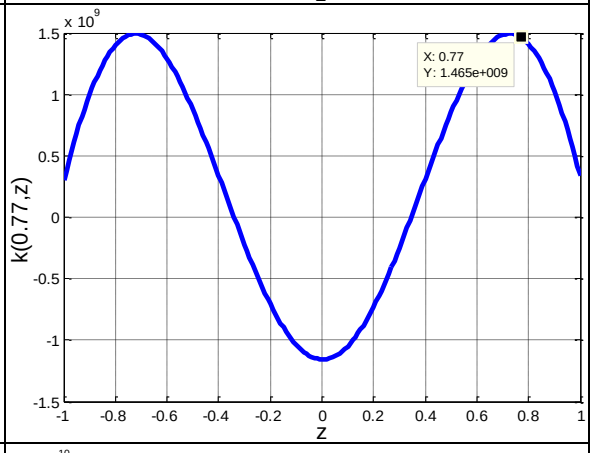
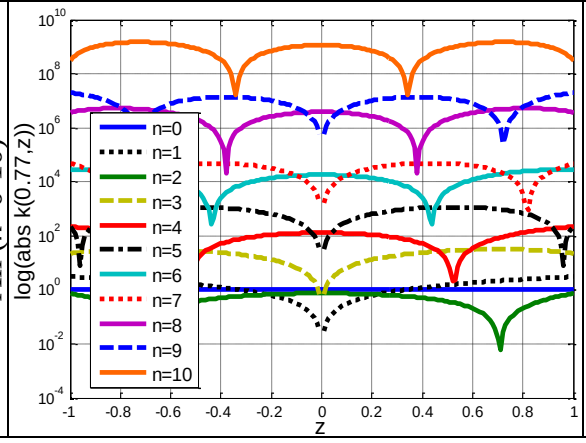
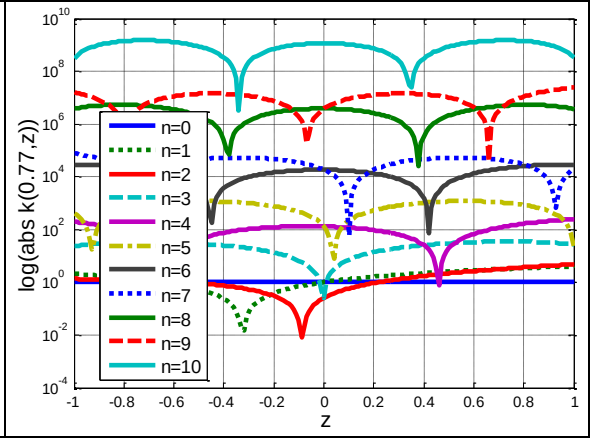
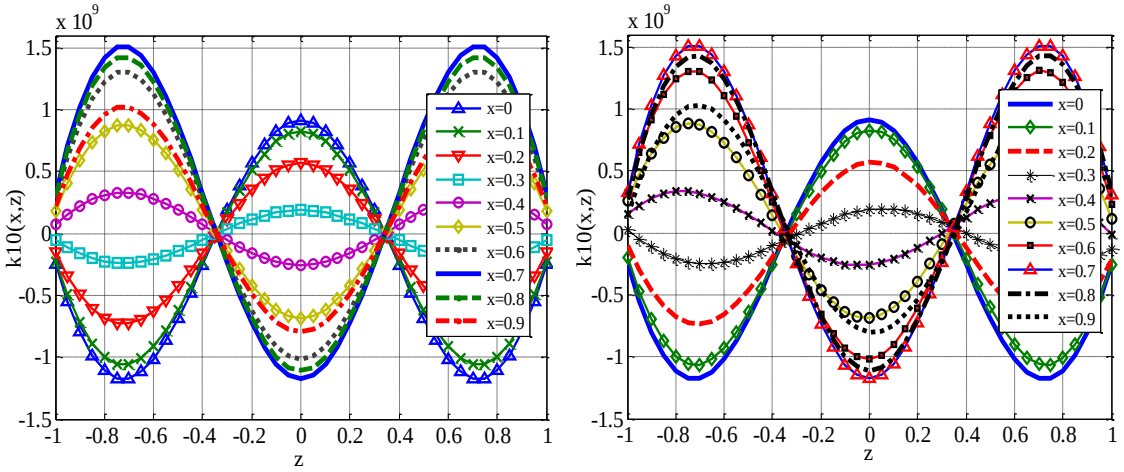

Hermite kernels without sum, at x=0
Hermite kernels with sum, at x=0
n=8
n=9
n=10
All (n=0-10)
k(0,z)
log(abs k(0,z))
z
x 10^6
x 10^9
n=0
n=1
n=2
n=3
n=4
n=5
n=6
n=7
n=8
n=9
n=10

Table 3.4 Comparison of the shape characteristics of the first 10 orders of the Hermite polynomial kernels with and without sum, as per (3.62) and (3.63) respectively, at x=0.77.

	Hermite kernels without sum, at x=0.77	**Hermite kernels with sum, at x=0.77**
n=0		
n=1		
n=2		
n=3		

	Hermite kernels without sum, at x=0.77	Hermite kernels with sum, at x=0.77
n=4		
n=5		
n=6		
n=7		

	Hermite kernels without sum, at x=0.77	Hermite kernels with sum, at x=0.77
n=8		
n=9		
n=10		
All (n=0-10)		



(a) 10th order Hermite kernel without sum at different *x*-values as per (3.65)

(b) 10th order Hermite kernel with sum at different *x*-values as per (3.64)

Figure 3.11 Effect of the variation of the input *x*-value on the shape characteristics of the 10th order Hermite polynomial kernel with and without sum as per (3.64) and (3.65), respectively.

On analysing the shape characteristics of the Hermite kernels plotted in Tables 3.3, 3.4 and Figure 3.11, and comparing them to their Chebyshev and Legendre kernels counterparts from the previous two sub-sections, one can immediately observe the following differences in their similarity-based shape properties. Unlike the Chebyshev and Legendre kernels, the Hermite kernels, on the other hand (either with or without the sum), were found to generally fail to develop a dominant maximum peak at the locations where the two inputs happen to be identical. Although the plots in Table 3.3 (at x=0) show that the Hermite kernels with sum has got a peak at x=0, yet, this peak is not the dominant maximum value within the normalized region of [-1,+1] as one would expect from an ideal measure of similarity.

Even worse, the location of this peak is not actually following the instances, where the two inputs happen to be identical wherever else in the input space (as was the case for example with the Chebyshev and Legendre kernels analysed in the previous sub-sections), as demonstrated by the plots shown in Table 3.4 and Figure 3.11. In other words, the Hermite kernels (either with or without the sum) tend to only change their amplitude and polarity with the variations in the input *x*-value, instead of developing a dominant maximum peak at its location. As such, it is clear that the Chebyshev and Legendre kernels with sum are capable of calculating more accurate similarity measures than those utilizing the Hermite polynomials, and would therefore be expected to yield better classification performance, as will be experimentally investigated later on in Chapters 4 and 6.

On the other hand, although the shape characteristics of the Hermite kernels with the sum (illustrated in Tables 3.3 and 3.4) do not develop a dominant maximum peak at identical inputs, yet, one can still notice that their values at identical inputs are still higher (although

infinitesimal) than their counterparts without the sum. As such, it would also be expected that the Hermite kernels with the sum to yield better classification performance than their counterparts without the sum, as will also be experimentally investigated later on in Chapters 4 and 6.

Finally, a common observation amongst all the three polynomial kernels (i.e., Chebyshev, Legendre, and Hermite) is that as the order increases, the region where the kernels with sum exhibit a monotonic decay behaviour also decreases. This means that more data points are likely to fall outside the monotonic window where the kernel tend to oscillate in a wavy pattern and is therefore more likely to yield inaccurate similarity measures leading to a degraded classification performance; a problem which will also be addressed in Chapter 5.

3.5 Summary

This chapter proposed a solution to the main research problem identified and addressed by the thesis, whereby the utilization of SVM kernels as implicit mapping tools to high-dimensional feature spaces has dominated majorly over their use as similarity functions in the literature. The theoretical foundation of this proposed solution is based on the shape characteristics exhibited by the underpinning kernel (being thought of as a measure of similarity) and how these characteristics should typically look like, to reflect the level of similarity between the kernel's two input patterns, and hence aid the classifier's decision as to whether they belong to the same or different classes.

Given that patterns from the same class share more similar characteristics than those belonging to different classes, the chapter therefore defined a similarity-based pictorial model which proposes that the similarity curve should typically be maximized when the two kernel inputs are identical, and it should decay monotonically as they differ more and more from each other. Motivated by the previously reported plots of some orthogonal polynomial kernels, such as the Chebyshev, Legendre, and Hermite kernels, the chapter then adopted their exhibited pictorial characteristics to underpin the assessment of the proposed similarity-based pictorial model for SVM pattern classification.

The chapter critically analyzed different configurations of these polynomial kernels (e.g., with and without sum; at identical and non-identical inputs, etc.) together with their shape characteristics corresponding to each case. It was observed that these polynomial kernels

exhibit different pictorial characteritics at each of these configurations, which, under certain conditions, can closely match the characteristics of the proposed similarity-based pictorial model, and hence enable the polynomial kernel to calculate more accurate similarity measures, and hence score better classification performance, as will be validated in the experimental investigations in the following subsequent chapters.

To the best knowledge of the author, the similarity-based pictorial model proposed in this chapter provides the first logical and easily understood theoretical foundation as to why some kernels can lead to better classification performance than others. This theoretical foundation provides new valuable and easy means for kernel design for the machine learning community. The analyses of the shape characteristics of the adopted orthogonal polynomial kernels, and how they relate to the proposed similarity-based pictorial model, have also never been done before in the way presented in this chapter.

Chapter 4
Orthogonal Polynomial Kernels in a Similarity Fusion Framework Context

4.1 Introduction

It can be shown from the study of orthogonal polynomial kernels (OPKs) conducted in Sections 3.4.1 and 3.5.2 that they are naturally constructed from a mixture of summative and multiplicative combination of base kernel blocks that synergistically contribute towards calculating better similarity measures, as exhibited by their shape characteristics. Framed by the information fusion theory, this chapter presents a novel similarity fusion framework within which the hierarchical structure of OPKs can be characterised and defined. To the extent that the individual kernel blocks of OPKs can provide complementary information (e.g., similarity measures) about the input data, the resulting performance of the classifier constructed from their fused kernel components is expected to outperform that of the best individual kernel building block.

A number of comprehensive experiments are conducted in this chapter to validate the synergy in both the summative and multiplicative fusion operations inherent in the hierarchical structure of OPKs, together with a critical comparison with the performance gained from other traditional SVM kernels that have been in common use.

4.2 Inherent fusion architecture of orthogonal polynomial kernels

Section 3.4.1 (in Chapter 3) explained the original mathematical construction of SVM kernels from orthogonal polynomials. That is, for scalar inputs, the kernel is constructed by summing up all the polynomial orders from 0 to n, as per (3.4). It has also been explained in Sections 3.4.1.1 and 3.4.1.2 how (3.4) was adopted to be also applicable to vector inputs, via the previously proposed ‘pairwise’ and ‘vectorial’ processing methodologies, introduced by [31] and [17], respectively, using the Chebyshev polynomials. Given that the vectorial approach has already been shown (through the work of [17]) to outperform its pairwise counterpart, all the investigations conducted in this chapter will continue to adopt

the vectorial approach[13] proposed by [17], in which case (3.4) can be re-written for vector inputs in a generic form as:

$$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} P_i(\mathbf{x}) P_i^T(\mathbf{z}) = \sum_{i=0}^{n} \langle P_i(\mathbf{x}), P_i(\mathbf{z}) \rangle, \tag{4.1}$$

where $P(\mathbf{x})$ and $P(\mathbf{z})$ are the evaluations of the employed orthogonal polynomial (be it Chebyshev, Legendre, etc.) on the first and second input vector kernel arguments as a whole, as shown, for example, in Table 3.2 for the Chebyshev polynomials case.

Based on the closure property of kernels, which states that the addition of valid kernels also yields a valid kernel, one can think about the kernel in (4.1) as being the summation of smaller valid kernel building blocks as:

$$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} k_i(\mathbf{x},\mathbf{z}) \;=\; k_0(\mathbf{x},\mathbf{z}) + k_1(\mathbf{x},\mathbf{z}) + \cdots + k_n(\mathbf{x},\mathbf{z}), \tag{4.2}$$

where each of the i^{th} order individual kernel blocks (k_i's) being summed up is a valid kernel (and therefore is a legitimate tool to measure the similarity between its two input vectors) representing the evaluation of each individual polynomial order kernel on the input vectors, and are given by:

$$\begin{aligned} k_0(\mathbf{x},\mathbf{z}) &= \langle P_0(\mathbf{x}), P_0(\mathbf{z}) \rangle \\ k_1(\mathbf{x},\mathbf{z}) &= \langle P_1(\mathbf{x}), P_1(\mathbf{z}) \rangle \\ &\cdots \\ k_n(\mathbf{x},\mathbf{z}) &= \langle P_n(\mathbf{x}), P_n(\mathbf{z}) \rangle \end{aligned} \tag{4.3}$$

On the other hand, as part of their study of the SVM kernels constructed from the Chebyshev polynomials, Ye et at. [31] and Ozer et al. [17] both showed that their corresponding weighting function is also a valid kernel, and therefore, by utilizing the multiplicative closure property, which states that the multiplication of two (or more) valid kernels also yields a valid kernel, the authors in [31] and [17] constructed their proposed overall composite Chebyshev kernels in the form of:

$$k(\mathbf{x},\mathbf{z}) = k_1(\mathbf{x},\mathbf{z}) \times k_2(\mathbf{x},\mathbf{z}) \tag{4.4}$$

where $k_1(\mathbf{x},\mathbf{z})$ is the summative Chebyshev polynomial element of the overall composite kernel, and $k_2(\mathbf{x},\mathbf{z})$ is its corresponding weighting kernel function. There is, however, no

[13] Although later, Chapter 6 will be proposing another new processing approach and will be showing, both theoretically and experimentally, that it outperforms both of the previously proposed vectorial and pairwise approaches.

theoretical guarantee that the best classification performance will be achieved if the corresponding weighting function of the Chebyshev polynomials $k_2(\mathbf{x},\mathbf{z})$ is the kernel that should be used to be combined with its sister polynomial kernel $k_1(\mathbf{x},\mathbf{z})$. Recall that the ultimate aim is to improve the classification performance (rather than retaining the orthogonality of the underpinning polynomial kernels unnecessarily), which might be the reason why the authors in [17] were urged to propose to replace the corresponding weighting function of the Chebyshev polynomials with the commonly used Gaussian kernel instead, to construct what they referred to as the 'Modified Chebyshev kernel'. Their reported experimental results showed that such practice (of combining the summative polynomial kernel $k_1(\mathbf{x},\mathbf{z})$ with some other more robust kernels instead of their corresponding weighting functions) could indeed be useful in achieving even better classification performance, and hence, showed that the choice of $k_2(\mathbf{x},\mathbf{z})$ should not only be restricted to the weighting function of the employed polynomials.

As such, one can generalize this approach, by defining the construction of the composite vectorial polynomial kernels as:

$$k(\mathbf{x},\mathbf{z}) = k_1(\mathbf{x},\mathbf{z}) \times k_2(\mathbf{x},\mathbf{z}) = \left[\sum_{i=0}^{n} \left\langle P_i(\mathbf{x}), P_i(\mathbf{z}) \right\rangle\right] \times w(\mathbf{x},\mathbf{z}), \qquad (4.5)$$

where $P(.)$ is the evaluation of the orthogonal polynomials (be it Chebyshev, Legendre, etc.) on the input vectors, and $w(\mathbf{x},\mathbf{z})$ can be either their corresponding weighting function or any other valid kernel.

So, by analysing closely the composite structure of the kernel in (4.5), and inline with the various kernel fusion and hybridization or combination strategies explained back in Section 2.6, one can therefore realize that such a composite kernel structure is inherently formulated from a mixture of summative and multiplicative combinations of base kernel building blocks that could synergistically produce better similarity measures and, hence, collectively contribute towards their enhanced classification performance. Viewing this approach from a synergistic similarity fusion perspective is what has unfortunately been overlooked in previous literature to appropriately characterise the behaviour and justify the performance of these composite polynomial kernels when implemented within the SVM algorithm. To the extent that each of the individual kernel blocks in (4.5) can provide complementary information (e.g., similarity measures) about the input data, the resulting performance of the classifier constructed from the fused kernels is expected to outperform that of the best individual kernel building block.

4.3 Hierarchical development of the synergistic similarity fusion framework for orthogonal polynomial kernels

To investigate if this perspective of fusion of OPKs holds true, this section will define the hierarchical structure of the composite kernel in (4.5) within a novel three-stage similarity fusion framework. This is achieved by breaking down the composite kernel into its individual base kernel building blocks and analysing their synergistic fusion behaviour in terms of their exhibited shape characteristics introduced earlier in the previous chapter. The extended investigations conducted herein will continue to adopt the kernels constructed from not only the Chebyshev, but also the Legendre and Hermite polynomials; and hence, it is worth to re-present their plots again (side by side) in Figures 4.1 – 4.6 for comparative convenience and ease of analysis.

4.3.1 Similarity fusion framework: First stage

In the first stage, the n^{th} order kernel is constructed using only the n^{th} order of the polynomial without summation and without being combined with any other kernel function, in the form of:

$$k_n(\mathbf{x},\mathbf{z}) = \langle P_n(\mathbf{x}), P_n(\mathbf{z}) \rangle, \tag{4.6}$$

where $P_n(\cdot)$ is the employed generic form of the orthogonal polynomial of order n.

Applying this first stage of the framework to the orthogonal polynomials under investigation would yield:

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle T_n(\mathbf{x}), T_n(\mathbf{z}) \rangle \tag{4.7}$$

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle L_n(\mathbf{x}), L_n(\mathbf{z}) \rangle \tag{4.8}$$

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle H_n(\mathbf{x}), H_n(\mathbf{z}) \rangle \tag{4.9}$$

where $T(\cdot)$, $L(\cdot)$, $H(\cdot)$, denote the first stage (S_1) in the construction of the composite OPKs without the sum using the Chebyshev, Legendre, or Hermite polynomials, respectively. Figure 4.1 and Figure 4.3 illustrate the shape characteristics of the first 10 orders of each of these kernels at x=0 and x=0.77, respectively; whereas Figure 4.5 demonstrates the effect on their 10^{th} order kernels' shape due to the variation of the input x-values in the range of [0, 0.1, …, 0.9].

Although these Stage 1 kernels are valid Mercer kernels, and are therefore legitimate tools to measure similarity, how good their yielded similarity measures are[14], is a different story. As such, the purpose here is to explore the SVM classification performance as a result of implementing such Stage 1 kernels evaluated using only the n^{th} order of the employed polynomial.

4.3.2 Similarity fusion framework: Second Stage

In Stage 2 kernels, all the i^{th} order polynomial kernels from 0 up to n are combined together via a summation fusion operation, as per (4.1). Recall that each of the i^{th} order individual kernel blocks being summed up is a valid Mercer kernel (and therefore is a legitimate tool to measure the similarity), and, their fusion by summation is expected to be synergistic. This means that the fused kernel should be able to yield better measures of similarity between the two input vectors than if each of these i^{th} order individual kernel blocks are used on their own. Such a hypothesis can now be validated by re-analysing the shape characteristics of the Stage 2 (S_2) Chebyshev, Legendre, and Hermite kernels, demonstrated in Figures 4.2, 4.4, and 4.6, and given respectively by:

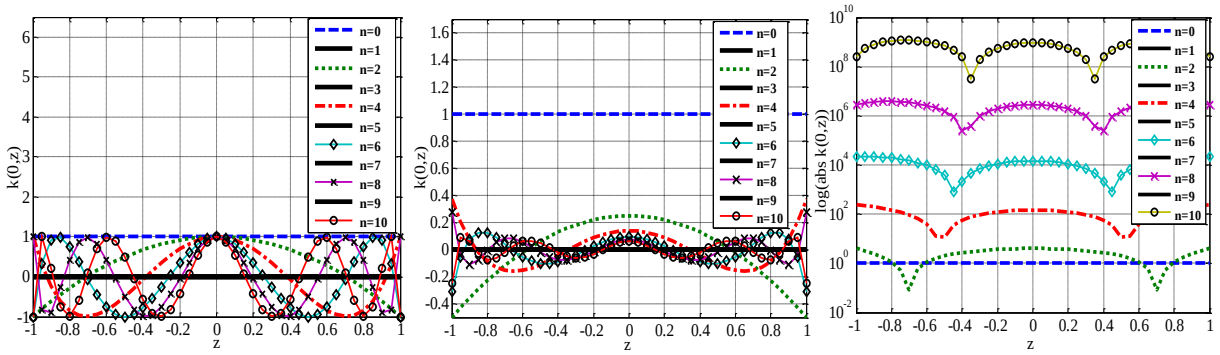
$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle, \tag{4.10}$$

$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle L_i(\mathbf{x}), L_i(\mathbf{z}) \rangle, \text{ and} \tag{4.11}$$

$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle H_i(\mathbf{x}), H_i(\mathbf{z}) \rangle. \tag{4.12}$$

On analysing the shape characteristics of Stage 2 kernels illustrated in Figure 4.2 at x=0, one can observe that as the order increases, the peak kernel value of both the Chebyshev and Legendre kernels increases. This indicates that the higher number of i^{th} order kernels fused together by summation, the more the kernel shape gets closer to the characteristics of the ideal similarity function illustrated back in Figure 3.1, showing that the addition of these i^{th} order kernels do actually help each other to achieve a better kernel (i.e., a better measure of similarity). This shows that the fusion by summation operation is indeed synergistic because it makes every n^{th} order Stage 2 kernel ($k_n^{S_2}$) yield better similarity measures compared to its corresponding n^{th} order Stage 1 counterparts ($k_n^{S_1}$) which do not exhibit any dominant maximum peaks, as illustrated by their plots in Figure 4.1.

[14] Compared, for example, to their Stage 2 counterparts.

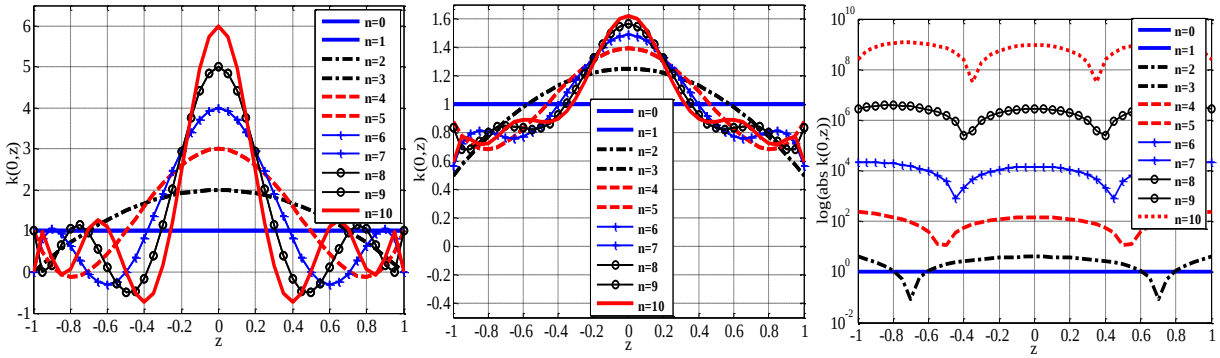


(a) Stage 1 Chebyshev kernels for x=0, as per (4.7)

(b) Stage 1 Legendre kernels for x=0, as per (4.8)

(c) Stage 1 Hermite kernels for x=0, as per (4.9)

Figure 4.1 First 10 orders of Stage 1 Chebyshev, Legendre, and Hermite kernels for x =0.

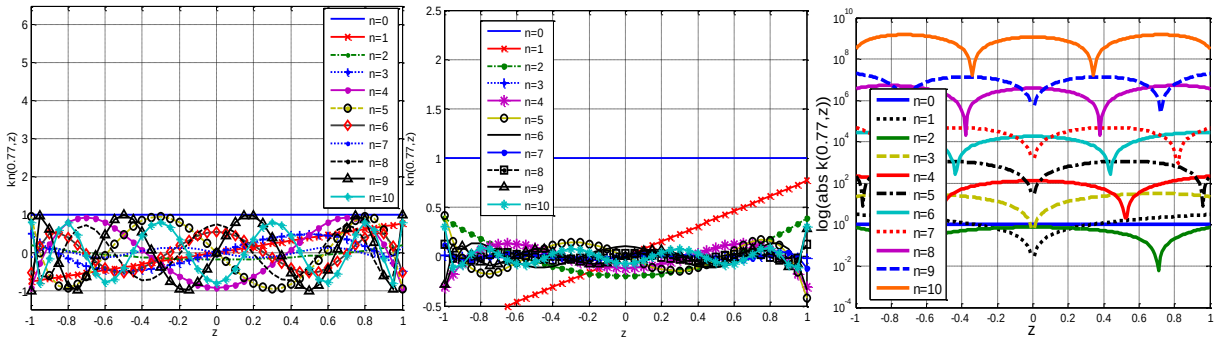


(a) Stage 2 Chebyshev kernels for x=0, as per (4.10)

(b) Stage 2 Legendre kernels for x=0, as per (4.11)

(c) Stage 2 Hermite kernels for x=0, as per (4.12)

Figure 4.2 First 10 orders of Stage 2 Chebyshev, Legendre, and Hermite kernels for x =0.

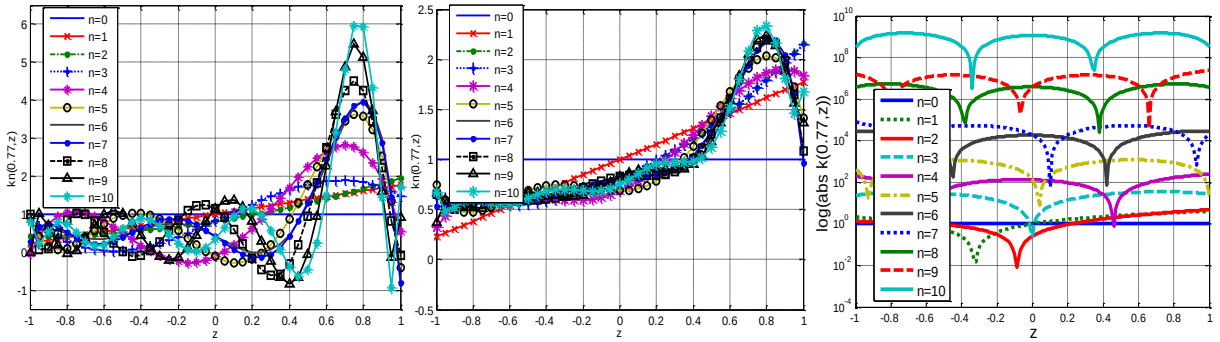


(a) Stage 1 Chebyshev kernels for x=0.77, as per (4.7)

(b) Stage 1 Legendre kernels for x=0.77, as per (4.8)

(c) Stage 1 Hermite kernels for x=0.77, as per (4.9)

Figure 4.3 First 10 orders of Stage 1 Chebyshev, Legendre, and Hermite kernels for x =0.77.

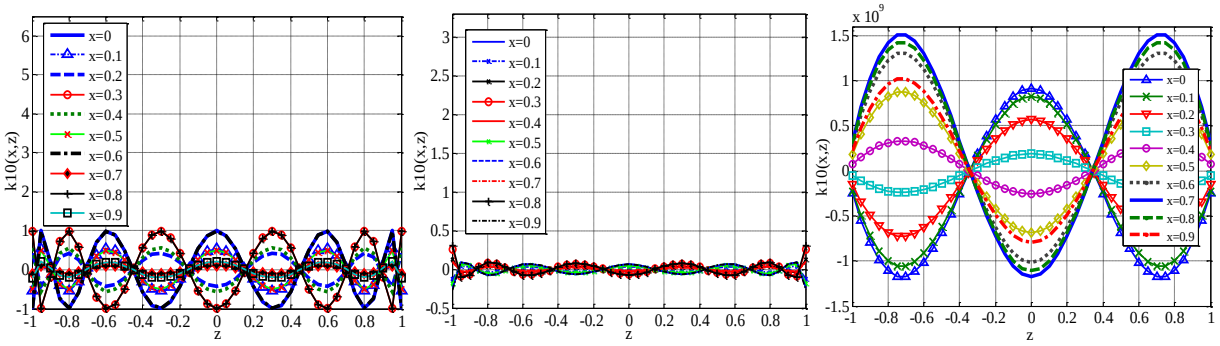


(a) Stage 2 Chebyshev kernels for x=0.77, as per (4.10)

(b) Stage 2 Legendre kernels for x=0.77, as per (4.11)

(c) Stage 2 Hermite kernels for x=0.77, as per (4.12)

Figure 4.4 First 10 orders of Stage 2 Chebyshev, Legendre, and Hermite kernels for x =0.77.

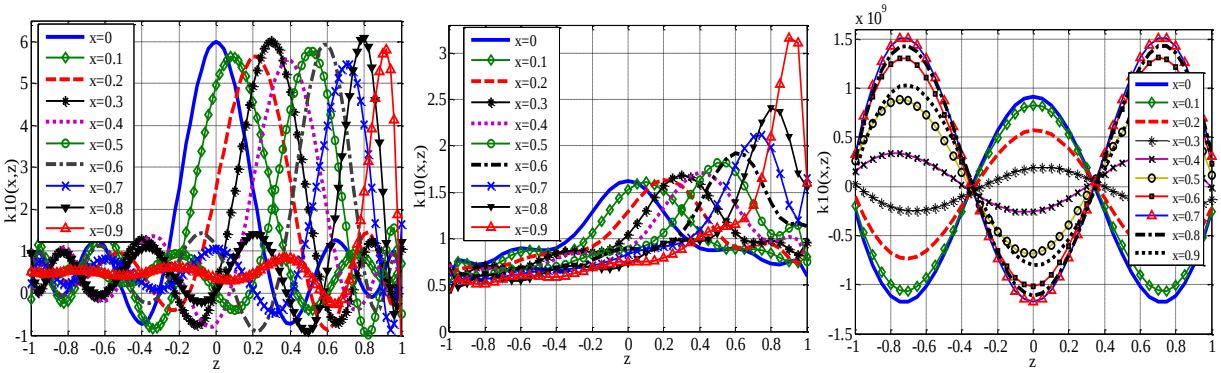


(a) Stage 1 Chebyshev kernels of polynomial order n=10, as per (4.7)

(b) Stage 1 Legendre kernel of polynomial order n=10, as per (4.7)

(c) Stage 1 Hermite kernels of polynomial order n=10, as per (4.9)

Figure 4.5 Effect of the variations in the x-value on Stage 1 Chebyshev, Legendre, and Hermite kernels of polynomial order n = 10.



(a) Stage 2 Chebyshev kernels of polynomial order n=10, as per (4.10)

(b) Stage 2 Legendre kernels of polynomial order n=10, as per (4.11)

(c) Stage 2 Hermite kernels of polynomial order n=10, as per (4.12)

Figure 4.6 Effect of the variations in the x-value on Stage 2 Chebyshev, Legendre, and Hermite kernels of polynomial order n = 10.

To exacerbate this situation, one can even notice that if the two input vectors happen to be identical at the origin (i.e., at x=0), the odd order Stage 1 kernels actually fail to calculate any measures of similarity at all (i.e., the kernel value is equal to zero) throughout the normalized data region of [-1,+1], as shown in Figure 4.1, even when their two inputs are identical. Stage 2 kernels, on the other hand, do not suffer from this catastrophic situation as they sum up all the Stage 1 kernels from polynomial order 0 to n, and as such, at x=0, their odd orders end up being the same as their preceding even orders, as demonstrated in Figure 4.2. One can also observe that although the Stage 2 kernels tend to oscillate in a wavy pattern after a certain threshold, yet, unlike their Stage 1 counterparts, the peak-to-peak amplitude of these oscillations tend to decay as the two inputs depart further away from each other, as shown in Figures 4.1 – 4.6; which is another reason why the similarity measures calculated by Stage 2 kernels should be more accurate than those calculated by their Stage 1 counterparts, and should therefore yield better classification performance.

As for the kernels constructed from the Hermite polynomial, although they do not tend to get closer to the characteristics of the ideal similarity function as the order increases (as

their Chebyshev and Legendre counterparts do), yet, the value of their Stage 2 kernels at identical inputs have still shown to be higher than the value of their Stage 1 counterparts. Although this infinitesimal difference might not be very obvious in their logarithmic plots in Figure 4.2 (c) and Figure 4.1 (c), for their Stage 2 and Stage 1 formulations, respectively, yet, as discussed before in the previous chapter, one can observe this difference from their original scale plots demonstrated in Table 3.4. However, similar to their Chebyshev and Legendre counterparts, one can also observe that the odd orders of Stage 1 Hermite kernels shown in Figure 4.1 (c) do also fail to calculate any measures of similarity at all when x=0, rendering their Stage 2 counterparts a lot better similarity measures, as in this case their odd orders also end up being the same as their preceding even orders, as shown in Figure 4.2 (c).

Based on these theoretical analyses of the similarity-based fusion and shape characteristics of the kernels constructed from the Chebyshev, Legendre, and Hermite polynomials that are under investigation, one can conclude that the Stage 2 kernels tend to exhibit better similarity measure characteristics (due to the fusion by summation operation that inherently takes place within their mathematical construction) than their Stage 1 counterparts; and as such, they should be able to produce better classification performance when implemented within the SVM algorithm; a hypothesis which will be experimentally validated later in this chapter.

4.3.3 Similarity fusion framework: Third stage

Although the Stage 2 kernels demonstrated superior similarity measure characteristics over their Stage 1 counterparts, as explained in the previous sub-section, they might still not be good enough to compete with the existing traditional SVM kernels to produce the sought-after classification performance. Therefore, in this final third stage of the similarity fusion framework, the Stage 2 kernels are fused by a multiplication operation with either their corresponding weighting function or a more robust kernel, to produce a composite-Stage 3 kernel in the form of equation (4.5), as explained back in Section 4.2. To investigate the effect on the classification performance as a result of this fusion by multiplication operation, this section will focus on the study of the following vectorial-composite kernels:

- Stage 3 generalized Chebyshev kernel [17]:

$$k_n^{S_3}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle \times \frac{1}{\sqrt{m - < \mathbf{x}, \mathbf{z} >}}, \tag{4.13}$$

- Stage 3 Modified Chebyshev kernel [17]:

$$k_n^{S_3}(\mathbf{x},\mathbf{z}) = \sum_{j=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right), \quad (4.14)$$

- Stage 3 Modified Legendre kernel:

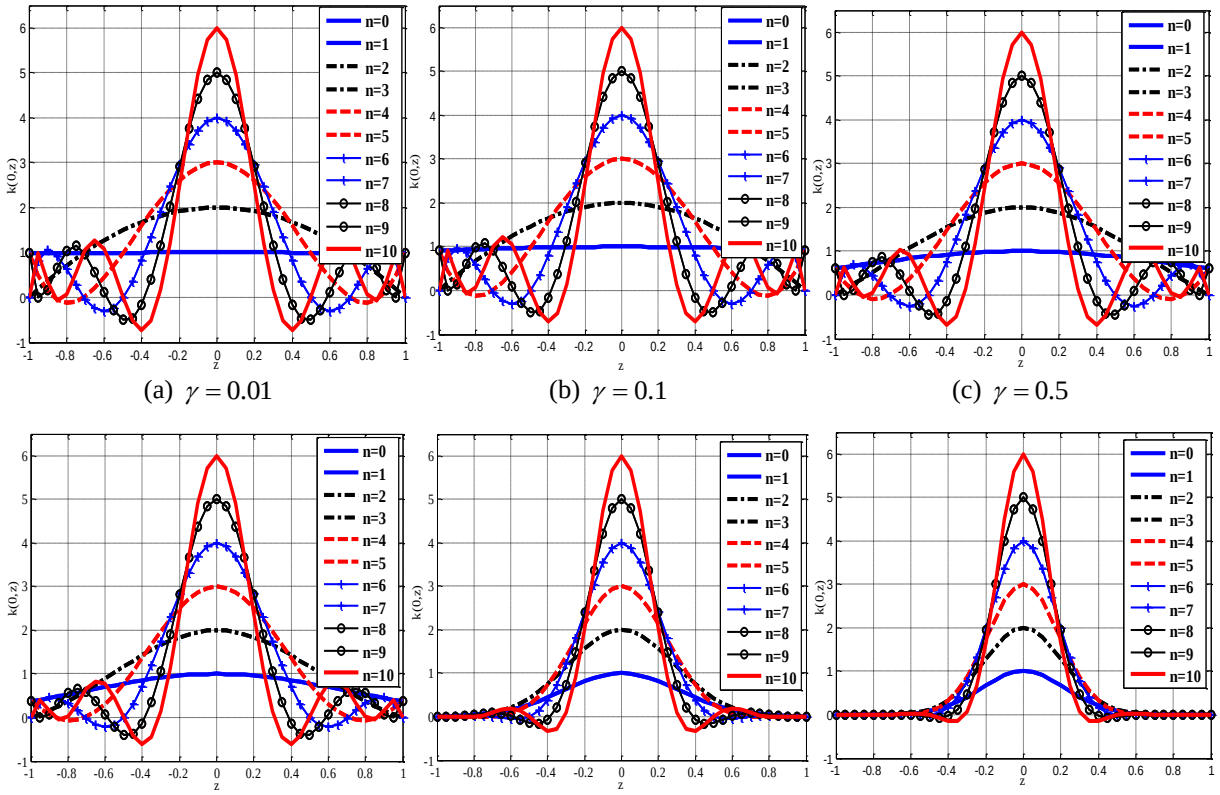
$$k_n^{S_3}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle L_i(\mathbf{x}), L_i(\mathbf{z}) \rangle \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right), \quad (4.15)$$

- Stage 3 Composite Hermite kernel:

$$k_n^{S_3}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle H_i(\mathbf{x}), H_i(\mathbf{z}) \rangle \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right), \quad (4.16)$$

where m is the number of dimensions (i.e., features) of the dataset, as defined by [17] and γ is the Gaussian kernel parameter. Note that the corresponding weighting function of the Legendre polynomials is $w(x)=1$, and hence its modified version proposed here in (4.15) will give us the chance to explore their resulting classification performance when they are combined with another kernel, such as the Gaussian.

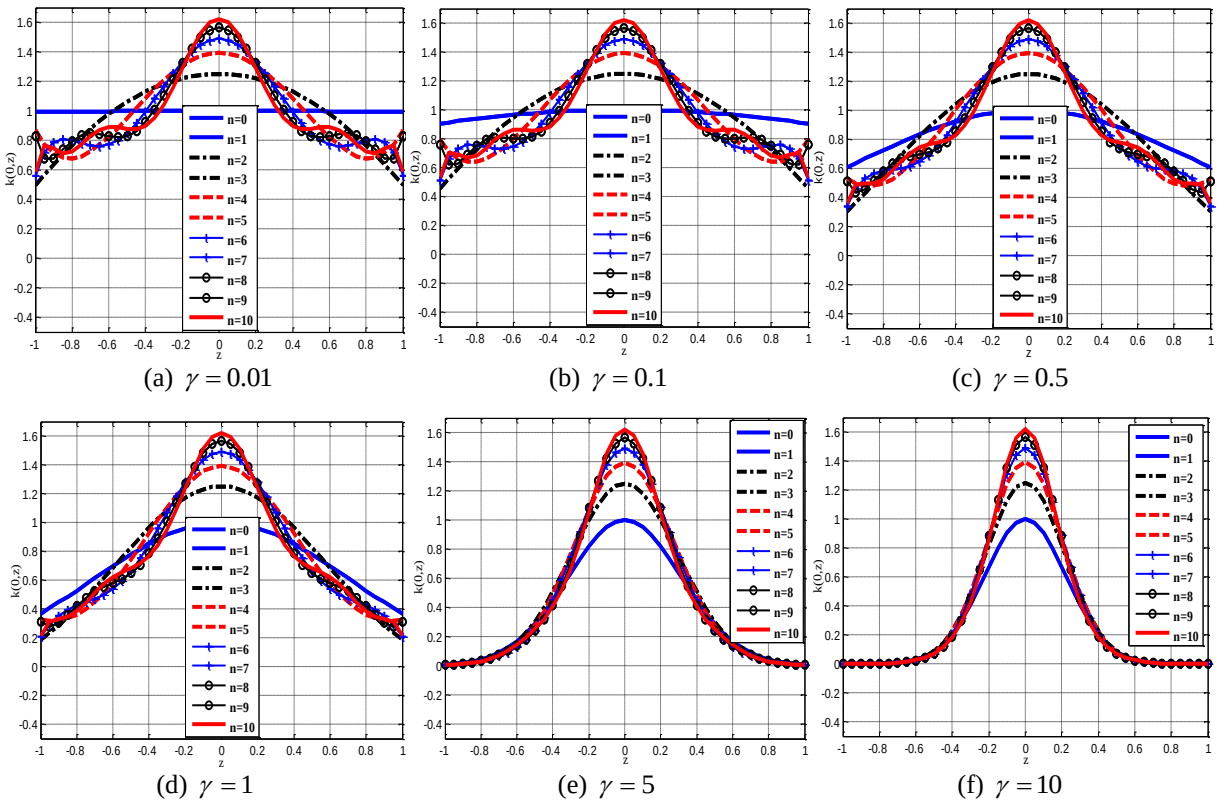
On the other hand, fusing (by multiplication) Stage 2 kernels with the Gaussian kernel will also provide us with an additional parameter γ which will give us more control over the behaviour of the polynomial kernels to tailor their shape to the ideal sought after characteristics, and hence enable the calculation of better similarity measures. Figures 4.7 – 4.9 demonstrate how this process can be achieved for the Stage 3 Chebyshev, Legendre, and Hermite kernels, respectively, when different values of γ are used. One can observe that by appropriately selecting suitable values for γ, the peak-to-peak oscillations can be greatly reduced, hence enabling more accurate similarity measures to be calculated by the underpinning kernels. It is therefore envisaged that this multiplication fusion operation, implemented in the composite-Stage 3 kernels in (4.13 - 4.16), will also be synergistic causing them to not only score better classification accuracies than their Stage 2 counterparts, but also compete with other traditional SVM kernels in common use. Again, this will be validated experimentally in the following sections.



(a) $\gamma = 0.01$ (b) $\gamma = 0.1$ (c) $\gamma = 0.5$

(d) $\gamma = 1$ (e) $\gamma = 5$ (f) $\gamma = 10$

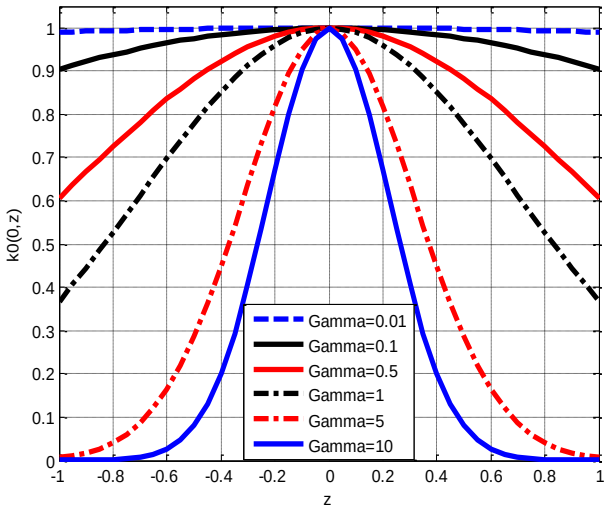
Figure 4.7. Shape characteristics of the Stage 3 Chebyshev kernels with different γ values, at $x = 0$.



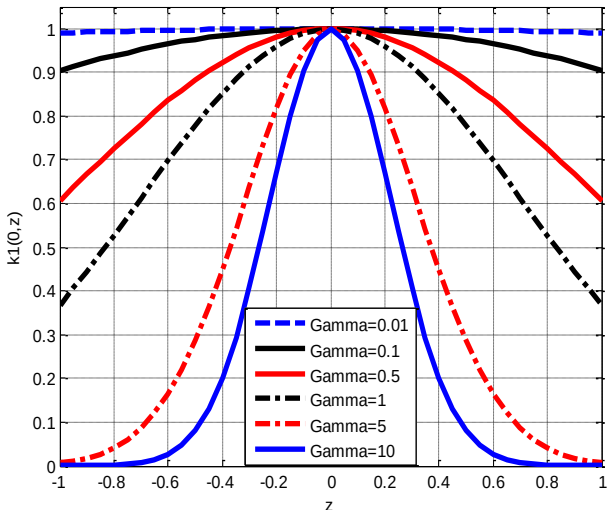
(a) $\gamma = 0.01$ (b) $\gamma = 0.1$ (c) $\gamma = 0.5$

(d) $\gamma = 1$ (e) $\gamma = 5$ (f) $\gamma = 10$

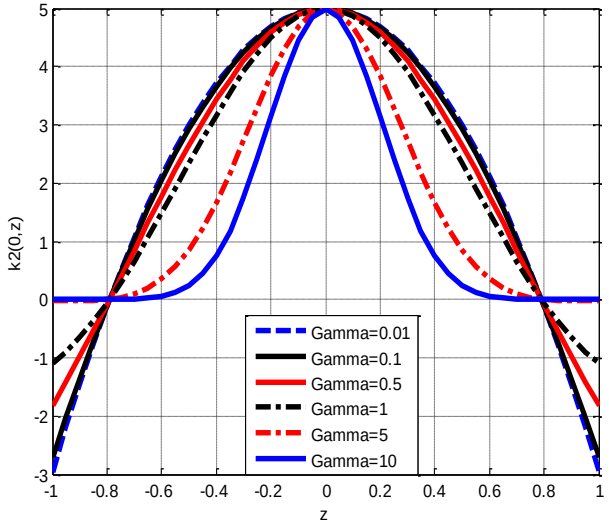
Figure 4.8. Shape characteristics of the Stage 3 Legendre kernels with different γ values, at $x = 0$.



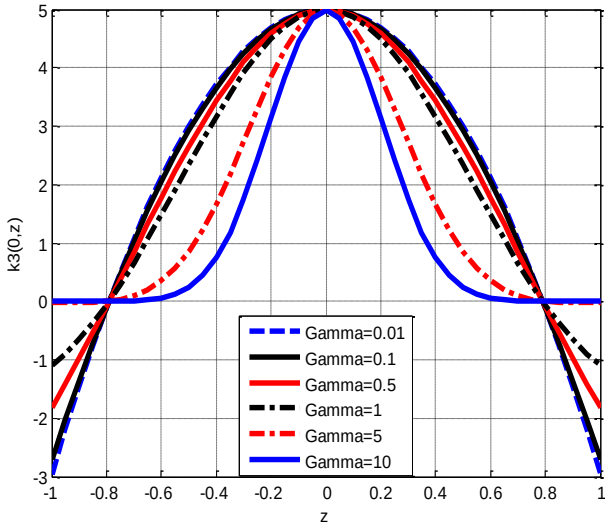
(a) $n = 0$



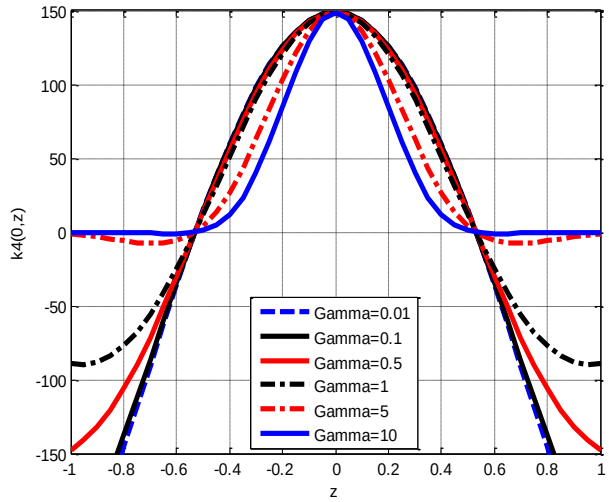
(b) $n = 1$



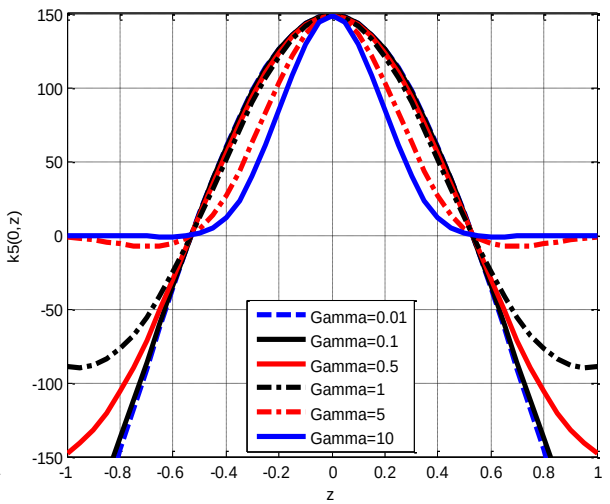
(c) $n = 2$



(d) $n = 3$



(e) $n = 4$



(f) $n = 5$

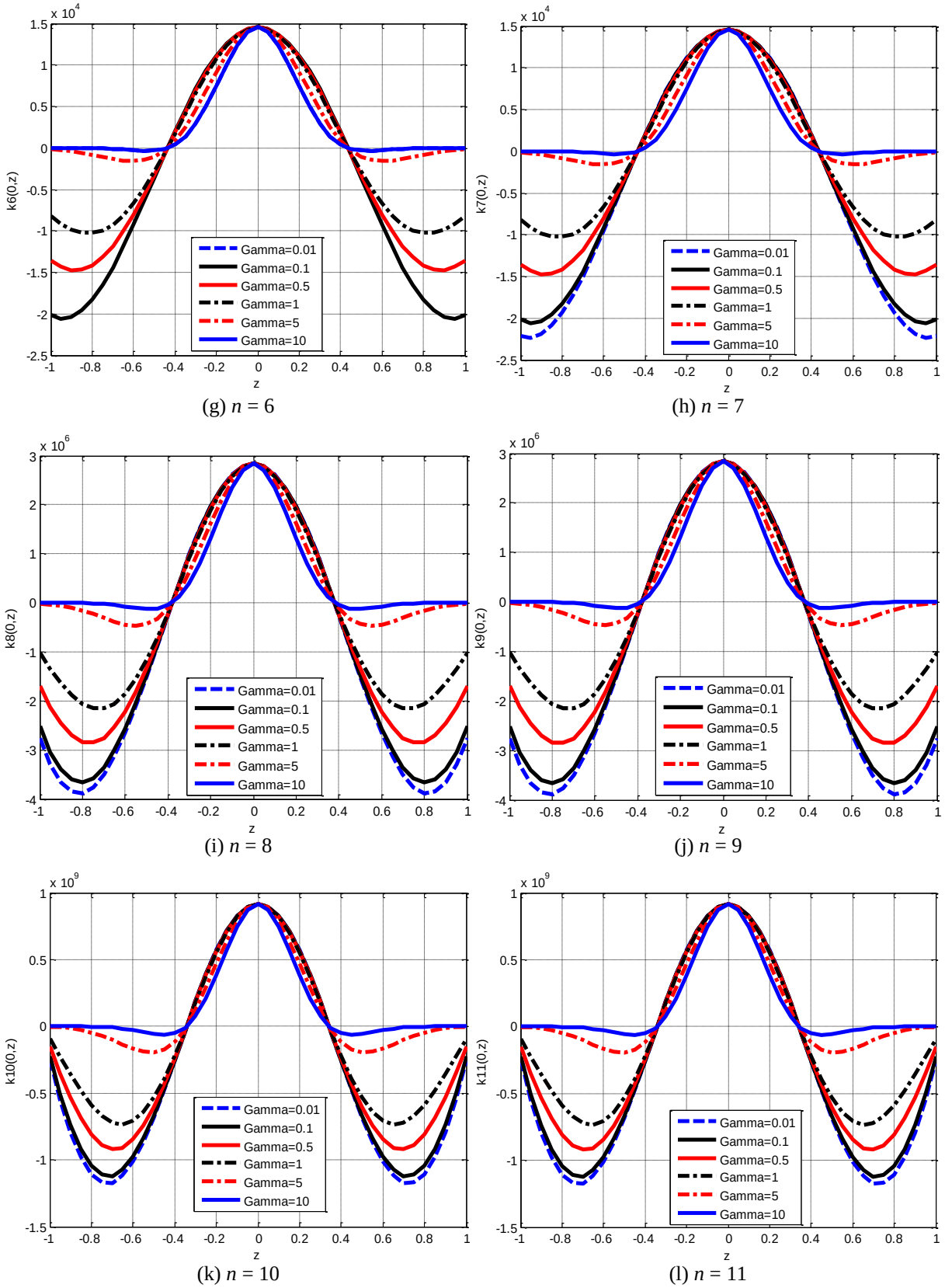


Figure 4.9. Shape characteristics of the Stage 3 Hermite kernels with different γ values, at $x = 0$.

4.4 Experimental validation of the similarity fusion framework for orthogonal polynomial kernels

This section presents a series of comprehensive experiments to test and validate the synergy of the summative and multiplicative fusion operations inherent in the hierarchical construction of the polynomial kernels defined within the 3-stage similarity fusion framework discussed in the previous section. Section 4.4.4 shows the effect of the summative fusion operation by comparing the SVM classification performance scored by the Stage 2 versus Stage 1 kernels. Similarly, Section 4.4.5 also shows the effect of the multiplicative fusion operation by comparing the SVM classification performance of the composite Stage 3 kernels versus their Stage 2 counterparts.

The following sub-sections will first explain the experimental setups, the classification performance assessment metric utilized, and the datasets employed in the experiments.

4.4.1 Experimental setup and model parameter selection

All the experiments were conducted using the *C*-SVM algorithm [12, 114] and the SVM toolbox in [115]. Using the standard grid search [116, 117], the penalization parameter *C* that yielded the best performance was found to be within the range $[10^{-3}, 10^{3}]$; similarly, the range of the Gaussian kernel parameter γ which yielded the best performance was [0.001, 100]. The experiments conducted on multi-class datasets adopted the well-known 'one-versus-all' multi-class decomposition technique [85, 118] to train the SVM on each class-group separately.

To be consistent with previous work, such as [17], the classification performance assessment metric used in all experiments is the classification accuracy. This is defined as the percentage of unseen examples that the classifier was able to 'correctly' classify during the testing phase. Although this performance metric does not tell us what has exactly happened with the 'un-correctly' classified examples (i.e., whether they have been classified to the opposite class, or were not classifiable at all – as explained back in Section 2.2.4), it is important to use it for a fair comparison of the achieved experimental results with previous work.

4.4.2 Dataset selection and methodology for estimating classification accuracy

In this thesis, careful deliberation has been made during the selection of different types of datasets used in the experimental work to assess and validate the similarity fusion framework presented in this chapter, as well as the approaches proposed in subsequent chapters. Several aspects have been taken into consideration, such as:

1- The problem of non-linearly separable class boundaries, as this is the main issue that the SVM kernel-based classifier is tackling by mapping the input space to a higher-dimensional feature space, where linear separability between overlapped classes can be improved.

2- A wide variety of dataset attributes are taken into account to reflect, as much as possible, the characteristics of real-world practical applications; such as: the dimensionality of the dataset (i.e., small, medium, and large number of features), binary and multi-class datasets, and most importantly is the the distribution of the data points throughout the normalized vector space and hence the severity of the non-linearity profile between the class labels.

3- The use of standard datasets available to the machine learning and pattern classification research community. This creates a unified environment for the fair comparison of the performance of different classification algorithms, as the data points to be used for training and testing are fixed and explicitly defined.

4- Consistency with previous published literature, which again enables a fair comparison with similar previous research work which used the same datasets to find out and assess how the performance of the approaches proposed in this thesis compares.

Accordingly, six benchmark datasets (summarised in Table 4.1) from the UCI repository have been used in the experiments conducted in this section [119]. Below is a brief summary of how these datasets have been utilized to estimate the classification accuracy during the testing phase of each of the conducted experiments.

Table 4.1 Benchmark datasets used in the experiments.

	BREAST CANCER	IONOSPHERE	TWO SPIRALS	IMAGE SEGMENTATION	IRIS	THYROID
NO. OF CLASSES	2	2	2	7	3	3
NO. OF EXAMPLES	569	351	194	2310	150	7200
NO. OF FEATURES	30	34	2	18	4	21

4.4.2.1 Breast Cancer Wisconsin dataset

This is a binary dataset that consists of 569 examples where each data vector has 30 features [119]. The features describe the characteristics of breast cells nuclei present in an image, and the aim of the classification algorithm under test is to be able to classify each data vector as belonging to one of two classes: malignant and benign. There are 357 and 212 benign and malignant data vectors, respectively. To be consistent with previous literature, such as [17], the first 50 data points of each class were used for training while the remaining 469 data points were used for testing.

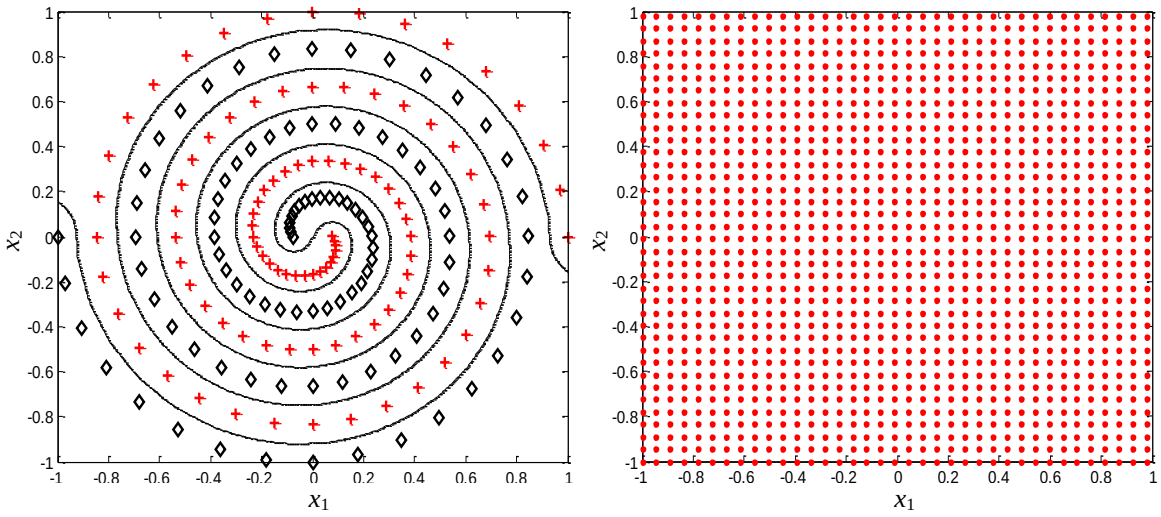
4.4.2.2 Ionosphere dataset

This is another binary dataset originating from some radar data collected by a phased array system consisting of 16 high frequency antennas in Goose Bay (located in Labrador, Canada) [119]. The system targets free electrons in the ionosphere layer, and 'Good' radar return signals means that there is some type of atmospheric structure in the ionosphere, whereas 'Bad' returns are those signals which pass through the ionosphere. The dataset consists of 225 'Good' examples and 126 'Bad' examples, respectively. The Goose Bay receiver system produced 17 pulse numbers, with two attributes each, resulting in a total of 34 features. Inline with previous work, such as [120] and [121], we used the first 200 training examples to classify the remaining 151 test examples.

4.4.2.3 Two Spirals dataset

This is a toy dataset that represents a highly non-linear problem that has challenged the pattern recognition research community since the 1990s [122-125]. This is because it incorporates important characteristics often found in real-time applications and several natural and physical domains [124]. The version of the Two Spirals dataset used in this thesis is the one available in [126] that consists of a balanced 194 two-dimensioanal binary

examples of two intertwined spirals, as shown in its normalized form in Figure 4.10 (a). The objective of the learning algorithm is to discriminate between the data points that are distributed on these two distinct strands (each representing a class) in the x-y plane. Hence, for a new data point, the classifier should be able to map it as belonging to either Class +1 or Class -1.



(a) Optimum decision boundary of the Two Spirals normalized dataset

(b) A pictorial example of data points on a uniformly distributed test grid created for the Two Spirals dataset

Figure 4.10 Visualization of the Two Spirals dataset used in the experiments.

The experiments presented in Sections 4.4.4 and 4.4.5 adopted the 10-fold cross validation technique to estimate the SVM classification accuracy corresponding to each kernel produced from the different stages of the similarity fusion framework. The experiments conducted in Section 4.4.6, however, took advantage of the 2D representation of the Two Spirals dataset to visualize the classification performance of the Stage 3 composite kernels and compare it to some of the traditional SVM kernels in common use. To achieve this purpose, all the 194 examples in this dataset were used for training. If the classifier is successful in appropriately learning the challenging non-linear function of this dataset, it should ideally be able to compute the decision regions and the decision boundaries between the two classes of this dataset to be as close as possible to the one shown in Figure 4.10 (a). For testing, however, a new uniformly distributed grid of test data points is created within the normalized interval of [-1,+1], similar to the one that is pictorially illustrated in Figure 4.10 (b).

Upon testing each data point in this test grid, the classification algorithm should classify it as either Class +1 or Class -1, based on its pre-calculations of the decision boundaries and regions during the training step. This can be visually observed by plotting each of the tested points for the two classes with a different shape/colour within their corresponding

calculated contours or decision regions, as shown in Section 4.4.6 for the results of the performance of the different kernels under investigation. For excellent comparative visual assessment, the plotted results also display the original training data points on top of the predicted data points. Note that, to avoid clutter, only the decision boundaries were plotted and the margins were ignored, as they are of minor importance in the context herein. Following this procedure, only the best results achieved for each kernel are reported in Section 4.4.6.

4.4.2.4 Image Segmentation dataset

This is a balanced multi-class dataset containing a total of 2310 data vectors each representing a randomly drawn image of 7 different outdoor environments (330 images for each class) [119]. These are: brickface, sky, foliage, cement, window, path, and grass. The task is to classify the centre pixel of a 3x3 patch of each image as belonging to one of these 7 categories, based on 18 image processing features of the patch with different minimum and maximum values. Note that the third feature was not used in the experiments as it is the same for all the classes and therefore does not contain any discriminative information. Following [17], we have also divided this dataset such that the first 30 data points of each class were used for training, and the remaining 2100 data points were used for testing, in all the experiments.

4.4.2.5 Iris dataset

This is one of the well known standard datasets in the machine learning and pattern recognition literature [119]. It consists of a balanced 3-class set of 150 multivariate data examples where each vector has 4 features. Each class refers to a species of a flowering plant called “Iris”. These are: Setosa, Versicolour, and Virginica. The features that characterize each species are the length and the width of the sepals and petals in centimetres. Based on the combination of these four features, the role of the classifier is to distinguish these species from each other. In all the experiments, the training dataset is formulated from the first 15 data examples from each class, and the remaining 105 data examples were used to form the testing dataset.

4.4.2.6 Thyroid dataset

This is another multi-class (3-class) dataset available from the UCI repository [119]. It consists of a total of 7200 examples, 21-features each, and the problem is to decide whether a patient (i.e., a data example) is diagnosed to be 'normal (not hypothyroid)', 'hyperfunction', or 'subnormal' functioning. The dataset is configured to use the first 3772 examples for training and the remaining 3428 examples for testing. This sets the train/test ratio amongst the 3-classes to be respectively equal to 93/73, 191/177, and 3488/3178.

4.4.3 Normalization of the range of each vector component

To be consistent with previous literature, such as [17, 31], feature values in all the datasets were normalized (where necessary) to be in the range [-1,+1]. Following the recommendation by Ozer et al. [17], each feature was normalized by considering its own maximum and minimum values throughout the entire dataset. As such, for a vector in the form of $\mathbf{x} = \{x_1, x_2, \cdots, x_m\}$, each j^{th} element of $\mathbf{x}$ was normalized with respect to the maximum and minimum values for that j^{th} element in the whole dataset. The normalized value for each element was then calculated as:

$$x_j^{new} = \frac{2\left(x_j - \text{Min}_j\right)}{\text{Max}_j - \text{Min}_j} - 1 \tag{4.17}$$

where Min_j and Max_j are the minimum and maximum values of the j^{th} feature amongst all the vectors in the dataset, respectively.

4.4.4 Experimental results and discussions for Stage 1 and Stage 2 kernels of the similarity fusion framework

It has been demonstrated in Section 4.3.2, via the analysis of the shape characteristics of Stage 1 and Stage 2 kernels, how the fusion by summation of the i^{th} order Stage 1 kernels exhibit a synergistic complementary behaviour to contribute positively towards the calculation of an improved measure of similarity between their two input vector arguments. The experiments conducted in this section aim to validate such a hypothesis by exploring the resulting SVM classification accuracy scored by the Chebyshev, Legendre, and Hermite Stage 2 kernels, which are formulated with the sum (as defined in (4.10 - 4.12)), compared to the classification accuracy scored by their Stage 1 counterparts, which are formulated without the sum (as defined in (4.7 – 4.9)).

Figures 4.11 – 4.16 illustrate pairwise graphical comparisons of the best classification accuracies scored by the first few orders of these Stage 1 and Stage 2 kernels using all the datasets described earlier in Section 4.4.2. As shown, the kernels under test have demonstrated different performances across the polynomial orders and the datasets investigated. However, in general, one can clearly observe the consistent superiority of the classification accuracies scored by the Stage 2 kernels over their Stage 1 counterparts for most, if not all, of the polynomial orders and datasets under study, which clearly validates the synergy in the fusion by summation operation inherent in the construction of such polynomial kernels.

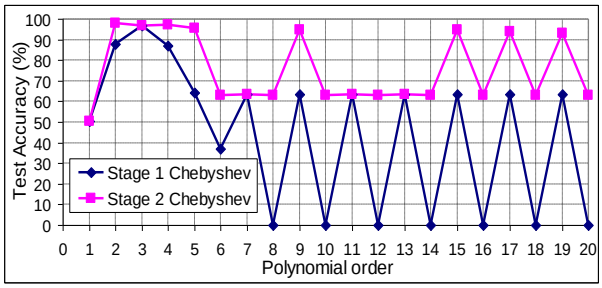
For example, in some datasets, such as the Breast Cancer, Ionosphere, and Two Spirals datasets, the improvement in the classification accuracy introduced by the Stage 2 kernels tend to be more dominant in higher order polynomial kernels than lower orders. Whereas in other datasets, such as the Image Segmentation and Iris datasets, it is the lower orders which benefited the most out of the fusion by summation process. Surprisingly, however, it also can be observed that the Hermite kernel was the one that benefited the most out of the fusion by summation process in the Thyroid dataset with only an infinitesimal improvement demonstrated by the Stage 2 Chebyshev and Legendre kernels, as shown in Figure 4.16.

Of particular interest, however, is the observation that the classification accuracy obtained from Stage 1 kernels can at times score a 0% accuracy, as shown for example in the results obtained from the Breast Cancer and the Ionosphere datasets, in Figures 4.11 and 4.12, respectively. Such a poor performance indeed indicates that Stage 1 kernels can at times struggle to decide on their own as to what is the correct class label of a test example that they have not seen before during the training phase, and hence a fusion by summation of their preceding i^{th} orders is essential to constructively contribute towards improving their generalization capabilities and hence their classification performance, as demonstrated by their Stage 2 kernels counterparts.

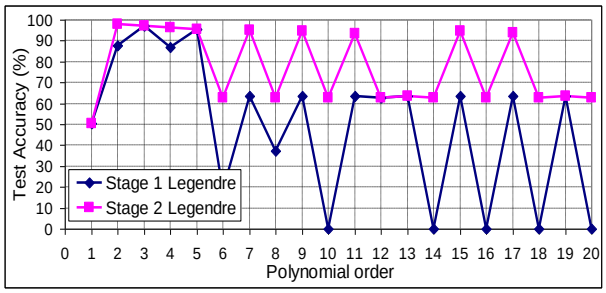
As these instances of 0% accuracy is an extreme case, the output of the SVM algorithm (as implemented by the employed Matlab toolbox) was investigated in more detail to find out what has exactly happened with the SVM classifier, both during the training and testing phases. As a result, it was found that at these instances the algorithm actually failed to make any decisions at all during the testing phase, and it was not even able to calculate a decision boundary during the training phase. The 0% accuracy observed at these instances therefore indicates a ‘classification inability’ status and not a ‘classification to an opposite

class' status, as explained earlier in Section 2.2.4. In Matlab language, the parameters of the dual optimisation problem calculated during the training phase, as well as the output of the decision function during the testing phase, were found to be not a number (NaN).

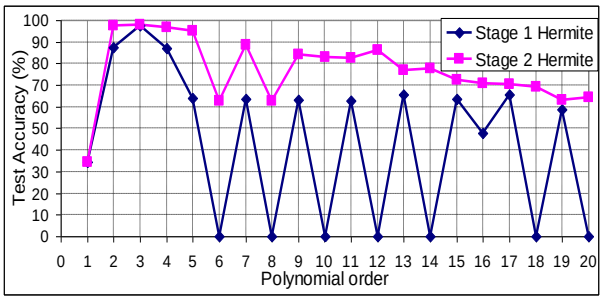
Furthermore, recall that Stage 1 kernels calculate relatively poor measures of similarity when their two inputs happen to be identical within the normalized vector space of [-1,+1], as explained in Section 4.3.2, as compared to the developed peaks of their Stage 2 counterparts. The superior results of Stage 2 over Stage 1 kernels illustrated in this section therefore demonstrate that such improvements in the calculated measures of similarity obtained by the synergistic fusion by summation of the i^{th} order Stage 1 kernels do indeed reflect constructively upon the resulting classification performance obtained from the Stage 2 kernels.



(a) Chebyshev kernels

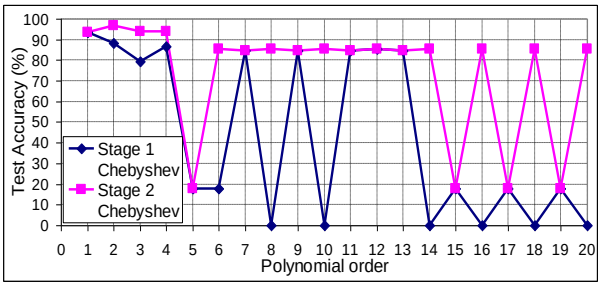


(b) Legendre kernels

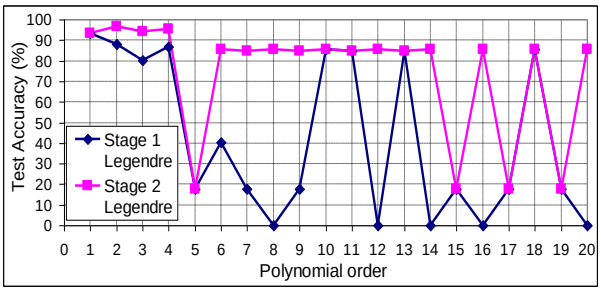


(c) Hermite kernels

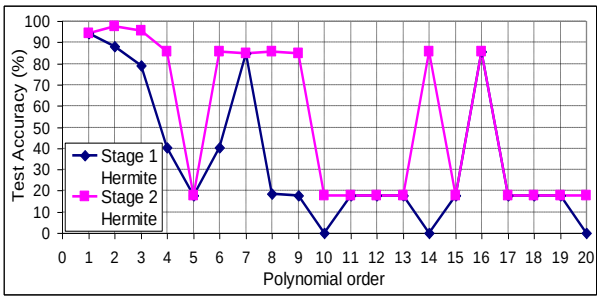
Figure 4.11 Classification accuracy scored by Stage 1 and Stage 2 kernels, using the Breast Cancer dataset



(a) Chebyshev kernels

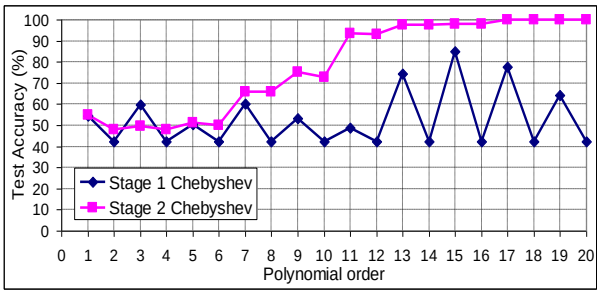


(b) Legendre kernels

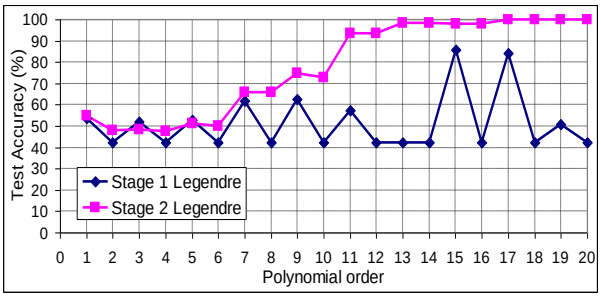


(c) Hermite kernels

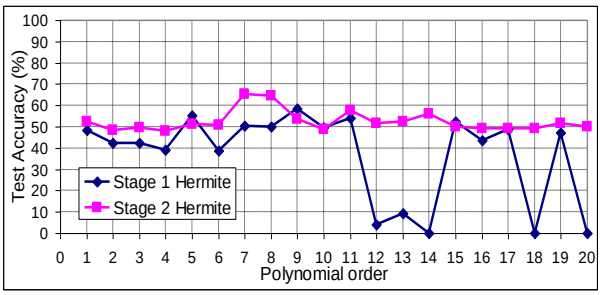
Figure 4.12 Classification accuracy scored by Stage 1 and Stage 2 kernels, using the Ionosphere dataset



(a) Chebyshev kernels

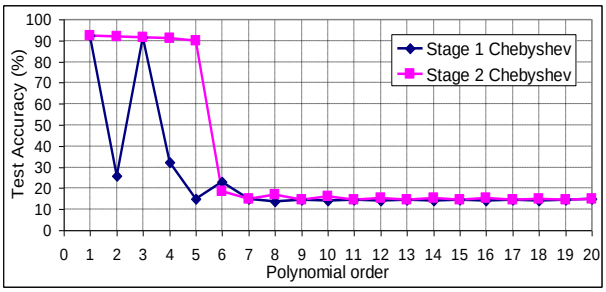


(b) Legendre kernels

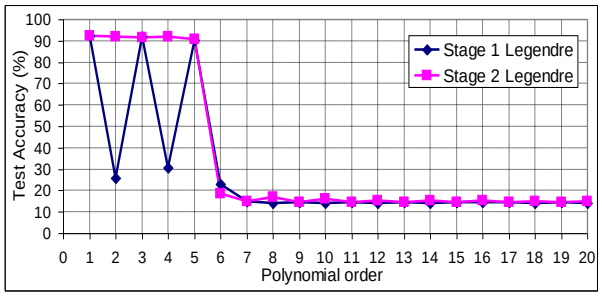


(c) Hermite kernels

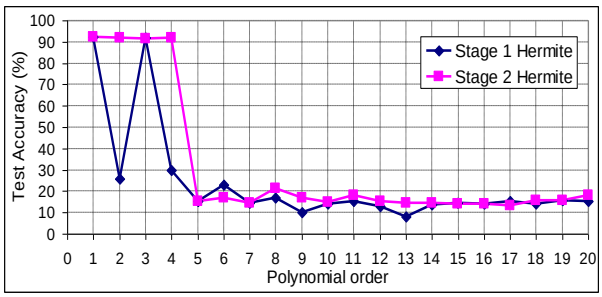
Figure 4.13 Classification accuracy scored by Stage 1 and Stage 2 kernels using the Two Spirals dataset



(a) Chebyshev kernels

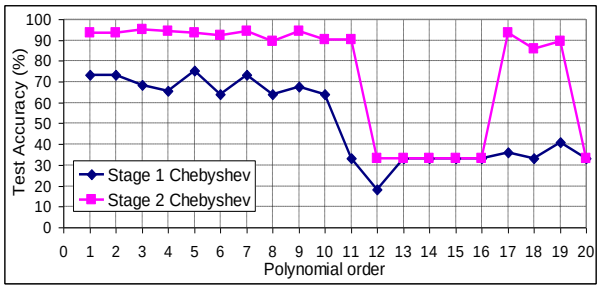


(b) Legendre kernels

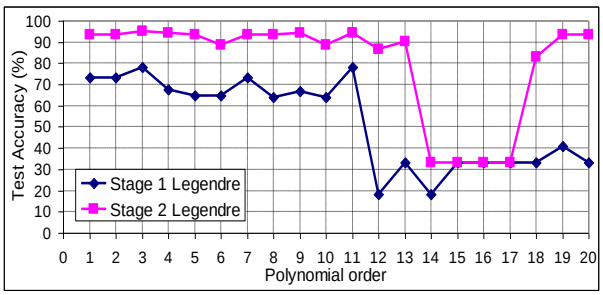


(c) Hermite kernels

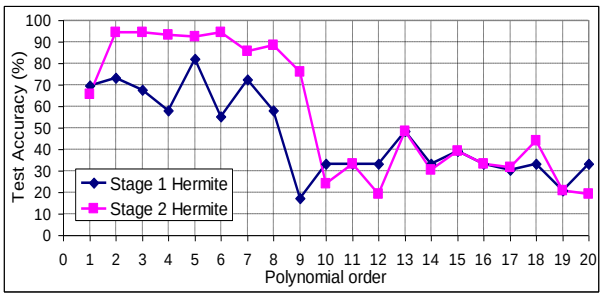
Figure 4.14 Classification accuracy scored by Stage 1 and Stage 2 kernels using the Image Segmentation dataset



(a) Chebyshev kernels

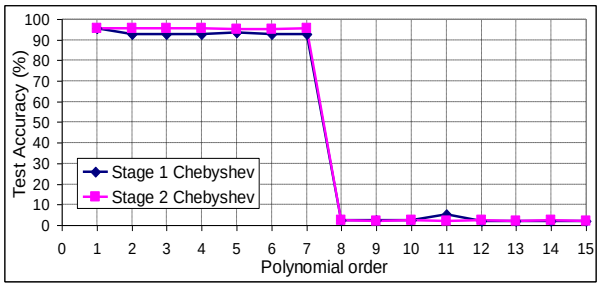


(b) Legendre kernels

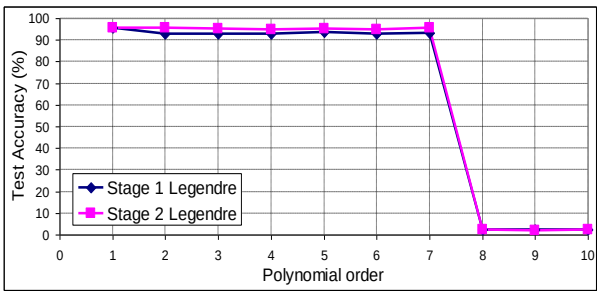


(c) Hermite kernels

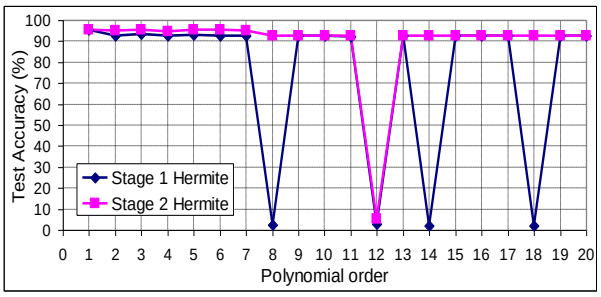
Figure 4.15 Classification accuracy scored by Stage 1 and Stage 2 kernels using the Iris dataset



(a) Chebyshev kernels



(b) Legendre kernels



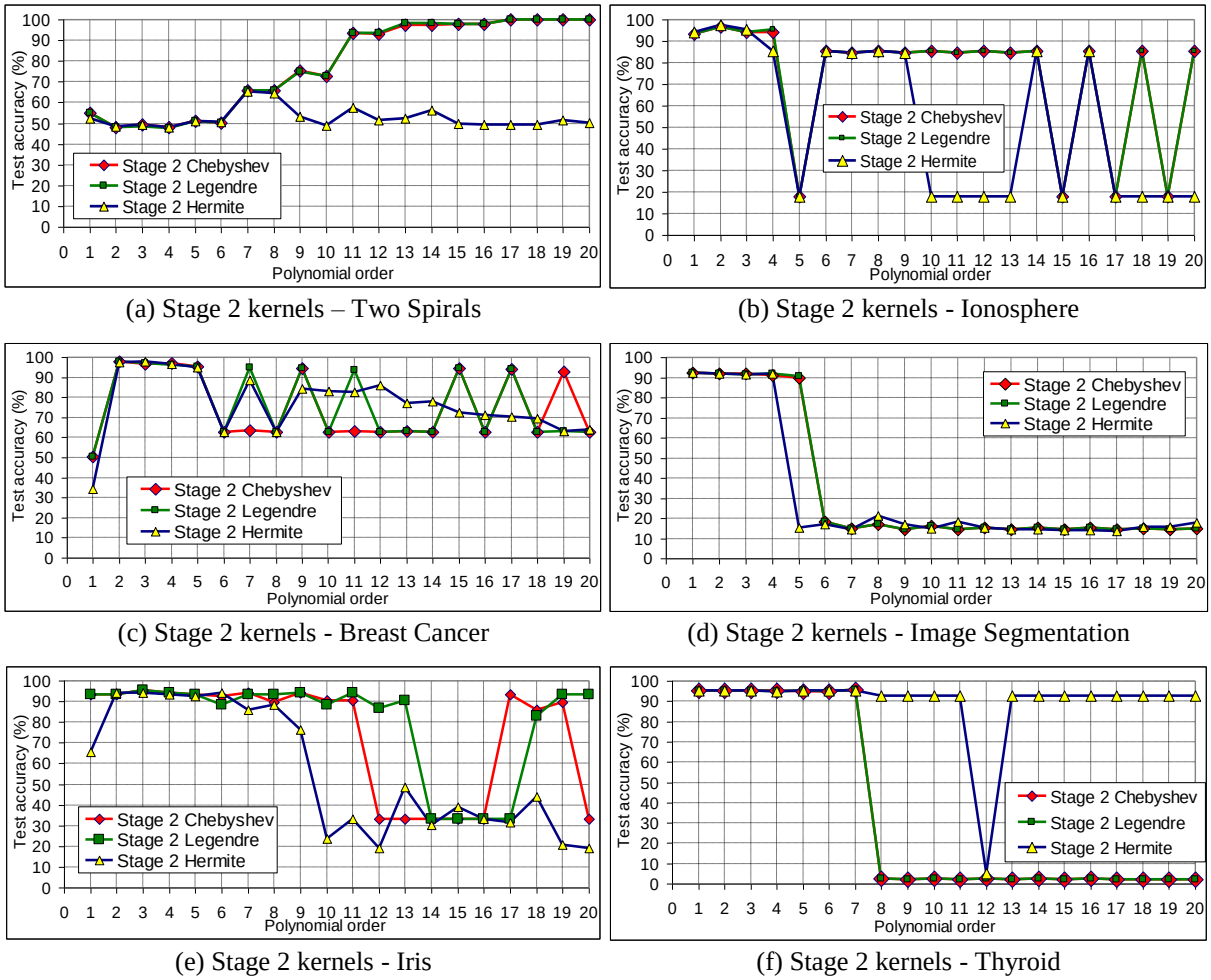
(c) Hermite kernels

Figure 4.16 Classification accuracy scored by Stage 1 and Stage 2 kernels using the Thyroid dataset

A further investigation has also been conducted on only Stage 2 kernels to explore how their performances compare within the datasets under test, and in particular in relation to their exhibited shape characteristics discussed earlier in Chapter 3 – Sections 3.4.2.2 and 3.4.2.3). Figure 4.17 therefore focuses on comparing these results obtained from only the Stage 2 kernels to show that the performance of the Stage 2 Chebyshev and Legendre kernels are relatively close to each other, and also relatively better than the results obtained from the Hermite kernels. Again, recall that the Stage 2 Chebyshev and Legendre kernels have shown to exhibit some common good shape characteristics that are not shared with the Hermite kernels, this could explain the reason behind their classification performance being closer to each other and better than the Hermite kernels, as observed in most of the results in this figure.

One can notice, however, that despite the Stage 2 Chebyshev and Legendre kernels have demonstrated a relatively superior performance to their Hermite counterpart for the Two Spirals, Ionosphere, Breast Cancer, Image Segmentation, and Iris datasets in Figure 4.17,

due to their better acquisition of similarity measures as discussed above, yet, this does not seem to be the case in the Thyroid dataset, especially in higher polynomial orders. This exceptional unique behaviour suggests that the influence of the similarity measures calculated by the underpinning kernels, as dictated by their shape characteristics, is not always prevail in all datasets (and hence in all pattern classification tasks), and that other factors (e.g., the uplifted dimensionality of the feature space, the severity of the non-linearity profile of the dataset at hand, the ratio of the train / test number of examples, etc.) can at times have a stronger influence on the kernels' classification performance. A deeper investigation into this point is indeed a subject of further research to explore in more detail what these possible factors could be.



