SUPERVISED FEATURE LEARNING VIA DEPENDENCY MAXIMIZATION

By CHETAN J. TONDE

A dissertation submitted to the

Graduate School-New Brunswick

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Doctor of Philosophy

Graduate Program in Computer Science

Written under the direction of

Ahmed Elgammal

And approved by

_____

_____

_____

_____

New Brunswick, New Jersey

May, 2016

ABSTRACT OF THE DISSERTATION

Supervised Feature Learning via Dependency Maximization

By CHETAN J. TONDE

Dissertation Director:

Ahmed Elgammal

A key challenge in machine learning is to automatically extract relevant feature representations of data for a given task. This becomes especially formidable task for structured data like images, which are often highly structured and complex. In this thesis, we propose frameworks for supervised feature learning for structured and unstructured data, via dependency maximization.

In the first part of this dissertation we look at the problem of learning kernels for structured prediction. We present a novel framework called Twin Kernel Learning which proposes the idea of polynomial expansions of kernels, to learn kernels over structured data so as to maximize a dependency criterion called Hilbert-Schmidt Independence criterion (HSIC). We also give an efficient, matrix-decomposition based algorithm for learning these expansions and use it to learn covariance kernels of Twin Gaussian Processes. We demonstrate state-of-the-art empirical results on several synthetic and real-world datasets.

In the second part of this work, we present a novel framework for supervised dimensionality reduction based on a dependency criterion called Distance Correlation. Our framework is based on learning low-dimensional features which maximize squared sum of Distance Correlations of low dimensional features, with both, the response, and the covariates. We propose a novel algorithm to maximize our proposed objective, and also show superior empirical results over state-of-the-art on multiple datasets.

Acknowledgements

I would like to thank my family for their love and support all along. I would like to express my deepest gratitude to my advisor, Prof. Ahmed Elgammal for always believing in me, and for granting me the freedom to pursue my own research direction. I am happy to have had such a good hearted and inspiring supervisor.

The Machine Learning and Computer Vision reading groups CBIM have been a fantastic environment, and I have learned so much from talking with others. I also thank my qualifying, and defense examination committee members, Prof. Swastik Kopparty, Prof. Tina Eliassi-Rad, Prof. Pranjal Awasthi, and Prof. Lee Dicker for their guidance and support.

I have enjoyed my time at Rutgers including the seminars and talks in Mathematics, Statistics and Psychology departments that have enriched my experience. I am immensely grateful to all my colleagues and friends that I have made during my time here.

Dedication

Dedicated to my parents and brothers whose love and support have sustained me throughout.

Table of Contents

List of Tables

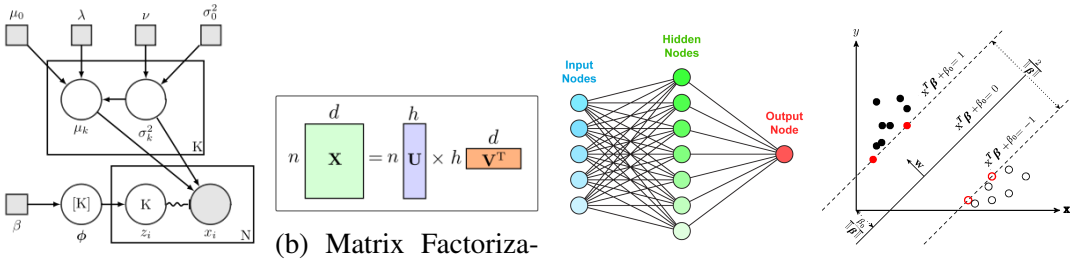
List of Figures

Chapter 1

Introduction

1.1 Overview

Approximately 2.4 billion people spent time on the internet everyday in 2013 [10]. It was estimated that every minute, 2.4 million posts were shared on Facebook ®, 4 million search queries made on Google®, 48 hours of video uploaded to YouTube®, 277,000 tweets posted on Twitter®, 216,000 photos shared on Instagram® and 416,667 messages sent on Whatsapp®. People have become web content creators. All of this generated data is highly complex, structured, and their scale is in thousands of exabytes. Synergistically, with rapid progress in computing and storage, and in machine learning models and algorithms, it has become a major challenge to find a way to transform this massive and complex, yet noisy data, coming from a variety of sources, into meaningful and useful representations. This is a required step to further extract knowledge and perform useful tasks. In this dissertation our goal is to address this specific problem of learning meaningful and useful representations of data, in context of machine learning.



(a) Bayesian Models (b) Matrix Factorization (c) Neural Networks (d) Kernel Methods

Figure 1.1: Different classes of machine learning models.

The goal of machine learning is to model empirical dependencies among multiple latent and observed variables given a set of observed input-output data instances (x_i, y_i). These

dependencies may be deterministic or probabilistic, and can be used to predict unobserved outputs y given observed input x. As a illustrative example, in the task of categorizing web videos with text, available from content sharing sites like YouTube®, input x is videos with text, and output y could be discrete categories such as entertainment, music, news and politics, science and technology among others. Figure 1.1 shows several different classes of models that are used to model these complex dependencies.

In this modeling process a common workflow that is used looks as follows:

$$\text{Data + Model + Algorithm} \implies \textit{Prediction/Inference.}$$

In the above workflow, design of model and algorithm depend on available data, and are tightly coupled with each other. Moreover, it is known that having a task relevant feature representation of data usually leads better performance using simpler algorithms. For models such as Neural Networks [11] and kernel methods [12] algorithms such as back-propagation and automatic kernel learning, respectively, have proven to be useful for feature learning. So an alternate common approach is that of first, choose a problem specific model, then use a general purpose algorithm to learn features, and then finally using a simple algorithm during prediction.

$$\underbrace{(\text{Data + Model + } \textit{General Purpose Algorithm})}_{\text{learn features}} \text{+ Simple Algorithm} \implies \text{Prediction/Inference}$$

In case of kernel methods, the general purpose algorithm would correspond automatic learning of kernel from data, which later is used in conjunction with a kernel method for prediction. In this dissertation, we focus on this second approach of learning features by maximizing statistical dependency between input and output. We propose two frameworks for supervised feature learning; first, for learning kernels for structured data, and second, to learn low-dimensional features for dimensionality reduction.

1.2 Feature Learning via Dependency Maximization

A natural objective for supervised feature learning is to maximize statistical dependency between input and output distributions. In this thesis, we utilize two equivalent measures of dependency described below.

1. RKHS embedding distance between joint distribution $P(x, y)$ and product of its marginals $P(x)P(y)$. This is also called Hilbert-Schmidt Independence criterion (HSIC) [13].

$$HSIC(P(x,y), \mathcal{K}, \mathcal{G}) := \|\mathbf{M}_{xy}\|^2_{HS}$$

where k, g are characteristic kernels and $\mathbf{M}_{xy} := \mathbf{E}_{x,y}\left[(k(x,\cdot) - \mu_x) \otimes (g(y,\cdot) - \mu_y)\right]$.

2. Distance Correlation of [14] which is a non-linear extension of Pearson correlation which is defined in terms of characteristic functions $f_{\mathbf{x}}$, $f_{\mathbf{y}}$, and $f_{\mathbf{x},\mathbf{y}}$ of random variables $\mathbf{x}$, and $\mathbf{y}$. It is derived by normalization of Distance Covariance which is defined as

$$\nu^2(\mathbf{x}, \mathbf{y}) = \int_{\mathbb{R}^{d+m}} |f_{\mathbf{x},\mathbf{y}}(t,s) - f_{\mathbf{x}}(t) f_{\mathbf{y}}(s)|^2 w(t,s) dt ds.$$

where $w(t, s)$ is a suitably defined weight function. Distance Correlation is then defined as

$$\rho^2(\mathbf{x}, \mathbf{y}) = \begin{cases} \frac{\nu^2(\mathbf{x},\mathbf{y})}{\sqrt{\nu^2(\mathbf{x},\mathbf{x})\nu^2(\mathbf{y},\mathbf{y})}}, & \nu^2(\mathbf{x},\mathbf{x})\nu^2(\mathbf{y},\mathbf{y}) > 0. \\ 0, & \nu^2(\mathbf{x},\mathbf{x})\nu^2(\mathbf{y},\mathbf{y}) = 0. \end{cases}$$

HSIC and Distance Covariance are closely related and infact equivalent some cases as shown in section 2.6.4. In sections below we briefly describe our two proposed frameworks.

1.2.1 Kernel Learning for Structured Prediction

Due to complex nature of internet data, and presence of structure within each input (e.g. video) and output (eg. category), a new flavor of models called structured prediction models have been developed in machine learning [15], and also in Computer Vision [16]. For these machine learning models, the inputs and outputs could be simple vectors, or structured objects such as images, videos, text, or graphs. In a web videos catagorization example, input would be the conventional representation of audio-visual features, the associated text metadata (like title, textual description), and even the intricate social network of the related videos. Output in this case would be a set of multiple labels with an associated hierarchy within these labels.

In the kernel methods approach of learning [17], we abstractly define representation space, say Z, characterized by its kernel function $k(x_i, x_j)$, which intuitively measures similarity

between argument instances x_i and x_j, which also helps in scaling up the learning problem by making it independent of the dimension of input data. Hence learning a 'good' representation for your data turns into an equivalent problem of learning a 'good' kernel function. This kernel function is so that it to takes advantage of the problem prediction task and uses all available information in your data.

Structured prediction models handle this data by choosing a kernel function for input instance pairs (x_i, x_j), and also for output instance pairs (y_i, y_j). More importantly they implicitly represent these input-output data instances jointly in a abstract feature space as a vector and this representation is critical for good performance of in a learning task.

In first part of this dissertation, we focus on learning these joint kernel feature spaces which are data and task-specific. We present a novel framework for automated kernel learning for the problem of structured prediction called Twin Kernel Learning [18, 19]. In this framework, we propose a effective and efficient algorithm for learning of kernel-space feature spaces, and demonstrate its efficacy on real world datasets by learning the covariance kernel of Structured Twin Gaussian Processes [20].

1.2.2 Supervised Dimensionality Reduction

Rapid developments of imaging technology, microarray data analysis, computer vision, neuroimaging, hyperspectral data analysis and many other applications call for the analysis of high-dimensional data. Supervised dimensionality reduction problem is concerned with finding a low-dimensional feature representation of data such that, this representation can be used effectively in a supervised learning task. Such representations help in provide a meaningful interpretation and visualization of data, and also helps to reduce sample complexity of learning algorithm.

In second part of this dissertation, we propose a framework for supervised dimensionality reduction for learning low-dimensional features that maximize a recently proposed dependency measure called distance correlation [21]. We also demonstrate state-of-the art results on many real world datasets [22].