(a) Stage 2 kernels – Two Spirals

(b) Stage 2 kernels - Ionosphere

(c) Stage 2 kernels - Breast Cancer

(d) Stage 2 kernels - Image Segmentation

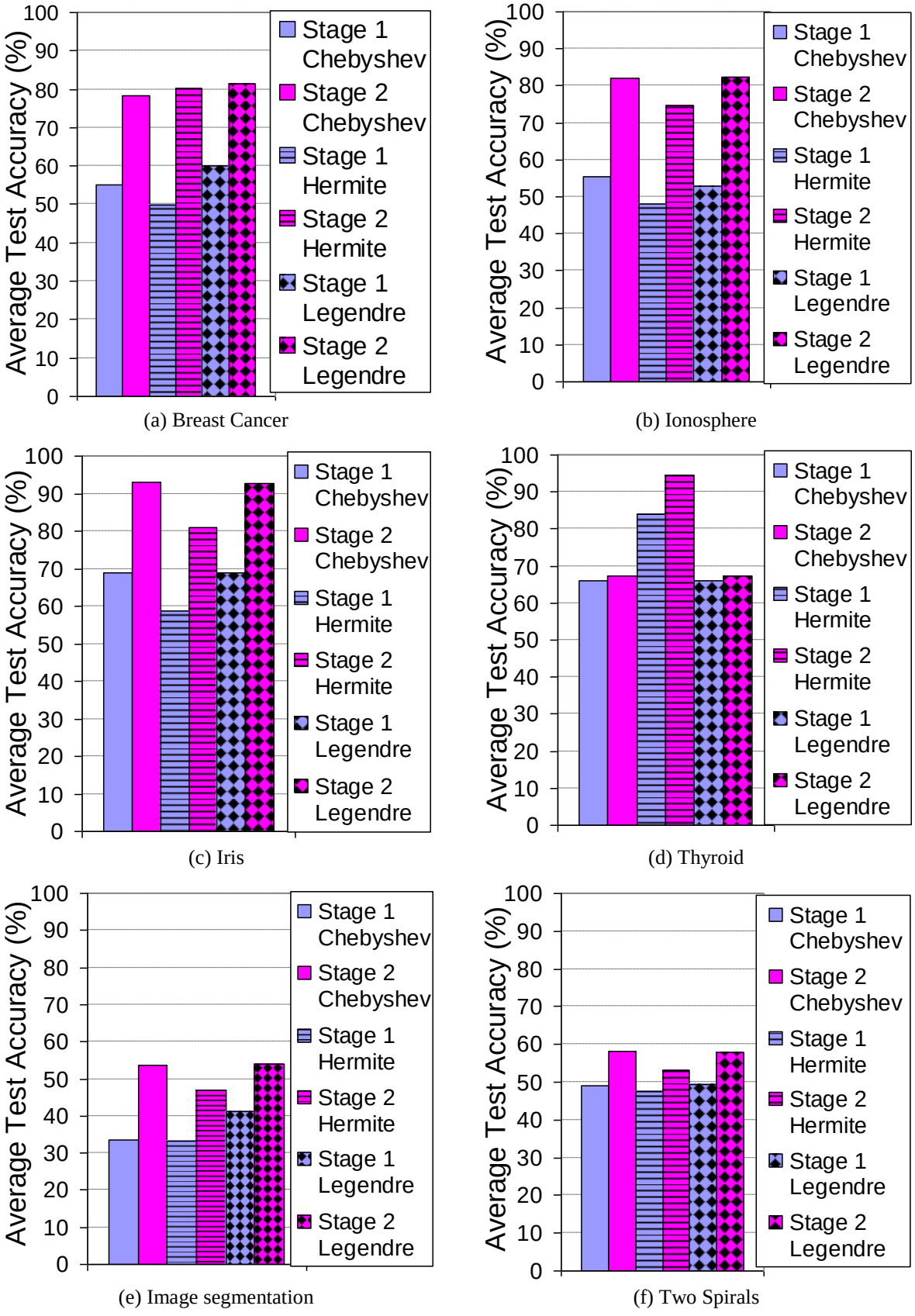
(e) Stage 2 kernels - Iris

(f) Stage 2 kernels - Thyroid

Figure 4.17 Comparison of the classification accuracy results for Stage 2 Chebyshev, Legendre, and Hermite kernels.

To further demonstrate the influence of the synergistic fusion by summation operation, Figure 4.18 also illustrates the obtained results in a different format using bar chart comparisons of the average classification accuracy scored by the first 10 orders of each of the Chebyshev, Legendre, and Hermite Stage 1 and Stage 2 kernels under test. One can observe the consistent superiority of the average classification accuracy scored by Stage 2

kernels over their Stage 1 counterparts for all the datasets investigated. As explained earlier, such improvement is evidently attributed to the synergistic fusion by summation process inherent in the Stage 2 kernels.



(a) Breast Cancer

(b) Ionosphere

(c) Iris

(d) Thyroid

(e) Image segmentation

(f) Two Spirals

Figure 4.18 Bar chart comparative results of the average classification accuracy scored by the first 10 orders of Stage 1 and Stage 2 kernels of the developed similarity fusion framework.

The classification accuracies scored by the Stage 1 and Stage 2 kernels are also analyzed more deeply in Table 4.2 to get a more quantitative assessment of how much improvement is gained as a result of the fusion by summation process. As such, the table is used to calculate the average improvements in the classification accuracy over the first 10 polynomial orders for all the datasets by directly subtracting the accuracies scored by Stage 1 kernels from their Stage 2 counterparts. As illustrated at the bottom of the table, the average improvement factors were calculated to be equal to 17.574%, 18.219%, and, 16.249% for the Chebyshev, Hermite, and Legendre polynomial kernels, respectively, making up an average accuracy improvement factor of 17.347% over all the datasets experimented.

Another observation worth noting is that the fusion by summation operation taking place in Stage 2 kernels resulted in rectifying the cases where their Stage 1 counterparts suffered from the 'classification inability' instances where they produced 0% accuracy, as demonstrated, for example, by the Breast Cancer and Ionosphere datasets in Figures 4.11 and 4.12, respectively. Consequently, one can therefore notice that the Stage 2 kernels have exhibited a relatively smoother performance across the spectrum of polynomial orders compared to their Stage 1 counterparts, as the amplitude of the accuracy oscillations have been reduced considerably amongst the polynomial orders.

To quantify this observation, Table 4.2 has also been used to calculate the standard deviation of the average improvement factors (shown in brackets), as well as the minimum and maximum improvement factors (shown in bold), introduced by the fusion by summation operation inherent in Stage 2 kernels. As it can be noticed, the standard deviation of the average improvement factors can be quite sparse and even higher than the mean, as shown for example in the Two Spirals and Image segmentation datasets, due to the huge difference between the two sets of results across the polynomial orders. Conversely, however, one can also realize the steadiness of the results by examining the standard deviation of the accuracy values themselves, reported in the statistical analysis tests tabulated in Appendix B.1, where the standard deviation of the classification accuracies scored by Stage 2 kernels is relatively smaller than the mean, compared to their Stage 1 counterparts, for most of the results obtained.

Table 4.2 Quantitative assessment of the average improvement in classification accuracy when using Stage 2 over Stage 1 kernels of the developed similarity fusion framework.

		Stage 1 & Stage 2 Cheb. Kernels										Stage 1 & Stage 2 Hermite kernels										Stage 1 & Stage 2 Legendre kernels									
		Order no.										Order no.										Order no.									
		1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
Two Spirals	**N1**	54.600	42.243	59.771	42.243	50.297	42.243	60.046	42.243	53.112	42.243	48.284	42.243	42.243	39.085	55.309	38.558	50.481	49.954	58.627	49.611	53.730	42.243	51.968	42.243	52.930	42.243	61.968	42.243	62.586	42.243
	N2	54.874	48.123	49.519	48.032	51.190	50.046	65.995	65.652	75.378	72.746	52.494	48.558	49.519	48.032	51.190	50.663	65.469	64.600	53.455	48.901	54.874	48.124	48.467	47.506	51.190	50.046	65.995	65.652	74.851	72.746
	I	0.275	5.881	**-10.252**	5.789	0.892	7.803	5.950	23.410	22.265	**30.503**	4.211	6.316	7.277	8.947	-4.119	12.105	**14.989**	14.645	**-5.172**	-0.709	1.144	5.881	**-3.501**	5.263	-1.739	7.803	4.027	23.410	12.265	**30.503**
	AI	**9.252 (12.421)**										**5.849 (7.2909)**										**8.506 (10.86)**									
Breast Cancer	**N1**	50.320	87.633	96.588	86.994	64.179	37.100	63.539	0	63.539	0	34.328	87.633	97.655	86.994	63.966	0	63.539	0	63.326	0	50.320	87.633	97.228	86.994	95.522	18.337	63.539	37.100	63.539	0
	N2	50.320	97.868	96.802	97.228	95.522	62.900	63.539	62.900	94.670	62.900	34.328	97.655	97.868	96.588	95.096	62.900	88.699	62.900	84.222	82.942	50.320	97.868	97.015	96.162	95.309	62.900	95.096	62.900	94.670	62.900
	I	**0**	10.235	0.213	10.235	31.343	25.800	**0**	62.900	31.130	**62.9**	**0**	10.021	0.213	9.595	31.130	62.900	25.160	62.900	20.896	**82.942**	0	10.234	**-0.213**	9.168	-0.213	44.563	31.557	25.800	31.130	**62.9**
	AI	**23.476 (24.103)**										**30.576 (29.202)**										**21.493 (21.454)**									
Iris	**N1**	73.333	73.333	68.571	65.714	75.238	63.810	73.333	63.81	67.619	63.810	69.524	73.333	67.619	58.095	81.905	55.238	72.381	58.095	17.143	33.333	73.333	73.333	78.095	67.619	64.762	64.762	73.333	63.810	66.667	63.810
	N2	93.333	93.333	95.238	94.286	93.333	92.381	94.286	89.524	94.286	90.476	65.714	94.286	94.286	93.333	92.381	94.286	85.714	88.571	76.190	23.810	93.333	93.333	95.238	94.286	93.333	88.571	93.333	93.333	94.286	88.571
	I	20.000	20.000	26.667	28.571	**18.095**	**28.571**	20.952	25.714	26.667	26.667	-3.81	20.952	26.667	35.238	10.476	39.048	13.333	30.476	**59.048**	**-9.524**	20.000	20.000	**17.143**	26.667	28.571	23.81	20.000	**29.524**	27.619	24.762
	AI	**24.190 (3.97)**										**22.190 (20.55)**										**23.810 (4.306)**									
Image Segmentation	**N1**	92.143	25.714	91.333	32.095	14.905	22.857	15.000	13.857	14.619	14.286	92.190	25.762	91.762	29.667	15.381	22.857	14.619	16.952	10.143	14.286	92.143	25.762	91.762	30.524	90.524	22.857	15.000	15.000	14.286	14.286
	N2	92.143	92.095	91.714	91.238	90.000	18.381	15.000	16.857	14.619	16.143	92.190	92.095	91.667	92.095	15.381	17.095	14.619	21.190	17.048	15.095	92.143	92.095	91.571	92.048	90.810	18.429	15.000	16.857	14.667	16.143
	I	0	66.381	0.381	59.143	**75.095**	**-4.476**	0	3.000	0	1.857	0	**66.333**	-0.095	62.429	0	**-5.762**	0	4.238	6.905	0.810	0	**66.333**	-0.190	61.524	0.286	**-4.429**	0	2.571	0.048	1.857
	AI	**20.138 (40.738)**										**13.486 (27.036)**										**12.8 (27.032)**									
Ionosphere	**N1**	93.377	88.079	79.470	86.755	17.881	17.881	84.768	0	84.768	0	94.040	88.079	78.808	40.397	17.881	40.397	84.768	18.543	17.881	0	93.377	88.079	80.132	86.755	17.881	40.397	17.881	0	17.881	85.430
	N2	93.377	96.689	94.040	94.040	17.881	85.430	84.768	85.430	84.768	85.430	94.040	97.351	95.364	85.430	17.881	85.430	84.768	85.430	84.768	17.881	93.377	96.689	94.040	95.364	17.881	85.430	84.768	85.430	84.768	85.430
	I	**0**	8.609	14.570	7.285	**0**	67.550	**0**	**85.430**	**0**	**85.430**	**0**	9.272	16.556	45.033	**0**	45.033	**0**	**66.887**	**66.887**	17.881	**0**	8.609	13.907	8.609	**0**	45.033	66.887	**85.430**	66.887	**0**
	AI	**26.887 (36.923)**										**26.755 (26.926)**										**29.536 (33.152)**									

		Stage 1 & Stage 2 Cheb. Kernels										Stage 1 & Stage 2 Hermite kernels										Stage 1 & Stage 2 Legendre kernels									
		Order no.										Order no.										Order no.									
		1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
Thyroid	**N1**	95.478	92.707	92.707	92.707	93.553	92.707	92.707	2.275	2.275	2.275	95.478	92.707	93.466	92.707	92.853	92.707	92.736	2.275	92.707	92.707	95.478	92.707	92.970	92.707	93.495	92.707	93.320	2.275	2.275	2.275
	N2	95.478	95.449	95.420	95.420	95.012	95.012	95.683	2.392	2.159	2.363	95.478	95.216	95.449	94.749	95.303	95.361	95.274	92.707	92.707	92.707	95.478	95.449	95.391	94.866	95.070	95.012	95.537	2.392	2.159	2.363
	I	0	2.742	2.713	2.713	1.459	2.305	**2.975**	0.117	**-0.117**	0.088	**0**	2.509	1.984	2.042	2.450	2.655	2.538	**90.432**	**0**	**0**	0	**2.742**	2.421	2.159	1.575	2.3046	2.217	0.117	**-0.117**	0.088
	AI	**1.499 (1.337)**										**10.461 (28.122)**										**1.351 (1.18)**									
AAI		**17.574 (23.85)**										**18.219 (25.023)**										**16.249 (21.551)**									
TAI		**17.347 (23.402)**																													

N1 Classification accuracy of Stage 1 kernels
N2 Classification accuracy of Stage 2 kernels
I Improvement = N2-N1
AI Average Improvement for the first 10 orders
AAI Average of Average Improvements of the first 10 orders over all the datasets under investigation
TAI Total Average Improvement over all datasets and all kernels under investigation for the first 10 orders

It should not be forgotten, however, that despite such a substantial improvement in the classification performance demonstrated by Stage 2 kernels, due to the fusion by summation of the i^{th} order Stage 1 kernels, yet, they are still not robust enough to compete with existing kernels to reliably solve real-world classification problems. Hence, a progression towards fusing them (i.e., Stage 2 kernels) with other kernels is necessary to up-scale their generalisation capabilities and hence improve their classification performance even further. This is achieved via the construction of the Stage 3 kernels of the similarity fusion framework presented in this chapter, and their classification performance is explored in the next section.

4.4.5 Experimental results and discussions for Stage 2 and Stage 3 kernels of the similarity fusion framework

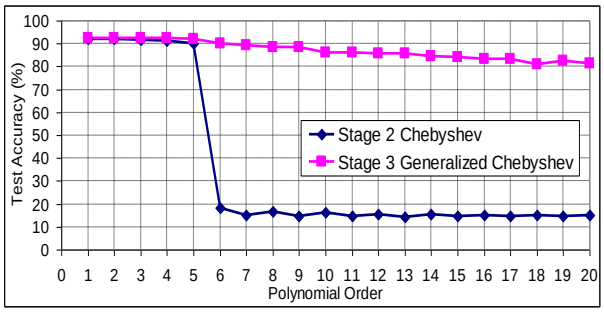
The experiments conducted in this section aim to explore the resulting effect on the SVM classification accuracy due to fusing (by multiplication) Stage 2 kernels with other kernels (e.g., the Gaussian kernel) to formulate the composite Stage 3 kernels of the similarity fusion framework. To be specific, the comparative experiments were conducted between the Stage 2 kernels, defined in (4.10 – 4.12), versus their composite Stage 3 counterparts, defined in (4.13 – 4.16).

Figures 4.19 – 4.24 demonstrate pairwise graphical comparisons of the best classification accuracies scored by the first few orders of these Stage 2 and Stage 3 kernels using all the datasets described earlier in Section 4.4.2. As shown, the kernels under test have once again demonstrated different performances across the polynomial orders and the datasets investigated. However, on average, it can be clearly observed that Stage 3 kernels have demonstrated a quite superior performance over their Stage 2 counterparts, for most of the polynomial orders and datasets investigated.

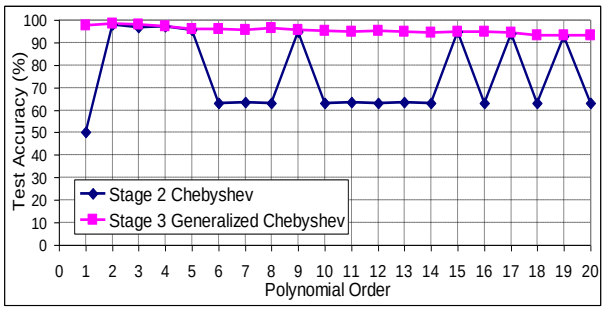
Despite this performance superiority (of Stage 3 over Stage 2 kernels), yet, the improvement in the classification accuracy introduced by the Stage 3 kernels over the polynomial orders does not seem to be consistent throughout all the datasets. In other words, depending on the characteristics of the dataset (e.g., number of features, number of classes, the severity of the non-linearity profile of the overlapped class labels, etc.) the improvement in the classification accuracy introduced by the Stage 3 kernels can be more dominant in higher polynomial orders than lower orders, as can be observed, for example, in the Image Segmentation, Iris, and Thyroid datasets, shown in Figures 4.19, 4.23, and 4.24, respectively. Whereas in other datasets, such as the Two Spirals and Ionosphere datasets, shown in Figures 4.21 and 4.22, respectively, it is the lower orders which benefited the most out of the fusion by multiplication process.

Either way, this demonstrates that the fusion by multiplication process of Stage 2 kernels by their corresponding weighting functions, or other kernels (such as the Gaussian), is indeed synergistic as it leads to better classification performance. Inline with the information fusion theory explained earlier in this chapter, this observation validates that, to the extent that each of the Stage 2 kernels and their combining kernels provide complementary information about the input data, the performance of the classifier constructed from their fused combination does indeed outperform the performance achieved when the Stage 2 kernels are used on their own.

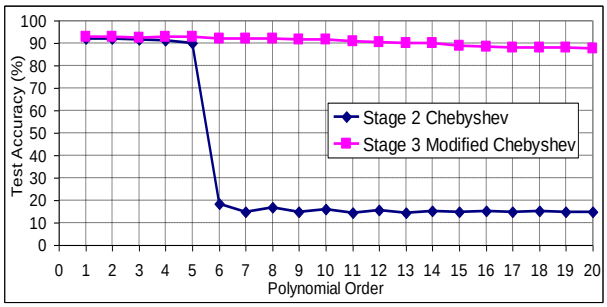
Another observation also worth noting is the fact that when Stage 2 kernels are fused with a more robust kernel than their own corresponding weighting functions, they can actually score even better classification accuracies, showing that the weighting functions are not always guaranteed to be the optimum choice. For example, this can be clearly observed here from the results obtained from the Legendre kernels. One can recall that the corresponding weighting function of the Legendre polynomials is $w(x)=1$ [34], and therefore constitute the Stage 2 status in the framework presented in this chapter. For their Stage 3 counterpart, however, it has been combined with the Gaussian kernel instead, as per (4.15) (hence the name Modified Legendre kernel), which resulted in a substantial improvement in the classification accuracy, as demonstrated in the majority of the results herein, such as the Image Segmentation, Breast Cancer, and Two Spirals datasets, shown in Figures 4.19, 4.20, and 4.21, respectively. A similar behaviour can also be observed with the Stage 2 Chebyshev kernels, where they tend to score higher classification accuracies when combined with the Gaussian kernel (i.e., the Stage 3 Modified Chebyshev kernels) compared to when combined with their corresponding weighting function (i.e., the Stage 3 Generalized Chebyshev kernels), as can be realized for example from the results obtained from the Image Segmentation and the Iris datasets shown in Figures 4.19 and 4.23, respectively. Thanks to the kernels' multiplicative closure property (amongst the others) that allows us to legitimately adapt the kernels to achieve best possible performance, as this is what we are ultimately aiming for, rather than retaining the orthogonality of the underpinning polynomial kernels unnecessarily.



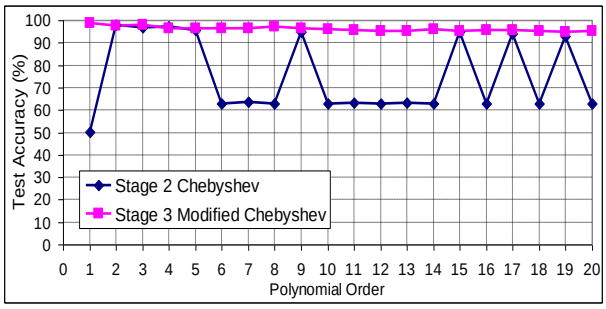
(a) Stage 2 Chebyshev & Stage 3 Generalised Chebyshev kernels



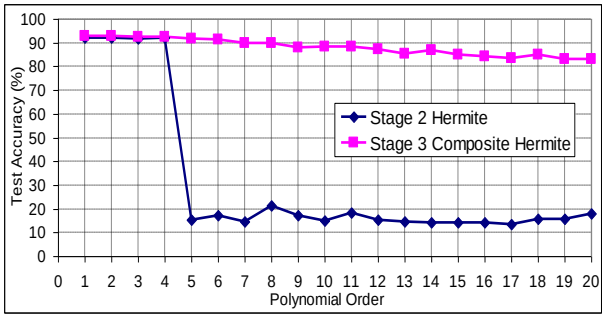
(a) Stage 2 Chebyshev & Stage 3 Generalised Chebyshev kernels



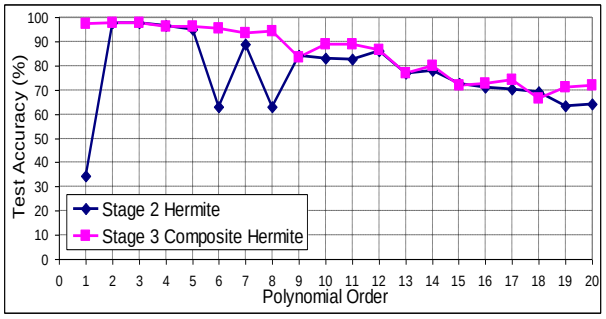
(b) Stage 2 Chebyshev & Stage 3 Modified Chebyshev kernels



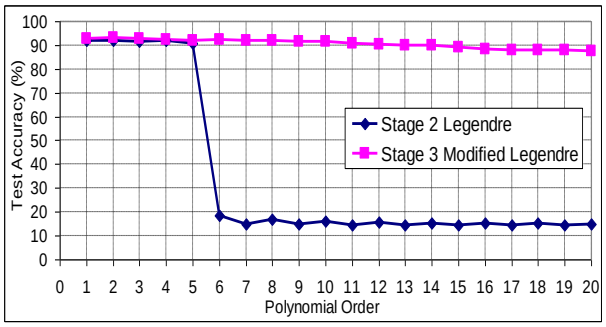
(b) Stage 2 Chebyshev & Stage 3 Modified Chebyshev kernels



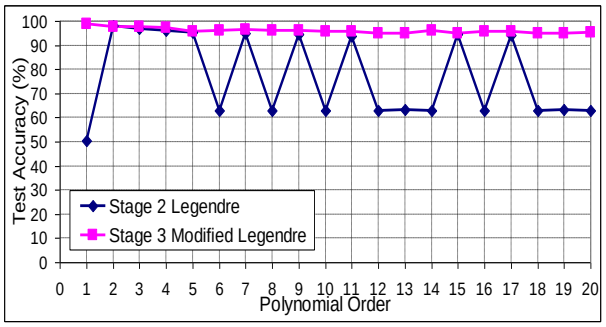
(c) Stage 2 Hermite & Stage 3 Composite Hermite kernels



(c) Stage 2 Hermite & Stage 3 Composite Hermite kernels



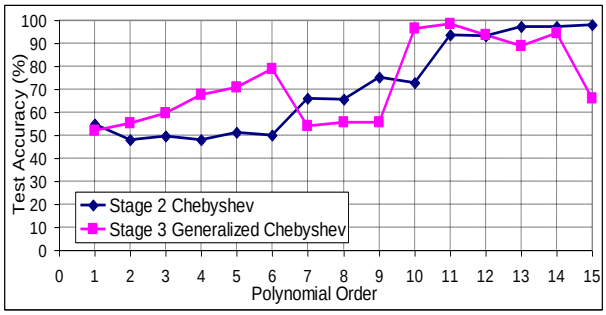
(d) Stage 2 Legendre & Stage 3 Modified Legendre kernels



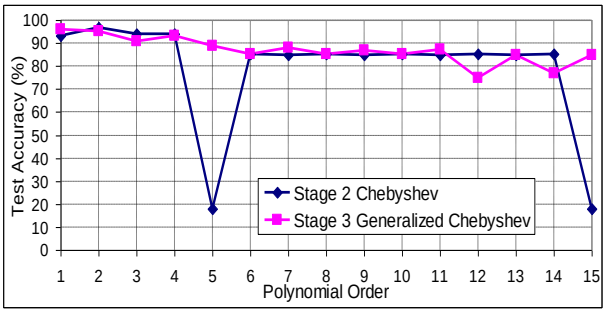
(d) Stage 2 Legendre & Stage 3 Modified Legendre kernels

Figure 4.19 Classification accuracy scored by Stage 2 and Stage 3 kernels using the Image Segmentation dataset

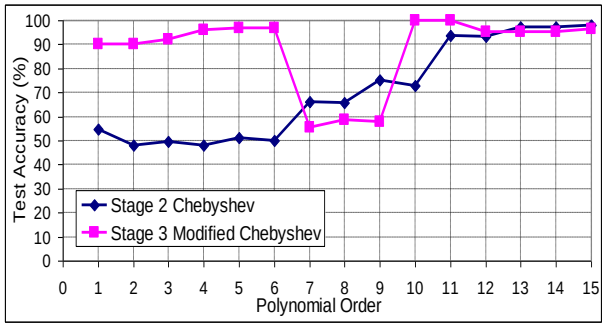
Figure 4.20 Classification accuracy scored by Stage 2 and Stage 3 kernels using the Breast Cancer dataset



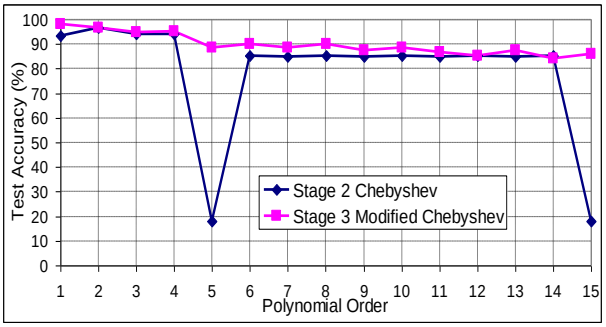
(a) Stage 2 Chebyshev & Stage 3 Generalised Chebyshev kernels



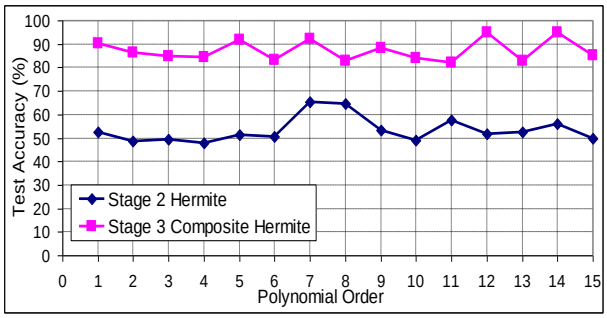
(a) Stage 2 Chebyshev & Stage 3 Generalised Chebyshev kernels



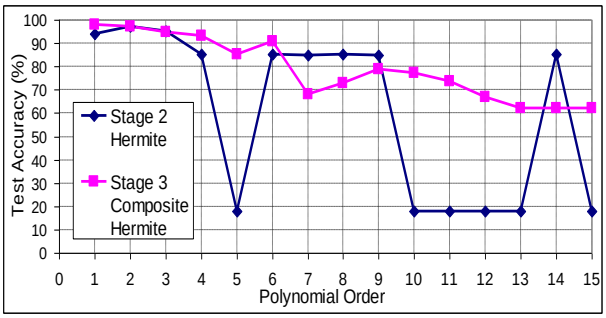
(b) Stage 2 Chebyshev & Stage 3 Modified Chebyshev kernels



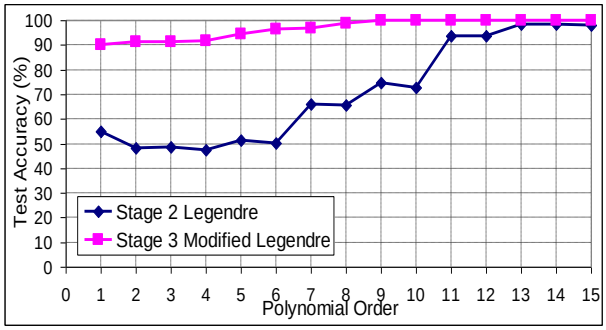
(b) Stage 2 Chebyshev & Stage 3 Modified Chebyshev kernels



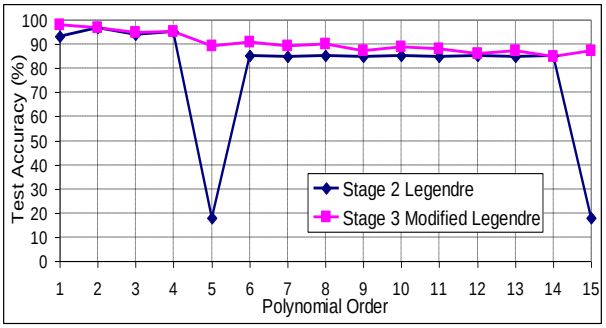
(c) Stage 2 Hermite & Stage 3 Composite Hermite kernels



(c) Stage 2 Hermite & Stage 3 Composite Hermite kernels



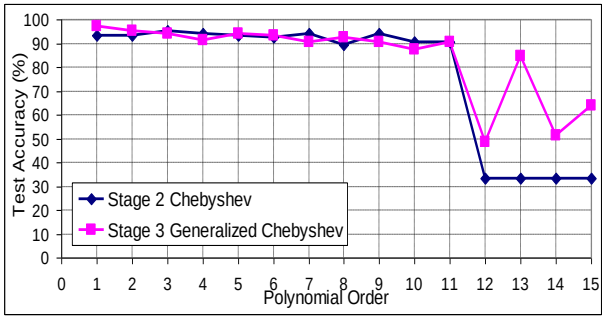
(d) Stage 2 Legendre & Stage 3 Modified Legendre kernels



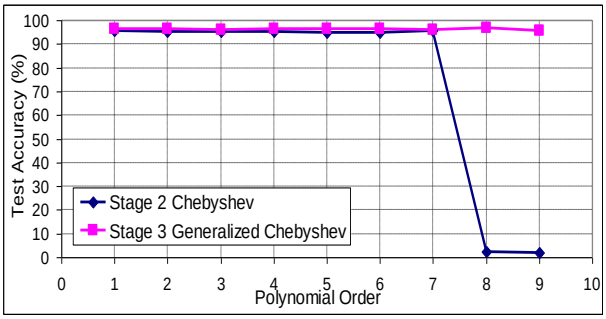
(d) Stage 2 Legendre & Stage 3 Modified Legendre kernels

Figure 4.21 Classification accuracy scored by Stage 2 and Stage 3 kernels using the Two Spirals dataset.

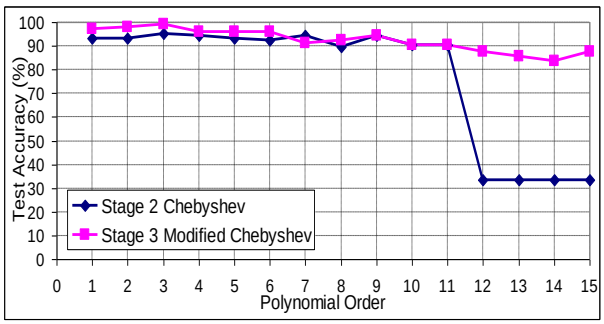
Figure 4.22 Classification accuracy scored by Stage 2 and Stage 3 kernels using the Ionosphere dataset.



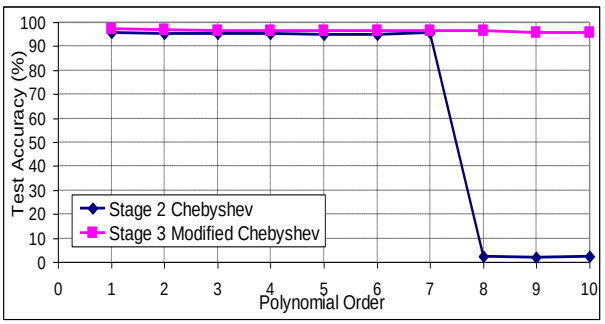
(a) Stage 2 Chebyshev & Stage 3 Generalised Chebyshev kernels



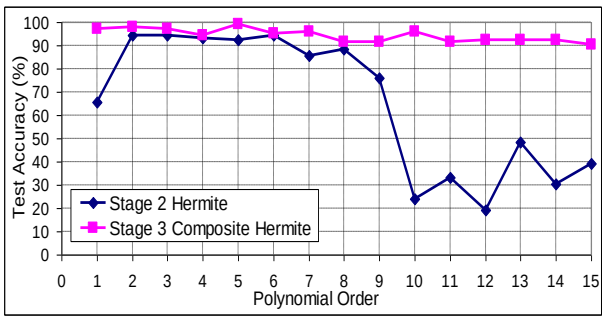
(a) Stage 2 Chebyshev & Stage 3 Generalised Chebyshev kernels



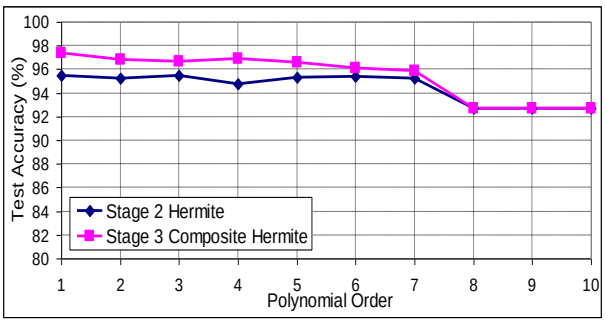
(b) Stage 2 Chebyshev & Stage 3 Modified Chebyshev kernels



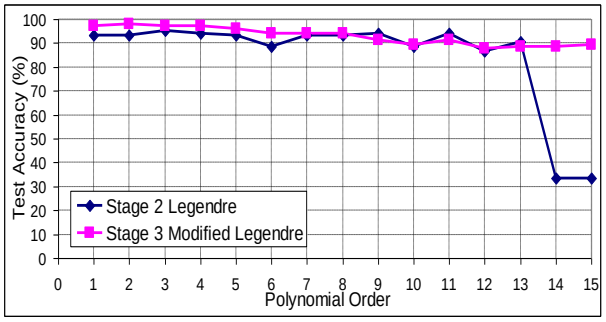
(b) Stage 2 Chebyshev & Stage 3 Modified Chebyshev kernels



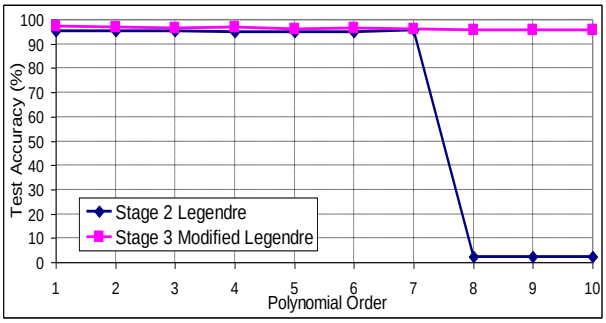
(c) Stage 2 Hermite & Stage 3 Composite Hermite kernels



(c) Stage 2 Hermite & Stage 3 Composite Hermite kernels



(d) Stage 2 Legendre & Stage 3 Modified Legendre kernels



(d) Stage 2 Legendre & Stage 3 Modified Legendre kernels

Figure 4.23 Classification accuracy scored by Stage 2 and Stage 3 kernels using the Iris dataset.

Figure 4.24 Classification accuracy scored by Stage 2 and Stage 3 kernels using the Thyroid dataset.

Figure 4.25 illustrates bar chart comparisons of the average classification accuracy scored by the first 10 orders of each of the Stage 2 and Stage 3 kernels under test, again demonstrating the substantial classification accuracy improvement resulting from the synergistic fusion by multiplication process, for most, if not all of, the kernels and the datasets investigated. To quantify this improvement, Table 4.3 is also used to calculate the average improvement factors gained in the classification accuracy as a results of using Stage 3 kernels over their Stage 2 counterparts. By averaging over the first 10 orders for all

the datasets, these improvement factors were calculated to be equal to 14.571%, 20.479%, 19.672%, and 21.91%, for the Stage 3 Generalized Chebyshev, Modified Chebyshev, Composite Hermite, and Modified Legendre kernels, respectively, making up an average accuracy improvement factor of 19.158%.

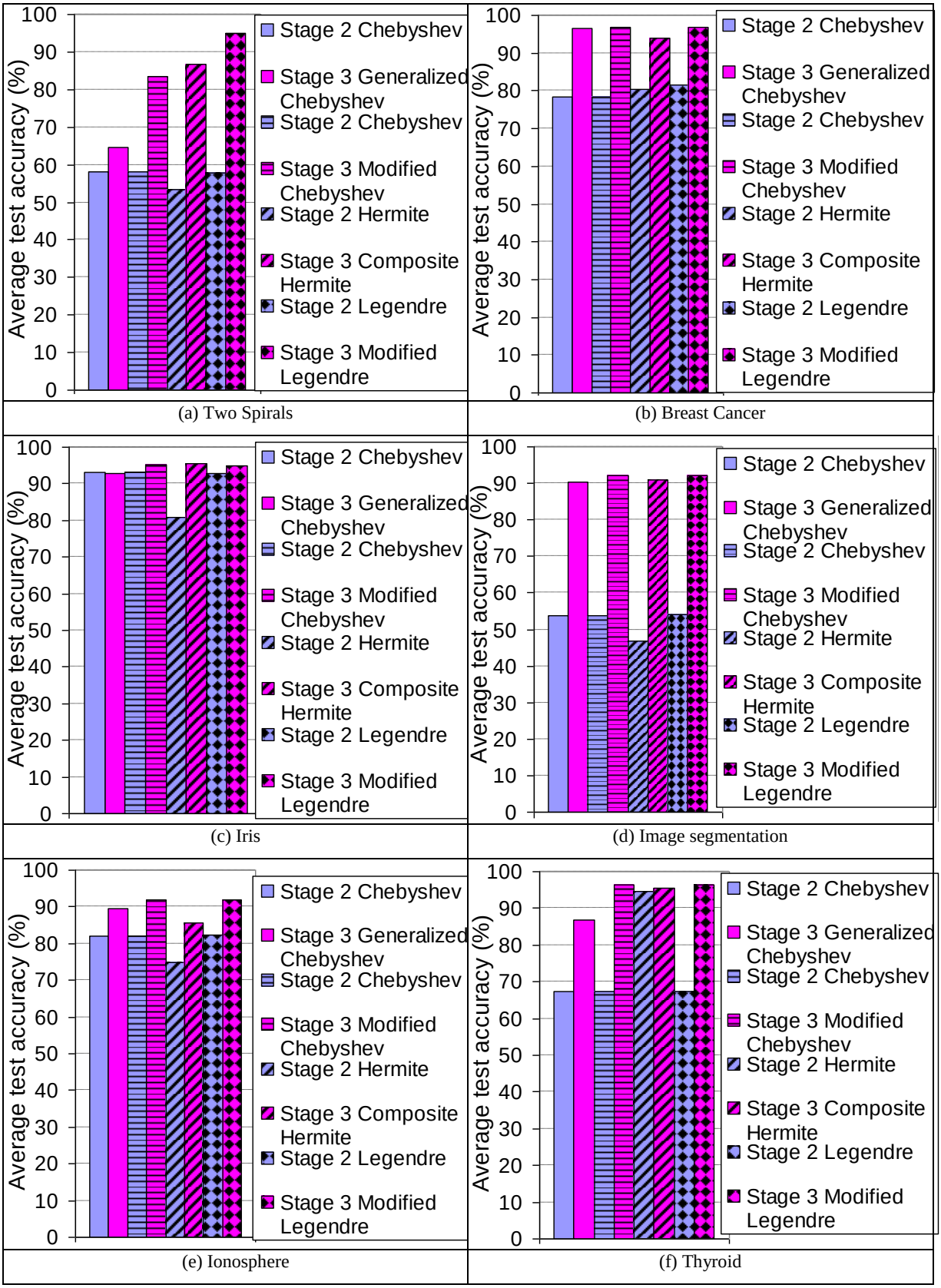


Figure 4.25 Bar chart comparative assessment of the average classification accuracy scored by the first 10 orders of Stage 2 and Stage 3 kernels of the developed similarity fusion framework.

Table 4.3 Quantitative assessment of the average improvements in classification accuracy when using Stage 3 over Stage 2 kernels of the similarity fusion framework.

		Stage 2 Chebyshev & Stage 3 Generalised Chebyshev kernels									
		Order no.									
		1	2	3	4	5	6	7	8	9	10
Two Spirals	N1	54.874	48.124	49.519	48.032	51.190	50.046	65.995	65.652	75.378	72.746
	N2	52.151	55.217	59.771	67.483	70.641	78.947	53.822	55.400	55.400	96.316
	I	-2.723	7.094	10.252	19.451	19.451	**28.902**	-12.174	-10.252	**-19.977**	23.570
	AI	6.359 (16.86)									
Breast Cancer	N1	50.320	97.868	96.802	97.228	95.522	62.900	63.539	62.900	94.670	62.900
	N2	97.441	98.294	97.868	97.015	95.949	95.949	95.522	96.375	95.522	95.309
	I	**47.122**	0.426	1.066	**-0.213**	0.426	33.049	31.983	33.475	0.853	32.409
	AI	18.060 (18.995)									

		Stage 2 Chebyshev & Stage 3 Modified Chebyshev kernels									
		Order no.									
		1	2	3	4	5	6	7	8	9	10
Two Spirals	N1	54.874	48.124	49.519	48.032	51.190	50.046	65.995	65.652	75.378	72.746
	N2	90.092	90.092	92.197	95.881	96.842	96.842	55.400	58.810	57.757	100.00
	I	35.217	41.968	42.677	**47.849**	45.652	46.796	-10.595	-6.842	**-17.620**	27.254
	AI	25.236 (26.311)									
Breast Cancer	N1	50.320	97.868	96.802	97.228	95.522	62.900	63.539	62.900	94.700	62.900
	N2	98.721	97.441	97.868	96.588	96.375	96.375	96.375	97.228	96.375	95.949
	I	**48.401**	-0.426	1.066	**-0.640**	0.853	33.475	32.836	34.328	1.706	33.049
	AI	18.465 (19.459)									

		Stage 2 Hermite & Stage 3 Composite Hermite kernels									
		Order no.									
		1	2	3	4	5	6	7	8	9	10
Two Spirals	N1	52.494	48.558	49.519	48.032	51.190	50.664	65.469	64.600	53.455	48.902
	N2	90.092	86.407	84.920	84.577	91.670	83.432	92.288	82.815	88.169	83.867
	I	37.597	37.849	35.400	**40.481**	36.545	32.769	26.819	**18.215**	34.714	34.966
	AI	33.535 (6.496)									
Breast Cancer	N1	34.328	97.655	97.868	96.588	95.096	62.900	88.699	62.900	84.222	82.942
	N2	97.441	97.655	97.655	96.162	96.162	95.309	93.603	94.243	83.582	88.699
	I	**63.113**	0	-0.213	-0.426	1.066	32.409	4.904	31.343	**-0.640**	5.757
	AI	13.731 (21.575)									

		Stage 2 Legendre & Stage 3 Modified Legendre kernels									
		Order no.									
		1	2	3	4	5	6	7	8	9	10
Two Spirals	N1	54.874	48.124	48.467	47.506	51.190	50.046	65.995	65.652	74.851	72.746
	N2	90.092	91.144	91.144	91.670	94.302	96.407	96.842	98.947	100	100
	I	35.217	43.021	42.677	44.165	43.112	**46.362**	30.847	33.295	**25.149**	27.254
	AI	37.120 (7.709)									
Breast Cancer	N1	50.320	97.868	97.015	96.162	95.309	62.900	95.096	62.900	94.670	62.900
	N2	98.720	97.655	97.655	97.441	95.736	96.162	96.375	95.949	95.949	95.522
	I	**48.401**	**-0.213**	0.640	1.279	0.426	33.262	1.279	33.049	1.279	32.623
	AI	15.203 (19.148)									

Ionosphere				Image Segmentation				Iris					
AI	I	N2	N1	AI	I	N2	N1	AI	I	N2	N1	Order no.	
7.351 (22.4)	2.649	96.026	93.377	36.605 (37.613)	**0.238**	92.381	92.143	-0.381 (2.847)	**3.810**	97.143	93.333	1	Stage 2 Chebyshev & Stage 3 Generalised Chebyshev kernels
	-1.325	95.364	96.689		**0.238**	92.333	92.095		1.905	95.238	93.333	2	
	-3.311	90.728	94.040		0.905	92.619	91.714		-0.952	94.286	95.238	3	
	-0.662	93.377	94.040		1.381	92.619	91.238		-2.857	91.429	94.286	4	
	70.861	88.742	17.881		1.952	91.952	90.000		0.952	94.286	93.333	5	
	0	85.430	85.430		71.619	90.000	18.381		0.952	93.333	92.381	6	
	3.311	88.079	84.768		**74.429**	89.429	15.000		**-3.810**	90.476	94.286	7	
	0	85.430	85.430		71.524	88.381	16.857		2.857	92.381	89.524	8	
	1.987	86.755	84.768		73.714	88.333	14.619		**-3.810**	90.476	94.286	9	
	0	85.430	85.430		70.048	86.190	16.143		-2.857	87.619	90.476	10	
9.669 (21.569)	4.636	98.013	93.377	38.429 (39.146)	**0.714**	92.857	92.143	2.095 (2.367)	3.810	97.143	93.333	1	Stage 2 Chebyshev & Stage 3 Modified Chebyshev kernels
	0	96.689	96.689		0.810	92.905	92.095		**4.762**	98.095	93.333	2	
	0.662	94.702	94.040		0.810	92.524	91.714		3.810	99.048	95.238	3	
	1.325	95.364	94.040		1.524	92.762	91.238		1.905	96.190	94.286	4	
	70.861	88.742	17.881		2.667	92.667	90.000		2.857	96.190	93.333	5	
	4.636	90.066	85.430		73.619	92.000	18.381		3.810	96.190	92.381	6	
	3.974	88.742	84.768		**76.857**	91.857	15.000		**-2.857**	91.429	94.286	7	
	4.636	90.066	85.430		75.143	92.000	16.857		2.857	92.381	89.524	8	
	2.649	87.417	84.768		76.810	91.429	14.619		0	94.286	94.286	9	
	3.311	88.742	85.430		75.333	91.476	16.143		0	90.476	90.476	10	
10.861 (28.89)	3.974	98.013	94.040	44.195 (37.492)	0.714	92.905	92.190	14.762 (22.278)	31.429	97.143	65.714	1	Stage 2 Hermite & Stage 3 Composite Hermite kernels
	0	97.351	97.351		0.762	92.857	92.095		3.810	98.095	94.286	2	
	-0.662	94.702	95.364		1.000	92.667	91.667		2.857	97.143	94.286	3	
	7.947	93.377	85.430		**0.333**	92.429	92.095		**0.952**	94.286	93.333	4	
	67.550	85.430	17.881		**76.238**	91.619	15.381		6.667	99.048	92.381	5	
	5.298	90.728	85.430		74.381	91.476	17.095		0.952	95.238	94.286	6	
	-16.556	68.212	84.768		75.238	89.857	14.619		10.476	96.190	85.714	7	
	-12.583	72.848	85.430		68.667	89.857	21.190		2.857	91.429	88.571	8	
	-5.960	78.808	84.768		71.143	88.190	17.048		15.238	91.429	76.190	9	
	59.603	77.483	17.881		73.476	88.571	15.095		**72.381**	96.190	23.810	10	
9.735 (21.804)	4.636	98.013	93.377	38.248 (39.363)	0.714	92.857	92.143	2.190 (2.42)	3.810	97.143	93.333	1	Stage 2 Legendre & Stage 3 Modified Legendre kernels
	0	96.689	96.689		0.905	93.000	92.095		4.762	98.095	93.333	2	
	0.662	94.702	94.040		1.286	92.857	91.571		1.905	97.143	95.238	3	
	0	95.364	95.364		**0.476**	92.524	92.048		2.857	97.143	94.286	4	
	71.523	89.404	17.881		1.190	92.000	90.810		2.857	96.190	93.333	5	
	5.298	90.728	85.430		73.810	92.238	18.429		**5.714**	94.286	88.571	6	
	4.636	89.404	84.768		**76.857**	91.857	15.000		0.952	94.286	93.333	7	
	4.636	90.066	85.430		75.095	91.952	16.857		0.952	94.286	93.333	8	
	2.649	87.417	84.768		76.810	91.476	14.667		**-2.857**	91.429	94.286	9	
	3.311	88.742	85.430		75.333	91.476	16.143		0.952	89.524	88.571	10	

		Stage 2 Chebyshev & Stage 3 Generalised Chebyshev kernels										Stage 2 Chebyshev & Stage 3 Modified Chebyshev kernels										Stage 2 Hermite & Stage 3 Composite Hermite kernels										Stage 2 Legendre & Stage 3 Modified Legendre kernels									
		Order no.										Order no.										Order no.										Order no.									
		1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
Thyroid	**N1**	95.478	95.449	95.420	95.420	95.012	95.012	95.683	2.392	2.159	2.363	95.478	95.449	95.420	95.420	95.012	95.012	95.683	2.392	2.159	2.363	95.478	95.216	95.449	94.749	95.303	95.362	95.274	92.707	92.707	92.707	95.478	95.449	95.391	94.866	95.070	95.012	95.537	2.3921	2.159	2.363
	N2	96.354	96.383	96.208	96.499	96.295	96.295	96.062	96.674	95.595	2.362	97.287	96.849	96.470	96.616	96.441	96.470	96.266	96.499	95.683	95.595	97.345	96.820	96.703	96.908	96.558	96.091	95.858	92.707	92.707	92.707	97.287	97.054	96.441	96.733	96.237	96.324	96.120	95.799	95.712	95.741
	I	0.875	0.933	0.788	1.079	1.284	1.284	0.379	**94.282**	93.436	**0**	1.809	1.400	1.050	1.196	1.429	1.459	**0.583**	**94.107**	93.524	93.232	1.867	1.604	1.254	**2.159**	1.254	0.729	0.583	**0**	**0**	**0**	1.809	1.604	1.050	1.867	1.167	1.313	**0.583**	93.407	**93.553**	93.378
	AI	**19.434 (39.228)**										**28.979 (44.609)**										**0.945 (0.803)**										**28.973 (44.492)**									
AAI		**14.571 (27.807)**										**20.479 (30.347)**										**19.672 (26.686)**										**21.910 (29.466)**									
TAI		**19.158 (28.567)**																																							

N1 Classification accuracy of Stage 2 kernels
N2 Classification accuracy of Stage 3 kernels
I Improvement = N2-N1
AI Average Improvement for the first 10 orders
AAI Average of Average Improvements of the first 10 orders over all the datasets under investigation
TAI Total Average Improvement over all the datasets and all the kernels under investigation for the first 10 orders

Furthermore, one of the remarkable implications of the fusion by multiplication process is that it makes the Stage 3 kernels substantially rectify the abrupt oscillating changes in the classification accuracy amongst the polynomial orders which both of the the Stage 1 and Stage 2 kernels suffered from. This can be clearly observed in the graphical representations of the experimental results illustrated in Figures 4.19–4.24. As shown, the Stage 3 kernels have exhibited a lot more stable and consistent classification performance throughout the spectrum of polynomial orders, compared to their Stage 2 counterparts, for all the datasets investigated.

To quantify this observation, Table 4.3 has also been used to calculate the standard deviation of the average improvement factors (shown in brackets), as well as the minimum and maximum improvement factors (shown in bold), introduced by the fusion by multiplication process taking place in Stage 3 kernels. As it can be noticed, due to the difference between the two sets of results, which can be huge at times due to the oscillating nature of the Stage 2 kernels, the standard deviation of the average improvement factors can therefore be quite sparse and even higher than the mean, as shown for example in the Breast Cancer and Ionosphere datasets. Conversely, however, one can also realize the consistency of the results by examining the standard deviation of the accuracy values themselves, reported in the statistical analysis tests tabulated in Appendix B.2, where the standard deviation of the classification accuracies scored by Stage 3 kernels is consistently smaller than their Stage 2 counterparts, for all the datasets experimented.

Such a significant improvement in the classification accuracy scored by the Stage 3 kernels over their Stage 2 counterparts clearly shows that using Stage 2 kernels on their own is not a recommended approach as they do not produce a consistently reliable and robust enough performance that can compete with existing kernels in solving real-world classification problems. To this end, one would therefore be interested to compare the performance of the composite Stage 3 kernels with some of the existing traditional kernels to explore how they compare and whether or not they can actually compete with them. This is investigated in more detail in the next sub-section.

4.4.6 Comparative experimental evaluation for the composite Stage 3 kernels and some traditional SVM kernels

Being combined with Stage 2 kernels of the similarity fusion framework, Stage 3 kernels therefore represent the final combined kernel formulated using both the summative and multiplicative similarity fusion operations. This section presents a comparison between the

classification performance obtained from these composite Stage 3 kernels and some of the traditional SVM kernels that have been in common use, to explore if they can actually compete with them. The following traditional SVM kernels have therefore been selected to conduct a further set of experiments to compare their performance with those yielded by the Stage 3 kernels:

- Linear kernel

$$k(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle \tag{4.18}$$

- Homogeneous polynomial kernel

$$k(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle^n \tag{4.19}$$

- Inhomogeneous polynomial kernel

$$k(\mathbf{x}, \mathbf{z}) = (\langle \mathbf{x}, \mathbf{z} \rangle + 1)^n \tag{4.20}$$

- Gaussian kernel

$$k(\mathbf{x}, \mathbf{z}) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{z}\|^2\right) \tag{4.21}$$

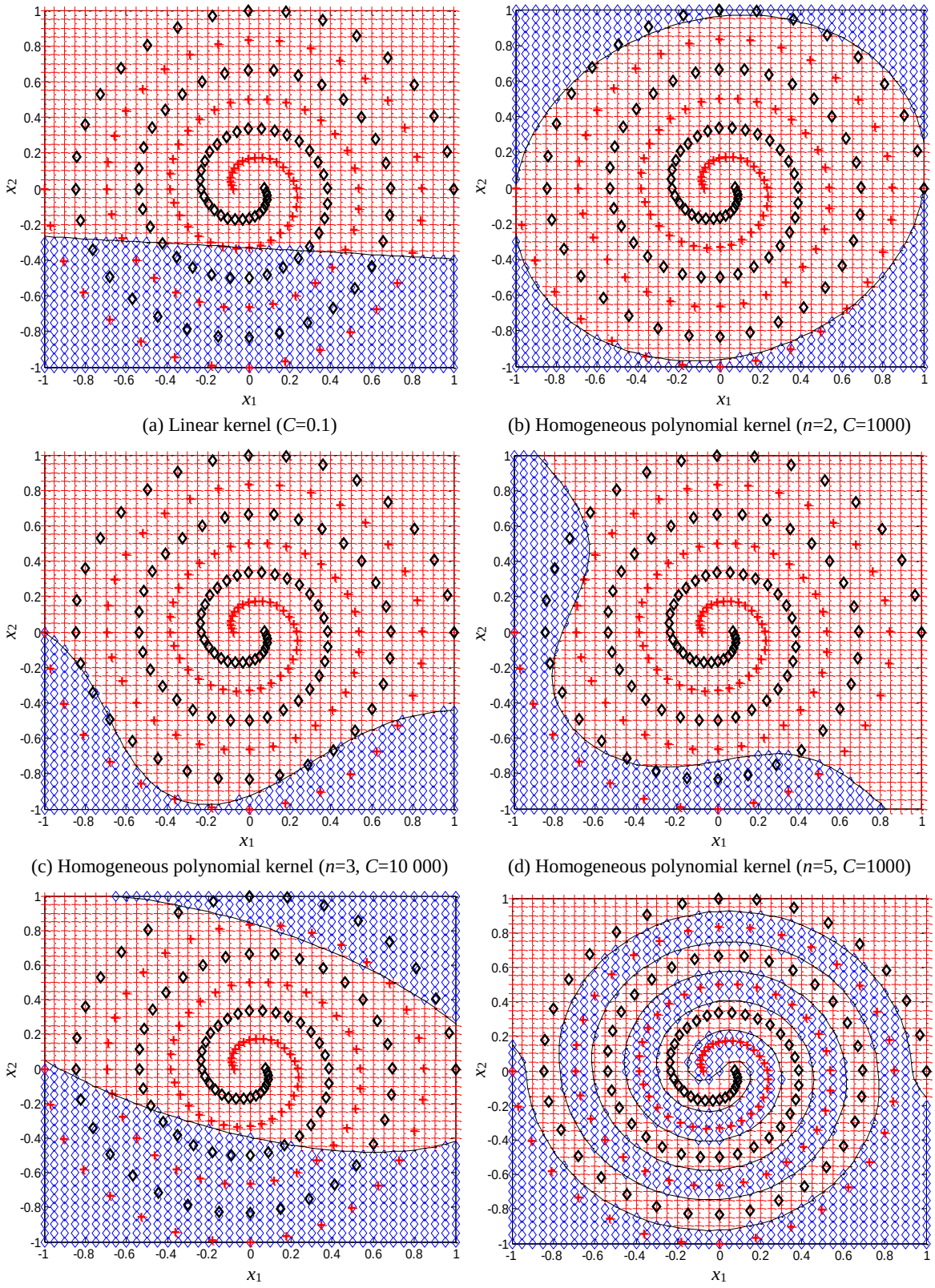
The first set of experiments designed for this purpose utilized the 2D characteristics of the Two Spirals dataset to visualize and assess the generalization capabilities and classification performance of the kernels under test in picking up its highly non-linear boundaries. Figure 4.26 shows the best results obtained by the traditional kernels selected in (4.18 – 4.21), whereas Figure 4.27 shows the best results obtained by the Stage 3 composite kernels, together with their model/kernel parameters.

As demonstrated in Figure 4.26, the linear kernel and both traditional SVM polynomial kernels have failed to capture the highly non-linear boundaries of the Two Spirals dataset. The Gaussian kernel, however, demonstrated a robust performance in equally splitting the two classes from each other, although without any attempt to follow the circular path of the spirals when they terminate at the edges.

Stage 3 kernels (shown in Figure 4.27), on the other hand, outperform the linear and traditional polynomial kernels in picking up the spiral shape of this dataset. Some have also demonstrated a competitive performance compared to the Gaussian kernel with the added advantage of being able to generalize outside the data region (as shown for example with the Stage 3 Composite Hermite kernel in Figure 4.27 (d)). Another interesting observation that can also be drawn from Figure 4.27 is the fact that the Modified Chebyshev kernel (Figure 4.27 (b)) demonstrated superior performance over the Generalised Chebyshev kernel (Figure 4.27 (a)). Similarly, the Stage 3 Modified Legendre

kernel (Figure 4.27 (f)) also demonstrated superior performance over the Stage 2 Legendre kernel (Figure 4.27 (e)). Such observation again supports the earlier discussion that the corresponding weighting function is not necessarily the best kernel to be fused with the Stage 2 kernels, and multiplying by a better kernel can indeed enhance the generalisation capabilities of the overall kernel and improve the classification performance even further.

Table 4.4 also shows the maximum classification accuracy scored by each of the kernels under comparison, using the same accuracy estimation methods used in the experiments conducted in the previous section for each dataset. The highest accuracy obtained for each dataset amongst all the kernels under test are those shown in bold. Inline with the previous research on the construction of SVM kernels from orthogonal polynomials, such as [17, 34, 35, 102], the numerical results illustrated in Table 4.4 show that the accuracies scored by Stage 3 kernels do indeed demonstrate that they can offer performance which is competitive, if not superior, to those offered by traditional SVM kernels that have been in common use. It must be noted, however, that this superior performance can only be achieved when these Stage 3 kernels are well prepared by synergistically combining (by multiplication) their summative polynomial element (i.e., Stage 2 kernels) with another robust kernel, which does not necessarily have to be their corresponding weighting function.



(a) Linear kernel (C=0.1)

(b) Homogeneous polynomial kernel (n=2, C=1000)

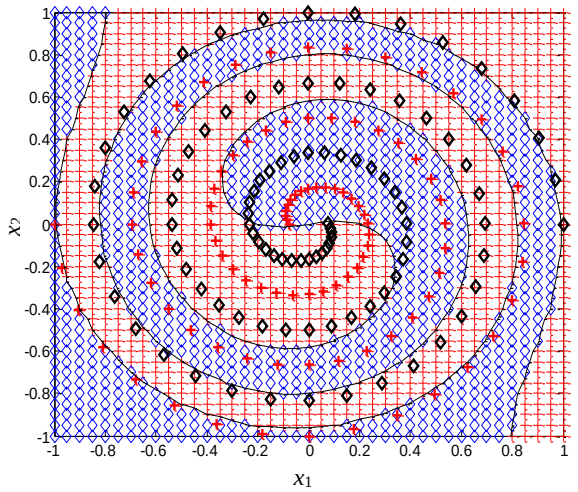
(c) Homogeneous polynomial kernel (n=3, C=10 000)

(d) Homogeneous polynomial kernel (n=5, C=1000)

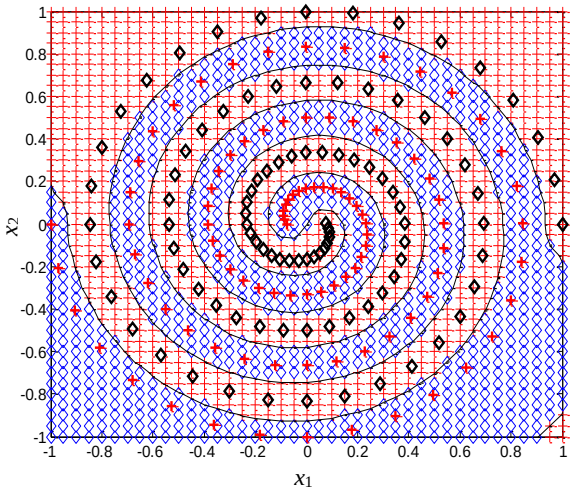
(e) Inhomogeneous kernel (n=5, C=1000)

(f) Gaussian kernel (γ=50, C=100)

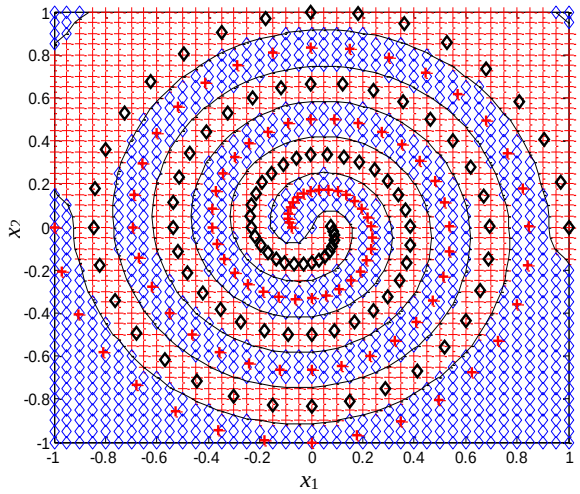
Figure 4.26 Visual assessment of the classification performance of some traditional SVM kernels using the Two Spirals dataset.



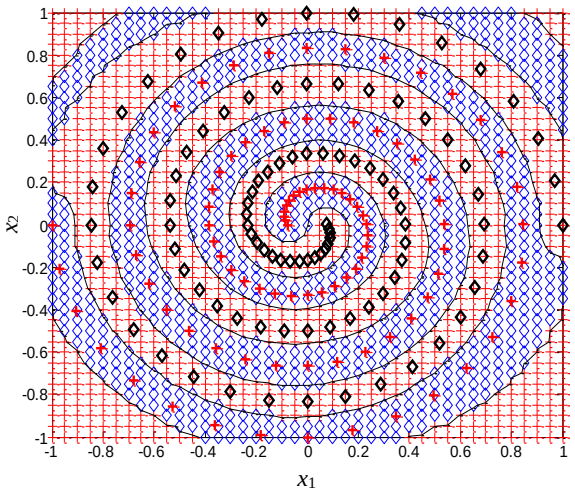
(a) Stage 3 Generalized Chebyshev kernel (n=8, C=100)



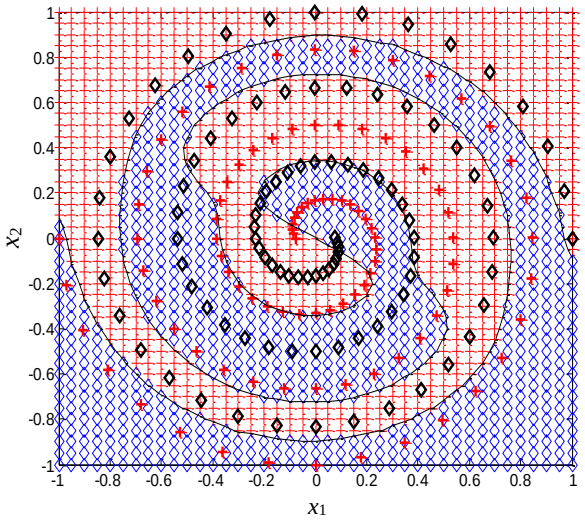
(b) Stage 3 Modified Chebyshev kernel (n=4, C=100, γ=100)



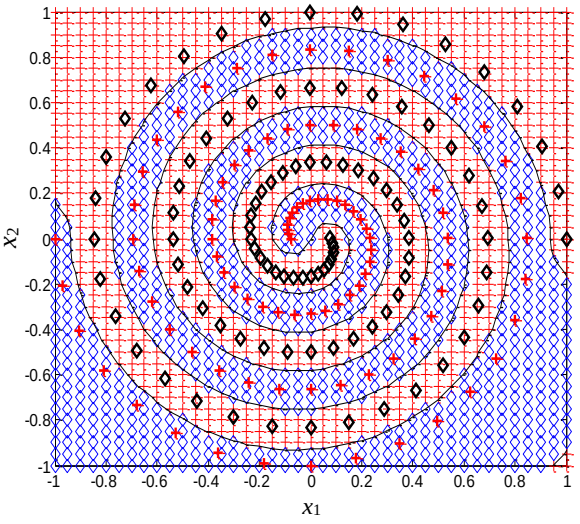
(c) Stage 3 Composite Hermite kernel (n=3, C=100, γ=100)



(d) Stage 3 Composite Hermite kernel (n=17, C=100, γ=180)



(e) Stage 2 Legendre kernel (n=7, C=100)



(f) Stage 3 Modified Legendre kernel (n=5, C=100, γ=100)

Figure 4.27 Visual assessment of the classification performance of Stage 3 kernels of the developed similarity fusion framework using the Two Spirals dataset.

Table 4.4 Comparative results of the classification accuracy scored by Stage 3 kernels of the similarity fusion framework and some of the most commonly used traditional SVM kernels.

	Two Spirals	Breast Cancer	Iris	Image Segmentation	Ionosphere	Thyroid
	Accuracy (%) & parameter (s)	**Accuracy (%) & parameter (s)**	**Accuracy (%) & parameter (s)**	**Accuracy (%) & parameter (s)**	**Accuracy (%) & parameter (s)**	**Accuracy (%) & parameter (s)**
Homogeneous polynomial	57.41 n=1	96.84 n=4	87.62 n=3	92.48 n=2	96.03 n=2	97.72 n=3
Inhomogeneous polynomial	56.27 n=3	97.01 n=4	97.14 n=4	92.86 n=5	97.35 n=2	**97.78** n=2
Gaussian	90.62 γ=10	98.29 γ=0.01	98.10 γ=3	**93.33** γ=0.5	**98.01** γ=0.1	97.11 γ=0.5
Stage 3 Generalized Chebyshev	98.42 n=11	98.29 n=2	97.14 n=1	92.62 n=3	96.03 n=1	96.67 n=8
Stage 3 Modified Chebyshev	96.84 n=6, γ=10	**98.72** n=1, γ=0.01	**99.05** n=3, γ=2	92.90 n=2, γ=1	**98.01** n=1, γ=0.1	97.29 n=1, γ=0.1
Stage 3 Composite Hermite	94.83 n=12, γ=100	98.24 n=2, γ=0.1	**99.05** n=5, γ=2	92.90 n=1, γ=0.1	**98.01** n=1, γ=0.01	97.35 n=1, γ=0.1
Stage 2 Legendre	98.42 n=13	98.24 n=2	95.24 n=3	92.14 n=1	96.69 n=2	95.54 n=7
Stage 3 Modified Legendre	**100** n=9, γ=10	98.42 n=1, γ=0.1	98.1 n=2, γ=1	93.00 n=2, γ=0.01	**98.01** n=1, γ=0.1	97.29 n=1, γ=0.1

4.4.7 Validation of the experimental setup via a comparison with previously reported results

To check the validity of the experimental results presented in this chapter, as well as those results that will also be presented in the next chapters, it is also important to demonstrate that the experimental setup being used in the experiments conducted in this thesis is correct, valid, and is also consistent with what other researchers have used, specifically those, who are used to compare results with (i.e., Ozer et al. [17]). To achieve this purpose, this section is dedicated to demonstrate that the experimental setup used in this thesis was able to produce the same results as those reported by Ozer et al. under the same experimental parameters (e.g., dataset used, train/test ratio, kernel function, Matlab toolbox, SVM model parameters, normalization process, etc.). Needless to mention that if one wishes to explore how the classification performance is affected when different kernels are used, it is important to keep all other experimental parameters unchanged, which is what most of the work presented in this thesis is all about. Note that exactly the same kernel plots were produced as the ones reported by Ozer et al. (as per Figure 3.5 in Section 3.4.2.1), which demonstrates the correctness of the mathematical formulation of the kernels that have been implemented in the proposed experimental setup.

4.4.7.1 Comparative experimental results on the Two Spirals dataset

Ozer et al. did not mention from where they obtained the Two Spirals dataset that they used in their experiments [17], whereas the one used in the experiments conducted in this thesis is the standard dataset obtained from [126]. Although they look similar, the examples in the dataset used in this thesis are more interwined over each other than the dataset used by Ozer et al. (especially the data points located close to the centre of the vector space). This renders the classification of the dataset used here to be more challenging than the classification of the dataset used by Ozer et al., as its class boundaries are more interwined over each other, and hence poses a more challenging non-linearly separable classification task. Despite this challenge, the experimental setup used in this thesis was still able to produce very close performance to the results reported by Ozer et al. when the same kernel functions are used, as shown below.

Figure 4.28 shows a comparison between the classification performance of the Modified Chebyshev kernel (as per Eq. (4.14)) produced by the experimental setup used in this thesis with its counterpart reported by Ozer et al. As can be observed, both results are nearly identical at the same polynomial order, which demonstrates that the experimental setup used in this thesis is correct, valid, and is consistent with what Ozer et al. have also used, which is important for a fair comparison. Analysing closely both results, one can also notice that the generalization performance of the proposed experimental setup is better than Ozer et al.'s because the original data points are shown to be located exactly in the middle of the decision region of each class (as shown in Figure 4.28 (b)), which is what one would wish to visualize from a maximum margin classifier like the SVM; whereas in the result reported by Ozer et al. (shown in Figure 4.28 (a)), most of the data points appear to be on the decision boundary itself, which does not demonstrate a very good generalization performance.

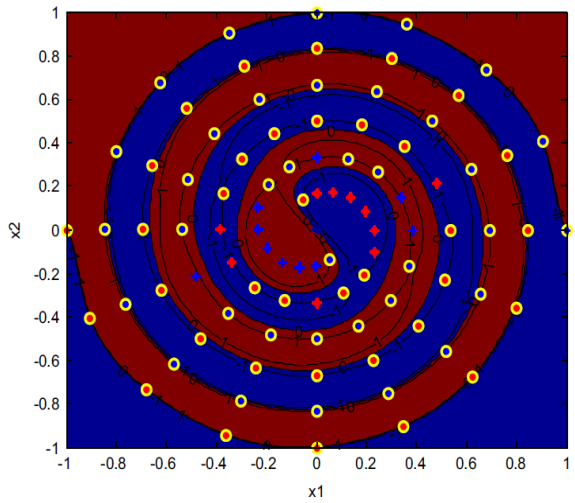
The same observation can also be noticed from the experimental results obtained when the Gaussian kernel (defined in (4.21)) is used, as shown in Figure 4.29, which, although also demonstrate that both results are close to each other, the result produced from the proposed experimental setup shows a better generalization ability. This difference in performance, however, might be due to a poor grid search conducted by Ozer et al. over the hyperparameter C (which they did not mention what value it was set to in their experiments) and the kernel parameter γ. They mentioned that they used γ=1, but they did not mention that this was the value that produced the best results, so may be if they searched along the range of possible values (rather than using just a single value), they might have been able to produce better results than the ones reported here. In the

experiments conducted in this thesis, however, a comprehensive grid search over these parameters was carried out and the parameters that produced the best results are reported, as shown in Figures 4.28 and 4.29.

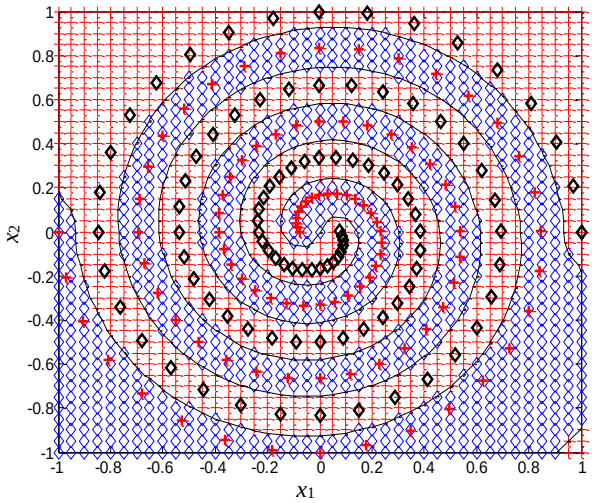
Figure 4.30 also shows a comparison of the results obtained on the Generalized Chebyshev kernel as defined in (4.13). When conducting the experiments, Ozer et al. mentioned that they modified this kernel by adding a small value ε to its denominator to eliminate the possibility of dividing by zero. This means that the Generalized Chebyshev kernel used in their experiments is formulated in the form of:

$$k(\mathbf{x},\mathbf{z}) = \frac{\sum_{i=0}^{n} T_i(\mathbf{x}) T_i^T(\mathbf{z})}{\varepsilon + \sqrt{m - < \mathbf{x},\mathbf{z} >}} . \tag{4.22}$$

However, there is no mention as to what value they decided to use for ε, so that one can try and reproduce their results. In all the experiments conducted in this thesis, the Generalized Chebyshev kernel has been implemented (whenever used) as defined in (4.13) without being modified. Together with the more challenging non-linear spiral flow of the version of the dataset used in this thesis (and again the absence of information about the penalization parameter *C*), perhaps explains the reason why the two results shown in Figure 4.30 (a) and (b) are not exactly identical to each other as in the case of the Gaussian and the Modified Chebyshev kernels.

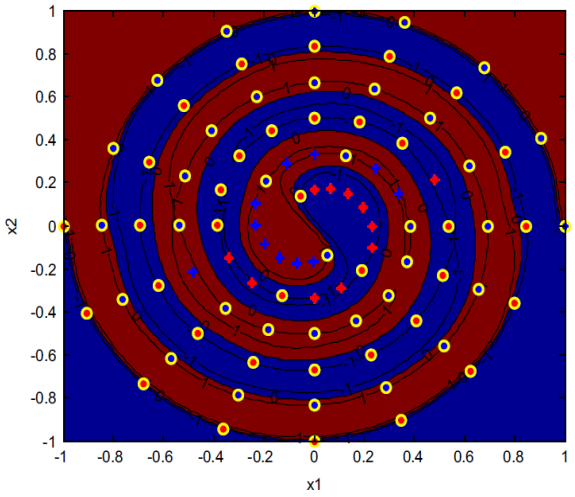


(a) Classification performance of the Modified Chebyshev kernel (n=4), as reported by Ozer et al. [17].

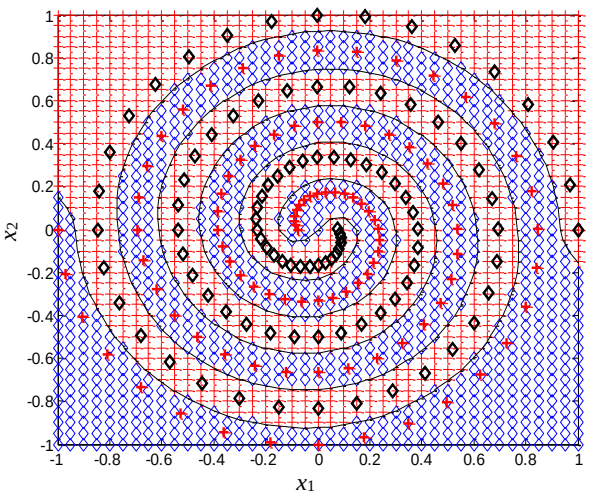


(b) Classification performance of the Modified Chebyshev kernel (n=4, C=100, γ=100) produced by the experimental setup used in this thesis.

Figure 4.28 Comparison of the classification performance achieved using the Modified Chebyshev kernel.

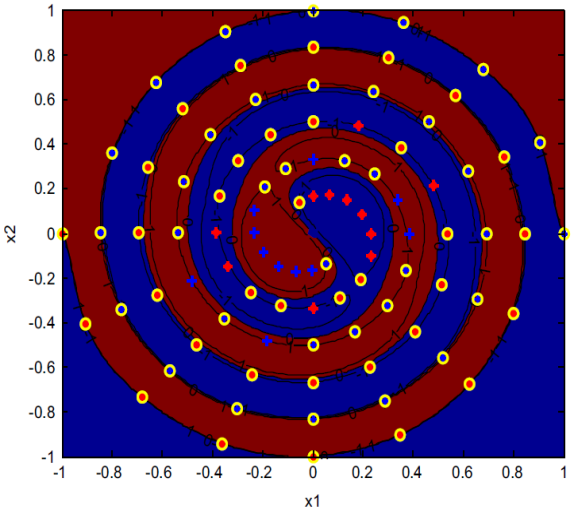


(a) Classification performance of the Gaussian kernel (γ=4.59), as reported by Ozer et al. [17].

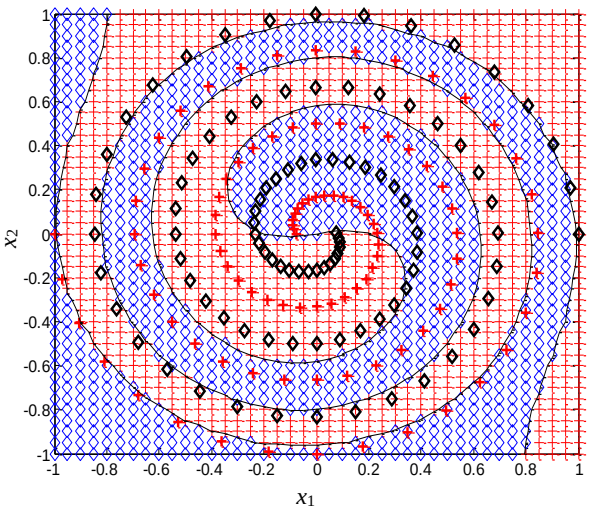


(b) Classification performance of the Gaussian kernel (γ=50, C=100) produced by the experimental setup used in this thesis.

Figure 4.29 Comparison of the classification performance achieved using the Gaussian kernel.



(a) Classification performance of the Generalized Chebyshev kernel (n=4), as reported by Ozer et al. [17].



(b) Classification performance of the Generalized Chebyshev kernel (n=4, C=100) produced by the experimental setup used in this thesis.

Figure 4.30 Comparison of the classification performance achieved using the Generalized Chebyshev kernel.

4.4.7.2 Comparative experimental results on the Breast Cancer dataset

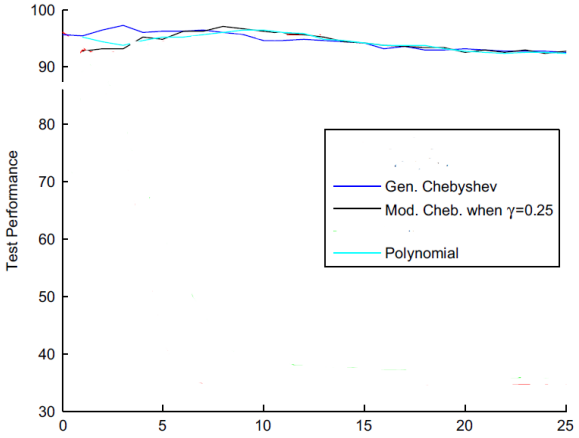
Whilst focusing on comparing the experimental results obtained in terms of the test accuracy as per the reported train/test ratio, one can notice that Ozer et al. presented their results on this dataset in two formats [17]: (1) in terms of the maximum test accuracy scored by each of the experimented kernels, and (2) in a graphical format to show how the classification accuracy of the different kernels is affected with the variation in the polynomial order. Both formats are compared with the results obtained from the experimental setup used in this thesis in Table 4.5 and Figure 4.31.

Table 4.5 shows the comparative results scored by the Gaussian, Modified Chebyshev, Generalized Chebyshev, and the polynomial kernels on the Breast Cancer dataset. As can be observed, when the experimental setup is configured to use the same reported kernel parameters, the produced test accuracies are nearly the same as those reported by Ozer et al. Interestingly, all the kernels have also produced exactly the same number of support vectors, which further demonstrates the consistency of the experimental setup used in this thesis. Moreover, with a more comprehensive grid search on the kernel and penalization parameter, the experimental setup proposed here was able to produce even better test accuracies than those reported by Ozer et al., again with exactly the same reported number of support vectors.

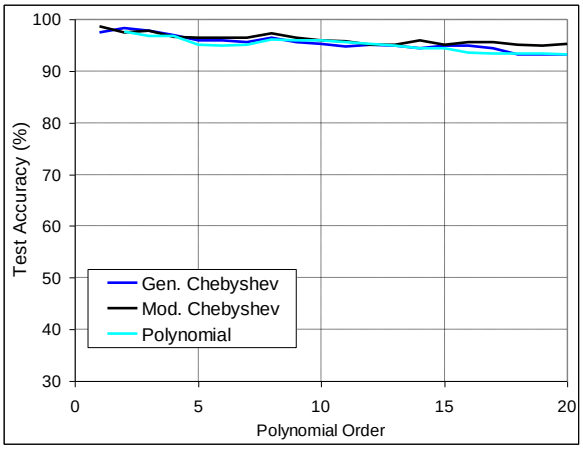
Figure 4.31 also shows that the results obtained by Ozer et al. for the first few orders of the Generalized Chebyshev, Modified Chebyshev, and polynomial kernels (shown in Figure 4.31 (a)) are very close to those produced by the experimental setup used in this thesis for the same kernels (shown in Figure 4.31 (b)). Note that the figure reported by Ozer et al. in Figure 4.31 (a) was edited here to remove the graphical results of the kernels that are irrelevant to this comparison, for easier analysis.

Table 4.5 Comparison between the maximum classification accuracy scored using the Gaussian, Modified Chebyshev, Generalized Chebyshev, and the polynomial kernels.

		Ozer et al. reported results	Experimental results with same kernel parameters	Best scored experimental results
Gaussian kernel	Test %	95.52	97.65	98.29
	γ	0.0034722	0.0034722	0.01
	C		1000	1000
	SV no.	30	30	30
Modified Chebyshev kernel	Test %	97.01	97.23	98.72
	n	8	8	1
	γ	0.25	0.25	0.01
	C		10	10
	SV no.	30	30	30
Generalized Chebyshev kernel	Test %	97.23	97.87	98.29
	n	3	3	2
	C		0.01	1
	SV no.	30	30	30
Polynomial kernel	Test %	96.38	95.95	97.65
	n	9	9	2
	C		1	0.01
	SV no.	30	30	33



(a) Classification accuracy of the Generalized Chebyshev, Modified Chebyshev, and polynomial kernels at different orders, as reported by Ozer et al. [17].



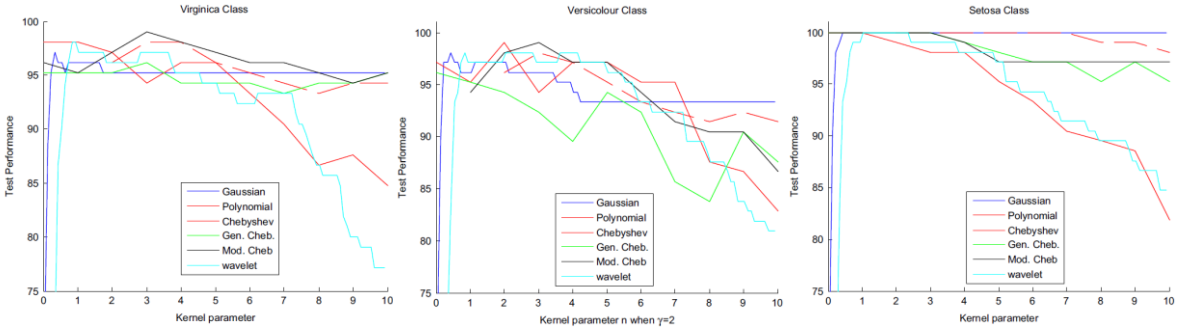
(b) Classification accuracy of the Generalized Chebyshev, Modified Chebyshev, and polynomial kernels at different orders produced by the experimental setup used in this thesis.

Figure 4.31 Graphical comparison of the classification accuracy obtained using the Breast Cancer dataset.

4.4.7.3 Comparative experimental results on the multi-class datasets

The other datasets used by Ozer et al. are the Iris and Image Segmentation multi-class datasets [17]. In their reported results, however, it is not clear why they preferred to report the scored test accuracy of successfully classifying each class individually rather than reporting the test accuracy of all the correctly classified data points in the whole test dataset for all the classes, as they did, for example, with the Breast Cancer dataset. Also,

the reported graphical results (which show how the test accuracy varies with the kernel parameter) are not consistent between the Iris and Image Segmentation dataset, which makes the comparison even harder. For example, in the Iris dataset, they decided to present the results of all the kernels on each individual class on a separate plot (as shown in Figure 4.32), whereas in the Image Segmentation dataset, they did the complete opposite; i.e., they plotted the results of all the classes on each individual kernel on a separate plot (as shown in Figure 4.33). In most of the plots, it is also difficult to differentiate between the different graphs due to the poor selection of the colours/legends.

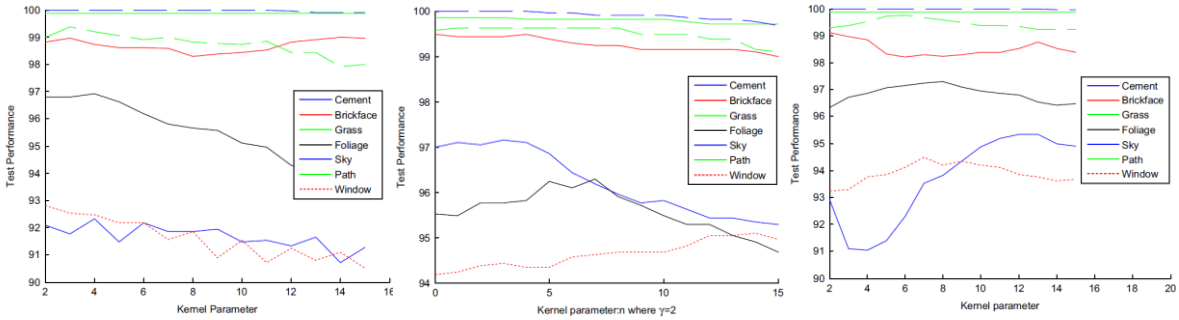


(a) Kernel parameter vs. test performance for Virginica Class

(b) Kernel parameter vs. test performance for Versicolour Class

(c) Kernel parameter vs. test performance for Setosa Class

Figure 4.32 Classification accuracy on the Iris dataset, as reported by Ozer et al. [17].



(a) Generalized Chebyshev kernel parameter vs. performance

(b) Modified Chebyshev kernel parameter vs. performance

(c) Polynomial kernel parameter vs. performance

Figure 4.33 Classification accuracy on the image segmentation dataset, as reported by Ozer et al. [17].

Due to this confusion, this section will compare the best scored test accuracy calculated from all the correctly classified examples in the whole test dataset produced from the experimental setup used in this thesis, with the 'average' test accuracy over the individual classes reported by Ozer et al., as illustrated in Table 4.6. Although this comparison is conducted between the results obtained from two different experimental setups, especially because Ozer et al. seem to have reported different kernel parameters for each class (which is another unexplained behaviour), the results shown in Table 4.6 illustrate that they are reasonably close to each other, and hence demonstrates the effectiveness of the utilized kernels in both experimental setups.

Table 4.6 Comparing the experimental results on the Iris and image segmentation multi-class datasets.

	Iris		**Image segmentation**	
	Total Average Test Accuracy reported by Ozer et al. (%)	**Best scored experimental results (%)**	**Total Average Test Accuracy reported by Ozer et al. (%)**	**Best scored experimental results (%)**
Gaussian	98.41	98.10	98.19	93.33
Generalized Chebyshev	97.46	97.14	97.24	92.62
Modified Chebyshev	99.37	99.05	98.36	92.90
Polynomial	98.73	97.14	97.97	92.86

4.5 Summary

This chapter proposed a novel similarity fusion framework within which the pictorial similarity-based characteristics of the adopted orthogonal polynomial kernels are analysed and their corresponding empirical classification performance is assessed. It has been shown that such polynomial kernels are naturally constructed from a mixture of summative and multiplicative combination of base kernel building blocks that synergistically contribute towards calculating more accurate similarity measures, as exhibited by their pictorial characteristics. Hence, the resulting performance of the classifier constructed from the fused kernels is expected to outperform that of the best individual kernel building block.

To validate this hypothesis, the proposed similarity fusion framework was therefore set to analyse and assess the performance of these polynomial kernels at three hierarchical stages. The first stage was defined as the dot-product between the evaluation of the employed n^{th} order polynomial on the kernel's two input vectors, whereas the second stage implemented the fusion by summation operation where all the n^{th} order Stage 1 kernels are summed up from order 0 to n. Using the Chebyshev, Legendre, and Hermite polynomials, the experimental results presented in this chapter revealed that such a summative fusion operation is indeed synergistic and that it can introduce an average classification improvement factor that can be as high as 17.347%, depending on the datasets utilized and the polynomials employed.

The third stage of the similarity fusion framework, on the other hand, combined by multiplication the Stage 2 kernels with another kernel which can either be their corresponding weighting function or any other valid kernel. Experimental results on four different Stage 3 kernels, defined in (4.13 – 4.16), have also revealed that such a multiplicative fusion operation is also synergistic and that it can introduce a further average classification improvement factor of 19.158%, depending on the datasets utilized and the polynomials employed. The experimental results have also shown that the OPKs' corresponding weighting functions are not necessarily the best choice to be combined by multiplication with the Stage 2 kernels, and that combining them with more robust kernels can indeed yield even better results.

Finally, it has also been shown in this chapter that the Stage 3 kernels can be very competitive alternatives (if not even superior) to some of the traditional SVM kernels that have been in common use, such as the linear, homogeneous and inhomogeneous polynomial kernels, and the Gaussian kernel. It must not be forgotten, however, that these Stage 3 kernels have demonstrated such a competitive performance even though the Stage 2 kernels that they are combined with are still suffering from the problem (alluded to in the previous chapter) of decreasing their monotonic decay window as the order increases. One would therefore expect that if this problem is properly addressed, such kernels can demonstrate even more superior performance. This is the subject of investigation of the next chapter.

Chapter 5
Solving the Monotonic Decay Window Problem of Stage 2 Polynomial Kernels

5.1 Introduction

This chapter addresses the problem alluded to in Section 3.4.2, where the Stage 2 polynomial kernels exhibit a monotonic decay shape behaviour over a narrower range than the normalized data region of [-1,+1]. The aim is to demonstrate that when the kernel exhibits a complete monotonic decay behaviour, throughout the whole vector space where the data is located, the classifier will perform better than if the kernel exhibits this behaviour within only a partial region within the standard normalized vector space, and then oscillates afterwards. This is because similarity measurements calculated between any two data points that happen to be located outside this monotonic threshold window would then be inaccurate and could result in the SVM misclassifying the input data. Stage 2 kernels are therefore a typical example that can be used to conduct this investigation, due to their shape characteristics that have shown to be suffering from this drawback, rather than, for example the Stage 3 kernels which can avoid this problem via the appropriate selection of the kernel parameter γ during the fusion by multiplication process, as exhibited by their shape characteristics illustrated in Section 4.3.3.

To solve this problem, this chapter proposes a simple adaptive data normalization approach to confine the data to the regions where the kernel demonstrates the sought after ideal monotonic characteristics. This way, the possibility of any data point to be located outside the monotonic threshold window of the employed kernel is eliminated, and therefore the underpinning kernel should be able to calculate more accurate similarity measures between its input arguments. Experimental results on a number of benchmark datasets validate the effectiveness of this approach in improving the resulting SVM classification accuracy.

Customizing a solution to this problem via this proposed adaptive data normalization approach demonstrates that the study of kernels as similarity measure tools, rather than their standard definition as positive semi-definite functions, can enable the machine learning practitioner to use more tangible and intuitive properties (such as the shape characteristics of the ideal similarity function utilized in this chapter) to design more effective kernels than relying on only their implicitly constructed high-dimensional feature spaces that one might not even be able to calculate.

5.2 The monotonic decay window problem of orthogonal polynomial kernels

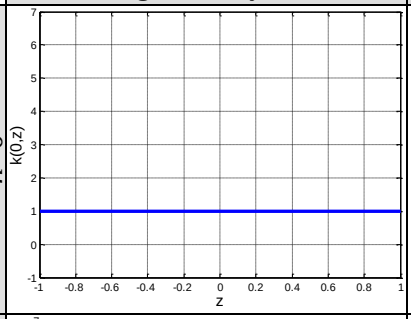
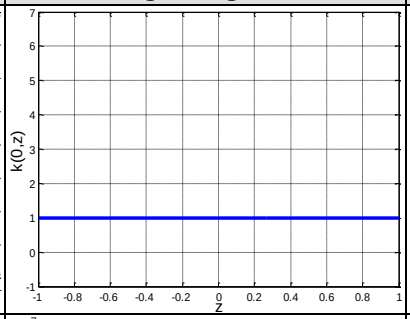
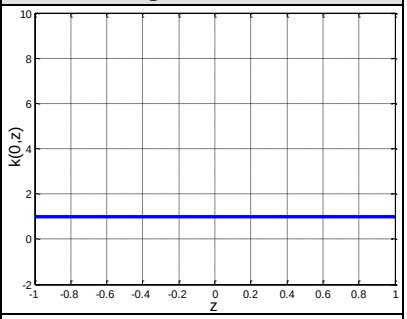
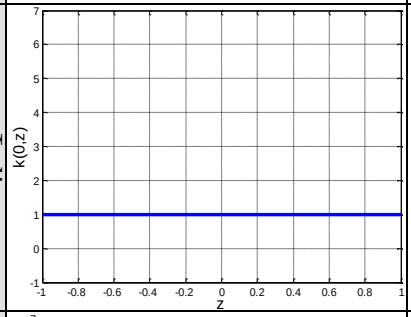
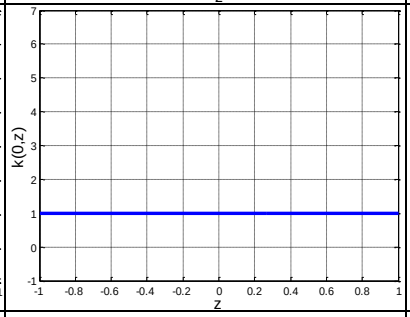
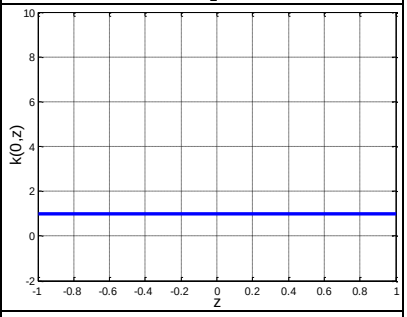
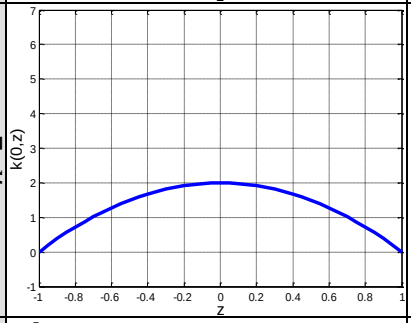
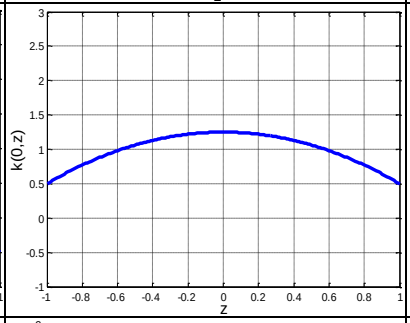
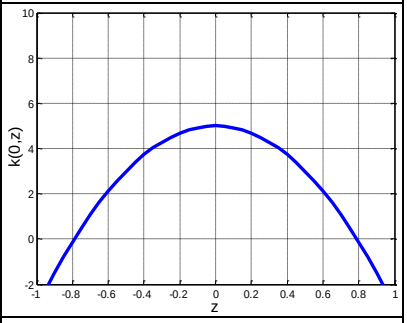
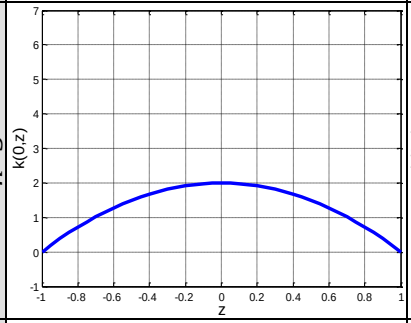
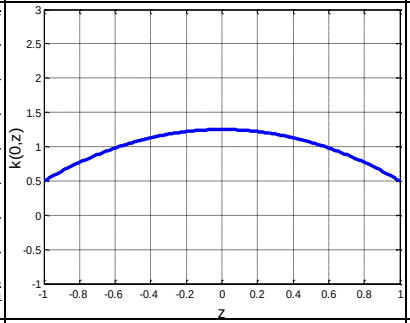
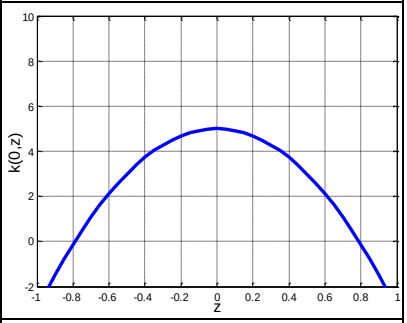
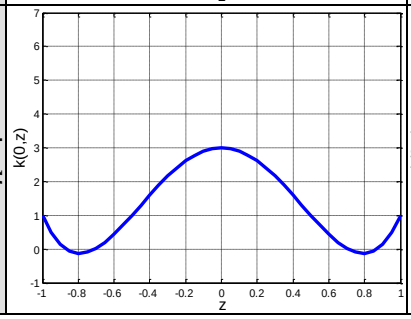
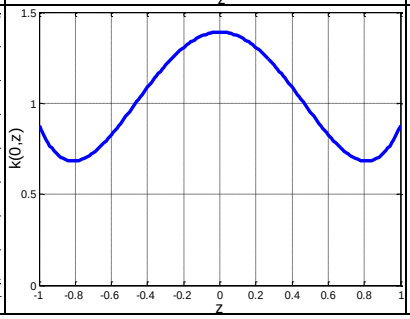
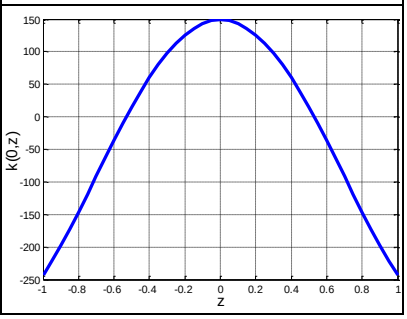
It has been explained in Chapter 3 how the study of kernels in terms of their similarity-based properties defined by their shape characteristics could indeed be a useful approach to design better kernels than relying on only their mathematical properties defined by their positive semi-definiteness property. It has also been explained (in Section 3.3.1) that if a kernel is regarded as a measure of similarity, it would be expected to return a high value when the two vectors are close to each other (i.e., highly similar and therefore belong to the same class) and maximized, when the two vectors are identical, and decreases monotonically as the two vectors depart away from each other (as typically demonstrated in Figure 3.1).

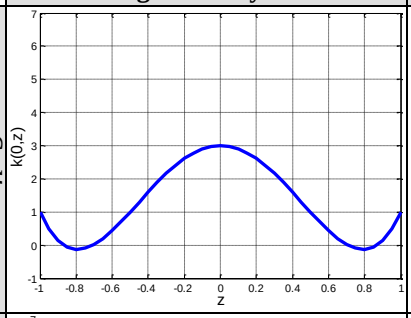
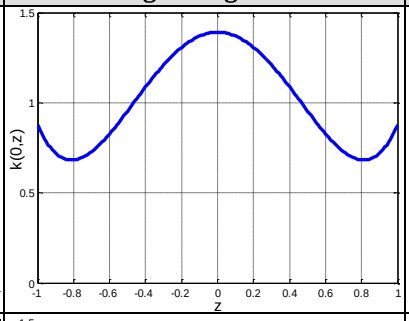
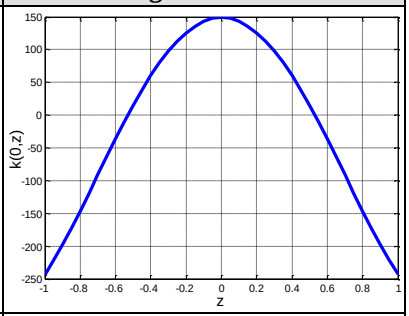
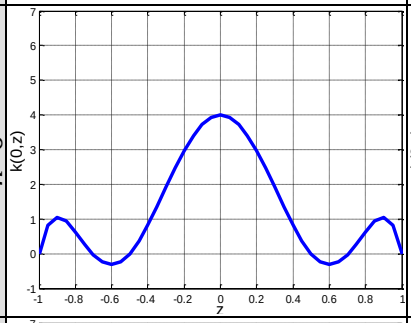
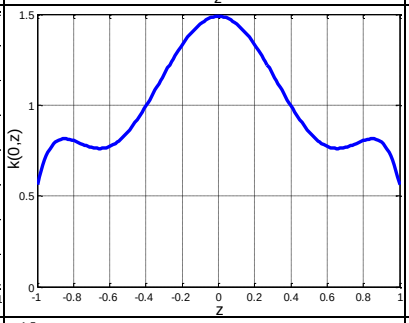
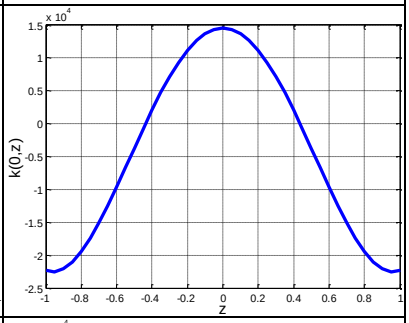
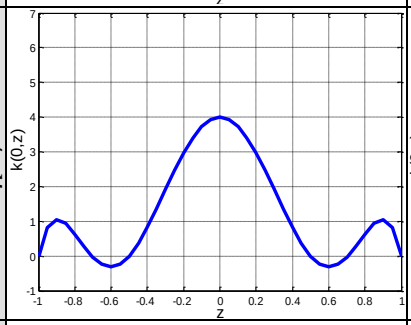
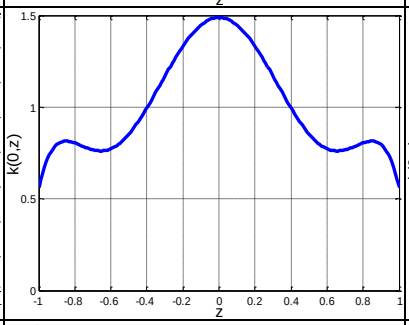
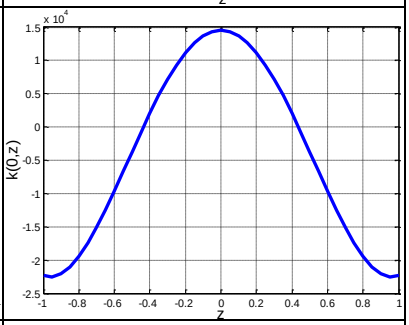
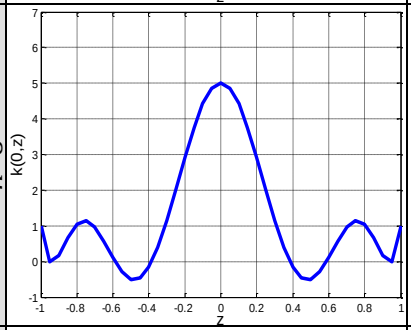
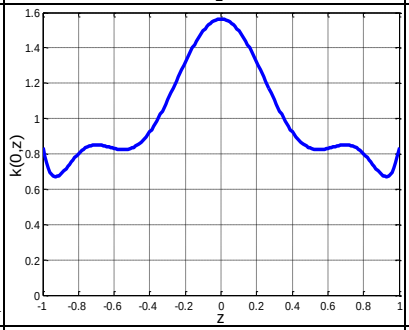
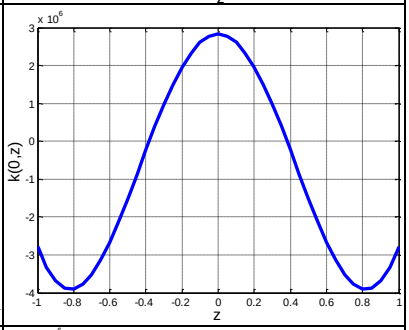
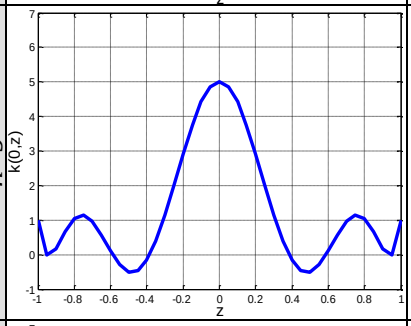
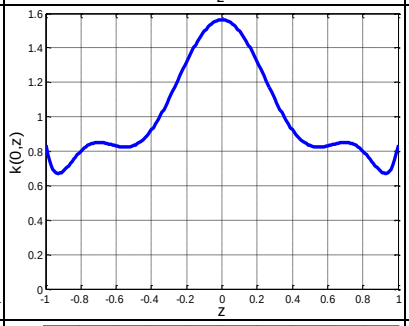
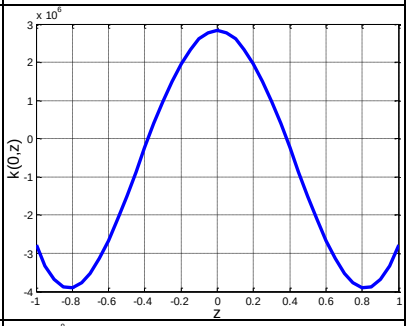
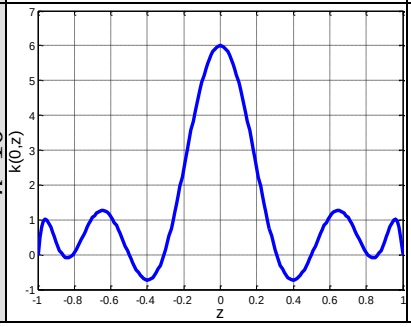
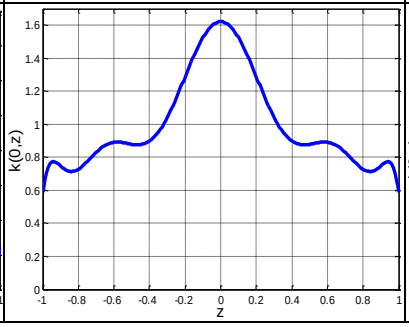
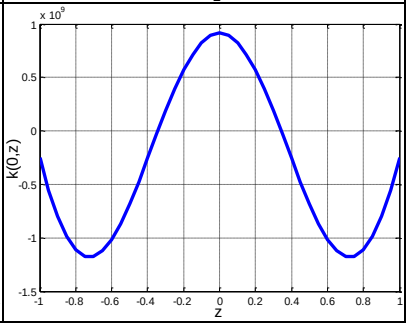
The shape illustrations of Stage 2 kernels of the similarity fusion framework developed in Chapter 4 have also revealed that they share some essential characteristics with this ideal similarity function. For example, as the order increases, the Stage 2 Chebyshev and Legendre kernels have shown to develop a dominant maximum peak wherever the two inputs happen to be identical, and then decrease in value as the two inputs depart away from each other. It has also been observed, however, that as the distance between the two inputs increases, the kernel value keeps on decreasing but only up to a certain threshold, after which it increases again and then fluctuates in a wavy pattern similar to the wave kernel. For ease of comparative assessment of such a problem, the graphs of the first 10 orders of each of the Stage 2 Chebyshev, Legendre, and Hermite polynomial kernels under investigation are plotted again individually in Table 5.1.

On re-examining the figures, one can notice that this is obviously not a complete monotonic decay behaviour throughout the whole normalized data region of [-1,+1] as it is the case with the ideal similarity function. Even worse, the window of monotonic decay (determined by the threshold) of each kernel has also been observed to shrink as the order increases. The trouble is that any measures of similarity calculated between any data points that happen to be located outside this monotonic decay window would therefore be inaccurate. To exacerbate this situation, one can notice that this window actually shrinks as the order increases, meaning that the chance of more data points falling outside this monotonic window will increase leaving more measures of similarity wrongly evaluated, a fact which can have a destructive effect on the resulting classification performance. Furthermore, the threshold of this monotonic window is different from one kernel to another at a given polynomial order, meaning that a solution devised to tackle this problem

for one kernel might need to be adapted before it can work on another kernel to suit its characteristics.

Table 5.1 Comparison of the first 10 orders of the Stage 2 polynomial kernels, at x=0.

	Stage 2 Chebyshev	Stage 2 Legendre	Stage 2 Hermite
n=0			
n=1			
n=2			
n=3			
n=4			

	Stage 2 Chebyshev	Stage 2 Legendre	Stage 2 Hermite
$n=5$			
$n=6$			
$n=7$			
$n=8$			
$n=9$			
$n=10$			

5.3 Customizing the solution via adaptive data normalization

5.3.1 Kernel shape modification versus data adaptation

To rectify this problem, one can think of modifying the shape of the kernel so that it gets closer to the shape of the ideal similarity function. Various operations can be implemented to achieve this purpose, for example, weighting, scaling, stretching etc. However, such operations could actually affect the positive semi-definiteness of the employed kernel (which is important to be kept intact), as well as changing the form of the kernel itself. Remember that it is also important in this context to keep the shape of the Stage 2 kernels intact so as to enable a fair comparison between the different types of polynomials (e.g., Chebyshev, Legendre, etc.) to explore and assess their robustness as stand-alone kernels in tackling classification problems. So, if every kernel is modified in a different way to match its shape to that of the ideal similarity function, comparing their resulting performances would be unfair.

For these reasons, this section proposes a simple alternative solution that actually adapts the data itself to the sought-after characteristics of the utilized kernel rather than adapting the kernel to match the shape characteristics of the ideal similarity function. Assuming that the shape of the polynomial kernel is not altered by the input data, this approach simply works by imposing a further data normalization step to confine the input data to the monotonic decay windows of the polynomial kernels where they demonstrate the ideal similarity measure characteristics. Recall that the data normalization region was initially selected to be [-1,+1], following the domain of common definition of the polynomial kernels considered in this thesis. To further confine the data to the ideal monotonic windows of the polynomial kernels, an additional data normalization step is introduced. By this way it is guaranteed that the measures of similarity calculated by the employed kernel are accurate and reflect the true level of similarity between the two input arguments.

5.3.2 Choosing a common monotonic window for data normalization

The problem in implementing this solution is that the thresholds of this monotonic window differ from one kernel to another and also from one order to another (as explained earlier, the thresholds decrease as the polynomial order increases). This would mean that the input data would need to be normalized to different scales for every type of polynomial kernel (i.e., Chebyshev, Legendre, etc.) and for every order of each kernel. For example, one can observe from the plots (at $x = 0$) of the first 10 orders of the Stage 2 kernels under

investigation illustrated in Table 5.1, that as the order increases, the thresholds decrease, and they can also be different from one kernel to another. Table 5.2 summarises these thresholds for the first 10 orders for each of these Stage 2 kernels under investigation.

Table 5.2 Stage 2 Chebyshev, Legendre, and Hermite kernels' monotonic windows.

Kernel / Order	Stage 2 Chebyshev	Stage 2 Legendre	Stage 2 Hermite
1	[-1,+1]	[-1,+1]	[-1,+1]
2	[-1,+1]	[-1,+1]	[-1,+1]
3	[-1,+1]	[-1,+1]	[-1,+1]
4	[-0.8,+0.8]	[-0.8,+0.8]	[-1,+1]
5	[-0.8,+0.8]	[-0.8,+0.8]	[-1,+1]
6	[-0.6,+0.6]	[-0.6,+0.6]	[-0.9,+0.9]
7	[-0.6,+0.6]	[-0.6,+0.6]	[-0.9,+0.9]
8	[-0.5,+0.5]	[-0.5,+0.5]	[-0.8,+0.8]
9	[-0.5,+0.5]	[-0.5,+0.5]	[-0.8,+0.8]
10	[-0.4,+0.4]	[-0.4,+0.4]	[-0.7,+0.7]
Common monotonic region	[-0.4,+0.4]	[-0.4,+0.4]	[-0.7,+0.7]

To prove the concept, however, and to facilitate the implementation of this approach, a common monotonic region has been selected for all the first 10 orders of each of the polynomial kernels under investigation. As this monotonic window shrinks as the order increases, the common monotonic region for all the first 10 orders would be the one prescribed by the 10^{th} order of each kernel. From Table 5.2, one can see that these regions, for the Stage 2 Chebyshev, Legendre, and Hermite kernels, are [-0.4,+0.4], [-0.4,+0.4] and [-0.7,+0.7], respectively. By this way, all the first 10 orders of each kernel will be processing the data that are normalized to this common monotonic window, rather than making each order process the data which are normalized to its own monotonic window. As such, the experiments conducted in the next section report the results achieved when the first 10 orders of each kernel were used to process the data that are confined to the common monotonic regions amongst all of them.

5.4 Experimental evaluation of the proposed adaptive data normalization approach

5.4.1 Implementation of the adaptive data normalization process

The additional normalization step to confine the data to the monotonic decay windows of each kernel would have the same effect as modifying the kernel shape itself to that of the ideal similarity function throughout the original data region of [-1,+1]. As such, the underlying kernel should be able to calculate more accurate measures of similarity (and hence better classification accuracies) than the kernels that process the 'untreated' data in the way explained above.

The experimental investigations conducted in this section will explore the effect of this additional adaptive data normalization step on the classification performance of the Stage 2 kernels under investigation, when implemented within the SVM algorithm. As a natural practice, pairwise comparisons are conducted on the classification accuracies scored by each order of the employed kernel in two cases. The first case is when the data are normalized to the standard domain of definition of the employed polynomial kernels, which is [-1,+1]. This will be referred to in the experimental results as the 'Standard normalization', for short. The second case is when the additional data normalization step is applied to confine the data to the common monotonic region of all the first 10 orders of the employed kernel, as shown in Table 5.2. Similarly, this will be referred to in the experimental results as the 'Monotonic normalization', for short. This additional normalization step is achieved by down-scaling the standard normalized data by a certain factor for each kernel to confine them to common monotonic region of the first 10 orders of each kernel. Modified from (4.18), this additional downscaling normalization step can be formally defined as:

$$x_i^{new} = \left[\frac{2\left(x_i - \mathrm{Min}_i\right)}{\mathrm{Max}_i - \mathrm{Min}_i} - 1 \right] \times \begin{cases} 0.4 & \text{for Stage 2 Chebyshev} \\ 0.4 & \text{for Stage 2 Legendre} \\ 0.7 & \text{for Stage 2 Hermite} \end{cases} \tag{5.1}$$

where Min_i and Max_i are the minimum and maximum values of the i_{th} feature amongst all the vectors in the dataset, respectively.

5.4.2 Experiments design and setup

With the exception of data normalization, all the other design and setup of the experiments conducted in this chapter are the same as those presented earlier in Chapter 4 (Sections

4.4.1 and 4.4.2). In other words, for consistency, the experiments conducted in this chapter will still utilize the classification accuracy as the method of assessment of the classification performance scored by the employed kernel before and after implementing the proposed approach. The same train/test data division technique for each dataset, as explained back in Section 4.4.2, will be followed for the experiments conducted in this section. The commonly used *C*-SVM will also be utilized as per its implementation in the toolbox utilized from [115].

The experiments conducted in this chapter are focused on only the Stage 2 kernels, which have been shown to outperform those of Stage 1. As such, the selection of the model and hyperparameters is restricted to only the penalization parameter C and the polynomial order capped to n=10. Again, a comprehensive grid search amongst these parameters was conducted, and only the best results scored by each kernel are reported.

Lastly, the experiments conducted in this section will be utilizing the Stage 2 kernels when they are employed to process the input data vectors using the vectorial processing approach proposed by [17] (explained back in Section 3.4.1.2), as this has already been shown to outperform the pairwise approach, originally proposed by [31]. The next chapter will, however, present an alternative approach, which has been shown to be superior in classification performance to these two previously proposed in the literature.

5.4.3 Experimental results and discussions

The experimental results achieved are presented below in three different formats: graphical representations of the maximum accuracies scored, average accuracies, and their bar chart representations of the first 10 polynomial orders. The aim is to provide a comprehensive analysis of the results obtained so that the performance improvement gained as a result of implementing the proposed adaptive data normalization approach can be properly assessed. Six datasets are utilized to conduct the experimental investigation herein. These are: the Breast Cancer, Iris, Image Segmentation, Ionosphere, Thyroid, and Two Spirals datasets; where the training and test examples for each dataset were selected as per the explanation in Section 4.4.2.

Table 5.3 illustrates the experimental results obtained in a pairwise graphical format of the best classification accuracy scored by the first 10 orders of the Stage 2 kernels under test for both the standard normalization and the proposed monotonic normalization approaches. As shown, the kernels under test have demonstrated different behaviours across the

datasets and the polynomial orders investigated. However, in general, it can be observed that the proposed monotonic normalization approach outperforms the standard normalization for most of the orders and the datasets experimented. Given that the only difference between the two sets of results is the implementation of the proposed adaptive data normalization approach, which confines the data to the monotonic decay regions, where the employed kernel demonstrates ideal similarity measure characteristics, it can be concluded that the implementation of such an approach did indeed make the employed kernels produce more accurate measures of similarity, a fact, which is reflected constructively on their resulting classification performance, as can be observed in the results herein.

In terms of the polynomial orders, one can also observe (for example, from the results obtained from the Image Segmentation and Thyroid datasets in Table 5.3) that the classification performance of the standard normalization drops significantly at higher orders. This is believed to be attributed to the fact that the monotonic decay window threshold of these Stage 2 kernels shrinks as the order increases, which increases the chance of more data points, that happen to be located outside this window, to be prone to miscalculation of the measures of similarity amongst them, due to the oscillating wavy pattern of these polynomial kernels, as explained earlier in Section 5.2. As the proposed monotonic normalization approach does not suffer from this symptom, as it is specifically designed to rectify this problem, its experimental results have therefore demonstrated a relatively steady classification performance across the polynomial orders compared to when the datasets are normalized to the standard region of [-1,1].

Moreover, by eliminating the chance of data points falling outside the kernels' monotonic decay window threshold, where the kernels oscillate in a wavy pattern, the proposed monotonic normalization approach also avoids any fluctuations in the calculated measures of similarity by correcting the cases where standard normalization fails. This can be observed, for example, in the oscillations appearing in the Breast Cancer dataset and the major collapse in the classification accuracy scored by the 5^{th} order kernels in the Ionosphere dataset. Consequently, the monotonic normalization approach was therefore able to produce a more consistent and steady classification accuracies, across the spectrum of polynomial orders, than the results obtained from the standard normalization. This can also be quantitatively realized by directly comparing the standard deviation (in relation to the mean) of the two sets of results (as listed in the tabulated statistical analysis tests in Appendix B.3) where the standard deviation of the monotonic normalization results is

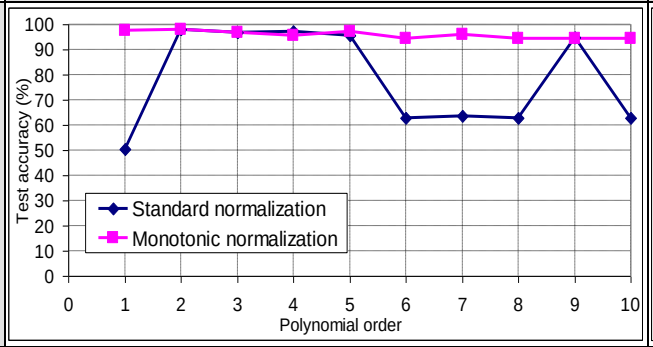
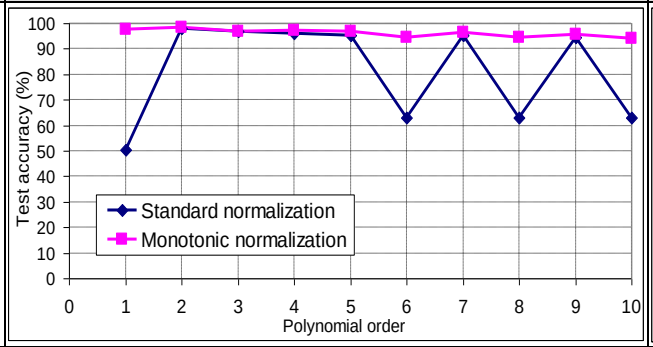
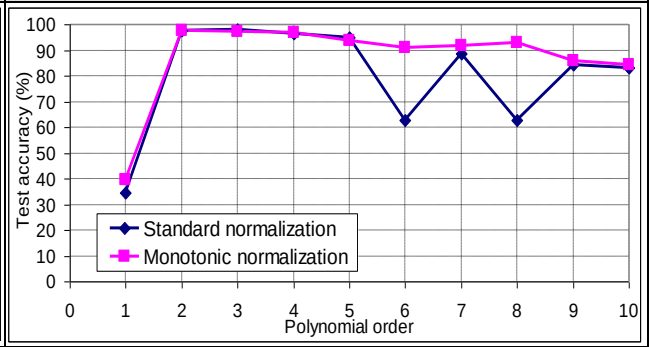
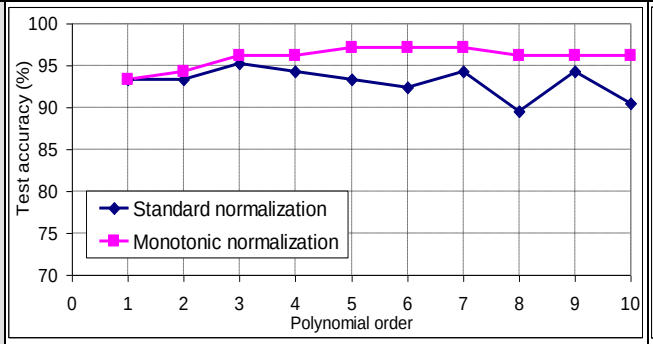
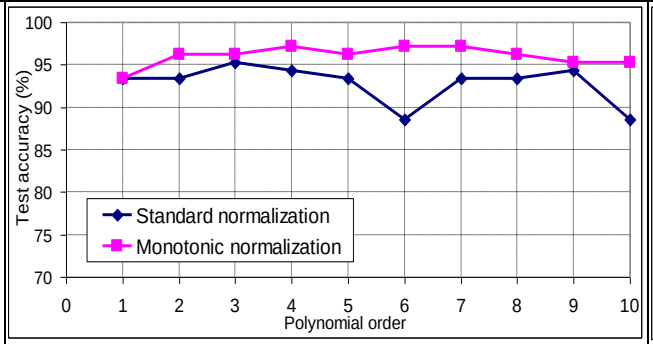
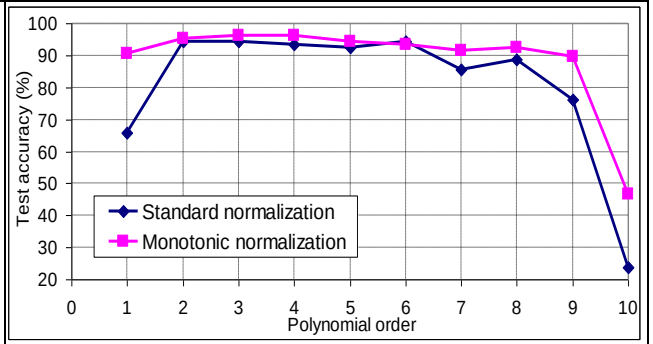
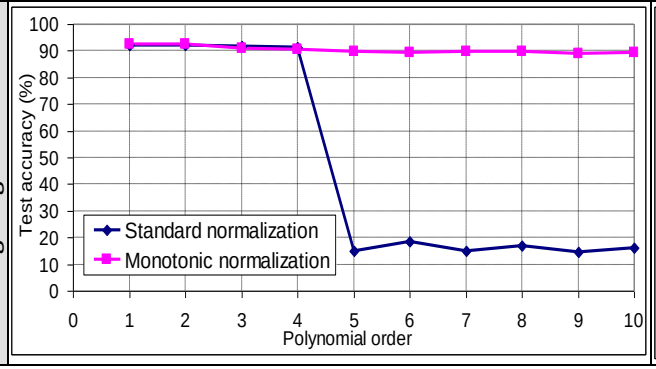
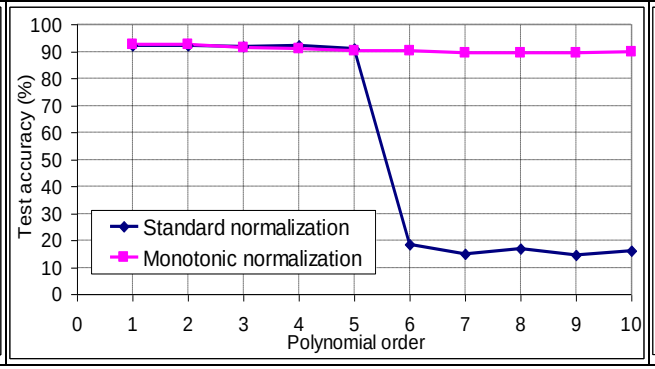
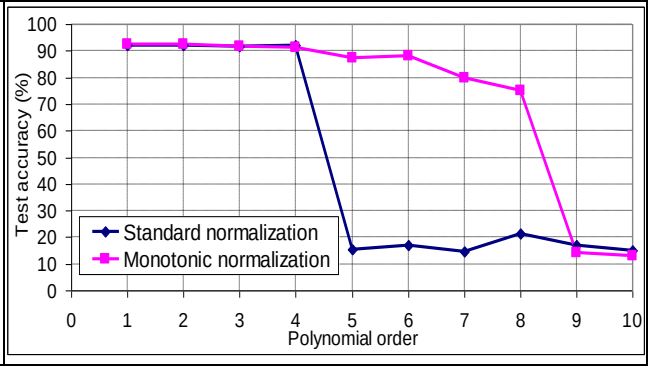
consistently smaller than the standard deviation of the results obtained from its standard normalization counterpart.

For a further quantitative assessment of these results, Table 5.4 also lists the actual values of the best classification accuracy scored by the Stage 2 kernels under test, for each of the data normalization approaches implemented. By directly subtracting the two sets of results from each other for each order, one can get an average figure of how much improvement in classification accuracy can be gained by implementing the proposed adaptive data normalization approach. At the bottom of the table, this was calculated to be equal to 16.48%, 15.04%, and 3.795% for the Stage 2 Chebyshev, Legendre, and Hermite kernels, respectively, making up a total average improvement factor of 11.772%, depending on the dataset and the polynomial utilized.

It should be noted, however, that the improvement amongst the polynomial orders is quite diverse (due to the oscillating performance of the standard normalization in some datasets, as explained earlier) and can sometimes vary significantly throughout the spectrum of polynomial orders. For this reason, Table 5.4 has also been used to calculate the standard deviation of the average improvement factors introduced by the proposed monotonic approach, which is shown in the table in brackets beside every average improvement factor calculated. The minimum and maximum values of the improvement factors achieved are also shown in bold in the table. Unlike the standard deviation of the actual results of the classification accuracies (alluded to above and listed in the statistical analysis tables in Appendix B.3), one can notice, however, that the standard deviation of the average improvement factors shown in Table 5.4 is quite sparse and can actually be larger than the mean at times, which clearly indicates how diverse the difference between the results are.

The bar charts in Figure 5.1 also illustrate a pairwise comparative assessment of the average classification accuracy scored by the first 10 orders of the Stage 2 kernels for each of the data normalization approaches under test. Again, it can be observed that the proposed adaptive monotonic data normalization approach demonstrated a superior average performance over its standard data normalization counterpart in most of the datasets under test; which again shows that the employed kernels are now able to calculate more accurate measures of similarity, and hence enable the subsequent SVM classification performance to improve.

Table 5.3 Experimental results of the best classification accuracies scored by the first 10 orders of Stage 2 kernels using the standard vs. the proposed monotonic data normalization approach.

	Stage 2 Chebyshev kernels	Stage 2 Legendre kernel	Stage 2 Hermite kernel
Breast Cancer			
Iris			
Image Segmentation			

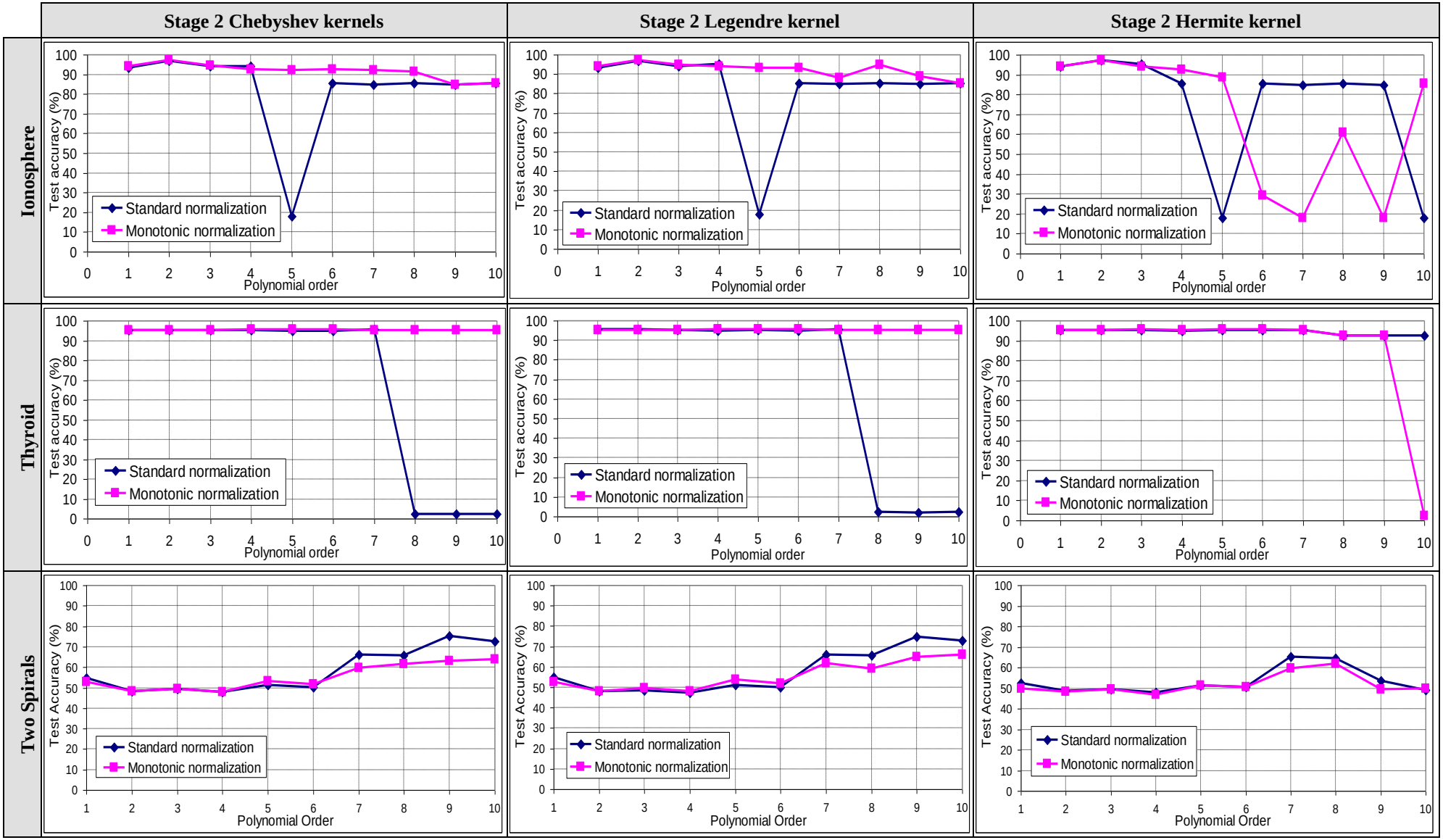
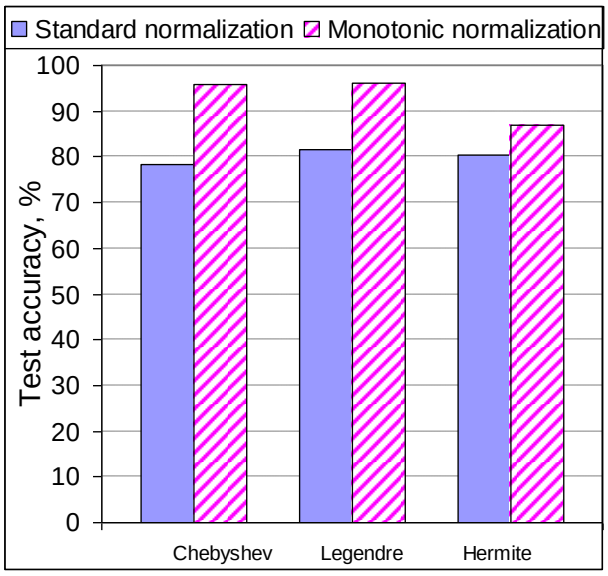

Stage 2 Chebyshev kernels
Stage 2 Legendre kernel
Stage 2 Hermite kernel
Ionosphere
Thyroid
Two Spirals
Test accuracy (%)
Test Accuracy (%)
Polynomial order
Polynomial Order
Standard normalization
Monotonic normalization

Table 5.4 Quantitative assessment of the classification improvements gained by implementing the proposed adaptive monotonic data normalization approach.

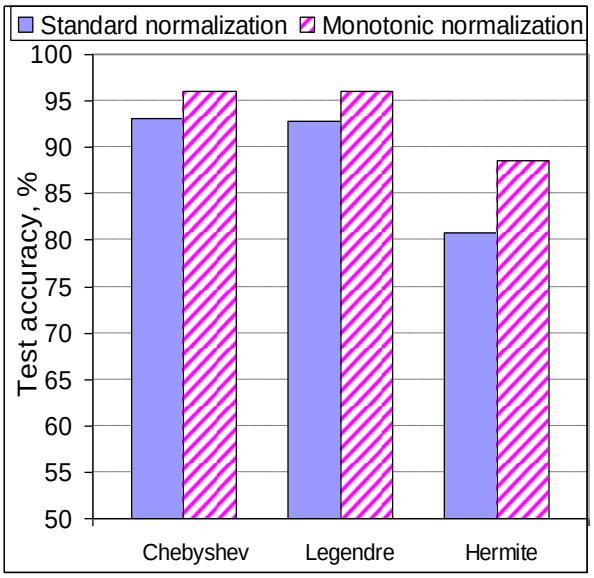
		Stage 2 Chebyshev kernels										Stage 2 Legendre kernels										Stage 2 Hermite kernels									
		Order no.										Order no.										Order no.									
		1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
Breast Cancer	**N1**	50.320	97.868	96.802	97.228	95.522	62.900	63.539	62.900	94.670	62.900	50.320	97.868	97.015	96.162	95.309	62.900	95.096	62.900	94.670	62.900	34.328	97.655	97.868	96.588	95.096	62.900	88.699	62.900	84.222	82.942
	N2	97.441	98.081	96.802	95.522	97.228	94.243	95.949	94.243	94.456	94.243	97.441	98.294	96.802	97.228	96.802	94.670	96.375	94.670	95.522	94.030	39.443	97.655	97.441	96.802	93.817	91.045	91.684	92.751	85.714	84.222
	I	**47.122**	0.213	0	**-1.706**	1.706	31.343	32.409	31.343	-0.213	31.343	**47.122**	0.426	**-0.213**	1.066	1.493	31.770	1.279	31.770	0.853	31.130	5.117	0	-0.426	0.213	**-1.279**	28.145	2.985	**29.851**	1.493	1.279
	AI	**17.356 (18.89)**										**14.670 (18.446)**										**6.738 (11.879)**									
Iris	**N1**	93.333	93.333	95.238	94.286	93.333	92.381	94.286	89.524	94.286	90.476	93.333	93.333	95.238	94.286	93.333	88.571	93.333	93.333	94.286	88.571	65.714	94.286	94.286	93.333	92.381	94.286	85.714	88.571	76.190	23.810
	N2	93.333	94.286	96.190	96.190	97.143	97.143	97.143	96.190	96.190	96.190	93.333	96.190	96.190	97.143	96.190	97.143	97.143	96.190	95.238	95.238	90.476	95.238	96.190	96.190	94.286	93.333	91.429	92.381	89.524	46.667
	I	**0**	0.9524	0.952	1.905	3.810	4.762	2.857	**6.667**	1.905	5.714	**0**	2.857	0.952	2.857	2.857	**8.571**	3.810	2.857	0.952	6.667	**24.762**	0.952	1.905	2.857	1.905	**-0.952**	5.714	3.810	13.333	22.857
	AI	**2.952 (2.22)**										**3.238 (2.626)**										**7.714 (9.325)**									
Image Segmentation	**N1**	92.143	92.095	91.714	91.238	14.905	18.381	15	16.857	14.619	16.143	92.143	92.095	91.571	92.048	90.81	18.429	15	16.857	14.667	16.143	92.19	92.095	91.476	92.095	15.381	17.095	14.619	21.19	17.048	15.095
	N2	92.476	92.381	90.952	90.619	89.857	89.333	89.714	89.81	88.905	89.429	92.476	92.381	91.429	91.143	90.143	90.095	89.571	89.476	89.238	89.81	92.476	92.381	91.571	91.476	87.429	88.333	80.001	75.01	14.381	13.095
	I	0.333	0.286	**-0.762**	-0.619	**74.952**	70.952	74.714	72.952	74.286	73.286	0.333	0.286	-0.143	**-0.905**	-0.667	71.667	74.571	72.619	**74.571**	73.667	0.286	0.286	-0.095	-0.619	**72.048**	71.238	65.381	53.81	**-2.667**	-2
	AI	**44.038 (38.083)**										**36.6 (38.822)**										**25.767 (34.654)**									
Ionosphere	**N1**	93.377	96.689	94.040	94.040	17.881	85.430	84.768	85.430	84.768	85.430	93.377	96.689	94.040	95.364	17.881	85.430	84.768	85.430	84.768	85.430	94.040	97.351	95.364	85.430	17.881	85.430	84.768	85.430	84.768	17.881
	N2	94.040	97.351	94.702	92.715	92.053	92.715	92.053	91.391	84.768	85.430	94.040	97.351	94.702	94.040	93.377	93.377	88.079	94.702	88.742	85.430	94.040	97.351	94.040	92.715	88.742	29.139	17.881	60.927	17.881	85.430
	I	0.662	0.662	0.662	**-1.325**	**74.172**	7.285	7.285	5.960	0	0	0.662	0.662	0.662	**-1.325**	**75.497**	7.947	3.311	9.271	3.974	0	0	0	-1.325	7.285	**70.861**	-56.291	-66.887	-24.503	**-66.887**	67.550
	AI	**9.536 (40.816)**										**10.066 (23.251)**										**-7.020 (49.287)**									

		Stage 2 Chebyshev kernels										Stage 2 Legendre kernels										Stage 2 Hermite kernels									
		Order no.										Order no.										Order no.									
		1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10	1	2	3	4	5	6	7	8	9	10
Thyroid	**N1**	95.478	95.449	95.420	95.420	95.012	95.012	95.683	2.392	2.159	2.363	95.478	95.449	95.391	94.866	95.070	95.012	95.537	2.392	2.159	2.363	95.478	95.216	95.449	94.749	95.303	95.362	95.274	92.707	92.707	92.707
	N2	95.420	95.391	95.478	95.566	95.537	95.595	95.333	95.303	95.303	95.303	95.420	95.391	95.420	95.566	95.508	95.537	95.420	95.362	95.274	95.245	95.478	95.449	95.537	95.508	95.624	95.624	95.449	92.707	92.707	2.188
	I	-0.058	-0.058	0.058	0.146	0.525	0.583	**-0.350**	92.911	**93.145**	92.940	-0.058	-0.058	0.029	0.700	0.438	0.525	**-0.117**	92.970	**93.116**	92.882	0	0.233	0.088	**0.758**	0.321	0.263	0.175	0	0	**-90.519**
	AI	**27.984 (44.865)**										**28.043 (44.818)**										**-8.868 (28.69)**									
Two Spirals	**N1**	54.874	48.124	49.519	48.032	51.19	50.046	65.995	65.652	75.378	72.746	54.874	48.124	48.467	47.506	51.190	50.046	65.995	65.652	74.851	72.746	52.494	48.558	49.519	48.032	51.190	50.664	65.469	64.600	53.455	48.902
	N2	52.677	48.124	49.519	48.032	53.204	51.716	59.664	61.755	63.007	64.007	52.677	48.124	49.519	48.032	53.730	51.808	61.664	59.190	64.979	65.979	49.863	48.124	49.519	46.979	51.190	50.572	59.783	61.991	49.284	49.954
	I	-2.197	0	0	0	**2.014**	1.67	-6.332	-3.897	**-12.371**	-8.739	-2.197	0	1.053	0.526	**2.54**	1.762	-4.332	-6.462	**-9.872**	-6.767	-2.632	-0.435	0	-1.053	0	-0.092	**-5.687**	-2.609	-4.172	**1.053**
	AI	**-2.985 (4.799)**										**-2.375 (4.257)**										**-1.562 (2.147)**									
AAI		**16.48 (30.364)**										**15.04 (29.346)**										**3.795 (29.199)**									
TAI		**11.772 (30.018)**																													

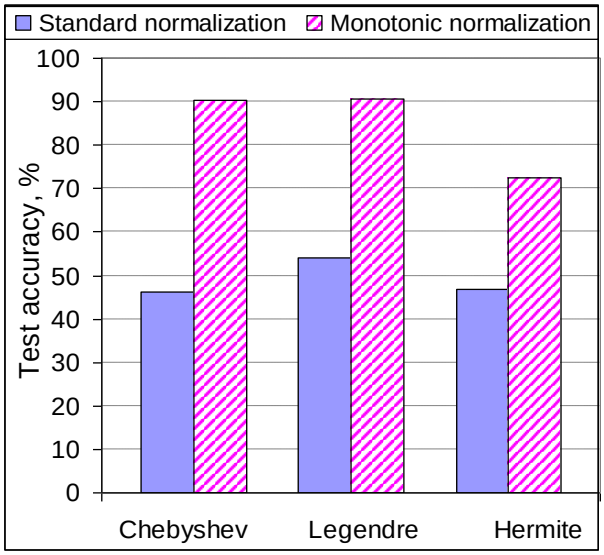
N1 Classification accuracy scores when the standard normalization is used
N2 Classification accuracy scores when the proposed monotonic data normalization approach is used
I Improvement = N2 - N1
AI Average Improvement for the first 10 orders
AAI Average of Average Improvements of the first 10 orders
TAI Total Average Improvement of all the experimented datasets and kernels for the first 10 orders



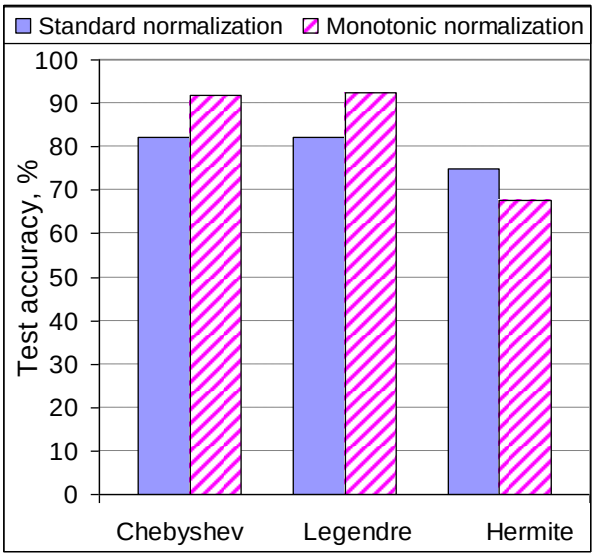
(a) Breast Cancer



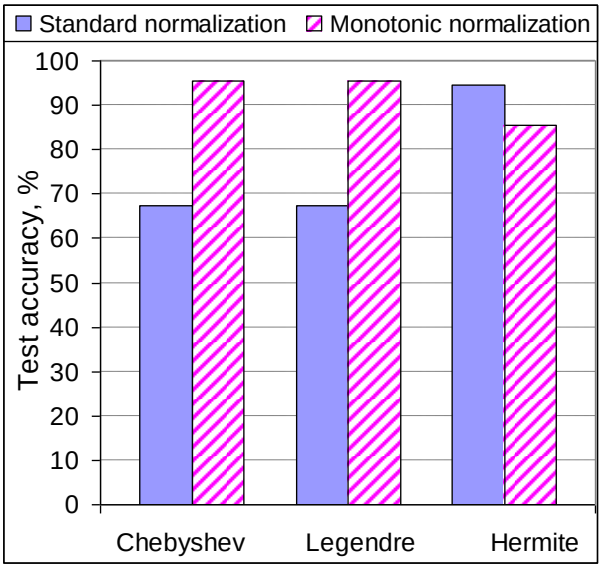
(b) Iris



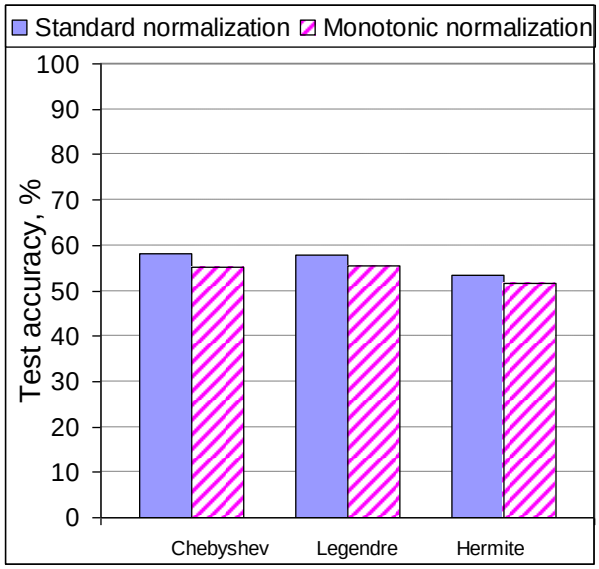
(c) Image segmentation



(d) Ionosphere



(e) Thyroid



(e) Two Spirals

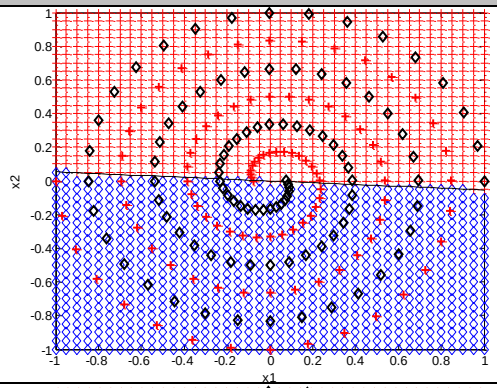
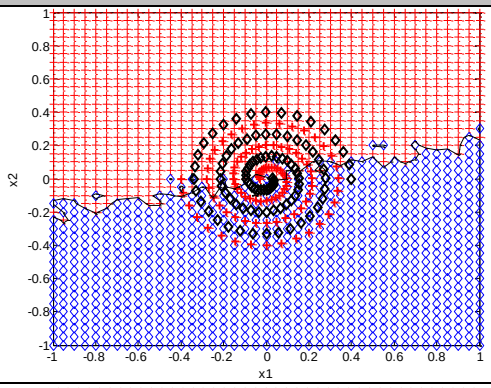
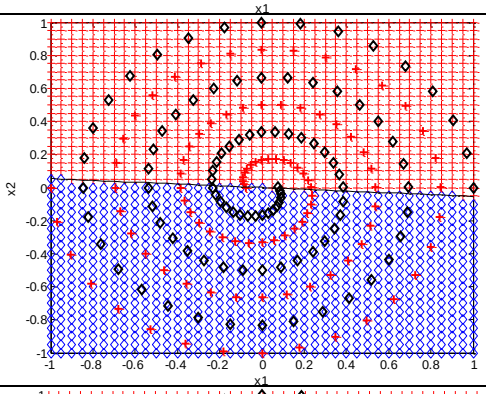
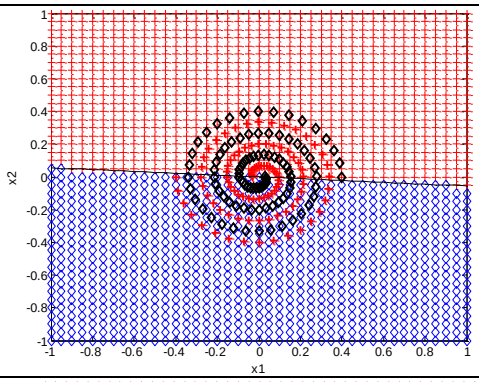
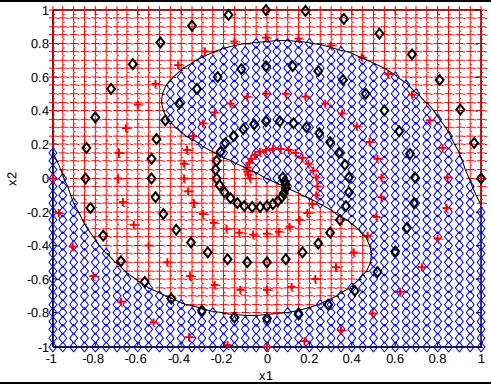
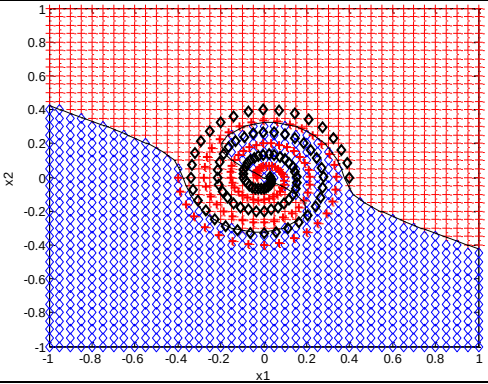
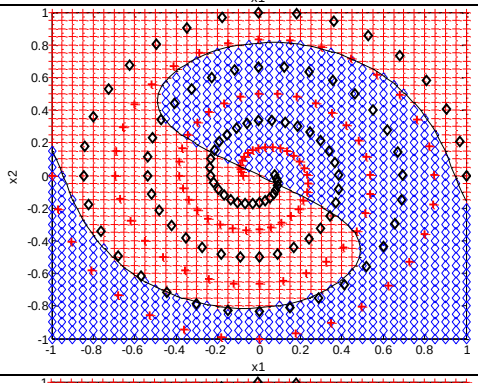
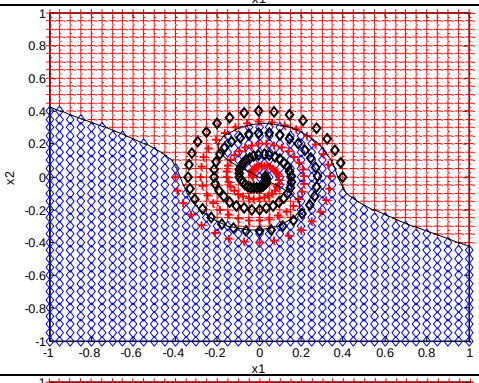
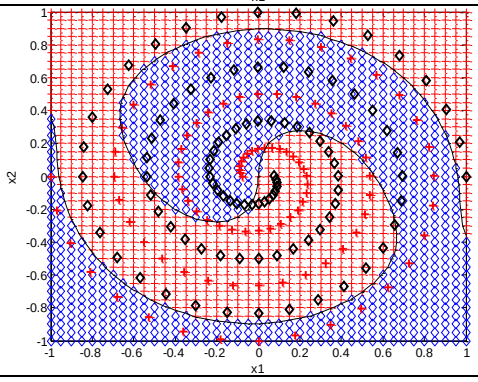
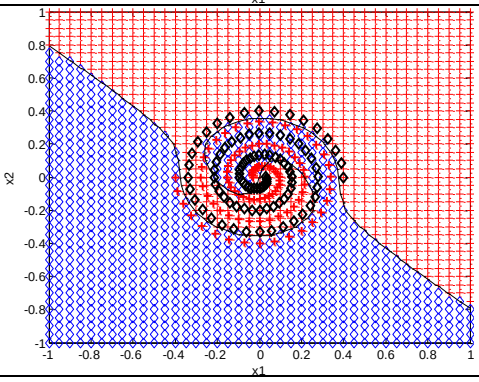
Figure 5.1 Bar chart comparative assessment of the average classification accuracy scored by Stage 2 kernels with standard data normalization and the proposed adaptive monotonic data normalization approach.

On the other hand, despite the superiority of the proposed monotonic approach over its standard normalization counterpart demonstrated in the majority of the results shown above, yet, one can still notice that there are cases where the monotonic normalization approach does not actually introduce any improvement at all and that the standard normalization is still superior. This can be observed in Table 5.3, for example, in some polynomial orders in some datasets, such as the higher orders Hermite kernel in the Ionosphere and Thyroid datasets, and the higher order polynomial kernels experimented on the Two Spirals dataset. To further investigate the potential reason behind this phenomenon, another set of experiments have also been conducted that took advantage of the 2D dimensionality of the Two Spirals dataset to enable the visualization of the classification performance amongst the two normalization approaches under investigation and throughout the first 10 polynomial orders of the kernels under test.

As such, Tables 5.5, 5.6, and 5.7 illustrate the visualization assessment of the classification performance on the Two Spirals dataset using the Stage 2 Chebyshesh, Legendre, and Hermite kernels, respectively. This is achieved via a pairwise comparison of the behaviour of each kernel for the two normalization approaches under investigation and for each of the first 10 orders under test. A common observation amongst the three kernels that can easily be realized from the figures is that as the polynomial order increases, both of the standard and monotonic normalization approaches try to gradually follow the spiral flow of the dataset. Hence higher polynomial orders will tend to score higher classification accuracies than lower polynomial orders, which is actually inline with the graphical results of this dataset, shown earlier in Table 5.3.

However, due to the high non-linearity profile of this dataset, which increases towards the origin, where the two spiral strands become more and more intertwined over each other, the implementation of the monotonic normalization approach results in squeezing the data more towards the origin. This increases the severity of the non-linearity between the two class labels around the immediate vicinity of the origin rather than distributing it around the normalization region; a fact which makes the classification algorithm struggle to separate the two classes. In other words, the experimental results of the Two Spirals dataset illustrated in Tables 5.5 – 5.7 seem to suggest that the proposed monotonic normalization approach works better when the non-linearity profile between the overlapped class labels is distributed throughout the normalization region rather than being condensed around the origin.

Table 5.5 Visualization assessment of the classification performance on the Two Spirals dataset using the Stage 2 Chebyshev kernels.

n	Stage 2 Chebyshev Standard Normalization	Stage 2 Chebyshev Monotonic Normalization
1		
2		
3		
4		
5		

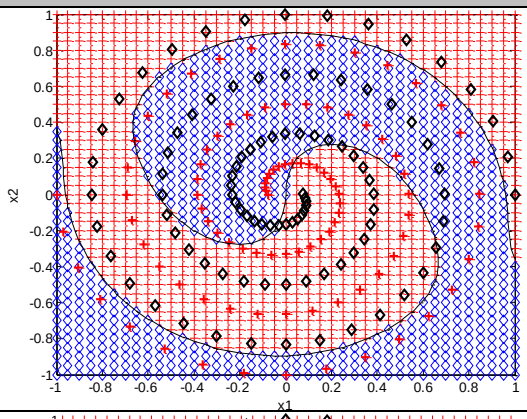
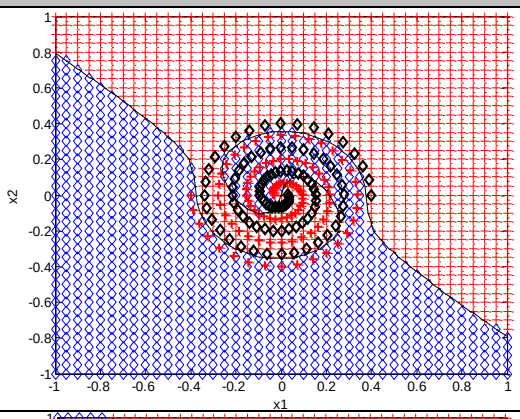
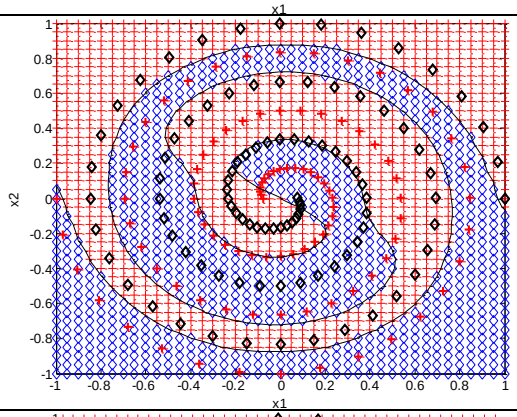
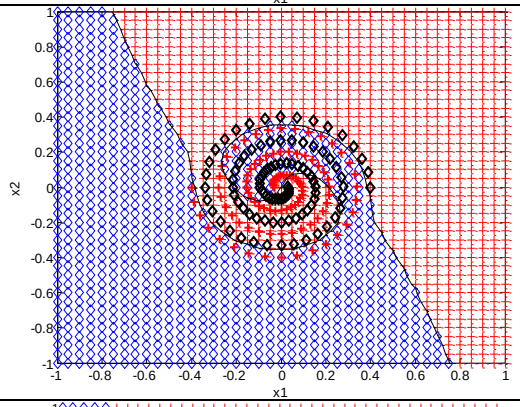
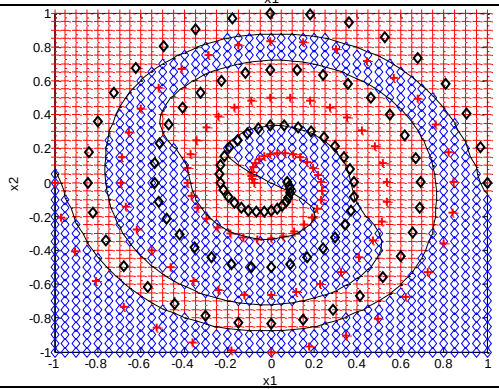
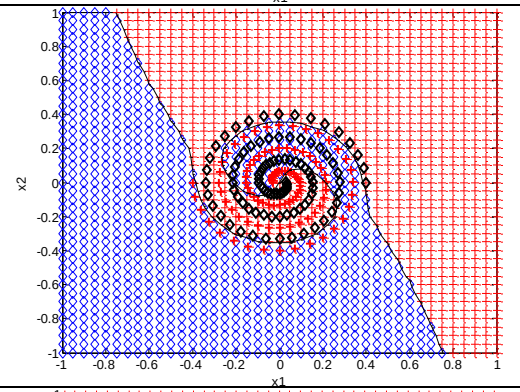
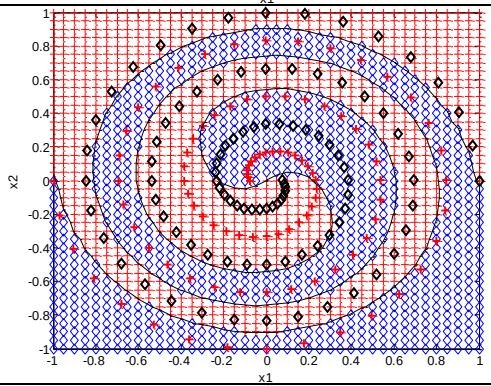
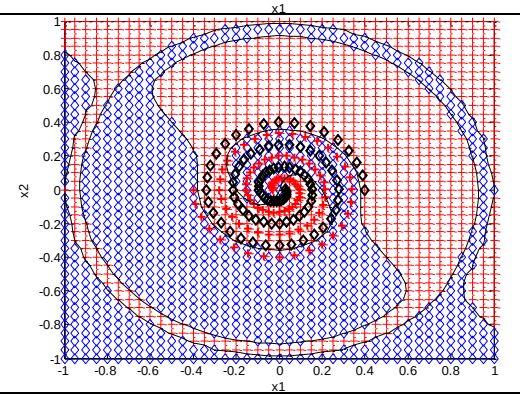
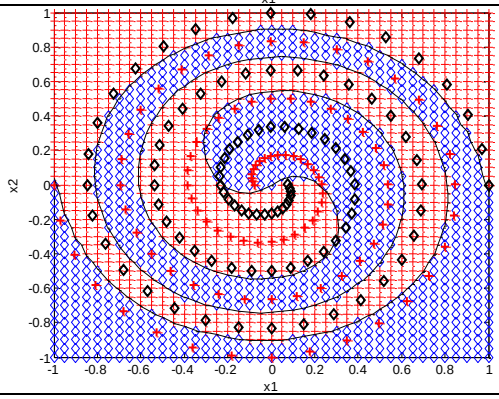
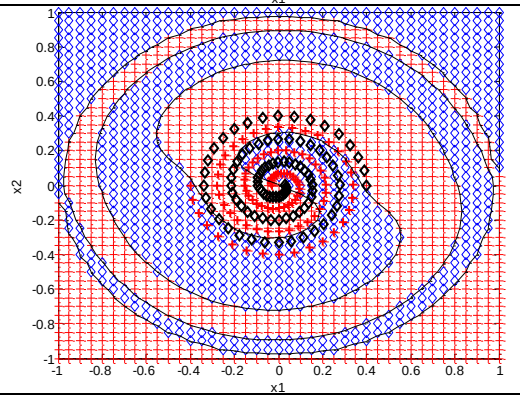
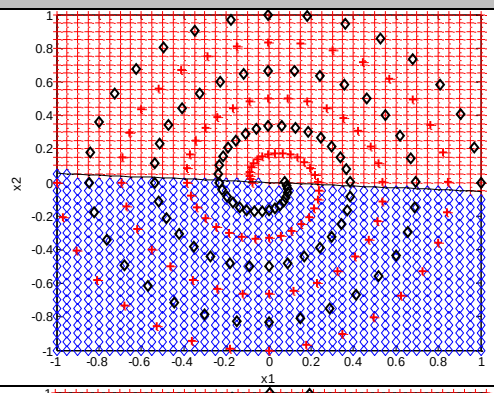
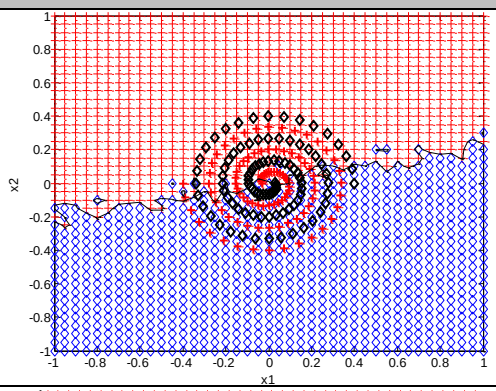
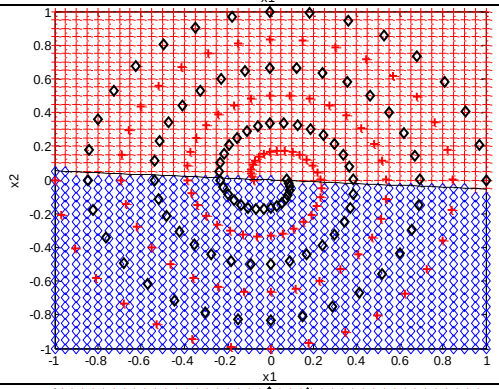
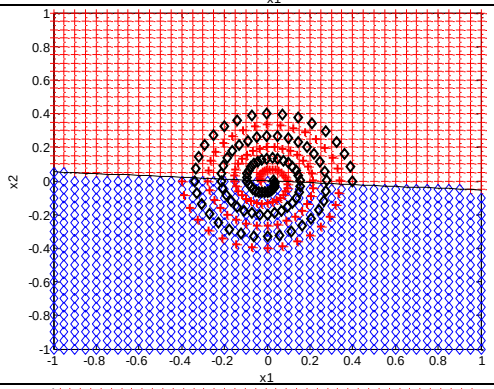
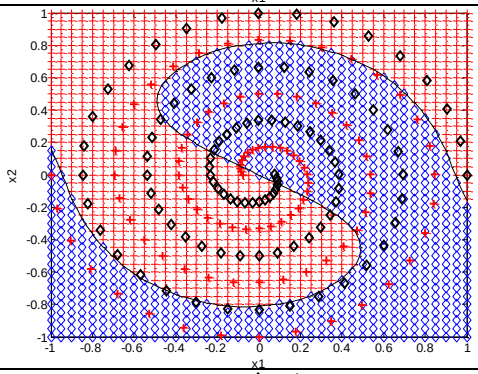
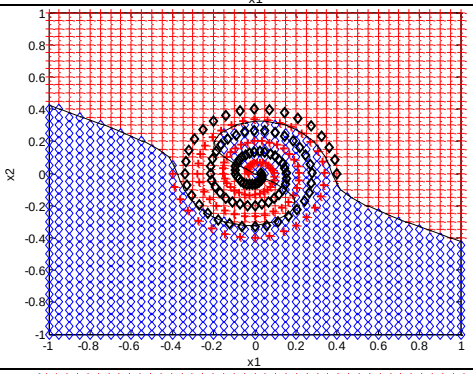
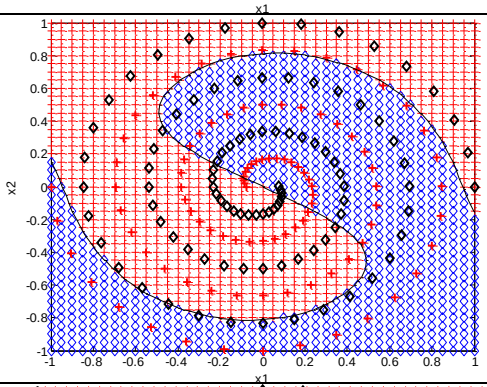
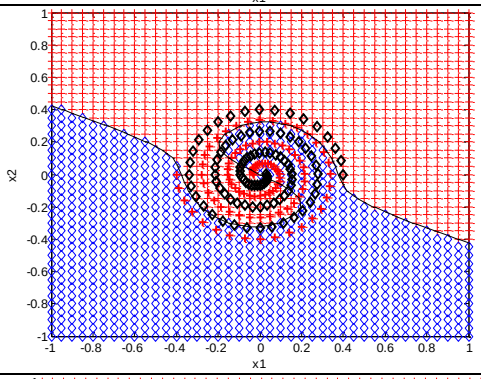
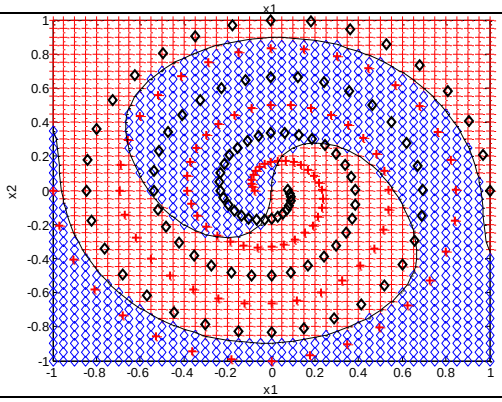
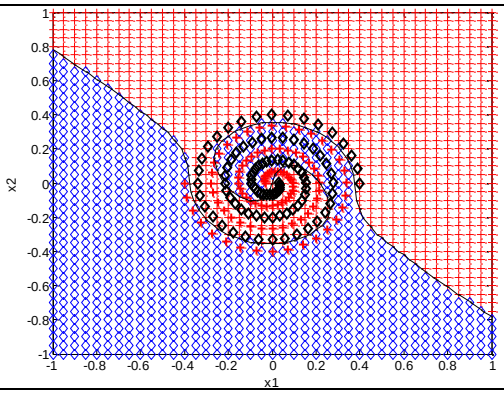
n	Stage 2 Chebyshev Standard Normalization	Stage 2 Chebyshev Monotonic Normalization
6		
7		
8		
9		
10		

Table 5.6 Visualization assessment of the classification performance on the Two Spirals dataset using the Stage 2 Legendre kernels.

n	Stage 2 Legendre Standard Normalization	Stage 2 Legendre Monotonic Normalization
1		
2		
3		
4		
5		