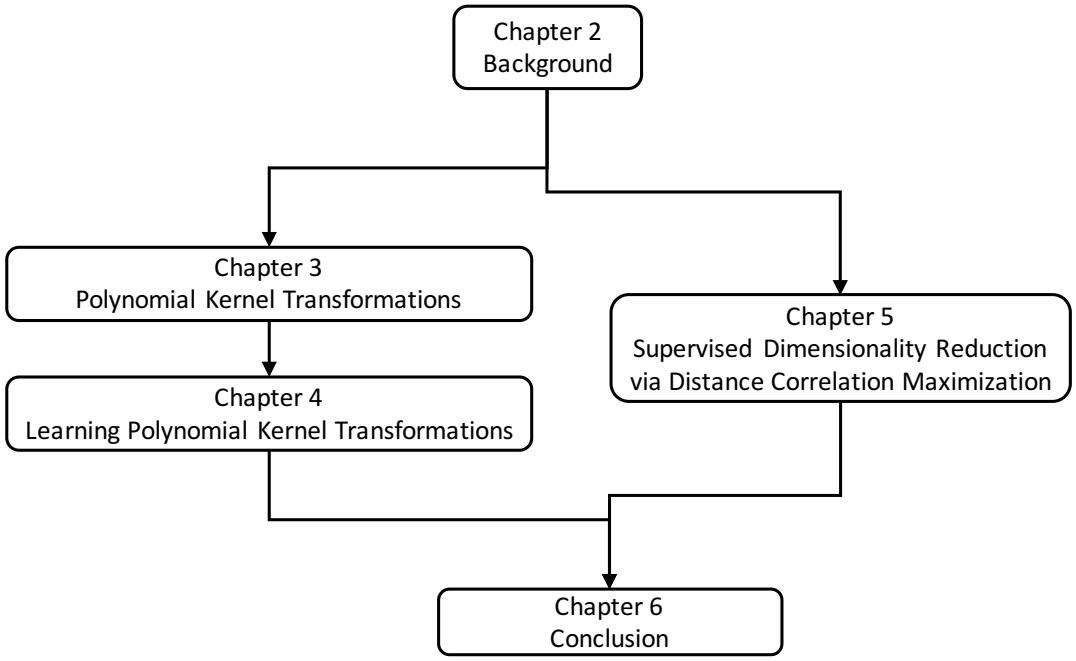


Figure 1.2: Thesis outline.

1.3 Thesis Outline

This thesis is organized into six chapters. Figure 1.2 shows lineage of the chapters. Main contents of each chapter are summarized below:

Chapter 2. Background In this chapter, we provide relevant background required for development of rest of the thesis. We start by describing the general problem of supervised learning and structured prediction. We then describe kernel methods for supervised learning using Support Vector Machines and Support Vector regression as a prototypical examples. We also describe positive definite kernel functions and their properties. Later we describe reproducing kernel Hilbert spaces which play an important role in kernel methods and also in kernel-based dependency measures. Then we describe the kernel functions for learning with example kernels, and describe universal and characteristic kernels which are important from the perspective of dependency measures and consistency of learning algorithms. We then describe the general framework of kernel methods for structured prediction with Kernel Dependency Estimation and

Twin Gaussian Processes as examples. In the last part of this chapter we propose two dependency measures Hilbert Schmidt Independence Criterion and Distance Correlation that we use in our work and discuss connections between them.

Chapter 3. Polynomial Kernel Transformations In this chapter, we propose polynomials expansion of kernels of radial and shift-invariant kernels which we refer to as Schoenberg transformations and Gegenbaur transformations, respectively. These arise from the seminal result of Schoenberg [23] and can be thought of as learning polynomial combination of input features in a high dimensional reproducing kernel Hilbert space (RKHS).

Chapter 4. Learning Polynomial Kernel Transformations In this chapter, we propose a framework for learning kernels for structured prediction. This framework learns two kernels one on input and the other on output such that, dependency between input and output kernel features is maximized. We use Hilbert-Schmidt Independence Criterion (HSIC) as a dependency to measure this. We give an efficient matrix decomposition-based algorithm to learn these transformations, and demonstrate state-of-the-art results on several synthetic and real-world datasets.

Chapter 5. Supervised Dimensionality Reduction via Distance Correlation Maximization In this chapter, we propose a novel framework for supervised dimensionality reduction based on a nonlinear dependency criterion called Statistical Distance Correlation, [14]. Our proposed formulation is based on learning a low-dimensional feature representation $\mathbf{z}$, which maximizes the squared sum of Distance Correlations between low dimensional features $\mathbf{z}$ and response y, and also between features $\mathbf{z}$ and covariates $\mathbf{x}$. We propose a novel algorithm to optimize our proposed objective using the Generalized Minimization Maximization method of Parizi et al. [24]. We show superior empirical results on multiple datasets proving the effectiveness of our proposed approach over several relevant state-of-the-art supervised dimensionality reduction methods.

Chapter 6. Conclusion In this chapter, we summarize the main results in this thesis. We also discuss some future directions for our approach of using nonlinear dependency measures like Hilbert Schmidt Independence Criterion (HSIC) and Distance Correlation (dCorr) for learning kernels for structured data, and learning features low dimensional features for structured data.

1.4 Thesis Contributions

This thesis makes conceptual and algorithmic contributions to the field of Machine Learning and Computer Vision. The contributions described in this dissertation are:

1. Propose polynomials expansion of kernel functions.
2. A framework for learning kernels for structured prediction.
3. Propose a Laplacian version of Distance Correlation.
4. A framework for supervised dimensionality reduction using Distance Correlation.

These contributions have been presented in the following papers of the author.

1. C. Tonde, A. Elgammal, Simultaneous Twin Kernel Learning using Polynomial Transformations for Structured Prediction, *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.
2. C. Tonde, A. Elgammal, Learning Polynomial Kernels Transformations for Structured Prediction, *arxiv:1601.01411 (cs.LG)*, 2016.
3. P. Vepakomma[1], C. Tonde[1], Supervised Dimensionality reduction via Distance Correlation Maximization, *arXiv:1601.00236 (cs.LG)*, 2016.

1.5 Notation

The notation we use in this thesis is as follows: To denote a particular (structured) object from a set of objects we use lower case letter x for the object, and calligraphic uppercase letter $\mathcal{X}$ for

[1]Authors contributed equally.

the set so that $x \in \mathcal{X}$. We denote real vectors as lower case bold letters $x \in \mathbb{R}^d$ where $d \in \mathbb{N}$ is the dimension of the space. In this thesis we mainly deal with vectorial data but many the results can be applied to vectors in Hilbert space indicated by uppercase calligraphic letters $\mathcal{K}$, $\mathcal{G}$ and $\mathcal{F}$. We group together several vectors together to form a set $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$. For scalars we use lower case letters like $y \in \mathbb{R}$, and for matrices we use upper case bold letters $\mathbf{X} \in \mathbb{R}^{m \times d}$. We indicate the $(i,j)^{th}$ element of a matrix $\mathbf{X}$ as $[\mathbf{X}]_{i,j}$. We stack together vectors get matrix $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_m]^T$, or equivalently, stack scalars to get vectors as $\mathbf{y} = [y_1, y_2, \ldots, y_m]$.

In addition, we use $\mathbf{1}_m$ and $\mathbf{0}_m$ to indicate the all ones and all zeros vector in $\mathbb{R}^m$, respectively. Matrix $\mathbf{J}_m$ indicates the centering matrix $\mathbf{J}_m = \mathbf{I} - \frac{1}{m}\mathbf{1}_m\mathbf{1}_m^T$, such that $\widehat{\mathbf{X}} = \mathbf{J}_m\mathbf{X}$ is column-centered (column sum zero) matrix, $\widehat{\mathbf{X}} = \mathbf{J}_m\mathbf{X}$ is row-centered (row sum zero) matrix, and $\widetilde{\mathbf{X}} = \mathbf{J}_m\mathbf{X}\mathbf{J}_p$ is a double-centered (row and column sum zero) matrix. We represent the Laplacian of a weighted adjacency matrix $\mathbf{W}$ (with self loops) is as $\mathbf{L} = \mathbf{D} - \mathbf{W}$ where $\mathbf{D}$ is a diagonal degree matrix with diagonal elements $[\mathbf{D}]_{i,i} = \sum_j [\mathbf{W}]_{i,j}$, and zero off-diagonal entries [25].

Furthermore, we denote spectral radius or maximum eigenvalue of a matrix $\mathbf{M}$ as $\lambda_{max}(\mathbf{M})$, i^{th} eigenvalue by $\lambda_i(\mathbf{M})$, and i^{th} generalized eigenvalue $\mathbf{A}\mathbf{x} = \lambda_i\mathbf{B}\mathbf{x}$ by $\lambda_i(\mathbf{A}, \mathbf{B})$. Moreover, $\lambda_{max}(M)$ ($\lambda_{max}(\mathbf{A}, \mathbf{B})$) and $\lambda_{max}(M)$ ($\lambda_{min}(\mathbf{A}, \mathbf{B})$), respectively, as maximum and minimum eigenvalues (generalized eigenvalues) of matrix $\mathbf{M}$ ($\mathbf{A}$ and $\mathbf{B}$). We use the usual partial ordering for symmetric matrices: $\mathbf{A} \succeq \mathbf{B}$ meaning $\mathbf{A} - \mathbf{B}$ is positive semidefinite; similarly for the relationships $\succeq, \preceq, \succ$. The norm $\|\cdot\|$ will be either the Euclidean norm for vectors or the norm that it induces for matrices, unless otherwise specified.

Chapter 2

Background

2.1 Supervised Learning Problem

In supervised learning the goal is to predict an output target, $y^* \in Y$, given an input object $x^* \in X$, given training data $\{(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)\} \subseteq X \times Y$, independent identically distributed (i.i.d.) samples from joint distribution $P(x, y)$. To do so we estimate a function $f\colon X \to Y$ from a set $\mathcal{H}$ so as to minimize regularized empirical risk

$$L(h) = \underbrace{\sum_{i=1}^{m} V(\mathbf{y}_i, h(\mathbf{x}_i))}_{Loss\ part} + \underbrace{\Omega(||h||^2)}_{regularization} \tag{2.1}$$

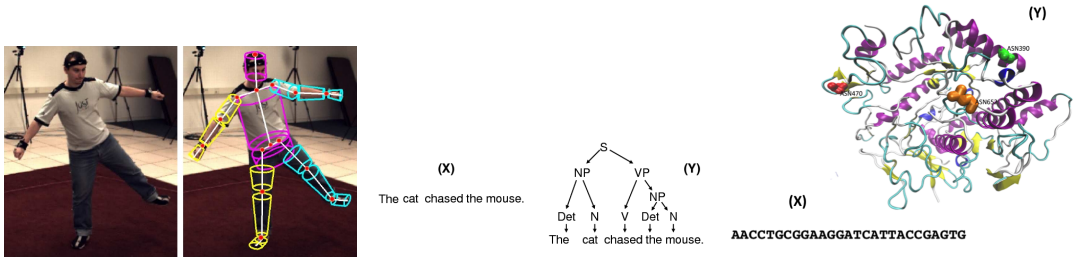
The *loss part* of above risk measures discrepancy between predicted label by function h and training data. The *regularization* part penalizes the estimated function h for being arbitrarily complex so as to achieve no loss on the training data. Regularization leads to a simpler function h that achieves smaller risk, but possibly non-zero risk on the seen training data and better generalization on unseen test data.