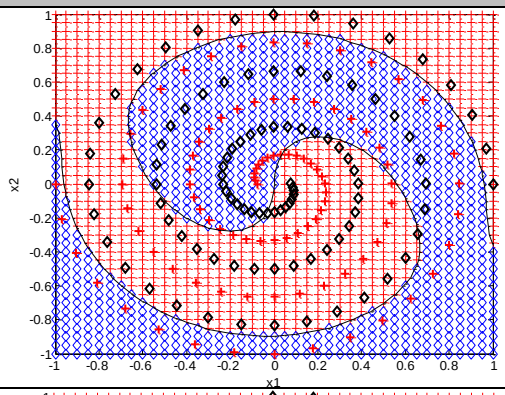
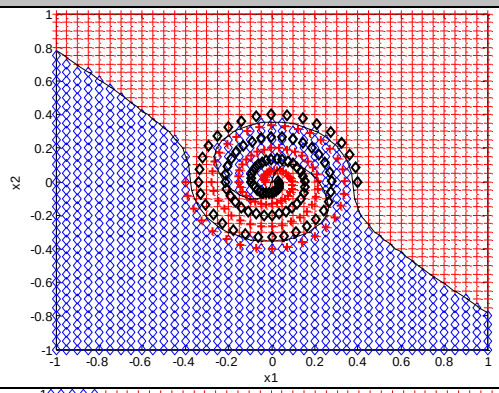
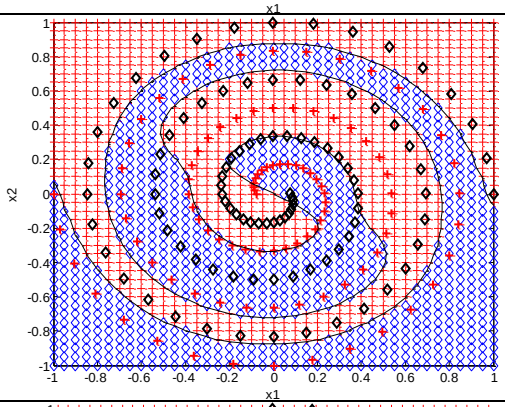
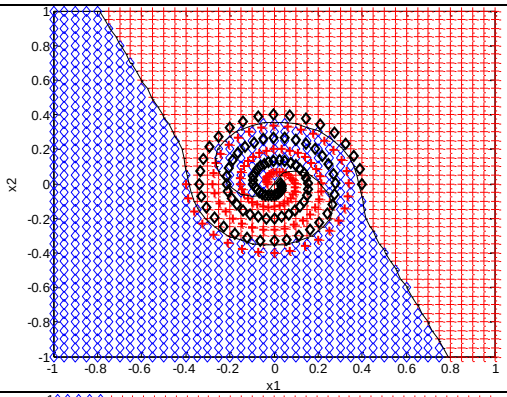
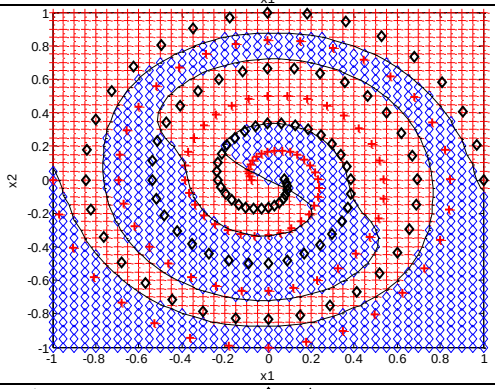
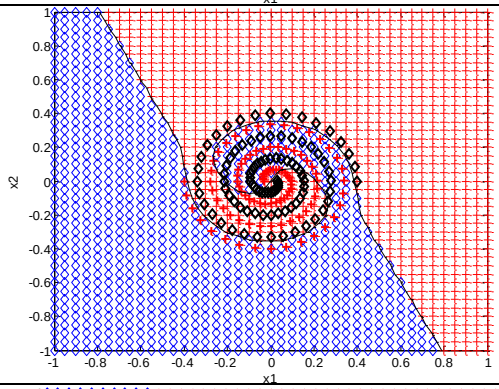
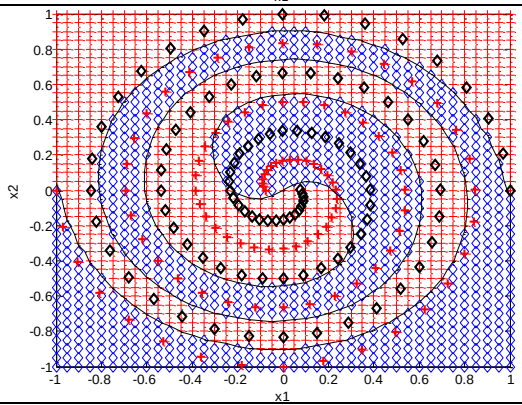
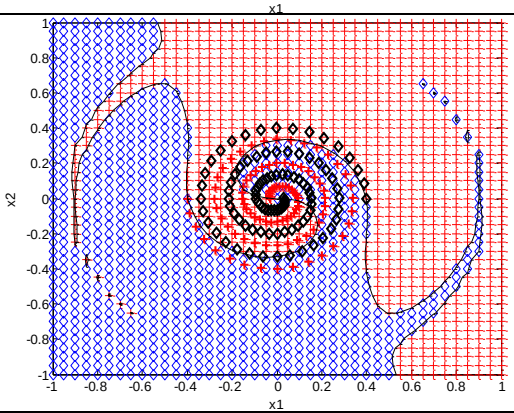
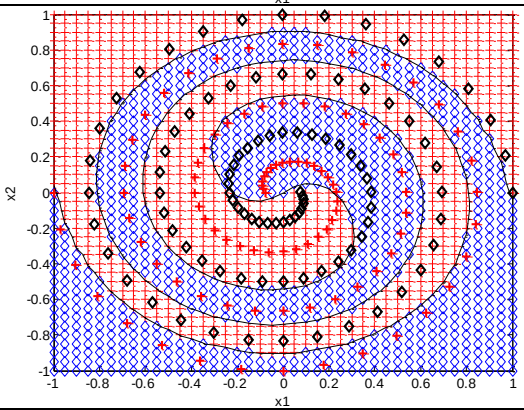
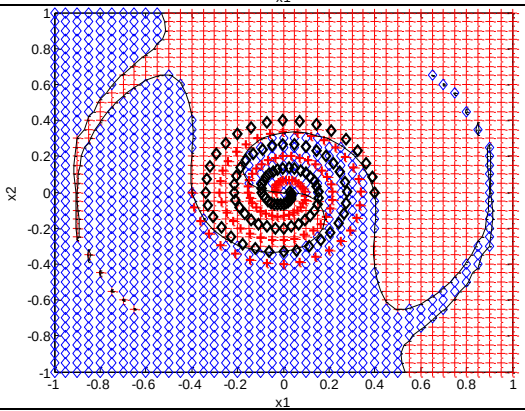
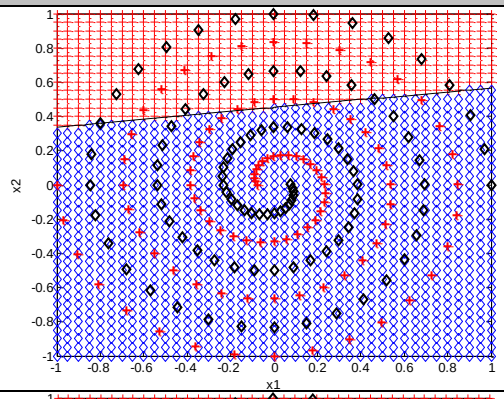
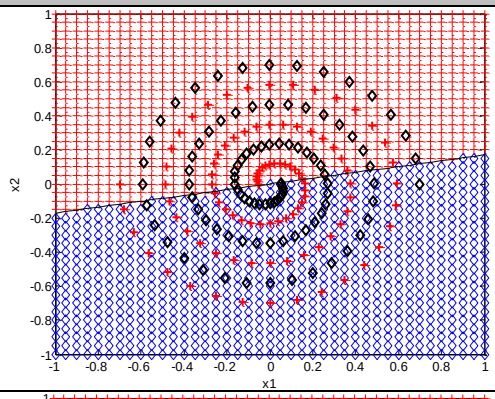
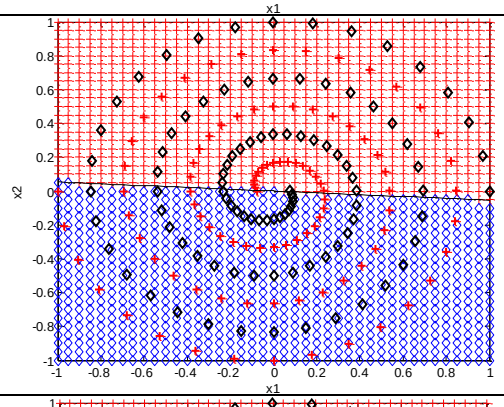
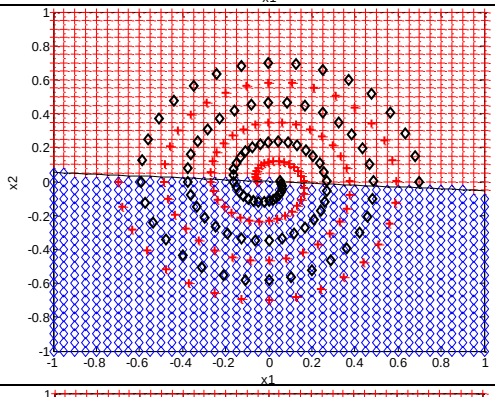
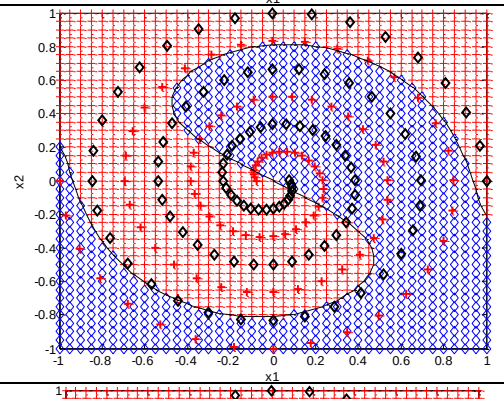
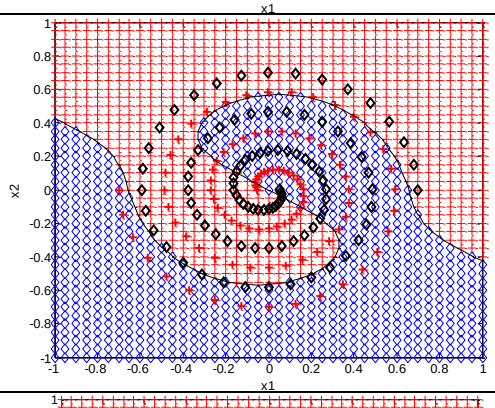
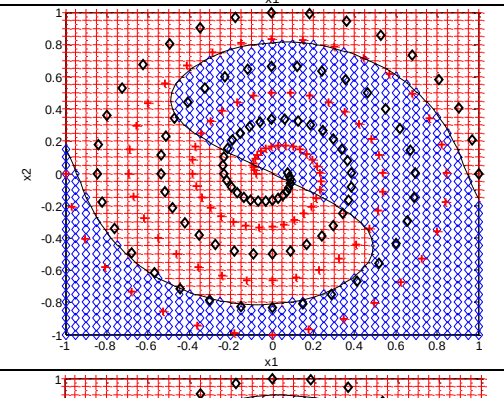
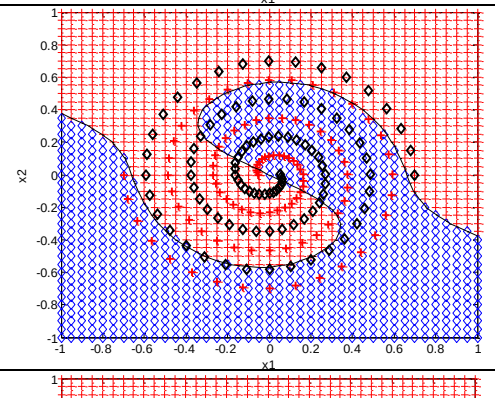
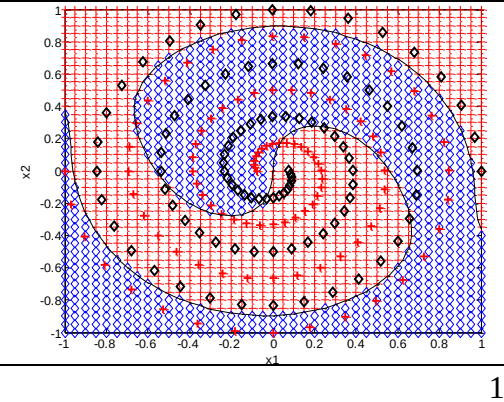
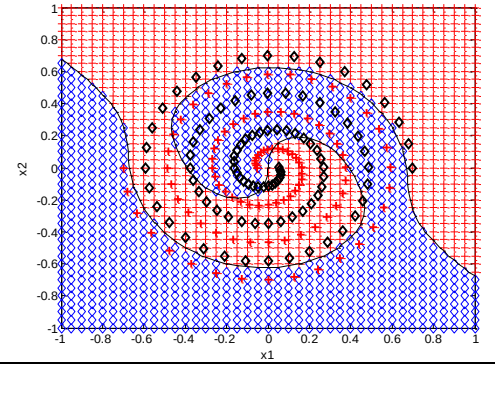
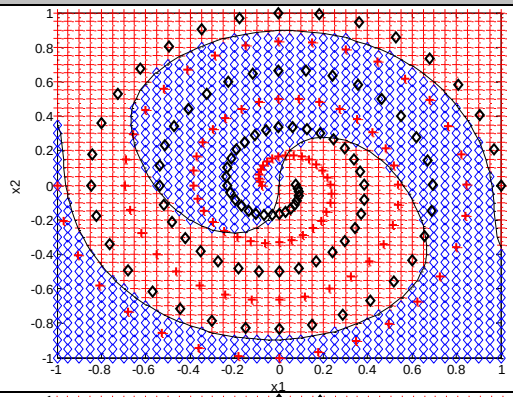
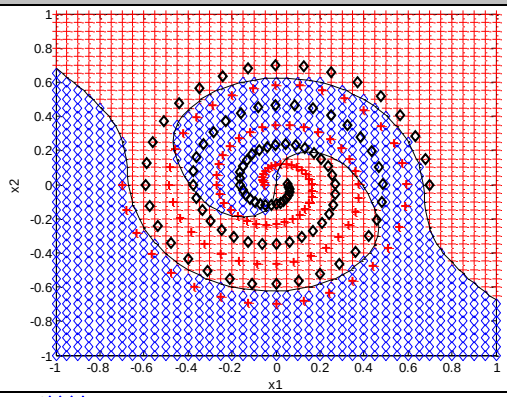
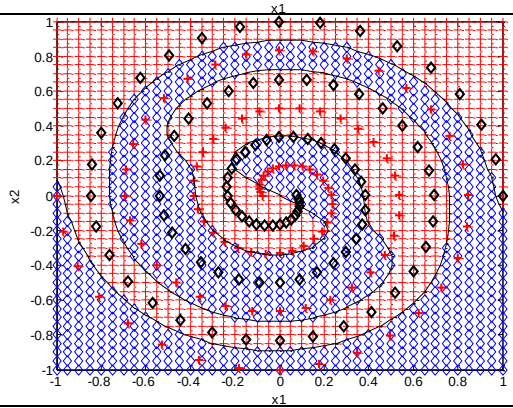
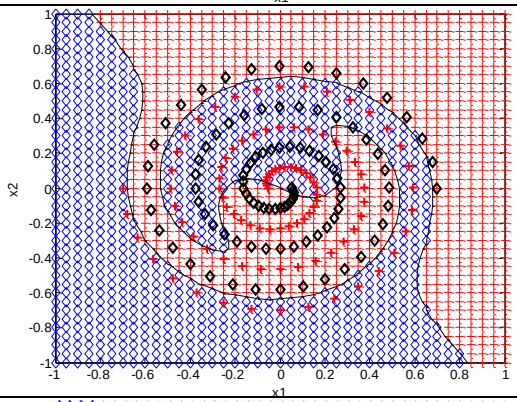
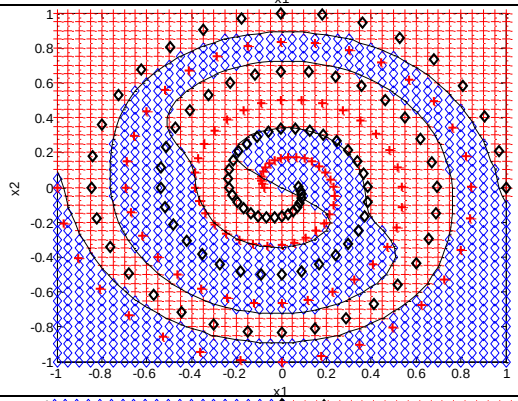
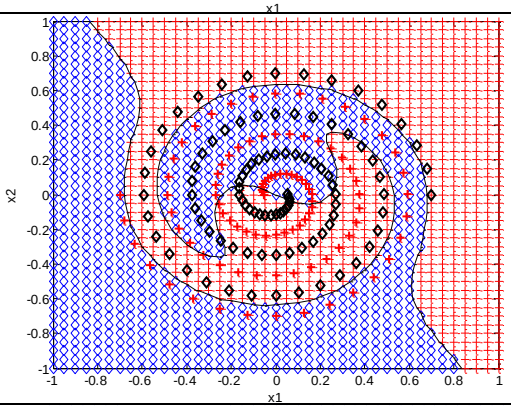
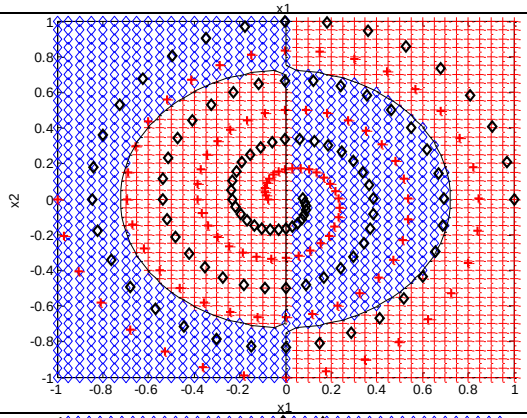
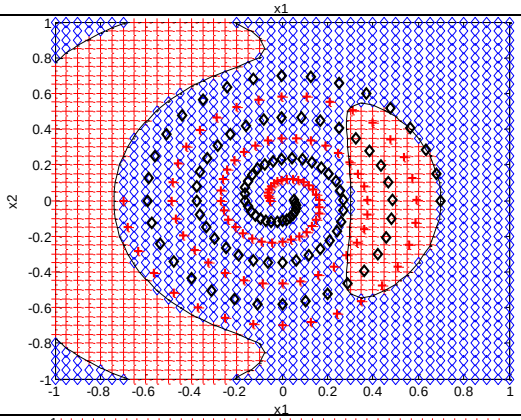
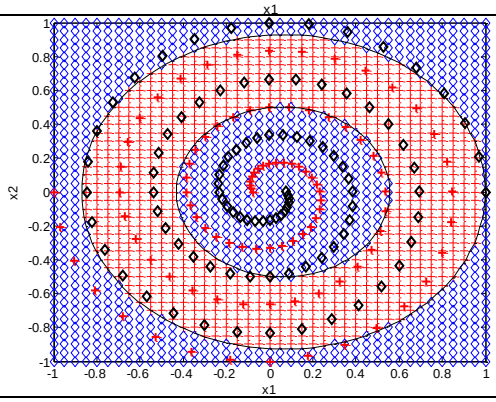
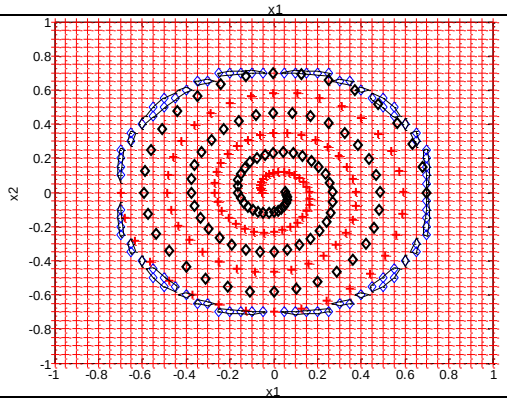
n	Stage 2 Legendre Standard Normalization	Stage 2 Legendre Monotonic Normalization
6		
7		
8		
9		
10		

Table 5.7 Visualization assessment of the classification performance on the Two Spirals dataset using the Stage 2 Hermite kernels.

n	**Stage 2 Hermite Standard Normalization**	**Stage 2 Hermite Monotonic Normalization**
1		
2		
3		
4		
5		

n	Stage 2 Hermite Standard Normalization	Stage 2 Hermite Monotonic Normalization
6		
7		
8		
9		
10		

5.5 Summary

This chapter proposed a solution to the problem where the Stage 2 polynomial kernels were found to oscillate in a wavy pattern after a certain threshold within the standard normalized vector space. The solution was based on applying an adaptive normalization approach to the input data to confine them to the regions where the adopted polynomial kernels exhibit the desired monotonic decay behaviour. Hence eliminate the possibility of any data point to be located outside this region, and accordingly enable the underpinning polynomial kernel to calculate more accurate similarity measures.

The effectiveness of this approach on the resulting classification performance was then assessed experimentally on the Stage 2-Chebyshev, Legendre, and Hermite polynomial kernels, and using all the six datasets previously selected in this thesis. Pairwise comparative analyses of the results demonstrate the superiority of the classification accuracy when the data are confined to the monotonic regions of the kernels as compared to when they are normalized to their standard region of [-1,+1].

On average, the analyses of the experimental results have shown that the classification performance of the kernels can be improved by a factor of 11.772% when implementing the proposed adaptive data normalization approach, but with a standard deviation factor of 30.0189%, due to the oscillating behaviour of the standard normalization amongst the polynomial orders in some datasets. The experimental results have also shown that in cases where the non-linearity profile of the dataset is quite severe around the immediate vicinity of the origin, the proposed monotonic normalization approach is not as effective as its standard normalization counterpart.

Chapter 6
Transformation of Multi-Dimensional Vectors Using Polynomial Kernels

6.1 Introduction

So far, the analyses conducted in this thesis on SVM kernels that are constructed from orthogonal polynomials, together with the previously proposed methodologies, by which these polynomials process multi-dimensional input vectors (referred to in this thesis as the 'pairwise' and 'vectorial' processing approaches), revealed that the application of these polynomial functions to the input vectors amounts to an explicit transformation of the input vectors to another vector space of the same dimensionality, prior to the kernel calculation step. Motivated by the reported empirical advantages of the kernels constructed from such polynomials (e.g., less number of support vectors, less memory and execution time, superior accuracy, etc.) over the traditional kernels that have been in common use, this chapter investigates this transformation perspective in more detail to construct a novel framework within which the use of such polynomials is appropriately articulated and defined to construct similarity measure kernels.

The chapter starts with a detailed analysis of this new transformation perspective using the Chebyshev, Legendre, and Hermite polynomials, and proposes the utilization of the Gaussian kernel (as opposed to the linear kernel used in previous approaches) to facilitate the linear separability of overlapped classes in the high-dimensional feature space. The chapter then proposes a new processing approach, based on vector concatenation, by which the polynomials process the input vectors, to ensure that the transformation process will always produce vector quantities, as well as rectifying the drawbacks identified in its 'pairwise' and 'vectorial' predecessor approaches.

A comprehensive set of experiments is conducted afterwards to explore the effect of this new approach on all the three kernel stages of the similarity fusion framework (proposed in Chapter 4) when using all the three different polynomials under study.

6.2 Transformation perspective of input data vectors

From the investigations conducted in the previous chapters on orthogonal polynomial kernels, one can notice that the different families of orthogonal polynomials investigated in this thesis (i.e., Chebyshev, Legendre, and Hermite polynomials) mainly differ from each

other in the coefficients of their monomials (see Table 3.1, in Chapter 3). Being utilized to process the input data vectors, these polynomials are perceived in this chapter as some transformation 'agents', whereby the different polynomials will be 'transforming' or 'processing' the input data vectors, using each one's own 'key'; i.e., coefficients, to another vector space, before applying a kernel function on the transformed vectors instead of the original input vectors.

6.2.1 Transforming the input data using Chebyshev polynomials

Recall that the vectorial processing approach (explained earlier in Section 3.4.1.2 and originally proposed by Ozer et al. [17]) constructs the Stage 1 and Stage 2 Chebyshev kernels for vector inputs in the form of:

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle T_n(\mathbf{x}), T_n(\mathbf{z}) \rangle \tag{6.1}$$

$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle \tag{6.2}$$

where $T(.)$ denotes the Chebyshev polynomials of order n and S_1 and S_2 denote the Stage 1 and Stage 2 kernels, respectively.

Framed by the data transformation perspective introduced in this chapter, it is important to realize that two things happen in the kernels constructed in the form of (6.1) and (6.2):

1- A certain family of orthogonal polynomials (in this case, it was the Chebyshev) of a certain order n is chosen to 'transform' or 'process' the input vector argument $\mathbf{x}$ to produce a new vector quantity denoted by $T(\mathbf{x})$. The same procedure is also done with the input vector $\mathbf{z}$ to produce the other vector quantity denoted by $T(\mathbf{z})$. In other words, $T(\mathbf{x})$ and $T(\mathbf{z})$ are perceived to be the 'transformed' image vectors of $\mathbf{x}$ and $\mathbf{z}$, respectively, in some new polynomial vector space (of the same dimensionality) embodying the image vectors that are the result of mapping the original vectors in the input space using a certain polynomial, which in this case is the Chebyshev.

2- And then, a 'valid Mercer kernel function' is chosen to calculate how similar $T(\mathbf{x})$ and $T(\mathbf{z})$ are to each other (since a valid Mercer kernel is considered a legitimate tool to measure similarity, as explained in Chapter 3). In this case, one can notice that this 'valid Mercer kernel function' just happened to be the 'Linear kernel'[15] (although it does not have to be), which is the standard dot-product taking place between the two

[15] Linear kernel: $k(\mathbf{x},\mathbf{z}) = \langle \mathbf{x},\mathbf{z} \rangle$

transformed vector quantities $T(\mathbf{x})$ and $T(\mathbf{z})$ in (6.1) and (6.2). Note that the shape characteristics of such a 'Chebyshev-linear' Stage 1 and Stage 2 kernel formulations is what have been studied before in Figure 4.1 (a) and Figure 4.2 (a), respectively, in Section 4.3.2.

As such, the utilization of the Chebyshev polynomials to construct SVM kernels using the vectorial approach illustrated in (6.1) and (6.2) is perceived in this chapter as a way of 'transforming' the input vectors (prior to the kernel calculation step) using a polynomial function (which, in this case, just happened to be the Chebyshev) to produce another set of vectors, upon which the kernel is to be afterwards evaluated.

Recall, however, that the core theory behind the use of kernel functions is to implicitly map the input space to a higher-dimensional feature space, so that overlapped classes can 'hopefully' become linearly separable. One can therefore notice that with the introduction of the additional transformation process via the Chebyshev polynomials, the overall kernel calculation process is becoming a three-fold operation:

1. First, the input vectors are explicitly processed by means of a certain polynomial function (which in this case are the Chebyshev polynomials $T(.)$ under study - although they do not have to be), to produce a new set of image vectors in some new polynomial vector space of the same dimensionality as the original input space.

2. The next step is then to apply a valid Mercer kernel function on the transformed image vectors rather than on the original input vectors directly. As shown in (6.1), this kernel function was chosen by the authors in [17] to be the linear (dot-product) kernel.

3. As explained before, the utilization of a valid Mercer kernel has also got the effect of mapping the 'polynomial-transformed' vectors to an implicit higher-dimensional feature space, where linear separability between overlapped classes can be improved.

However, although the linear kernel is indeed a valid Mercer kernel, it does not actually amount to any transformation to a higher number of dimensions in feature space, and hence it is more suitable for linearly separable classification tasks, which is not the usual case in many practical applications. Moreover, the linear kernel does not provide relatively accurate measures of similarity between its two input vector arguments, as exhibited by their shape characteristics discussed earlier in Figure 3.7 (Section 3.4.2.1).

To rectify these issues, this chapter proposes to use the Gaussian kernel instead. This is because of two reasons. The first reason is the renowned ability of the Gaussian kernel to

implicitly map the input space to an infinite-dimensional feature space, and, hence, it can improve the separability between classes when tackling non-linearly separable classification problems. The second reason is its shape characteristics, which are close to those of the ideal similarity function (as discussed earlier in Figure 3.6 – Section 3.4.2.1), and, hence, its calculated similarity measures would be a lot more accurate than those calculated by the linear kernel.

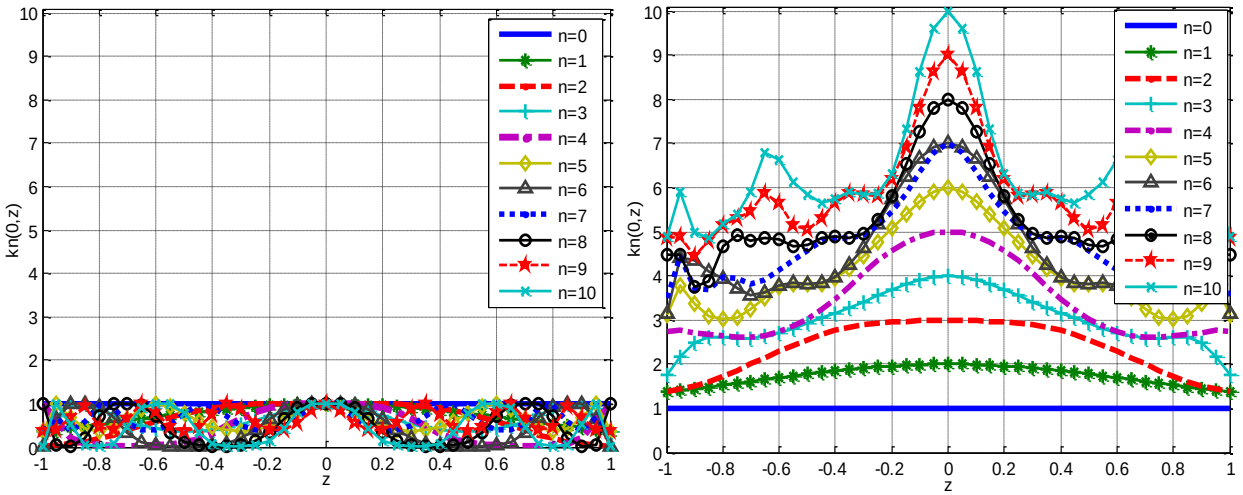
Consequently, the alternative versions of the (6.1) and (6.2) would then be calculated as:

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \exp\left(-\gamma\left\|T_n(\mathbf{x}) - T_n(\mathbf{z})\right\|^2\right) \tag{6.3}$$

$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma\left\|T_i(\mathbf{x}) - T_i(\mathbf{z})\right\|^2\right). \tag{6.4}$$

The similarity-based shape properties of the Stage 1 and Stage 2 linear kernel evaluation on the Chebyshev-processed inputs have been analysed before in Figure 4.1 (a) and Figure 4.2 (a), respectively, in Section 4.3.2. For a similar comparison, Figure 6.1 below investigates the behaviour of the Gaussian kernel evaluation on the Chebyshev-processed inputs for the same kernel stages, as per (6.3) and (6.4), for the first 10 polynomial orders, at $x = 0$, and where the Gaussian kernel parameter γ is set to 1, for simplicity.

Similar to the linear kernel evaluation on the Stage 1 and Stage 2 Chebyshev-processed inputs, one can observe (from the analysis of the kernel plots in Figure 6.1) that the summation fusion operation taking place in (6.4) is synergistic, as it enables the kernels to calculate better similarity measures by bringing their shape closer to the ideal similarity function model. One can also note that the odd orders of the Stage 1 Gaussian kernel evaluations (shown in Figure 6.1 (a)) do not fail anymore to contribute towards the calculation of the overall measure of similarity, as was the case with the Stage 1 linear kernel evaluations (shown in Figure 4.1 (a)), at $x = 0$. More importantly, however, is the fact that the Stage 2 Gaussian kernel evaluations on the Chebyshev-processed inputs (shown in Figure 6.1 (b)) have got oscillations with a lot less peak-to-peak amplitude than their counterparts evaluated using the linear kernel (shown in Figure 4.2 (a)), which is another reason why the utilization of the Gaussian kernel would enable the calculation of more accurate similarity measures, and, hence, it should lead to better classification performance.



(a) Stage 1 Chebyshev-processed-Gaussian kernel (b) Stage 2 Chebyshev-processed-Gaussian kernel

Figure 6.1 Comparison of the Stage 1 and Stage 2 Gaussian kernel evaluations on the Chebyshev-processed input data, for x=0.

6.2.2 Transforming the input data using Legendre polynomials

In a similar manner, one can choose any other polynomial function to process the input data instead of the Chebyshev polynomial, before the kernel calculation step. For example, one can choose the Legendre polynomials to process the input data vectors and then evaluate the similarity between them using the linear kernel, as we did before; in which case, the resulting Stage 1 and Stage 2 Legendre-processed-linear kernels can be defined as:

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle L_n(\mathbf{x}), L_n(\mathbf{z}) \rangle, \tag{6.5}$$

$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle L_i(\mathbf{x}), L_i(\mathbf{z}) \rangle, \tag{6.6}$$

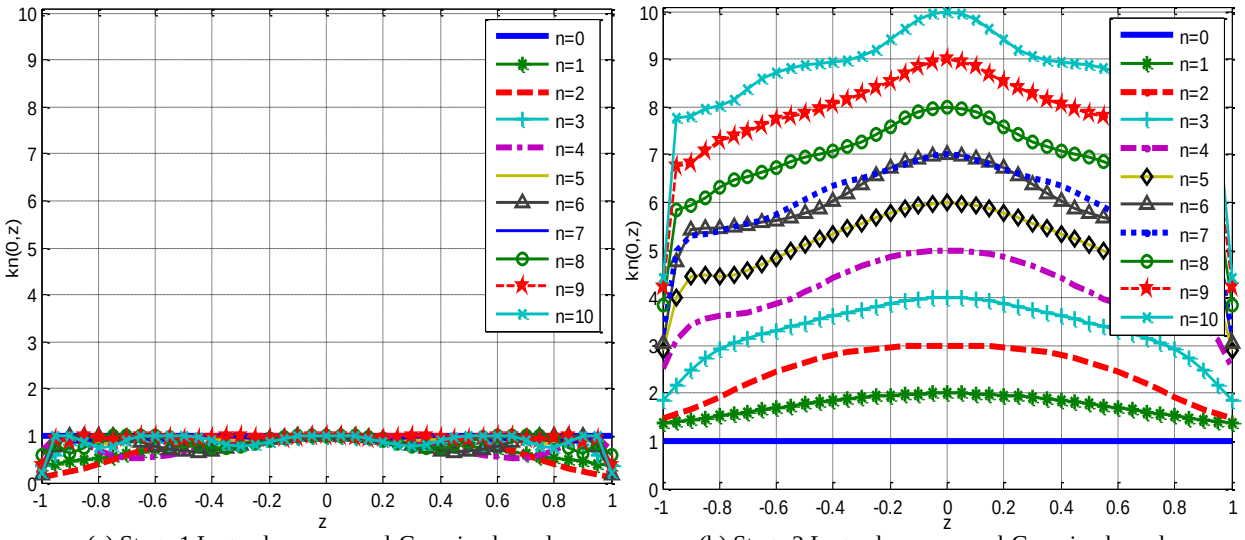
whose similarity-based shape and fusion characteristics were also analysed before in Figure 4.1 (b) and Figure 4.2 (b), in Section 4.3.2.

And again, if we choose to apply the Gaussian kernel, instead of the linear kernel, to calculate the similarity between the Legendre-processed inputs, the resulting Stage 1 and Stage 2 kernels would then be defined as:

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \exp\left(-\gamma \left\| L_n(\mathbf{x}) - L_n(\mathbf{z}) \right\|^2\right) \tag{6.7}$$

$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma \left\| L_i(\mathbf{x}) - L_i(\mathbf{z}) \right\|^2\right) \tag{6.8}$$

whose shape characteristics are also compared in Figure 6.2 below at x=0 and $\gamma = 1$. One can notice that the same observations derived from the shape characteristics of the Chebyshev-processed-Gaussian kernels (in the previous sub-section) also applies to the Legendre-processed-Gaussian kernels demonstrated in Figure 6.2, where in this case the oscillations are almost eliminated.



(a) Stage 1 Legendre-processed-Gaussian kernel (b) Stage 2 Legendre-processed-Gaussian kernel

Figure 6.2 Comparison of the Stage 1 and Stage 2 Gaussian kernel evaluations on the Legendre-processed input data, for x=0.

6.2.3 Transforming the input data using Hermite polynomials

To complete the investigation, this section shows how the input data can also be processed using the Hermite polynomials, and then evaluate the similarity measures using either the linear kernel, as per (6.9) and (6.10) for the Stage 1 and Stage 2 kernels respectively, or using the Gaussian kernel instead, as per (6.11) and (6.12) below.

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle H_n(\mathbf{x}), H_n(\mathbf{z}) \rangle \tag{6.9}$$

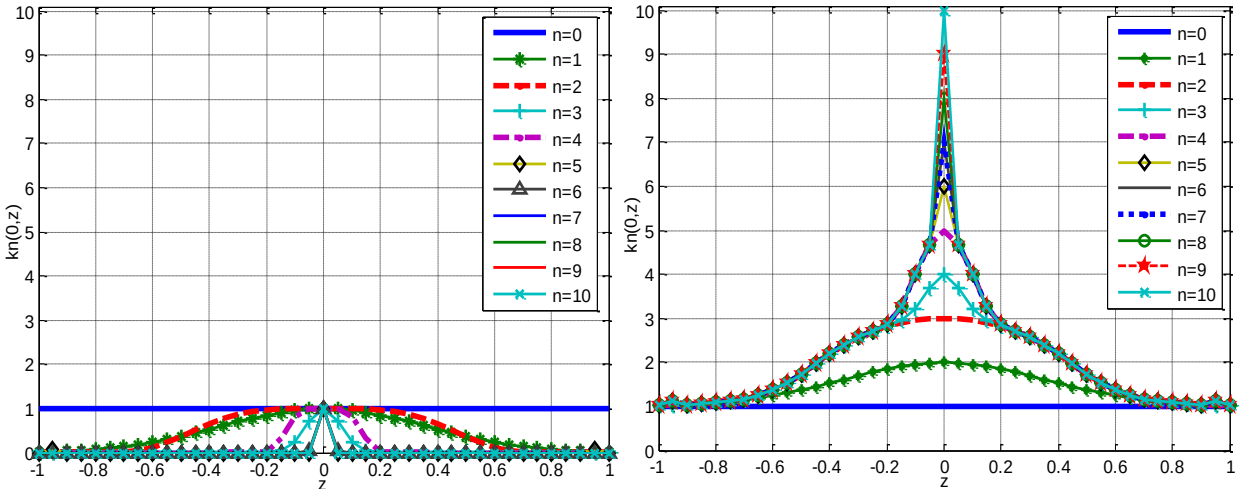
$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle H_i(\mathbf{x}), H_i(\mathbf{z}) \rangle \tag{6.10}$$

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \exp\left(-\gamma \left\| H_n(\mathbf{x}) - H_n(\mathbf{z}) \right\|^2\right) \tag{6.11}$$

$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma \left\| H_i(\mathbf{x}) - H_i(\mathbf{z}) \right\|^2\right) \tag{6.12}$$

Again, the similarity-based shape characteristics of (6.9) and (6.10) have been analysed before in Table 3.3, Figure 4.1 (c), and Figure 4.2 (c); whereas the characteristics of (6.11) and (6.12) can also be compared via Figure 6.3 below, at x=0 and $\gamma = 1$, as an example. One can notice that although the same observations derived from the Chebyshev- and

Legendre-processed-Gaussian kernels, illustrated in the previous two sub-sections, also apply to the Hermite-processed-Gaussian kernels herein, what is quite obvious is that the large peak-to-peak oscillations that used to exist in the linear kernel evaluations (as shown in Table 3.3) are completely eliminated with the Gaussian kernel evaluations, as shown in Figure 6.3 (b). Also, the quite elevated difference between the kernel values from one order to another disappears when the Gaussian kernel is applied, hence it was possible to plot all the first 10 orders of the Stage 1 and Stage 2 kernels together within the same figures, as shown in Figure 6.3.



(a) Stage 1 Hermite-processed-Gaussian kernel (b) Stage 2 Hermite-processed-Gaussian kernel

Figure 6.3 Comparison of the Stage 1 and Stage 2 Gaussian kernel evaluations on the Hermite-processed input data, for x=0.

6.2.4 Shape characteristics of the Stage 3 Gaussian kernel evaluation on the polynomial-transformed input vectors

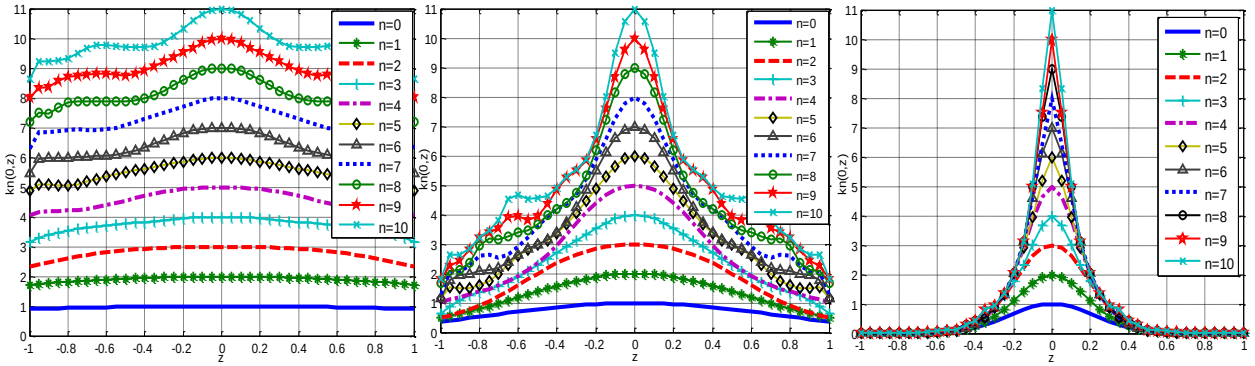
Similar to the investigation conducted on the shape characteristics of the Stage 3 linear kernel evaluation on the polynomial-transformed inputs in Section 4.3.3, this section will also investigate the effect of the fusion by multiplication operation of the Gaussian kernel evaluation on the polynomial-transformed vectors with another kernel. For consistency, the Gaussian kernel will also be multiplied by the Stage 2 Chebyshev, Legendre, and Hermite kernels, given in (6.4), (6.8), and (6.12), respectively, to achieve this purpose. As such, the resulting composite Stage 3 kernels studied in this section are defined as:

$$k_n^{S_3}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma\left\|T_i(\mathbf{x}) - T_i(\mathbf{z})\right\|^2\right) \times \exp\left(-\gamma\left\|\mathbf{x}-\mathbf{z}\right\|^2\right) \quad \text{for Chebyshev} \qquad (6.13)$$

$$k_n^{S_3}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma\left\|L_i(\mathbf{x}) - L_i(\mathbf{z})\right\|^2\right) \times \exp\left(-\gamma\left\|\mathbf{x}-\mathbf{z}\right\|^2\right) \quad \text{for Legendre} \qquad (6.14)$$

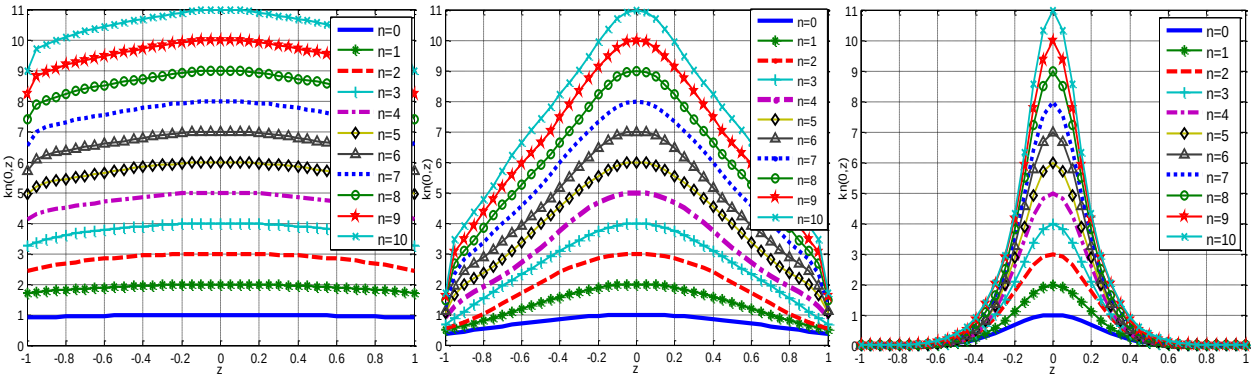
$$k_n^{S_3}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma\left\|H_i(\mathbf{x}) - H_i(\mathbf{z})\right\|^2\right) \times \exp\left(-\gamma\left\|\mathbf{x}-\mathbf{z}\right\|^2\right) \quad \text{for Hermite} \qquad (6.15)$$

Figures 6.4 – 6.6 demonstrate the shape characteristics of each of these composite Stage 3 polynomial kernels, respectively, at different values of the Gaussian kernel parameter γ. One can observe that the fusion by multiplication operation has also got the effect of getting the shape of the kernels closer to the ideal similarity measure function, and when an appropriate value for the Gaussian kernel parameter is selected, the polynomial oscillations can be completely eliminated.



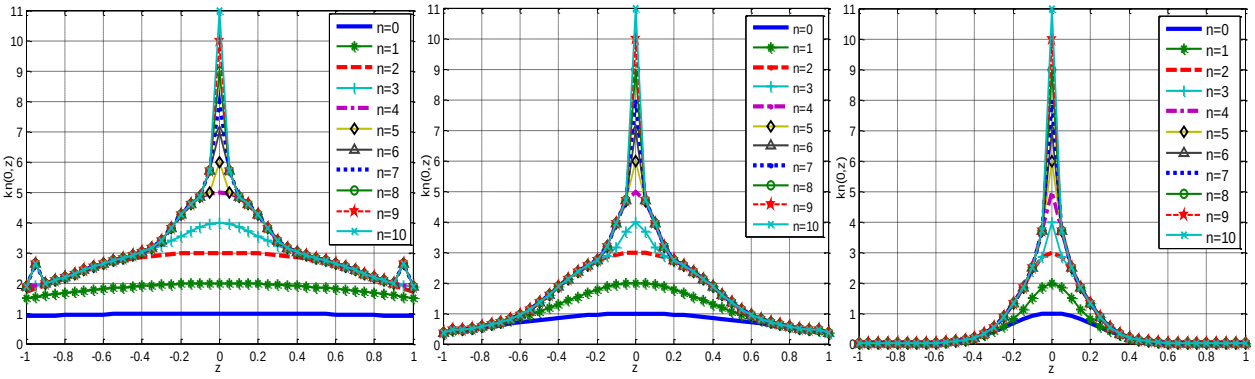
(a) $\gamma = 0.1$ (b) $\gamma = 1$ (c) $\gamma = 10$

Figure 6.4. Shape characteristics of the Stage 3 Gaussian kernel evaluation on the Chebyshev-processed input vectors, as per (6.13), at different values of γ and for x=0.



(a) $\gamma = 0.1$ (b) $\gamma = 1$ (c) $\gamma = 10$

Figure 6.5. Shape characteristics of the Stage 3 Gaussian kernel evaluation on the Legendre-processed input vectors, as per (6.14), at different values of γ and for x=0.



(a) $\gamma = 0.1$ (b) $\gamma = 1$ (c) $\gamma = 10$

Figure 6.6. Shape characteristics of the Stage 3 Gaussian kernel evaluation on the Hermite-processed input vectors, as per (6.15), at different values of γ and for x=0.

6.3 Advantages of the proposed transformation perspective of input data vectors

In the light of the polynomial-transformation approach discussed in the previous section, one can define the general construction of kernels from such polynomials to be a two-step process, where the first step is to process each data vector in the input space to produce another vector as dictated by the unique coefficients of the polynomial being applied. In other words, this step amounts to constructing some new 'polynomial space' of vectors (of the same dimensionality as the original input space), which contains the polynomial-processed image vectors of the original input space data. The second stage is then to apply whichever valid Mercer kernel function on the 'image vectors' in the newly created polynomial space, rather than the vectors in the original input space.

One question naturally arises as a result of introducing this additional transformation process: Why do we need to produce image versions of the input vectors (using whichever employed transformation function) and then apply the Mercer kernel afterwards, instead of just following the standard technique of applying the kernel directly to the original input vectors? In other words, what benefit is gained from doing the transformation step prior to the kernel calculation step? Does the transformation step help to better structure the data in a way that can facilitate the recognition of patterns amongst them and therefore enhance the classification performance?

Experimental results reported in the literature have showed that some advantages can be gained from the use of orthogonal polynomials to construct SVM kernels (e.g., superior classification accuracy, minimum data redundancy, low number of support vectors, less memory and execution time, etc.). However, there is very little, if any, theoretical justification as to why such experimental observations are happening. However, if one thinks of it from the data transformation perspective, introduced in this chapter, and from the definition of kernels as similarity measures, discussed in Chapter 3, one can realize that this transformation process is done in the 'hope' that the new vectors can better reveal how similar to each other their original vectors are in the original input space; and hence if the transformation function is successful to achieve this purpose, the resulting classification performance could be improved. This is in a sense similar to the notion of the implicit kernel mapping to the high-dimensional feature space, which is also performed in the 'hope' that the overlapped classes can become linearly separable, or linear separability between them can be improved.

It is, however, important to note that the Chebyshev, Legendre, and Hermite polynomials are not the only processing functions that can best be used to perform such a transformation process. Their promising empirical performance, however, encourages further investigations into this subject that can help define a 'measure of goodness' which can be used to appropriately select a good transformation function to the problem at hand, which can better reveal the (possibly) hidden similarity between the input data vectors. For example, with the three polynomial families studied in this thesis (i.e., the Chebyshev, Legendre, and Hermite polynomials) it has been shown that this measure of goodness could be the accuracy of the similarity measures calculated by their corresponding kernels as exhibited by their pictorial characteristics.

6.4 Construction of the transformed polynomial vector space

6.4.1 Previously proposed processing approaches of SVM orthogonal polynomial kernels (re-visited)

On re-analysing the previously proposed approaches, by which orthogonal polynomials process input data vectors (explained back in Section 3.4.1) to construct kernels for SVM pattern classification, one can notice that the 'pairwise' approach, initially proposed by Ye et al. [31] using the Chebyshev polynomials, does not actually fit in very well within the transformation framework presented in this chapter. This is because it does not produce transformed image 'vectors' of the input data, as explained in Section 6.2. Instead, it applies the polynomial to each corresponding feature pair and then multiplies the result to calculate the overall value of the kernel.

Ozer et al. [17] have also explained why this multiplication process can be problematic, and to avoid it, they proposed the 'vectorial' approach, where they applied the Chebyshev polynomials to the input vector as a whole rather than to its individual features, as previously illustrated in Table 3.2 (Section 3.4.1.2). They then decided to adopt the linear kernel as a tool to measure the similarity between the Chebyshev-processed quantities $T(\mathbf{x})$ and $T(\mathbf{z})$, and evaluated their Stage 2 Chebyshev kernel for vector inputs in the form of:

$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle , \tag{6.16}$$

where the 6th order Chebyshev-processed linear kernel of such a vectorial formulation is shown in Figure 6.7 below, as an example.

$$k(\mathbf{x},\mathbf{z}) = 1 + c + (2a\text{-}1)(2b\text{-}1) + c\,(4a\text{-}3)(4b\text{-}3) + (8a^2\text{-}8a\text{+}1)(8b^2\text{-}8b\text{+}1) + c\,(16a^2\text{-}20a\text{+}5)(16b^2\text{-}20b\text{+}5) + (32a^3\text{-}48a^2\text{+}18a\text{-}1)(32b^3\text{-}48b^2\text{+}18b\text{-}1)$$

Figure 6.7 Evaluation of the 6th order Stage 2 Chebyshev-processed linear kernel using the 'vectorial' approach proposed by Ozer et al. [17], where a=<**x**,**x**>, b=<**z**,**z**>, and c=<**x**,**z**>.

However, on analysing closely the first few orders of the Chebyshev-processed quantities shown in Table 3.2, one can notice that although the odd orders of the vectorial processing approach yielded processed-image 'vectors' of the input vectors, yet the even orders did not, as they ended up being evaluated as scalar quantities rather than vectors. In other words, when the even order polynomials are applied, the processing operation drops down the input vector to a scalar quantity calculated from the term <**x**,**x**>, which corresponds to the length (or the square of the norm) of vector **x**. As such, only partial information about the input vectors is conveyed to the classifier by the even-order polynomials, because vector orientation is discarded. Furthermore, during the experiments, each vector component is scaled to the normalized region [-1, +1] (see Section 4.4.3) which may distort class-specific vector length information and, hence, might reduce the reliability of using vector lengths to represent the input vectors.

Another drawback, that can also be realized from the odd order terms of the kernels utilizing this vectorial approach, is the amalgamation of **x** and **z** into the dot-product c=<**x**,**z**>, as demonstrated for example by the 6th order kernel shown in Figure 6.7. One can notice that this operation was mathematically possible to be achieved due to the common factor operation that was performed back in Table 3.2, which was also only possible due to the fact that the odd orders of the Chebyshev polynomials consist of only odd monomials (and similarly, the even orders consist of only even monomials). Although some other orthogonal polynomials (e.g., the Legendre or Hermite polynomials) share this same characteristic, other polynomials (e.g., the Laguerre polynomials) do not; a fact which renders this vectorial approach inapplicable and not mathematically feasible to be generalized to any orthogonal polynomial family.

One can also notice that the Stage 2 kernel of the vectorial approach in (6.16) is evaluated in the form of:

$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} k_i(\mathbf{x},\mathbf{z})$$
$$= k_0(\mathbf{x},\mathbf{z}) + k_1(\mathbf{x},\mathbf{z}) + ... + k_n(\mathbf{x},\mathbf{z}), \quad (6.17)$$

where the k_i's are the linear kernels evaluated on the Chebyshev-processed quantities. Based on the closure property that says that the addition of valid kernels also produces a valid kernel, this means that each of the k_i's in (6.17) is also a valid kernel, and, hence, is a legitimate tool to measure similarity. As explained earlier, however, the utilization of the linear kernel is not a good choice, because it is neither a good tool to measure the similarity between the Chebyshev-processed quantities $T(\mathbf{x})$ and $T(\mathbf{z})$, nor does it trigger a transformation to a higher number of dimensions so that the classifier can tackle non-linearly separable classification problems. That is why this chapter investigates the proposition of applying the Gaussian kernel instead, as explained earlier.

6.4.2 Customizing a solution via vector concatenation

In order to come up with a polynomial-processing approach that really fits well within the transformation perspective presented in this chapter, and at the same time rectify the pitfalls identified in the pairwise and the vectorial approaches, this section proposes a new processing methodology, which ensures that when any polynomial function is utilized to process the input data vectors, the result will always be a new vector and, hence, maintains the attributes embedded in the structure of the original vectors. To achieve this purpose, the transformation process is performed as follows.

Consider the two m-dimensional input vectors $\mathbf{x} = \{x_1, x_2, \cdots, x_m\}$ and $\mathbf{z} = \{z_1, z_2, \cdots, z_m\}$. Similar to the 'pairwise' approach, in this new processing methodology, the i^{th} order polynomial will also be utilized to process the individual scalar features of the input vectors. Unlike the 'pairwise' approach, however, this methodology will not be applied in a pairwise fashion and multiply the results. Instead, the evaluations of the polynomial will be restricted to the features of each of the two input vectors separately, and then the individual results are 'concatenated' to form two new vectors, denoted by $\mathbf{C}_i(\mathbf{x})$ and $\mathbf{C}_i(\mathbf{z})$, where $\mathbf{C}_i(\mathbf{x})$ is the transformed image vector of $\mathbf{x}$ given by:

$$\mathbf{C}_i(\mathbf{x}) = \left[P_i(x_1), P_i(x_2), \cdots, P_i(x_m)\right], \quad (6.18)$$

and similarly, $\mathbf{C}_i(\mathbf{z})$ is the transformed image vector of $\mathbf{z}$ given by:

$$\mathbf{C}_i(\mathbf{z}) = \left[P_i(z_1), P_i(z_2), \cdots, P_i(z_m)\right], \text{ and} \tag{6.19}$$

$P_i(\cdot)$ is the chosen i^{th} order polynomial (e.g., Chebyshev, Legendre, etc.).

This process is represented pictorially in Figure 6.8 below for two 4-dimensional input vectors $\mathbf{x}_1$ and $\mathbf{x}_2$, as an example. Due to the nature of concatenating the processed features into the two new vectors $\mathbf{C}_i(\mathbf{x})$ and $\mathbf{C}_i(\mathbf{z})$, this approach is referred to in this thesis as the '*concatenated*' processing approach.

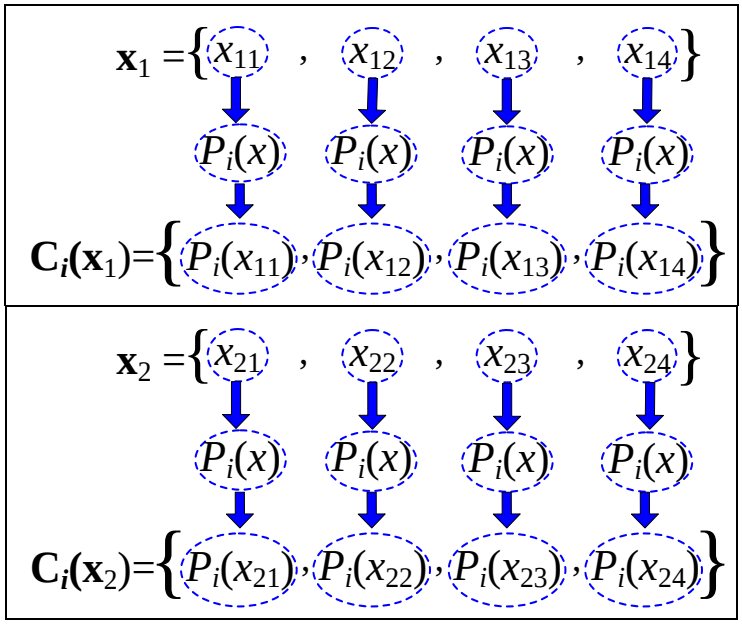


Figure 6.8 An illustrative example of two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$, four features each, to pictorially clarify the mathematical procedure of the proposed concatenated approach on multi-dimensional input data vectors.

Unlike the vectorial approach, the concatenated approach, which is illustrated in Figure 6.8, clearly shows that the resulting processed quantities $\mathbf{C}_i(\mathbf{x}_1)$ and $\mathbf{C}_i(\mathbf{x}_2)$ are always formulated in a vector form for any polynomial order applied, whether odd or even. As such, every input vector will always be represented by another image vector transformed in some new polynomial vector space of the same dimensionality as the original input space. Furthermore, the implementation of such a concatenated approach is also not restricted to a certain polynomial family (as was the case with the vectorial approach); i.e., any polynomial can be used regardless of its monomial order combination, whether it is only odd, only even, or a combination of both. As such, a comparative analysis can be effectively conducted amongst various polynomial functions to investigate which one would be the best achieving polynomial for a given application. This is explored in the experiments conducted in the next section on the Chebyshev, Legendre, and Hermite

polynomials, as an example, by comparing the resulting classification accuracy scored by each.

As explained earlier, the next step is then to apply whichever valid Mercer kernel on these two transformed vectors $\mathbf{C}_i(\mathbf{x})$ and $\mathbf{C}_i(\mathbf{z})$. So, if one decides to adopt the linear kernel to calculate the similarity between these two transformed image vectors, the Stage 1 (without sum) and Stage 2 (with sum) kernels would then be formulated as:

- Stage 1 – linear kernel – concatenated:

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = <\mathbf{C}_n(\mathbf{x}), \mathbf{C}_n(\mathbf{z})> \tag{6.20}$$

- Stage 2 – linear kernel – concatenated:

$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} <\mathbf{C}_i(\mathbf{x}), \mathbf{C}_i(\mathbf{z})> \tag{6.21}$$

However, as mentioned earlier, one can also choose to apply any other kernel. For example, if we apply the Gaussian kernel instead, the Stage 1 and Stage 2 kernels would then be formulated as:

- Stage 1 – Gaussian kernel – concatenated:

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \exp\left(-\gamma \left\|\mathbf{C}_n(\mathbf{x}) - \mathbf{C}_n(\mathbf{z})\right\|^2\right) \tag{6.22}$$

- Stage 2 – Gaussian kernel – concatenated:

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma \left\|\mathbf{C}_i(\mathbf{x}) - \mathbf{C}_i(\mathbf{z})\right\|^2\right) \tag{6.23}$$

6.5 Experimental evaluation of the proposed concatenated approach

6.5.1 Polynomial kernel functions under test

The set of experiments conducted in this section explore how the proposed concatenated processing approach performs, compared to the other existing methods, in terms of the classification accuracy. However, given that the superiority of the vectorial (over the pairwise) approach has already been demonstrated through the work of Ozer et al. [17], the comparative experiments conducted herein will only be performed between the concatenated approach and its vectorial counterpart.

Furthermore, to highlight the effectiveness of the proposed concatenated approach, the experiments are conducted on three different polynomials: the Chebyshev, Legendre, and

Hermite polynomials; as opposed to the work of Ye et al. [31] and Ozer et al. [17], who both used only the Chebyshev polynomials to demonstrate the performance of their pairwise and vectorial approaches, respectively. Furthermore, for a comprehensive evaluation of the proposed concatenated approach, a separate set of experiments is conducted for every stage of the kernels produced from the similarity fusion framework presented earlier in Chapter 4 using each of these chosen polynomials (i.e., Chebyshev, Legendre, and Hermite). As such, Sections 6.5.2.1, 6.5.2.2, and 6.5.2.3 demonstrate the comparative experimental results achieved from the Stage 1, Stage 2, and Stage 3 kernels, respectively, using the same experimental setup and datasets explained earlier in Chapter 3.

6.5.2 Experimental results and discussions

6.5.2.1 Comparative experimental results on Stage 1 kernels

This section presents the comparative classification accuracy results scored by the Stage 1 kernels that are evaluated on the Chebyshev-, Legendre-, and Hermite-processed image vectors using the new proposed concatenated approach versus its vectorial counterpart. Furthermore, the experiments also aim to explore how the Gaussian kernel performs, when evaluated on the polynomial-processed inputs, instead of the linear kernel, as explained in Section 6.2. To achieve this purpose, two sets of experiments have been developed in this section to evaluate each of these propositions separately. To be clear, the first set of experiments evaluates the empirical performance when the linear kernel is evaluated on both the vectorial- and concatenated-processed inputs, using the Chebyshev, Legendre, and Hermite polynomials under study, as depicted in Table 6.1. The second set of experiments, on the other hand, utilizes the evaluation of the Gaussian kernel on the concatenated-processed vectors compared to the linear kernel evaluations on the vectorial-processed inputs, as depicted in Table 6.2.

Table 6.1. Mathematical formulations of the Stage 1 linear kernel evaluation on the vectorial- and concatenated inputs using the Chebyshev, Legendre, and Hermite polynomials.

	Vectorial-linear kernel	**Concatenated-linear kernel**
Chebyshev	$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle T_n(\mathbf{x}), T_n(\mathbf{z}) \rangle$	$k_n^{S_1}(\mathbf{x},\mathbf{z}) =< [T_n(x_1),\cdots,T_n(x_m)],[T_n(z_1),\cdots,T_n(z_m)] >$
Legendre	$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle L_n(\mathbf{x}), L_n(\mathbf{z}) \rangle$	$k_n^{S_1}(\mathbf{x},\mathbf{z}) =< [L_n(x_1),\cdots,L_n(x_m)],[L_n(z_1),\cdots,L_n(z_m)] >$
Hermite	$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle H_n(\mathbf{x}), H_n(\mathbf{z}) \rangle$	$k_n^{S_1}(\mathbf{x},\mathbf{z}) =< [H_n(x_1),\cdots,H_n(x_m)],[H_n(z_1),\cdots,H_n(z_m)] >$

Table 6.2. Mathematical formulations of the Stage 1 Gaussian kernel evaluation on the concatenated vectors versus the linear kernel evaluation on the vectorial-processed inputs using the Chebyshev, Legendre, and Hermite polynomials.

	Vectorial-linear kernel	Concatenated-Gaussian kernel
Chebyshev	$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle T_n(\mathbf{x}), T_n(\mathbf{z}) \rangle$	$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \exp\left(-\gamma \left\| [T_n(x_1),\cdots,T_n(x_m)] - [T_n(z_1),\cdots,T_n(z_m)] \right\|^2\right)$
Legendre	$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle L_n(\mathbf{x}), L_n(\mathbf{z}) \rangle$	$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \exp\left(-\gamma \left\| [L_n(x_1),\cdots,L_n(x_m)] - [L_n(z_1),\cdots,L_n(z_m)] \right\|^2\right)$
Hermite	$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle H_n(\mathbf{x}), H_n(\mathbf{z}) \rangle$	$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \exp\left(-\gamma \left\| [H_n(x_1),\cdots,H_n(x_m)] - [H_n(z_1),\cdots,H_n(z_m)] \right\|^2\right)$

Recall that although the Stage 1 kernels have been shown in Chapter 4 to demonstrate a relatively low classification performance compared to their Stage 2 counterparts, due to their weak acquisition of similarity measures, utilizing them to formulate a comparison between the concatenated and vectorial approaches will demonstrate a straightforward assessment of how the proposed concatenated approach performs in terms acquiring more accurate similarity measures and accordingly an enhanced classification performance.

The figures shown in Table 6.3 demonstrate the experimental results in a graphical format (in terms of the best scored classification accuracy) obtained by the first 20 orders of the Stage 1 linear kernels constructed using the concatenated and vectorial approaches. As shown, the implementation of the proposed concatenated approach has had a different influence on the classification performance of the kernels under test amongst the datasets investigated depending on each dataset's own characteristics. However, on average, one can still clearly observe that the kernels constructed using the concatenated approach have demonstrated a quite superior performance over their vectorial counterparts for most of the datasets and the polynomial orders tested.

As can be observed, some datasets have shown a relatively huge improvement in the classification accuracy introduced by the concatenated approach than other datasets. For example, in most of the results, such as the Breast Cancer, Image Segmentation, and Thyroid datasets, the proposed concatenated approach has shown to significantly outperform its vectorial counterpart, especially for higher polynomial orders. In the Iris and Two Spirals datasets, on the other hand, the concatenated approach has shown a relatively smaller overall improvement over the vectorial approach, compared to the significant improvements observed in the other datasets. This observation can also be realized by examining the bar chart comparisons, shown in Figure 6.9, which demonstrate the average accuracy scored by the first 20 orders of the concatenated and the vectorial approaches, for

each of the datasets experimented. As can be observed, the concatenated approach demonstrated a consistent superior average accuracy over its vectorial counterpart for most of the datasets experimented, with only an infinitesimal improvement in the Iris and Two Spirals datasets.

To enable a crisper quantitative evaluation of how much improvement in classification accuracy can the concatenated approach achieve over its vectorial counterpart for these Stage 1 kernels, Table 6.4 has also been used to calculate the average improvement in classification accuracy for the first 20 orders experimented. As shown at the bottom of the table, the average accuracy improvement factors (in %) introduced by the concatenated approach for the Stage 1- Chebyshev, Legendre, and Hermite kernels was calculated to be equal to 26.857%, 27.212% and 17.649%, respectively; making up a total average accuracy improvement factor of 23.906% depending on the kernel and the dataset used; which clearly demonstrates the outperformance of the proposed concatenated approach over its vectorial counterpart.

Of particular importance, however, is the observation that some polynomial orders of the vectorial-linear kernels in some datasets are still unable to recognize any examples at all during the testing phase, as shown for example in the experimental results on the Breast Cancer and Ionosphere datasets, illustrated in Table 6.3, and discussed before in Section 4.4.4. On examining closely the results at the orders tested, one can notice that it is mostly the even orders that suffer from this recognition inability. Recall that one of the drawbacks of the vectorial approach is that when the even orders process the input vectors as a whole, they produce scalar quantities rather than vectors, and hence only partial information about the input vectors is conveyed to the classifier, as vector orientation is discarded (see the theoretical discussion/justification in Section 6.4.1). Such defect in the even-order vectorial kernels can therefore explain the reason behind their classification inability observed in the results herein.

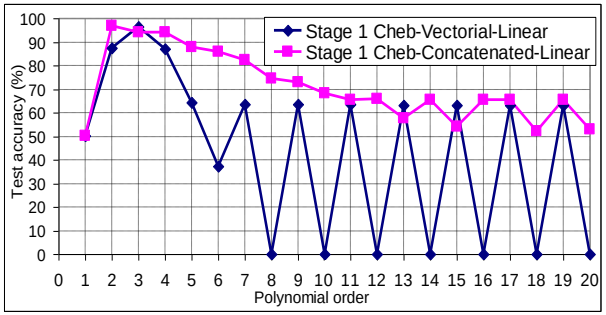
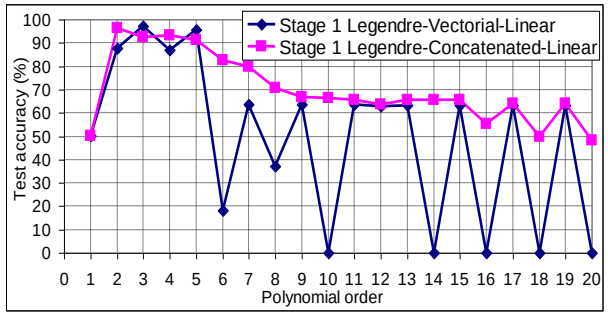
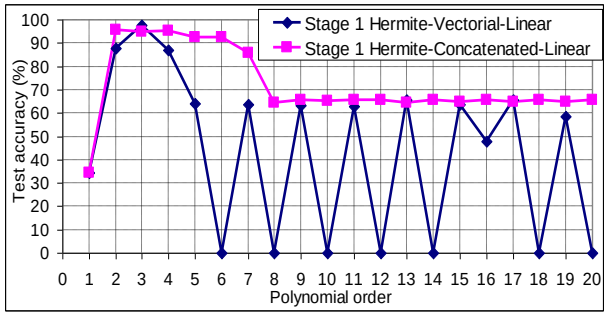
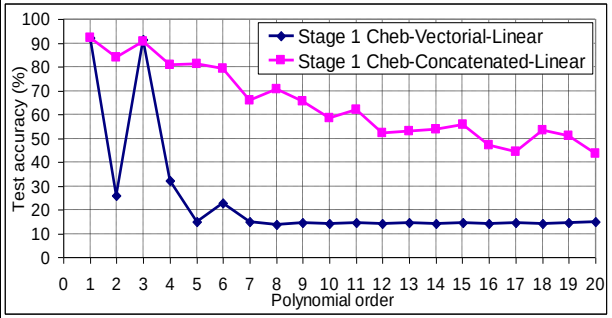
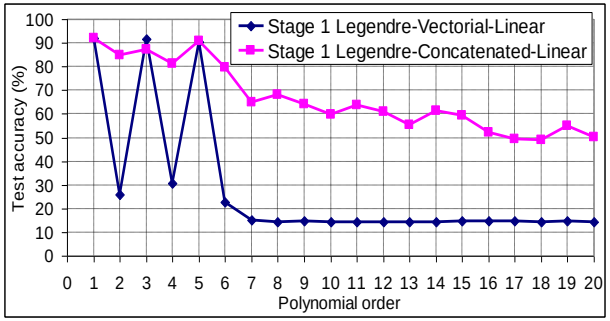
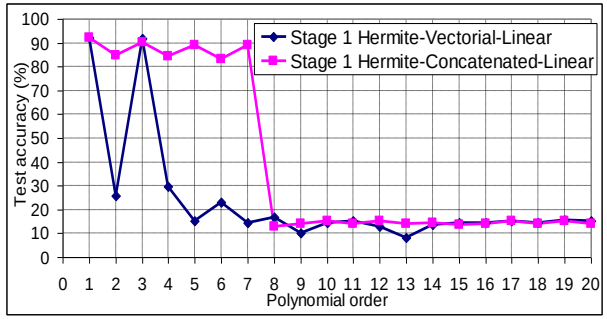
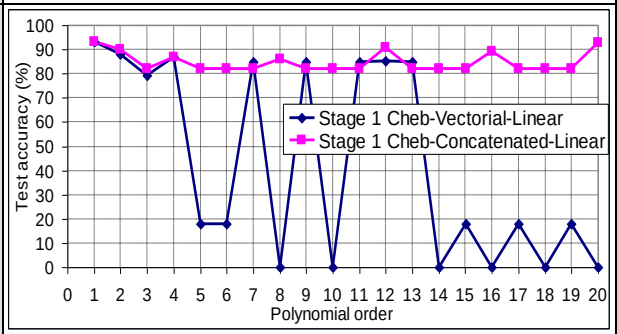
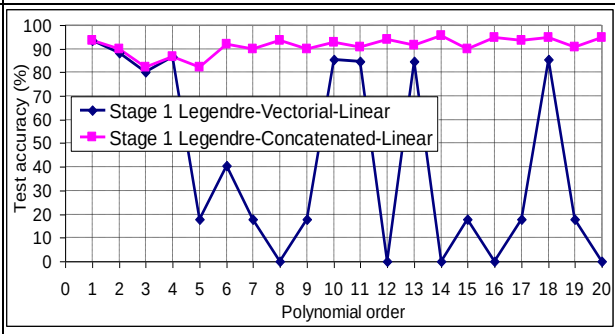
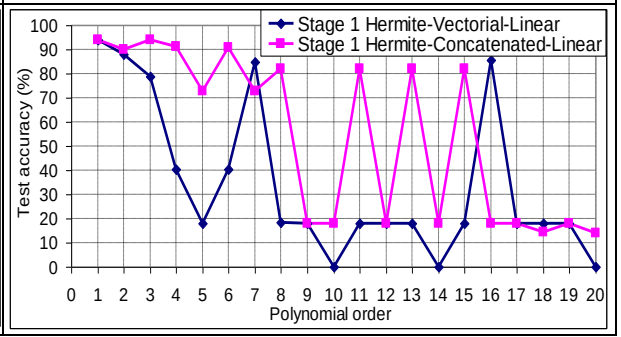
On the other hand, the transformation of the input data vectors using the proposed concatenated approach does not suffer from this defect, as it is intentionally designed in a way that ensures that when any polynomial with any order is utilized to process a certain input vector, it will always produce an image vector transformed into the newly constructed polynomial vector space. This way, whichever kernel we then choose to apply, will always be guaranteed to be getting the correct input vector arguments and, hence, should be able to calculate more accurate similarity measures and produce better classification performance than the kernels applied to the quantities processed using the

vectorial approach, as shown in the results illustrated in Table 6.3, not only for the even orders, but also for most of the odd orders too.

Consequently, and as can be clearly observed from most of the results illustrated in Table 6.3, the concatenated approach has therefore demonstrated a lot more steady and consistent classification accuracy across the spectrum of polynomial orders, with a significant reduction in the abrupt oscillating behaviour between the even and odd orders, as is the case with the vectorial approach. To quantify this observation, Table 6.4 has also been used to calculate the standard deviation of the average improvement factors (shown in brackets), as well as the minimum and maximum improvement factors (shown in bold), introduced by the proposed concatenated approach.

As it can be observed, the standard deviation of the average improvement factors can be quite large relative to the mean, as shown for example in the Breast Cancer, Ionosphere, and Iris datasets, due to the huge difference between the two sets of results across the polynomial orders. Conversely, however, one can also realize the steadiness of the results by examining the standard deviation of the accuracy values themselves, reported in the statistical analysis tests tabulated in Appendix B.4, where not only the mean of the concatenated kernels is consistently larger than the mean of the vectorial kernels, but also the standard deviation of the classification accuracies scored by the concatenated kernels is a lot smaller than the mean, compared to the standard deviation calculated for the vectorial kernels, for most of the results obtained.

Table 6.3 Comparative experimental results of Stage 1 concatenated-linear and vectorial-linear kernels.

	Stage 1 Chebyshev kernels	Stage 1 Legendre kernels	Stage 1 Hermite kernels
Breast Cancer			
Image Segmentation			
Ionosphere			

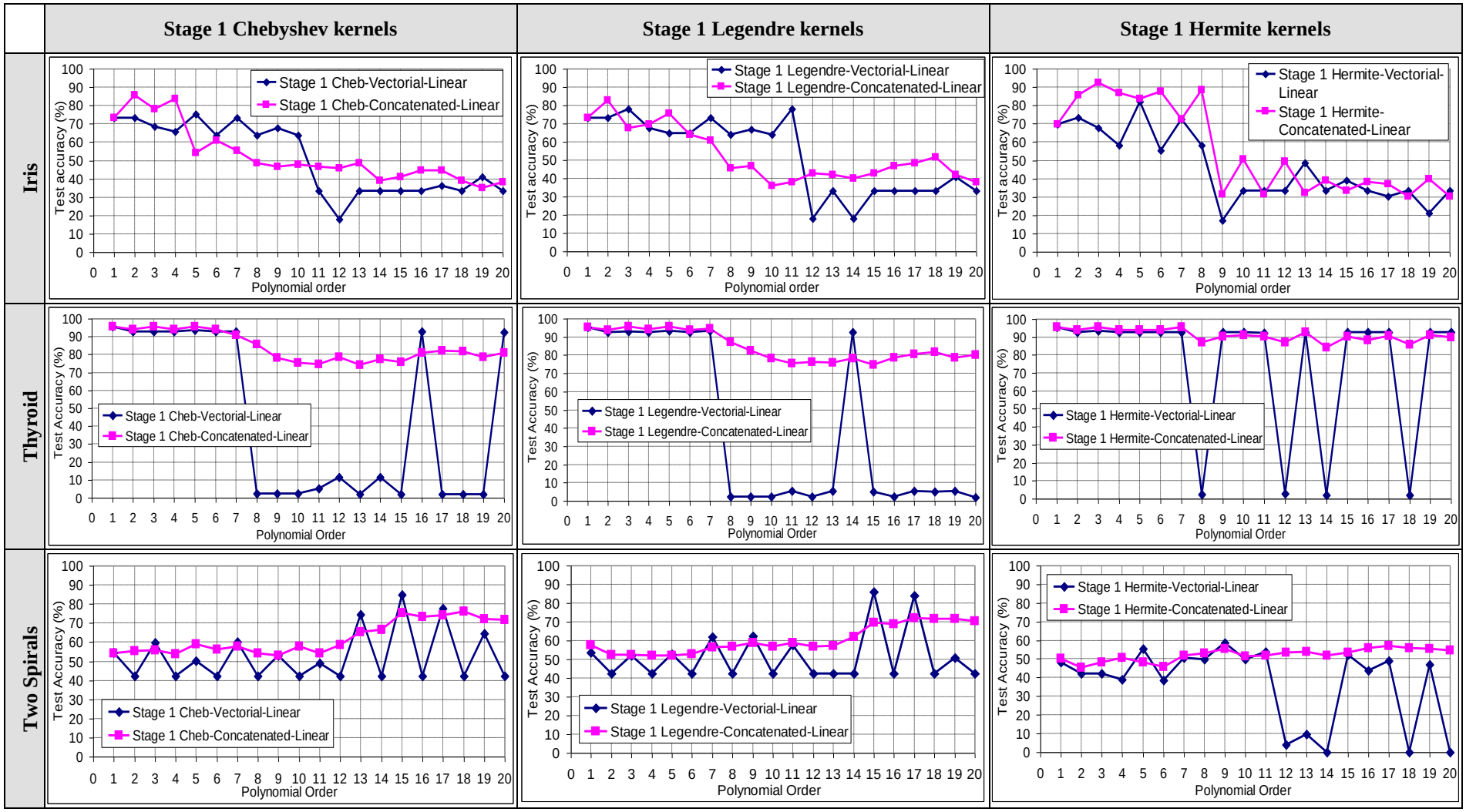

Stage 1 Chebyshev kernels
Stage 1 Legendre kernels
Stage 1 Hermite kernels
Iris
Thyroid
Two Spirals
Test accuracy (%)
Test Accuracy (%)
Polynomial order
Polynomial Order
Stage 1 Cheb-Vectorial-Linear
Stage 1 Cheb-Concatenated-Linear
Stage 1 Legendre-Vectorial-Linear
Stage 1 Legendre-Concatenated-Linear
Stage 1 Hermite-Vectorial-Linear
Stage 1 Hermite-Concatenated-Linear

Table 6.4 Quantitative assessment of the average improvements in the scored classification accuracy of Stage 1 concatenated-linear and vectorial-linear kernels.

		Stage 1 Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	50.320	87.633	96.588	86.994	64.179	37.100	63.539	0	63.539	0	63.539	0	63.326	0	63.326	0	63.326	0	63.326	0
	N2	50.320	96.802	94.030	94.030	87.846	85.714	82.516	74.414	73.134	68.230	65.458	65.885	57.783	65.458	54.158	65.458	65.458	52.026	65.458	52.878
	I	0	9.168	-2.559	7.036	23.667	48.614	18.977	**74.414**	9.595	68.230	1.919	65.885	-5.544	65.458	**-9.168**	65.458	2.132	52.026	2.132	52.878
	AI	27.516 (30.006)																			
Image Segmentation	N1	92.143	25.714	91.333	32.095	14.904	22.857	15.000	13.857	14.619	14.286	14.523	14.286	14.476	14.286	14.619	14.286	14.619	14.286	14.619	15.048
	N2	92.143	84.095	90.667	80.667	81.143	79.048	66.000	70.667	65.476	58.524	62.095	52.190	52.810	53.714	55.762	47.095	44.238	53.476	50.857	43.429
	I	0	58.381	**-0.667**	48.571	**66.238**	56.190	51	56.810	50.857	44.238	47.571	37.905	38.333	39.429	41.143	32.81	29.619	39.190	36.238	28.381
	AI	40.112 (17.06)																			

		Stage 1 Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	50.320	87.633	97.228	86.994	95.522	18.337	63.539	37.100	63.539	0	63.539	62.900	63.326	0	63.326	0	63.326	0	63.326	0
	N2	50.320	96.375	92.324	93.390	91.471	82.516	79.957	70.789	66.951	66.311	65.458	63.539	65.458	65.458	65.458	55.437	64.000	49.893	64.000	48.401
	I	0	8.742	**-4.904**	6.397	-4.051	64.179	16.418	33.689	3.412	**66.311**	1.919	0.640	2.132	65.458	2.132	55.437	0.674	49.893	0.674	48.401
	AI	20.878 (26.7312)																			
Image Segmentation	N1	92.143	25.762	91.762	30.524	90.524	22.857	15.000	14.286	14.619	14.286	14.524	14.286	14.476	14.286	14.619	14.619	14.619	14.286	14.619	14.286
	N2	92.143	84.667	87.286	81.143	91.000	79.524	64.952	68.000	64.095	59.667	63.714	61.048	55.476	61.190	59.429	52.000	49.524	48.952	55.143	50.190
	I	0	**58.905**	**-4.476**	50.619	0.4762	56.667	49.952	53.714	49.476	45.381	49.190	46.762	41	46.905	44.81	37.381	34.905	34.667	40.524	35.905
	AI	38.638 (18.549)																			

		Stage 1 Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	34.328	87.633	97.655	86.994	63.966	0	63.539	0	63.326	0	62.900	0	65.458	0	63.753	47.974	65.458	0	58.635	0
	N2	34.329	95.736	94.883	95.096	92.324	92.537	85.714	64.392	65.458	65.245	65.458	65.458	64.606	65.458	64.819	65.458	64.819	65.458	64.819	65.458
	I	0	8.102	**-2.772**	8.102	28.358	**92.537**	22.175	64.392	2.132	65.245	2.559	65.458	-0.853	65.458	1.066	17.484	-0.640	65.458	6.183	65.458
	AI	28.795 (31.883)																			
Image Segmentation	N1	92.190	25.762	91.762	29.667	15.381	22.857	14.619	16.952	10.143	14.286	15.143	13.000	8.238	13.714	14.381	14.286	15.190	14.286	15.810	15.143
	N2	92.190	84.810	90.190	84.286	88.952	83.190	89.095	12.810	14.095	15.048	14.143	15.381	14.143	14.429	13.857	14.190	15.048	14.238	15.286	14.238
	I	0	59.048	-1.571	54.619	73.571	60.333	**74.476**	**-4.143**	3.952	0.762	-1	2.381	5.905	0.714	-0.524	-0.095	-0.143	-0.048	-0.524	-0.905
	AI	16.340 (28.845)																			

		Stage 1 Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ionosphere	N1	93.378	88.079	79.470	86.755	17.881	17.881	84.768	0	84.768	0	84.768	85.430	84.768	0	17.881	0	17.881	0	17.881	0
	N2	93.377	90.066	82.119	86.755	82.119	82.119	82.119	86.093	82.119	82.119	82.119	90.728	82.119	82.112	82.119	89.404	82.119	82.119	82.119	92.715
	I	0	1.987	2.649	0	64.238	64.238	**-2.649**	86.093	-2.649	82.119	-2.649	5.298	-2.649	82.119	64.238	89.404	64.238	82.119	64.234	**92.715**
	AI	**41.755 (39.777)**																			
Iris	N1	73.333	73.333	68.571	65.714	75.238	63.81	73.333	63.81	67.619	63.810	33.333	18.095	33.333	33.333	33.333	33.333	36.190	33.333	40.952	33.333
	N2	73.333	85.714	78.095	83.810	54.286	60.952	55.238	48.571	46.667	47.619	46.667	45.714	48.571	39.048	40.952	44.762	44.762	39.048	35.238	38.095
	I	0	12.381	9.524	18.095	**-20.952**	-2.857	-18.095	-15.238	-20.952	-16.190	13.333	**27.619**	15.238	5.714	7.619	11.429	8.571	5.714	-5.714	4.762
	AI	**2.000 (24.298)**																			

		Stage 1 Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ionosphere	N1	93.377	88.079	80.132	86.755	17.881	40.397	17.881	0	17.881	85.430	84.768	0	84.768	0	17.881	0	17.881	85.430	17.881	0
	N2	93.377	90.066	82.119	86.755	82.119	92.053	90.066	93.377	90.066	92.715	90.728	94.040	91.391	95.364	90.066	94.702	93.377	94.702	90.728	94.702
	I	0	1.987	1.987	**0**	64.238	51.656	72.185	93.377	72.185	7.285	5.960	94.040	6.623	**95.364**	72.185	94.702	75.497	9.272	72.848	94.702
	AI	**49.305 (39.493)**																			
Iris	N1	73.333	73.333	78.095	67.619	64.762	64.762	73.333	63.810	66.667	63.810	78.095	18.095	33.333	18.095	33.333	33.333	33.333	33.333	40.952	33.333
	N2	73.333	82.857	67.619	69.524	75.500	63.810	60.952	45.714	46.667	36.190	38.095	42.857	41.905	40.000	42.857	46.667	48.571	51.429	41.905	38.095
	I	0	9.524	-10.476	1.905	10.738	-0.952	-12.381	-18.095	-20.000	-27.619	**-40.000**	**24.762**	8.571	21.905	9.524	13.333	15.238	18.095	0.952	4.762
	AI	**0.489 (17.133)**																			

		Stage 1 Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ionosphere	N1	94.040	88.079	78.808	40.397	17.881	40.397	84.768	18.543	17.881	0	17.881	17.881	17.881	0	17.881	85.430	17.881	17.881	17.881	0
	N2	94.040	90.066	94.040	91.391	72.848	90.728	72.848	82.119	17.881	17.888	82.119	17.881	82.119	17.881	82.119	17.881	17.881	14.570	17.881	13.907
	I	0	1.987	15.232	50.993	54.967	50.331	-11.921	63.576	0	17.881	64.238	0	**64.238**	17.881	64.238	**-67.550**	0	-3.311	0	13.907
	AI	**19.834 (34.407)**																			
Iris	N1	69.524	73.333	67.619	58.095	81.905	55.238	72.381	58.095	17.143	33.333	33.333	33.333	48.571	33.333	39.048	33.333	30.476	33.333	20.952	33.333
	N2	69.524	85.714	92.381	86.667	83.810	87.619	72.381	88.571	31.429	50.476	31.429	49.524	32.381	39.048	33.333	38.095	37.143	30.476	40.000	30.762
	I	0	12.381	24.762	28.571	1.905	**32.381**	0.000	30.476	14.286	17.143	-1.905	16.190	**-16.190**	5.714	-5.714	4.762	6.667	-2.857	19.048	-2.857
	AI	**9.238 (13.366)**																			

		Stage 1 Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	95.478	92.707	92.707	92.707	93.553	92.707	92.707	2.275	2.275	2.275	5.163	11.348	2.130	11.348	2.130	92.707	2.130	2.130	2.130	92.532
	N2	95.478	93.932	95.566	93.991	95.478	93.991	90.807	85.838	78.035	75.549	74.652	78.647	74.265	77.567	75.956	80.991	82.256	81.565	78.656	81.016
	I	0	1.225	2.859	1.284	1.925	1.284	-1.900	**83.563**	75.759	73.273	69.489	67.299	72.135	66.219	73.826	**-11.716**	80.127	79.435	76.527	-11.516
	AI	40.055 (39.254)																			
Two Spirals	N1	54.600	42.243	59.771	42.243	50.297	42.243	60.046	42.243	53.112	42.243	48.810	42.243	74.485	42.243	85.011	42.243	77.574	42.243	64.416	42.243
	N2	54.073	55.243	55.860	53.959	59.051	56.037	57.714	54.015	53.007	57.858	54.135	58.626	65.311	66.363	75.444	73.266	74.252	76.288	72.295	71.826
	I	-0.526	13.000	-3.911	11.717	8.753	13.794	-2.332	11.773	-0.105	15.616	5.325	16.384	-9.174	24.120	**-9.568**	31.024	-3.322	**34.046**	7.878	29.584
	AI	9.704 (13.013)																			
AAI		26.857 (31.574)																			
TAI		23.906 (31.211)																			

		Stage 1 Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	95.478	92.707	92.970	92.707	93.495	92.707	93.320	2.275	2.275	2.275	5.543	2.275	5.572	92.707	5.163	2.275	5.601	5.163	5.572	2.130
	N2	95.478	93.932	95.566	94.020	95.566	93.961	94.457	87.155	82.656	78.257	75.555	76.297	75.895	78.266	74.653	78.596	80.656	81.662	78.466	80.037
	I	0	1.225	2.596	1.313	2.071	1.254	1.138	**84.879**	80.381	75.981	70.012	74.021	70.324	**-14.442**	69.489	76.320	75.056	76.499	72.894	77.907
	AI	44.946 (38.464)																			
Two Spirals	N1	53.730	42.243	51.968	42.243	52.929	42.243	61.968	42.243	62.586	42.243	57.483	42.243	42.243	42.243	85.973	42.243	84.050	42.243	50.664	42.243
	N2	57.506	52.243	52.572	51.979	51.954	52.659	56.309	56.768	58.882	56.967	58.822	56.903	57.060	61.990	69.405	68.736	71.924	71.593	71.730	70.375
	I	3.776	10.000	0.604	9.736	-0.975	10.416	-5.659	14.526	-3.703	14.725	1.339	14.660	14.817	19.747	**-16.568**	26.494	-12.127	**29.350**	21.066	28.132
	AI	9.018 (13.003)																			
AAI		27.212 (32.67)																			
TAI		23.906 (31.211)																			

		Stage 1 Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	95.478	92.707	93.466	92.707	92.853	92.707	92.736	2.275	92.707	92.707	92.211	2.975	92.678	2.130	92.707	92.707	92.707	2.130	92.707	92.620
	N2	95.478	93.932	95.508	93.961	93.900	93.991	95.537	87.130	90.130	91.071	90.130	86.901	92.707	84.217	90.130	88.195	90.520	85.951	91.051	90.052
	I	0	1.225	2.042	1.254	1.047	1.284	2.800	**84.854**	-2.578	-1.636	-2.082	83.925	0.029	82.088	-2.578	**-4.512**	-2.188	83.822	-1.656	-2.568
	AI	16.229 (34.651)																			
Two Spirals	N1	48.284	42.243	42.243	39.085	55.309	38.558	50.481	49.954	58.627	49.611	53.982	4.211	9.474	0.000	52.243	43.638	48.810	0.000	46.979	0.000
	N2	50.389	45.243	48.005	50.611	48.071	45.979	51.822	52.966	55.265	51.442	51.808	53.455	53.822	51.716	53.295	55.972	56.924	55.813	55.478	54.746
	I	2.105	3.000	5.762	11.526	**-7.238**	7.421	1.341	3.011	-3.362	1.831	-2.174	49.245	44.348	51.716	1.053	12.334	8.114	**55.813**	8.499	54.746
	AI	15.455 (21.777)																			
AAI		17.649 (28.55)																			
TAI		23.906 (31.211)																			

N1 Classification accuracy of Stage 1 kernels implemented using the vectorial approach & the Linear dot-product kernel
N2 Classification accuracy of Stage 1 kernels implemented using the concatenated approach & the Linear dot-product kernel
I Improvement = N2-N1
AI Average Improvement for the first 20 orders
AAI Average of Average Improvements of the first 20 orders over all the datasets under investigation
TAI Total Average Improvement over all datasets and all kernels under investigation for the first 20 orders

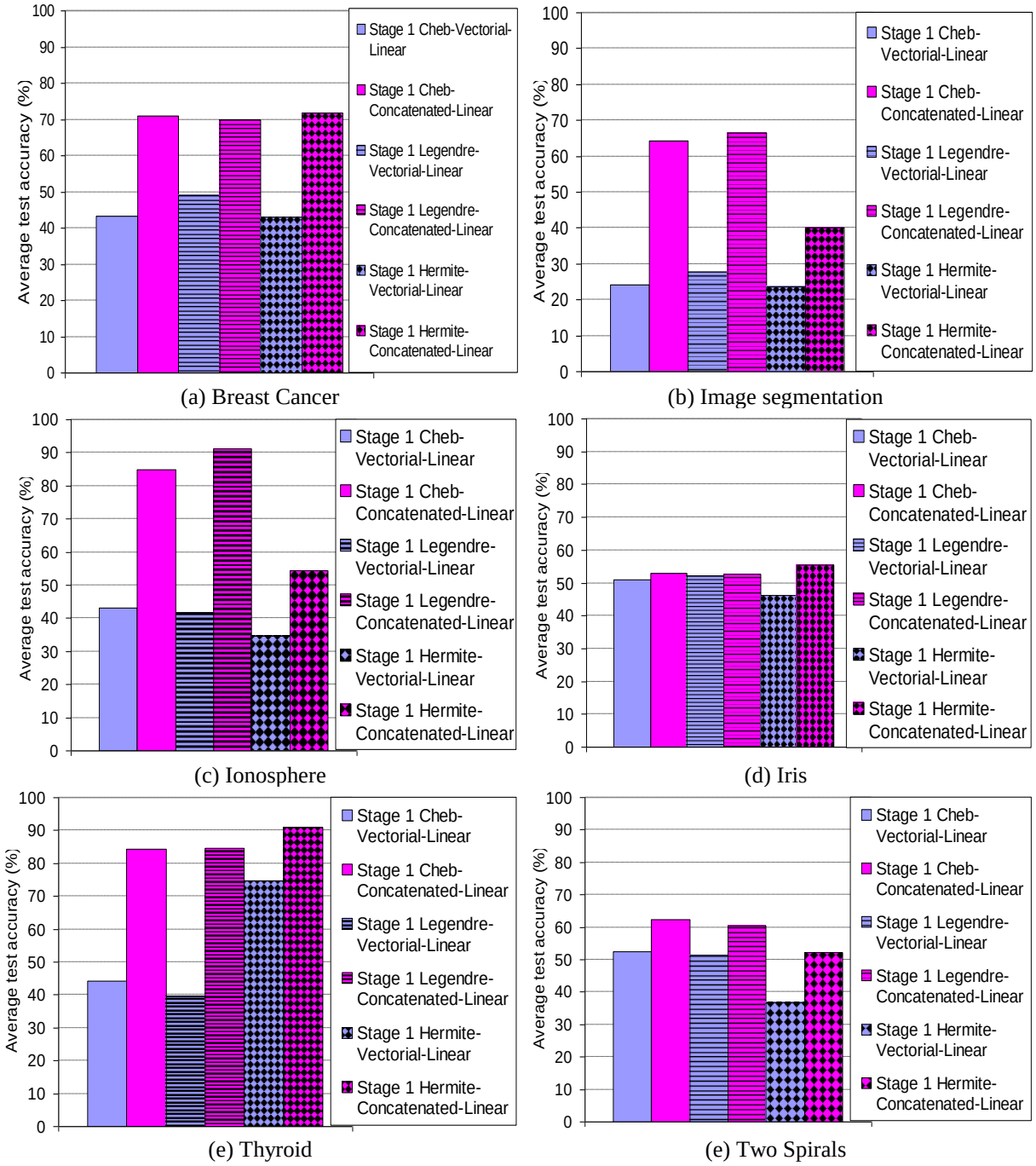


Figure 6.9 Bar chart comparative assessment of the average classification accuracy scored by the first 20 orders of the Stage 1 concatenated-linear and vectorial-linear kernels.