Some traditional examples of supervised learning like binary classification, where target is discrete $y_i \in \{+1, -1\}$ with a 0-1 loss or hinge loss (e.g. handwritten digit recognition), or univariate regression where target is scalar $y \in \mathbb{R}$ with squared or exponential loss (e.g. 2D object pose estimation from images). There are pros and cons of each of these loss functions which are described in detail in Bartlett et al. [26]. The regularization term control the complexity of the estimated function h and is also required as the problem is ill-posed [27, 28].

2.2 Structured Prediction Problem

In machine learning a vast majority of prediction problems have *complex* output as opposed to simple ones like binary/multi-class classification or regression. These complex output problems have a target that contain several variables dependent on each other that need to be predicted simultaneously. These problems are referred to as structured prediction or structured learning problems. Examples include the following problems.

- Human 3D-pose reconstruction (Figure 2.1a): given an image or sequence of images, predict 3D x, y, z positions of joint locations of human skeleton.
- Language syntactic parsing (Figure 2.1b): given a input sentence, build a parse tree whose leaves are words of the sentence and whose structure obeys grammar rules.
- Protein 3D-structure prediction (Figure 2.1c): given a string of amino acid sequences, predict its 3D-structure.



(a) Human 3D-pose prediction. (b) Language syntactic parsing. (c) Protein 3D-structure prediction.

In kernel methods for structured prediction a feature function $\Phi(x, y)$ is defined jointly over input $x \in X$ and output $y \in Y$. For example, in part of speech tagging, an element of vector $\Phi(x, y)$ could be the number of times the word "the" appears as a determiner and the next word is a noun. The goal is to learn a weight vector $\mathbf{w}$ such that during testing for a new test example x^* we find the best prediction such that

$$y^* = \arg\max_{y \in Y} \mathbf{w}^T \Phi(x, y) \tag{2.2}$$

The above $\arg\max$ problem is intractable in the general case, but tractable under certain case [15, 29].

2.3 Kernel Methods

Vapnik [27, 30] proposed a *structural risk minimization* (SRM) framework to minimize the empirical risk in Equation 2.1. This framework introduced the use of positive definite kernel functions in solving learning problems. It also gave necessary and sufficient conditions for consistency of any learning process, and generalization bounds in terms of VC-dimension of the hypothesis class $\mathcal{H}$. In sections below we give two seminal algorithms born out above framework that performs SRM for binary classification and regression.

2.3.1 Support Vector Machines

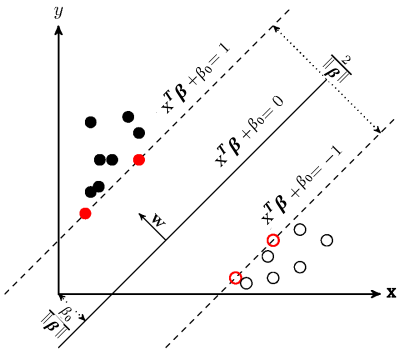


Figure 2.2: Support Vector Machine

The seminal work of Cortes and Vapnik [31] introduced Support Vector Machines (SVM). A hard margin linear SVM for classification is as follows. Given training data $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2) \ldots, (\mathbf{x}_n, y_n)$ with $\mathbf{x}_i \in \mathbf{R}^d$ and $y_i \in \{-1, +1\}$. Define a hyperplane $f(\mathbf{x}) = \mathbf{x}^T\boldsymbol{\beta} + \beta_0$ where $\boldsymbol{\beta}$ is a unit vector $\|\boldsymbol{\beta}\|^2 = 1$, and a classification rule is given by $G(\mathbf{x}) = sign[f(\mathbf{x})]$.

The optimization problem then turn out to be

$$\begin{aligned} &\text{minimize} && \frac{1}{2}\|\boldsymbol{\beta}\|^2 \\ &\text{subject to} && y_i(\mathbf{x}_i^T\boldsymbol{\beta} + \beta_0) \geq 1, \qquad i = 1, 2, \ldots, N. \end{aligned} \tag{2.3}$$

The above formulation works only for the separable case as shown in Figure 2.2 and a solution exists on when the two classes can be separated into two classes by a hyperplane. In case of non-separable data as shown in Figure 2.4, a relaxed version of SVM called soft-margin SVM is used with appropriate slack variables. This optimization problem with slack variables is as

follows.

$$\begin{aligned} &\text{minimize} && \frac{1}{2}\|\boldsymbol{\beta}\|^2 + C\sum_{i=1}^{N}\xi_i \\ &\text{subject to} && y_i(\mathbf{x}_i^T\boldsymbol{\beta} + \beta_0) \geq 1 - \xi_i, \\ & && \xi_i \geq 0, i = 1, 2, \ldots, n. \end{aligned} \tag{2.4}$$

The above formulation allows you to do classification in presence of non-separable data which could be due to problem structure or presence of noise and outliers. The dual problem of the soft-margin formulation is as follows.

$$\begin{aligned} &\text{maximize} && \sum_{i=1}^{N}\alpha_i - \frac{1}{2}y_i y_j \alpha_i \alpha_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle \\ &\text{subject to} && 0 \leq \alpha_i \leq C, \sum_i y_i \alpha_i = 0, \qquad i = 1, 2, \ldots, n. \end{aligned} \tag{2.5}$$

After computing the solution $\boldsymbol{\alpha}^*$, the discriminant function is give as,

$$f(\mathbf{x}) = \boldsymbol{\beta}^{*T}\mathbf{x} + b^* = \sum_{i=1}^{N} y_i \alpha_i^* \langle \mathbf{x}_i, \mathbf{x}_j \rangle + b^*$$

The SVM problem can be solved in dual or primal form with complexity $O(dn^2 + n^3)$ or $O(nd^2 + d^3)$, respectively. The formulation used depends on the scale of the problem. For low dimensional datasets the primal form is preferable, and for high dimensional data the dual for is preferred. So the resulting complexity is $O(\max(n, d), \min(n, d)^2)$ [32].

2.3.2 Support Vector Regression

The idea of Support Vector Machine can also be generalized to solve the regression problem with a scalar response $y \in \mathbb{R}$. The linear regression model is as follows

$$f(\mathbf{x}^T) = \mathbf{x}^T\boldsymbol{\beta} + \beta_0$$

To estimate $\boldsymbol{\beta}$, we solve the minimization problem,

$$L(\boldsymbol{\beta}, \beta_0) = \sum_{i=1}^{N} V(y_i - f(\boldsymbol{x}_i)) + \frac{\lambda}{2} \|\boldsymbol{\beta}\|^2$$

where

$$V_\epsilon(r) = \begin{cases} 0 & if|r| < \epsilon \\ |r| - \epsilon & \text{otherwise} \end{cases}$$

The minimizers of L, are solution of the form,

$$\hat{\beta} = \sum_{i=1}^{N} (\hat{\alpha}_i^* - \alpha_i)\boldsymbol{x}_i, \hat{f}(\boldsymbol{x}) = \sum_{i=1}^{N} (\hat{\alpha}_i^* - \alpha_i) \langle \mathbf{x}, \mathbf{x}_i \rangle + \beta_0$$

where

$$\min_{\alpha_i, \alpha_i^*} \epsilon \sum_{i=1}^{N} (\alpha_i^* + \alpha_i) - \sum_{i=1}^{N} y_i(\alpha_i^* - \alpha_i) + \frac{1}{2} \sum_{i,j=1}^{N} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) \langle \mathbf{x}_i, \mathbf{x_j} \rangle$$

subject to,

$$\begin{aligned} & 0 \leq \alpha_i, \alpha_i^* \leq \frac{1}{\lambda}, \\ & \sum_{i=1}^{N} (\alpha_i^* - \alpha_i) = 0, \\ & \alpha_i \alpha_i^* = 0, i = 1, 2, \ldots, n. \end{aligned} \tag{2.6}$$

Due to constraints most of $(\alpha_i^* - \alpha_i)$ values are zero, and the associated data points are called support vectors. Similar to the classification case the solution depends only on the inner products $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ between data points. We can generalize this algorithm to more general spaces called reproducing kernel Hilbert spaces (RKHS) where inner products are well defined. This is called as the *kernel trick* which is further explained in subsection 2.3.3 below.

2.3.3 Kernel Trick

For binary classification with classes which are not separable, data-points can be mapped to higher dimensional feature space $\Phi \colon X \to \mathcal{H}$, so as to obtain linear separability (Figure 2.4).

This is be done by replacing inner product between data-points a $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ with a positive definite function $k(\mathbf{x}_i, \mathbf{x}_j)$ representing inner product between higher dimensional feature spaces $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j) \rangle$. This positive definite kernel function implicitly represents data points $\mathbf{x}_i$ as points $\Phi(\mathbf{x}_i)$ feature space $\mathcal{H}$, named as reproducing kernel Hilbert space (RKHS). We formally defined them in section 2.3.5.

2.3.4 Positive-Definite Kernels

Positive definite kernel functions used as a kernel are defined as follows:

Definition 1. *A symmetric function $k : X \times X \to \mathbb{R}$ is called positive definite (pd) (resp. conditionally pd) if, for any $n \in \mathbb{N}$ and choice of $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ (resp. with $\sum_j^n \alpha_j = 0$) and $x_1, \ldots, x_n \in X$ we have*

$$\sum_{i,j=1}^{n} \alpha_i \alpha_j k(x_i, x_j) \geq 0. \tag{2.7}$$

Furthermore, k is said to be strictly pd (resp. conditionally strictly pd) if, for mutually distinct $x_1, \ldots x_n \in X$, equality in 2.7 only holds for $\alpha_1 = \ldots = \alpha_n = 0$.