The same set of experiments conducted on Stage 1 kernels was carried out again below to assess the performance of the concatenated approach versus its vectorial counterpart when using the Gaussian instead of the linear kernel, as per their mathematical formulations shown in Table 6.2 for the Stage 1- Chebyshev, Legendre, and Hermite polynomials. The highest classification accuracy scores are shown in the comparative graphical results illustrated in Table 6.5, whereas the average accuracy scored by the first 20 orders are shown in Figure 6.10.

Inline with the results achieved from the linear kernel, one can also still observe that in the experiments conducted using the Gaussian kernel, the proposed concatenated approach was actually able to achieve even more superior performance than its vectorial counterpart. For example, in the graphical results illustrated in Table 6.5, one can notice that the concatenated-Gaussian kernels demonstrated a more superior performance than the concatenated-linear kernels, resulting in even bigger improvement relative to the vectorial-linear kernels, not only for the higher polynomial orders, but also for some of the lower orders too. This observation can be realised, for example, from the results achieved from the Breast Cancer, Image Segmentation, and Thyroid datasets.

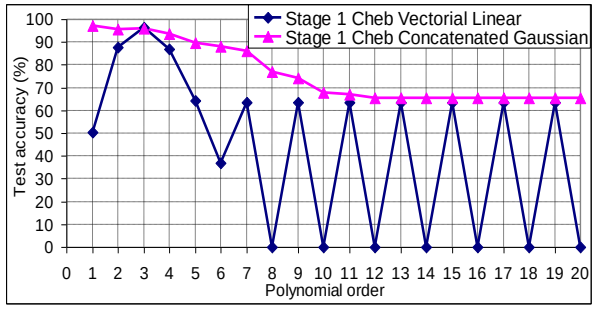
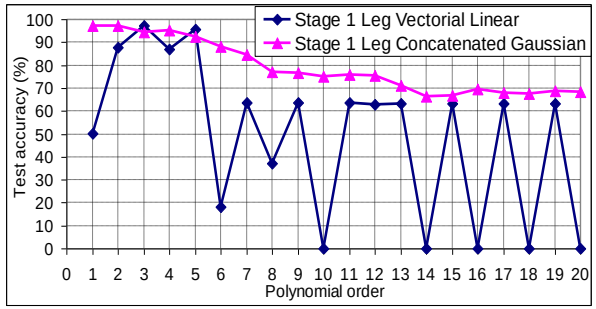
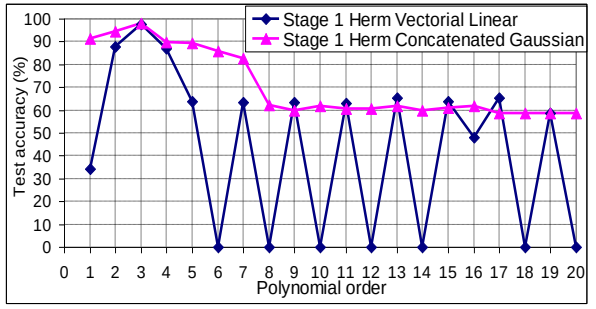
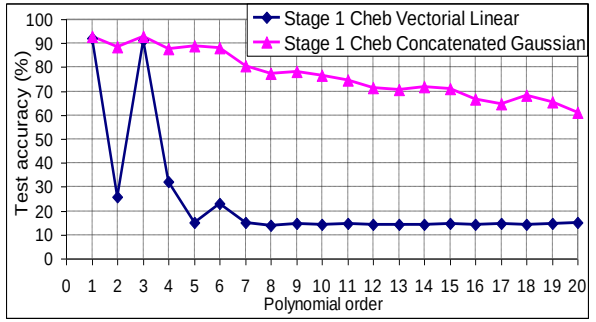
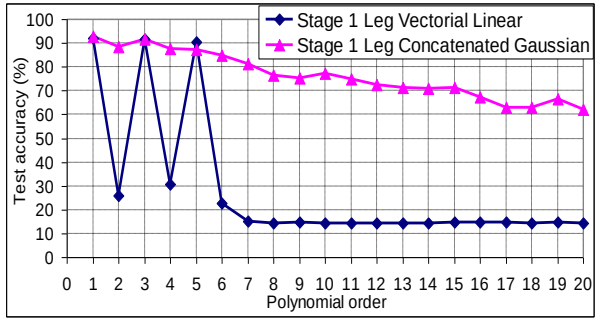
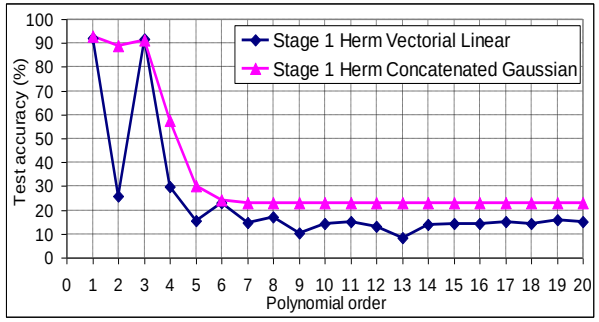
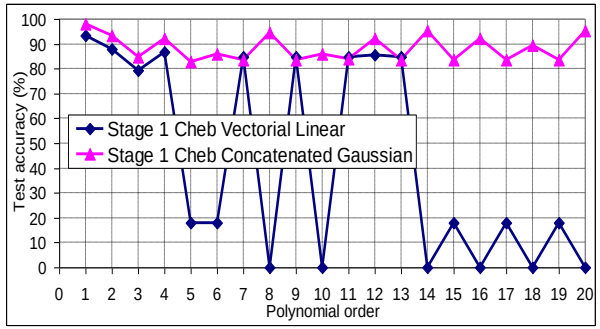
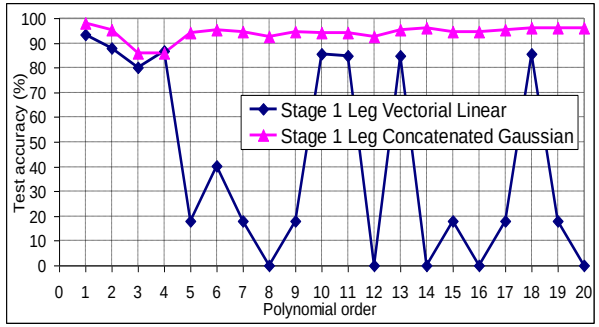
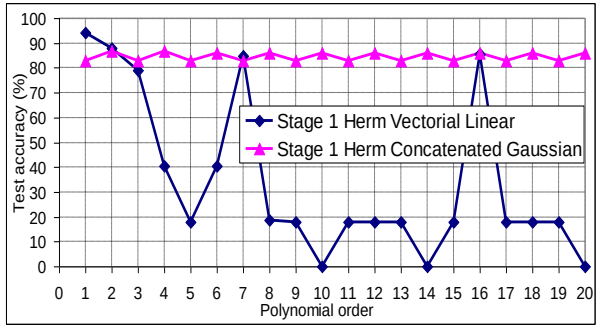
Similar to the results obtained form the concatenated-linear kernels, however, one can also still observe that the improvement in the classification performance introduced by the concatenated-Gaussian kernels can still be relatively smaller in some datasets than others, as can be observed, for example, from the Iris and Two Spirals datasets in Table 6.5. Despite this small improvement relative to the vectorial-linear kernels, yet, the Gaussian kernel was still able to constructively influence the classification performance in these datasets by increasing the scored classification accuracies compared to when the linear kernel is evaluated on the concatenated processed vectors.

To better quantify the constructive effect of the Gaussian kernel on the performance of the concatenated polynomial kernels, Table 6.6 has also been similarly used to calculate the average accuracy improvement factors between the Stage 1 concatenated-Gaussian kernels and their Stage 1 vectorial-linear counterparts. At the bottom of the table, these average improvement factors were calculated to be equal to 32.999%, 33.857%, and 21.445% for the Stage 1- Chebyshev, Legendre, and Hermite kernels respectively; making up a total average accuracy improvement factor of 29.434% depending on the kernel and the dataset used; which again clearly demonstrates the outperformance of the proposed concatenated approach over its vectorial counterpart. Compared to its corresponding average improvement factor using the concatenated-linear kernel (23.906%), this result clearly

demonstrates the effectiveness of utilizing the Gaussian instead of the linear kernel by introducing a further average improvement factor of 5.528%. This empirical superiority of the Gaussian kernel over its linear counterpart, when evaluated on the concatenated polynomial-processed input vectors, is believed to stem from its renowned ability of mapping the input space to an infinite dimensional feature space, as well as it accurate similarity measure shape characteristics, as explained earlier in Section 6.2.

Another benefit gained from utilizing the Gaussian kernel to evaluate the similarity between the concatenated-processed input vectors is that it resulted in a lot smoother classification performance of the overall concatenated-Gaussian polynomial kernels than the concatenated-linear polynomial kernels, across the spectrum of polynomial orders. This behaviour can be realized by comparing the two sets of results from Tables 6.5 and 6.3, respectively, which is quite evident especially in the Hermite kernels in the Ionosphere dataset where the abrupt oscillating changes in the classification accuracy between the polynomial orders are significantly reduced. Consequently, this resulted in even more dispersed improvements in the classification accuracies between the concatenated-Gaussian and the vectorial-linear kernels, which can also be quantitatively evaluated from the standard deviation of the average improvement factors (shown in brackets), as well as the minimum and maximum improvement factors (shown in bold) which are also calculated in Table 6.6.

Table 6.5 Comparative experimental results of Stage 1 concatenated-Gaussian and vectorial-linear kernels.

	Stage 1 Chebyshev kernels	Stage 1 Legendre kernels	Stage 1 Hermite kernels
Breast Cancer			
Image Segmentation			
Ionosphere			

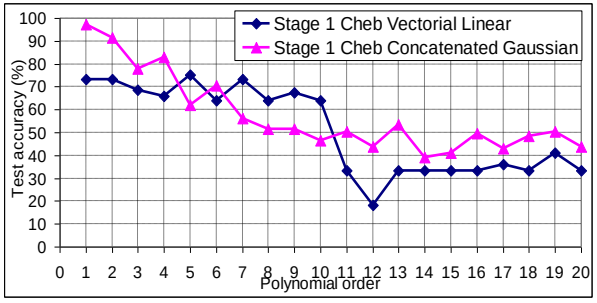
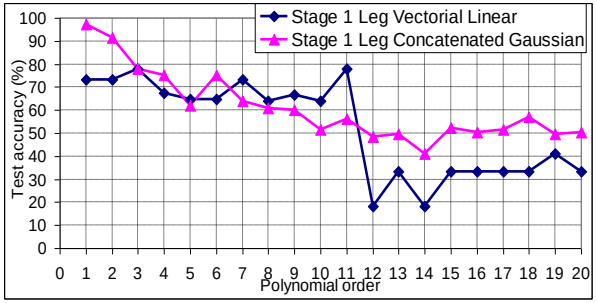
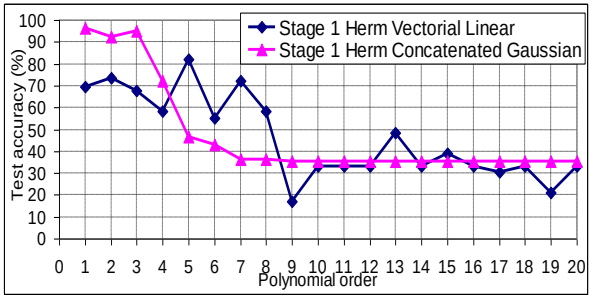
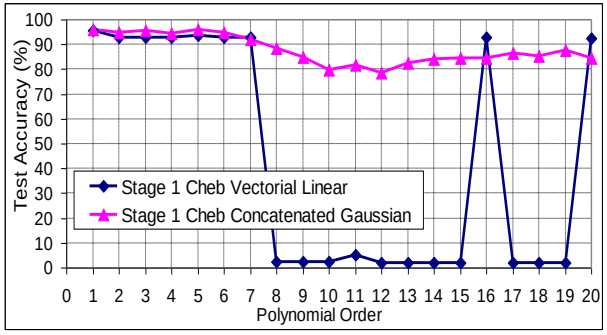
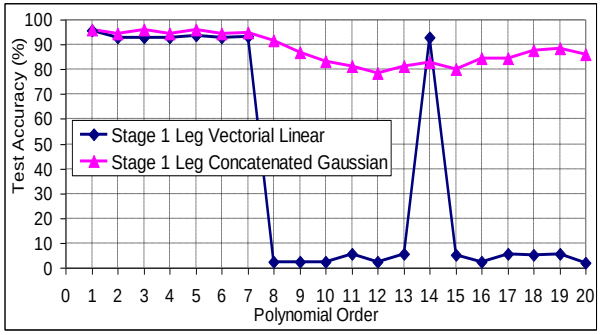
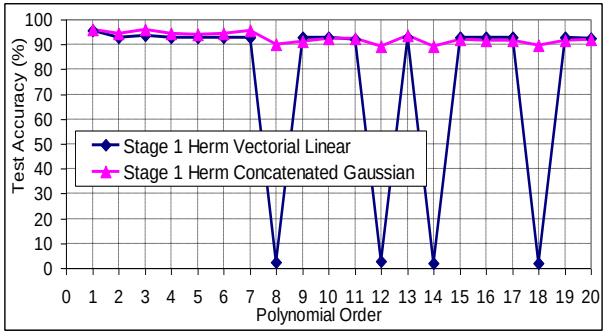
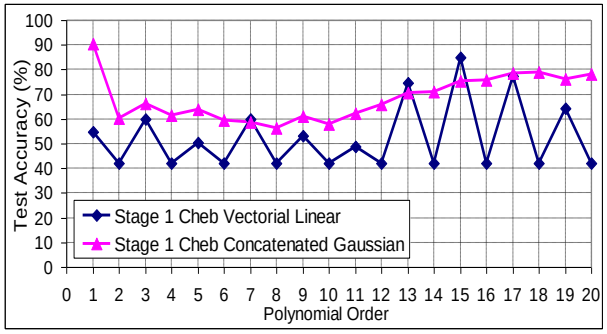
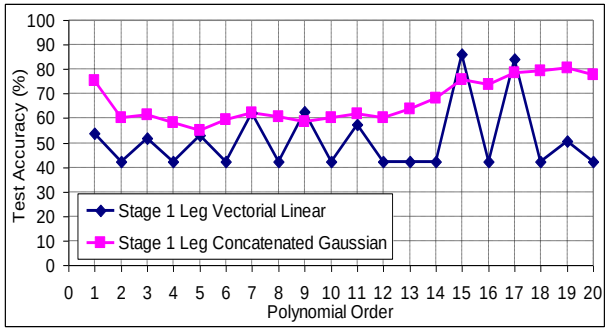
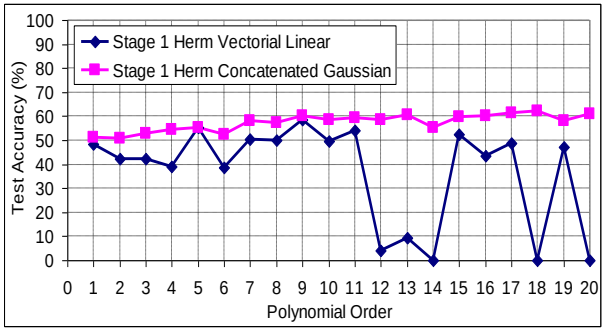
	Stage 1 Chebyshev kernels	Stage 1 Legendre kernels	Stage 1 Hermite kernels
Iris			
Thyroid			
Two Spirals			

Table 6.6 Quantitative assessment of the average improvements in the scored classification accuracy of Stage 1 concatenated-Gaussian and vectorial-linear kernels.

		Stage 1 Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	50.320	87.633	96.588	86.994	64.179	37.100	63.539	0.000	63.539	0.000	63.539	0.000	63.326	0.000	63.326	0.000	63.326	0.000	63.326	0.000
	N2	97.228	95.522	95.949	93.817	89.552	88.273	86.141	76.972	74.200	67.804	66.951	65.458	65.458	65.458	65.458	65.458	65.458	65.458	65.458	65.458
	I	46.908	7.889	**-0.640**	6.823	25.373	51.173	22.601	**76.972**	10.661	67.804	3.412	65.458	2.132	65.458	2.132	65.458	2.132	65.458	2.132	65.458
	AI	**32.740 (29.678)**																			
Image Segmentation	N1	92.143	25.714	91.333	32.095	14.905	22.857	15.000	13.857	14.619	14.286	14.524	14.286	14.476	14.286	14.619	14.286	14.619	14.286	14.619	15.048
	N2	92.952	88.619	92.714	87.762	88.714	87.952	80.667	77.571	78.048	76.476	74.524	71.333	70.762	71.905	71.095	66.714	64.857	68.190	65.381	60.952
	I	**0.810**	62.905	1.381	55.667	**73.810**	65.095	65.667	63.714	63.429	62.190	60.000	57.048	56.286	57.619	56.476	52.429	50.238	53.905	50.762	45.905
	AI	**52.767 (18.798)**																			

		Stage 1 Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	50.320	87.633	97.228	86.994	95.522	18.337	63.539	37.100	63.539	0.000	63.539	62.900	63.326	0.000	63.326	0.000	63.326	0.000	63.326	0.000
	N2	97.228	97.228	94.456	95.096	92.537	88.060	84.648	77.186	76.546	75.267	75.906	75.480	71.215	66.311	66.951	69.723	67.804	67.591	68.657	68.444
	I	46.908	9.595	-2.772	8.102	**-2.985**	69.723	21.109	40.085	13.006	**75.267**	12.367	12.580	7.889	66.311	3.625	69.723	4.478	67.591	5.330	68.444
	AI	**29.819 (29.278)**																			
Image Segmentation	N1	92.143	25.762	91.762	30.524	90.524	22.857	15.000	14.286	14.619	14.286	14.524	14.286	14.476	14.286	14.619	14.619	14.619	14.286	14.619	14.286
	N2	92.952	88.381	91.714	87.571	87.095	85.000	81.095	76.524	75.190	77.143	74.714	72.667	71.190	70.905	71.238	67.381	62.810	62.905	66.571	62.333
	I	0.810	62.619	-0.048	57.048	**-3.429**	62.143	**66.095**	62.238	60.571	62.857	60.190	58.381	56.714	56.619	56.619	52.762	48.190	48.619	51.952	48.048
	AI	**48.450 (21.885)**																			

		Stage 1 Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	34.328	87.633	97.655	86.994	63.966	0.000	63.539	0.000	63.326	0.000	62.900	0.000	65.458	0.000	63.753	47.974	65.458	0.000	58.635	0.000
	N2	91.471	94.456	97.968	89.877	89.346	85.735	82.755	62.348	60.007	61.747	60.547	60.647	61.829	59.775	60.858	61.855	58.644	58.644	58.644	58.644
	I	57.143	6.823	0.314	2.883	25.380	**85.735**	19.215	62.348	-3.319	61.747	-2.353	60.647	-3.629	59.775	-2.894	13.880	**-6.815**	58.644	0.008	58.644
	AI	**27.709 (31.149)**																			
Image Segmentation	N1	92.190	25.762	91.762	29.667	15.381	22.857	14.619	16.952	10.143	14.286	15.143	13.000	8.238	13.714	14.381	14.286	15.190	14.286	15.810	15.143
	N2	92.810	88.857	91.190	57.381	30.143	24.381	23.048	23.000	22.952	22.952	22.952	22.952	22.952	22.952	22.952	22.952	22.952	22.952	22.952	22.952
	I	0.619	**63.095**	**-0.571**	27.714	14.762	1.524	8.429	6.048	12.810	8.667	7.810	9.952	14.714	9.238	8.571	8.667	7.762	8.667	7.143	7.810
	AI	**11.671 (13.475)**																			

		Stage 1 Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ionosphere	N1	93.378	88.079	79.470	86.755	17.881	17.881	84.768	0.000	84.768	0.000	84.768	85.430	84.768	0.000	17.881	0.000	17.881	0.000	17.881	0.000
	N2	98.013	93.377	84.768	92.053	82.781	86.093	83.447	94.702	83.444	86.093	84.106	92.053	83.444	95.364	83.447	92.053	83.447	89.404	83.444	95.364
	I	4.636	5.298	5.298	5.298	64.901	68.212	-1.325	94.702	-1.325	86.093	-0.662	6.623	**-1.325**	**95.364**	65.563	92.053	65.562	89.404	65.562	95.364
	AI	**45.265 (41.022)**																			
Iris	N1	73.333	73.333	68.571	65.714	75.238	63.810	73.333	63.810	67.619	63.810	33.333	18.095	33.333	33.333	33.333	33.333	36.190	33.333	40.952	33.333
	N2	97.143	91.429	78.095	82.857	61.905	70.476	56.190	51.429	51.429	46.667	50.476	43.810	53.333	39.048	40.952	49.524	42.857	48.571	50.476	43.810
	I	23.810	18.095	9.524	17.143	-13.333	6.667	-17.143	-12.381	-16.190	**-17.143**	17.143	**25.714**	20.000	5.714	7.619	16.190	6.667	15.238	9.524	10.476
	AI	**6.667 (21.473)**																			

		Stage 1 Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ionosphere	N1	93.377	88.079	80.132	86.755	17.881	40.397	17.881	0.000	17.881	85.430	84.768	0.000	17.881	0.000	17.881	0.000	17.881	85.430	17.881	0.000
	N2	98.013	95.364	86.093	86.093	94.040	95.364	94.702	92.715	94.702	94.040	94.040	92.715	95.364	96.026	94.702	94.702	95.364	96.026	96.026	96.026
	I	4.636	7.285	5.960	**-0.662**	76.159	54.967	76.821	92.715	76.821	8.609	9.272	92.715	10.596	96.026	76.821	94.702	77.483	10.596	78.146	**96.026**
	AI	**52.285 (39.109)**																			
Iris	N1	73.333	73.333	78.095	67.619	64.762	64.762	73.333	63.810	66.667	63.810	78.095	18.095	33.333	33.333	33.333	33.333	33.333	33.333	40.952	33.333
	N2	97.143	91.429	78.095	75.238	61.905	75.238	63.810	60.952	60.000	51.429	56.190	48.571	49.524	40.952	52.381	50.476	51.429	57.143	49.524	50.476
	I	23.810	18.095	0.000	7.619	-2.857	10.476	-9.524	-2.857	-6.667	-12.381	**-21.905**	**30.476**	16.190	22.857	19.048	17.143	18.095	23.810	8.571	17.143
	AI	**8.857 (14.344)**																			

		Stage 1 Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ionosphere	N1	94.040	88.079	78.808	40.397	17.881	40.397	84.768	18.543	17.881	0.000	17.881	17.881	17.881	17.881	85.430	17.881	17.881	17.881	17.881	0.000
	N2	82.781	86.755	82.781	86.755	82.781	86.093	82.781	86.093	82.781	86.093	82.781	86.093	82.781	86.093	86.093	82.781	86.093	82.781	86.093	86.093
	I	**-11.258**	-1.325	3.974	46.358	64.901	45.695	-1.987	67.550	64.901	**86.093**	64.901	68.212	64.901	86.093	64.901	0.662	64.901	64.901	68.212	64.901	86.093
	AI	**49.934 (32.531)**																			
Iris	N1	69.524	73.333	67.619	58.095	81.905	55.238	72.381	58.095	17.143	33.333	33.333	33.333	48.571	33.333	39.048	33.333	30.476	33.333	20.952	33.333
	N2	96.190	92.381	95.238	72.381	46.667	42.857	36.190	36.190	35.238	35.238	35.238	35.238	35.238	35.238	35.238	35.238	35.238	35.238	35.238	35.238
	I	26.667	19.048	**27.619**	14.286	-35.238	-12.381	**-36.190**	-21.905	18.095	1.905	1.905	1.905	-13.333	1.905	-3.810	1.905	4.762	1.905	14.286	1.905
	AI	**0.762 (17.749)**																			

		Stage 1 Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	95.478	92.707	92.707	92.707	93.553	92.707	92.707	2.275	2.275	2.275	5.163	2.000	2.130	2.000	2.130	92.707	2.130	2.130	2.130	92.532
	N2	95.965	94.981	95.545	94.596	95.862	94.655	91.962	88.653	84.963	79.630	81.627	78.548	82.649	83.956	84.664	84.534	86.655	85.324	87.654	84.570
	I	0.487	2.274	2.838	1.889	2.309	1.948	-0.745	**86.378**	82.687	77.355	76.464	76.548	80.520	81.956	82.535	**-8.173**	84.526	83.195	85.525	-7.962
	AI	**44.628 (42.119)**																			
Two Spirals	N1	54.600	42.243	59.771	42.243	50.297	42.243	60.046	42.243	53.112	42.243	48.810	42.243	74.485	42.243	85.011	42.243	77.574	42.243	64.416	42.243
	N2	90.618	60.160	66.087	61.660	63.904	59.526	58.652	56.270	61.167	57.959	62.296	65.930	70.769	71.226	75.430	75.660	78.481	78.960	76.060	78.263
	I	36.018	17.917	6.316	19.417	13.606	17.284	-1.394	14.027	8.055	15.717	13.486	23.687	-3.716	28.984	**-9.581**	33.417	0.906	**36.717**	11.643	36.020
	AI	**15.926 (13.658)**																			
AAI		**32.999 (33.014)**																			
TAI		**29.434 (32.472)**																			

		Stage 1 Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	95.478	92.707	92.970	92.707	93.495	92.707	93.320	2.275	2.275	2.275	5.543	2.275	5.572	92.707	5.163	2.275	5.601	5.163	5.572	2.130
	N2	95.986	94.554	95.852	94.569	95.853	94.236	94.967	91.563	86.937	83.254	81.426	78.548	81.237	82.963	80.254	84.527	84.326	87.655	88.264	86.158
	I	0.507	1.847	2.882	1.861	2.358	1.528	1.647	**89.288**	84.661	80.979	75.883	76.273	75.665	**-9.744**	75.090	82.251	78.725	82.492	82.693	84.029
	AI	**48.546 (40.582)**																			
Two Spirals	N1	53.730	42.243	51.968	42.243	52.929	42.243	61.968	42.243	62.586	42.243	57.483	42.243	42.243	42.243	85.973	42.243	84.050	42.243	50.664	42.243
	N2	75.157	60.152	61.162	58.160	55.166	59.270	62.263	60.526	58.657	60.265	61.655	60.256	63.656	68.264	75.596	73.595	78.415	79.264	80.598	77.634
	I	21.427	17.909	9.194	15.917	2.237	17.027	0.295	18.283	-3.929	18.023	4.172	18.014	21.414	26.022	**-10.376**	31.352	-5.635	**37.022**	29.935	35.392
	AI	**15.185 (13.755)**																			
AAI		**33.857 (32.849)**																			
TAI		**29.434 (32.472)**																			

		Stage 1 Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	95.478	92.707	93.466	92.707	92.853	92.707	92.736	2.275	92.707	92.707	92.211	2.975	92.678	2.130	92.707	92.707	92.707	2.130	92.707	92.620
	N2	95.956	94.537	95.965	94.563	94.237	94.563	95.632	90.155	91.446	92.542	92.265	89.265	93.566	89.297	92.245	91.557	91.654	89.655	91.862	92.189
	I	0.478	1.830	2.500	1.856	1.384	1.856	2.896	87.880	**-1.261**	-0.165	0.054	86.290	0.888	**87.167**	-0.462	-1.150	-1.053	87.525	-0.845	-0.430
	AI	**17.862 (35.6)**																			
Two Spirals	N1	48.284	42.243	42.243	39.085	55.309	38.558	50.481	49.954	58.627	49.611	53.982	4.211	9.474	0.000	52.243	43.638	48.810	0.000	46.979	0.000
	N2	51.267	50.946	52.985	54.296	55.215	52.595	58.285	57.257	60.260	58.665	59.265	58.457	60.741	55.446	59.646	60.216	61.456	62.264	58.265	60.846
	I	2.983	8.704	10.742	15.212	**-0.093**	14.036	7.804	7.302	1.633	9.054	5.283	54.247	51.268	55.446	7.404	16.577	12.646	**62.264**	11.285	60.846
	AI	**20.732 (21.885)**																			
AAI		**21.445 (30.252)**																			
TAI		**29.434 (32.472)**																			

N1 Classification accuracy of Stage 1 kernels implemented using the vectorial approach & the Linear dot-product kernel
N2 Classification accuracy of Stage 1 kernels implemented using the concatenated approach & the Gaussian kernel
I Improvement = N2-N1
AI Average Improvement for the first 20 orders
AAI Average of Average Improvements of the first 20 orders over all the datasets under investigation
TAI Total Average Improvement over all datasets and all kernels under investigation for the first 20 orders

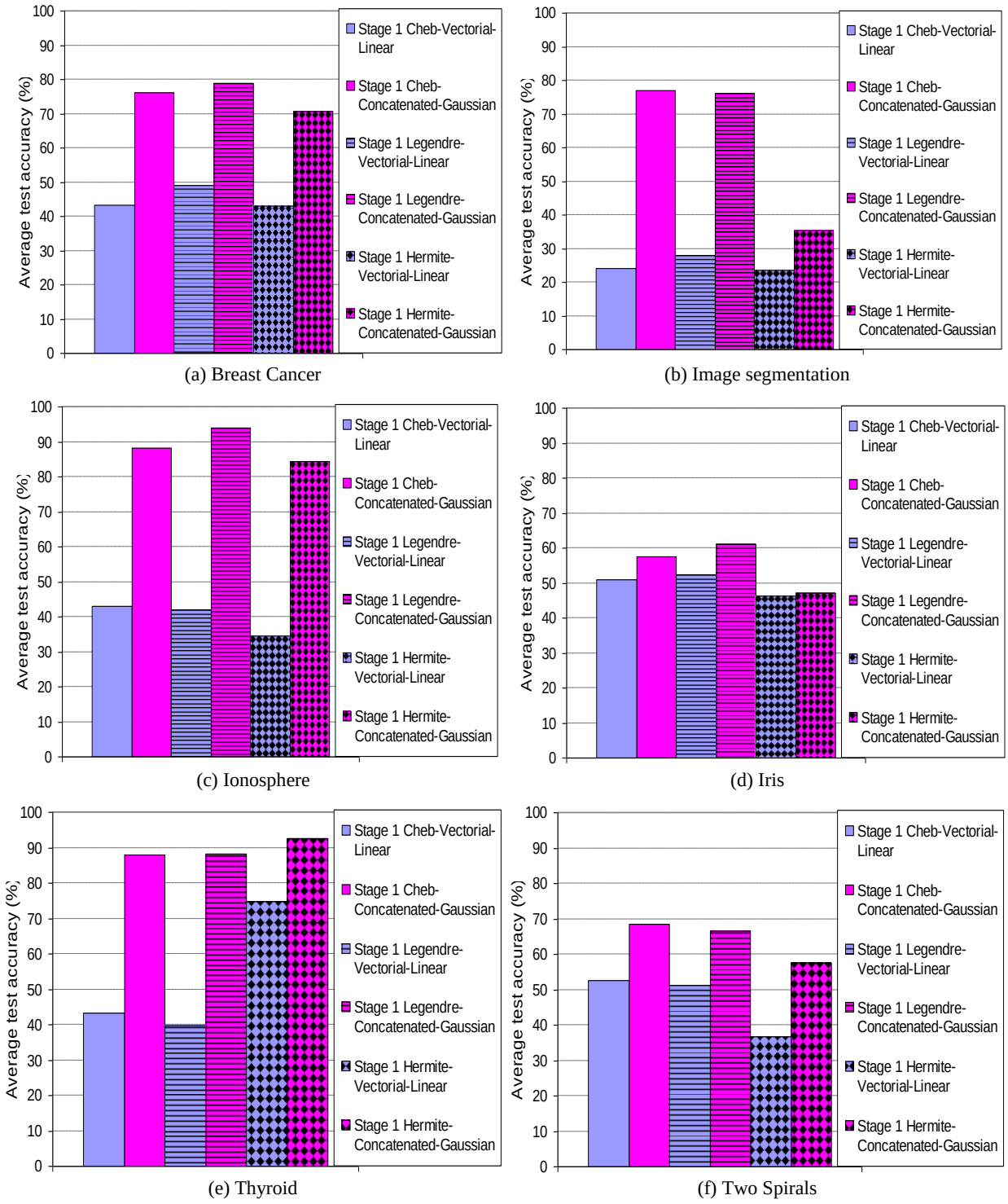


Figure 6.10 Bar chart comparative assessment of the average classification accuracy scored by the first 20 orders of the Stage 1 concatenated-Gaussian and vectorial-linear kernels.

6.5.2.2 Comparative experimental results on Stage 2 kernels

Stage 2 kernels might be the most appropriate kernels that can be used to establish a fair comparison between the classification performance of the concatenated and the vectorial processing approaches under investigation. This is mainly because, as previously explained in Chapter 4 via the analysis of their shape characteristics, they are actually able to acquire more accurate similarity measures than their Stage 1 counterparts due to their inherent fusion by summation process. Their calculated similarity measures will also be a faithful indication of what the unweighted polynomial kernel can produce, because it is not being affected by any other weighting or combining kernel (as is the case for example with the Stage 3 kernels), although this can be quite useful to elevate the classification accuracy, as previously shown in Chapter 4.

Similar to Stage 1 kernels, Tables 6.7 and 6.8 also show the mathematical formulation of each of the Stage 2 kernels utilized in the comparative experiments conducted in this section between the concatenated and vectorial approaches, when adopting both the linear and Gaussian kernels.

Table 6.7. Mathematical formulations of the Stage 2 linear kernel evaluation on the vectorial- and concatenated-processed inputs using the Chebyshev, Legendre, and Hermite polynomials.

	Vectorial-linear kernel	**Concatenated-linear kernel**
Chebyshev	$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle$	$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} < [T_i(x_1),\cdots,T_i(x_m)], [T_i(z_1),\cdots,T_i(z_m)] >$
Legendre	$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle L_i(\mathbf{x}), L_i(\mathbf{z}) \rangle$	$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} < [L_i(x_1),\cdots,L_i(x_m)], [L_i(z_1),\cdots,L_i(z_m)] >$
Hermite	$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle H_i(\mathbf{x}), H_i(\mathbf{z}) \rangle$	$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} < [H_i(x_1),\cdots,H_i(x_m)], [H_i(z_1),\cdots,H_i(z_m)] >$

Table 6.8. Mathematical formulations of the Stage 2 Gaussian kernel evaluation on the concatenated-processed vectors versus the linear kernel evaluation on the vectorial-processed inputs using the Chebyshev, Legendre, and Hermite polynomials.

	Vectorial-linear kernel	**Concatenated-Gaussian kernel**
Chebyshev	$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle$	$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma \left\| [T_i(x_1),\cdots,T_i(x_m)] - [T_i(z_1),\cdots,T_i(z_m)] \right\|^2\right)$
Legendre	$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle L_i(\mathbf{x}), L_i(\mathbf{z}) \rangle$	$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma \left\| [L_i(x_1),\cdots,L_i(x_m)] - [L_i(z_1),\cdots,L_i(z_m)] \right\|^2\right)$
Hermite	$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle H_i(\mathbf{x}), H_i(\mathbf{z}) \rangle$	$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma \left\| [H_i(x_1),\cdots,H_i(x_m)] - [H_i(z_1),\cdots,H_i(z_m)] \right\|^2\right)$

The graphical experimental results shown in Table 6.9 compare the best scored classification accuracies obtained from the first 20 polynomial orders of the Stage 2 kernels shown in Table 6.7, where the linear kernel is evaluated on both the vectorial- and concatenated-processed inputs for the same datasets investigated. Inline with the results obtained from Stage 1, the results from Stage 2 kernels also demonstrate not only the superiority of the concatenated approach over its vectorial counterpart, but also the fact that the kernels constructed using the concatenated approach are less sensitive to the changes between the parity of the orders of the employed polynomials, compared to their counterparts constructed using the vectorial approach. It is believed by the author that this more stable performance of the concatenated approach is mainly due to its ability to produce transformed image vectors of the input data for any polynomial order, as opposed to the vectorial approach, which does this only with the odd orders, as previously explained in Section 6.4.1.

One can also observe that, depending on the characteristics of the dataset, both the vectorial- and concatenated-linear kernels can have different behaviours. For example, at the time when the classification performance of the kernels increases as the polynomial order increases in the Two Spirals dataset, as shown in Table 6.9, yet, in other datasets, such as the Thyroid and Image Segmentation datasets, it is usually the lower orders which score the highest accuracies.

Furthermore, it can also be clearly observed that the improvement in the classification accuracy introduced by the proposed concatenated approach can be very large in some datasets than others. For example, in the majority of the results illustrated in Table 6.9, such as the Breast Cancer, Image Segmentation, Ionosphere, and Thyroid datasets, the proposed concatenated approach has shown to significantly outperform its vectorial counterpart mostly in higher polynomial orders. A similar behaviour has also been observed in the experimental results of Stage 1 kernels in Section 6.5.2.1.

In the Two Spirals dataset, however, although the concatenated approach has still demonstrated a better performance than its vectorial counterpart, yet, the introduced improvement is quite infinitesimal and tends to vanish as the order increases, especially for the Chebyshev and Legendre kernels. A similar behaviour can also be observed in the Iris dataset, where the concatenated approach has also demonstrated a relatively small improvement over its vectorial counterpart, apart from some weird few orders which tend to form a convex hull in the Chebyshev and Legendre kernels where the vectorial approach collapses considerably.

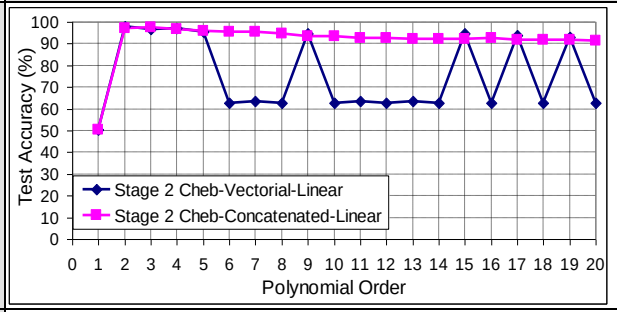
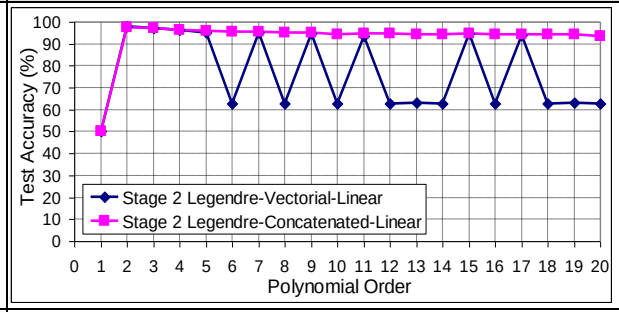
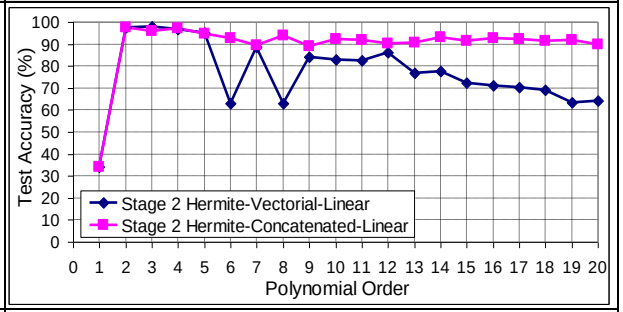
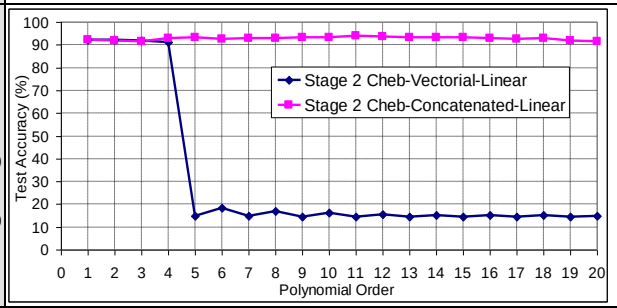
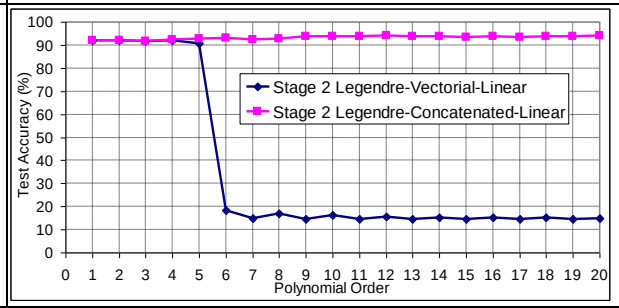
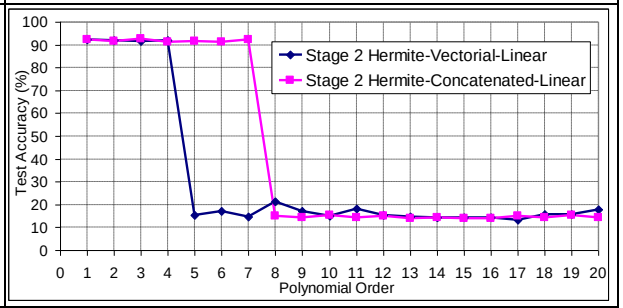
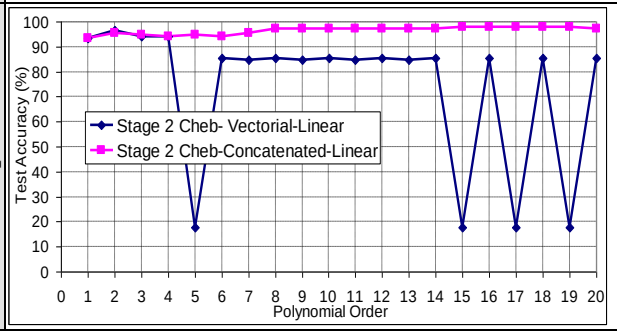
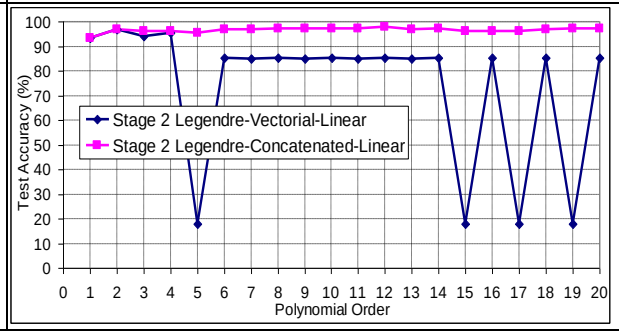
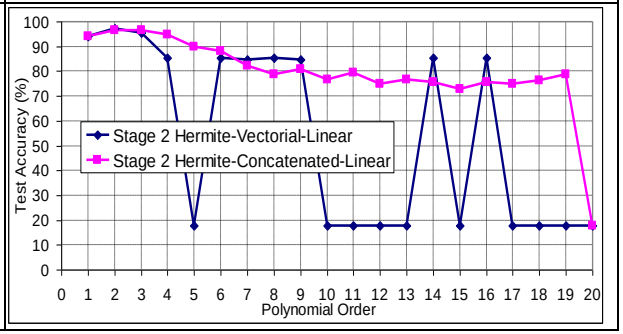
One can also evaluate the amount of improvement introduced by the proposed concatenated approach by examining the bar chart comparisons, shown in Figure 6.11, which demonstrate the average accuracy scored by the first 20 orders of the concatenated and the vectorial approaches, for each of the datasets experimented. As can be observed, the concatenated approach demonstrated a consistent superior average accuracy over its vectorial counterpart for all the datasets experimented, but with varying amounts of improvements from one dataset to another.

To quantify this variation in the improvements scored for each dataset, Table 6.10 has also been similarly used to calculate the average accuracy improvement factors that the concatenated approach was able to achieve over its vectorial counterpart for the first 20 orders of the Stage 2 kernels under test. As shown at the bottom of the table, the average accuracy improvement factors introduced by the concatenated approach for the Stage 2-Chebyshev, Legendre, and Hermite kernels was calculated to be equal to 29.369%, 28.456% and 11.224%, respectively; making up a total average accuracy improvement factor of 23.016%, which clearly demonstrates the outperformance of the proposed concatenated approach over its vectorial counterpart, even with the linear kernel implemented.

On the other hand, similar to the behaviour of the Stage 1 concatenated kernels illustrated in Section 6.5.2.1, one can also clearly observe that the Stage 2 concatenated kernels has also demonstrated a lot more steady and consistent classification accuracy across the spectrum of polynomial orders than their Stage 2 vectorial counterparts, with a significant reduction in the abrupt oscillating behaviour between the even and odd orders, as shown for example in the Breast Cancer and Ionosphere datasets in Table 6.9. To quantify this observation, Table 6.10 has also been used to calculate the standard deviation of the average improvement factors (shown in brackets), as well as the minimum and maximum improvement factors (shown in bold), introduced by the Stage 2 concatenated kernels.

As it can be observed, the standard deviation of the average improvement factors can also be quite large relative to the mean, due to the huge difference between the two sets of results across the polynomial orders. Conversely, however, one can also evaluate the steadiness of the results by examining the standard deviation of the accuracy values themselves, reported in the statistical analysis tests tabulated in Appendix B.5, where, not only the mean of the Stage 2 concatenated kernels is consistently larger than the mean of the vectorial kernels, but also their standard deviation is a lot smaller for most of the results obtained.

Table 6.9 Comparative experimental results of Stage 2 concatenated-linear and vectorial-linear kernels.

	Stage 2 Chebyshev kernels	Stage 2 Legendre kernels	Stage 2 Hermite kernels
Breast Cancer			
Image Segmentation			
Ionosphere			

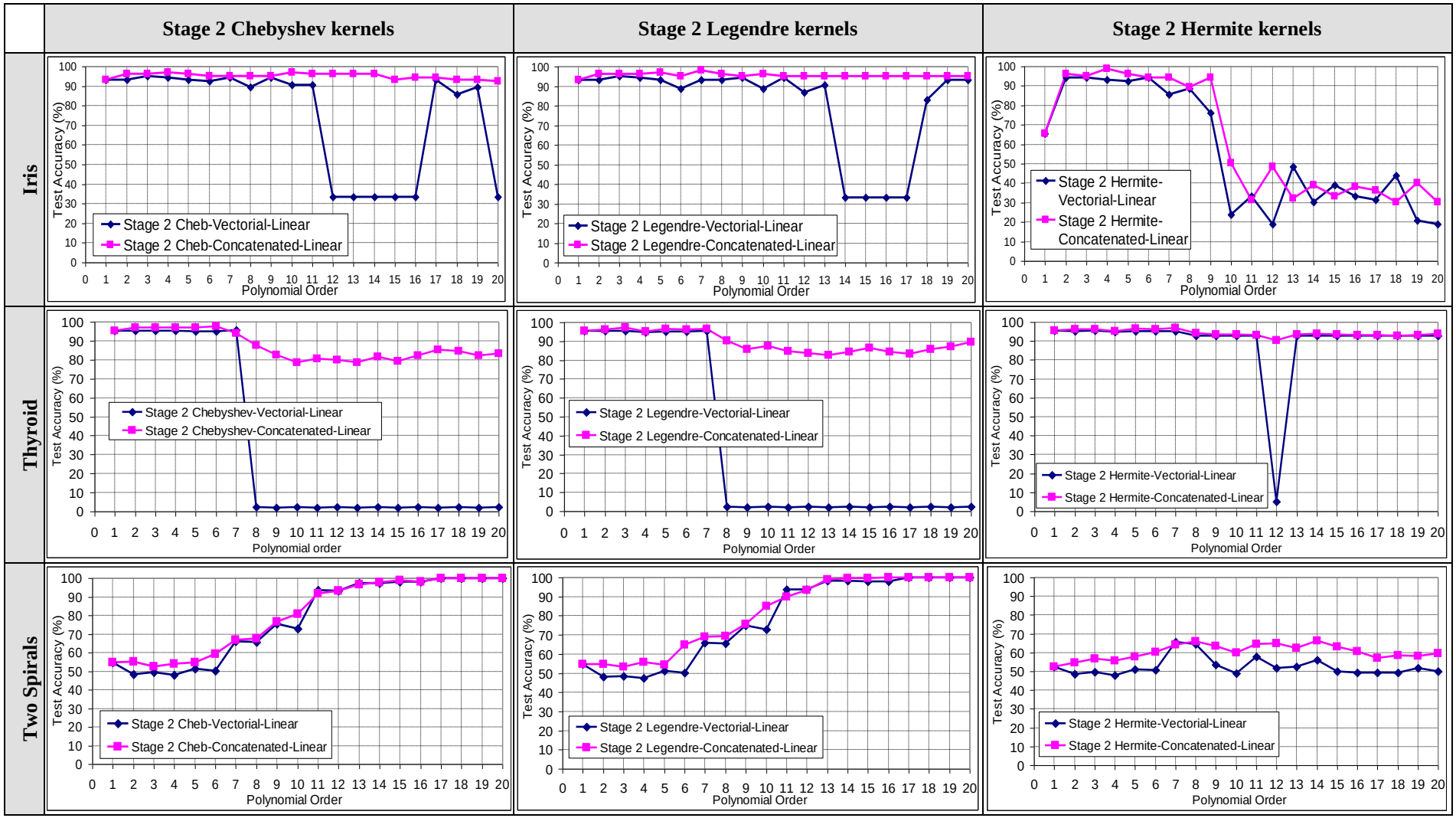

Stage 2 Chebyshev kernels
Stage 2 Legendre kernels
Stage 2 Hermite kernels
Iris
Thyroid
Two Spirals
Test Accuracy (%)
Polynomial Order
Polynomial order
Stage 2 Cheb-Vectorial-Linear
Stage 2 Cheb-Concatenated-Linear
Stage 2 Legendre-Vectorial-Linear
Stage 2 Legendre-Concatenated-Linear
Stage 2 Hermite-Vectorial-Linear
Stage 2 Hermite-Concatenated-Linear
Stage 2 Chebyshev-Vectorial-Linear
Stage 2 Chebyshev-Concatenated-Linear

Table 6.10 Quantitative assessment of the average improvements in the scored classification accuracy of Stage 2 concatenated-linear and vectorial-linear kernels.

		Stage 2 Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	50.320	97.868	96.802	97.228	95.522	62.900	63.539	62.900	94.670	62.900	63.326	62.900	63.326	62.900	94.670	62.900	62.900	94.030	62.900	62.900
	N2	50.320	97.228	97.441	96.802	95.949	95.522	95.522	94.670	93.390	93.603	92.751	92.537	92.324	92.324	92.324	92.537	91.898	91.898	91.898	91.258
	I	0.000	-0.640	0.640	-0.426	0.426	**32.623**	31.983	31.770	-1.279	30.704	29.424	29.638	28.998	29.424	**-2.345**	29.638	-2.132	28.998	-1.066	28.358
	AI	**16.237 (15.82)**																			
Image Segmentation	N1	92.143	92.095	91.714	91.238	14.905	18.381	15.000	16.857	14.619	16.143	14.524	15.524	14.476	15.333	14.619	15.190	14.619	15.095	14.619	15.000
	N2	92.143	92.000	91.429	92.857	93.238	92.619	93.000	93.095	93.333	93.286	93.857	93.476	93.381	93.143	93.286	92.762	92.667	92.810	92.048	91.667
	I	0.000	-0.095	**-0.286**	1.619	78.333	74.238	78.000	76.238	78.714	77.143	**79.333**	77.952	78.905	77.810	78.667	77.571	78.048	77.714	77.429	76.667
	AI	**62.200 (31.77)**																			

		Stage 2 Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	50.320	97.868	97.015	96.162	95.309	62.900	95.096	62.900	94.670	62.900	93.603	62.900	63.326	62.900	94.670	62.900	94.030	62.900	63.326	62.900
	N2	50.320	97.655	97.228	96.375	95.949	95.736	95.736	95.309	95.096	94.456	94.883	94.883	94.456	94.243	94.670	94.456	94.243	94.456	94.243	93.603
	I	0.000	**-0.213**	0.213	0.213	0.640	**32.836**	0.640	32.409	0.426	31.557	1.279	31.983	31.130	31.343	0.000	31.557	0.213	31.557	30.917	30.704
	AI	**15.970 (16.044)**																			
Image Segmentation	N1	92.143	92.095	91.571	92.048	90.810	18.429	15.000	16.857	14.667	16.143	14.524	15.524	14.476	15.381	14.619	15.190	14.619	15.095	14.619	15.000
	N2	92.143	91.905	91.524	92.429	92.667	93.095	92.524	92.810	93.714	93.810	93.905	94.286	93.714	93.762	93.429	93.667	93.524	93.667	93.762	93.952
	I	0.000	**-0.190**	-0.048	0.381	1.857	74.667	77.524	75.952	79.048	77.667	**79.381**	78.762	79.238	78.381	78.810	78.476	78.905	78.571	79.143	78.952
	AI	**58.774 (34.599)**																			

		Stage 2 Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	34.328	97.655	97.868	96.588	95.096	62.900	88.699	62.900	84.222	82.942	82.516	86.141	76.972	77.825	72.495	71.002	70.362	69.296	63.326	64.179
	N2	34.328	97.441	95.949	97.015	94.883	92.751	89.339	93.817	89.126	92.111	91.898	90.192	90.618	92.964	91.471	92.537	92.111	91.471	91.684	89.979
	I	0.000	-0.213	**-1.919**	0.426	-0.213	29.851	0.640	**30.917**	4.904	9.168	9.382	4.051	13.646	15.139	18.977	21.535	21.748	22.175	28.358	25.800
	AI	**12.719 (11.476)**																			
Image Segmentation	N1	92.190	92.095	91.667	92.095	15.381	17.095	14.619	21.190	17.048	15.095	18.286	15.286	14.667	14.333	14.190	14.238	13.429	15.810	15.905	17.952
	N2	92.190	91.762	92.524	91.190	91.476	91.238	92.381	15.000	14.190	15.429	14.190	15.143	14.143	14.429	13.905	14.143	15.048	14.238	15.286	14.190
	I	0.000	-0.333	0.857	-0.905	76.095	74.143	**77.762**	**-6.190**	-2.857	0.333	-4.095	-0.143	-0.524	0.095	-0.286	-0.095	1.619	-1.571	-0.619	-3.762
	AI	**10.476 (28.307)**																			

Iris				Ionosphere						
AI	I	N2	N1	AI	I	N2	N1	Order no.		
20.667 (32.037)	**0.000**	93.333	93.333	22.914 (29.339)	0.000	93.377	93.377	1	Stage 2 Chebyshev kernels	
	2.857	96.190	93.333		**-1.325**	95.364	96.689	2		
	0.952	96.190	95.238		0.662	94.702	94.040	3		
	2.857	97.143	94.286		0.000	94.040	94.040	4		
	2.857	96.190	93.333		76.821	94.702	17.881	5		
	2.857	95.238	92.381		8.609	94.040	85.430	6		
	0.952	95.238	94.286		10.596	95.364	84.768	7		
	5.714	95.238	89.524		11.921	97.351	85.430	8		
	0.952	95.238	94.286		12.583	97.351	84.768	9		
	6.667	97.143	90.476		11.921	97.351	85.430	10		
	5.714	96.190	90.476		12.583	97.351	84.768	11		
	62.857	96.190	33.333		11.921	97.351	85.430	12		
	62.857	96.190	33.333		12.583	97.351	84.768	13		
	62.857	96.190	33.333		11.921	97.351	85.430	14		
	60.000	93.333	33.333		**80.132**	98.013	17.881	15		
	60.952	94.286	33.333		12.583	98.013	85.430	16		
	0.952	94.286	93.333		80.132	98.013	17.881	17		
	7.619	93.333	85.714		12.583	98.013	85.430	18		
	3.810	93.333	89.524		80.132	98.013	17.881	19		
	59.048	92.381	33.333		11.921	97.351	85.430	20		
15.524 (23.983)	**0.000**	93.333	93.333	22.914 (28.773)	0.000	93.377	93.377	1	Stage 2 Legendre kernels	
	2.857	96.190	93.333		**0.000**	96.689	96.689	2		
	0.952	96.190	95.238		1.987	96.026	94.040	3		
	1.905	96.190	94.286		0.662	96.026	95.364	4		
	3.810	97.143	93.333		77.483	95.364	17.881	5		
	6.667	95.238	88.571		11.258	96.689	85.430	6		
	4.762	98.095	93.333		11.921	96.689	84.768	7		
	2.857	96.190	93.333		11.921	97.351	85.430	8		
	0.952	96.190	94.286		12.583	97.351	84.768	9		
	7.619	96.190	88.571		11.921	97.351	85.430	10		
	0.952	95.238	94.286		12.583	97.351	84.768	11		
	8.571	95.238	86.667		12.583	98.013	85.430	12		
	4.762	95.238	90.476		11.921	96.689	84.768	13		
	61.905	95.238	33.333		11.921	97.351	85.430	14		
	61.905	95.238	33.333		78.146	96.026	17.881	15		
	61.905	95.238	33.333		10.596	96.026	85.430	16		
	61.905	95.238	33.333		78.146	96.026	17.881	17		
	12.381	95.238	82.857		11.258	96.689	85.430	18		
	1.905	95.238	93.333		**79.470**	97.351	17.881	19		
	1.905	95.238	93.333		11.921	97.351	85.430	20		
5.381 (11.575)	0.000	65.714	65.715	25.960 (31.995)	0.000	94.040	94.040	1	Stage 2 Hermite kernels	
	1.905	96.190	94.286		-0.662	96.689	97.351	2		
	0.952	95.238	94.286		1.325	96.689	95.364	3		
	5.714	99.048	93.333		9.272	94.702	85.430	4		
	3.810	96.190	92.381		**72.185**	90.066	17.881	5		
	0.000	94.286	94.286		2.649	88.079	85.430	6		
	8.571	94.286	85.714		-2.649	82.119	84.768	7		
	0.952	89.524	88.571		-6.623	78.808	85.430	8		
	18.095	94.286	76.190		-3.974	80.795	84.768	9		
	26.667	50.476	23.810		58.940	76.821	17.881	10		
	-1.905	31.429	33.333		61.589	79.470	17.881	11		
	29.524	48.571	19.048		56.954	74.834	17.881	12		
	-16.190	32.381	48.571		58.940	76.821	17.881	13		
	8.571	39.048	30.476		**-9.934**	75.497	85.430	14		
	-5.714	33.333	39.048		54.967	72.848	17.881	15		
	4.762	38.095	33.333		-9.934	75.497	85.430	16		
	4.762	36.190	31.429		56.954	74.834	17.881	17		
	-13.333	30.476	43.810		58.278	76.159	17.881	18		
	19.048	40.000	20.952		60.927	78.808	17.881	19		
	11.429	30.476	19.048		0.000	17.881	17.881	20		

		Stage 2 Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	95.478	95.449	95.420	95.420	95.012	95.012	95.683	2.392	2.159	2.363	2.159	2.363	2.159	2.363	2.159	2.363	2.159	2.334	2.159	2.334
	N2	95.478	96.849	97.054	97.141	97.141	97.666	93.958	87.546	82.651	78.467	80.463	80.013	78.561	81.645	79.255	82.155	85.249	84.625	82.216	83.217
	I	0	1.400	1.634	1.721	2.130	2.655	**-1.725**	**85.154**	80.493	76.104	78.304	77.650	76.403	79.282	77.096	79.792	83.090	82.291	80.057	80.883
	AI	**52.221 (38.543)**																			
Two Spirals	N1	54.874	48.124	49.519	48.032	51.190	50.046	65.995	65.652	75.378	72.746	93.684	93.158	97.368	97.368	97.986	97.986	100	100	100	100
	N2	54.874	55.257	52.266	53.955	54.863	59.266	66.927	67.532	76.527	80.863	91.563	93.265	96.565	97.855	98.853	98.126	100	100	100	100
	I	0.000	7.133	2.746	5.922	3.673	**9.220**	0.931	1.880	1.149	8.117	**-2.121**	0.107	-0.803	0.487	0.867	0.139	0	0	0	0
	AI	**1.972 (3.179)**																			
AAI		**29.369 (33.787)**																			
TAI																					

		Stage 2 Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	95.478	95.449	95.391	94.866	95.070	95.012	95.537	2.3921	2.159	2.363	2.159	2.363	2.159	2.363	2.159	2.363	2.159	2.334	2.159	2.334
	N2	95.478	96.020	97.054	95.129	96.499	96.185	96.682	90.382	85.832	87.534	84.726	83.654	82.524	84.545	86.355	84.532	83.326	85.632	87.224	89.512
	I	**0**	0.571	1.663	0.263	1.429	1.173	1.145	**87.990**	83.674	85.171	82.567	81.291	80.365	82.182	84.197	82.170	81.167	83.299	85.066	87.179
	AI	**54.628 (40.499)**																			
Two Spirals	N1	54.874	48.124	48.467	47.506	51.190	50.046	65.995	65.652	74.851	72.746	93.684	93.684	98.421	98.421	97.986	97.986	100	100	100	100
	N2	54.874	54.565	53.348	55.786	54.456	64.756	69.124	69.439	75.636	85.110	89.915	93.206	98.851	99.522	99.535	100.000	100.000	100.000	100.000	100.000
	I	0.000	6.442	4.881	8.281	3.266	**14.711**	3.128	3.787	0.785	12.364	**-3.769**	-0.478	0.430	1.101	1.549	2.014	0	0	0	0
	AI	**2.925 (4.53)**																			
AAI		**28.456 (34.059)**																			
TAI																					

		Stage 2 Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	95.478	95.216	95.449	94.749	95.303	95.362	95.274	92.707	92.707	92.707	92.707	5.163	92.678	92.707	92.707	92.707	92.707	92.707	92.678	92.649
	N2	95.478	96.149	96.254	95.365	96.462	96.255	96.865	94.265	93.565	93.564	92.981	90.465	93.545	93.752	93.524	93.256	93.027	92.826	93.027	93.975
	I	**0**	0.934	0.804	0.616	1.159	0.893	1.591	1.558	0.857	0.857	0.274	**85.301**	0.867	1.045	0.816	0.549	0.319	0.119	0.349	1.326
	AI	**5.012 (18.903)**																			
Two Spirals	N1	52.494	48.558	49.519	48.032	51.190	50.664	65.469	64.600	53.455	48.902	57.666	51.625	52.403	56.178	49.863	49.336	49.336	49.336	51.716	50.046
	N2	52.494	54.565	56.585	55.565	57.646	60.265	64.265	65.849	63.546	59.968	64.546	65.026	62.496	66.412	63.265	60.596	57.233	58.464	58.026	59.465
	I	0	6.007	7.065	7.533	6.456	9.601	**-1.204**	1.249	10.091	11.066	6.881	**13.402**	10.094	10.233	13.402	11.260	7.896	9.128	6.309	9.419
	AI	**7.794 (3.993)**																			
AAI		**11.224 (21.085)**																			
TAI																					

N1 Classification accuracy of Stage 2 kernels implemented using the vectorial approach & the Linear dot-product kernel
N2 Classification accuracy of Stage 2 kernels implemented using the concatenated approach & the Linear dot-product kernel
I Improvement = N2-N1
AI Average Improvement for the first 20 orders
AAI Average of Average Improvements of the first 20 orders over all the datasets under investigation
TAI Total Average Improvement over all datasets and all kernels under investigation for the first 20 orders

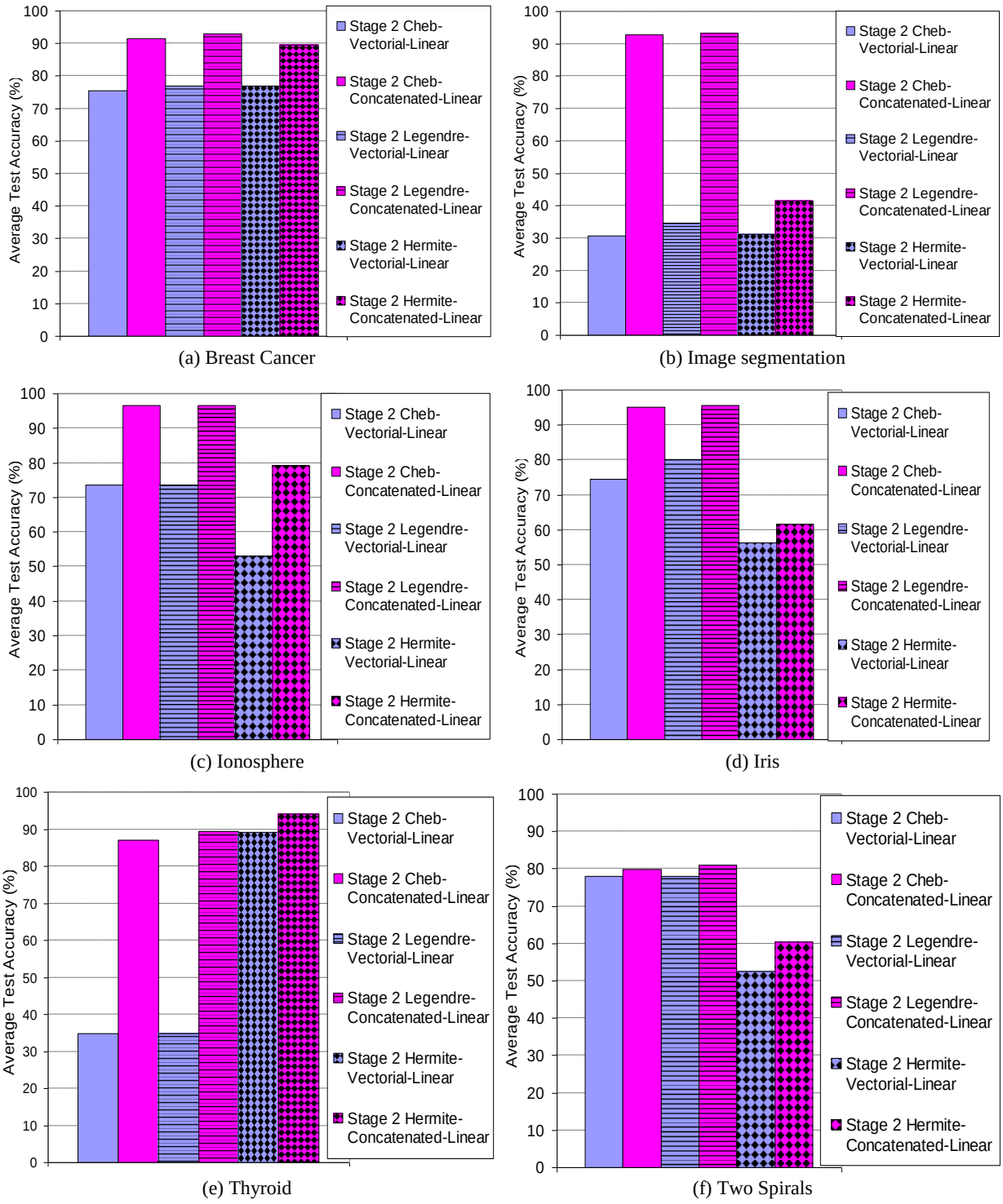
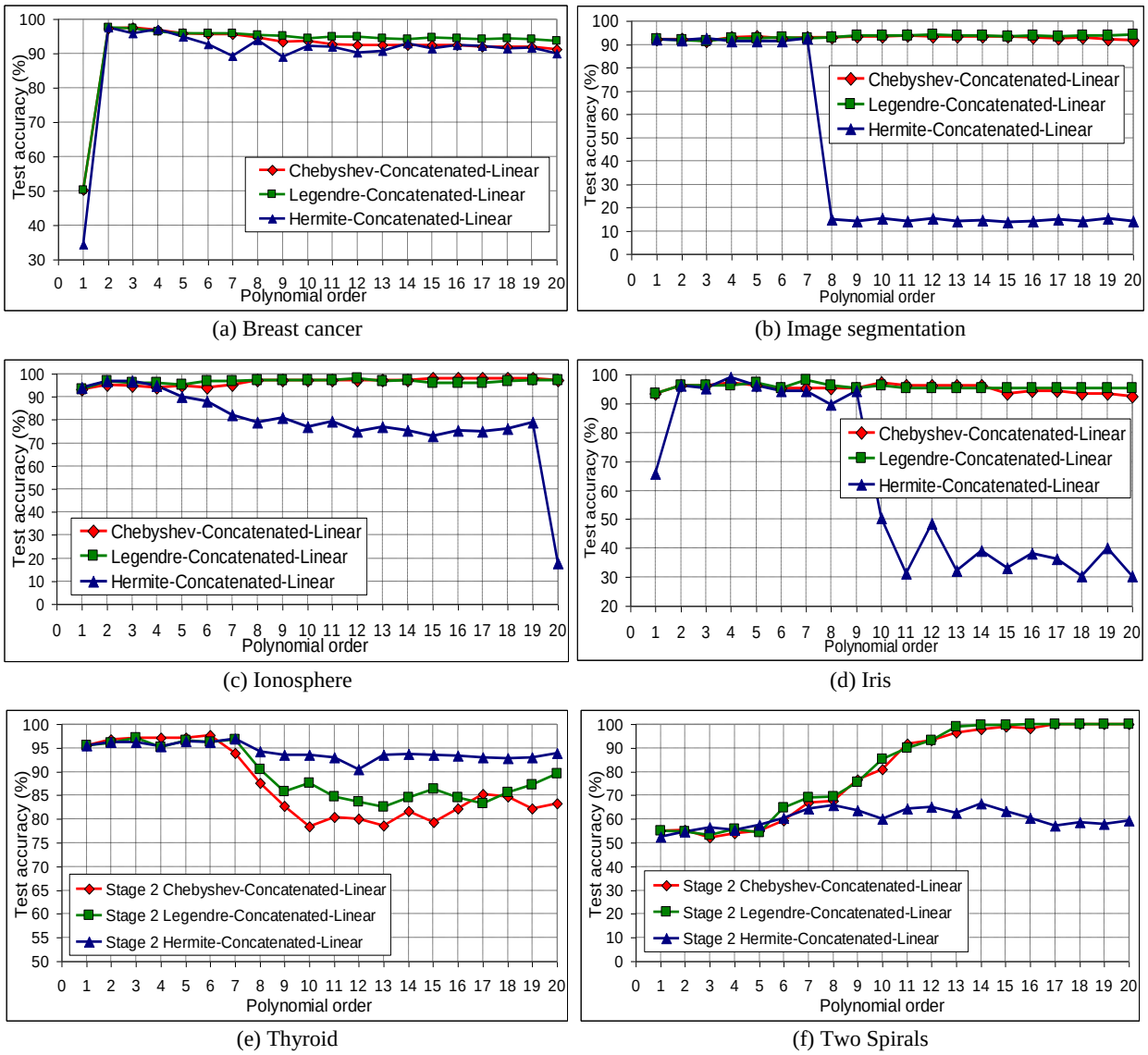


Figure 6.11 Bar chart comparative assessment of the average classification accuracy scored by the first 20 orders of the Stage 2 concatenated-Linear and vectorial-Linear kernels.

As the proposed concatenated approach provides a more meaningful processing technique to the data transformation perspective introduced in this chapter than its previously proposed vectorial counterpart (due to the reasons explained in Section 6.4.1), one can utilize the achieved concatenated results to establish a comparative analysis between the employed polynomials under study (i.e., the Chebyshev, Legendre, and Hermite polynomials) within the context of their exhibited similarity measure shape characteristics, explained in Sections 3.4.1 and 4.3. To achieve this purpose, Figure 6.12 studies the results obtained from these Stage 2 concatenated kernels separately for each of the datasets experimented.

On investigating these results, one can observe the quite similar performance of the Stage 2 Chebyshev and Legendre kernels, which were actually able to score higher classification accuracies compared to their Stage 2 Hermite counterparts, especially in higher polynomial orders. Inline with the analysis of the shape characteristics of these Stage 2 kernels in Section 3.4.2, which revealed that both of the Chebyshev and Legendre kernels exhibit common similarity measure characteristics that are more accurate than the Hermite kernels, the results observed herein do indeed validate that the better similarity measures acquired by the Chebyshev and Legendre kernels are reflected accordingly on their classification performance compared to the Hermite kernels.

However, similar to the observation drawn from the experimental results of the Stage 2 vectorial-linear kernels back in Section 4.4.4, one can still notice that the Thyroid dataset remains an interesting exception to this rule, as shown in Figure 6.12 (e), where the classification performance of the Stage 2 concatenated-linear Hermite kernel outperforms its Chebyshev and Legendre counterparts. This consistent observation, therefore, strengthens the earlier implication that there must be other factors which can at times have a stronger influence on the classification performance than the similarity measure shape characteristics of the underpinning kernels, which can be an interesting subject of future research.



(a) Breast cancer

(b) Image segmentation

(c) Ionosphere

(d) Iris

(e) Thyroid

(f) Two Spirals

Figure 6.12 Classification performance comparison of Stage 2 kernels implemented using the proposed concatenated approach on the various datasets experimented.

Similar to the experimental investigation conducted in Section 6.5.2.1 on the Stage 1 kernels, the second set of experiments presented in this section is also aimed at assessing the effectiveness of evaluating the Gaussian kernel, instead of the linear kernel, on the polynomial-processed inputs. This is achieved by comparing the results obtained from the evaluation of the Gaussian kernel on the concatenated-processed vectors against those obtained from the linear kernel evaluations on the vectorial-processed inputs, using the kernels illustrated in Table 6.8.

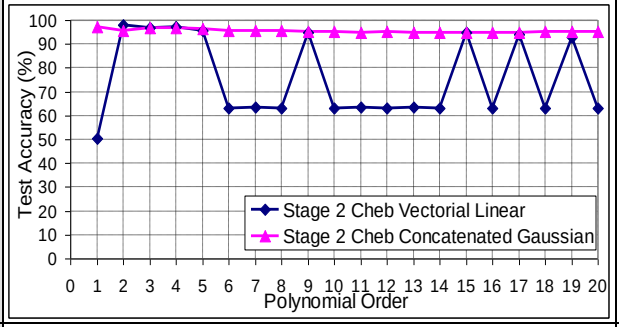
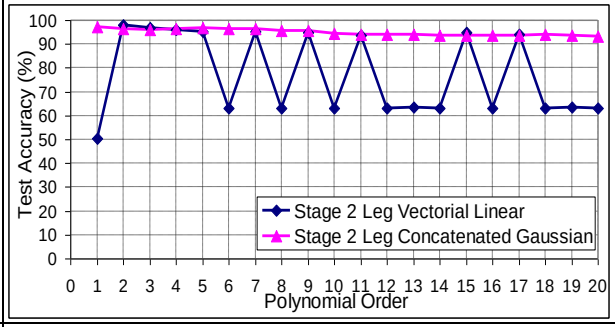
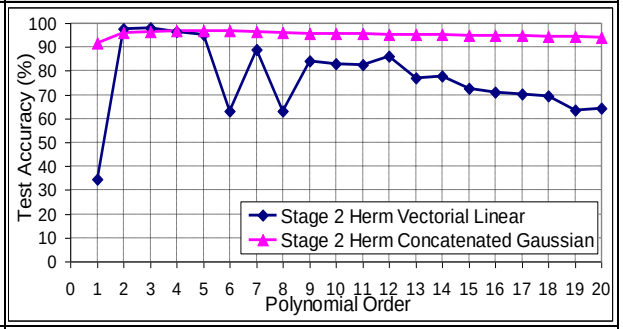
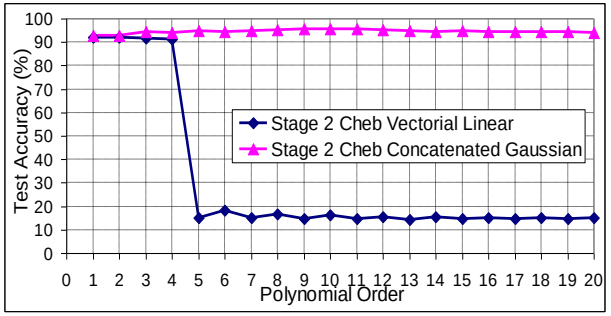
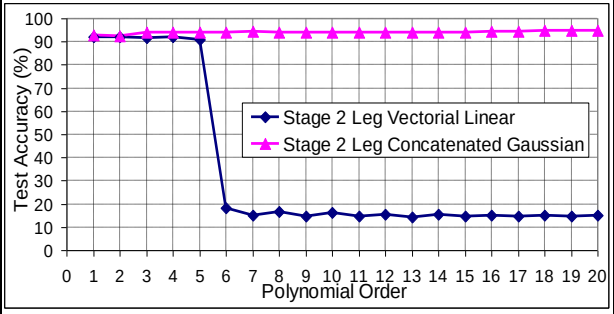
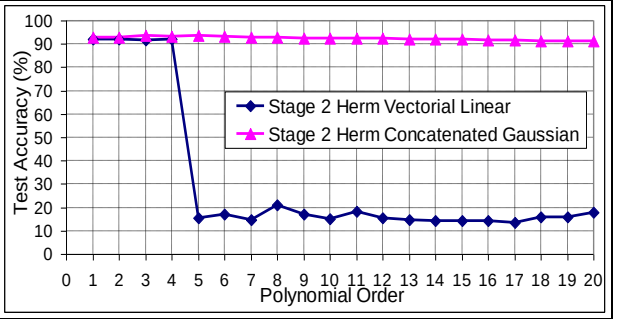
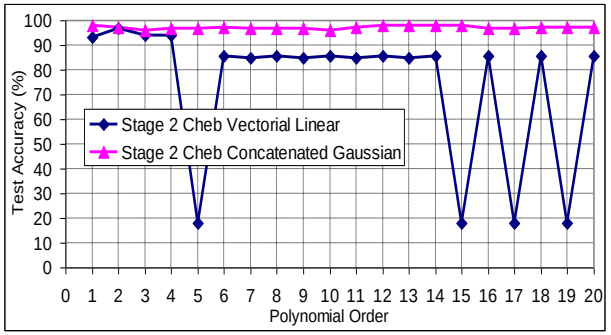
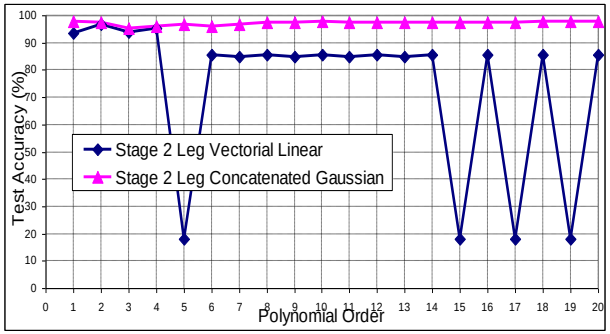
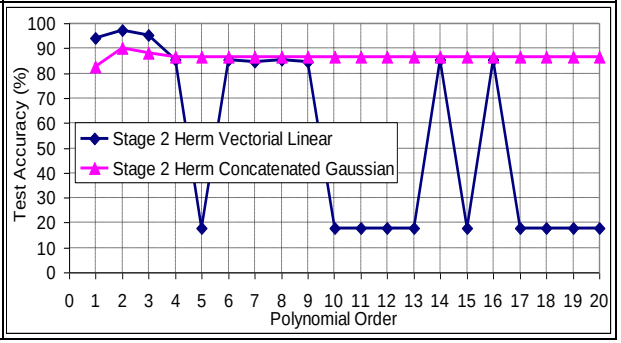
Table 6.11 illustrates the graphical comparative results of the highest classification accuracy scored in each of these two cases for the Chebyshev, Legendre, and Hermite polynomials, whereas the average accuracy scored by the first 20 orders is also shown in Figure 6.13. Inline with the previous results obtained in this section from the concatenated-linear kernels, one can also observe that the concatenated-Gaussian kernels were actually able to achieve even better classification performance than their vectorial-linear counterparts. By comparing the two sets of graphical results in Tables 6.9 and 6.11, this observation can be evidently realized, for example, from the classification performance of the Hermite kernels in the Breast Cancer, Image Segmentation, and Iris datasets, especially in higher polynomial orders. The Chebyshev and Legendre kernels have also exhibited some improvements in the other datasets experimented, which are relatively steady along the spectrum of polynomial orders. However, they are not as significant as their sister Hermite kernels.

The overall improvement introduced by the evaluation of the Gaussian kernel on the concatenated-processed vectors can therefore be better assessed quantitatively by calculating the average accuracy improvement factors as shown in Table 6.12. These were found to be equal to 32.287%, 30.446%, and 29.657% for the Chebyshev, Legendre, and Hermite polynomials, respectively; making up a total average accuracy improvement factor of 30.796% depending on the kernel and the dataset used. Compared to its corresponding average improvement factor using the concatenated-linear kernel (23.016%), this result also clearly demonstrates the effectiveness of utilizing the Gaussian instead of the linear kernel by introducing a further average improvement factor of 7.78%. This superiority of the Gaussian kernel over its linear counterpart, when evaluated on the polynomial-processed inputs, is also believed to stem from its renowned ability of mapping the input space to an infinite dimensional feature space, as well as it accurate similarity measure shape characteristics, as explained earlier in Section 6.2.