Positive-definite kernel functions also have the following properties for any $x, y \in X$,

- (conjugate symmetry) $k(x, y) = \overline{k(y, x)}$
- (bilinearity) $k(c_1 x + c_2 y, z) = c_1 k(x, z) + c_2 k(y, z)$ for any $c_1, c_2 \in \mathbb{R}$.
- (positive definiteness) $k(x, x) \geq 0$, and $k(x, x) = 0$ if and only if $x = 0$
- (Cauchy-Schwarz inequality) $k(x, y)^2 \leq k(x, x) k(y, y)$

Moreover, positive definite kernel functions have the following closure properties:

Theorem 2. *[33] For kernels $k, k_1, k_2, \ldots$ and so on, defined on $X \times X$, where X is a nonempty set:*

1. *If k_1 and k_2 are positive definite then for any $c_1, c_2 \geq 0$ $c_1 k_1 + c_2 k_2$ is positive definite.*
2. *if $\lim_{n \to \infty} k_n(x, x') = k(x, x')$ exists, then $k(x, x')$ is positive definite. Hence, including all above facts the set of positive definite kernels form a convex cone.*

3. *The point-wise product of kernels $k_1 k_2$ is positive definite.*

4. *If k_1 and k_2 are kernels defined on $X_i \times X_i$ where X_i are nonempty set. Then the tensor product $k_1 \otimes k_2$ and direct sum $k_1 \oplus k_2$ are positive definite kernels on $(X_1 \times X_2) \times (X_1 \times X_2)$.*

These closure properties allow us to derive new kernel functions in sophisticated ways. For example, a polynomial function of the form

$$\phi(t) = \sum_{i=0}^{\infty} \alpha_i t^i \qquad \alpha_i \geq 0$$

when applied to a kernel functions also gives us a kernel function as the operations only involve point-wise product, positive scaling and addition of kernels. Furthermore, if look at Taylor series expansion of the exponential function $\phi(t) = e^t$

$$\phi(t) = e^t = 1 + \frac{t}{1} + \frac{t^2}{2!} + \frac{t^3}{3!} + \ldots, \tag{2.8}$$

We observe that $\alpha_i = \frac{1}{i!}$. Hence $\phi(t) = e^t$ when applied to a positive definite kernel results in a positive definite kernel.

2.3.5 Reproducing Kernel Hilbert Spaces

Definition 3. *Reproducing kernel Hilbert Spaces [34] Let X be a arbitrary set and $\mathcal{H}$ a Hilbert space of functions on X. $L_x(f)$ is a linear functional that evaluates any function f at x, i.e. $L_x \colon f \mapsto f(x)\ \forall f \in \mathcal{H}$. Then $\mathcal{H}$ is a reproducing kernel Hilbert space, if for all $x \in X$ L_x is a bounded operator on $\mathcal{H}$, that is, there exists some $M > 0$ such that*

$$L_x(f) := f(x) \leq M \left\| f \right\|_{\mathcal{H}} \qquad \forall f \in \mathcal{H}.$$

For bounded linear functionals as above we also the following to be true.

Theorem 4. *Riesz representation theorem [35] If L is a bounded linear functional on a Hilbert*

space $\mathcal{H}$ then there exists some $g \in \mathcal{H}$ such that for every $f \in \mathcal{H}$ we have

$$L(f) = \langle f, g \rangle_{\mathcal{H}} .$$

Moreover, $\|T\| = \|g\|$ (here $\|L\|$ denotes the operator norm of L, while $||g||$ is the Hilbert space norm of g.

From this the following simplified definition of reproducing kernel Hilbert space follows.

Definition 5. *Reproducing kernel Hilbert Spaces [34] $k(\cdot, \cdot)$ is a reproducing kernel of a Hilbert space $\mathcal{H}$ if $f \in \mathcal{H}$, $f(x) = \langle k(x, \cdot), f(\cdot) \rangle$.*

So a RKHS is a Hilbert space of functions with all evaluation functionals (L) that are bounded and linear. Equivalently, a RKHS is a Hilbert space $\mathcal{H}$ with a reproducing kernel whose span is dense in $\mathcal{H}$. From a another view, the Moore-Aronszajn theorem below guarantees existence of a feature space E for any chosen positive definite kernel function k.

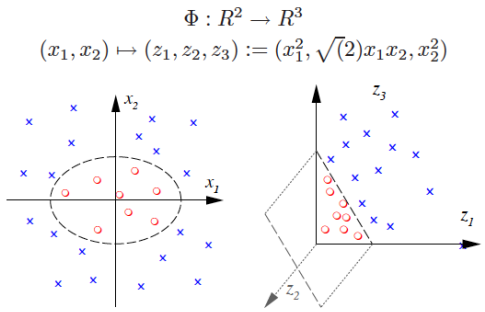


Figure 2.4: High dimensional feature mapping.

Theorem 6. *Moore-Aronszajn-Theorem [34] Suppose $k(\cdot, \cdot)$ is a symmetric, positive definite kernel on a set E. Then there is a unique Hilbert space of functions on E for which $k(\cdot, \cdot)$ is a reproducing kernel.*

RKHS theory also gives a representation of kernel in terms of eigenfunctions and nonnegative eigenvalues of the corresponding integral operator T_k, which is compact, positive and self-adjoint [33].

Theorem 7. *[35] Suppose $k(\cdot, \cdot)$ is a continuous symmetric non-negative definite kernel. Then there is an orthonormal basis $\{e_i\}_i$ of $L_2[a, b]$ consisting of eigenfunctions of T_k such that the*

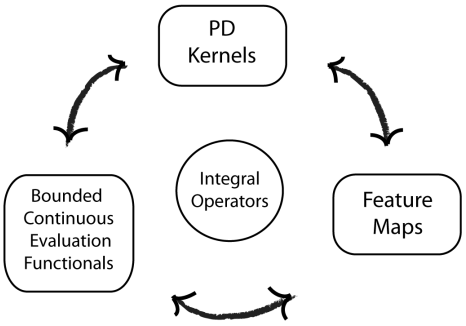


Figure 2.5: Equivalence between RKHS, kernels, linear functionals and integral operators.

corresponding sequence of eigenvalues $\{\lambda_i\}_i$ *is nonnegative. The eigenfunctions corresponding to non-zero eigenvalues are continuous on* $[a, b]$ *and* $k(\cdot, \cdot)$ *has the representation*

$$k(s,t) = \sum_{j=1}^{\infty} \lambda_j e_j(s) e_j(t)$$

where the convergence is absolute and uniform and T_k *is the integral operator,*

$$[T_k \circ \phi](x) = \int_a^b k(x,s)\phi(s)ds$$

and, $\phi \in L^2[a, b]$.

To summarize the above three perspectives of RKHS spaces (Figure 2.5). The following notions are equivalent:

- $k(\mathbf{x}_i, \mathbf{x}_j)$ represents an inner product and $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{y}_j) \rangle$.
- Matrix $[\mathbf{K}]_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$ is a *Gram matrix, such that* $\mathbf{z}^T \mathbf{K} \mathbf{z} \geq 0, \forall \mathbf{z} \in \mathbb{R}^n$.
- $\{\Phi(\mathbf{x}_i)\}_{i=1}^n$ *form the basis of a unique Hilbert Space (RKHS).*

This gives us a comprehensive theory of kernel feature spaces and allows use to define feature spaces by either defining positive definite kernel functions or positive definite Gram matrices.

2.4 Kernel Methods for Learning

In practice when kernel methods are used, we choose a positive definite kernel function $k(\cdot,\cdot)$. This implicitly maps input $x \in X$ into a high dimensional RKHS space (Section 2.3.5) for some $\Phi\colon X \to \mathcal{H}$. The objective is then to define a problem-specific loss function of the form below.

$$L(h) = \underbrace{\sum_{i=1}^{m} V(\mathbf{y}_i, h(\mathbf{x}_i))}_{Loss\ part} + \underbrace{\Omega(||h||^2)}_{regularization} \tag{2.9}$$

A general form for the solution of equation 2.9 is then given by the representer theorem below.

Theorem 8. *Representer Theorem [36, 37] If we let $\mathcal{H}$ be defined by, $k(\cdot,\cdot)$ then $V : X \times Y \times \mathcal{H} \to \mathbb{R}$ be an arbitrary loss functional $\Omega : [0,\infty] \to \mathbb{R}^+$ be a non-decreasing function each minimizer of $h \in \mathcal{H}$ of Eqn 2.9 admits a representation of the form,*

$$h(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i k(\boldsymbol{x}_i, \boldsymbol{x}) \tag{2.10}$$

If loss function V is convex, as is the case for SVM (Section 2.3.1) and SVR (Section 2.3.2), then $L(h)$ is convex, giving us a globally optimal solution. The choice of algorithm used to minimize equation 2.9 depends on the structure of the loss function used.

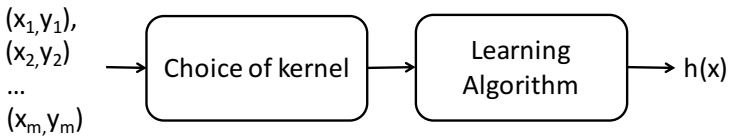


Figure 2.6: Kernel methods in machine learning.

The strategy for using kernel methods looks as follows (Figure 2.6):

1. Choose a suitable kernel function k (Section 2.4.1).

2. Define a suitable problem-specific loss function (Equation 2.9) over training data.

3. Minimize loss equation 2.9 using an appropriate optimization algorithm.

Kernels play two key roles in learning; first, they provide a basis for expansion of the learned funtion (Theorem 8), and second, kernels encode similarity between objects. In many domain specific tasks, hand-crafted kernels which incorporates prior knowledge are used, or else, fixed kernels with parameters chosen heuristically or by cross validation. Performance of learning algorithm critically depends on the choice of the kernel function and parameters used.

2.4.1 Examples of Kernels

Kernels can also be defined over various domains such as strings, graphs and probability distributions [12, 38, 39]. The choice of the kernel typically depends on the particular domain and prior knowledge about the prediction task. A few examples of kernels used in the literature are described below.

Kernel	Expression
Linear	$\langle \mathbf{x}, \mathbf{x}' \rangle$
Gaussian	$e^{\left(-\frac{\|\|\mathbf{x}-\mathbf{x}'\|\|^2}{\sigma^2}\right)}$
Polynomial	$\left(1 + \frac{<\mathbf{x},\mathbf{x}'>}{\sigma}\right)^d$
Exponential	$\exp\left(\sigma \langle \mathbf{x}, \mathbf{x}' \rangle\right)$
Spectral Mixture	$k_{SM}(\tau) = \sum_{q=1}^{Q} w_q \cos\left(2\pi\tau^T \mu_q\right) \prod_{p=1}^{P} \exp\left(-2\pi^2 \tau_p^2 v_q^{(p)}\right)$

Table 2.1: Examples of kernel functions.

2.4.2 Universal and Characteristic Kernels

Universal kernels were introduced in the context of achieving Bayes risk in kernel-based classification and regression methods [40]. It is also know that universality is required for consistency of kernel-based algorithms [41, 42, 43]. They are defined as follows:

Definition 9. *Universal Kernels, [44]: A continuous positive definite kernel k defined on X is called universal if the RKHS, $\mathcal{H}$ induced by k is dense in $C(X)$ with respect to uniform*

norm, that is, for every function $g \in C(X)$ *and all* $\epsilon > 0$*, there exists an* $f \in \mathcal{H}$ *such that* $\|f - g\|_u \leq \epsilon$.

The definition above says if a learning algorithm is consistent, that is, for any target function f^* the kernel based learning algorithm is able to converge to an appropriate f in $\mathcal{H}$, then whether whether f approximates any f^* depends on how rich RKHS $\mathcal{H}$ is. So if kernel k is dense in $C(X)$, then RKHS $\mathcal{H}$ is rich enough to approximate any continuous function from $C(X)$. Such a RKHS is called as universal RKHS and the kernel is called a universal kernel.

Moreover,

Theorem 10. *[44] Assume,*

$$\phi(t) = \sum_{i=0}^{\infty} \alpha_i t^i$$

with $\alpha_i \geq 0$ *and* $\phi(t)$ *is convergent for all* t*. Let* k *be a positive definite kernel on* X*. Then* $k'(x, x') = \phi(k(x, x'))$ *for all* $x, x' \in X$ *is universal if all* $\{\alpha_i | i \in \mathbb{Z}_+\}$ *are positive.*

The above statement gives us conditions for universality of a kernel k' in terms of Taylor series coefficients of the applied function ϕ.

Characteristic kernels have been used for injective embedding of distributions in RKHS spaces. They provide a one-to-one mapping between distributions and vectors in RKHS spaces. This allows us to measure dependency between two distributions by measuring distance between their corresponding RKHS vector representations. For characteristic kernels two distributions are equal if and only if their RKHS vectors coincide (Section 2.6.1). They are defined as follows:

Definition 11. *Characteristic Kernels, [45]: A bounded measurable kernel,* k *is said to be characteristic if the map* $p \mapsto \int_{x \in X} k(\cdot, x) dp(x)$ *is injective, where* $p(x)$ *is a probability measure on* X.

It was shown by Gretton et al. [46] that, if a kernel is universal then it is characteristic, but the converse is not true.

2.5 Kernel Methods for Structured Prediction

In supervised structured prediction the goal is to learn a prediction function $f : \mathcal{X} \rightarrow \mathcal{Y}$, from an input domain $\mathcal{X}$ to an output domain $\mathcal{Y}$. As an example, in articulated human pose estimation, input $x \in \mathcal{X}$ would be an image of a human performing an action, and output would be the a interdependent vector of joint positions (x, y, z). Typically, the space of functions $\mathcal{H}$ is fixed and parametrized we need to estimate these parameters given set of training examples $S = \{(x_1, y_1), (x_2, x_2), \ldots, (x_m, y_m)\} \subseteq \mathcal{X} \times \mathcal{Y}$, drawn i.i.d. from $P(x, y)$. We formulate a meaningful structured loss $\mathcal{L} : \mathcal{Y} \times \mathcal{Y}$ [20, 47, 48, 49]. During prediction for a input $x^* \in \mathcal{X}$ we search for best possible label y^* so that loss $\mathcal{L}(f(x^*), y)$ is minimized over all training data and for all possible labels $y \in \mathcal{Y}$

$$y^* = f(x^*) = \arg\min_{y \in \mathcal{Y}} \mathcal{L}(f(x^*), y)$$

In case of kernel methods for structured prediction [2, 15, 16, 20, 47, 48] the space of functions $\mathcal{H}$ is specified by positive definite kernel functions, which further are jointly defined on the input and output space as $h((x, y), (x', y'))$. In the most common case this kernel is factorized over input and output as $k(x, x')$ and $g(y, y')$, respectively. These individual kernels jointly map the arguments to reproducing kernel Hilbert space (RKHS) or a kernel feature spaces. They are denoted by $\mathcal{K}$ and $\mathcal{G}$. It is well known that performance of kernel algorithms critically depends on the choice of kernel functions ($k(x, x')$ and $g(y, y')$), and learning them is challenging.

2.5.1 Kernel Dependency Estimation

Kernel Dependency Estimation (KDE) of Weston et al. [2] was one of the earliest and general structured prediction algorithm which could handle dependencies in both input and output. The goal is model dependency, $f(\mathbf{x}; \boldsymbol{\alpha}) : \mathcal{X} \rightarrow \mathcal{Y}$, between input $\mathbf{x} \in \mathcal{X}$ and output $\mathbf{y} \in \mathcal{Y}$, given training data $\{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots (\mathbf{x}_n, \mathbf{y}_n)\} \supseteq \mathcal{X} \times \mathcal{Y}$, so as to minimize expected risk

$$R(\boldsymbol{\alpha}) = \int_{\mathcal{X} \times \mathcal{Y}} L(\mathbf{y}, f(\mathbf{y}; \boldsymbol{\alpha})) dP(\mathbf{x}, \mathbf{y}).$$

over distribution true $P(\mathbf{x}, \mathbf{y})$. The function $L(\mathbf{y}, \mathbf{y}')$ is a loss function measuring distance between $\mathbf{y}$ and $\mathbf{y}'$. The choice of loss function related to the similarity function (kernel function) used on output data. Table 2.2 shows different dependency estimation problems for different loss functions and their corresponding output kernels.

Learning Problem	$L(\mathbf{y}, \mathbf{y}')$	Kernel function $g(y, y')$
Multi-class Classification	$\begin{cases} 0 & \mathbf{y} = \mathbf{y}'. \\ 1 & \text{otherwise.} \end{cases}$	$\frac{1}{2}\left(1 - L(\mathbf{y}, \mathbf{y}')\right)$
Multi-Output Regression	$\langle \mathbf{y}, \mathbf{y} \rangle + \langle \mathbf{y}', \mathbf{y}' \rangle - 2 \langle \mathbf{y}, \mathbf{y}' \rangle$	$\langle \mathbf{y}, \mathbf{y}' \rangle$
General Dependency Estimation	$g(\mathbf{y}, \mathbf{y}) + g(\mathbf{y}', \mathbf{y}') - 2g(\mathbf{y}, \mathbf{y}')$	$g(\mathbf{y}, \mathbf{y}')$

Table 2.2: Structured prediction with different loss functions $L(\mathbf{y}, \mathbf{y}')$, and their corresponding output kernels $g(y, y')$.

The prediction algorithm proceeds in steps as follows,

1. **Output decomposition**: Given kernel matrix $\mathbf{G}$ perform kernel principal component analysis on double-centered matrix $\widetilde{\mathbf{G}} = \mathbf{J}_n \mathbf{G} \mathbf{J}_n$ to obtain low dimensional feature vector $\hat{\mathbf{y}} = [\langle \mathbf{v}_1, g(\mathbf{y}, \cdot) \rangle, \langle \mathbf{v}_2, g(\mathbf{y}, \cdot) \rangle, \ldots, \langle \mathbf{v}_p, g(\mathbf{y}, \cdot) \rangle]^T$ where $\mathbf{v}_1, \mathbf{v}_2, \ldots \mathbf{v}_p$ are the principal component vectors.

2. **Map learning**: Learn p independent maps $f_i \colon \mathcal{X} \to \langle \mathbf{v}_i, k(\mathbf{y}, \cdot) \rangle$ using kernel ridge regression such that

$$f_i(\mathbf{x}) = \sum_{j=1}^{n} \beta_j^i k(\mathbf{x}, \mathbf{x}_j), \qquad \boldsymbol{\beta}^i = (\mathbf{K} + \gamma \mathbf{I})^{-1} \hat{\mathbf{y}}_i,$$

   and $\hat{\mathbf{y}}_i = [\langle \mathbf{v}_i, g(\mathbf{y}_1, \cdot) \rangle, \langle \mathbf{v}_i, g(\mathbf{y}_2, \cdot) \rangle, \ldots, \langle \mathbf{v}_i, g(\mathbf{y}_n, \cdot) \rangle]^T$.

3. **Pre-image problem**: For a new test example $\mathbf{x}^*$, we have to solve the minimization problem.

$$\mathbf{y} = \arg\min_{\mathbf{y} \in \mathcal{Y}} \left\| [f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_p(\mathbf{x})]^T - \hat{\mathbf{y}} \right\|^2$$

It is to be noted that, for the case of vectorial input, and one dimensional output with linear kernel, KDE is equivalent to ridge regression [50]. Also, for classification the above method is related to kernel Fisher Discriminant Analysis [50].

2.5.2 Twin Gaussian Processes

Twin Gaussian Processes (TGP) of Bo and Sminchisescu [20] are a recent and popular form of structured prediction methods, which model input-output domains using Gaussian processes with covariance functions, represented by $\mathbf{K}$ and $\mathbf{G}$. These covariance matrices encode prior knowledge about the underlying process that is being modeled. In TGP choice of the auxiliary evaluation function is typically some form of information measure, e.g. KL-Divergence or HSIC, which are known to be special cases of Bregman divergences [51]. KL-Divergence is an asymmetric measure of information, while HSIC is symmetric in its arguments. We refer to these two versions of Twin Gaussian Processes corresponding to each of these measures of information as, TGP with KL-Divergence or simply TGP, and TGP with HSIC.

TGP with KL-Divergence: In this version of TGP, we minimize the KL-divergence between the kernels, $\mathbf{K}$ and $\mathbf{G}$, given the training data $X \times Y$, and a test example x^*. The prediction function for TGP is give by

$$y^* = \arg\min_y D_{KL}(\boldsymbol{G}_{Y \cup y} || \boldsymbol{K}_{X \cup x^*}) \tag{2.11}$$

TGP with HSIC: For this version of TGP with HSIC criteria, the prediction function maximizes the HSIC between the kernels $\mathbf{K}$ and $\mathbf{G}$ given the training data, and test example x^*. The prediction function is as follows

$$y^* = \arg\max_y \overline{HSIC}(\boldsymbol{G}_{Y \cup y}, \boldsymbol{K}_{X \cup x^*}) \tag{2.12}$$

2.6 Dependency Measures

Below we describe three measures nonlinear measures of dependency for structured data and also describe relations between them.

2.6.1 Hilbert Schmidt Independence Criterion

To measure cross-correlation or dependence between structured input and output data in kernel feature Gretton et al. [13] proposed Hilbert Schmidt Independence criterion (HSIC). Given

i.i.d. sampled input-output data $\{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\} \sim P(x, y)$ HSIC measures the dependence between random variables x and y.