Inline with the behaviour observed in the Stage 1 concatenated-Gaussian kernels in Section 6.5.2.1, one can also realize that the Stage 2 concatenated-Gaussian kernels have also demonstrated a lot smoother classification performance than the Stage 2 concatenated-linear kernels, across the spectrum of polynomial orders. This behaviour can again be easily realized by comparing the two sets of results from Tables 6.11 and 6.9, respectively, which is quite evident, especially in the Breast Cancer, Image Segmentation, and Iris datasets where the abrupt oscillating changes in the classification accuracy between the polynomial orders are significantly reduced. Consequently, this resulted in even more dispersed improvements in the classification accuracies between the Stage 2 concatenated-Gaussian and the Stage 2 vectorial-linear kernels, which can also be quantitatively evaluated from the standard deviation of the average improvement factors (shown in brackets), as well as the minimum and maximum improvement factors (shown in bold) calculated in Table 6.12.

.

Table 6.11 Comparative experimental results of Stage 2 concatenated-Gaussian and vectorial-Linear kernels.

	Stage 2 Chebyshev kernels	Stage 2 Legendre kernels	Stage 2 Hermite kernels
Breast Cancer			
Image Segmentation			
Ionosphere			

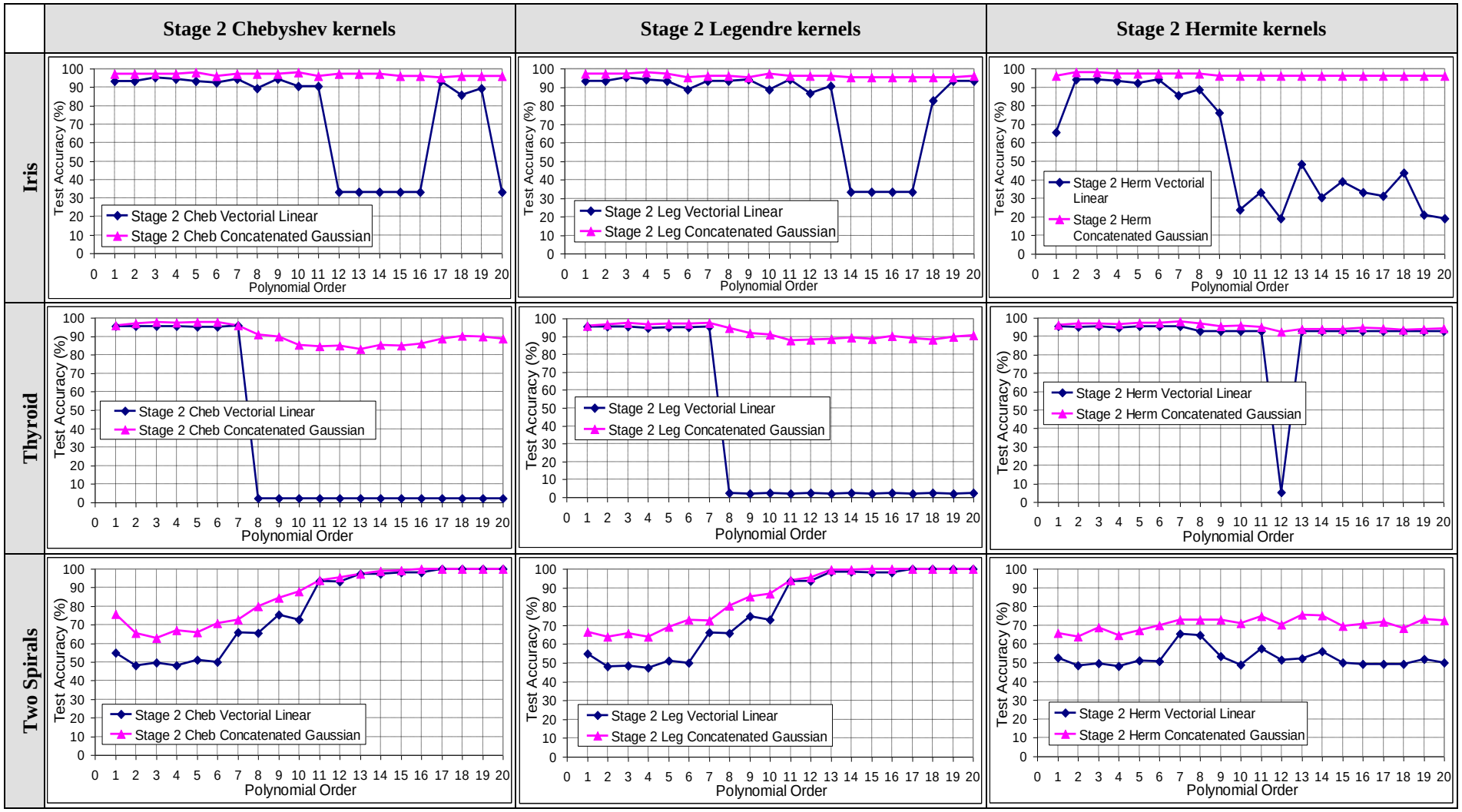

Stage 2 Chebyshev kernels
Stage 2 Legendre kernels
Stage 2 Hermite kernels
Iris
Thyroid
Two Spirals
Test Accuracy (%)
Polynomial Order
Stage 2 Cheb Vectorial Linear
Stage 2 Cheb Concatenated Gaussian
Stage 2 Leg Vectorial Linear
Stage 2 Leg Concatenated Gaussian
Stage 2 Herm Vectorial Linear
Stage 2 Herm Concatenated Gaussian

Table 6.12 Quantitative assessment of the average improvements in the scored classification accuracy of Stage 2 concatenated-Gaussian and vectorial-Linear kernels.

		Stage 2 Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	50.320	97.868	96.802	97.228	95.522	62.900	63.539	62.900	94.670	62.900	63.326	62.900	63.326	62.900	94.670	62.900	94.030	62.900	92.964	62.900
	N2	97.228	95.522	97.015	96.802	96.588	95.522	95.522	95.522	95.309	95.096	94.883	95.096	94.883	94.883	94.883	94.670	94.670	95.096	95.309	95.096
	I	**46.908**	**-2.345**	0.213	-0.426	1.066	32.623	31.983	32.623	0.640	32.196	31.557	32.196	31.557	31.983	0.213	31.770	0.640	32.196	2.345	32.196
	AI	**20.107 (16.936)**																			
Image Segmentation	N1	92.143	92.095	91.714	91.238	14.905	18.381	15.000	16.857	14.619	16.143	14.524	15.524	14.476	15.333	14.619	15.190	14.619	15.095	14.619	15.000
	N2	92.952	92.857	94.476	94.143	94.762	94.619	94.762	95.286	95.619	95.810	95.619	95.048	94.905	94.524	94.714	94.333	94.571	94.524	94.429	94.048
	I	0.810	**0.762**	2.762	2.905	79.857	76.238	79.762	78.429	81.000	79.667	**81.095**	79.524	80.429	79.190	80.095	79.143	79.952	79.429	79.810	79.048
	AI	**63.995 (31.919)**																			

		Stage 2 Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	50.320	97.868	97.015	96.162	95.309	62.900	95.096	62.900	94.670	62.900	93.603	62.900	63.326	62.900	94.670	62.900	94.030	62.900	63.326	62.900
	N2	97.228	96.588	96.162	96.375	97.015	96.588	96.375	95.736	95.736	94.456	94.243	94.243	94.243	93.817	93.817	93.817	93.603	94.030	93.817	93.390
	I	**46.908**	**-1.279**	-0.853	0.213	1.706	33.689	1.279	32.836	1.066	31.557	0.640	31.343	30.917	30.917	-0.853	30.917	-0.426	31.130	30.490	30.490
	AI	**18.134 (17.045)**																			
Image Segmentation	N1	92.143	92.095	91.571	92.048	90.810	18.429	15.000	16.857	14.667	16.143	14.524	15.524	14.476	15.381	14.619	15.190	14.619	15.095	14.619	15.000
	N2	92.952	92.571	93.857	93.952	93.952	94.048	94.429	94.048	94.048	93.905	93.857	94.000	94.000	93.857	94.143	94.286	94.333	94.714	94.714	94.667
	I	0.810	**0.476**	2.286	1.905	3.143	75.619	79.429	77.190	79.381	77.762	79.333	78.476	79.524	78.476	79.524	79.095	79.714	79.619	**80.095**	79.667
	AI	**59.576 (34.288)**																			

		Stage 2 Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	34.328	97.655	97.868	96.588	95.096	62.900	88.699	62.900	84.222	82.942	82.516	86.141	76.972	77.825	72.495	71.002	70.362	69.296	63.326	64.179
	N2	91.471	96.162	96.375	96.802	96.802	96.802	96.375	95.949	95.736	95.736	95.522	95.309	95.096	95.096	94.883	94.883	94.670	94.456	94.456	94.243
	I	**57.143**	**-1.493**	-1.493	0.213	1.706	33.902	7.676	33.049	11.514	12.793	13.006	9.168	18.124	17.271	22.388	23.881	24.307	25.160	31.130	30.064
	AI	**18.475 (14.673)**																			
Image Segmentation	N1	92.190	92.095	91.667	92.095	15.381	17.095	14.619	21.190	17.048	15.095	18.286	15.286	14.667	14.333	14.190	14.238	13.429	15.810	15.905	17.952
	N2	92.810	92.952	93.524	93.381	93.524	93.333	92.905	92.857	92.571	92.381	92.286	92.286	92.143	92.095	91.905	91.619	91.571	91.333	91.190	91.143
	I	**0.619**	0.857	1.857	1.286	78.143	76.238	**78.286**	71.667	75.524	77.286	74.000	77.000	77.476	77.762	77.714	77.381	78.143	75.524	75.286	73.190
	AI	**61.262 (30.884)**																			

Iris				Ionosphere						
AI	I	N2	N1	AI	I	N2	N1	Order no.		
22.333 (31.543)	3.810	97.143	93.333	23.543 (28.866)	4.636	98.013	93.377	1	Order no.	Stage 2 Chebyshev kernels
	3.810	97.143	93.333		**0.662**	97.351	96.689	2		
	1.905	97.143	95.238		1.987	96.026	94.040	3		
	2.857	97.143	94.286		2.649	96.689	94.040	4		
	4.762	98.095	93.333		78.808	96.689	17.881	5		
	3.810	96.190	92.381		11.921	97.351	85.430	6		
	2.857	97.143	94.286		11.921	96.689	84.768	7		
	7.619	97.143	89.524		11.258	96.689	85.430	8		
	2.857	97.143	94.286		11.921	96.689	84.768	9		
	7.619	98.095	90.476		10.596	96.026	85.430	10		
	5.714	96.190	90.476		12.583	97.351	84.768	11		
	63.810	97.143	33.333		12.583	98.013	85.430	12		
	63.810	97.143	33.333		13.245	98.013	84.768	13		
	63.810	97.143	33.333		12.583	98.013	85.430	14		
	62.857	96.190	33.333		**80.132**	98.013	17.881	15		
	62.857	96.190	33.333		11.258	96.689	85.430	16		
	1.905	95.238	93.333		78.808	96.689	17.881	17		
	10.476	96.190	85.714		11.921	97.351	85.430	18		
	6.667	96.190	89.524		79.470	97.351	17.881	19		
	62.857	96.190	33.333		11.921	97.351	85.430	20		
16.048 (23.695)	3.810	97.143	93.333	23.543 (28.996)	4.636	98.013	93.377	1	Order no.	Stage 2 Legendre kernels
	3.810	97.143	93.333		**0.662**	97.351	96.689	2		
	1.905	97.143	95.238		1.325	95.364	94.040	3		
	3.810	98.095	94.286		0.662	96.026	95.364	4		
	3.810	97.143	93.333		78.808	96.689	17.881	5		
	6.667	95.238	88.571		10.596	96.026	85.430	6		
	2.857	96.190	93.333		11.921	96.689	84.768	7		
	2.857	96.190	93.333		11.921	97.351	85.430	8		
	0.952	95.238	94.286		12.583	97.351	84.768	9		
	8.571	97.143	88.571		12.583	98.013	85.430	10		
	1.905	96.190	94.286		12.583	97.351	84.768	11		
	9.524	96.190	86.667		11.921	97.351	85.430	12		
	5.714	96.190	90.476		12.583	97.351	84.768	13		
	61.905	95.238	33.333		11.921	97.351	85.430	14		
	61.905	95.238	33.333		79.470	97.351	17.881	15		
	61.905	95.238	33.333		11.921	97.351	85.430	16		
	61.905	95.238	33.333		79.470	97.351	17.881	17		
	12.381	95.238	82.857		12.583	98.013	85.430	18		
	1.905	95.238	93.333		**80.132**	98.013	17.881	19		
	2.857	96.190	93.333		12.583	98.013	85.430	20		
40.238 (29.663)	30.476	96.190	65.715	33.676 (36.277)	**-11.258**	82.781	94.040	1	Order no.	Stage 2 Hermite kernels
	3.810	98.095	94.286		-7.285	90.066	97.351	2		
	3.810	98.095	94.286		-7.285	88.079	95.364	3		
	3.810	97.143	93.333		1.325	86.755	85.430	4		
	4.762	97.143	92.381		**68.874**	86.755	17.881	5		
	2.857	97.143	94.286		1.325	86.755	85.430	6		
	11.429	97.143	85.714		1.987	86.755	84.768	7		
	8.571	97.143	88.571		1.325	86.755	85.430	8		
	20.000	96.190	76.190		1.987	86.755	84.768	9		
	72.381	96.190	23.810		68.874	86.755	17.881	10		
	62.857	96.190	33.333		68.874	86.755	17.881	11		
	77.143	96.190	19.048		68.874	86.755	17.881	12		
	47.619	96.190	48.571		68.874	86.755	17.881	13		
	65.714	96.190	30.476		1.325	86.755	85.430	14		
	57.143	96.190	39.048		68.874	86.755	17.881	15		
	62.857	96.190	33.333		1.325	86.755	85.430	16		
	64.762	96.190	31.429		68.874	86.755	17.881	17		
	52.381	96.190	43.810		68.874	86.755	17.881	18		
	75.238	96.190	20.952		68.874	86.755	17.881	19		
	77.143	96.190	19.048		68.874	86.755	17.881	20		

		Stage 2 Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	95.478	95.449	95.420	95.420	95.012	95.012	95.683	2.392	2.159	2.363	2.159	2.363	2.159	2.363	2.159	2.363	2.159	2.334	2.159	2.334
	N2	96.02862	96.96354	97.56332	97.45662	97.63254	97.85632	96.02554	91.12554	90.02336	85.23554	84.56321	84.85235	83.02254	85.43255	85.06354	86.23365	88.63256	90.23544	89.95232	88.86321
	I	0.550	1.514	2.143	2.037	2.621	2.845	**0.343**	**88.733**	87.865	82.873	82.405	82.489	80.864	83.070	82.905	83.871	86.474	87.902	87.794	86.529
	AI	**55.791 (40.767)**																			
Two Spirals	N1	54.874	48.124	49.519	48.032	51.190	50.046	65.995	65.652	75.378	72.746	93.684	93.158	97.368	97.368	97.986	97.986	100	100	100	100
	N2	75.867	65.468	62.988	66.867	65.874	70.848	72.898	80.088	84.656	87.866	93.855	95.368	97.458	98.879	99.287	99.837	100	100	100	100
	I	**20.992**	17.344	13.468	18.835	14.684	20.802	6.902	14.435	9.278	15.120	0.171	2.210	0.089	1.510	1.300	1.850	**0**	0	0	0
	AI	**7.95 (8.095)**																			
AAI		**32.287 (33.984)**																			
TAI		**30.796 (32.838)**																			

		Stage 2 Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	95.478	95.449	95.391	94.866	95.070	95.012	95.537	2.392	2.159	2.363	2.159	2.363	2.159	2.363	2.159	2.363	2.159	2.334	2.159	2.334
	N2	95.968	96.858	97.477	96.568	97.088	97.280	97.546	94.533	92.046	91.257	87.846	88.174	88.550	89.379	88.735	90.054	88.846	88.269	89.847	90.638
	I	**0.489**	1.408	2.086	1.702	2.018	2.268	2.009	**92.141**	89.887	88.894	85.687	85.811	86.391	87.016	86.576	87.691	86.687	85.935	87.688	88.304
	AI	**57.534 (42.054)**																			
Two Spirals	N1	54.874	48.124	48.467	47.506	51.190	50.046	65.995	65.652	74.851	72.746	93.684	93.684	98.421	98.421	97.986	97.986	100	100	100	100
	N2	66.586	63.865	65.866	63.864	69.165	73.024	72.578	80.597	85.368	86.787	93.898	95.398	99.787	99.637	100	100	100	100	100	100
	I	11.712	15.742	17.400	16.359	17.976	**22.978**	6.583	14.944	10.516	14.041	0.213	1.713	1.366	1.216	2.014	2.014	**0**	0	0	0
	AI	**7.839 (7.834)**																			
AAI		**30.446 (34.21)**																			
TAI		**30.796 (32.838)**																			

		Stage 2 Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	95.478	95.216	95.449	94.749	95.303	95.362	95.274	92.707	92.707	92.707	92.707	5.163	92.678	92.707	92.707	92.707	92.707	92.707	92.678	92.649
	N2	96.265	96.827	97.022	96.541	97.364	97.531	97.953	96.949	95.299	95.857	95.048	92.596	93.958	93.846	94.039	94.600	94.269	93.635	93.927	94.275
	I	**0.786**	1.611	1.572	1.792	2.061	2.169	2.679	4.242	2.592	3.150	2.341	**87.433**	1.280	1.139	1.332	1.893	1.562	0.927	1.249	1.627
	AI	**6.172 (19.144)**																			
Two Spirals	N1	52.494	48.558	49.519	48.032	51.190	50.664	65.469	64.600	53.455	48.902	57.666	51.625	52.403	56.178	49.863	49.336	49.336	49.336	51.716	50.046
	N2	65.950	63.965	68.646	64.787	67.476	70.047	72.979	73.048	72.840	70.946	74.888	70.265	75.585	75.087	69.446	70.526	71.947	68.565	73.195	72.526
	I	13.455	15.406	19.127	16.755	16.286	19.383	**7.510**	8.448	19.384	22.045	17.222	18.640	**23.183**	18.908	19.584	21.190	22.610	19.228	21.478	22.481
	AI	**18.116 (4.296)**																			
AAI		**29.657 (30.402)**																			
TAI		**30.796 (32.838)**																			

N1 Classification accuracy of Stage 2 kernels implemented using the vectorial approach & the Linear dot-product kernel
N2 Classification accuracy of Stage 2 kernels implemented using the concatenated approach & the Gaussian kernel
I Improvement = N2-N1
AI Average Improvement for the first 20 orders
AAI Average of Average Improvements of the first 20 orders over all the datasets under investigation
TAI Total Average Improvement over all datasets and all kernels under investigation for the first 20 orders

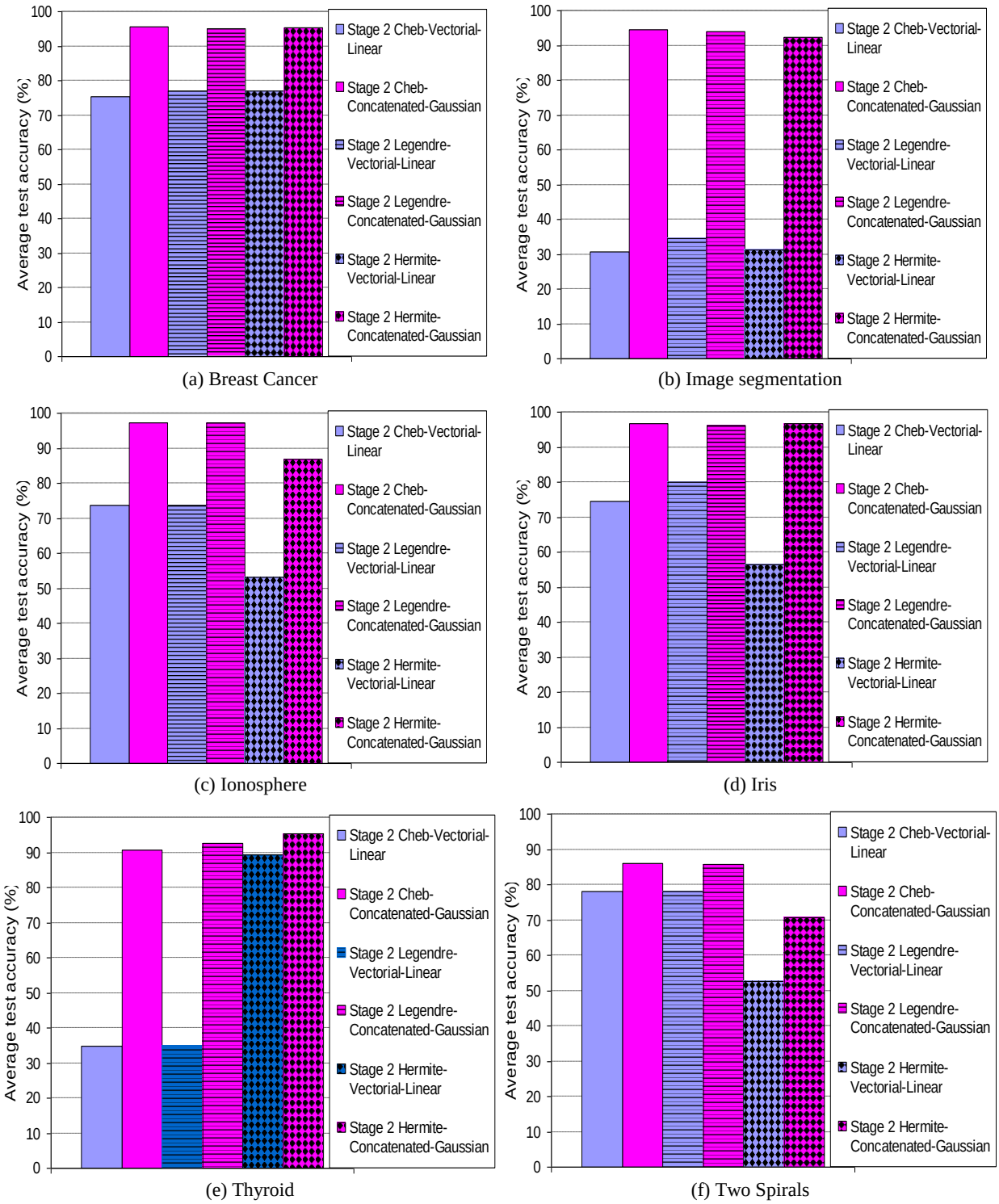


Figure 6.13 Bar chart comparative assessment of the average classification accuracy scored by the first 20 orders of the Stage 2 concatenated-Gaussian and vectorial-Linear kernels.

To further analyse the results obtained from the three different kernel approaches under test (i.e., vectorial-linear, concatenated-linear, and concatenated-Gaussian kernels), Table 6.13 illustrates the maximum classification accuracy scored by each of the Stage 2 kernels experimented in this section for the first 20 orders and for all the datasets under test. As can be observed, for the three different polynomials under study (i.e., Chebyshev, Legendre, and Hermite), the maximum accuracies scored by the concatenated-linear and concatenated-Gaussian kernels are better than those scored by the vectorial-linear kernels, for most of the results. This again demonstrates the effectiveness of the proposed concatenated processing approach as well as the utilization of the Gaussian kernel (instead of the linear kernel), when evaluating the similarities between the polynomial-processed inputs.

Table 6.13. Maximum classification accuracy scored by the Stage 2 linear and Gaussian kernels evaluation on the vectorial- and concatenated-polynomial processed inputs.

		Breast Cancer		**Image Segmentat.**		**Ionosphere**		**Iris**		**Thyroid**		**Two Spirals**	
		Test %	*n*	Test %	*n*	Test %	*n*	Test %	*n*	Test %	*n*	Test %	*n*
Chebyshev	**Vectorial-linear**	97.868	2	92.143	1	96.689	2	95.238	3	95.683	7	100	17
	Concatenat-linear	97.441	3	93.857	11	98.013	15	97.143	4	97.666	6	100	17
	Concatenat-Gauss	97.228	1	95.81	10	98.013	1	98.095	5	97.856	6	100	17
Legendre	**Vectorial-linear**	97.868	2	92.143	1	96.689	2	95.238	3	95.537	7	100	17
	Concatenat-linear	97.655	2	94.286	12	98.013	12	98.095	7	97.054	3	100	16
	Concatenat-Gauss	97.228	1	94.714	18	98.013	1	98.095	4	97.546	7	100	15
Hermite	**Vectorial-linear**	97.868	3	92.190	1	97.351	2	94.286	3	95.478	1	65.469	7
	Concatenat-linear	97.441	2	92.524	3	96.689	2	99.048	4	96.865	7	66.412	14
	Concatenat-Gauss	96.802	5	93.524	3	90.066	2	98.095	2	97.953	7	75.585	13

6.5.2.3 Comparative experimental results on Stage 3 kernels

It has been demonstrated (back in Section 4.3.3) how the fusion by multiplication operation of Stage 2 kernels by another kernel was able to tailor the overall shape of the resulting composite Stage 3 kernels closer to the ideal similarity function, and, hence, enabled the acquisition of more accurate similarity measures, and accordingly enhanced the classification performance. One can notice, however, that although the combining kernel (e.g., the Gaussian) in Stage 3 has got nothing to do with how the polynomials are utilized to process the multi-dimensional vector inputs, it is still important to assess the classification performance of the overall composite Stage 3 kernels adopting the proposed concatenated processing approach in comparison with those using its previously proposed vectorial counterpart.

To achieve this purpose, another two sets of experiments are also conducted in this section, using the three composite kernels shown in Tables 6.14 and 6.15. These are: the 'Modified Chebyshev kernel', the 'Modified Legendre kernel', and the 'Composite Hermite kernel'. Similar to previous sections, the first set of experiments conducted in this section utilizes the evaluation of the linear kernel on both of the vectorial- and concatenated-processed inputs, using the composite kernels depicted in Table 6.14. Whereas the second set of experiments utilizes the evaluation of the Gaussian kernel on the concatenated-processed vectors compared to the linear kernel evaluations on the vectorial-processed inputs, using the same composite kernels, as shown in Table 6.15.

Table 6.14. Mathematical formulations of the composite Stage 3 linear kernel evaluation on the vectorial- and concatenated-processed inputs using the Chebyshev, Legendre, and Hermite polynomials.

	Vectorial-linear kernel	**Concatenated-linear kernel**
Modified Chebyshev kernel	$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)$	$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} < [T_i(x_1),\cdots,T_i(x_m)],[T_i(z_1),\cdots,T_i(z_m)] > \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)$
Modified Legendre kernel	$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle L_i(\mathbf{x}), L_i(\mathbf{z}) \rangle \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)$	$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} < [L_i(x_1),\cdots,L_i(x_m)],[L_i(z_1),\cdots,L_i(z_m)] > \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)$
Composite Hermite kernel	$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle H_i(\mathbf{x}), H_i(\mathbf{z}) \rangle \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)$	$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} < [H_i(x_1),\cdots,H_i(x_m)],[H_i(z_1),\cdots,H_i(z_m)] > \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)$

Table 6.15. Mathematical formulations of the composite Stage 3 Gaussian kernel evaluation on the concatenated-processed vectors versus the linear kernel evaluation on the vectorial-processed inputs using the Chebyshev, Legendre, and Hermite polynomials.

	Vectorial-linear kernel	**Concatenated-Gaussian kernel**
Modified Chebyshev kernel	$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle T_i(\mathbf{x}), T_i(\mathbf{z}) \rangle \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)$	$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma \|[T_i(x_1),\cdots,T_i(x_m)]-[T_i(z_1),\cdots,T_i(z_m)]\|^2\right) \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)$
Modified Legendre kernel	$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle L_i(\mathbf{x}), L_i(\mathbf{z}) \rangle \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)$	$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma \|[L_i(x_1),\cdots,L_i(x_m)]-[L_i(z_1),\cdots,L_i(z_m)]\|^2\right) \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)$
Composite Hermite kernel	$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle H_i(\mathbf{x}), H_i(\mathbf{z}) \rangle \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)$	$k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \exp\left(-\gamma \|[H_i(x_1),\cdots,H_i(x_m)]-[H_i(z_1),\cdots,H_i(z_m)]\|^2\right) \times \exp\left(-\gamma \|\mathbf{x}-\mathbf{z}\|^2\right)$

Table 6.16 illustrates the experimental results of the first set of experiments conducted in this section using the Stage 3 composite kernels shown in Table 6.14. The graphical results demonstrate the best SVM classification accuracy scored by the first 20 orders of each of these composite Stage 3 kernels when the linear kernel is evaluated on both the concatenated and vectorial processing approaches under comparison. Inline with the experimental results obtained from Stage 1 and Stage 2 kernels, it can also be observed from the Stage 3 results that the proposed concatenated approach continues to outperform the vectorial approach for all the composite kernels under comparison and for most, if not all, of the tested polynomial orders and the datasets experimented.

A common observation that can be realized, however, is that the classification performance of most of the Stage 3 vectorial-linear kernels tends to asymptotically decrease as the polynomial order increases, as shown for example in the Breast Cancer, Image Segmentation, and Ionosphere datasets in Table 6.16. The performance of the Stage 3 concatenated-linear kernels, on the other hand, tends to stay relatively stable at a higher level than their vectorial-linear counterparts across the spectrum of polynomial orders. This behaviour results in making the performance improvement introduced by the proposed concatenated approach more significant at higher polynomial orders than at lower orders, although the classification performance demonstrated by the Stage 3 Modified Chebyshev and Legendre kernels in the Two Spirals dataset seems to be an exception to this rule.

One can also evaluate the amount of improvement introduced by the proposed concatenated approach on the Stage 3 kernels by examining the bar chart comparisons, shown in Figure 6.14, which demonstrate the average accuracy scored by the first 20 orders of the Stage 3 concatenated-linear and the vectorial-linear kernels, for each of the datasets experimented. As can be observed, the concatenated approach again demonstrated a consistent superior average accuracy over its vectorial counterpart for all of the Stage 3 kernels under test, but with varying amounts of improvements across the datasets experimented. One can easily notice, however, that due to the fusion by multiplication operation taking place in the Stage 3 kernels, their corresponding classification accuracy has therefore been uplifted for both of the vectorial and concatenated approaches. This results in the amount of improvements scored by the concatenated approach over its vectorial counterpart in the Stage 3 kernels to be relatively smaller than their equivalents in the Stage 1 and Stage 2 kernels.

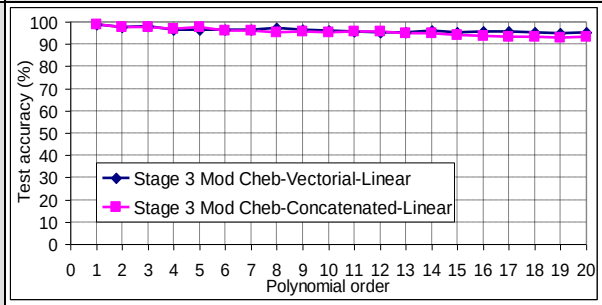
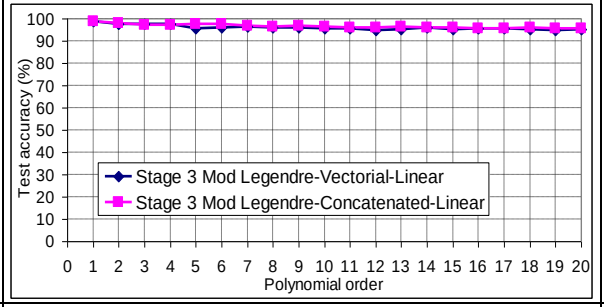
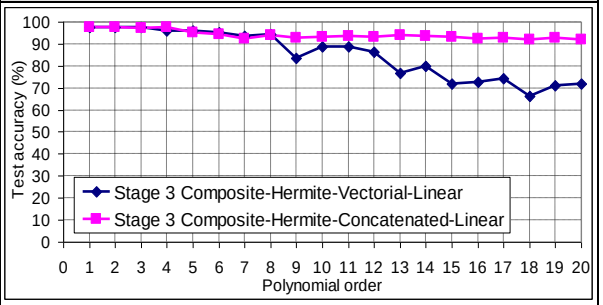
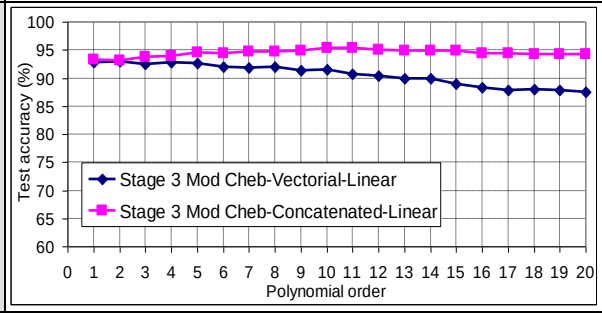
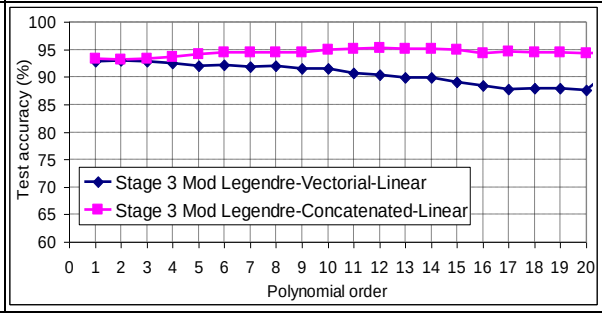
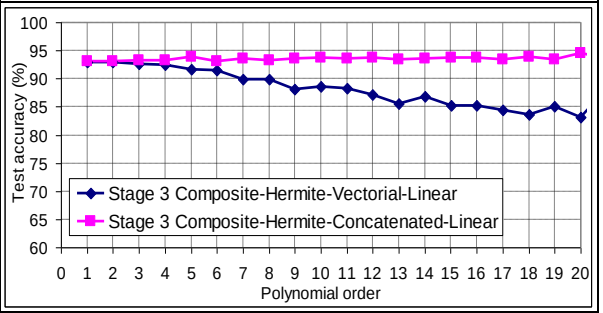
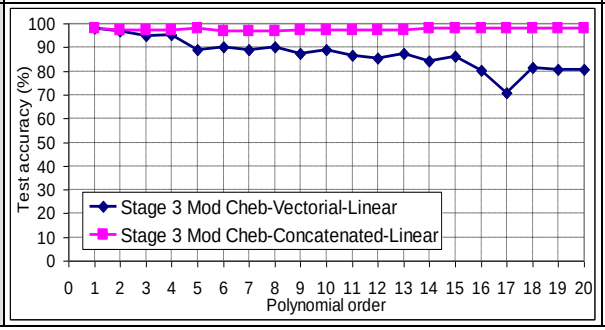
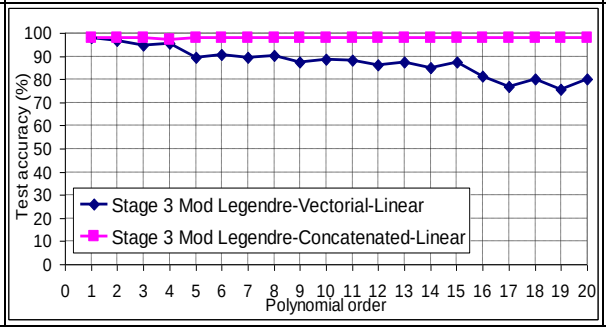
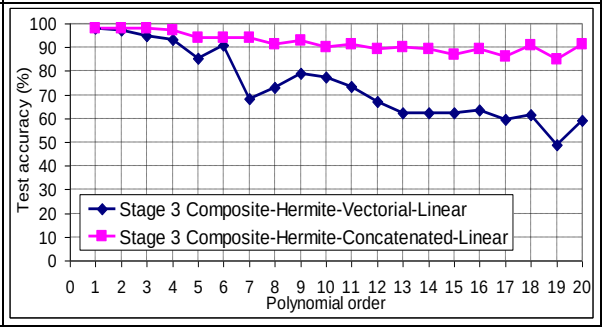
To enable a better quantitative assessment of the improvements introduced by the Stage 3 concatenated kernels over their vectorial counterparts, the average accuracy improvement

factors for the first 20 polynomial orders of each kernel have also been calculated in Table 6.17. At the bottom of the table, these average accuracy improvement factors were calculated to be equal to 4.865%, 4.365% and 6.927% for the Stage 3 Modified Chebyshev, Modified Legendre and Composite Hermite kernels, respectively; making up a total average accuracy improvement factor of 5.385%, which again demonstrates the outperformance of the proposed concatenated approach over its vectorial counterpart, even with the linear kernel implemented.

Furthermore, besides the overall superior performance demonstrated by the Stage 3 concatenated kernels over their vectorial counterparts, and inline with the similar behaviour exhibited by the Stage 1- and Stage 2-concatenated kernels in Sections 6.5.2.1 and 6.5.2.2, it can also be observed that concatenated approach continues to rectify the cases where the vectorial approach encounters sudden abrupt changes in the classification accuracy across the polynomial orders. This behaviour can be clearly observed in the graphical results illustrated in Table 6.16, such as the Iris, Thyroid, and Two Spirals datasets. To quantify this observation, Table 6.17 has also been used to calculate the standard deviation of the average improvement factors (shown in brackets), as well as the minimum and maximum improvement factors (shown in bold), introduced by the Stage 3 concatenated kernels.

As it can be observed, the standard deviation of the average improvement factors can also be quite large relative to the mean, due to the oscillating difference between the two sets of results across the polynomial orders in some datasets. Conversely, however, one can also evaluate the steadiness of the results by examining the standard deviation of the classification accuracy values themselves, reported in the statistical analysis tests tabulated in Appendix B.6, where, not only the mean of the Stage 3 concatenated kernels is larger than the mean of the vectorial kernels, but also their standard deviation is a lot smaller for most of the results obtained.

Table 6.16 Comparative experimental results of Stage 3 concatenated-linear and vectorial-linear kernels.

	Stage 3 Modified Chebyshev kernels	Stage 3 Modified Legendre kernels	Stage 3 Composite Hermite kernels
Breast Cancer			
Image Segmentation			
Ionosphere			

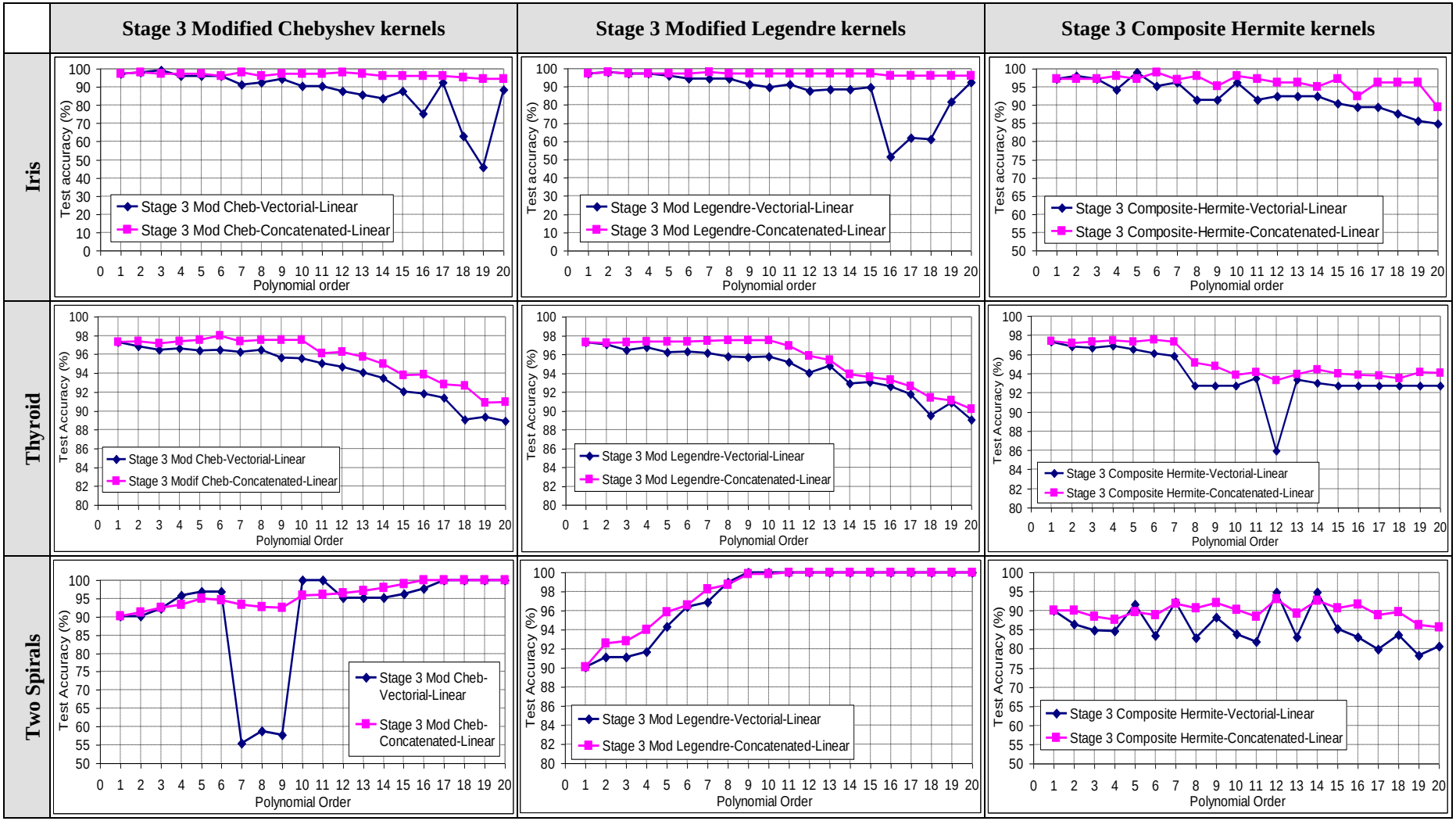

Stage 3 Modified Chebyshev kernels
Stage 3 Modified Legendre kernels
Stage 3 Composite Hermite kernels
Iris
Thyroid
Two Spirals
Test accuracy (%)
Test Accuracy (%)
Polynomial order
Polynomial Order
Stage 3 Mod Cheb-Vectorial-Linear
Stage 3 Mod Cheb-Concatenated-Linear
Stage 3 Modif Cheb-Concatenated-Linear
Stage 3 Mod Cheb-Vectorial-Linear
Stage 3 Mod Cheb-Concatenated-Linear
Stage 3 Mod Legendre-Vectorial-Linear
Stage 3 Mod Legendre-Concatenated-Linear
Stage 3 Composite-Hermite-Vectorial-Linear
Stage 3 Composite-Hermite-Concatenated-Linear
Stage 3 Composite Hermite-Vectorial-Linear
Stage 3 Composite Hermite-Concatenated-Linear

Table 6.17 Quantitative assessment of the average improvements in the scored classification accuracy of Stage 3 concatenated-linear and vectorial-linear kernels.

		Stage 3 Modified Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	98.721	97.441	97.868	96.588	96.375	96.375	96.375	97.228	96.375	95.949	95.736	95.096	95.096	95.949	95.096	95.522	95.522	95.096	94.883	95.309
	N2	98.721	97.655	97.441	97.015	97.441	95.949	95.949	95.096	95.522	95.096	95.522	95.522	94.883	94.670	94.030	93.817	93.390	93.390	92.964	93.177
	I	0.000	0.213	-0.426	0.426	**1.066**	-0.426	-0.426	**-2.132**	-0.853	-0.853	-0.213	0.426	-0.213	-1.279	-1.066	-1.706	-2.132	-1.706	-1.919	-2.132
	AI	**-0.768 (0.967)**																			
Image Segmentation	N1	92.857	92.905	92.524	92.762	92.667	92.000	91.857	92.000	91.429	91.476	90.714	90.429	89.905	89.857	89.000	88.333	87.810	88.048	87.905	87.524
	N2	93.238	93.095	93.714	93.952	94.571	94.429	94.762	94.762	94.952	95.429	95.381	95.048	94.905	94.905	94.952	94.476	94.333	94.238	94.238	94.286
	I	0.381	**0.190**	1.190	1.190	1.905	2.429	2.905	2.762	3.524	3.952	4.667	4.619	5.000	5.048	5.952	6.143	6.524	6.190	6.333	**6.762**
	AI	**3.883 (2.158)**																			

		Stage 3 Modified Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	98.721	97.655	97.655	97.441	95.736	96.162	96.375	95.949	95.949	95.522	95.736	94.883	95.096	95.949	95.096	95.522	95.522	95.096	94.883	95.309
	N2	98.721	97.868	97.228	97.228	97.441	97.441	96.802	96.588	96.802	96.588	96.162	96.162	96.375	95.949	95.949	95.736	95.736	95.949	95.522	95.736
	I	0.000	0.213	**-0.426**	-0.213	**1.706**	1.279	0.426	0.640	0.853	1.066	0.426	1.279	1.279	0.000	0.853	0.213	0.213	0.853	0.640	0.426
	AI	**0.586 (0.561)**																			
Image Segmentation	N1	92.857	93.000	92.857	92.524	92.000	92.238	91.857	91.952	91.476	91.476	90.762	90.429	89.952	89.905	89.048	88.333	87.810	88.000	87.905	87.524
	N2	93.238	93.190	93.381	93.619	94.095	94.429	94.429	94.524	94.476	94.905	95.095	95.190	95.048	95.095	95.000	94.333	94.571	94.429	94.524	94.238
	I	0.381	**0.190**	0.524	1.095	2.095	2.190	2.571	2.571	3.000	3.429	4.333	4.762	5.095	5.190	5.952	6.000	6.762	6.429	6.619	**6.714**
	AI	**3.795 (2.277)**																			

		Stage 3 Composite Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	97.441	97.655	97.655	96.162	96.162	95.309	93.603	94.243	83.582	88.699	88.699	86.354	76.759	79.957	71.855	72.495	74.200	66.311	71.215	71.855
	N2	97.441	97.441	97.015	97.655	95.096	94.456	92.537	94.030	92.751	93.177	93.603	93.177	93.817	93.390	93.177	92.324	92.964	91.898	92.964	92.111
	I	0.000	-0.213	-0.640	1.493	**-1.066**	-0.853	-1.066	-0.213	9.168	4.478	4.904	6.823	17.058	13.433	21.322	19.829	18.763	**25.586**	21.748	20.256
	AI	**9.041 (9.611)**																			
Image Segmentation	N1	92.905	92.857	92.667	92.429	91.619	91.476	89.857	89.857	88.190	88.571	88.238	87.095	85.524	86.857	85.190	85.190	84.381	83.667	85.000	83.143
	N2	93.095	93.048	93.286	93.238	93.905	93.095	93.619	93.286	93.571	93.810	93.524	93.810	93.381	93.619	93.810	93.810	93.476	93.905	93.429	94.524
	I	0.190	**0.190**	0.619	0.810	2.286	1.619	3.762	3.429	5.381	5.238	5.286	6.714	7.857	6.762	8.619	8.619	9.095	10.238	8.429	**11.381**
	AI	**5.326 (3.562)**																			

		Stage 3 Modified Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ionosphere	N1	98.013	96.689	94.702	95.364	88.742	90.066	88.742	90.066	87.417	88.742	86.755	85.430	87.417	84.106	86.093	80.132	70.861	81.457	80.795	80.795
	N2	98.013	97.351	97.351	97.351	98.013	96.689	96.689	96.689	97.351	97.351	97.351	97.351	97.351	98.013	98.013	98.013	98.013	98.013	98.013	98.013
	I	**0**	0.662	2.649	1.987	9.272	6.623	7.947	6.623	9.934	8.609	10.596	11.921	9.934	13.907	11.921	17.881	**27.152**	16.556	17.219	17.219
	AI	**10.430 (6.719)**																			
Iris	N1	97.143	98.095	99.048	96.190	96.190	96.190	91.429	92.381	94.286	90.476	90.476	87.619	85.714	83.810	87.619	75.238	92.381	62.857	45.714	88.571
	N2	97.143	98.095	97.143	97.143	97.143	96.190	98.095	96.190	97.143	97.143	97.143	98.095	97.143	96.190	96.190	96.190	96.190	95.238	94.286	94.286
	I	0	0	**-1.905**	0.952	0.952	0	6.667	3.810	2.857	6.667	6.667	10.476	11.429	12.381	8.571	20.952	3.810	32.381	**48.571**	5.714
	AI	**9.048 (26.551)**																			

		Stage 3 Modified Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ionosphere	N1	98.013	96.689	94.702	95.364	89.404	90.728	89.404	90.066	87.417	88.742	88.079	86.093	87.417	84.768	87.417	81.457	76.821	80.132	75.497	80.132
	N2	98.013	98.013	98.013	97.351	98.013	98.013	98.013	98.013	98.013	98.013	98.013	98.013	98.013	98.013	98.013	98.013	98.013	98.013	98.013	98.013
	I	**0**	1.325	3.311	1.987	8.609	7.285	8.609	7.947	10.596	9.272	9.934	11.921	10.596	13.245	10.596	16.556	21.192	17.881	**22.517**	17.881
	AI	**10.563 (6.317)**																			
Iris	N1	97.143	98.095	97.143	97.143	96.190	94.286	94.286	94.286	91.429	89.524	87.619	88.571	88.571	89.524	51.429	61.905	60.952	81.905	92.381	
	N2	97.143	98.095	97.143	97.143	97.143	97.143	98.095	97.143	97.143	97.143	97.143	97.143	97.143	97.143	97.143	96.190	96.190	96.190	96.190	96.190
	I	0	0	0	**0**	0.952	2.857	3.810	2.857	5.714	7.619	5.714	9.524	8.571	8.571	7.619	**44.762**	34.286	35.238	14.286	3.810
	AI	**9.810 (12.908)**																			

		Stage 3 Composite Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ionosphere	N1	98.013	97.351	94.702	93.377	85.430	90.728	68.212	72.848	78.808	77.483	73.510	66.887	62.252	62.252	62.252	63.576	59.603	61.589	49.007	58.940
	N2	98.013	98.013	98.013	97.351	94.040	94.040	94.040	91.391	92.715	90.066	91.391	89.404	90.066	89.404	86.755	89.404	86.093	90.728	84.768	91.391
	I	**0**	0.662	3.311	3.974	8.609	3.311	25.828	18.543	13.907	12.583	17.881	22.517	27.815	27.152	24.503	25.828	26.490	29.139	**35.762**	32.450
	AI	**18.013 (11.409)**																			
Iris	N1	97.143	98.095	97.143	94.286	99.048	95.238	96.190	91.429	91.429	96.190	91.429	92.381	92.381	92.381	90.476	89.524	89.524	87.619	85.714	84.762
	N2	97.143	97.143	97.143	98.095	97.143	99.048	97.000	98.095	95.238	98.095	97.143	96.190	96.190	91.429	97.143	92.381	96.190	96.190	96.190	89.524
	I	0	-0.952	0	3.810	**-1.905**	3.810	0.810	6.667	3.810	1.905	5.714	3.810	3.810	2.619	6.667	2.857	6.667	8.571	**10.476**	4.762
	AI	**3.695 (3.205)**																			

		Stage 3 Modified Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	97.287	96.849	96.470	96.616	96.441	96.470	96.266	96.499	95.683	95.595	95.069	94.656	94.047	93.487	92.077	91.847	91.364	89.055	89.376	88.884
	N2	97.287	97.404	97.170	97.375	97.520	97.950	97.375	97.520	97.520	97.491	96.073	96.273	95.730	94.973	93.773	93.884	92.773	92.638	90.827	90.924
	I	**0**	0.554	0.700	0.758	1.079	1.479	1.109	1.021	1.838	1.896	1.004	1.617	1.683	1.486	1.696	2.037	1.409	**3.584**	1.452	2.040
	AI	**1.422 (0.737)**																			
Two Spirals	N1	90.092	90.092	92.197	95.881	96.842	96.842	55.400	58.810	57.757	100	100	95.263	95.263	95.263	96.316	97.695	100	100	100	100
	N2	90.092	91.144	92.512	93.294	95.039	94.583	93.364	92.746	92.499	95.737	95.974	96.436	96.974	97.828	98.983	100	100	100	100	100
	I	0	1.053	0.315	-2.587	-1.803	-2.259	**37.963**	33.936	34.741	**-4.263**	-4.026	1.173	1.710	2.564	2.667	2.305	0	0	0	0
	AI	**5.175 (13.255)**																			
AAI		**4.865 (8.725)**																			
TAI		**5.385 (8.223)**																			

		Stage 3 Modified Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	97.287	97.054	96.441	96.733	96.237	96.324	96.120	95.799	95.712	95.741	95.174	94.063	94.826	92.928	93.084	92.583	91.738	89.530	90.837	89.027
	N2	97.2871	97.1704	97.2579	97.3746	97.3746	97.3746	97.4621	97.5204	97.4913	97.5204	96.9374	95.8263	95.3975	93.8654	93.5628	93.2749	92.5766	91.3795	91.084	90.1739
	I	**0**	0.11668	0.81681	0.64177	1.13769	1.05017	1.34189	1.72112	1.77946	1.77946	1.76348	1.76355	0.57113	0.93705	0.47915	0.69158	0.83823	**1.8497**	0.24661	1.1465
	AI	**1.034 (0.6)**																			
Two Spirals	N1	90.092	91.144	91.144	91.670	94.302	96.407	96.842	98.947	100	100	100	100	100	100	100	100	100	100	100	100
	N2	90.092	92.540	92.794	93.973	95.828	96.586	98.274	98.735	99.879	99.838	100	100	100	100	100	100	100	100	100	100
	I	0	1.396	1.650	**2.302**	1.526	0.179	1.432	**-0.213**	-0.121	-0.162	0	0	0	0	0	0	0	0	0	0
	AI	**0.399 (0.77)**																			
AAI		**4.365 (7.239)**																			
TAI		**5.385 (8.223)**																			

		Stage 3 Composite Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	97.345	96.820	96.704	96.908	96.558	96.091	95.858	92.707	92.707	92.707	93.497	85.937	93.373	93.028	92.707	92.707	92.707	92.707	92.707	92.707
	N2	97.4037	97.1995	97.3454	97.4329	97.2871	97.5496	97.3454	95.0939	94.7554	93.8624	94.1239	93.2553	93.9522	94.4326	94.0239	93.8232	93.7626	93.5325	94.1237	94.0633
	I	**0.05834**	0.37923	0.64177	0.52509	0.72929	1.45857	1.48775	2.38675	2.0483	1.15524	0.62656	**7.31792**	0.57927	1.40475	1.31673	1.11609	1.05546	0.82542	1.41653	1.35613
	AI	**1.394 (1.503)**																			
Two Spirals	N1	90.092	86.407	84.920	84.577	91.670	83.432	92.288	82.815	88.169	83.867	81.945	94.828	82.998	94.828	85.195	82.983	79.826	83.654	78.378	80.765
	N2	90.092	90.092	88.465	87.737	89.546	88.773	91.837	90.583	91.952	90.245	88.355	92.969	89.217	92.579	90.734	91.598	88.933	89.654	86.330	85.748
	I	0	3.684	3.546	3.161	-2.124	5.340	-0.451	7.768	3.783	6.378	6.410	-1.860	6.219	**-2.249**	5.540	8.615	**9.106**	6.001	7.952	4.984
	AI	**4.09 (3.63)**																			
AAI		**6.927 (8.47)**																			
TAI		**5.385 (8.223)**																			

N1 Classification accuracy of Stage 3 kernels implemented using the vectorial approach & the Linear dot-product kernel
N2 Classification accuracy of Stage 3 kernels implemented using the concatenated approach & the Linear dot-product kernel
I Improvement = N2-N1
AI Average Improvement for the first 20 orders
AAI Average of Average Improvements of the first 20 orders over all the datasets under investigation
TAI Total Average Improvement over all datasets and all kernels under investigation for the first 20 orders

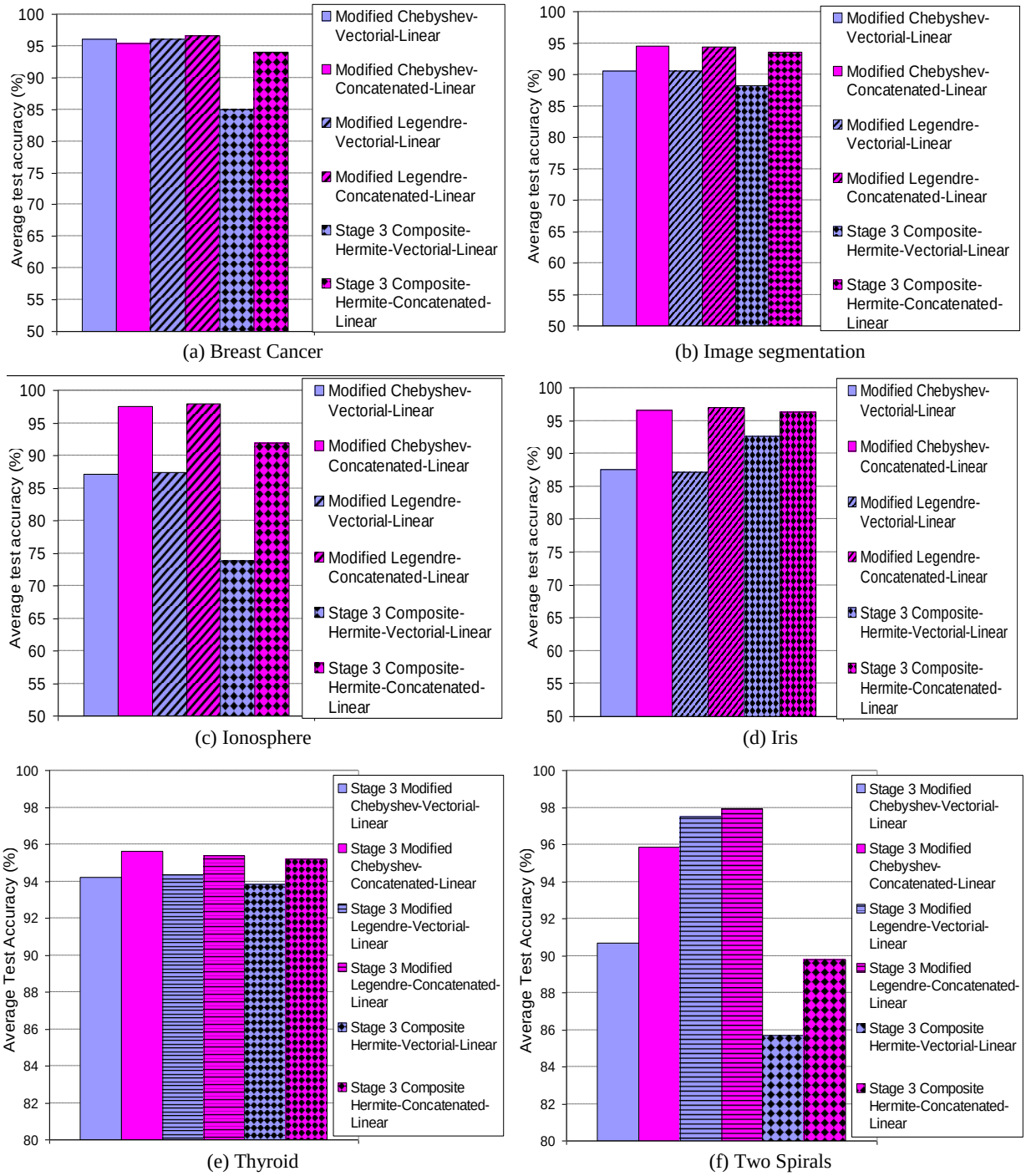


Figure 6.14 Bar chart comparative assessment of the average classification accuracy scored by the first 20 orders of the Stage 3 concatenated-linear and vectorial-linear kernels.

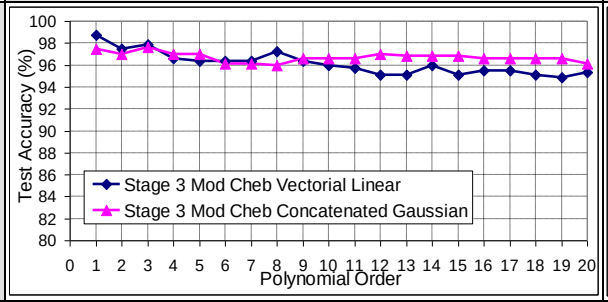
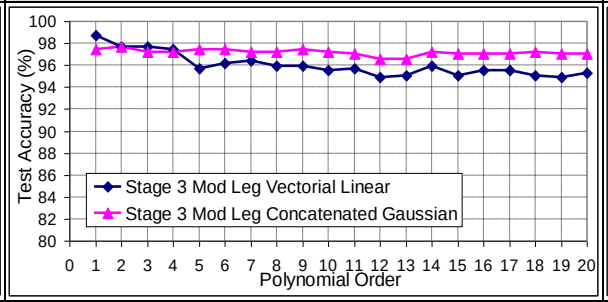
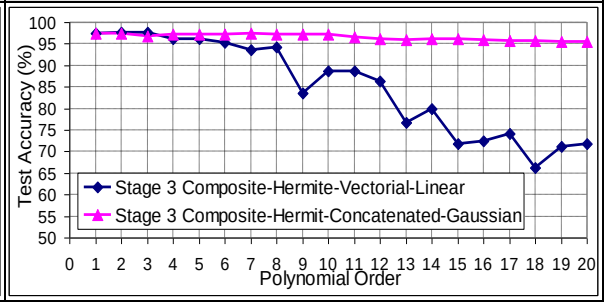
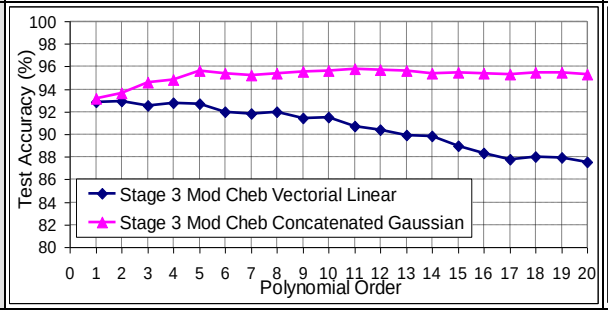
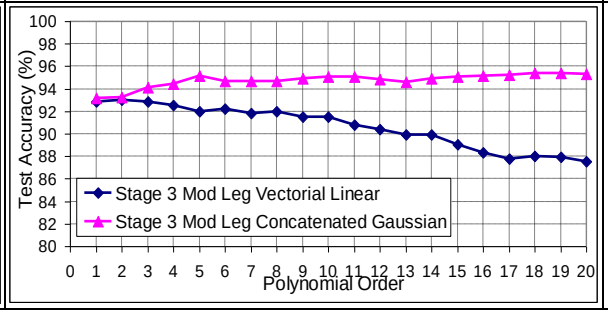
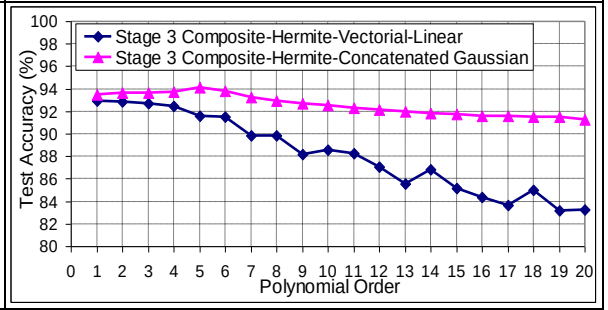
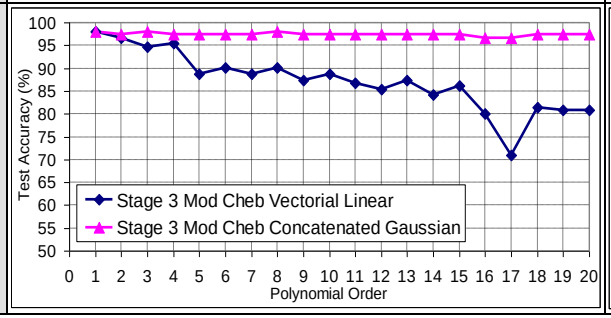
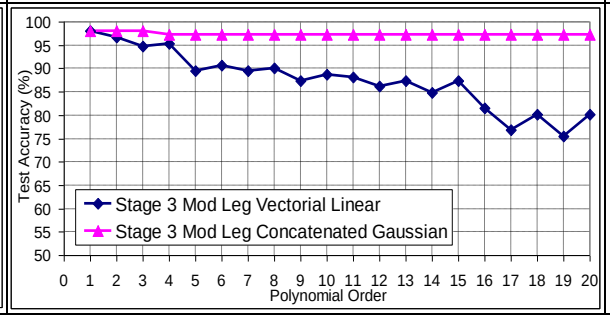
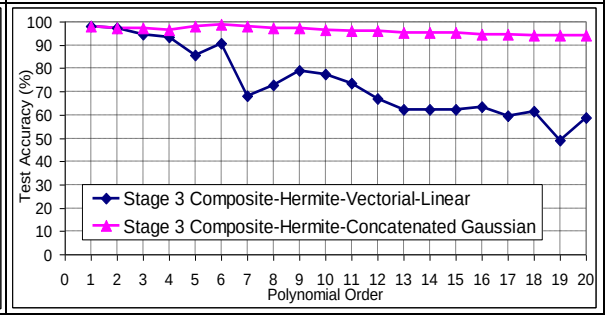
Similar to the experimental investigations conducted in Sections 6.5.2.1 and 6.5.2.2 on the Stage 1 and Stage 2 kernels, respectively, the second set of experiments conducted in this section explores the effect of using the Gaussian kernel, instead of the linear kernel, on the polynomial-processed inputs, as per their composite Stage 3 mathematical formulations shown in Table 6.15. Table 6.18 illustrates the graphical comparative results of the highest classification accuracy scored in each of these two cases for the Modified Chebyshev, Modified Legendre, and Composite Hermite kernels, whereas the average accuracy scored by the first 20 orders is also illustrated in the bar charts in Figure 6.15.

Inline with the results obtained in this section from the Stage 3 concatenated-linear kernels, one can also observe that the Stage 3 concatenated-Gaussian kernels were actually able to achieve even better classification performance than their vectorial-linear counterparts. By comparing the two sets of graphical results in Tables 6.18 and 6.16, for the Stage 3 concatenated-Gaussian and Stage 3 concatenated-linear kernels, respectively, such a constructive effect on the classification performance of the concatenated approach that the Gaussian kernel introduces can probably be observed in the Thyroid, and Two Spirals datasets, and the Hermite kernel performance in the Ionosphere and Iris datasets.

The overall improvement introduced by the evaluation of the Gaussian kernel on the concatenated-processed vectors can, however, be better assessed quantitatively by calculating the average accuracy improvement factors for each kernel in each dataset, as shown in Table 6.19. These were found to be equal to 5.899%, 4.861%, and 8.973% for the Stage 3 Modified Chebyshev, Modified Legendre, and Composite Hermite kernels respectively; making up a total average accuracy improvement factor of 6.577% depending on the kernel and the dataset used. Compared to its corresponding average improvement factor using the Stage 3 concatenated-linear kernels (5.385%), this result again shows that the Gaussian kernel was able to introduce a further accuracy improvement factor of about 1.192%, due to the reasons explained earlier in Section 6.2.

Besides this additional improvement that the Gaussian kernel brings to the Stage 3 concatenated polynomial kernels, it can also be observed from the results in Table 6.18 that the Stage 3 concatenated-Gaussian kernels are also a lot more consistent and steady across the polynomial orders than their vectorial-linear counterparts, and even smoother than the Stage 3 concatenated-linear kernels. This observation can also be assessed quantitatively using Table 6.19 by comparing the standard deviation (shown in brackets), as well as the minimum and maximum improvement factors (shown in bold), to the average improvement factors for each kernel and dataset experimented.

Table 6.18 Comparative experimental results of Stage 3 concatenated-Gaussian and vectorial-linear kernels.

	Stage 3 Modified Chebyshev kernels	Stage 3 Modified Legendre kernels	Stage 3 Composite Hermite kernels
Breast Cancer			
Image Segmentation			
Ionosphere			

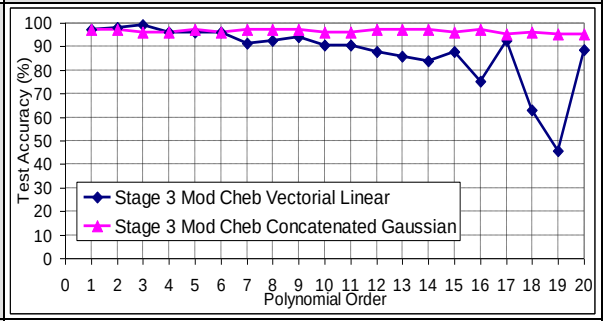
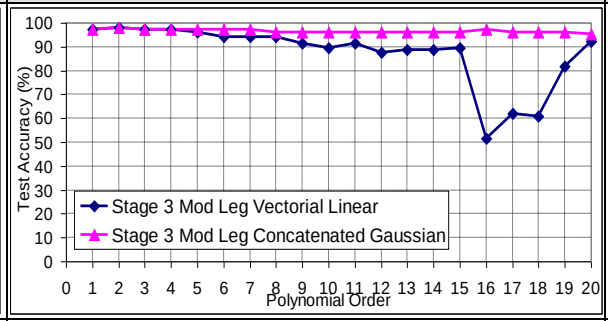
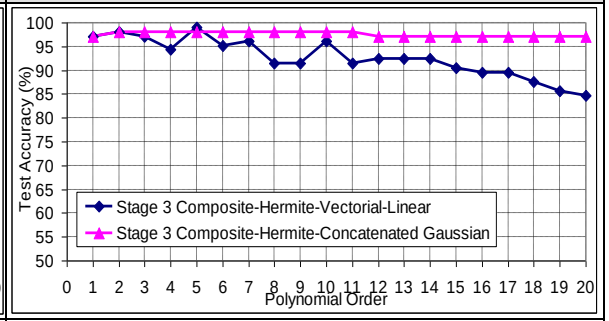
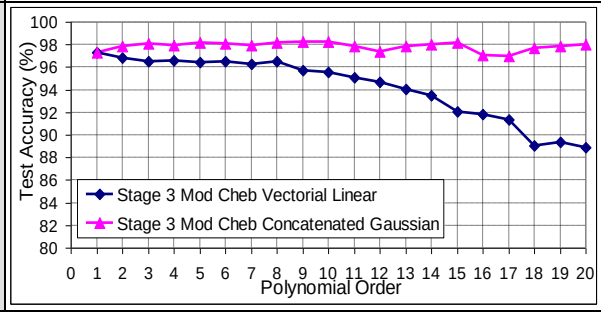
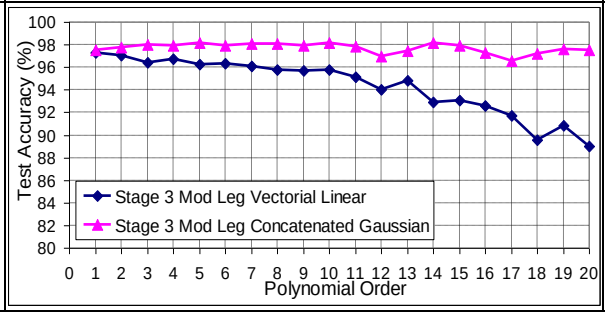
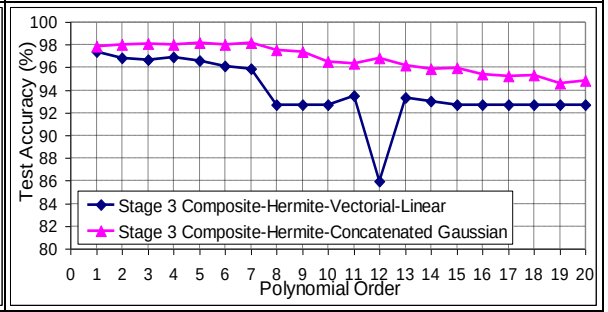
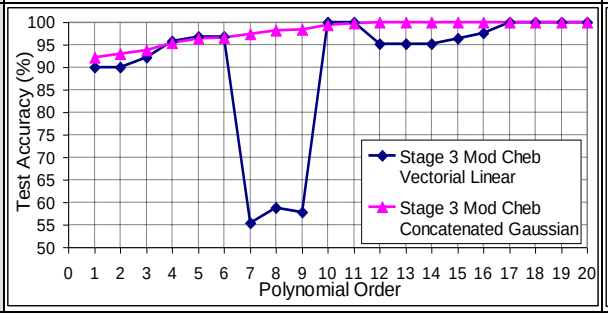
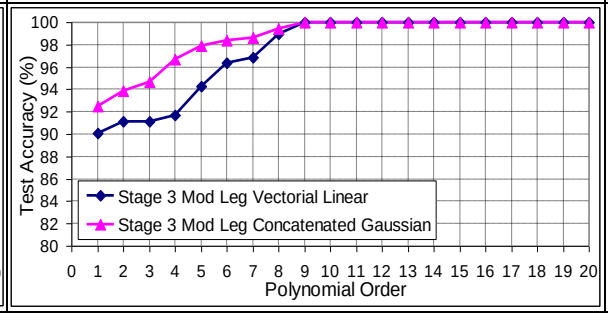
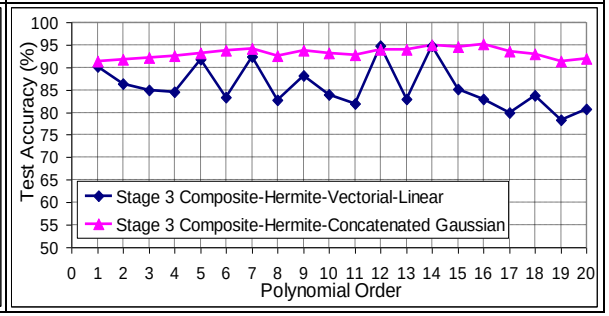
	Stage 3 Modified Chebyshev kernels	Stage 3 Modified Legendre kernels	Stage 3 Composite Hermite kernels
Iris			
Thyroid			
Two Spirals			

Table 6.19 Quantitative assessment of the average improvements in the scored classification accuracy of Stage 3 concatenated-Gaussian and vectorial-linear kernels.

		Stage 3 Modified Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	98.721	97.441	97.868	96.588	96.375	96.375	96.375	97.228	96.375	95.949	95.736	95.096	95.096	95.949	95.096	95.522	95.522	95.096	94.883	95.309
	N2	97.441	97.015	97.655	97.015	97.015	96.162	96.162	95.949	96.588	96.588	96.588	97.015	96.802	96.802	96.802	96.588	96.588	96.588	96.588	96.162
	I	-1.279	-0.426	-0.213	0.426	0.640	-0.213	-0.213	**-1.279**	0.213	0.640	0.853	**1.919**	1.706	0.853	1.706	1.066	1.066	1.493	1.706	0.853
	AI	0.576 (0.949)																			
Image Segmentation	N1	92.857	92.905	92.524	92.762	92.667	92.000	91.857	92.000	91.429	91.476	90.714	90.429	89.905	89.857	89.000	88.333	87.810	88.048	87.905	87.524
	N2	93.143	93.667	94.619	94.810	95.667	95.381	95.238	95.429	95.571	95.667	95.810	95.714	95.619	95.381	95.476	95.429	95.333	95.476	95.476	95.333
	I	**0.286**	0.762	2.095	2.048	3.000	3.381	3.381	3.429	4.143	4.190	5.095	5.286	5.714	5.524	6.476	7.095	7.524	7.429	7.571	**7.810**
	AI	4.612 (2.313)																			

		Stage 3 Modified Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	98.721	97.655	97.655	97.441	95.736	96.162	96.375	95.949	95.949	95.522	95.736	94.883	95.096	95.949	95.096	95.522	95.522	95.096	94.883	95.309
	N2	97.441	97.655	97.228	97.228	97.441	97.441	97.228	97.228	97.441	97.228	97.015	96.588	96.588	97.228	97.015	97.015	97.015	97.228	97.015	97.015
	I	**-1.279**	0.000	-0.426	-0.213	1.706	1.279	0.853	1.279	1.493	1.706	1.279	1.706	1.493	1.279	1.919	1.493	1.493	**2.132**	2.132	1.706
	AI	1.151 (0.916)																			
Image Segmentation	N1	92.857	93.000	92.857	92.524	92.000	92.238	91.857	91.952	91.476	91.476	90.762	90.429	89.952	89.905	89.048	88.333	87.810	88.000	87.905	87.524
	N2	93.143	93.238	94.095	94.429	95.143	94.667	94.714	94.714	94.952	95.095	95.048	94.810	94.619	94.952	95.095	95.190	95.238	95.429	95.381	95.286
	I	0.286	**0.238**	1.238	1.905	3.143	2.429	2.857	2.762	3.476	3.619	4.286	4.381	4.667	5.048	6.048	6.857	7.429	7.429	7.476	**7.762**
	AI	4.167 (2.403)																			

		Stage 3 Composite Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Breast Cancer	N1	97.441	97.655	97.655	96.162	96.162	95.309	93.603	94.243	83.582	88.699	88.699	86.354	76.759	79.957	71.855	72.495	74.200	66.311	71.215	71.855
	N2	97.441	97.441	96.802	97.228	97.228	97.228	97.441	97.228	97.228	97.228	96.588	96.162	95.949	96.162	96.162	95.949	95.736	95.736	95.522	95.522
	I	0.000	-0.213	**-0.853**	1.066	1.066	1.919	3.838	2.985	13.646	8.529	7.889	9.808	19.190	16.205	24.307	23.454	21.535	**29.424**	24.307	23.667
	AI	11.588 (10.296)																			
Image Segmentation	N1	92.905	92.857	92.667	92.429	91.619	91.476	89.857	89.857	88.190	88.571	88.238	87.095	85.524	86.857	85.190	84.381	83.667	85.000	83.143	83.286
	N2	93.524	93.667	93.619	93.762	94.095	93.810	93.286	92.905	92.667	92.571	92.286	92.143	91.952	91.810	91.762	91.619	91.571	91.524	91.524	91.238
	I	**0.619**	0.810	0.952	1.333	2.476	2.333	3.429	3.048	4.476	4.000	4.048	5.048	6.429	4.952	6.571	7.238	7.905	6.524	**8.381**	7.952
	AI	4.426 (2.538)																			

		Stage 3 Modified Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ionosphere	N1	98.013	96.689	94.702	95.364	88.742	90.066	88.742	90.066	87.417	88.742	86.755	85.430	87.417	84.106	86.093	80.132	70.861	81.457	80.795	80.795
	N2	98.013	97.351	98.013	97.351	97.351	97.351	97.351	98.013	97.351	97.351	97.351	97.351	97.351	97.351	97.351	96.689	96.689	97.351	97.351	97.351
	I	**0**	0.662	3.311	1.987	8.609	7.285	8.609	7.947	9.934	8.609	10.596	11.921	9.934	13.245	11.258	16.556	**25.828**	15.894	16.556	16.556
	AI	**10.265 (6.251)**																			
Iris	N1	97.143	98.095	99.048	96.190	96.190	96.190	91.429	92.381	94.286	90.476	90.476	87.619	85.714	83.810	87.619	75.238	92.381	62.857	45.714	88.571
	N2	97.143	97.143	96.190	96.190	97.143	96.190	97.143	97.143	97.143	96.190	96.190	97.143	97.143	97.143	96.190	97.143	95.238	96.190	95.238	95.238
	I	0	-0.952	**-2.857**	0	0.952	0	5.714	4.762	2.857	5.714	5.714	9.524	11.429	13.333	8.571	21.905	2.857	33.333	**49.524**	6.667
	AI	**8.952 (27.475)**																			

		Stage 3 Modified Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ionosphere	N1	98.013	96.689	94.702	95.364	89.404	90.728	89.404	90.066	87.417	88.742	88.079	86.093	87.417	84.768	87.417	81.457	76.821	80.132	75.497	80.132
	N2	98.013	98.013	98.013	97.351	97.351	97.351	97.351	97.351	97.351	97.351	97.351	97.351	97.351	97.351	97.351	97.351	97.351	97.351	97.351	97.351
	I	**0**	1.325	3.311	1.987	7.947	6.623	7.947	7.285	9.934	8.609	9.272	11.258	9.934	12.583	9.934	15.894	20.530	17.219	**21.854**	17.219
	AI	**10.033 (6.124)**																			
Iris	N1	97.143	98.095	97.143	97.143	96.190	94.286	94.286	94.286	91.429	89.524	91.429	87.619	88.571	88.571	89.524	51.429	61.905	60.952	81.905	92.381
	N2	97.143	98.095	97.143	97.143	97.143	97.143	97.143	96.190	96.190	96.190	96.190	96.190	96.190	96.190	96.190	97.143	96.190	96.190	96.190	95.238
	I	0	0	**0**	0	0.952	2.857	2.857	2.857	1.905	4.762	6.667	4.762	8.571	7.619	7.619	6.667	**45.714**	34.286	35.238	14.286	2.857
	AI	**9.381 (13.187)**																			

		Stage 3 Composite Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Ionosphere	N1	98.013	97.351	94.702	93.377	85.430	90.728	68.212	72.848	78.808	77.483	73.510	66.887	62.252	62.252	62.252	63.576	59.603	61.589	49.007	58.940
	N2	98.013	97.351	97.351	96.689	98.013	98.676	98.013	97.351	97.351	96.689	96.026	96.026	95.364	95.364	95.364	94.702	94.702	94.040	94.040	94.040
	I	**0**	0	2.649	3.311	12.583	7.947	29.801	24.503	18.543	19.205	22.517	29.139	33.113	33.113	33.113	31.126	35.099	32.450	**45.033**	35.099
	AI	**22.417 (13.668)**																			
Iris	N1	97.143	98.095	97.143	94.286	99.048	95.238	96.190	91.429	91.429	96.190	91.429	92.381	92.381	92.381	90.476	89.524	89.524	87.619	85.714	84.762
	N2	97.143	98.095	98.095	98.095	98.095	98.095	98.095	98.095	98.095	98.095	98.095	97.143	97.143	97.143	97.143	97.143	97.143	97.143	97.143	97.143
	I	0	0	0.952	3.810	**-0.952**	2.857	1.905	6.667	6.667	1.905	6.667	4.762	4.762	4.762	6.667	7.619	7.619	9.524	11.429	**12.381**
	AI	**5.000 (3.757)**																			

		Stage 3 Modified Chebyshev kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	97.287	96.849	96.470	96.616	96.441	96.470	96.266	96.499	95.683	95.595	95.069	94.656	94.047	93.487	92.077	91.847	91.364	89.055	89.376	88.884
	N2	97.287	97.837	98.084	97.974	98.184	98.098	97.974	98.174	98.276	98.287	97.835	97.397	97.837	98.039	98.174	97.029	96.974	97.730	97.836	98.035
	I	**0**	0.988	1.614	1.358	1.743	1.628	1.708	1.674	2.594	2.692	2.766	2.741	3.790	4.552	6.097	5.182	5.609	8.675	8.460	**9.151**
	AI	**3.651 (2.719)**																			
Two Spirals	N1	90.092	90.092	92.197	95.881	96.842	96.842	55.400	58.810	57.757	100	100	95.263	95.263	95.263	96.316	97.695	100	100	100	100
	N2	92.283	92.937	93.727	95.313	96.429	96.539	97.336	98.228	98.493	99.380	99.838	100	100	100	100	100	100	100	100	100
	I	2.192	2.846	1.530	-0.568	-0.413	-0.303	**41.936**	39.418	40.736	**-0.620**	-0.162	4.737	4.737	4.737	3.684	2.305	0	0	0	0
	AI	**7.34 (14.504)**																			
AAI		**5.899 (8.894)**																			
TAI		**6.577 (8.841)**																			

		Stage 3 Modified Legendre kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	97.287	97.054	96.441	96.733	96.237	96.324	96.120	95.799	95.712	95.741	95.174	94.063	94.826	92.928	93.084	92.583	91.738	89.530	90.837	89.027
	N2	97.563	97.775	98.044	97.937	98.190	97.937	98.093	98.124	97.939	98.194	97.879	96.974	97.427	98.129	97.937	97.293	96.571	97.229	97.579	97.497
	I	**0.276**	0.721	1.603	1.205	1.953	1.613	1.973	2.324	2.227	2.453	2.705	2.911	2.601	5.201	4.854	4.710	4.833	7.700	6.742	**8.470**
	AI	**3.354 (2.312)**																			
Two Spirals	N1	90.092	91.144	91.144	91.670	94.302	96.407	96.842	98.947	100	100	100	100	100	100	100	100	100	100	100	100
	N2	92.533	93.837	94.639	96.728	97.892	98.398	98.633	99.428	100	100	100	100	100	100	100	100	100	100	100	100
	I	2.441	2.693	3.495	**5.058**	3.590	1.991	1.791	0.481	0	**0**	0	0	0	0	0	0	0	0	0	0
	AI	**1.077 (1.591)**																			
AAI		**4.861 (7.013)**																			
TAI		**6.577 (8.841)**																			

		Stage 3 Composite Hermite kernels																			
		Order no.																			
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Thyroid	N1	97.345	96.820	96.704	96.908	96.558	96.091	95.858	92.707	92.707	92.707	93.497	85.937	93.373	93.028	92.707	92.707	92.707	92.707	92.707	92.707
	N2	97.837	98.026	98.120	98.028	98.174	98.037	98.187	97.538	97.383	96.528	96.375	96.828	96.198	95.839	95.937	95.430	95.237	95.284	94.583	94.838
	I	**0.491**	1.205	1.417	1.121	1.616	1.946	2.330	4.831	4.676	3.821	2.878	**10.891**	2.825	2.811	3.230	2.723	2.530	2.577	1.876	2.131
	AI	**2.896 (2.183)**																			
Two Spirals	N1	90.092	86.407	84.920	84.577	91.670	83.432	92.288	82.815	88.169	83.867	81.945	94.828	82.998	94.828	85.195	82.983	79.826	83.654	78.378	80.765
	N2	91.265	91.726	92.147	92.559	93.129	93.812	94.255	92.659	93.723	93.106	92.714	93.945	94.064	95.033	94.522	95.256	93.565	92.946	91.355	92.014
	I	1.174	5.319	7.227	7.983	1.458	10.380	1.966	9.844	5.554	9.239	10.769	**-0.883**	11.066	0.204	9.328	12.273	**13.738**	9.292	12.977	11.249
	AI	**7.508 (4.542)**																			
AAI		**8.973 (9.915)**																			
TAI		**6.577 (8.841)**																			

N1 Classification accuracy of Stage 3 kernels implemented using the vectorial approach & the Linear dot-product kernel
N2 Classification accuracy of Stage 3 kernels implemented using the concatenated approach & the Gaussian kernel
I Improvement = N2-N1
AI Average Improvement for the first 20 orders
AAI Average of Average Improvements of the first 20 orders over all the datasets under investigation
TAI Total Average Improvement over all datasets and all kernels under investigation for the first 20 orders

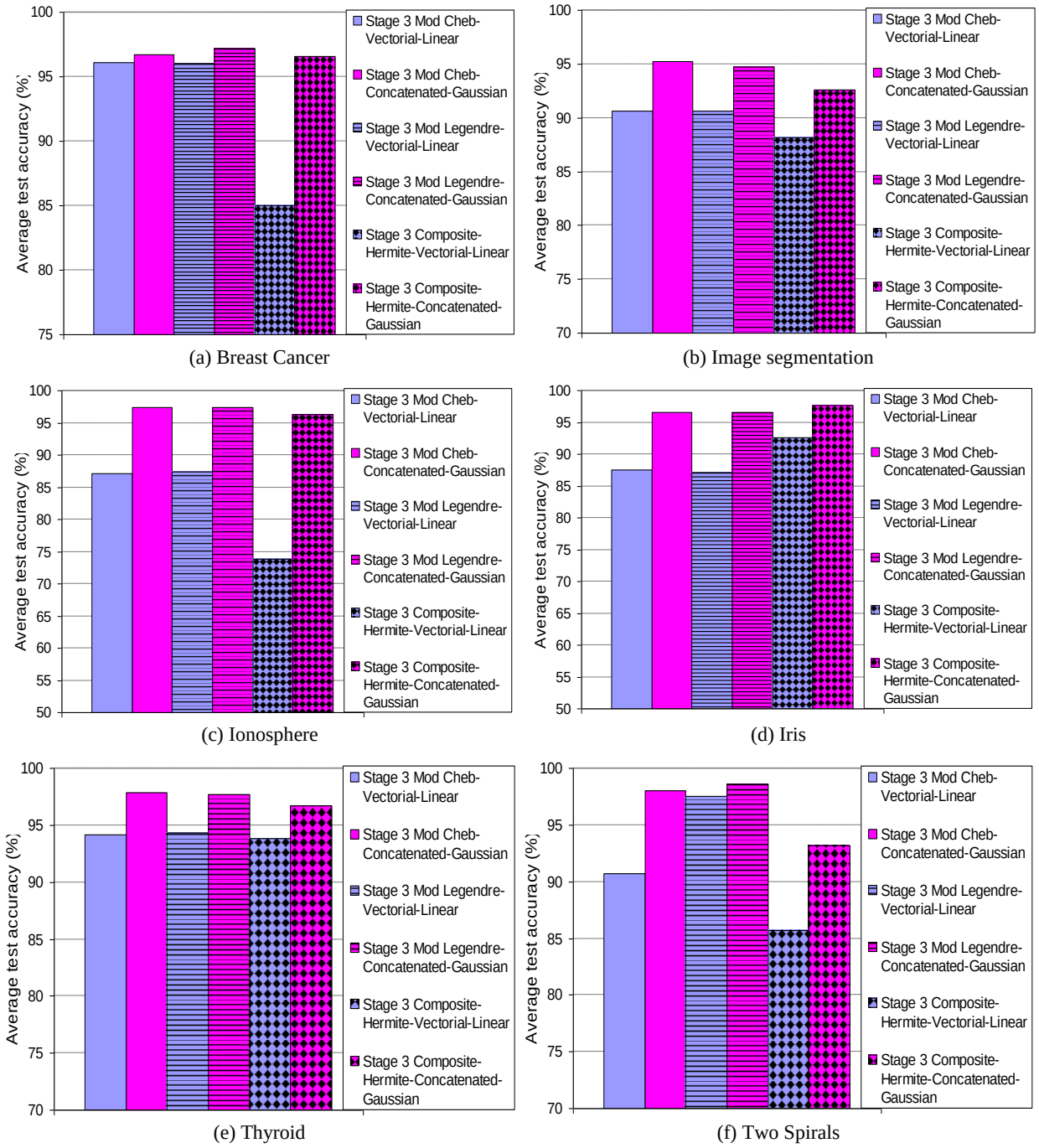


Figure 6.15 Bar chart comparative assessment of the average classification accuracy scored by the first 20 orders of the Stage 3 concatenated-Gaussian and vectorial-linear kernels.