Definition 12. *If we have two RKHS's $\mathcal{K}$ and $\mathcal{G}$, then a measure of statistical dependence between $\mathcal{X}$ and $\mathcal{Y}$ is given by the norm of the cross-covariance operator $\mathbf{M}_{xy} : \mathcal{G} \to \mathcal{K}$, which is defined as,*

$$\mathbf{M}_{xy} := \mathbf{E}_{x,y}\left[(k(x,\cdot) - \mu_x) \otimes (g(y,\cdot) - \mu_y)\right] \tag{2.13}$$

$$= \mathbf{E}_{x,y}\left[(k(x,\cdot) \otimes g(y,\cdot))\right] - \mu_x \otimes \mu_y \tag{2.14}$$

and it's measure is given by the Hilbert-Schmidt norm of $\mathbf{M}_{xy}$ which is,

$$HSIC(p_{xy}, \mathcal{K}, \mathcal{G}) := ||\mathbf{M}_{xy}||^2_{HS} \tag{2.15}$$

So the larger the above norm, the higher the statistical dependence between x and y. The advantages of using HSIC for measuring statistical dependence, as stated in Gretton et al. [13] are as follows: first, it has good uniform convergence guarantees; second, it has low bias even in high dimensions; and third a number of algorithms can be viewed as maximizing HSIC subject to constraints on the labels/outputs.

Empirically, in terms of kernel matrices it is defined as,

Definition 13. *Let $Z := \{(x_1, y_1), \ldots (x_m, y_m)\} \subseteq X \times Y$ be a series of m independent observations drawn from p_{xy}. An unbiased estimator of $HSIC(Z, \mathcal{K}, \mathcal{G})$ is given by,*

$$HSIC(Z, \mathcal{K}, \mathcal{G}) = (m-1)^{-2} trace(\mathbf{KHGH}) \tag{2.16}$$

where $\mathbf{K}, \mathbf{H}, \mathbf{G} \in \mathbb{R}^{m \times m}$, $[\mathbf{K}]_{i,j} := k(x_i, x_j)$, $[\mathbf{G}]_{i,j} := g(y_i, y_j)$ and $[\mathbf{H}]_{i,j} := \delta_{ij} - m^{-1}$

For well defined (bounded) and normalized[1] kernels, $\mathbf{K}$ and $\mathbf{G}$. We have $HSIC(Z, \mathcal{K}, \mathcal{G}) \in [0, 1]$. For the ease of discussion we denote $HSIC(Z, \mathcal{K}, \mathcal{G})$ by $\overline{HSIC}(\mathbf{K}, \mathbf{G})$.

[1]Normalized kernel matrix corresponds to feature normalization in kernel feature space, and is obtained by pre and post-multiplying the by matrix $\mathbf{D}^{-\frac{1}{2}}$ where $\mathbf{D}$ is a diagonal matrix with $[\mathbf{D}]_{i,i} = [\mathbf{K}]_{i,i}$, $\mathbf{D}^{-\frac{1}{2}}\mathbf{K}\mathbf{D}^{-\frac{1}{2}}$.

2.6.2 Maximum Mean Discrepancy

Gretton et al. [52] propose a kernel distance between distributions which can be used way of measuring statistical dependency between random variables. It is measured as the maximum difference between the expectation over functions in the unit ball of a reproducing kernel Hilbert space (RKHS), it is called maximum mean discrepancy (MMD). It is defined as follows:

Definition 14. *Mean Maximum Discrepancy Gretton et al. [52] Let $\mathcal{F}$ be a class of functions $f\colon \mathcal{X} \to \mathbb{R}$ in a unit ball of a universal RKHS, and let p and q be distributions over random variables $\mathbf{x}$ and $\mathbf{y}$. We define the maximum mean discrepancy (MMD) as*

$$MMD(p, q; \mathcal{F}) \colon = \sup_{f \in F} \left[\mathbf{E}_{\mathbf{x}} \left[f(\mathbf{x}) \right] - \mathbf{E}_{\mathbf{y}} \left[f(\mathbf{y}) \right] \right].$$

For measurable and bounded kernel $k(\cdot, \cdot)$ we have

$$MMD^2(p, q; \mathcal{F}) = \mathbf{E}_{\mathbf{x},\mathbf{x}'} \left[k(\mathbf{x}, \mathbf{x}') \right] - 2\mathbf{E}_{\mathbf{x},\mathbf{y}} \left[k(\mathbf{x}, \mathbf{y}) \right] + \mathbf{E}_{\mathbf{y},\mathbf{y}'} \left[k(\mathbf{y}, \mathbf{y}') \right]$$

For characteristic kernel like Gaussian and Laplace kernels $MMD(p, q; \mathcal{F}) = 0$ if and only if $p = q$ and it is also a metric on distributions.

An empirical estimate of mean maximum discrepancy is given by

Definition 15. *Sample Mean Maximum Discrepancy Gretton et al. [52] Given observations $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ and $Y = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n\}$ independently and identically distributed (i.i.d.) from p and q, respectively. We define the sample maximum mean discrepancy (MMD) as*

$$MMD(p, q; \mathcal{F}) \colon = \sup_{f \in F} \left[\frac{1}{m} \sum_{i=1}^{m} f(\mathbf{x}_i) - \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{y}_i) \right].$$

The biased estimate of MMD in terms of kernel is given by

$$\begin{aligned} MMD_b^2(X, Y; \mathcal{F}) = & \frac{1}{m^2} \sum_{i=1}^{m} \sum_{j=1}^{m} k(\mathbf{x}_i, \mathbf{x}_j) \\ & + \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} k(\mathbf{y}_i, \mathbf{y}_j) - \frac{2}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} k(\mathbf{x}_i, \mathbf{y}_j) \end{aligned}$$

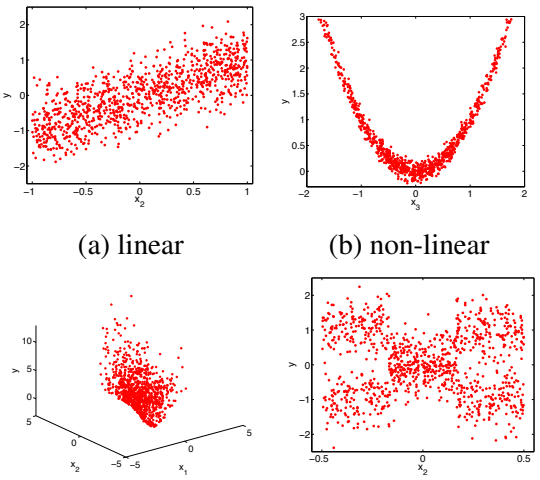
To test for statistical independence between random variables $\mathbf{x}$ and $\mathbf{y}$, let p_x, p_y and p_z be distributions over X, Y and $Z = X \times Y$, respectively. If $\mathcal{K}$ and $\mathcal{G}$ correspond to RKHS's defined using characteristic kernels on X and Y, respectively, then from Sejdinovic et al. [53] we have

$$MMD_b^2(p_z, p_x p_y; \mathcal{F}) = HSIC(p_z, \mathcal{K}, \mathcal{G}).$$

Hence random variables $\mathbf{x}$ and $\mathbf{y}$ are independent if and only if $MMD_b^2(p_z, p_x p_y; \mathcal{F}) = HSIC(p_z, \mathcal{K}, \mathcal{G}) = 0$.

2.6.3 Statistical Distance Correlation

Pearson correlation measures linear correlation between random variables (Figure 2.7a) but fails to measure non-linear dependencies between variables as shown in Figures 2.7b, 2.7c and 2.7d. Distance Correlation introduced by Szekely et al. [54] and Székely et al. [55], Székely and Rizzo [56, 57] is a measure nonlinear dependencies between random vectors of arbitrary dimensions, and is zero if and only if random variables $\mathbf{x}$ and $\mathbf{y}$ are independent. We describe below α-distance covariance which is an extended version of standard distance covariance for $\alpha = 1$.



(a) linear (b) non-linear

(c) non-linear (d) non-linear

Figure 2.7: Synthetic data with linear (Figure 2.7a) and non-linear (Figures 2.7b to 2.7c) correlations from Yamada et al. [8].

Definition 16. *Distance Covariance [14], α-dCov: Distance covariance between random variables $\mathbf{x} \in \mathbb{R}^d$ and $\mathbf{y} \in \mathbb{R}^m$ with finite first moments is a nonnegative number given by*

$$\nu^2(\mathbf{x}, \mathbf{y}) = \int_{\mathbb{R}^{d+m}} |f_{\mathbf{x},\mathbf{y}}(t, s) - f_{\mathbf{x}}(t) f_{\mathbf{y}}(s)|^2 w(t, s) dt ds$$

where $f_{\mathbf{x}}, f_{\mathbf{y}}$ *are characteristic functions of* $\mathbf{x}, \mathbf{y}$*,* $f_{\mathbf{x},\mathbf{y}}$ *is the joint characteristic function, and* $w(t,s)$ *is a weight function defined as* $w(t,s) = (C(p,\alpha)C(q,\alpha)|t|_p^{\alpha+p}|s|_q^{\alpha+q})^{-1}$ *with* $C(d,\alpha) = \frac{2\pi^{d/2}\Gamma(1-\alpha/2)}{\alpha 2^{\alpha}\Gamma((\alpha+d)/2)}$.

From above definition of distance covariance, we have the following expression for Distance Correlation.

Definition 17. *Distance Correlation [14] (*α*-dCorr): The squared Distance Correlation between random variables* $\mathbf{x} \in \mathbb{R}^d$ *and* $\mathbf{y} \in \mathbb{R}^m$ *with finite first moments is a nonnegative number defined as*

$$\rho^2(\mathbf{x},\mathbf{y}) = \begin{cases} \frac{\nu^2(\mathbf{x},\mathbf{y})}{\sqrt{\nu^2(\mathbf{x},\mathbf{x})\nu^2(\mathbf{y},\mathbf{y})}}, & \nu^2(\mathbf{x},\mathbf{x})\nu^2(\mathbf{y},\mathbf{y}) > 0. \\ 0, & \nu^2(\mathbf{x},\mathbf{x})\nu^2(\mathbf{y},\mathbf{y}) = 0. \end{cases}$$

Distance Correlation defined above has the following interesting properties; 1) $\rho^2(\mathbf{x},\mathbf{x})$ is defined for arbitrary dimensions of $\mathbf{x}$ and $\mathbf{y}$, 2) $\rho^2(\mathbf{x},\mathbf{y}) = 0$ if and only if $\mathbf{x}$ and $\mathbf{y}$ are independent, and 3) $\rho^2(\mathbf{x},\mathbf{y})$ satisfies the relation $0 \leq \rho^2(\mathbf{x},\mathbf{y}) \leq 1$. In our work, we use α-Distance Covariance with $\alpha = 2$ and in the following paper for simplicity just refer to it as Distance Correlation.