6.6 Re-evaluation of the similarity fusion framework using the proposed concatenated processing approach

Although the experimental results illustrated in the previous section demonstrate the superiority of the proposed concatenated processing approach over its vectorial counterpart, the hierarchical structure of the kernels constructed from orthogonal polynomials, as defined by the developed similarity fusion framework in Chapter 4, is still the same. In other words, regardless of which kernel (e.g., Linear, Gaussian, etc.) is being evaluated on the polynomial-processed vectors, one should still expect the fusion by summation process taking place in Stage 2 concatenated kernels to continue to be synergistic and hence produce better classification performance than their Stage 1 counterparts. Similarly, the fusion by multiplication process taking place in Stage 3 kernels would also be expected to be synergistic and produce better classification performance than their Stage 2 counterparts. The following two sub-sections demonstrate the experimental results which explore whether these two hypotheses continue to hold true, when the input vectors are processed using the newly proposed concatenated approach. Both the linear and Gaussian kernels are used, as an example.

6.6.1 Evaluation of the summative fusion process via the comparative experimental results of the concatenated-Stage 1 and Stage 2 kernels

Tables 6.20 and 6.21 demonstrate the comparative experimental results of the scored classification accuracy of the concatenated-Stage 1 and Stage 2 kernels when both the linear and Gaussian kernels are utilized, respectively. Inline with the experimental results presented back in Section 4.4.4 (Chapter 4), it can be observed from the results obtained herein, that the concatenated-Stage 2 kernels have been able to score a lot higher classification accuracy than their Stage 1 counterparts for most, if not all, of the polynomial orders and datasets investigated. This clearly validates that the fusion by summation process inherent in Stage 2 kernels is indeed synergistic, a fact which has been attributed in Chapter 4 to their complementary fusion properties leading to more accurate acquisition of similarity measures than their Stage 1 counterparts, as exhibited by their shape characteristics.

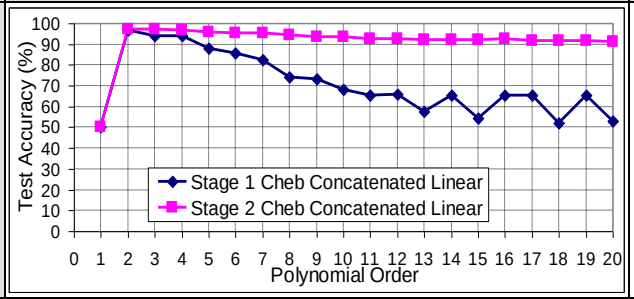
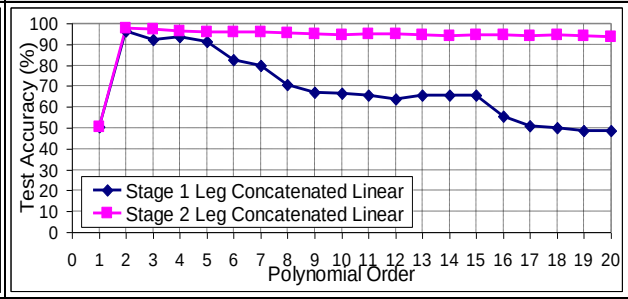
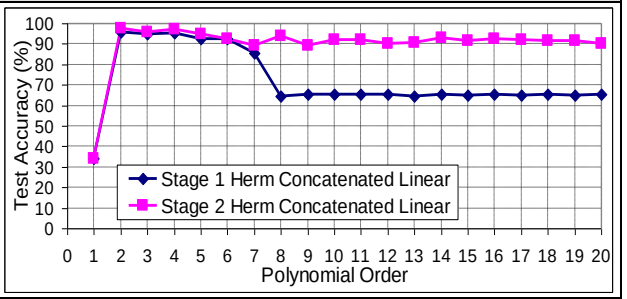
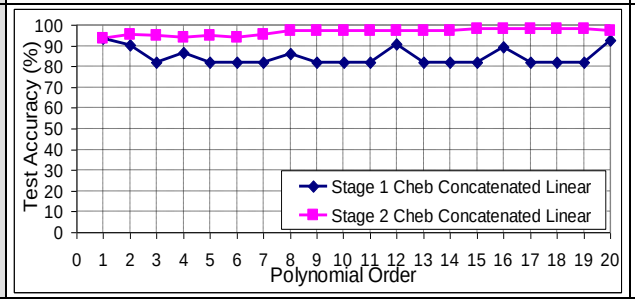
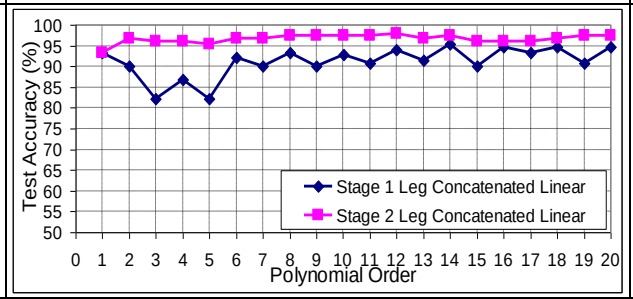
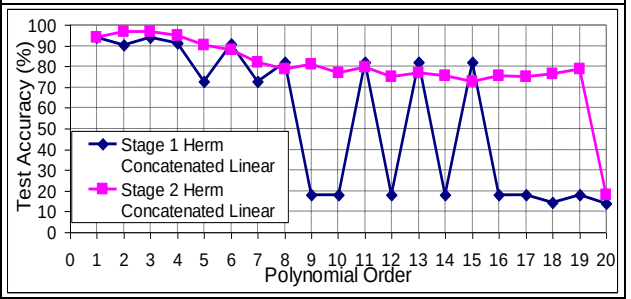
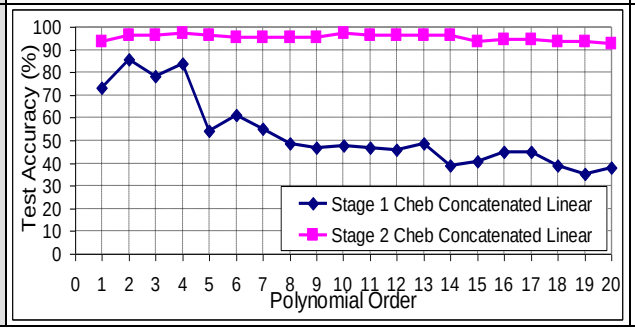
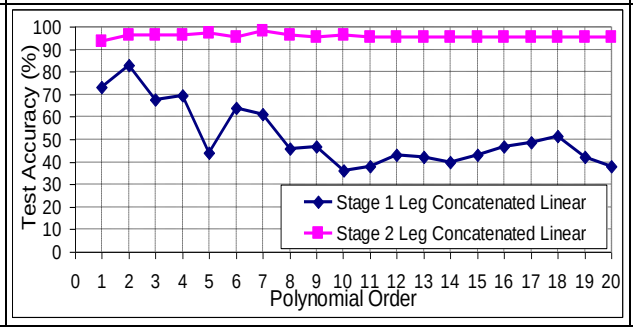
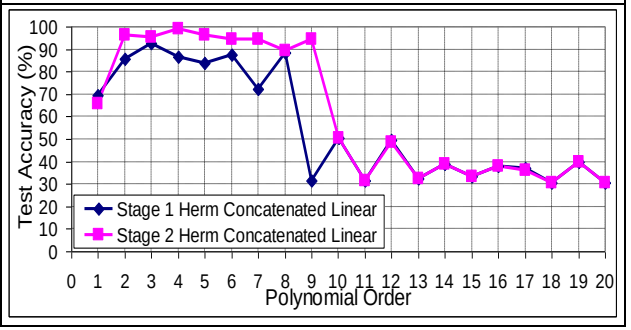
On analyzing closely the results obtained amongst the kernels and datasets investigated, however, one can also draw the following observations. First, the amount of improvement in the classification accuracy introduced by the fusion by summation process tends to

increase as the polynomial order increases. This can be clearly realized from the performance of the Stage 2 kernels compared to their Stage 1 counterparts in some datasets, such as the Breast Cancer, Iris, and Image Segmentation, in both of the results illustrated in Tables 6.20 and 6.21. Recall that as the polynomial order increases, the Stage 2 kernels tend to measure more accurate similarity measures, as their shape characteristics get closer to the ideal model of similarity function, therefore, this can probably explain the reason behind their superior performance over their Stage 1 counterparts in higher order polynomials as observed in the results herein. A similar behaviour has also been observed in the results obtained from the Stage 1- and Stage 2- vectorial-linear kernels in Section 4.4.4.

Second, both of the Stage 1- and Stage 2- concatenated kernels have actually demonstrated different performances depending on the characteristics of the datasets investigated. For example, at the time when their asymptotic performance increases as the order increases, as shown for example in the results obtained from the Two Spirals dataset in Tables 6.20 and 6.21, yet, in other datasets, such as the Thyroid and Image Segmentation, it is mostly the lower orders which score the highest accuracies. Moreover, the amount of improvement introduced by the fusion by summation process tends to be a lot higher in some datasets than others. For example, as shown in the results illustrated in Tables 6.20 and 6.21, both the Stage 2 Chebyshev and Legendre kernels have demonstrated a huge improvement over their Stage 1 counterparts in some datasets, such as the Iris and Image Segmentation, as opposed to a relatively smaller improvement in other datasets, such as the Ionosphere and Thyroid datasets.

A remarkable observation, however, is the fact that both the Stage 1 and Stage 2 concatenated kernels demonstrated a lot smoother and consistent performance across the polynomial orders than their vectorial counterparts. Consequently, this resulted in a smooth transitional change in the amount of improvement introduced by the fusion by summation operation, as demonstrated by their classification performance in most of the datasets, especially when the Gaussian kernel is evaluated on the concatenated processed input vectors (as shown in the results illustrated in Table 6.21) as compared to when the linear kernel is used instead (as shown in the results illustrated in Table 6.20).

Table 6.20 Comparison between the concatenated-Stage 1 and Stage 2 linear kernels to validate the fusion by summation hypothesis.

	Chebyshev	Legendre	Hermite
Breast Cancer			
Ionosphere			
Iris			

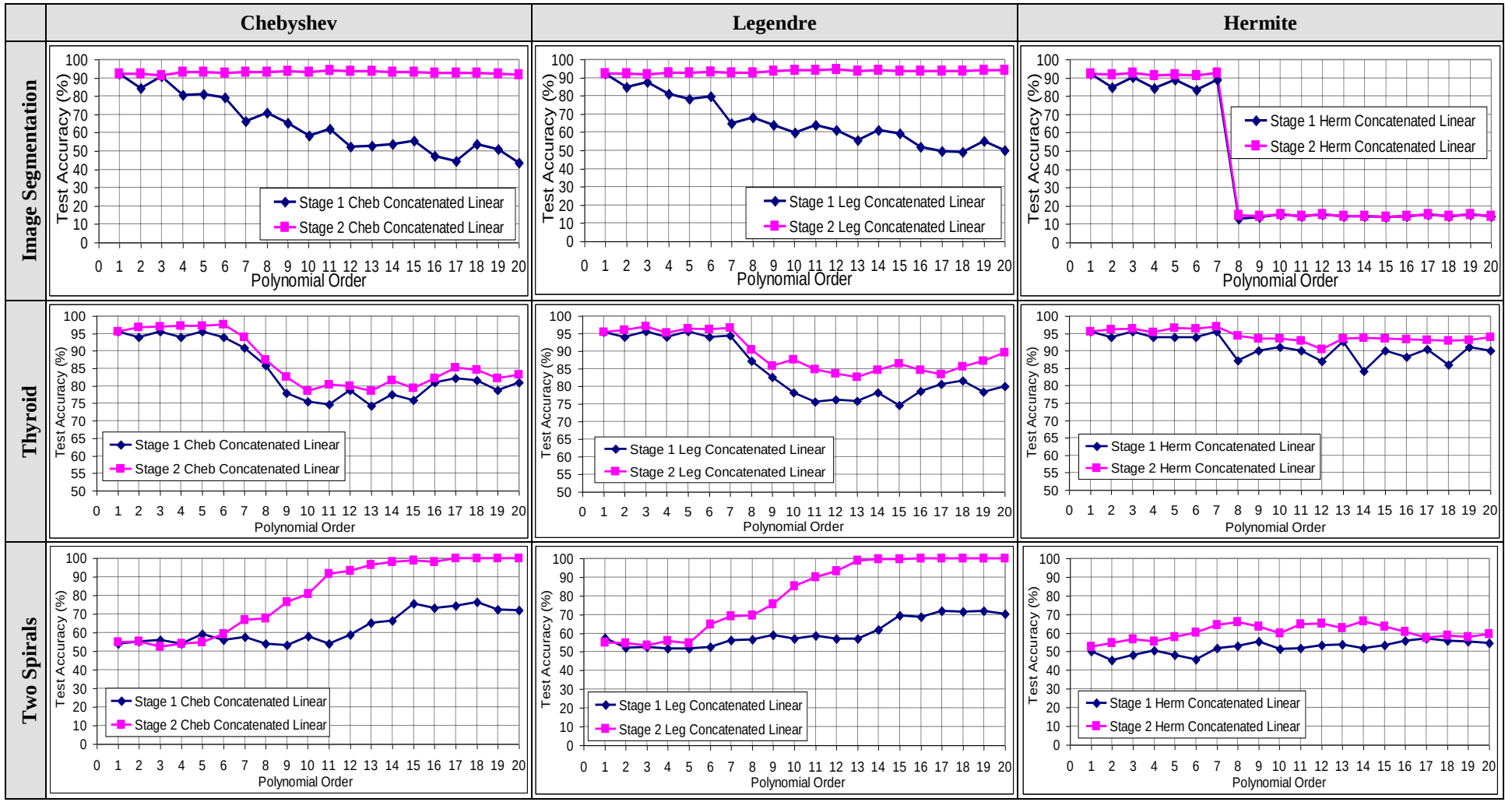
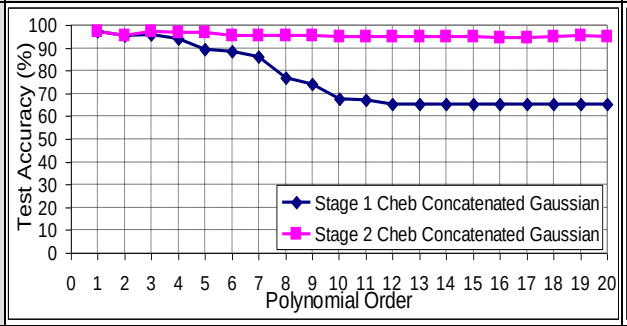
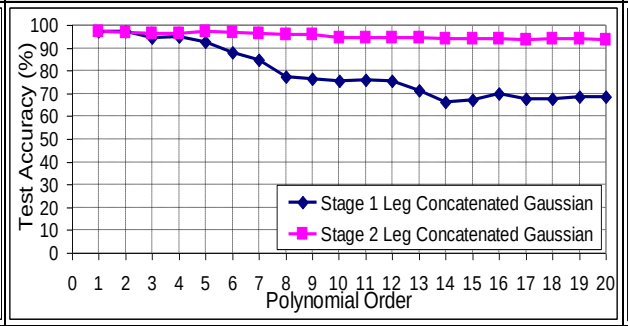
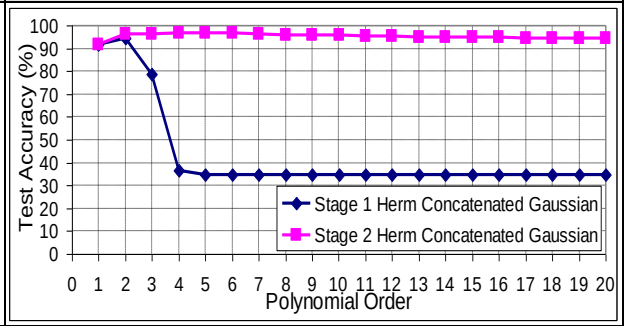
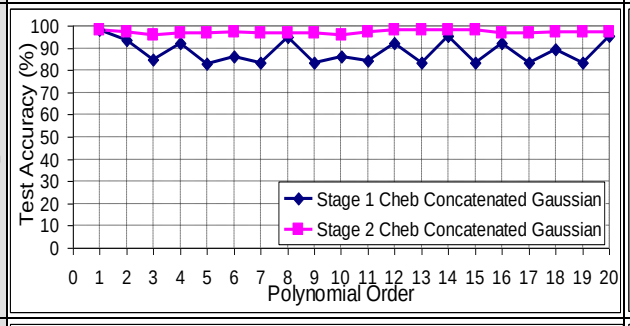
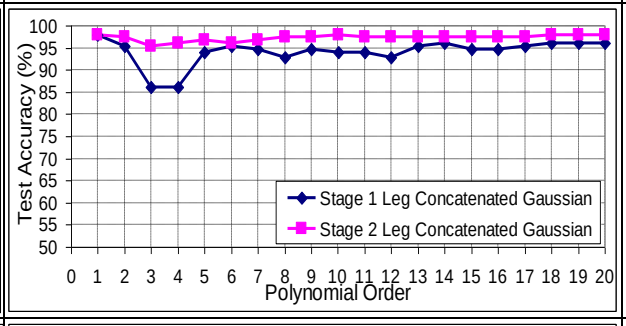
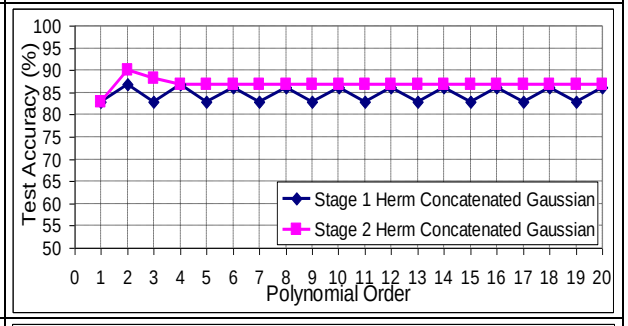
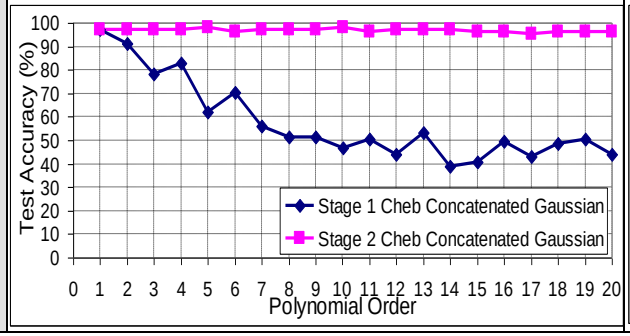
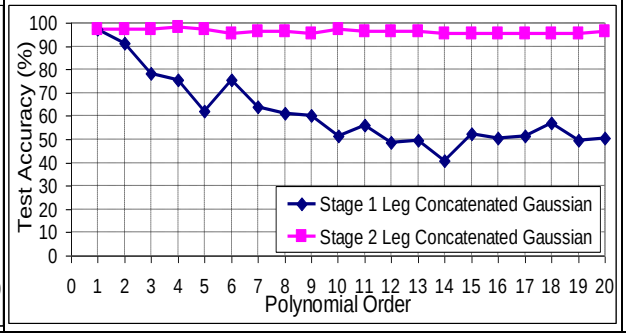
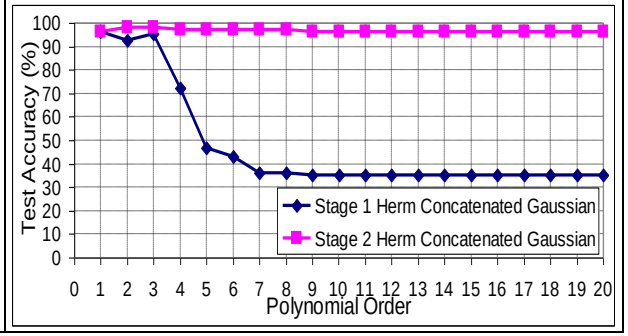
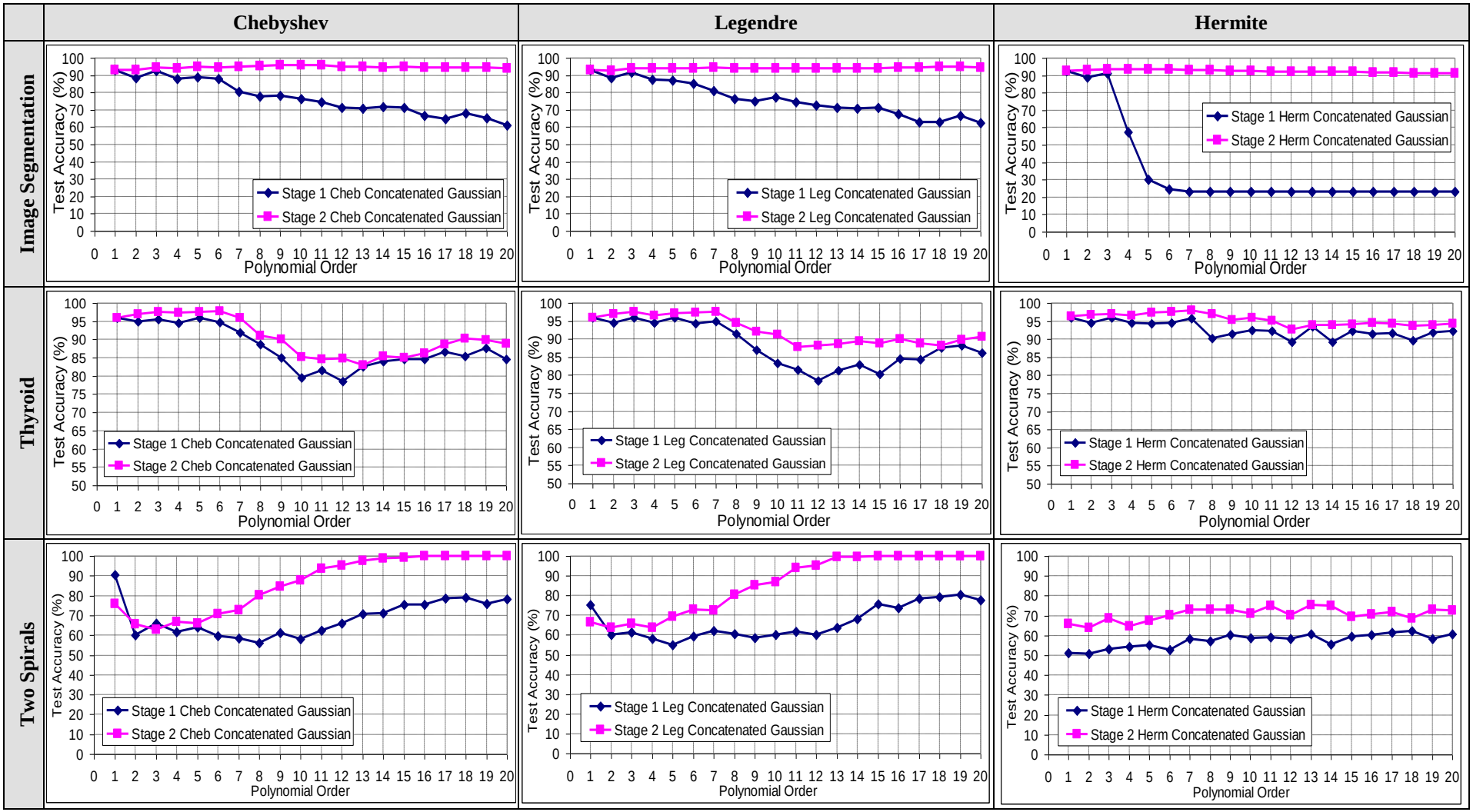

Chebyshev
Legendre
Hermite
Image Segmentation
Thyroid
Two Spirals
Test Accuracy (%)
Polynomial Order
Stage 1 Cheb Concatenated Linear
Stage 2 Cheb Concatenated Linear
Stage 1 Leg Concatenated Linear
Stage 2 Leg Concatenated Linear
Stage 1 Herm Concatenated Linear
Stage 2 Herm Concatenated Linear

Table 6.21 Comparison between the concatenated-Stage 1 and Stage 2 Gaussian kernels to validate the fusion by summation hypothesis.

	Chebyshev	Legendre	Hermite
Breast Cancer			
Ionosphere			
Iris			


Chebyshev
Legendre
Hermite
Image Segmentation
Thyroid
Two Spirals
Test Accuracy (%)
Polynomial Order
Stage 1 Cheb Concatenated Gaussian
Stage 2 Cheb Concatenated Gaussian
Stage 1 Leg Concatenated Gaussian
Stage 2 Leg Concatenated Gaussian
Stage 1 Herm Concatenated Gaussian
Stage 2 Herm Concatenated Gaussian

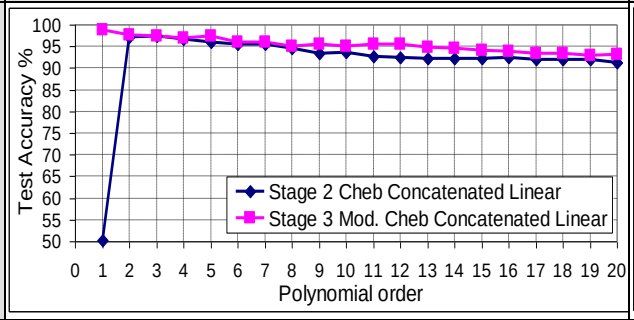
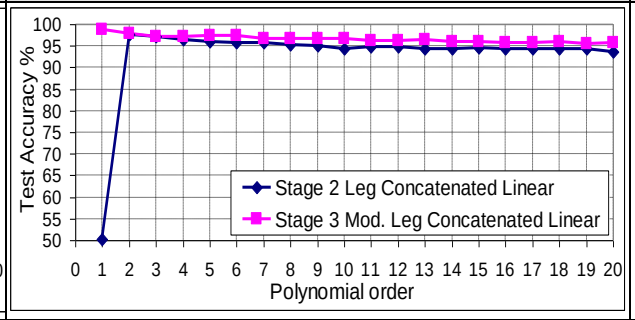
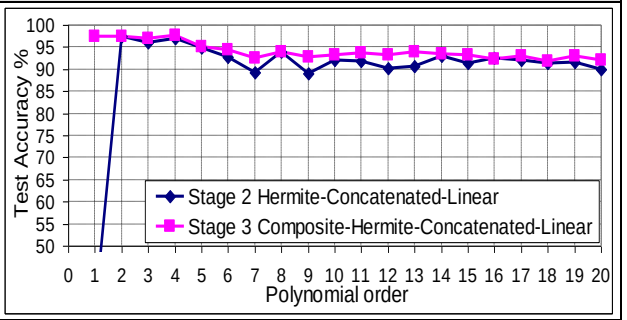
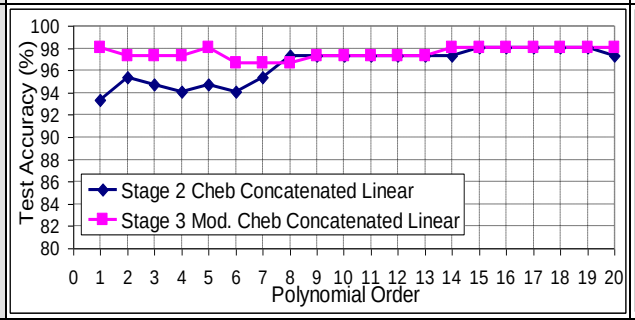
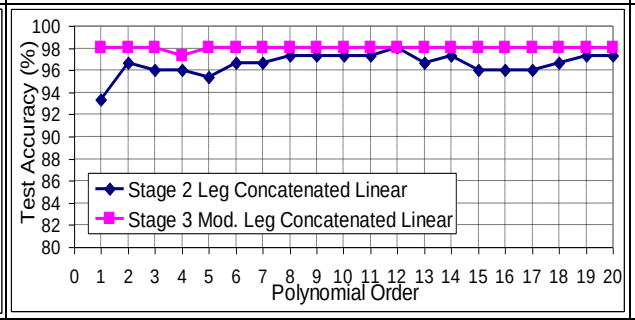
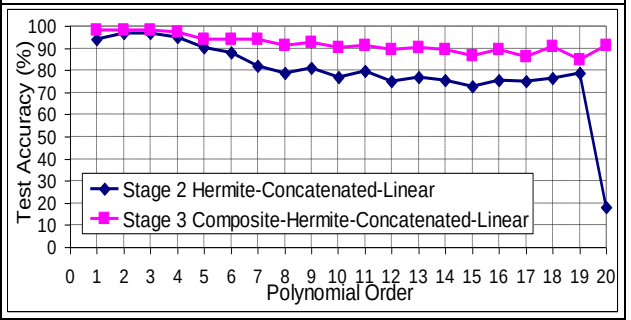
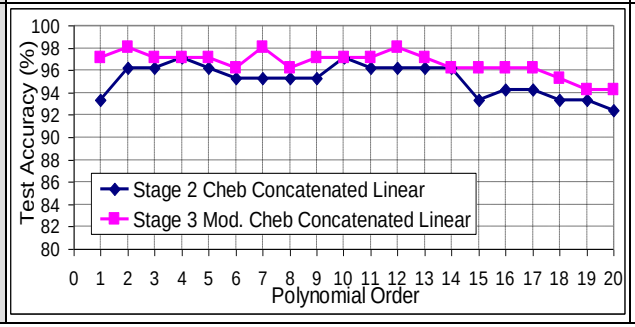
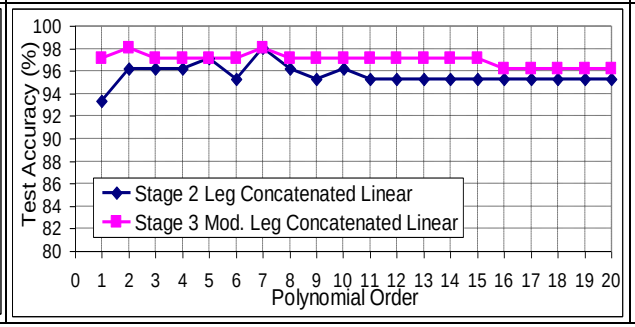
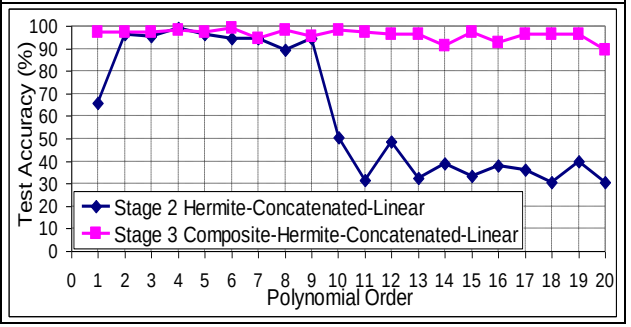
6.6.2 Evaluation of the multiplicative fusion process via the comparative experimental results of the concatenated-Stage 2 and Stage 3 kernels

Tables 6.22 and 6.23 also demonstrate the comparative experimental results of the scored classification accuracy of the concatenated-Stage 2 and Stage 3 kernels when both the linear and Gaussian kernels are utilized, respectively. In line with the experimental results presented in Section 4.4.5 (Chapter 4), it can be observed from the results obtained herein, for all of the polynomials and datasets investigated, that the concatenated-Stage 3 kernels have been able to score higher classification accuracy than their Stage 2 counterparts for most, if not all, of the polynomial orders tested. This again clearly confirms that the fusion by multiplication process, which takes place in the Stage 3 kernels, is also synergistic in nature as it acquires more accurate similarity measures than the Stage 2 kernels and, hence, improves the classification performance.

An observation worth noting, however, is that when a good kernel, such as the Gaussian, is used to measure the similarity between the image vectors processed using the concatenated approach, the resulting classification performance is very satisfactory, and only an infinitesimal improvement can be obtained by combining this kernel with another kernel. This can be observed, for example, from the results obtained from the Iris and Image Segmentation datasets in Table 6.23, where only a small improvement can be achieved when the concatenated-Stage 2 Gaussian kernels are fused by multiplication with another Gaussian kernel.

One can also notice that the fusion by multiplication process has also had a further smoothness influence on the classification performance of the Stage 3 kernels (alongside the smoothing effect already introduced by the concatenated approach). This can be clearly observed, for example in the Ionosphere, Iris, and Image Segmentation datasets, shown in Table 6.22, where the fusion by multiplication taking place in the Stage 3 concatenated-linear Hermite kernels tends to correct the sudden abrupt changes in the classification accuracy encountered in their Stage 2 counterparts. Interestingly, one can also notice that the utilization of the Gaussian kernel (instead of the linear kernel) in the experimental results illustrated in Table 6.23, has actually introduced a further level of smoothness amongst the polynomial orders, resulting in hardly any abrupt changes in the classification performance can be observed.

Table 6.22 Comparison between the concatenated-Stage 2 and Stage 3 linear kernels to validate the fusion by multiplication hypothesis.

	Chebyshev	Legendre	Hermite
Breast Cancer			
Ionosphere			
Iris			

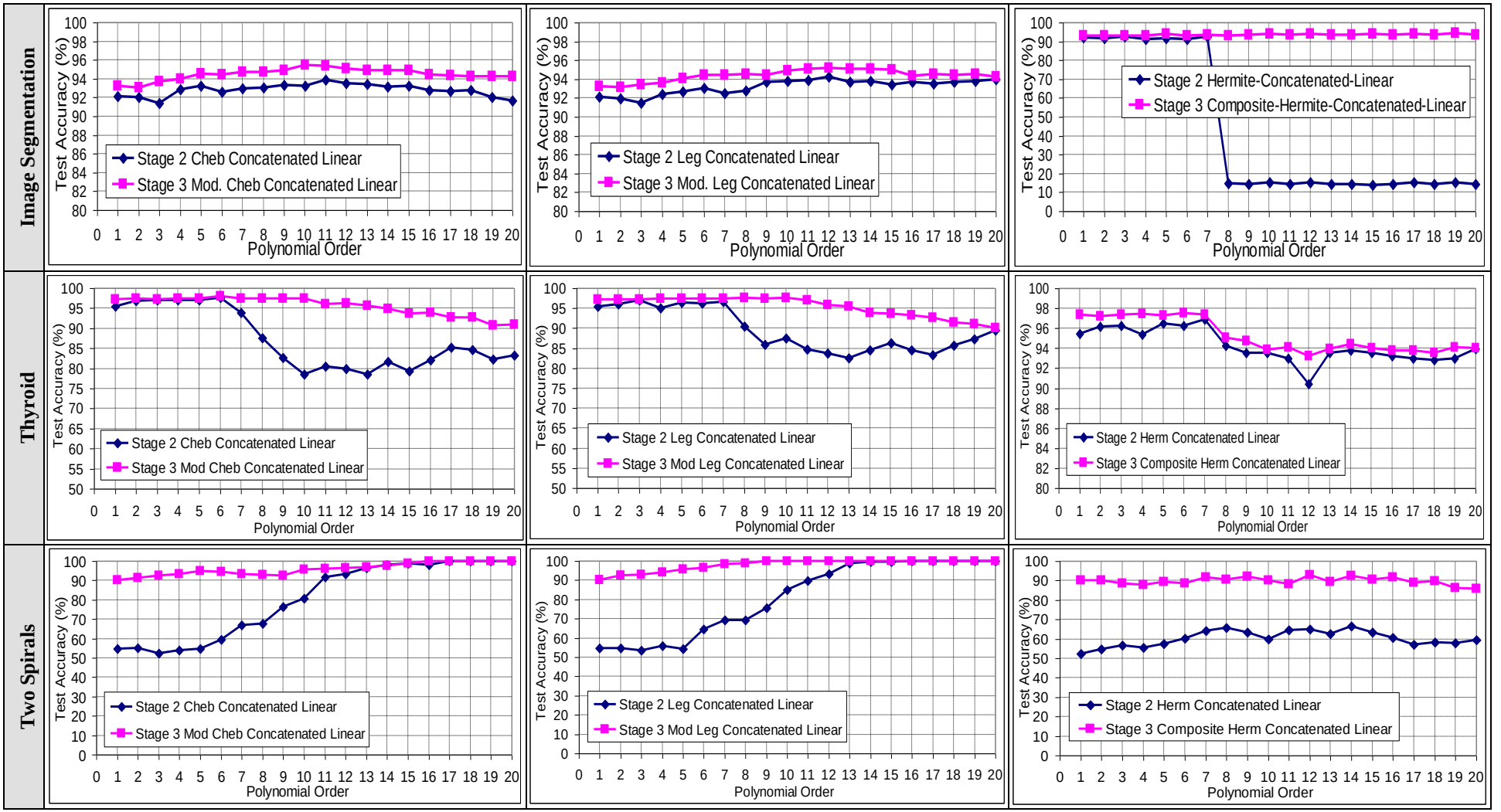
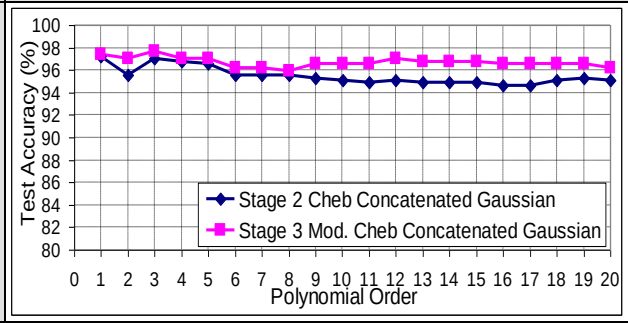
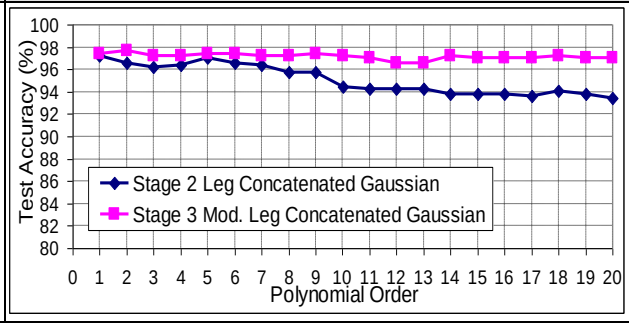
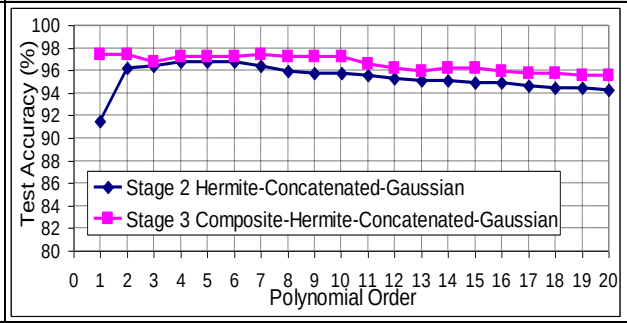
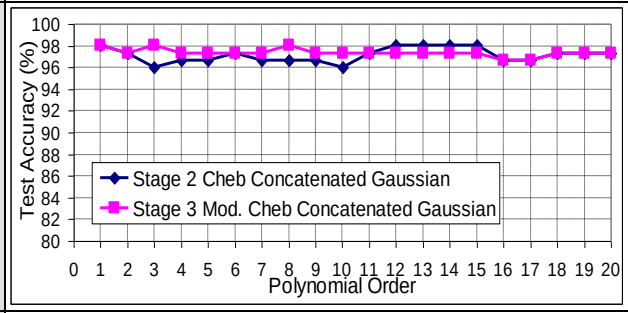
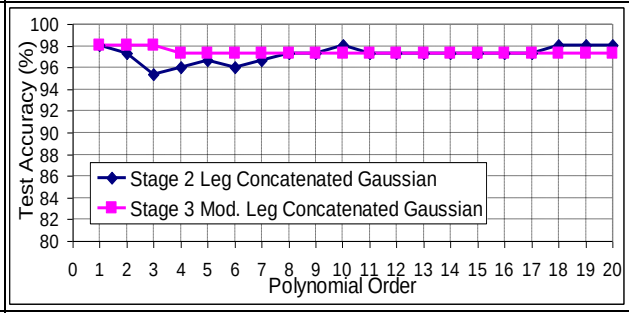
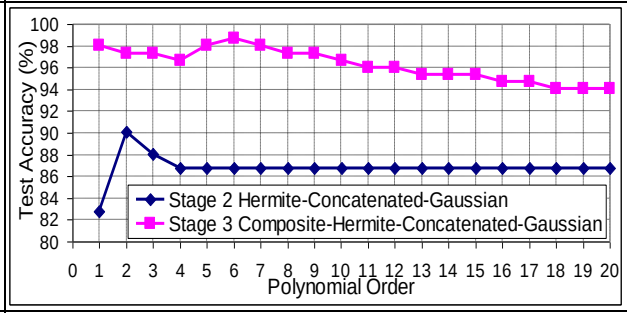
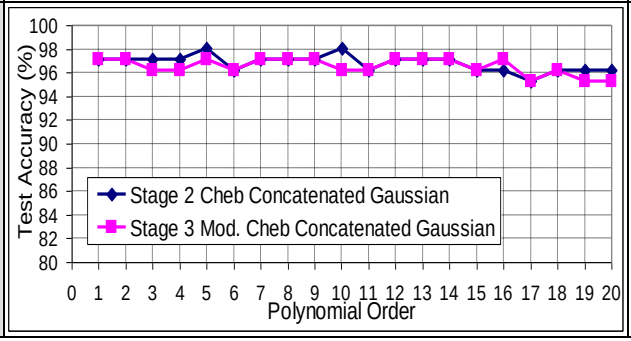
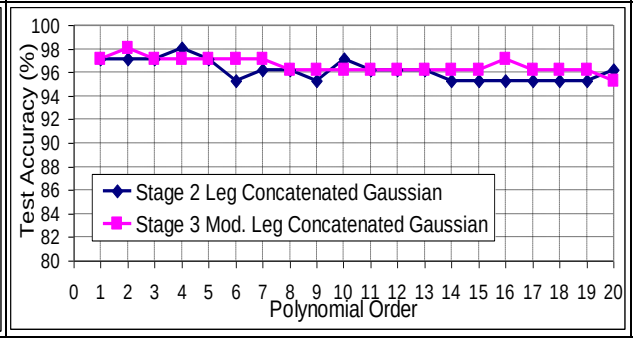
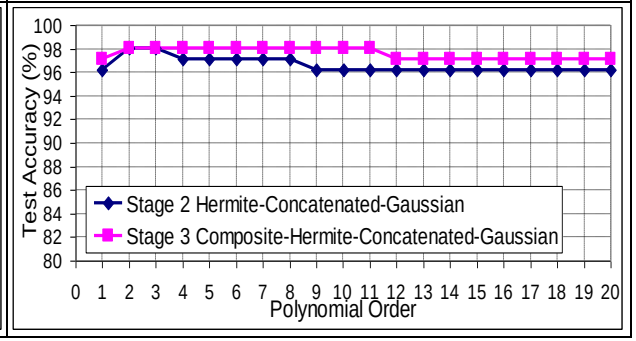


Image Segmentation
Thyroid
Two Spirals
Test Accuracy (%)
Polynomial Order
Stage 2 Cheb Concatenated Linear
Stage 3 Mod. Cheb Concatenated Linear
Stage 2 Leg Concatenated Linear
Stage 3 Mod. Leg Concatenated Linear
Stage 2 Hermite-Concatenated-Linear
Stage 3 Composite-Hermite-Concatenated-Linear
Stage 2 Cheb Concatenated Linear
Stage 3 Mod Cheb Concatenated Linear
Stage 2 Leg Concatenated Linear
Stage 3 Mod Leg Concatenated Linear
Stage 2 Herm Concatenated Linear
Stage 3 Composite Herm Concatenated Linear
Stage 2 Cheb Concatenated Linear
Stage 3 Mod Cheb Concatenated Linear
Stage 2 Leg Concatenated Linear
Stage 3 Mod Leg Concatenated Linear
Stage 2 Herm Concatenated Linear
Stage 3 Composite Herm Concatenated Linear

Table 6.23 Comparison between the concatenated-Stage 2 and Stage 3 Gaussian kernels to validate the fusion by multiplication hypothesis.

	Chebyshev	Legendre	Hermite
Breast Cancer			
Ionosphere			
Iris			

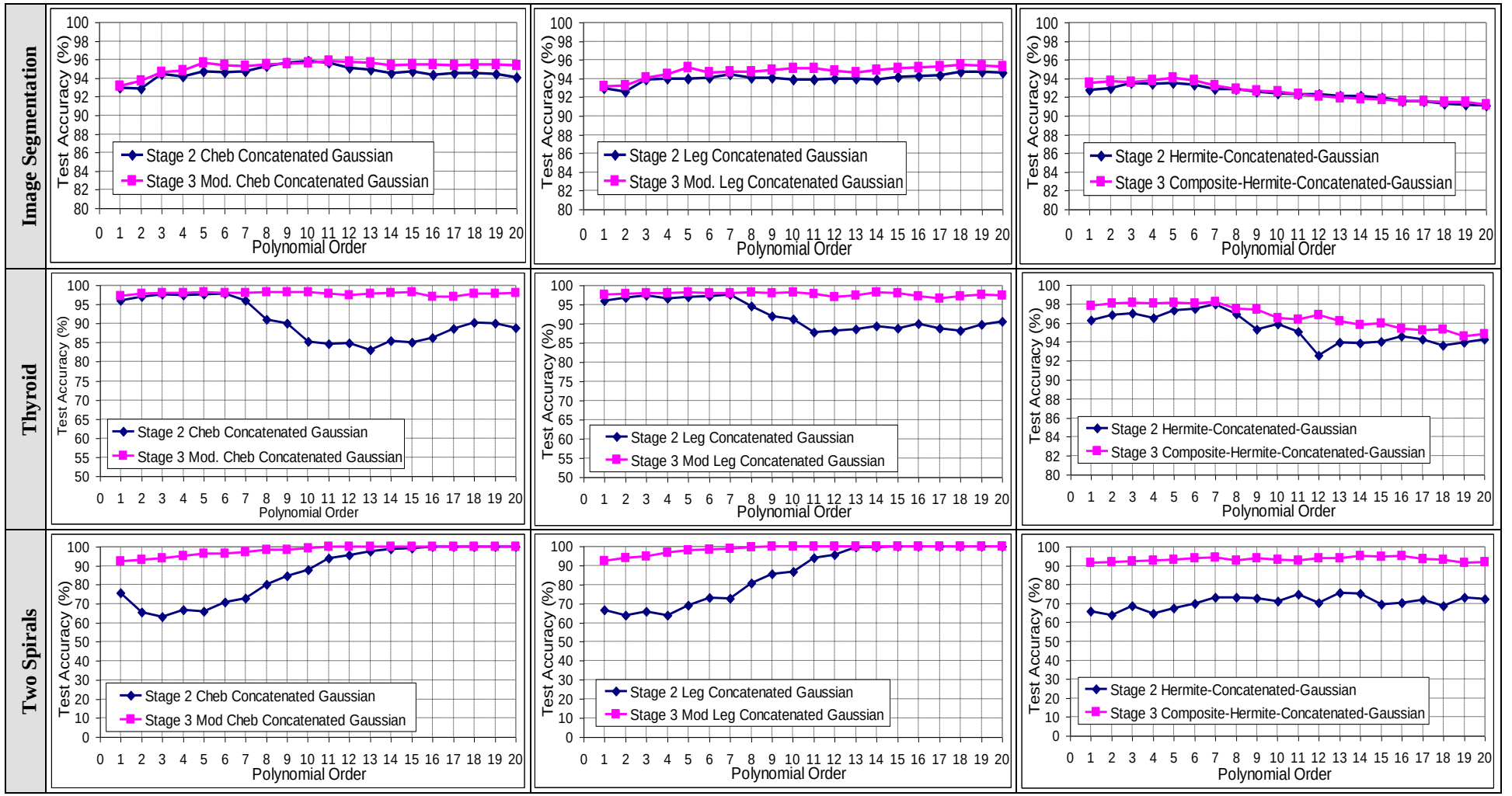


Image Segmentation
Thyroid
Two Spirals
Test Accuracy (%)
Polynomial Order
Stage 2 Cheb Concatenated Gaussian
Stage 3 Mod. Cheb Concatenated Gaussian
Stage 2 Leg Concatenated Gaussian
Stage 3 Mod. Leg Concatenated Gaussian
Stage 2 Hermite-Concatenated-Gaussian
Stage 3 Composite-Hermite-Concatenated-Gaussian
Stage 3 Mod Leg Concatenated Gaussian
Stage 3 Mod Cheb Concatenated Gaussian

6.7 Summary

This chapter introduced a new framework within which the construction of OPKs is perceived as a way of transforming the input space to another vector space (of the same dimensionality), using the employed orthogonal polynomial 'processing agents' (e.g., Chebyshev, Legendre, Hermite, etc.), prior to the kernel calculation step using a legitimate Mercer kernel. As such, the transformed space requires the data to always be formulated in a vector form regardless of the transformation function used to process the original input data vectors.

To achieve this purpose, the chapter proposed a new processing approach, based on vector concatenation, by which the employed polynomials process the input data vectors. Unlike the processing approaches previously proposed in the literature, referred to in this thesis as the pairwise and vectorial approaches, the concatenated approach proposed in this chapter, on the other hand, guarantees that when any polynomial function with any order processes an input data vector, it always produces another vector representing the transformed image of the original input vector in some new polynomial vector space of the same number of dimensions as the original input space. The chapter also proposed to apply the Gaussian kernel on the transformed image vectors (instead of the linear kernel used in previous approaches) to benefit from its renowned ability of implicitly mapping the input space to an infinite-dimensional feature space, as well as its accurate calculation of similarity measures, as exhibited by its shape characteristics.

Experimental results have shown the consistent superiority of the proposed concatenated approach over its vectorial counterpart for all the kernel stages of the similarity fusion framework. When using the linear kernel for both the concatenated and vectorial approaches, the overall average classification accuracy improvement factors were calculated to be: 23.906%, 23.016%, and 5.385% for the Stage 1, Stage 2, and Stage 3 kernels, respectively, depending on the polynomial and dataset used; making up a total average accuracy improvement factor of 17.436%, which clearly demonstrates the superiority of the proposed concatenated approach over its previously proposed vectorial counterpart.

Moreover, when the Gaussian kernel is utilized instead of the linear kernel, the observed classification accuracy improvements were even better. Similarly, the average classification accuracy improvement factors when utilizing the Gaussian kernel with the concatenated approach (over the linear kernel with the vectorial approach) were calculated to be: 29.434%, 30.796%, and 6.577% for the Stage 1, Stage 2, and Stage 3 kernels,

respectively, depending on the polynomial and dataset used; making up a total average accuracy improvement factor of 22.269%. Compared to the above calculated improvement scored by the linear kernel (17.436%), this figure of 22.269% clearly demonstrates the superiority of the Gaussian kernel over the linear kernel in terms of the resulting SVM classification performance.

Finally, the chapter ends with a re-evaluation of the developed similarity fusion framework using the newly proposed concatenated processing approach when both the linear and Gaussian kernel are used. Inline with the experimental results obtained from the vectoial approach back in Section 4.4, the results obtained in this chapter using the concatenated approach have also revealed that both of the summative and multiplicative fusion operations taking place in Stage 2 and Stage 3 kernels, respectively, are indeed synergistic as they were able to consistently score better results than if such exploitation of synergy was not achieved.

Chapter 7
Conclusions and Directions for Future Work

7.1 Conclusions

The research work presented in this thesis investigated the utilization of SVM kernels as similarity functions, in terms of how the input data are related to each other, rather than the mathematical properties of the implicit high-dimensional feature space, which one might not even be able to calculate. Given that a similarity function returns a scalar quantity indicative of how its two input vector arguments resemble each other (i.e., the larger the similarity, the more alike the two vectors are, and vice versa) [30], the thesis utilizes this intuitive 'similarity-based' property to define the shape characteristics of what one would wish a typical similarity measure kernel to look like, as illustrated in Figure 3.1. The similarity curve should be maximized, when its two input arguments are identical, and it should decay monotonically, when they depart away from each other, and are therefore more likely to belong to different classes. As such, by following some footsteps akin to Balcan et al. [20-22], the investigations conducted in this thesis demonstrate that the study of kernels as similarity measures can enable a machine learning practitioner to design kernels in terms of some natural and intuitive 'similarity-based' properties (e.g., the shape properties), which are more tangible quantities than the 'mathematical' properties of the implicit high-dimensional feature spaces, which one might not even be able to create.

Motivated by the plots of the Chebyshev kernel, reported by Ozer et al. [17], the thesis adopts the previously proposed SVM kernels that are constructed from orthogonal polynomials to explore how the classification performance is affected by how close the shape of the kernel is to the shape of the ideal similarity function. Chapter 3 extends Ozer et al.'s work to include a deep analysis on the shape characteristics of not only the kernels that are constructed from the Chebyshev polynomials, but also from other polynomials, such as the Legendre polynomials and Hermite polynomials. However, for a fair comparison among the graphical results of the polynomial kernels being investigated, the chapter focused on only the polynomial element of the kernel formulated with and without the sum in the form of $k(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle P_i(\mathbf{x}), P_i(\mathbf{z}) \rangle$ and $k(\mathbf{x},\mathbf{z}) = \langle P_n(\mathbf{x}), P_n(\mathbf{z}) \rangle$, respectively, to eliminate the effect of any other weighting or combining function. The chapter also presents a comparative analysis with the shape characteristics exhibited by

some of the traditional kernels that have also been in common use, such as the Gaussian, linear, homogeneous, and inhomogeneous polynomial kernels.

The outcome of the analyses of the shape characteristics of these two forms of polynomial kernels (i.e., with and without sum) revealed that, as the polynomial order increases, the kernels constructed with the sum have generally exhibited shape characteristics closer to the ideal similarity function than the kernels constructed without the sum, in terms of developing a maximum peak wherever the two input arguments are identical, and then decrease in value (although not in a monotonic behaviour), when the inputs depart away from each other. It was therefore apparent that the summation of the individual kernel components that takes place in the kernels with the sum does actually produce a synergistic effect on the accuracy of the similarity measures calculated by the underpinning kernels.

This phenomenon was investigated in more detail in Chapter 4 via a new three-stage similarity fusion framework whereby the overall kernel construction is broken down into the following three stages. Stage 1 kernels involves the kernels formulated using only the polynomial order under consideration, in the form of $k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle P_n(\mathbf{x}), P_n(\mathbf{z}) \rangle$. Stage 2 kernels, on the other hand, undertake a fusion by summation operation for all the preceding kernels from order 0 up to n in the form of $k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle P_i(\mathbf{x}), P_i(\mathbf{z}) \rangle$. Lastly, Stage 3 kernels undertake another fusion operation by multiplying Stage 2 kernels either with another kernel derived from their corresponding weighting function, or any other valid kernel, as $k_n^{S_3}(\mathbf{x},\mathbf{z}) = \left[\sum_{i=0}^{n} \langle P_i(\mathbf{x}), P_i(\mathbf{z}) \rangle \right] \times w(\mathbf{x},\mathbf{z})$. It is apparent that the kernels constructed this way from orthogonal polynomials are actually formulated using a summative and multiplicative mixture of base kernel building blocks that synergistically contribute towards achieving an accurate calculation of similarity measures, and, hence, enhancing classification performance.

To the extent that each of these individual kernel blocks can provide complementary information about the input data (i.e., similarity measures), the resulting performance of the classifier constructed from their combination is expected to outperform that of the best individual kernel building block. Following the vectorial processing approach proposed by Ozer et al. [17], whereby the orthogonal polynomials are utilized to process the input data vectors as a whole rather than their individual feature components, a number of experiments have been conducted on six benchmark datasets to validate such a hypothesis, using the kernels constructed from the Chebyshev, Legendre, and Hermite polynomials.

The comparative experimental results show that the summative fusion operation is indeed synergistic and that the average improvement in the classification accuracy due to the utilization of Stage 2 kernels over their Stage 1 counterparts was calculated to be 17.35%, depending on the dataset utilized and the polynomial function employed. Similarly, the results also show that Stage 3 kernels outperform their Stage 2 counterparts by an average accuracy improvement factor of 19.16%.

On the other hand, while the analysis of the shape characteristics of Stage 2 kernels has shown that they exhibit better measures of similarity than their Stage 1 counterparts, they were still found to fail to display a strictly monotonic decay behaviour over the whole of the normalized input space. For example, the Stage 2 kernels constructed from the Chebyshev polynomials and Legendre polynomials were found to be maximized, when their two input arguments are identical and then decrease in value, when they start to depart from each other, but only up to a certain threshold, after which they start to increase again and then fluctuate in a wavy pattern. As such, their shape behaviour after this threshold is not completely monotonic in nature, a fact which renders the similarity measures calculated between any two data points that happen to be outside the monotonic threshold window to be inaccurate, and could therefore affect destructively the resulting classification performance.

This problem is addressed in Chapter 5 by proposing a simple adaptive data normalization approach that confines the data to the regions, where the employed kernel demonstrates the sought after ideal monotonic decrease characteristic. By this way, the possibility of any data point to be located outside the monotonic threshold window of the employed kernel is eliminated. Experimental results using the three polynomials under investigation (Chebyshev, Legendre, and Hermite) demonstrated the effectiveness of this approach on the resulting SVM classification accuracy, compared to the results achieved when the data are normalized to the standard region of [-1,+1]. The analysis of the results for the first 10 polynomial orders showed that this approach can further enhance the classification accuracy by an average improvement factor of 11.772% (and a standard deviation of 30.019%) depending on the dataset and the polynomial function utilized.

Last but not least, on studying closely the ‘vectorial’ polynomial-processing approach proposed by Ozer et al. [17], together with its ‘pairwise’ predecessor proposed by Ye et al. [31], it has been realized that the construction of the kernel from these polynomials can be viewed as a two step process. First, the employed polynomial (Chebyshev, Legendre, etc.) is utilized to ‘process’ the input vectors to produce another set of vectors in some

polynomial vector space of the same dimensionality as the input space. Second, as proposed by Ozer et al. apply the linear kernel to calculate the similarity between the quantities produced by the polynomials, as demonstrated by the formulation of Stage 1 and Stage 2 kernels, as $k_n^{S_1}(\mathbf{x},\mathbf{z}) = \langle P_n(\mathbf{x}), P_n(\mathbf{z}) \rangle$ and $k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n} \langle P_i(\mathbf{x}), P_i(\mathbf{z}) \rangle$, respectively.

This approach is perceived in this thesis as a way of 'transforming' the input space to another vector space (ideally without undue loss of input-space dimensionality), prior to the kernel calculation step. It was observed, however, that the even polynomial orders of the vectorial approach produce a scalar quantity rather than a vector. This transformation discards potentially important attributes (such as the angle), associated with the structure of the original input vectors, that are essential for the similarity measure calculation process and eventually the estimation of the class label corresponding to the input vector. Moreover, the vectorial processing approach was also found to be applicable only to those polynomial functions consisting of only odd- or only even-order parity, such as the Chebyshev polynomials and Legendre polynomials, and therefore it cannot be generalized to any other polynomials not sharing this characteristic, such as the Laguerre polynomials, for example. Also, the application of the linear kernel is not a good choice, especially when used to solve non-linearly separable classification problems. This is because it is neither a good measure of similarity to be used (as compared to the shape characteristics of the ideal similarity function), nor does it trigger a transformation to higher number of dimensions so that linear separability between overlapped classes can be enhanced in the feature space.

To rectify these issues, Chapter 6 proposed a new processing methodology, referred to as the 'concatenated' processing approach, whereby the employed polynomial function is utilized to process the individual features of the input vectors and then the new 'processed' features are concatenated to form a new set of vectors (of the same dimensionality as the original input space) on which the kernel is to be afterwards evaluated. This way, the processed quantities will always be formulated in a vector form, and hence the attributes embedded in the structure of the original vectors are retained as much as possible. Moreover, using the proposed concatenated approach, any polynomial function can also be used to process the input vectors with no restriction on the parity of the polynomial, as was the case, for example, with the vectorial approach. Furthermore, due to its better pictorial characteristics (in relation to the shape of the curve representing the similarity measure), as well as due to its ability to implicitly map the input space to an infinite-dimensional feature space, the chapter also proposes the use of the Gaussian kernel, instead of the linear kernel,

to measure the similarity between the vectors produced by concatenating the outputs of the polynomials. When both the Gaussian kernel and the concatenated processing approach are used together, the resulting Stage 1 and Stage 2 kernels were then formulated as:

$$k_n^{S_1}(\mathbf{x},\mathbf{z}) = \exp\left(-\gamma\left\|[P_n(x_1),\cdots,P_n(x_m)]-[P_n(z_1),\cdots,P_n(z_m)]\right\|^2\right) \text{ and}$$

$$k_n^{S_2}(\mathbf{x},\mathbf{z}) = \sum_{i=0}^{n}\exp\left(-\gamma\left\|[P_i(x_1),\cdots,P_i(x_m)]-[P_i(z_1),\cdots,P_i(z_m)]\right\|^2\right), \text{ respectively.}$$

Experimental results presented in the thesis show the consistent superiority of the proposed concatenated approach over its vectorial counterpart, for all the stages of the similarity fusion framework, and using the same three polynomial functions under investigation (namely, Chebyshev, Legendre, and Hermite). When using the linear kernel with the concatenated processing approach compared to the vectorial processing approach, the overall average accuracy improvement factor was calculated to be 17.436%. When the Gaussian kernel was used with the concatenated approach, the average accuracy improvement factor (over the linear kernel with the vectorial approach) was calculated to be 22.269%, which clearly shows the effectiveness of both the concatenated processing approach over its vectorial counterpart, as well as the utilization of a more accurate similarity measure kernel.

In the general sense, it can be concluded that the effectiveness of an SVM kernel is actually a combination of two important elements. (1) its ability to map the input space to implicit high-dimensional feature spaces (as dictated by its prescribed positive semi-definiteness property), where linear separability between overlapped classes can be improved, and hence easier to classify; and (2) its ability to accurately calculate the similarity between its two input arguments, to aid the classifier decide as to whether they belong to the same or different classes. The novel approaches presented in this thesis have shown how effective the latter definition of kernels as similarity functions can be compared to their former role as implicit mapping tools to high-dimensional feature spaces.

The research work presented in this thesis was therefore able to fully achieve the aim and objectives, as set earlier in Chapter 1 – Section 1.3. Not only was a systematic investigation conducted to address how the SVM kernels can be used as similarity measures, but also novel analytical solutions have been proposed and empirically evaluated using the polynomial kernels under study and a diverse set of benchmark datasets. The

statistical significance tests of the comprehensive experimental results achieved in this thesis are also summarised in Appendix B, the majority of which were found to be statistically significant within a statistical confidence level of 95%.

To the knowledge of the author, no previous work in the literature identified or addressed the research problems highlighted in this thesis. Therefore, neither the solutions or approaches presented in this thesis have been reported in the past literature, nor the experimental investigations conducted in this thesis have been done before. As such, the comparative analyses, research investigations and findings, and the empirical assessments conducted in this thesis are all considered herein as novel and original contributions to knowledge.

7.2 Future Work

One possible direction of future work is to investigate how the proposed concatenated processing approach would perform in terms of other classifier performance assessment measures, such as the number of support vectors and the processing time. It would also be interesting to investigate the impact of the utilization of the polynomial kernels studied in this thesis, but implemented with the proposed concatenated approach, on other kernel-based learning algorithms, such as regression and kernel principal component analysis. Given the promising results achieved from the transformation of the input space to another vector space, prior to the kernel calculation step, using the studied polynomial functions, it would also be interesting to explore the effectiveness of using other transformation or processing functions that could enhance the classification performance even further.

In the general sense, however, it should never be forgotten that the statistical pattern classification process is heavily anchored on the concept of ‘similarity’ (as highlighted in this thesis), whereby patterns from the same class are usually assumed to share more similar characteristics with each other than those belonging to different classes. This similarity-based concept, however, is also applicable not only to supervised learning algorithms, as is the case for example with the SVM algorithm studied in this thesis, but also to various other areas within the field of machine learning, such as clustering, which target the solution of many different complex problems in such areas as information retrieval, bioinformatics, and data compression. It would therefore be interesting to explore the effectiveness of the approaches presented in this thesis (e.g., the utilization of orthogonal polynomials, the concatenated processing approach, etc.) in unsupervised learning algorithms as well, and whether they can also facilitate the solutions to the real-life problems that they address.

References

[1] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. New York, USA: Cambridge University Press, 2004.

[2] N. Cristianini and B. Schölkopf, "Support Vector Machines and Kernel Methods: The New Generation of Learning Machines," *AI Magazine,* vol. 23, pp. 31-41, 2002.

[3] A. K. Jain, R. P. W. Duin, and J. Mao, "Statistical Pattern Recognition: A Review," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 22, pp. 4-37, 2000.

[4] S. Theodoridis and K. Koutroumbas, *Pattern Recognition,* 4, revised ed., Academic Press, 2008.

[5] V. N. Vapnik, "An Overview of Statistical Learning Theory," *IEEE Transactions on Neural Networks,* vol. 10(5), pp. 988-999, 1999.

[6] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," *proceedings of the Fifth Annual Workshop of Computational Learning Theory - COLT,* ACM, Pittsburgh, USA, pp. 144-152, 1992.

[7] K.-R. Müller, S. Mika, G. Rätsch, et al., "An Introduction to Kernel-Based Learning Algorithms," *IEEE Transactions on Neural Networks,* vol. 12, No. 2, pp. 181-201, 2001.

[8] J.-P. Vert, K. Tsuda, and B. Scho¨lkopf, *A primer on kernel methods*: Chapter 2, pages 35–70, MIT Press, Cambridge, Massachusetts, 2004.

[9] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*: MIT Press, MA, USA, 2002.

[10] C. Cortes and V. Vapnik, "Support Vector Networks," *Machine Learning,* vol. 20, pp. 273-297, 1995.

[11] V. N. Vapnik, *Statistical Learning Theory*: Wiley, USA, 1998.

[12] S. Abe, *Support Vector Machines for Pattern Classification*: second edition, Springer-Verlag London, 2010.

[13] C. Campbell, "Kernel methods: a survey of current techniques," *Neurocomputing,* vol. 48, no.1-4, pp. 63–84, 2002.

[14] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. USA: Cambridge University Press, 2000.

[15] C. R. Souza, "Kernel Functions for Machine Learning Applications," Online: http://crsouza.blogspot.co.uk/2010/03/kernel-functions-for-machine-learning.html , 17 March, 2010.

[16] T. Hofmann, B. Sch¨olkopf, and A. J. Smola, "Kernel methods in machine learning," *Annals of Statistics,* vol. 36, no. 3, pp. 1171–1220, 2008.

[17] S. Ozer, C. H. Chen, and H. A. Cirpan, "A set of new Chebyshev kernel functions for support vector machine pattern classification," *Pattern Recognition,* vol. 44, pp. 1435-1447, 2011.

[18] M. G. Genton, "Classes of Kernels for Machine Learning: A Statistics Perspective," *Journal of Machine Learning Research,* vol. 2, pp. 299-312, 2001.

[19] Y. Chen, E. K. Garcia, M. R. Gupta, et al., "Similarity-based Classification: Concepts and Algorithms," *Journal of Machine Learning Research,* vol. 10, pp. 747-776, 2009.

[20] M.-F. Balcan, A. Blum, and N. Srebro, "A theory of learning with similarity functions," *Machine Learning,* vol. 72, pp. 89-112, 2008.

[21] M.-F. Balcan, A. Blum, and N. Srebro, "Improved Guarantees for Learning via Similarity Functions," *Proceedings of the 21st Annual Conference on Learning Theory (COLT)*, pp. 287-298, 2008.

[22] N. Srebro, "How Good is a Kernel When Used as a Similarity Measure," *Proceedings of the 20th Annual Conference on Learning Theory (COLT)*, 2007.

[23] P. Kar and P. Jain, "Supervised Learning with Similarity Functions," 26th Annual Conference on Neural Information Processing Systems (NIPS), 2012.

[24] G. Loosli, S. Canu, and C. S. Ong, "Learning SVM in Krein Spaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 38, pp. 1204-1216, June, 2016.

[25] I. Alabdulmohsin, X. Gao, and X. Zhang, "Support vector machines with indefinite kernels," Sixth Asian Conference on Machine Learning (ACML), Nha Trang, Vietnam, pp. 32-47, 2014.

[26] S. Gu and Y. Guo, "Learning SVM Classifiers with Indefinite Kernels," *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[27] B. Haasdonk, "Feature Space Interpretation of SVMs with Indefinite Kernels," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 27, no. 4, pp. 482-492, April, 2005.

[28] Y. Chen, M. R. Gupta, and B. Recht, "Learning Kernels from Indenite Similarities," *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009.

[29] R. Luss and A. d'Aspremont, "Support vector machine classification with indefinite kernels," *Advances in Neural Information Processing Systems,* 2007.

[30] K. Abou-Moustafa, "What Is the Distance Between Objects in a Data Set? A brief review of distance and similarity measures for data analysis," *IEEE PULSE,* pp. 41-47, March/April, 2016.

[31] N. Ye, R. Sun, Y. Liu, et al., "Support vector machine with orthogonal Chebyshev kernel," *Proceedings of the 18th International Conference on Pattern Recognition (ICPR)*, Hong Kong, pp. 752-755, 2006.

[32] X. Wei and Z. Pan, "Chebyshev kernel with orthogonal features," *Proceeding of the International Conference on Computer Science and Information Processing (CSIP)*, pp. 941-944, August, 2012.

[33] S. Z. Jafarzadeh, M. Aminian, and S. Efati, "A set of new kernel function for support vector machines: An approach based on Chebyshev polynomials," *Proceedings of the 3rd International Conference on Computer and Knowledge Engineering (ICCKE)*, Ferdowsi University of Mashhad, pp. 412-416, 2013.

[34] Z.-B. Pan, H. Chen, and X.-H. You, "Support vector machine with orthogonal Legendre kernel," *Proceedings of the International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR)*, Xian, China, pp. 125-130, July, 2012.

[35] V. H. Moghaddam and J. Hamidzadeh, "New Hermite orthogonal polynomial kernel and combined kernels in support vector machine classifier," *Pattern Recognition*, vol. 60, pp. 921-935, December, 2016.

[36] V. Vapnik, *The Nature of Statistical Learning Theory*: Springer, 1995.

[37] B. Scholkopf, S. Mika, C. J. C. Burges, et al., "Input Space Versus Feature Space in Kernel-Based Methods," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, 1999.

[38] Y.-J. Lee and S.-Y. Huang, "Reduced Support Vector Machines: A Statistical Theory," *IEEE Transactions on Neural Networks*, vol. 18, no. 1, pp. 1-13, 2007.

[39] B. Scholkopf, K.-K. Sung, C. J. C. Burges, et al., "Comparing Support Vector Machines with Gaussian Kernels to Radial Basis Function Classifiers," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2758-2765, 1997.

[40] D. Meyer, F. Leisch, and K. Hornik, "The support vector machine under test," *Neurocomputing*, vol. 55, pp. 169-186, 2003.

[41] G. Wang, "A Survey on Training Algorithms for Support Vector Machine Classifiers," *proceedings of the Fourth International Conference on Networked Computing and Advanced Information Management*, pp. 123-128, 2008.

[42] R. Hable and A. Christmann, "On qualitative robustness of support vector machines," *Journal of Multivariate Analysis*, vol. 102, pp. 993-1007, 2011.

[43] B. Scholkopf, A. Smola, R. C. Williamson, et al., "New Support Vector Algorithms," *Neural Computation*, vol. 12, pp. 1207-1245, 2000.

[44] F. Zhu, N. Ye, S. Xu, et al., "Support Vectors Classification and Incremental Learning," 6th IEEE Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, pp. 206-210, 2011.

[45] A. J. SMOLA and B. SCHOLKOPF, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, pp. 199–222, 2004.

[46] H. Byun and S.-W. Lee2, "Applications of Support Vector Machines for Pattern Recognition: A Survey," *the International workshop on pattern recognition with support vector machines (SVM2002)*, Niagara Falls, Canada, pp. 213–236, 2002.

[47] *Support vector machine*. Available: http://en.wikipedia.org/wiki/Support_vector_machine , April, 2014.

[48] *Duality (optimization)*. Available: http://en.wikipedia.org/wiki/Dual_problem , April, 2014.

[49] M. E. Abbasnejad, D. Ramachandram, and R. Mandava, "A survey of the state of the art in learning the kernels," *Knowledge and Information Systems*, vol. 31, no. 2, pp. 193-221, 2012.

[50] B. McCune, J. B. Grace, and D. L. Urban, *Analysis of Ecological Communities*: Chapter 6:Distance Measures, MjM Software Design, Gleneden Beach, Oregon, USA, http://www.pcord.com, 2002.

[51] J. H. Friedman, "Flexible Metric Nearest Neighbor Classification," Technical Report, Stanford University, 1994.

[52] T. Hastie and R. Tibshirani, "Discriminant Adaptive Nearest Neighbor Classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, No. 6, 1996.

[53] C. Domeniconi, J. Peng, and D. Gunopulos, "Locally adaptive metric nearest neighbor classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, No. 9, pp. 1281-1285, 2002.

[54] J. Wang, P. Neskovic, and L. N. Cooper, "Improving nearest neighbor rule with a simple adaptive distance measure," *Pattern Recognition Letters*, vol. 28, pp. 207–213, 2007.

[55] A. Bellet, A. Habrard, and M. Sebban, "A Survey on Metric Learning for Feature Vectors and Structured Data," *ArXiv e-prints*, 2014.

[56] B. Kulis, "Metric Learning: A Survey," *Machine Learning*, vol. 5, No. 4, pp. 287-364, 2012.

[57] L. Yang, "Distance Metric Learning: A Comprehensive Survey," Technical Report, Michigan State University, Department of Computer Science and Engineering, 2006.

[58] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, et al., "On kernel-target alignment," Neural Information Processing Systems Conference (NIPS), pp. 367–373, 2001.

[59] M. Gonen and E. Alpaydın, "Multiple Kernel Learning Algorithms," *Journal of Machine Learning Research*, vol. 12, pp. 2211-2268, 2011.

[60] M. Hofmann, "Support Vector Machines - Kernels and the Kernel Trick," An elaboration for the Hauptseminar "Reading Club: Support Vector Machines" Report, June, 2006.

[61] Jian-Wu Xu, Puskal P. Pokharel, Kyu-Hwa Jeong, et al., "An Explicit Construction of a Reproducing Gaussian Kernel Hilbert Space," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pp. 573-576, 2006.

[62] J.-W. Xu, António R. C. Paiva, I. Park, et al., "A Reproducing Kernel Hilbert Space Framework for Information-Theoretic Learning," *IEEE Transactions on Signal Processing*, vol. 56, no. 12, pp. 5891-5902, 2008.

[63] I. Steinwart, D. Hush, and C. Scovel, "An Explicit Description of the Reproducing Kernel Hilbert Spaces of Gaussian RBF Kernels," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4635-4643, 2006.

[64] H. Daum'eIII, "From zero to reproducing kernel hilbert spaces in twelve pages or less," February, 2004.

[65] B. D. Moor, "Kernel Models for Large Scale Applications," PhD Thesis, KATHOLIEKE UNIVERSITEIT LEUVEN, FACULTEIT TOEGEPASTE WETENSCHAPPEN, DEPARTEMENT ELEKTROTECHNIEK, 2004.

[66] I. Guyon, B. Boser, and V. Vapnik, "Automatic capacity tuning of very large VC-dimension classifiers," *In: Hanson S.J., Cowan J.D., and Giles C.L. (Eds.), Advances in Neural Information Processing Systems 5. Morgan Kaufmann Publishers,* , pp. 147–155, 1993.

[67] H.-T. Lin and C.-J. Lin, "A Study on Sigmoid Kernels for SVM and the Training of non-PSD Kernels by SMO-type Methods," Technical report, Department of Computer Science and Information Engineering, National Taiwan University. Available at: http://www.csie.ntu.edu.tw/~cjlin/papers/tanh.pdf, 2003.

[68] Y. Tan and J. Wang, "A support vector machine with a hybrid kernel and minimal Vapnik-Chervonenkis dimension," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, pp. 385-395, April, 2004.

[69] H. Song, Z. Ding, C. Guo, et al., "Research on Combination Kernel Function of Support Vector Machine," *Proceedings of the 2008 International Conference on Computer Science and Software Engineering*, Wuhan, Hubei, pp. 838-841, December, 2008.

[70] L. Cong-cong, G. Ai-ling, and L. Dan, "Combined Kernel SVM and Its Application on Network Security Risk Evaluation," International Symposium on Intelligent Information Technology Application Workshops, IITAW '08, pp. 36-39, Shanghai, December, 2008.

[71] M. Varma and B. R. Babu, "More Generality in Efficient Multiple Kernel Learning," *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009.

[72] C. Cortes, M. Mohri, and A. Rostamizadeh, "Learning Non-Linear Combinations of Kernels," *Advances in Neural Information Processing Systems (NIPS) 22*, 2010b.

[73] B. V. Dasarathy, "Industrial applications of multi-sensor multi-source information fusion," *Proceedings of the IEEE International Conference on Industrial Technology*, Dynetics Inc., Huntsville, AL, USA, vol. 1, pp. 5-11, 2000.

[74] A. R. Mirza, "Data fusion architectures for sensor platforms," *Proceedings of the 2008 IEEE Aerospace Conference*, Big Sky, MT, USA, pp. 1-13, March, 2008.

[75] B. V. Dasarathy, "Decision fusion strategies in multisensor environments," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 21, pp. 1140-1154, 1991.

[76] R. J. Stanley, P. D. Gader, and K. C. Ho, "Feature and decision level sensor fusion of electromagnetic induction and ground penetrating radar sensors for landmine detection with hand-held units," *Information Fusion*, vol. 3, pp. 215-223, 2002.

[77] T. Nicosevici, R. Garcia, M. Carreras, et al., "A review of sensor fusion techniques for underwater vehicle navigation," *Proceedings MTS/IEEE Techno-Oceans Conference Exhibition* pp. 1600-1605, Kobe, Japan, 2004.

[78] Z. Liu, D. S. Forsyth, J. P. Komorowski, et al., "Survey: State of the Art in NDE Data Fusion Techniques," *IEEE Transactions on Instrumentation and Measurement*, vol. 56, No. 6, pp. 2435-2451, December, 2007.

[79] M. D. Mura, S. Prasad, F. Pacifici, et al., "Challenges and Opportunities of Multimodality and Data Fusion in Remote Sensing," *Proceedings of the IEEE*, vol. 103, no. 9, pp. 1585 - 1601, September, 2015.

[80] P. Kocmanova, L. Zalud, F. Burian, et al., "Multispectral Data Fusion for Robotic Reconnaissance and Mapping," *Proceedings 11th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, vol. 2, Vienna, pp. 459 - 466, September, 2014.

[81] Y. M. Baek, S. Tanaka, K. Harada, et al., "Robust Visual Tracking of Robotic Forceps Under a Microscope Using Kinematic Data Fusion," *IEEE/ASME Transactions on Mechatronics*, vol. 19, no. 1, pp. 278 - 288, February, 2014.

[82] T. Adali, Y. Levin-Schwartz, and V. D. Calhoun, "Multimodal Data Fusion Using Source Separation: Application to Medical Imaging," *Proceedings of the IEEE*, vol. 103, pp. 1494 - 1506, September, 2015.

[83] L. Wang, B. Li, and L. Tian, "Multimodal Medical Volumetric Data Fusion Using 3-D Discrete Shearlet Transform and Global-to-Local Rule," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 1, pp. 197 - 206 January, 2014

[84] L. I. Kuncheva, J. C. Bezdek, and R. P. W. Duin, "Decision templates for multiple classifier fusion: an experimental comparison " *Pattern Recognition,* vol. 34, pp. 299-314, 2001.

[85] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*: John Wiley & Sons Inc., 2004.

[86] J. Kittler, M. Hatef, R. P. W. Duin, et al., "On Combining Classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 20, no. 3, pp. 226-239, March, 1998.

[87] J. Yang, J.-y. Yang, D. Zhang, et al., "Feature fusion: Parallel strategy vs. serial strategy," *Pattern Recognition,* vol. 36, pp. 1369-1381, 2003.

[88] U. G. Mangai, S. Samanta, S. Das, et al., "A Survey of Decision Fusion and Feature Fusion Strategies for Pattern Classification," *IETE Technical Review,* vol. 27, no. 4, pp. 293-307, Jul-Aug, 2010.

[89] W.-J. Lee, S. Verzakov, and R. P. W. Duin, "Kernel combination versus classifier combination," *Proceedings of the 7th international conference on Multiple classifier systems,* pp. 22-31, Prague, Czech Republic, May, 2007.

[90] A. Rakotomamonjy, F. R. Bach, S. e. Canu, et al., "SimpleMKL," *Journal of Machine Learning Research,* vol. 9, pp. 2491-2521, 2008.

[91] G. R. G. Lanckriet, N. Cristianini, P. Bartlett, et al., "Learning the Kernel Matrix with Semidefinite Programming," *Journal of Machine Learning Research,* vol. 5, pp. 27-72, 2004.

[92] S. Sonnenburg, G. Rätsch, C. Schäfer, et al., "Large Scale Multiple Kernel Learning," *Journal of Machine Learning Research,* vol. 7, pp. 1531–1565, 2006.

[93] Y. Chen and M. R. Gupta, "Fusing Similarities and Kernels for Classification," *Proceeding of the 12th International Conference on Information Fusion,* Seattle, WA, USA, pp. 474-481, 2009.

[94] D. Windridge and F. Yan, "Kernel combination via debiased object correspondence analysis," *Information Fusion,* vol. 27, pp. 228–239, 2016.

[95] J. Bao, Y. Chen, L. Yu, et al., "A multi-scale kernel learning method and its application in image classification," *Neurocomputing,* vol. 257, pp. 16-23, 2017.

[96] S. S. Shiju, A. Salim, and S. Sumitra, "Multiple kernel learning using composite kernel functions," *Engineering Applications of Artificial Intelligence,* vol. 64, pp. 391-400, 2017.

[97] T. Phienthrakul and B. Kijsirikul, "GPES: An algorithm for evolving hybrid kernel functions of Support Vector Machines," *Proceedings of the IEEE Congress on Evolutionary Computation (CES),* Singapore, pp. 2636-2643, 2007.

[98] L. Dio¸san, A. Rogozan, and J.-P. Pecuchet, "Improving classification performance of Support Vector Machine by genetically optimising kernel shape and hyper-parameters," *Applied Intelligence,* vol. 36, No. 2, pp. 280-294, March 2012.

[99] M.-F. Balcan and A. Blum, "On a Theory of Learning with Similarity Functions," *Proceedings of the 23rd International Conference on Machine Learning,* Pittsburgh, PA, pp. 73–80, 2006.

[100] G. Wu, E. Y. Chang, and Z. Zhang, "An Analysis of Transformation on Non-Positive Semidefinite Similarity Matrix for Kernel Machines," Proceedings of the 22nd International Conference on Machine Learning (ICML), 2005.

[101] D. Huson, "SVMs and Kernel Functions," Algorithms in Bioinformatics II, SoSe'07, ZBIT, 2007.

[102] F. Zhou, Z. Fang, and J. Xu, "Constructing support vector machine kernels from orthogonal polynomials for face and speaker verification," *Proceedings of the Fourth International Conference on Image and Graphics (ICIG)*, Sichuan, China, pp. 627-632, August, 2007.