Now we define sample version of distance covariance given samples $\{(\mathbf{x}_k,\mathbf{y}_k)|k = 1,2,\ldots,n\}$ sampled i.i.d. from joint distribution of random vectors $\mathbf{x} \in \mathbb{R}^d$ and $\mathbf{y} \in \mathbb{R}^m$. To do so, we define two squared Euclidean distance matrices $\mathbf{E_X}$ and $\mathbf{E_Y}$, where each entry $[\mathbf{E_X}]_{k,l} = \|\mathbf{x}_k - \mathbf{x}_l\|^2$ and $[\mathbf{E_Y}]_{k,l} = \|\mathbf{y}_k - \mathbf{y}_l\|^2$ with $k,l \in \{1,2,\ldots,n\}$. These squared distance matrices are when double-centered, by making their row and column sums zero, and are denoted as $\widehat{\mathbf{E}}_\mathbf{X}, \widehat{\mathbf{Q}}_\mathbf{X}$, respectively. So given a double-centering matrix $\mathbf{J} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T$, we have $\widehat{\mathbf{E}}_\mathbf{X} = \mathbf{J}\mathbf{E}_\mathbf{X}\mathbf{J}$ and $\widehat{\mathbf{E}}_\mathbf{Y} = \mathbf{J}\mathbf{E}_\mathbf{Y}\mathbf{J}$. Hence the sample distance correlation (for $\alpha = 2$) is defined as follows.

Definition 18. *Sample Distance Correlation [14]: Given i.i.d samples* $X \times Y = \{(\mathbf{x}_k,\mathbf{y}_k)|k = 1,2,3,\ldots,n\}$ *and corresponding double centered Euclidean distance matrices* $\widehat{\mathbf{E}}_\mathbf{X}$ *and* $\widehat{\mathbf{E}}_\mathbf{Y}$ *the squared sample distance correlation is defined as,*

$$\hat{\nu}^2(\mathbf{X},\mathbf{Y}) = \frac{1}{n^2}\sum_{k,l=1}^{n}[\widehat{\mathbf{E}}_\mathbf{X}]_{k,l}[\widehat{\mathbf{E}}_\mathbf{Y}]_{k,l},$$

and equivalently sample distance correlation is given by

$$\hat{\rho}^2(\mathbf{X},\mathbf{Y}) = \begin{cases} \frac{\hat{\nu}^2(\mathbf{X},\mathbf{Y})}{\sqrt{\hat{\nu}^2(\mathbf{X},\mathbf{X})\hat{\nu}^2(\mathbf{Y},\mathbf{Y})}}, & \hat{\nu}^2(\mathbf{X},\mathbf{X})\hat{\nu}^2(\mathbf{Y},\mathbf{Y}) > 0. \\ 0, & \hat{\nu}^2(\mathbf{X},\mathbf{X})\hat{\nu}^2(\mathbf{Y},\mathbf{Y}) = 0. \end{cases}$$

.

2.6.4 Relation between HSIC and Distance Correlation

Sejdinovic et al. [53] showed that RKHS based dependency measure (HSIC) are a formal extension of Distance Covariance measure when they are restricted to translation invariant kernels (distance inducing kernels).

Definition 19. *Let (X, d_X) and (Y, d_Y) semimetric spaces of negative type (i.e. $d_{\mathbf{x}}$ and $d_{\mathbf{y}}$ are distances induced by a positive semidefinite kernel), and let $X \sim P_{\mathbf{x}}$ and $Y \sim P_{\mathbf{y}}$[2], having joint distributions $P_{\mathbf{xy}}$. The generalized covariance of $\mathbf{x}$ and $\mathbf{y}$ is*

$$\nu^2_{d_{\mathbf{x}},d_{\mathbf{y}}}(\mathbf{x},\mathbf{y}) = \mathbf{E}_{\mathbf{xy}}\mathbf{E}_{\mathbf{x'y'}} d_{\mathbf{x}}(\mathbf{x},\mathbf{x}')d_{\mathbf{y}}(\mathbf{y},\mathbf{y}') + \mathbf{E}_{\mathbf{x}}\mathbf{E}_{\mathbf{x}'} d_{\mathbf{x}}(\mathbf{x},\mathbf{x}')\mathbf{E}_{\mathbf{y}}\mathbf{E}_{\mathbf{y}'} d_{\mathbf{y}}(\mathbf{y},\mathbf{y}') - 2\mathbf{E}_{\mathbf{xy}}\left[\mathbf{E}_{\mathbf{x}'} d_{\mathbf{x}}(\mathbf{x},\mathbf{x}')\mathbf{E}_{\mathbf{y}'} d_{\mathbf{y}}(\mathbf{y},\mathbf{y}')\right].$$

The above definition of distance covariance above is a generalization of distance covariance to general metric spaces we know that distance covariance is an instance of the Hilbert Schmidt Independence Criterion (HSIC).

Theorem 20. *[53] Let (X, d_X) and (Y, d_Y) semimetric spaces of negative type (i.e. $d_{\mathbf{x}}$ and $d_{\mathbf{y}}$ are distances induced by a positive semidefinite kernel), and let $X \sim P_{\mathbf{x}}$ and $Y \sim P_{\mathbf{y}}$[3], having joint distributions $P_{\mathbf{xy}}$. Let k and g be any two kernels on X and Y that generate $d_{\mathbf{x}}$ and $d_{\mathbf{y}}$, repectively, and denote*

$$h((\mathbf{x},\mathbf{x}'),(\mathbf{y},\mathbf{y}')) = k(\mathbf{x},\mathbf{x}')g(\mathbf{y},\mathbf{y}').$$

Then, $\nu^2_{d_{\mathbf{x}},d_{\mathbf{y}}}(\mathbf{x},\mathbf{y}) = 4HSIC(X \times Y, \mathcal{K}, \mathcal{G}) = 4MMD_b^2(p_z, p_x p_y; \mathcal{H})$.

---

[2] *$P_{\mathbf{x}}$ and $P_{\mathbf{y}}$ have finite 2-moment. See Sejdinovic et al. [53] definition 4.*

[3] *$P_{\mathbf{x}}$ and $P_{\mathbf{y}}$ have finite 2-moment. See Sejdinovic et al. [53] definition 4.*

2.7 Summary

In this chapter we have introduced the general problem of supervised learning and structured prediction. We have also introduced several notions of dependency measures (HSIC, MMD, and Distance Correlation) that measure nonlinear dependency for structured data. In our work, we propose to use HSIC and Distance correlation to learn features for structured prediction (Chapter 4) and supervised dimensionality reduction (Chapter 5).

Chapter 3

Polynomials Kernel Transformations

3.1 Introduction

A classical result of Schoenberg [23] states that any continuous, isotropic kernel k is positive definite, if and only if, $k(x, x') = \phi(\langle x, x' \rangle)$ and $\phi(t)$ is a real valued function such that

$$\phi(t) = \sum_{i=0}^{\infty} \alpha_k G_k^{\lambda}(t), \qquad t \in [-1, 1] \tag{3.1}$$

where $\alpha_k \geq 0$, $\sum_{k=0}^{\infty} \alpha_k \geq 0$, $k \in \mathcal{Z}^+$ and $\alpha_k G_k^{\lambda}(1) < \infty$. The symbol $G_k^{\lambda}(t)$ stands for what is known as ultraspherical (Gegenbaur basis) polynomials. Examples of such kernels include Gaussian and Laplacian kernels.

Moreover, a result of Bochner [58] says that a continuous, shift-invariant kernel k is positive definite, if and only if, it is a inverse Fourier transform of a finite non-negative probability measure μ on $\mathbb{R}^d$.

$$k(x - y) = \phi(z) = \int_{\mathbb{R}^d} e^{\sqrt{-1}\langle z, s \rangle} d\mu(s), \qquad x, y, s \in \mathbb{R}^d$$

This results allows us to represent a shift-invariant kernel uniquely as a spectral distribution in a spectral domain. If we take the Fourier transform of equation 3.1 we have a unique representation of positive definite kernel $\phi(t)$, and also a unique representations of positive definite base kernels $G_k^{\lambda}(t)$, $k = 1, 2, \ldots$, in the spectral domain. Hence kernel expansion $\phi(t)$ is a nonnegative mixture of base spectral distributions whose kernels are given by kernels $G_k^{\lambda}(t)$ with nonnegative weights α_k's.

If we look at the monomial basis instead of Gegenbaur basis a similar interpretation can be given for dot product kernels from Smola et al. [59]. Additionally, we know that the span of

monomials form a dense basis in $L_2[-1,1]$ (Weierstrass approximation theorem [60]) so any continuous function on a $[-1,1]$ can be approximated uniformly on that interval by polynomials to any degree of accuracy. Gegenbaur basis is orthonormal and provides additional benefits in interpolation accuracy for functions with sharp changes. Thus avoiding the so called Gibbs phenomenon [61].

3.2 Kernel Transformations

In our work, we use above expansions of kernel $k'(\cdot,\cdot)$ on features obtained from an initial kernel $k(\cdot,\cdot)$, defined on $\mathcal{X} \times \mathcal{X}$. We refer to $k(\cdot,\cdot)$ as initial kernel, and estimate $\phi(t)$ which when applied to kernel matrix $[\mathbf{K}]_{i,j} = \langle x, y\rangle$ gives us a new kernel matrix $[\mathbf{K}]'_{i,j} = \phi([\mathbf{K}]_{i,j}$. Figure 3.1 illustrates this pictorially.

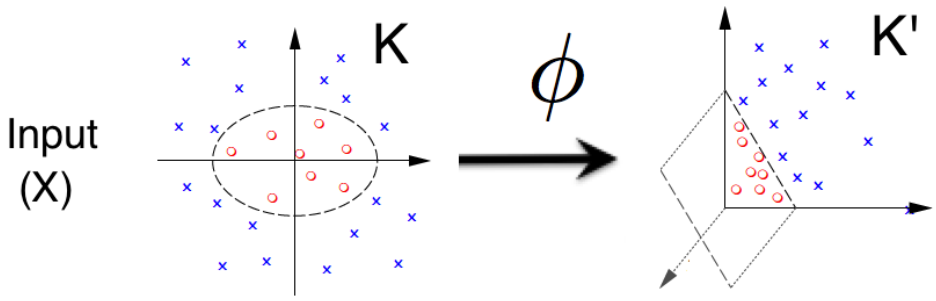


Figure 3.1: Kernel transformations

3.2.1 Monomial Transformations

In case of monomial basis functions we have the following expansion.

Theorem 21 (Schoenberg [23]). *For a continuous function $\phi\colon [-1,1] \to \mathbb{R}$ and $k(x,y) = \phi(\langle x, y\rangle)$, the kernel matrix $\mathbf{K}'$ defined as $\mathbf{K}' = \phi([\mathbf{K}]_{i,j})$ is positive definite for any positive definite matrix $\mathbf{K}$ if and only if $\phi(\cdot)$ is real entire, and of the form below*

$$k'(\langle x, y\rangle) = \phi(t) = \sum_{i=0}^{\infty} \alpha_i t^i \tag{3.2}$$

with $\alpha_i \geq 0$ for all $i \geq 0$.

So $\phi(t)$ is an infinitely differentiable and all coefficients α_j are non-negative (eg. $\phi(t) = e^t$). A benefit of this monomial expansion is that it is easy to compute and can be efficiently evaluated in parallel.