[103] S. Ozer and C. H. Chen, "Generalized Chebyshev kernels for support vector classification," *Proceedings of the 19th International Conference on Pattern Recognition ICPR*, Tampa, FL, pp. 1-4, December, 2008.

[104] A. Afifi, "Laguerre kernels–based SVM for image classification," *International Journal of Advanced Computer Science and Applications (IJACSA),* vol. 5, No. 1, pp. 15-23, 2014.

[105] M. Tian and W. Wang, "Research on the properties of orthogonal polynomial kernel functions," Pattern Recognition and Artificial Intelligence (in Chinese), 27(5), pp. 386-393, 2014.

[106] R. Zhang, W. Wang, Y. Zhang, et al., "Legendre kernel function for support vector classification," Computer Science (Chinese), 39(7), pp. 222-224, 2012.

[107] J. Zhao, G. Yan, B. Feng, et al., "An adaptive support vector regression based on a new sequence of unified orthogonal polynomials," *Pattern Recognition* vol. 46(3), p. 899−913, 2013.

[108] M. Tian and W. Wang, "Some sets of orthogonal polynomial kernel functions," *Applied Soft Computing,* vol. In Press, Accepted Manuscript, pp. 1-33, 2017.

[109] J. Qu, Z. Zhang, X. Luo, et al., "State recognition of viscoelastic sandwich structures based on permutation entropy and generalized Chebyshev support vector machine," IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Pisa, Italy, pp. 957-962, 2015.

[110] H. G. Jung and G. Kim, "Support Vector Number Reduction: Survey and Experimental Evaluations," *IEEE Transactions on Intelligent Transportation Systems,* vol. 15(2), pp. 463-476, 2014.

[111] F. Fleuret and H. Sahbi, "Scale-Invariance of Support Vector Machines based on the Triangular Kernel," 3rd International Workshop on Statistical and Computational Theories of Vision, Paris, INRIA, 2003.

[112] *Legendre Polynomials*. Available: http://en.wikipedia.org/wiki/Legendre_polynomials, Jul., 2016.

[113] *Hermite polynomials*. Available: https://en.wikipedia.org/wiki/Hermite_polynomials, Jul., 2016.

[114] C.-C. Chang and C.-J. Lin, "LIBSVM: A Library for Support Vector Machines," *ACM Transactions on Intelligent Systems and Technology,* vol. 2, pp. 27:1-27:27, 2011.

[115] S. Canu, Y. Grandvalet, V. Guigue, et al., "SVM and Kernel Methods Matlab Toolbox," Perception Systèmes et Information, INSA de Rouen, 2005.

[116] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A practical guide to support vector classification," Technical Report, Department of Computer Science, National Taiwan University, May,2016.

[117] A. Ben-Hur and J. Weston, "A user's guide to support vector machines," *Methods in Molecular Biology,* vol. 609, pp. 223-239, 2010.

[118] C.-W. Hsu and C.-J. Lin, "A Comparison of Methods for Multiclass Support Vector Machines," *IEEE Transactions On Neural Networks,* vol. 13, pp. 415-425, 2002.

[119] K. Bache and M. Lichman, "UCI Machine Learning Repository," University of California Irvine, School of Information and Computer Science. Available online [http://archive.ics.uci.edu/ml] , 2013.

[120] V. G. Sigillito, S. P. Wing, L. V. Hutton, et al., "Classification of radar returns from the ionosphere using neural networks," *Johns Hopkins APL Technical Digest,* vol. 10, No. 3, pp. 262-266, 1989.

[121] O. Gunay, B. U. Toreyin, K. Kose, et al., "Entropy-functional-based online adaptive decision fusion framework with application to wildfire detection in video," *IEEE Transactions on Image Processing,* vol. 21, No. 5, pp. 2853-2865, 2012.

[122] J. Jia and H.-C. Chua, "Solving Two-Spiral Problem Through Input Data Representation," *Proceesings of the IEEE International Conference on Neural Networks,* pp. 132-135, 1995.

[123] S. Singh, "2D Spiral Pattern Recognition with Possibilistic Measures," *Pattern Recognition Letters,* vol. 19, pp. 141-147, 1998.

[124] S. Singh, "Quantifying Structural Time Varying Changes in Helical Data," *Neural Computing & Applications,* vol. 10, pp. 148-154, 2001.

[125] J. Zhang, N. Ji, J. Liu, et al., "Enhancing performance of the backpropagation algorithm via sparse response regularization," *Neurocomputing,* vol. 153, pp. 20-40, 2015.

[126] B. Margolis. *The Two Spirals Problem Solved*. Available: http://www.benmargolis.com/compsci/ai/two_spirals_problem.htm , 2014

[127] V. Totik, "Orthogonal Polynomials," *Surveys in Approximation Theory,* vol. 1, pp. 70-125, 2005.

[128] *Orthogonal Polynomials*. Available: http://en.wikipedia.org/wiki/Orthogonal_polynomials, June, 2013.

[129] M. Spock, *Front-End Vision and Multi-Scale Image Analysis - Gaussian Derivatives* vol. 27, pp. 53-69: Springer Netherlands, 2003.

[130] Weisstein and E. W., " Hermite Polynomial," From MathWorld--A Wolfram Web Resource. http://mathworld.wolfram.com/HermitePolynomial.html, July, 2016.

[131] *Chebyshev polynomials*. Available: http://en.wikipedia.org/wiki/Chebyshev_polynomials, Oct., 2016.

[132] E. W. Weisstein. *"Chebyshev Polynomial of the First Kind", From MathWorld--A Wolfram Web Resource*. Available: http://mathworld.wolfram.com/ChebyshevPolynomialoftheFirstKind.html, Aug, 2013.

Appendix A

Further Discussions and Illustrative Examples

Appendix A1 Evaluating the feature space using the kernel trick

Example A.1 Consider the two-dimensional binary input space $X \subseteq \Re^2$, shown in Figure A.1, transformed into a three-dimensional feature space $H \subseteq \Re^3$ using the feature map [9]

$$\Phi : \Re^2 \mapsto \Re^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) = \left(x_1^2, x_2^2, \sqrt{2}x_1x_2\right)$$

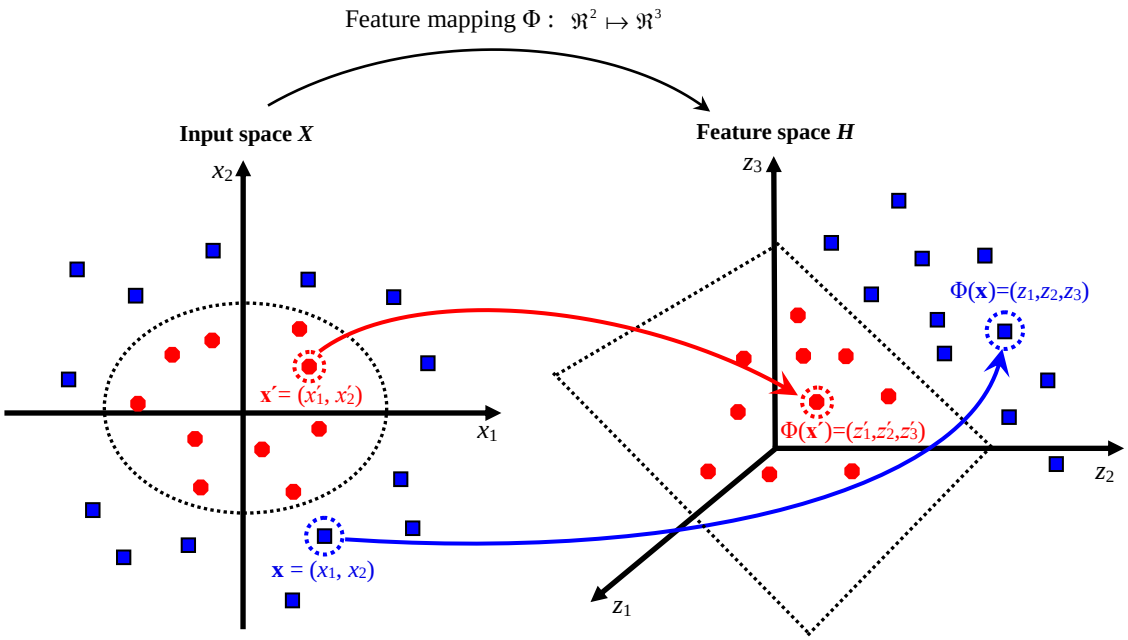


Figure A.1 An illustrative toy example of the feature mapping from 2D to 3D to elaborate the idea of the kernel trick.

Transforming a vector under this mapping will involve the reformulation of its 2D x-coordinates to the 3D z-coordinates. To clarify this transformation, consider the two annotated vectors $\mathbf{x}$ and $\mathbf{x}'$, one from each class, shown in Figure A.1. Their transformation under the above mapping will be as follows:

$$\mathbf{x} = (x_1, x_2) \quad \mapsto \quad \Phi(\mathbf{x}) = (z_1, z_2, z_3) = (x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

$$\mathbf{x}' = (x'_1, x'_2) \quad \mapsto \quad \Phi(\mathbf{x}') = (z'_1, z'_2, z'_3) = (x_1'^2, x_2'^2, \sqrt{2}x'_1x'_2) \qquad \text{(A.7.1)}$$

The feature map transforms the data from a two-dimensional to a three-dimensional space in a way that linear relations in the feature space correspond to quadratic relations in the

input space. The above composition of the feature map can establish an inner product in the feature space as follows

$$\begin{aligned}\langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle &= \langle \Phi(x_1, x_2), \Phi(x_1', x_2') \rangle \\ &= \langle (z_1, z_2, z_3), (z_1', z_2', z_3') \rangle \\ &= \langle (x_1^2, x_2^2, \sqrt{2}x_1x_2), (x_1'^2, x_2'^2, \sqrt{2}x_1'x_2') \rangle \\ &= x_1^2 x_1'^2 + 2x_1x_2x_1'x_2' + x_2^2 x_2'^2 \\ &= (x_1x_1' + x_2x_2')^2 \\ &= (<\mathbf{x}, \mathbf{x}'>)^2 \equiv k(\mathbf{x}, \mathbf{x}') . \end{aligned} \qquad \text{(A.7.2)}$$

Hence the function

$$k(\mathbf{x}, \mathbf{x}') = (<\mathbf{x}, \mathbf{x}'>)^2$$

is a kernel function with H as its corresponding feature space; i.e., the desired kernel k is simply the square of the dot product in the input space. As shown in the derivation above, a suitably chosen kernel function can compute the inner product between the projections of two points into the feature space, directly from the input space, without explicitly evaluating their coordinates; i.e., the construction and computation of the feature space is achieved implicitly without even knowing what it might look like.

Note that the same kernel can also compute the inner product corresponding to the four-dimensional feature map

$$\begin{aligned} \Phi: \quad \Re^2 &\mapsto \Re^4 \\ \mathbf{x} = (x_1, x_2) &\mapsto \Phi(\mathbf{x}) = (z_1, z_2, z_3, z_4) = (x_1^2, x_2^2, x_1x_2, x_2x_1) \subseteq H , \end{aligned}$$

which shows that the feature space is not always uniquely determined by the kernel function. Using this mapping, it can be shown that the above example can be readily generalised to higher dimensional input spaces. Consider, for example, an n-dimensional input space $X \subseteq \Re^n$; the function

$$k(\mathbf{x}, \mathbf{x}') = (<\mathbf{x}, \mathbf{x}'>)^2$$

is a kernel function corresponding to the feature map

$$\Phi: \mathbf{x} \mapsto \Phi(\mathbf{x}) = (x_i x_j)^n \in H = \Re^{n^2} ,$$

where $(i, j) \in \{1,2,3,\cdots,n\} \times \{1,2,3,\cdots,n\}$

since

$$\begin{aligned} \langle \Phi(\mathbf{x}), \Phi(\mathbf{x}') \rangle &= \langle (x_i x_j)_{i,j=1}^{n}, (x_i' x_j')_{i,j=1}^{n} \rangle \\ &= \sum_{i,j=1}^{n} x_i x_j \, x_i' x_j' \\ &= \sum_{i=1}^{n} x_i x_i' \sum_{j=1}^{n} x_j x_j' \\ &= (< \mathbf{x}, \mathbf{x}' >)^2 \end{aligned} \qquad \text{(A.7.3)}$$

This example shows how the construction of a simple mapping can convert a linear polynomial kernel into an inner product in a higher dimensional feature space, and enables the kernel to implicitly define a vastly expanded feature space. Section 2.4.3 has shown that it is also possible to construct more complex kernels from simpler ones using a variety of operations, making the range of constructed valid kernels very large, some of which correspond to even an infinite-dimensional feature spaces at the cost of only a few extra operations in the kernel evaluation [1].

Appendix A2 Numerical example to evaluate the entries of the kernel matrix

Example A.2 To elaborate the process of constructing the kernel matrix by evaluating the kernel function on the pairs of the input data, consider a simple 2-dimensional dataset $X \in \Re^2$, where

$\mathbf{x} = [x_1 \quad x_2] \in \Re^2$, and

$\mathbf{z} = [z_1 \quad z_2] \in \Re^2$.

And let

$$k(\mathbf{x}, \mathbf{z}) = (\langle \mathbf{x}, \mathbf{z} \rangle)^2 = (x_1 z_1 + x_2 z_2)^2 \qquad \text{(A.7.4)}$$

be the kernel function. For simplicity, assume that the dataset consists of only the following two examples $\mathbf{x}_1$ and $\mathbf{x}_2$.

$\mathbf{x_1} = [0 \quad 0]$, and

$\mathbf{x_2} = [1 \quad 1]$.

The kernel matrix can therefore be constructed by evaluating the kernel function on each of the examples of the above dataset as follows:

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) \end{bmatrix}$$

$$= \begin{bmatrix} (0\times 0 + 0\times 0)^2 & (0\times 1 + 0\times 1)^2 \\ (1\times 0 + 1\times 0)^2 & (1\times 1 + 1\times 1)^2 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 \\ 0 & 4 \end{bmatrix}. \qquad \text{(A.7.5)}$$

Appendix A3 Evaluation of the positive semi-definite kernel matrix

Example A.3 Show that the following matrices are positive semi-definite

(a) $\mathbf{K} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}$

(b) $\mathbf{K} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$

Consider first the inequality method $\mathbf{a}^T\mathbf{K}\mathbf{a} \geq 0$, to analyse the above matrices, where the vector **a** can be taken as

$\mathbf{a} = [a_1 \quad a_2]$.

Evaluating the approach on the above examples therefore yields

(a) $\mathbf{a}^T\mathbf{K}\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} [a_1 \quad a_2]$

$$= \begin{bmatrix} 2a_1 \\ 0 \end{bmatrix} [a_1 \quad a_2]$$

$$= 2a_1^2 \geq 0 \quad , \quad \forall a_1, a_2 \in \Re$$

The matrix therefore is positive semi-definite.

(b) $\mathbf{a}^{\mathrm{T}}\mathbf{K}\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\begin{bmatrix} a_1 & a_2 \end{bmatrix}$

$= \begin{bmatrix} a_1 - a_2 \\ -a_1 + a_2 \end{bmatrix}\begin{bmatrix} a_1 & a_2 \end{bmatrix}$

$= a_1^2 - a_1 a_2 - a_1 a_2 + a_2^2 \ ,$

$= (a_1 - a_2)^2 \geq 0, \quad \forall a_1, a_2 \in \Re$

The matrix therefore is positive semi-definite.

Considering now the non-negative eigenvalues approach, the example matrices can also be analysed as follows.

(a) The matrix $\begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}$ is a diagonal matrix (in which $\mathbf{K}_{ij} = 0$ whenever $i \neq j$). As such, the eigenvalues of a diagonal matrix are the elements of its diagonals; i.e., $\lambda = \mathbf{K}_{ii}$. As these elements in this example are non-negative, this matrix is positive semi-definite.

(b) To calculate the eigenvalues of the matrix $\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$ we need to solve the determinant equation $\det(\mathbf{K} - \lambda \mathbf{I}) = 0$ as follows

$$\begin{vmatrix} 1-\lambda & -1 \\ -1 & 1-\lambda \end{vmatrix} = (1-\lambda)^2 - 1 = \lambda^2 - 2\lambda + 1 - 1 = \lambda(\lambda - 2) = 0$$

Therefore $\lambda = 0 \geq 0$ or $\lambda = 2 \geq 0$

As both eigenvalues are non-negative, therefore the matrix is positive semi-definite.

Appendix A4 Evaluation of the positive semi-definite kernel function

Example A.4 Show that

$k(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle$

is a finitely positive semi-definite function.

Referring back to Proposition 2.2, it is important to show that the candidate function satisfies the PSD property by evaluating it on ANY subset of data. Therefore, given the generic sub-set $S = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l\}$, it can be shown that the $k(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle$ is a PSD function by showing that its corresponding kernel matrix is also PSD as follows:

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}_1) & k(\mathbf{x}_1,\mathbf{x}_2) & \cdots & k(\mathbf{x}_1,\mathbf{x}_l) \\ k(\mathbf{x}_2,\mathbf{x}_1) & k(\mathbf{x}_2,\mathbf{x}_2) & \cdots & k(\mathbf{x}_2,\mathbf{x}_l) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_l,\mathbf{x}_1) & k(\mathbf{x}_l,\mathbf{x}_2) & \cdots & k(\mathbf{x}_l,\mathbf{x}_l) \end{bmatrix}$$

$$= \begin{bmatrix} \langle \mathbf{x}_1,\mathbf{x}_1 \rangle & \langle \mathbf{x}_1,\mathbf{x}_2 \rangle & \cdots & \langle \mathbf{x}_1,\mathbf{x}_l \rangle \\ \langle \mathbf{x}_2,\mathbf{x}_1 \rangle & \langle \mathbf{x}_2,\mathbf{x}_2 \rangle & \cdots & \langle \mathbf{x}_2,\mathbf{x}_l \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \mathbf{x}_l,\mathbf{x}_1 \rangle & \langle \mathbf{x}_l,\mathbf{x}_2 \rangle & \cdots & \langle \mathbf{x}_l,\mathbf{x}_l \rangle \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{x}_1^T\mathbf{x}_1 & \cdots & \mathbf{x}_1^T\mathbf{x}_l \\ \vdots & \ddots & \vdots \\ \mathbf{x}_l^T\mathbf{x}_1 & \cdots & \mathbf{x}_l^T\mathbf{x}_l \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_l^T \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_l \end{bmatrix} \qquad \text{(A.7.6)}$$

The kernel matrix **K** can now be checked for the PSD property using any vector **y** as follows:

$$\mathbf{y}^T\mathbf{K}\mathbf{y} = \mathbf{y}^T \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_l^T \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_l \end{bmatrix} \mathbf{y}$$

$$= \left(\begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_l \end{bmatrix} \mathbf{y} \right)^T \left(\begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_l \end{bmatrix} \mathbf{y} \right)$$

$$= \left\| \begin{bmatrix} \mathbf{x}_1 & \cdots & \mathbf{x}_l \end{bmatrix} \mathbf{y} \right\|^2 \geq 0 \ , \ \forall \mathbf{y} \qquad \text{(A.7.7)}$$

It can therefore be concluded that, if a function $k : \Re^d \times \Re^d \to \Re$ and its corresponding matrix

$$\mathbf{K} = \begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}_1) & \cdots & k(\mathbf{x}_1,\mathbf{x}_l) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_l,\mathbf{x}_1) & \cdots & k(\mathbf{x}_l,\mathbf{x}_l) \end{bmatrix}$$

is positive semi-definite matrix evaluated on any set $\{\mathbf{x}_1,\ldots,\mathbf{x}_l\} \subseteq \Re^d$, then k is called a positive semi-definite kernel function.

Appendix A5 Orthogonal polynomials

Orthogonal polynomials are various families of polynomials which are useful in solving differential equations arising in physics and engineering. In general, they have many important applications in such areas as mathematical physics, interpolation theory, the theory of random matrices, computer approximations, and many others [102]. The term orthogonality is the higher-dimensional analogue of perpendicularity. These polynomials are defined such that any two different polynomials in the same sequence are orthogonal to each other under some inner product and with respect to a certain weight function.

The most common orthogonal polynomials are the "*classical orthogonal polynomials*" consisting of the *Hermite polynomials*, the *Laguerre polynomials*, the *Jacobi polynomials* which include the *Gegenbauer polynomials*, the *Chebyshev polynomials*, and the *Legendre polynomials* [127]. Figure A.2 shows a schematic diagram of the most common orthogonal polynomials in mathematics.

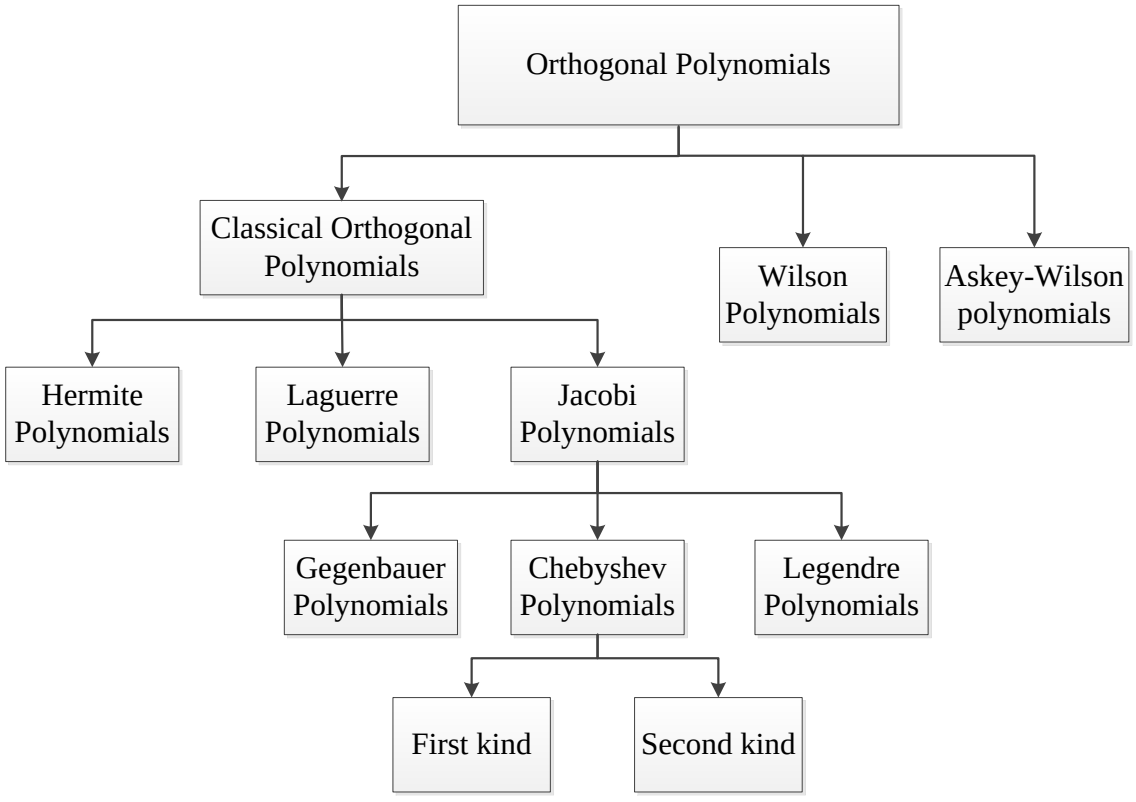


Figure A.2 A taxonomy of the most commonly used orthogonal polynomial functions.

The general setting of orthogonal polynomials has emerged from the definition of a finite integral for all polynomials *f*. In this case, an inner product on any two pairs of polynomials *f* and *g* can be defined as [128]:

$$\langle f, g \rangle = \int_{x_1}^{x_2} f(x) g(x) W(x) dx \qquad \text{(A.7.8)}$$

where

$$W : [x_1, x_2] \rightarrow \Re$$

The operation defined by (A.7.8) is a non-negative function defined on some interval [x_1, x_2] in the real line (where this interval could be x_1= - ∞ and x_2= +∞). This function W is commonly known as the *weight function* within the context of orthogonal polynomials [128]. This operation is also positive semi-definite inner product on the vector space of all polynomials, and it introduces the notion of orthogonality on the two polynomials if their inner product is zero.

Therefore, any orthogonal polynomial sequence $[P_n]_{n=0}^{\infty}$ of degree n can be defined as:

$$\langle P_m, P_n \rangle = 0 \quad \text{for } m \neq n \qquad \text{(A.7.9)}$$

Hermite polynomials

Hermite polynomials are amongst the families of polynomials of the Classical orthogonal polynomials, named after Charles Hermite, a French mathematician [129]. They have got a number of applications in such areas as [113]: probability, combinatorics, numerical analysis as Gaussian quadrature, physics, and in systems theory within non-linear operations on Gaussian noise. They are usually denoted by $H_n(x)$ for n = 0, 1, 2, 3, . . . and some of their main characteristics are [129]:

Some characteristics of the Hermite orthogonal polynomials are:

1- They are the only orthogonal polynomials within the set that are defined within the whole range of the real line (- ∞, + ∞). This property is very important when we come to implementation within a machine learning algorithm such as SVM, for example, because it might avoid (or at least reduce) the need for the data normalization, which is essential for the classification step, and hence facilitating the calculation of the quadratic programming problem and reducing the complexity of the overall algorithm.

2- They are orthogonal to each other with respect to the basic Gaussian weighting function e^{-x^2} as:

$$\int_{-\infty}^{\infty} H_n(x)\, H_m(x)\, e^{-x^2} dx = 2^{n+m}\, n!\sqrt{\pi}\, \delta_{nm} \qquad \text{(A.7.10)}$$

where δ_{nm} is the Kronecker delta, or delta tensor, which is a function of the two variables *n* and *m*. The function is 1 if the two variables are equal, and 0 otherwise:

3- The Hermite polynomials also satisfy the symmetry condition [130]:

$$H_n(-x) = (-1)^n H_n(x) \qquad \text{(A.7.11)}$$

4- They can be defined by the contour integral [102, 130]:

$$H_n(z) = \frac{n!}{2\pi i} \oint e^{-t^2+2tz} t^{-n-1} dt \qquad \text{(A.7.12)}$$

However, the domain of definition that is used in this thesis is the recurrence relation given by:

$$H_0(x) = 1,$$

$$H_1(x) = 2x,$$

$$H_n(x) = 2x\, H_{n-1}(x) - 2(n-1)\, H_{n-2}(x). \qquad \text{(A.7.13)}$$

Chebyshev polynomials

The Chebyshev polynomials are another family of orthogonal polynomial functions. They are commonly being used in many applications, such as filtering, numerical analysis, and more generally in mathematical applications, such as approximation theory and differential equations [131]. As Figure A.2 indicates, there are two types of Chebyshev polynomials: "Chebyshev polynomials of the first kind", usually denoted by T_n and "Chebyshev polynomials of the second kind", usually denoted by U_n. Both kinds are composed of a polynomial sequence of degree *n*. Mathematically, they are quite closely related to each other and differ only in the coefficients assigned to the monomial terms.

For the limited space and to avoid unnecessary redundancy, this thesis considers only the Chebyshev polynomials of the first kind for verification and validation purposes, but it also shows how easily the presented approaches could be extended to be also applied on the Chebyshev polynomials of the second kind.

The Chebyshev polynomials of the first kind $T_n(z)$ can be defined by the contour integral [102, 132]:

$$T_n(z) = \frac{1}{4\pi i}\oint \frac{\left(1-t^2\right)t^{-n-1}}{\left(1-2tz+t^2\right)}dt\,, \tag{A.7.14}$$

where the contour encloses the origin and is traversed in a counterclockwise direction.

However, for consistency with previous literature, such as [31] and [17], and ease of implementation, the following recursive approach is used as the main domain of definition in this thesis. The orthogonal set of the Chebyshev polynomials of the first kind is denoted by $T_n(x)$ where n is the order of the polynomial given by n = 0, 1, 2, 3, …, and for the x values bounded between [-1, 1]. The first two polynomial sequences of the Chebyshev polynomials of the first kind $T_n(x)$ of order n is defined as:

$T_0(x) = 1$, and

$$T_1(x) = x. \tag{A.7.15}$$

Any other higher order polynomial can be formulated using the following recursive formula:

$$T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x)\,. \tag{A.7.16}$$

The first few Chebyshev polynomials of the first kind can therefore be written as:

$$T_0(x) = 1 \tag{A.7.17}$$
$$T_1(x) = x \tag{A.7.18}$$
$$T_2(x) = 2x^2 - 1 \tag{A.7.19}$$
$$T_3(x) = 4x^3 - 3x \tag{A.7.20}$$
$$T_4(x) = 8x^4 - 8x^2 + 1 \tag{A.7.21}$$
$$T_5(x) = 16x^5 - 20x^3 + 5x \tag{A.7.22}$$
$$T_6(x) = 32x^6 - 48x^4 + 18x^2 - 1 \tag{A.7.23}$$
$$T_7(x) = 64x^7 - 112x^5 + 56x^3 - 7x \tag{A.7.24}$$

The Chebyshev polynomials of the first kind are orthogonal with respect to the weighting function

$$w(x) = \frac{1}{\sqrt{1-x^2}}\,, \tag{A.7.25}$$

such that any two different given Chebyshev polynomials, if integrated between the interval [-1, 1], we have:

$$\int_{-1}^{1} T_i(x)T_j(x)w(x)dx = 0 \tag{A.7.26}$$

where $i \neq j$.

Legendre polynomials

The Legendre polynomials are another family of orthogonal polynomial functions investigated in this thesis for validation and comparison purposes. They are named after Adrien Marie Legendre (1752-1833), and in mathematics, Legendre functions are most commonly used as solutions to Legendre differential equations [112]. They are also frequently encountered in physics and other technical fields, in particular when solving Laplace's equation in spherical coordinates [112].

Similar to the Chebyshev polynomials, the Legendre polynomials, denoted by $L_n(x)$, can be defined by the contour integral [102]:

$$L_n(x) = \frac{1}{2\pi i}\oint \left(1 - 2xt + t^2\right)^{-1/2} t^{-n-1} dt\,, \tag{A.7.27}$$

where the contour encloses the origin and is traversed in a counterclockwise direction.

However, similar to the other polynomials implemented in this thesis, the following recursive approach is used as the main domain of definition in this thesis, for consistency:

$L_0(x) = 1$, and

$$L_1(x) = x. \tag{A.7.28}$$

Any other higher order polynomial can therefore be formulated using the following recursive formula:

$$L_n(x) = \frac{1}{n}\left[(2n-1)xL_{n-1}(x) - (n-1)L_{n-2}(x)\right]. \tag{A.7.29}$$

The first few Legendre polynomials can therefore be written as:

$$L_0(x) = 1 \tag{A.7.30}$$

$$L_1(x) = x \tag{A.7.31}$$

$$L_2(x) = \frac{1}{2}(3x^2 - 1) \tag{A.7.32}$$

$$L_3(x) = \frac{1}{2}(5x^3 - 3x) \tag{A.7.33}$$

$$L_4(x) = \frac{1}{8}(35x^4 - 30x^2 + 3) \tag{A.7.34}$$

$$L_5(x) = \frac{1}{8}(63x^5 - 70x^3 + 15x) \tag{A.7.35}$$

One of the salient characteristics of the Legendre polynomials is that they are orthogonal with respect to a unity weighting function; i.e., the weighting function is simply:

$$w(x) = 1. \tag{A.7.36}$$

This makes the implementation of such a polynomial as a kernel a lot simpler, and greatly reduces the complexity of the algorithms. Luckily, their domain of orthogonality definition is the same as that for Chebyshev polynomials; i.e., if integrated between the interval [-1, 1], we have:

$$\int_{-1}^{1} L_i(x) L_j(x) w(x) dx = 0 \tag{A.7.37}$$

where $i \neq j$.

Appendix B

Statistical Significance Tests

In this appendix, the statistical significance tests of the experimental results presented in this thesis are conducted using the SPSS statistical analysis software (version 24.0) provided by Staffordshire University[16]. Due to the pairwise nature of the comparative experimental results, the 'independent samples t-test'[17] is the test chosen to determine the statistical significance of the difference between each two groups of the pattern classification accuracy scores obtained from each of the experiments under test. All the statistical significance t-tests are performed with a confidence level of 95%.

The results of each of the conducted t-tests in the following sections are presented in two interlinked tables titled: (a) Group Statistics and (b) Independent Samples Test. The *Group Statistics* table provide information about the two groups of the experimental results being tested, their number of samples (N), their mean (M), their standard deviation (Std. Deviation), and their standard error mean (Std. Error Mean). The *Independent Samples Test* table, on the other hand, provides the results of two t-tests conducted with two different methods based on whether or not the compared two groups of data have equal variances. This is determined by the F-value (and its corresponding significance level (Sig.)) of the Levene's Test column. If the 'Sig.' value of the Levene's Test is greater than 0.05 (as our confidence level is chosen to be 95%), then the two groups of data under comparison are assumed to be of equal variances. Otherwise, this assumption is not applicable, and the bottom row of the table applies.

In both cases, if the probability of error (shown in the Sig. (2-tailed) column) is less than or equal 0.05 (for a confidence level of 95%), then the two groups of experimental results under comparison are considered to be statistically significant from each other. Otherwise, there is no statistically significant difference between them. As shown in the statistical significance t-tests presented in the following sections, the majority of the experimental results obtained in this thesis were found to be statistically significant.

[16] https://www.staffs.ac.uk/software/
[17] E. Huizingh, *Applied Statistics with SPSS*: SAGE Publications Ltd, 2007.

B.1 Statistical significance tests on the experimental results of Stage 1-Stage 2 kernels – Chapter 4 – Section 4.4.4

B.1.1 Two Spirals dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Cheb kernel - vectorial	15	52.12204	13.114225	3.386078
	Stage 2 Cheb kernel - vectorial	15	70.74142	20.359412	5.256778

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	6.148	0.019	-2.97	28	0.006	-18.619	6.25293	-31.427	-5.8108
	Equal variances not assumed			-2.97	23.9	0.007	-18.619	6.25293	-31.527	-5.7114

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Herm kernel - vectorial	15	39.62014	19.123972	4.937788
	Stage 2 Herm kernel - vectorial	15	53.37452	5.440630	1.404765

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	9.669	0.004	-2.67	28	0.012	-13.754	5.13372	-24.270	-3.2384
	Equal variances not assumed			-2.67	16.2	0.016	-13.754	5.13372	-24.623	-2.8850

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Leg kernel - vectorial	15	50.97178	12.378766	3.196184
	Stage 2 Leg kernel - vectorial	15	70.77651	20.713399	5.348177

(b) Independent Samples Test									
	Levene's Test for Equality of Variances		t-test for Equality of Means						
								95% Confidence Interval of the Difference	
	F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Differe		

								nce	Lower	Upper
Accuracy %	Equal variances assumed	7.629	0.010	-3.17	28	0.004	-19.804	6.23045	-32.567	-7.0422
	Equal variances not assumed			-3.17	22.8	0.004	-19.804	6.23045	-32.697	-6.9119

B.1.2 Breast cancer dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Cheb kernel - vectorial	15	49.33902	34.039267	8.788901
	Stage 2 Cheb kernel – vectorial	15	75.45132	17.787069	4.592602

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
								Std. Error Difference	95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference		Lower	Upper
Accuracy %	Equal variances assumed	5.653	0.024	-2.63	28	0.014	-26.112	9.91649	-46.425	-5.7992
	Equal variances not assumed			-2.63	21.1	0.015	-26.112	9.91649	-46.727	-5.4966

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Herm kernel - vectorial	15	45.97015	36.555444	9.438575
	Stage 2 Herm kernel - vectorial	15	79.94314	16.947647	4.375864

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
								Std. Error Difference	95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference		Lower	Upper
Accuracy %	Equal variances assumed	16.942	0.000	-3.26	28	0.003	-33.972	10.4035	-55.283	-12.662
	Equal variances not assumed			-3.26	19.7	0.004	-33.972	10.4035	-55.691	-12.254

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Leg kernel - vectorial	15	56.88699	31.186884	8.052419
	Stage 2 Leg kernel - vectorial	15	79.50249	18.057405	4.662402
(b) Independent Samples Test					

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	1.792	0.192	-2.43	28	0.022	-22.615	9.30480	-41.675	-3.5554
	Equal variances not assumed			-2.43	22.4	0.024	-22.615	9.30480	-41.890	-3.3403

B.1.3 Iris

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Cheb kernel - vectorial	10	68.85714	4.579579	1.448190
	Stage 2 Cheb kernel - vectorial	10	93.04762	1.798630	0.568777

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	14.382	0.001	-15.5	18	7.0626 E-12	-24.190	1.55588	-27.459	-20.921
	Equal variances not assumed			-15.5	11.7	3.5307 E-09	-24.190	1.55588	-27.589	-20.791

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Herm kernel - vectorial	10	58.66667	19.775509	6.253565
	Stage 2 Herm kernel - vectorial	10	80.85714	22.156137	7.006386

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	0.030	.864	-2.36	18	0.030	-22.190	9.39130	-41.920	-2.4600
	Equal variances not assumed			-2.36	17.7	0.030	-22.190	9.39130	-41.938	-2.4419

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean

Accuracy %	Stage 1 Leg kernel - vectorial	10	68.95238	5.122831	1.619981
	Stage 2 Leg kernel - vectorial	10	92.76190	2.298026	0.726700

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	12.219	0.003	-13.4	18	8.2740 E-11	-23.809	1.77550	-27.539	-20.079
	Equal variances not assumed			-13.4	12.4	8.8191 E-09	-23.809	1.77550	-27.661	-19.957

B.1.4 Image Segmentation

(a) Group Statistics

	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Cheb kernel - vectorial	10	33.68095	31.196095	9.865071
	Stage 2 Cheb kernel - vectorial	10	46.30953	39.165608	12.385253

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	3.972	0.062	0.79	18	0.436	-12.628	15.8339	-45.894	20.6373
	Equal variances not assumed			-0.79	17.1	0.436	-12.628	15.8339	-46.014	20.7570

(a) Group Statistics

	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Herm kernel - vectorial	10	33.36190	31.446662	9.944308
	Stage 2 Herm kernel - vectorial	10	46.89048	38.968045	12.322778

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	3.671	0.071	-0.85	18	0.404	-13.528	15.8347	-46.796	19.7390

	Equal variances not assumed			-0.85	17.2	0.405	-13.528	15.8347	-46.902	19.8457

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Leg kernel - vectorial	10	41.17619	35.142622	11.113073
	Stage 2 Leg kernel - vectorial	10	53.97619	39.814133	12.590334

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	2.562	0.127	-0.76	18	0.456	-12.800	16.7933	-48.081	22.4815
	Equal variances not assumed			-0.76	17.7	0.456	-12.800	16.7933	-48.120	22.5205

B.1.5 Ionosphere

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Cheb kernel - vectorial	10	55.29801	40.485014	12.802486
	Stage 2 Cheb kernel - vectorial	10	82.18543	23.086014	7.300438

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	12.757	0.002	-1.82	18	0.085	-26.887	14.7377	-57.850	4.07535
	Equal variances not assumed			-1.82	14.2	0.089	-26.887	14.7377	-58.435	4.66106

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Herm kernel - vectorial	10	48.07947	35.164116	11.119870
	Stage 2 Herm kernel - vectorial	10	74.83444	30.402731	9.614188

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means

		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	1.157	0.296	-1.82	18	0.085	-26.754	14.6998	-57.638	4.12816
	Equal variances not assumed			-1.82	17.6	0.086	-26.754	14.6998	-57.684	4.17444

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Leg kernel - vectorial	10	52.78146	37.201611	11.764182
	Stage 2 Leg kernel - vectorial	10	82.31788	23.165245	7.325494

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	9.802	0.006	-2.13	18	0.047	-29.536	13.8585	-58.652	-0.4207
	Equal variances not assumed			-2.13	15.0	0.050	-29.536	13.8585	-59.063	-0.0091

B.1.6 Thyroid

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Cheb kernel - vectorial	10	65.93932	43.940548	13.895221
	Stage 2 Cheb kernel - vectorial	10	67.43874	44.947365	14.213605

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	0.012	0.913	-0.07	18	0.941	-1.4994	19.8772	-43.259	40.2610
	Equal variances not assumed			-0.07	17.9	0.941	-1.4994	19.8772	-43.261	40.2626

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Herm kernel - vectorial	10	84.03442	28.740256	9.088467
	Stage 2 Herm kernel - vectorial	10	94.49533	1.250036	.395296

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	4.411	0.050	-1.15	18	0.265	-10.460	9.09705	-29.573	8.65130
	Equal variances not assumed			-1.15	9.03	0.280	-10.460	9.09705	-31.028	10.1062

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Leg kernel - vectorial	10	66.02100	43.996156	13.912806
	Stage 2 Leg kernel - vectorial	10	67.37165	44.901133	14.198985

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	0.010	0.922	-0.06	18	0.947	-1.3506	19.8790	-43.115	40.4137
	Equal variances not assumed			-0.06	17.9	0.947	-1.3506	19.8790	-43.116	40.4149

B.2 Statistical significance tests on the experimental results of Stage 2-Stage 3 kernels – Chapter 4 – Section 4.4.5

B.2.1 Two Spirals dataset

(a) Group Statistics					
	OPK	**N**	**Mean**	**Std. Deviation**	**Std. Error Mean**
Accuracy %	Stage 2 Chebyshev kernel - vectorial	15	70.74142	20.359412	5.256778
	Stage 3 Modified Chebyshev kernel - vectorial	15	87.73456	16.010239	4.133826

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		**t-test for Equality of Means**						
									95% Confidence Interval of the Difference	
		F	**Sig.**	**t**	**df**	**Sig. (2-tailed)**	**Mean Difference**	**Std. Error Difference**	**Lower**	**Upper**
Accuracy %	Equal variances assumed	2.552	0.121	-2.54	28	0.017	-16.993	6.6874	-30.6917	-3.2944
	Equal variances not assumed			-2.54	26.52	0.017	-16.993	6.6874	-30.7261	-3.2600

(a) Group Statistics					
	OPK	**N**	**Mean**	**Std. Deviation**	**Std. Error Mean**
Accuracy %	Stage 2 Chebyshev kernel - vectorial	15	70.74142	20.359412	5.256778
	Stage 3 Generalized Chebyshev kernel - vectorial	15	72.43326	17.625977	4.551008

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	**Sig.**	**t**	**df**	**Sig. (2-tailed)**	**Mean Difference**	**Std. Error Difference**	**95% Confidence Interval of the Difference**	
									Lower	**Upper**
Accuracy %	Equal variances assumed	0.543	0.467	-0.24	28	0.810	-1.6918	6.9530	-15.9345	12.5509
	Equal variances not assumed			-0.24	27.43	0.810	-1.6918	6.9530	-15.9477	12.5640

(a) Group Statistics					
	OPK	**N**	**Mean**	**Std. Deviation**	**Std. Error Mean**
Accuracy %	Stage 2 Hermite kernel - vectorial	15	53.37452	5.440630	1.404765
	Stage 3 Composite Hermite kernel - vectorial	15	87.20214	4.456921	1.150772

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	**Sig.**	**t**	**df**	**Sig. (2-tailed)**	**Mean Difference**	**Std. Error Difference**	**95% Confidence Interval of the Difference**	
									Lower	**Upper**
Accuracy %	Equal variances assumed	0.056	0.815	-18.62	28	2.59310 E-17	-33.82761	1.81594	-37.54739	-30.10788
	Equal variances not assumed			-18.62	26.956	6.36182 E-17	-33.8276	1.8159	-37.55390	-30.1013

(a) Group Statistics					
	OPK	**N**	**Mean**	**Std. Deviation**	**Std. Error Mean**
Accuracy %	Stage 2 Legendre kernel - vectorial	15	70.77651	20.713399	5.348177
	Stage 3 Modified Legendre kernel - vectorial	15	96.70328	3.944609	1.018494

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	**Sig.**	**t**	**df**	**Sig. (2-tailed)**	**Mean Difference**	**Std. Error Difference**	**95% Confidence Interval of the Difference**	
									Lower	**Upper**
Accuracy %	Equal variances assumed	35.7	0.000	-4.76	28	0.0000	-25.926	5.4442	-37.078	-14.774
	Equal variances not assumed			-4.76	15.014	0.0002	-25.926	5.4442	-37.530	-14.323

B.2.2 Breast cancer dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev kernel - vectorial	15	75.45132	17.787069	4.592602
	Stage 3 Generalized Chebyshev kernel -vectorial	15	95.94883	1.206151	0.311427

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	153.	0.000	-4.45	28	0.000	-20.497	4.6031	-29.926	-11.068
	Equal variances not assumed			-4.45	14.129	0.001	-20.497	4.6031	-30.361	-10.633

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev kernel - vectorial	15	75.45132	17.787069	4.592602
	Stage 3 Modified Chebyshev kernel -vectorial	15	96.41791	1.045178	0.269864

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	157.	0.000	-4.55	28	0.000	-20.966	4.6005	-30.390	-11.542
	Equal variances not assumed			-4.55	14.097	0.000	-20.966	4.6005	-30.827	-11.105

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite kernel - vectorial	15	79.94314	16.947647	4.375864
	Stage 3 Composite Hermite kernel - vectorial	15	89.60910	8.318090	2.147721

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	3.09	0.090	-1.98	28	0.057	-9.6659	4.8745	-19.650	0.31903
	Equal variances not assumed			-1.98	20.375	0.061	-9.6659	4.8745	-19.822	0.49011

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre kernel - vectorial	15	79.50249	18.057405	4.662402
	Stage 3 Modified Legendre kernel - vectorial	15	96.26155	1.113377	0.287473

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	314.	0.000	-3.58	28	0.001	-16.759	4.6712	-26.327	-7.1904
	Equal variances not assumed			-3.58	14.106	0.003	-16.759	4.6712	-26.770	-6.74730

B.2.3 Iris

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev kernel - vectorial	15	76.95238	27.272516	7.041733
	Stage 3 Generalized Chebyshev kernel -vectorial	15	84.38095	16.000486	4.131308

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	7.21	0.012	-0.91	28	0.371	-7.4285	8.1641	-24.1521	9.29497
	Equal variances not assumed			-0.91	22.617	0.372	-7.4285	8.1641	-24.3332	9.47615

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev kernel - vectorial	15	76.95238	27.272516	7.041733
	Stage 3 Modified Chebyshev kernel -vectorial	15	92.44445	4.768250	1.231157

(a) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	32.8	0.000	-2.16	28	0.039	-15.492	7.1485	-30.135	-0.84892
	Equal variances not assumed			-2.16	14.855	0.047	-15.492	7.1485	-30.741	-0.24234

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite kernel - vectorial	15	65.26984	29.497171	7.616137
	Stage 3 Composite Hermite kernel - vectorial	15	94.34921	2.822161	0.728679

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	65.6	0.000	-3.80	28	0.001	-29.079	7.6509	-44.751	-13.407
	Equal variances not assumed			-3.80	14.256	0.002	-29.079	7.6509	-45.461	-12.697

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre kernel - vectorial	15	81.68254	22.579713	5.830057
	Stage 3 Modified Legendre kernel - vectorial	15	93.01587	3.682696	0.950868

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	14.5	0.001	-1.91	28	0.065	-11.333	5.9070	-23.433	0.76679
	Equal variances not assumed			-1.91	14.744	0.075	-11.333	5.9070	-23.943	1.27637

B.2.4 Image Segmentation

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev kernel - vectorial	15	35.83810	34.944860	9.022724
	Stage 3 Generalized Chebyshev kernel -vectorial	15	88.68889	3.168615	0.818133

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	41.0	0.000	-5.83	28	2.8663 E-06	-52.850	9.0597	-71.408	-34.2927
	Equal variances not assumed			-5.83	14.230	0.0000	-52.850	9.0597	-72.252	-33.4490

(a) Group Statistics					
	OPK	**N**	**Mean**	**Std. Deviation**	**Std. Error Mean**
Accuracy %	Stage 2 Chebyshev kernel - vectorial	15	35.83810	34.944860	9.022724
	Stage 3 Modified Chebyshev kernel -vectorial	15	91.49206	1.245226	0.321516

(b) Independent Samples Test											
		Levene's Test for Equality of Variances		t-test for Equality of Means							
		F	**Sig.**	**t**	**df**	**Sig. (2-tailed)**	**Mean Difference**	**Std. Error Difference**	**95% Confidence Interval of the Difference**	
									Lower	**Upper**
Accuracy %	Equal variances assumed	46.6	0.000	-6.16	28	1.1775 E-06	-55.653	9.0284	-74.147	-37.1600
	Equal variances not assumed			-6.16	14.036	0.0000	-55.653	9.0284	-75.013	-36.2944

(a) Group Statistics					
	OPK	**N**	**Mean**	**Std. Deviation**	**Std. Error Mean**
Accuracy %	Stage 2 Hermite kernel - vectorial	15	36.37778	34.839991	8.995647
	Stage 3 Composite Hermite kernel - vectorial	15	89.55556	2.691132	0.694847

(b) Independent Samples Test											
		Levene's Test for Equality of Variances		t-test for Equality of Means							
		F	**Sig.**	**t**	**df**	**Sig. (2-tailed)**	**Mean Difference**	**Std. Error Difference**	**95% Confidence Interval of the Difference**	
									Lower	**Upper**
Accuracy %	Equal variances assumed	42.042	0.000	-5.89	28	2.4353 E-06	-53.177	9.0224	-71.659	-34.6961
	Equal variances not assumed			-5.89	14.167	0.0000	-53.177	9.0224	-72.507	-33.8479

(a) Group Statistics					
	OPK	**N**	**Mean**	**Std. Deviation**	**Std. Error Mean**
Accuracy %	Stage 2 Legendre kernel - vectorial	15	40.95238	37.183037	9.600619
	Stage 3 Modified Legendre kernel - vectorial	15	91.48889	1.217578	0.314377

(b) Independent Samples Test											
		Levene's Test for Equality of Variances		t-test for Equality of Means							
		F	**Sig.**	**t**	**df**	**Sig. (2-tailed)**	**Mean Difference**	**Std. Error Difference**	**95% Confidence Interval of the Difference**	
									Lower	**Upper**
Accuracy %	Equal variances assumed	104.	0.000	-5.26	28	0.0000	-50.536	9.6057	-70.213	-30.8599
	Equal variances not assumed			-5.26	14.030	0.0001	-50.536	9.6057	-71.134	-29.9383

B.2.5 Ionosphere

(a) Group Statistics					
	OPK	**N**	**Mean**	**Std. Deviation**	**Std. Error Mean**
Accuracy %	Stage 2 Chebyshev kernel - vectorial	10	82.18543	23.086014	7.300438
	Stage 3 Generalized Chebyshev kernel -vectorial	10	89.53642	4.127505	1.305232

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	**Sig.**	**t**	**df**	**Sig. (2-tailed)**	**Mean Difference**	**Std. Error Difference**	**Lower**	**Upper**
Accuracy %	Equal variances assumed	2.49	0.131	-0.99	18	0.335	-7.3509	7.4162	-22.931	8.229865
	Equal variances not assumed			-0.99	9.575	0.346	-7.3509	7.4162	-23.975	9.273344

(a) Group Statistics					
	OPK	**N**	**Mean**	**Std. Deviation**	**Std. Error Mean**
Accuracy %	Stage 2 Chebyshev kernel - vectorial	10	82.18543	23.086014	7.300438
	Stage 3 Modified Chebyshev kernel -vectorial	10	91.85431	3.899858	1.233243

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	**Sig.**	**t**	**df**	**Sig. (2-tailed)**	**Mean Difference**	**Std. Error Difference**	**Lower**	**Upper**
Accuracy %	Equal variances assumed	2.51	0.130	-1.30	18	0.208	-9.6688	7.4038	-25.223	5.88607
	Equal variances not assumed			-1.30	9.513	0.222	-9.6688	7.4038	-26.280	6.94310

(a) Group Statistics					
	OPK	**N**	**Mean**	**Std. Deviation**	**Std. Error Mean**
Accuracy %	Stage 2 Hermite kernel - vectorial	10	74.83444	30.402731	9.614188
	Stage 3 Composite Hermite kernel - vectorial	10	85.69536	10.743094	3.397265

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	**Sig.**	**t**	**df**	**Sig. (2-tailed)**	**Mean Difference**	**Std. Error Difference**	**Lower**	**Upper**
Accuracy %	Equal variances assumed	5.02	0.038	-1.065	18	0.301	-10.860	10.196	-32.283	10.5616
	Equal variances not assumed			-1.065	11.2	0.309	-10.860	10.196	-33.251	11.5300

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre kernel - vectorial	10	82.31788	23.165245	7.325494
	Stage 3 Modified Legendre kernel - vectorial	10	92.05298	3.759248	1.188778

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	2.59	0.125	-1.312	18	0.206	-9.7351	7.4213	-25.326	5.85652
	Equal variances not assumed			-1.312	9.47	0.220	-9.7351	7.4213	-26.396	6.92600

B.2.6 Thyroid

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev kernel - vectorial	10	67.43874	44.947365	14.213605
	Stage 3 Generalized Chebyshev kernel-vectorial	10	86.87281	29.695142	9.390429

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	5.54	0.030	-1.141	18	0.269	-19.434	17.035	-55.224	16.3560
	Equal variances not assumed			-1.141	15.5	0.271	-19.434	17.035	-55.623	16.7550

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev kernel - vectorial	10	67.43874	44.947365	14.213605
	Stage 3 Modified Chebyshev kernel-vectorial	10	96.41774	0.498256	0.157562

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	46.4	0.000	-2.039	18	0.056	-28.978	14.214	-58.842	0.88451
	Equal variances not assumed			-2.039	9.00	0.072	-28.978	14.214	-61.133	3.17518

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite kernel - vectorial	10	94.49533	1.250036	0.395296
	Stage 3 Composite Hermite kernel - vectorial	10	95.44049	1.930472	0.610469

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	3.148	0.093	-1.30	18	0.210	-0.9451	0.72727	-2.4731	0.58279
	Equal variances not assumed			-1.30	15.4	0.213	-0.9451	0.72727	-2.4916	0.60133

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre kernel - vectorial	10	67.37165	44.901133	14.198985
	Stage 3 Modified Legendre kernel - vectorial	10	96.34481	0.545238	0.172419

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	46.201	0.000	-2.04	18	0.056	-28.973	14.2000	-58.806	0.85999
	Equal variances not assumed			-2.04	9.00	0.072	-28.973	14.2000	-61.094	3.14809

B.3 Statistical significance tests on the experimental results of the data adaptation approach proposed in Chapter 5 – Section 5.4.3

B.3.1 Breast cancer dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev - Standard normalization	10	78.46482	19.321978	6.110146
	Stage 2 Chebyshev - Monotonic normalization	10	95.82090	1.494903	0.472730

(b) Independent Samples Test									
		Levene's Test for Equality of Variances		t-test for Equality of Means					
									95% Confidence Interval of the Difference
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Differe	

								nce	Lower	Upper
Accuracy %	Equal variances assumed	177.883	0.000	-2.83	18	0.011	-17.356	6.12840	-30.231	-4.4807
	Equal variances not assumed			-2.83	9.10	0.019	-17.356	6.12840	-31.194	-3.5176

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre - Standard normalization	10	81.51386	19.098217	6.039387
	Stage 2 Legendre - Monotonic normalization	10	96.18337	1.397993	0.442084

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	92.529	.000	-2.42	18	0.026	-14.669	6.05554	-27.391	-1.9472
	Equal variances not assumed			-2.42	9.09	0.038	-14.669	6.05554	-28.345	-0.9930

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite - Standard normalization	10	80.31983	20.832342	6.587765
	Stage 2 Hermite - Monotonic normalization	10	87.05756	17.343623	5.484535

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	1.037	0.322	-0.78	18	0.442	-6.7377	8.57197	-24.746	11.2713
	Equal variances not assumed			-0.78	17.4	0.442	-6.7377	8.57197	-24.789	11.3138

B.3.2 Iris dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev - Standard normalization	10	93.04762	1.798630	0.568777
	Stage 2 Chebyshev - Monotonic normalization	10	96.00000	1.253870	0.396509

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	1.180	0.292	-4.25	18	0.0004	-2.9523	0.69334	-4.4090	-1.4957
	Equal variances not assumed			-4.25	16.0	0.0005	-2.9523	0.69334	-4.4216	-1.4831

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre - Standard normalization	10	92.76190	2.298026	0.726700
	Stage 2 Legendre - Monotonic normalization	10	96.00000	1.170737	0.370220

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	2.555	0.127	-3.97	18	0.001	-3.2381	0.81557	-4.9515	-1.5246
	Equal variances not assumed			-3.97	13.3	0.002	-3.2381	0.81557	-4.9949	-1.4812

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite - Standard normalization	10	80.85714	22.156137	7.006386
	Stage 2 Hermite - Monotonic normalization	10	88.57142	14.903760	4.712983

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	1.312	0.267	-0.91	18	0.373	-7.7142	8.44403	-25.454	10.0259
	Equal variances not assumed			-0.91	15.7	0.375	-7.7142	8.44403	-25.636	10.2083

B.3.3 Image segmentation dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev - Standard normalization	10	46.30953	39.165608	12.385253
	Stage 2 Chebyshev - Monotonic normalization	10	90.34762	1.246825	0.394281

(b) Independent Samples Test		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	198.836	0.000	-3.55	18	0.002	-44.038	12.3915	-70.071	-18.004
	Equal variances not assumed			-3.55	9.01	0.006	-44.038	12.3915	-72.061	-16.015

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre - Standard normalization	10	53.97619	39.814133	12.590334
	Stage 2 Legendre - Monotonic normalization	10	90.57619	1.198690	0.379059

(b) Independent Samples Test		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	9414.67	0.000	-2.90	18	0.009	-36.600	12.5960	-63.063	-10.136
	Equal variances not assumed			-2.90	9.01	0.017	-36.600	12.5960	-65.086	-8.1136

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite - Standard normalization	10	46.84762	38.913001	12.305371
	Stage 2 Hermite - Monotonic normalization	10	72.61429	31.553818	9.978193

(b) Independent Samples Test		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	3.572	0.075	-1.62	18	0.121	-25.766	15.8425	-59.050	7.51730

	Equal variances not assumed			-1.62	17.2	0.122	-25.766	15.8425	-59.152	7.61944

B.3.4 Ionosphere dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev - Standard normalization	10	82.18543	23.086014	7.300438
	Stage 2 Chebyshev - Monotonic normalization	10	91.72185	3.889849	1.230078

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	2.890	0.106	-1.28	18	0.214	-9.5364	7.40334	-25.090	6.01742
	Equal variances not assumed			-1.28	9.51	0.228	-9.5364	7.40334	-26.147	7.07503

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre - Standard normalization	10	82.31788	23.165245	7.325494
	Stage 2 Legendre - Monotonic normalization	10	92.38411	3.697161	1.169145

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	2.758	0.114	-1.35	18	0.192	-10.066	7.41820	-25.651	5.51884
	Equal variances not assumed			-1.35	9.45	0.206	-10.066	7.41820	-26.724	6.59180

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite - Standard normalization	10	74.83444	30.402731	9.614188
	Stage 2 Hermite - Monotonic normalization	10	67.81457	33.576154	10.617712

(b) Independent Samples Test		
	Levene's Test for Equality of Variances	t-test for Equality of Means

		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	0.743	0.400	0.49	18	0.630	7.01986	14.3237	-23.073	37.1128
	Equal variances not assumed			0.49	17.8	0.630	7.01986	14.3237	-23.094	37.1339

B.3.5 Thyroid dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev - Standard normalization	10	67.43874	44.947365	14.213605
	Stage 2 Chebyshev - Monotonic normalization	10	95.42299	0.114603	0.036241

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	47.009	0.000	-1.96	18	0.065	-27.984	14.2136	-57.846	1.87752
	Equal variances not assumed			-1.96	9.00	0.080	-27.984	14.2136	-60.137	4.16920

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre - Standard normalization	10	67.37165	44.901133	14.198985
	Stage 2 Legendre - Monotonic normalization	10	95.41424	0.104549	0.033061

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	47.056	0.000	-1.97	18	0.064	-28.042	14.1990	-57.873	1.78845
	Equal variances not assumed			-1.97	9.00	0.080	-28.042	14.1990	-60.162	4.07777

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite - Standard normalization	10	94.49533	1.250036	.395296
	Stage 2 Hermite - Monotonic normalization	10	85.62719	29.341038	9.278451

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	4.419	0.050	0.95	18	0.352	8.86814	9.28686	-10.642	28.3791
	Equal variances not assumed			0.95	9.03	0.364	8.86814	9.28686	-12.128	29.8649

B.3.6 Two Spirals dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev - Standard normalization	10	58.15561	10.697385	3.382810
	Stage 2 Chebyshev - Monotonic normalization	10	55.17049	6.303469	1.993332

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	7.120	0.016	0.760	18	0.457	2.985121	3.926420	-5.264	11.234
	Equal variances not assumed			0.760	14.58	0.459	2.985121	3.926420	-5.405	11.3753

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre - Standard normalization	10	57.94508	10.759156	3.402344
	Stage 2 Legendre - Monotonic normalization	10	55.57025	6.846862	2.165168

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper

Accuracy %	Equal variances assumed	5.416	0.032	0.589	18	0.563	2.374826	4.032852	-6.0979	10.8475
	Equal variances not assumed			0.589	15.26	0.565	2.374826	4.032852	-6.2081	10.9578

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite - Standard normalization	10	53.28833	6.426636	2.032281
	Stage 2 Hermite - Monotonic normalization	10	51.72586	4.998301	1.580602

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	0.430	0.520	0.607	18	0.552	1.562472	2.574581	-3.846522	6.971466
	Equal variances not assumed			0.607	16.97	0.552	1.562472	2.574581	-3.8701	6.995061

B.4 Statistical significance tests on the experimental results of the proposed concatenated approach using Stage 1-linear kernels – Chapter 6 – Section 6.5.2.1

B.4.1 Breast cancer dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Chebyshev - Vectorial - Linear	20	43.33689	34.939943	7.812809
	Stage 1 Chebyshev – Concatenated - Linear	20	70.85288	14.755936	3.299528

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	25.800	0.000	-3.24	38	0.002	-27.515	8.48097	-44.684	-10.347
	Equal variances not assumed			-3.24	25.5	0.003	-27.515	8.48097	-44.963	-10.068

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Legendre - Vectorial - Linear	20	48.99787	34.153875	7.637039

		Stage 1 Legendre - Concatenated - Linear		20	68.44350		15.987656		3.574949	
(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	13.444	0.001	-2.30	38	0.027	-19.445	8.43235	-36.516	-2.3752
	Equal variances not assumed			-2.30	26.9	0.029	-19.445	8.43235	-36.749	-2.1422

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Hermite - Vectorial - Linear	20	43.08102	35.027803	7.832455
	Stage 1 Hermite - Concatenated - Linear	20	70.33049	17.740141	3.966816

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	18.325	0.000	-3.10	38	0.004	-27.249	8.77969	-45.023	-9.4759
	Equal variances not assumed			-3.10	28.1	0.004	-27.249	8.77969	-45.229	-9.2692

B.4.2 Image segmentation dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Chebyshev - Vectorial - Linear	20	24.09286	23.617484	5.281030
	Stage 1 Chebyshev - Concatenated - Linear	20	64.20476	15.576239	3.482953

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	0.083	0.774	-6.34	38	1.9430 E-07	-40.111	6.32615	-52.918	-27.305
	Equal variances not assumed			-6.34	32.8	3.6001 E-07	-40.111	6.32615	-52.984	-27.239

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Legendre - Vectorial - Linear	20	27.81905	27.799590	6.216177
	Stage 1 Legendre - Concatenated - Linear	20	65.81667	13.429480	3.002923

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	3.235	0.080	-5.50	38	0.0000	-37.997	6.90350	-51.973	-24.022
	Equal variances not assumed			-5.5	27.4	7.5080 E-06	-37.997	6.90350	-52.152	-23.842

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Hermite - Vectorial - Linear	20	23.64048	23.876859	5.339028
	Stage 1 Hermite - Concatenated - Linear	20	39.98095	35.853990	8.017196

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	15.087	0.000	-1.69	38	0.098	-16.340	9.63227	-35.839	3.15903
	Equal variances not assumed			-1.69	33.0	0.099	-16.340	9.63227	-35.935	3.25466

B.4.3 Ionosphere dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Chebyshev - Vectorial - Linear	20	43.07947	40.289771	9.009067
	Stage 1 Chebyshev - Concatenated - Linear	20	84.83444	4.098756	0.916510

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	343.546	0.000	-4.61	38	0.0000	-41.754	9.05556	-60.087	-23.422

	Equal variances not assumed			-4.61	19.3	0.0001	-41.754	9.05556	-60.682	-22.827

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Legendre - Vectorial - Linear	20	41.82119	38.363257	8.578285
	Stage 1 Legendre - Concatenated - Linear	20	91.12583	3.767151	0.842361

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	135.477	0.000	-5.72	38	1.3783 E-06	-49.304	8.61954	-66.753	-31.855
	Equal variances not assumed			-5.72	19.3	0.0000	-49.304	8.61954	-67.322	-31.286

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Hermite - Vectorial - Linear	20	34.56953	32.347096	7.233031
	Stage 1 Hermite - Concatenated - Linear	20	54.40397	35.107742	7.850330

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	2.697	0.109	-1.85	38	0.071	-19.834	10.6744	-41.443	1.77490
	Equal variances not assumed			-1.85	37.7	0.071	-19.834	10.6744	-41.448	1.77964

B.4.4 Iris dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Chebyshev - Vectorial - Linear	20	50.85714	19.146112	4.281201
	Stage 1 Chebyshev - Concatenated - Linear	20	52.85714	15.460470	3.457066

(b) Independent Samples Test		
	Levene's Test for Equality of Variances	t-test for Equality of Means

		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	6.275	0.017	-0.36	38	0.718	-2.0000	5.50272	-13.139	9.13968
	Equal variances not assumed			-0.36	36.3	0.718	-2.0000	5.50272	-13.155	9.15593

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Legendre - Vectorial - Linear	20	52.23809	20.902824	4.674013
	Stage 1 Legendre - Concatenated - Linear	20	52.72738	14.472676	3.236189

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	10.298	0.003	-0.08	38	0.932	-0.4892	5.68500	-11.997	11.0194
	Equal variances not assumed			-0.08	33.8	0.932	-0.4892	5.68500	-12.044	11.0663

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Hermite - Vectorial - Linear	20	46.28571	19.205859	4.294561
	Stage 1 Hermite - Concatenated - Linear	20	55.52381	24.389136	5.453577

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	4.067	0.051	-1.33	38	0.191	-9.2380	6.94152	-23.290	4.81428
	Equal variances not assumed			-1.33	36.0	0.192	-9.2380	6.94152	-23.315	4.83970

B.4.5 Thyroid dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 1 Chebyshev-Vectorial-Linear	20	44.15694	45.493696	10.172700

		Stage 1 Chebyshev-Concatenated-Linear			20	84.21180		8.025580		1.794574
(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	700.123	4.38E-26	-3.878	38	0.0004	-40.05	10.329778	-60.966	-19.143
	Equal variances not assumed			-3.878	20.18	0.001	-40.05	10.32977	-61.59	-18.5197

(a) Group Statistics										
		OPK			N	Mean		Std. Deviation		Std. Error Mean
Accuracy %		Stage 1 Legendre-Vectorial-Linear			20	39.61056		44.965341		10.054556
		Stage 1 Legendre-Concatenated-Linear			20	84.55654		8.124930		1.816790
(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	275.955	5.14E-19	-4.399	38	0.00008	-44.95	10.21738	-65.63	-24.262
	Equal variances not assumed			-4.399	20.24	0.00027	-44.946	10.2174	-66.243	-23.649

(a) Group Statistics										
		OPK			N	Mean		Std. Deviation		Std. Error Mean
Accuracy %		Stage 1 Hermite-Vectorial-Linear			20	74.79580		37.155782		8.308286
		Stage 1 Hermite-Concatenated-Linear			20	91.02454		3.349321		0.748931
(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	27.573	6.05E-06	-1.945	38	0.059	-16.229	8.341973	-33.116	0.658705
	Equal variances not assumed			-1.945	19.31	0.066	-16.229	8.341973	-33.6698	1.212336

B.4.5 Two Spirals dataset

(a) Group Statistics						
	OPK	N	Mean	Std. Deviation	Std. Error Mean	
Accuracy %	Stage 1 Chebyshev-Vectorial-Linear	20	52.52746	13.565164	3.033263	
	Stage 1 Chebyshev-Concatenated-Linear	20	62.23120	8.589816	1.920741	

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	2.834	0.100	-2.703	38	0.010	-9.704	3.590255	-16.972	-2.4356
	Equal variances not assumed			-2.703	32.13	0.011	-9.7037	3.590255	-17.016	-2.392

(a) Group Statistics						
	OPK	N	Mean	Std. Deviation	Std. Error Mean	
Accuracy %	Stage 1 Legendre-Vectorial-Linear	20	51.30091	13.506317	3.020104	
	Stage 1 Legendre-Concatenated-Linear	20	60.31873	7.423986	1.660054	

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	3.029	0.090	-2.617	38	0.013	-9.0178	3.446275	-15.994	-2.0412
	Equal variances not assumed			-2.617	29.52	0.014	-9.0178	3.446275	-16.061	-1.9748

(a) Group Statistics						
	OPK	N	Mean	Std. Deviation	Std. Error Mean	
Accuracy %	Stage 1 Hermite-Vectorial-Linear	20	36.68650	20.828011	4.657285	
	Stage 1 Hermite-Concatenated-Linear	20	52.14110	3.330980	0.744830	

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper

Accuracy %	Equal variances assumed	30.634	2.47E-06	-3.277	38	0.002	-15.455	4.716468	-25.003	-5.907
	Equal variances not assumed			-3.277	19.97	0.004	-15.455	4.716468	-25.294	-5.615

B.5 Statistical significance tests on the experimental results of the proposed concatenated approach using Stage 2-linear kernels – Chapter 6 – Section 6.5.2.2

B.5.1 Breast cancer dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev - Vectorial - Linear	20	76.92964	17.317013	3.872202
	Stage 2 Chebyshev - Concatenated - Linear	20	91.60981	9.916136	2.217315

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	33.329	0.000	-3.29	38	0.002	-14.680	4.46211	-23.713	-5.6470
	Equal variances not assumed			-3.29	30.2	0.003	-14.680	4.46211	-23.789	-5.5704

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre - Vectorial - Linear	20	76.92964	17.369440	3.883925
	Stage 2 Legendre - Concatenated - Linear	20	92.89979	10.076231	2.253114

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	32.478	0.000	-3.55	38	0.001	-15.970	4.49014	-25.059	-6.8803
	Equal variances not assumed			-3.55	30.4	0.001	-15.970	4.49014	-25.134	-6.8061

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite - Vectorial - Linear	20	76.86567	15.629616	3.494888

		Stage 2 Hermite - Concatenated - Linear			20	89.58422		13.215502		2.955076
(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	3.649	0.064	-2.77	38	0.008	-12.718	4.57675	-21.983	-3.4533
	Equal variances not assumed			-2.77	36.9	0.009	-12.718	4.57675	-21.992	-3.4449

B.5.2 Image segmentation dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev - Vectorial - Linear	20	30.60476	31.405298	7.022438
	Stage 2 Chebyshev - Concatenated - Linear	20	92.80476	0.647393	0.144762

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	32.290	0.000	-8.85	38	8.9396 E-11	-62.199	7.02393	-76.419	-47.980
	Equal variances not assumed			-8.85	19.0	3.5651 E-08	-62.199	7.02393	-76.900	-47.499

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre - Vectorial - Linear	20	34.44048	33.950955	7.591664
	Stage 2 Legendre - Concatenated - Linear	20	93.21429	0.776873	0.173714

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	54.196	0.000	-7.74	38	2.5105 E-09	-58.773	7.59365	-74.146	-43.401
	Equal variances not assumed			-7.74	19.0	2.7097 E-07	-58.773	7.59365	-74.666	-42.881

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite - Vectorial - Linear	20	31.15000	31.326364	7.004788
	Stage 2 Hermite - Concatenated - Linear	20	41.60476	37.811108	8.454821

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	4.787	0.035	-0.95	38	0.347	-10.454	10.9795	-32.681	11.7722
	Equal variances not assumed			-0.95	36.7	0.347	-10.454	10.9795	-32.707	11.7974

B.5.3 Ionosphere dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev - Vectorial - Linear	20	73.60927	28.834298	6.447545
	Stage 2 Chebyshev - Concatenated - Linear	20	96.52318	1.591653	0.355904

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	28.247	0.000	-3.54	38	0.001	-22.913	6.45736	-35.986	-9.8416
	Equal variances not assumed			-3.54	19.1	0.002	-22.913	6.45736	-36.423	-9.4040

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre - Vectorial - Linear	20	73.67549	28.885168	6.458920
	Stage 2 Legendre - Concatenated - Linear	20	96.05960	1.366956	0.305661

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	29.003	0.000	-3.46	38	0.001	-22.384	6.46614	-35.474	-9.2940

	Equal variances not assumed			-3.46	19.0	0.003	-22.384	6.46614	-35.913	-8.8543

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite - Vectorial - Linear	20	53.11258	36.314740	8.120223
	Stage 2 Hermite - Concatenated - Linear	20	79.07285	16.490494	3.687387

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	69.355	0.000	-2.91	38	0.006	-25.960	8.91823	-44.014	-7.9062
	Equal variances not assumed			-2.91	26.5	0.007	-25.960	8.91823	-44.274	-7.6459

B.5.4 Iris dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev - Vectorial - Linear	20	74.47619	27.719290	6.198222
	Stage 2 Chebyshev - Concatenated - Linear	20	95.14286	1.412612	0.315870

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	86.765	0.000	-3.33	38	0.002	-20.666	6.20626	-33.230	-8.1027
	Equal variances not assumed			-3.33	19.0	0.004	-20.666	6.20626	-33.651	-7.6813

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre - Vectorial - Linear	20	80.09524	24.180371	5.406895
	Stage 2 Legendre - Concatenated - Linear	20	95.61905	0.947355	0.211835

(b) Independent Samples Test									
	Levene's Test for Equality of Variances		t-test for Equality of Means						
								95% Confidence Interval of the Difference	
	F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Differe	Lower	Upper

								nce	Lower	Upper
Accuracy %	Equal variances assumed	29.951	0.000	-2.86	38	0.007	-15.523	5.41104	-26.477	-4.5697
	Equal variances not assumed			-2.86	19.0	0.010	-15.523	5.41104	-26.846	-4.2007

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite - Vectorial - Linear	20	56.38095	30.198610	6.752615
	Stage 2 Hermite - Concatenated - Linear	20	61.76191	28.923933	6.467588

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	0.080	0.779	-0.57	38	0.568	-5.3809	9.35026	-24.309	13.5476
	Equal variances not assumed			-0.57	37.9	0.568	-5.3809	9.35026	-24.310	13.5488

B.5.5 Thyroid dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev-Vectorial-Linear	20	34.84685	45.553387	10.186047
	Stage 2 Chebyshev-Concatenated-Linear	20	87.06740	7.436096	1.662762

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	130.355	7.57E-14	-5.060	38	0.00001	-52.22	10.321	-73.114	-31.327
	Equal variances not assumed			-5.060	20.01	0.00005	-52.221	10.321	-73.749	-30.69

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Legendre-Vectorial-Linear	20	34.81330	45.506514	10.175566
	Stage 2 Legendre-Concatenated-Linear	20	89.44130	5.396388	1.206669
(b) Independent Samples Test					

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	147.129	1.23E-14	-5.331	38	0.000004699	-54.628	10.247	-75.372	-33.884
	Equal variances not assumed			-5.331	19.53	0.00003	-54.628	10.2469	-76.035	-33.22

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Hermite-Vectorial-Linear	20	89.21820	19.823570	4.432685
	Stage 2 Hermite-Concatenated-Linear	20	94.22998	1.627967	.364024

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	3.139	.084	-1.13	38	0.267	-5.012	4.4476	-14.02	3.9919
	Equal variances not assumed			-1.13	19.3	0.274	-5.012	4.4476	-14.31	4.2888

B.5.6 Two Spirals dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 2 Chebyshev-Vectorial-Linear	20	77.95538	21.678022	4.847353
	Stage 2 Chebyshev-Concatenated-Linear	20	79.92778	19.572776	4.376606

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	0.754	0.391	-0.302	38	0.764	-1.972	6.530812	-15.19	11.2485
	Equal variances not assumed			-0.302	37.6	0.764	-1.9724	6.530812	-15.198	11.25

(a) Group Statistics						
	OPK	N	Mean	Std. Deviation	Std. Error Mean	
Accuracy %	Stage 2 Legendre-Vectorial-Linear	20	77.98169	21.914614	4.900257	
	Stage 2 Legendre-Concatenated-Linear	20	80.90624	19.343291	4.325291	

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	1.114	0.298	-0.447	38	0.657	-2.925	6.536	-16.156	10.307
	Equal variances not assumed			-0.447	37.4	0.657	-2.9245	6.5361	-16.163	10.314

(a) Group Statistics						
	OPK	N	Mean	Std. Deviation	Std. Error Mean	
Accuracy %	Stage 2 Hermite-Vectorial-Linear	20	52.51945	4.933951	1.103265	
	Stage 2 Hermite-Concatenated-Linear	20	60.31385	4.002174	0.894913	

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	0.032	0.860	-5.487	38	0.00000288	-7.7944	1.420586	-10.67	-4.9186
	Equal variances not assumed			-5.487	36.45	3.25E-06	-7.7944	1.420586	-10.674	-4.915

B.6 Statistical significance tests on the experimental results of the proposed concatenated approach using Stage 3-linear kernels – Chapter 6 – Section 6.5.2.3

B.6.1 Breast cancer dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 3 Modified Chebyshev-Vectorial - Linear	15	96.41791	1.045178	0.269864
	Stage 3 Modified Chebyshev-Concatenated - Linear	15	96.03412	1.323728	0.341785

(b) Independent Samples Test		
	Levene's Test for Equality of Variances	t-test for Equality of Means

		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	1.523	0.227	0.88	28	0.386	0.38379	0.43548	-0.5082	1.27583
	Equal variances not assumed			0.88	26.5	0.386	0.38379	0.43548	-0.5104	1.27800

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 3 Modified Legendre-Vectorial - Linear	15	96.26155	1.113377	0.287473
	Stage 3 Modified Legendre-Concatenated - Linear	15	96.88699	0.776337	0.200449

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	1.597	0.217	-1.78	28	0.085	-0.6254	0.35045	-1.3433	0.09243
	Equal variances not assumed			-1.78	25.0	0.086	-0.6254	0.35045	-1.3472	0.09632

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 3 Composite Hermite-Vectorial - Linear	15	89.60910	8.318090	2.147721
	Stage 3 Composite Hermite-Concatenated - Linear	15	94.58422	1.865432	0.481653

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	21.141	0.000	-2.26	28	0.032	-4.9751	2.20106	-9.4838	-0.4664
	Equal variances not assumed			-2.26	15.4	0.039	-4.9751	2.20106	-9.6558	-0.2943

B.6.2 Image segmentation dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 3 Modified Chebyshev-Vectorial - Linear	15	91.49206	1.245226	0.321516

		Stage 3 Modified Chebyshev-Concatenated - Linear		15	94.53968		0.721204		0.186214	
(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	4.925	0.035	-8.20	28	6.2855 E-09	-3.0476	0.37154	-3.8087	-2.2865
	Equal variances not assumed			-8.20	22.4	3.3496 E-08	-3.0476	0.37154	-3.8172	-2.2779

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 3 Modified Legendre-Vectorial - Linear	15	91.48889	1.217578	0.314377
	Stage 3 Modified Legendre-Concatenated - Linear	15	94.38095	0.717001	0.185129

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	4.016	0.055	-7.92	28	1.2381 E-08	-2.8920	0.36483	-3.6393	-2.1447
	Equal variances not assumed			-7.92	22.6	5.5628 E-08	-2.8920	0.36483	-3.6474	-2.1367

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 3 Composite Hermite-Vectorial - Linear	15	89.55556	2.691132	0.694847
	Stage 3 Composite Hermite-Concatenated - Linear	15	93.47302	0.290473	0.075000

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	38.987	0.000	-5.60	28	5.3203 E-06	-3.9174	0.69888	-5.3490	-2.4858
	Equal variances not assumed			-5.6	14.3	0.0000	-3.9174	0.69888	-5.4132	-2.4216

B.6.3 Ionosphere dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 3 Modified Chebyshev-Vectorial - Linear	15	89.88963	4.302574	1.110920
	Stage 3 Modified Chebyshev-Concatenated - Linear	15	97.39514	0.466050	0.120334

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	23.059	0.000	-6.71	28	2.7213 E-07	-7.5055	1.11741	-9.7944	-5.2165
	Equal variances not assumed			-6.71	14.3	8.7333 E-06	-7.5055	1.11741	-9.8969	-5.1140

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 3 Modified Legendre-Vectorial - Linear	15	90.28698	4.033501	1.041446
	Stage 3 Modified Legendre-Concatenated - Linear	15	97.96910	0.170995	0.044151

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	27.847	0.000	-7.37	28	5.0255 E-08	-7.6821	1.04238	-9.8173	-5.5469
	Equal variances not assumed			-7.37	14.0	3.4378 E-06	-7.6821	1.04238	-9.9170	-5.4471

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 3 Composite Hermite-Vectorial - Linear	15	78.94040	13.377295	3.454003
	Stage 3 Composite Hermite-Concatenated - Linear	15	92.98013	3.634205	0.938348

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper

Accuracy %	Equal variances assumed	25.668	0.000	-3.92	28	0.001	-14.039	3.57919	-21.371	-6.7080
	Equal variances not assumed			-3.92	16.0	0.001	-14.039	3.57919	-21.625	-6.4543

B.6.4 Iris dataset

(a) Group Statistics						
	OPK	N	Mean	Std. Deviation	Std. Error Mean	
Accuracy %	Stage 3 Modified Chebyshev-Vectorial - Linear	15	92.44445	4.768250	1.231157	
	Stage 3 Modified Chebyshev-Concatenated - Linear	15	97.07937	0.670220	0.173050	

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	32.423	0.000	-3.72	28	0.001	-4.6349	1.24325	-7.1816	-2.0882
	Equal variances not assumed			-3.72	14.5	0.002	-4.6349	1.24325	-7.2919	-1.9778

(a) Group Statistics						
	OPK	N	Mean	Std. Deviation	Std. Error Mean	
Accuracy %	Stage 3 Modified Legendre-Vectorial - Linear	15	93.01587	3.682696	0.950868	
	Stage 3 Modified Legendre-Concatenated - Linear	15	97.26984	0.335110	0.086525	

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	61.821	0.000	-4.45	28	0.0001	-4.2539	0.95479	-6.2097	-2.2981
	Equal variances not assumed			-4.45	14.2	0.0005	-4.2539	0.95479	-6.2986	-2.2092

(a) Group Statistics						
	OPK	N	Mean	Std. Deviation	Std. Error Mean	
Accuracy %	Stage 3 Composite Hermite-Vectorial - Linear	15	94.34921	2.822161	0.728679	
	Stage 3 Composite Hermite-Concatenated - Linear	15	97.05397	1.078527	0.278475	
(b) Independent Samples Test						

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	23.209	0.000	-3.46	28	0.002	-2.7047	0.78007	-4.3026	-1.1068
	Equal variances not assumed			-3.46	18.0	0.003	-2.7047	0.78007	-4.3436	-1.0659

B.6.5 Thyroid dataset

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 3 Mod Chebyshev-Vectorial-Linear	20	94.20183	2.800487	0.626208
	Stage 3 Mod Chebyshev-Concatenated-Linear	20	95.62398	2.337266	0.522629

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	0.879	0.354	-1.744	38	0.089	-1.422149	0.815645	-3.073336	0.229038
	Equal variances not assumed			-1.744	36.82	0.090	-1.42215	0.815645	-3.0751	0.230775

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 3 Mod Legendre-Vectorial-Linear	20	94.36197	2.522772	0.564109
	Stage 3 Mod Legendre-Concatenated-Linear	20	95.39557	2.537395	0.567379

(b) Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
									95% Confidence Interval of the Difference	
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	Lower	Upper
Accuracy %	Equal variances assumed	0.036	0.850	-1.292	38	0.204	-1.0336	0.800086	-2.6533	0.586087
	Equal variances not assumed			-1.292	38.0	0.204	-1.0336	0.800086	-2.6533	0.586089

(a) Group Statistics						
	OPK	N	Mean	Std. Deviation	Std. Error Mean	
Accuracy %	Stage 3 Composite Hermite-Vectorial-Linear	20	93.82415	2.618745	0.585569	
	Stage 3 Composite Hermite-Concatenated-Linear	20	95.21841	1.663475	0.371964	

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	1.240	0.272	-2.010	38	0.052	-1.3943	0.693721	-2.7986	0.010105
	Equal variances not assumed			-2.010	32.19	0.053	-1.3943	0.693721	-2.807	0.018483

B.6.6 Two Spirals dataset

(a) Group Statistics						
	OPK	N	Mean	Std. Deviation	Std. Error Mean	
Accuracy %	Stage 3 Mod Chebyshev-Vectorial-Linear	20	90.68565	14.717299	3.290888	
	Stage 3 Mod Chebyshev-Concatenated-Linear	20	95.86020	3.274768	0.732260	

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	9.753	0.003	-1.535	38	0.133	-5.1745	3.371372	-11.9995	1.650438
	Equal variances not assumed			-1.535	20.88	0.140	-5.1745	3.371372	-12.188	1.839124

(a) Group Statistics						
	OPK	N	Mean	Std. Deviation	Std. Error Mean	
Accuracy %	Stage 3 Mod Legendre-Vectorial-Linear	20	97.52746	3.689213	0.824933	
	Stage 3 Mod Legendre-Concatenated-Linear	20	97.92693	3.160725	0.706760	

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	0.885	0.353	-0.368	38	0.715	-0.3995	1.086289	-2.5985	1.799607

	Equal variances not assumed			-0.368	37.126	0.715	-0.3995	1.086289	-2.6002	1.801308

(a) Group Statistics					
	OPK	N	Mean	Std. Deviation	Std. Error Mean
Accuracy %	Stage 3 Composite Hermite-Vectorial-Linear	20	85.68191	4.790208	1.071123
	Stage 3 Composite Hermite-Concatenated-Linear	20	89.77201	1.925383	0.430529

(b) Independent Samples Test										
		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Lower	Upper
Accuracy %	Equal variances assumed	12.769	0.001	-3.543	38	0.001	-4.0901	1.154409	-6.4271	-1.7531
	Equal variances not assumed			-3.543	24.98	0.002	-4.0901	1.154409	-6.4677	-1.7125