3.2.2 Gegenbaur Transformations

We obtain Gegenbaur's basis if we insist on having a orthonormal expansion. These expansions are also more general, in the sense that they include a weight function $w(t;\gamma)$, which controls the size of the function space used for representation, controlled by a parameter $\gamma > -1/2$.

To formally state,

Theorem 22. *[23] For a real polynomial $\phi\colon [-1,1] \to \mathbb{R}$ and for any finite $X = \{x_1, x_2, x_3, \ldots\}$ the matrix $[\phi(\langle x_i, x_j\rangle)]_{i,j}$ is positive definite if and only if $\phi(t)$ is a nonnegative linear combination of Gegenbauer's polynomials $G_i^\gamma(t)$, which is,*

$$\phi(t) = \sum_{i=0}^{\infty} \alpha_i G_i^\gamma(t) \tag{3.3}$$

with $\alpha_i \geq 0$, and $\sum_i a_i G_i^\gamma(1) < \infty$.

Definition 23. *Gegenbauer's polynomials are defined as below,*

$$G_0^\gamma(t) = 1, G_1^\gamma(t) = 2\gamma t, \ldots, \tag{3.4}$$

$$G_{i+1}^\gamma(t) = \left(\frac{2(\gamma+i)}{i+1}\right) t G_i^\gamma(t) - \left(\frac{2\gamma+i-1}{i+1}\right) G_{i-1}^\gamma(t) \tag{3.5}$$

As stated earlier G_k^γ and G_l^γ are orthogonal in $[-1,1]$ and all polynomials are orthogonal with respect to the weight function $w(t;\gamma) = (1-t^2)^{(\gamma-1/2)}$ and $\gamma > -1/2$.

$$\int_{-1}^{1} G_k^\gamma(t) G_l^\gamma(t) w(t;\gamma) dt = 0, \quad k \neq l. \tag{3.6}$$

The weight function above defines a weighted inner product on the space of functions with a

norm and a inner product given by

$$\|\phi\|^2 = \int_{-1}^{1} w(t;\gamma)\phi(t)\overline{\phi(t)}dt, \tag{3.7}$$

$$\langle \phi, \psi \rangle = \int_{-1}^{1} w(t;\gamma)\phi(t)\overline{\psi(t)}dt \tag{3.8}$$

This weight function controls the space of functions over which we estimate these polynomial maps. Figure 3.2 shows the weight function obtained by different γ parameter. We observe that γ value controls the behavior of the function at the boundary points $\{ 1, -1 \}$. Having this weight function also improves the the quality of interpolation by avoiding the Gibbs phenomenon as further explained in Gottlieb et al. [61]

.

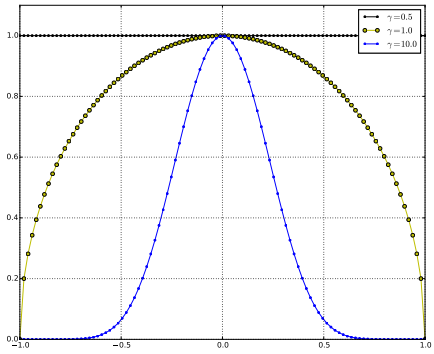


Figure 3.2: Weight function $w(t;\gamma)$ as a function of γ. The γ controls behavior of the weight function at boundaries, and controls the size of interpolation function space.

To summarize given a base kernel matrix $\mathbf{K}$ on input data, we expand target kernel $\mathbf{K}'$ as a transformation $\phi(t)$ applied to $\mathbf{K}$. This maps the initial RKHS features $k(x,\cdot)$ to new RKHS features $k'(x,\cdot)$. These two forms of polynomial mapping representations have been known in literature and have used for non-structured prediction tasks like regression and classification.

3.3 Summary

We have proposed polynomial kernel expansions of kernels that we use to learn kernels for structured prediction. We propose an approach that will use these expansions on both input and output data, so as to maximize dependence between kernel feature spaces using the Hilbert Schmidt Independence Criterion (HSIC).

Chapter 4

Learning Polynomial Kernel Transformations for Structured Prediction

In this chapter we propose a framework to learn kernels for the problem of structured prediction. Outline of this chapter is as follows; first, we propose the use polynomial kernel expansions so as to maximize statistical dependency, second, we propose an efficient matrix-decomposition based algorithm to solve for these expansions, and third, we show state-of-the art results on synthetic and real-world datasets by learning covariance kernels of Twin Gaussian Processes [20] for structured prediction.

4.1 Related Work

In the seminal work of Micchelli and Pontil [62], they showed that we can parametrize a set of kernels in $\mathcal{F}$ over a compact set Ω as

$$\mathcal{F} = \left\{ \int_{\Omega} G_{\omega}(x) d\omega \colon p \in \mathcal{P}(\Omega) \right\},$$

where $\mathcal{P}(\Omega)$ is the set of all probability measures on Ω, $G(\omega)$ is a base kernel parametrized by ω. To illustrate this with an example, if we set $\Omega \subseteq \mathbb{R}_+$ and $G_{\omega}(x)$ as multivariate Gaussian kernel over x, with variance ω, then $\mathcal{F}$ corresponds to a subset of the class of radial kernels.

Most kernel learning frameworks in the past have focussed on learning a single kernel from a family of kernels ($\mathcal{F}$) defined using the above equation. Almost all of these frameworks have focussed on problem of classification or regression. Table 4.1 illustrates previous work on learning kernels for different choices of Ω, and base kernel $G_{\omega}(x, y)$, over data domain $\mathcal{X}$.

Some of the above approaches work iteratively, [63, 64] while others use optimization methods such as, semi-infinite programming [68], or QCQP [67]). A review paper by Suzuki and

Kernel family and base kernel $G_\omega(x, y)$	Related Work
Radial kernels (Iterative), $G_\omega(x, y) = e^{-\omega\|x-y\|^2}$	Argyriou et al. [63, 64].
Dot product kernels, $G_\omega(x, y) = e^{\omega\langle x,y\rangle}$	Argyriou et al. [63, 64].
Finite convex sum of kernels, SimpleMKL	Rakotomamonjy et al. [65].
Radial kernels (Semi-infinite Programing, Infinite Kernel Learning)	Gehler and Nowozin [66]
Shift-invariant kernels (Radial and Anisotropic), $G_\omega(x - y)$	Shirazi et al. [67]

Table 4.1: Kernel learning frameworks which learn kernels as convex combination of base kernels $G_\omega(x, y)$.

Sugiyama [69] surveys many of these works on various Multiple Kernel Learning algorithms. The relevant works which uses results on polynomial expansion of kernels has been of Smola et al. [59] which learn dot product kernels using monomial basis $\{1, x, x^2, \ldots\}$, and learning shift-invariant kernels using radial base kernels of Shirazi et al. [67], both of these have been proposed for classification and regression.

In our work we use results on expansion of kernels for learning kernels for the problem of structured regression. We use monomial and Gegenbaur expansions to learn a positive combination of base kernels. The Gegenbaur basis basis has an advantage of being orthonormal, and providing a control parameter γ, that helps avoid Gibbs phenomenon on interpolation [61].

4.2 Learning Kernel Transformations

The goal in structured prediction is to predict output label $y \in \mathcal{Y}$, given a input example $x \in \mathcal{X}$, our thesis is that if output kernel feature $g'(y, \cdot)$ is more correlated (dependent) with input kernel feature $k'(x, \cdot)$, then we can significantly improve the regression performance. We propose to use HSIC for this. Also, maximizing HSIC (or equivalently Kernel Target Alignment) has been used as an objective for learning kernels in the past [70]. These frameworks maximize the HSIC or alignment between input and outputs, which in our case are structured objects. Cristianini et al. [71] in their work provide generalization bounds and which confirm that maximizing alignment (i.e HSIC) does indeed lead to better generalization.

Following on this, if we let $\mathbf{K}' = \phi(\mathbf{K})$, and $\mathbf{G}' = \psi(\mathbf{G})$, where $\mathbf{K}$ and $\mathbf{G}$ are normalized base kernels on input and output data. Using the empirical definition $\overline{HSIC}$ we define the

following objective function to optimize

$$\mathbf{L}(\boldsymbol{\alpha}*, \boldsymbol{\beta}*) = \max_{\boldsymbol{\alpha},\boldsymbol{\beta}} \overline{HSIC}(\phi(\mathbf{K}), \psi(\mathbf{G})) \tag{4.1}$$

$$\text{subject to } \alpha_i \geq 0, \beta_j \geq 0, \forall i, j \geq 0 \tag{4.2}$$

In case of monomial basis as in section 3.2.1, we can use equation 3.2 on kernel matrices $\mathbf{K}$ and $\mathbf{G}$ we get equations for $\phi : \mathbf{K} \rightarrow \mathbf{K}'$ and $\psi : \mathbf{G} \rightarrow \mathbf{G}'$ as follows,

$$\phi(\mathbf{K}) = \sum_{i=0}^{\infty} \alpha_i \mathbf{K}^{(i)}, \alpha_i \geq 0, \ \forall i \geq 0 \tag{4.3}$$

$$\phi(\mathbf{G}) = \sum_{j=0}^{\infty} \beta_j \mathbf{G}^{(j)}, \beta_j \geq 0, \ \forall j \geq 0 \tag{4.4}$$

where $\mathbf{K}^{(i)}$ is the kernel obtained by applying the i^{th} polynomial basis t^i to the initial kernel matrix $\mathbf{K}$ (similarly $G_k^{\lambda}(t)$ for Gegenbaur basis). Figure 4.1 illustrates this. Assuming our initial kernels are both bounded and normalized we also need our new kernels $\phi(\mathbf{K})$ and $\psi(\mathbf{G})$ to be bounded. For this purpose we impose l_2-norm regularization constraint on α_i's and β_j's, that is $\|\boldsymbol{\alpha}\|^2 = 1$ and $\|\boldsymbol{\beta}\|^2 = 1$. These constraints are similar to the l_2-norm regularization constraint on positive mixture coefficient's in the Multiple Kernel Learning framework of Cortes et al. [70]. These helps generalization to unseen test data and also avoid arbitrary increase of optimization objective so as to maximize it.

Substituting the expansions equations 4.3 and 4.4 in equation 4.2, and simplifying the main objective 4.2 and we get,

$$\text{maximize } \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} \alpha_i \beta_j \mathbf{C}_{i,j} \tag{4.5}$$

$$\text{subject to, } \|\boldsymbol{\alpha}\|_2 = 1, \ \|\boldsymbol{\beta}\|_2 = 1$$

where the $\mathbf{C}$-matrix is such that $[\mathbf{C}]_{i,j} = \overline{HSIC}(\mathbf{K}^{(i)}, \mathbf{G}^{(j)})$. Also, from the properties of HSIC we know that the entries of the $\mathbf{C}$-matrix is a non-negative non-negative with entries.

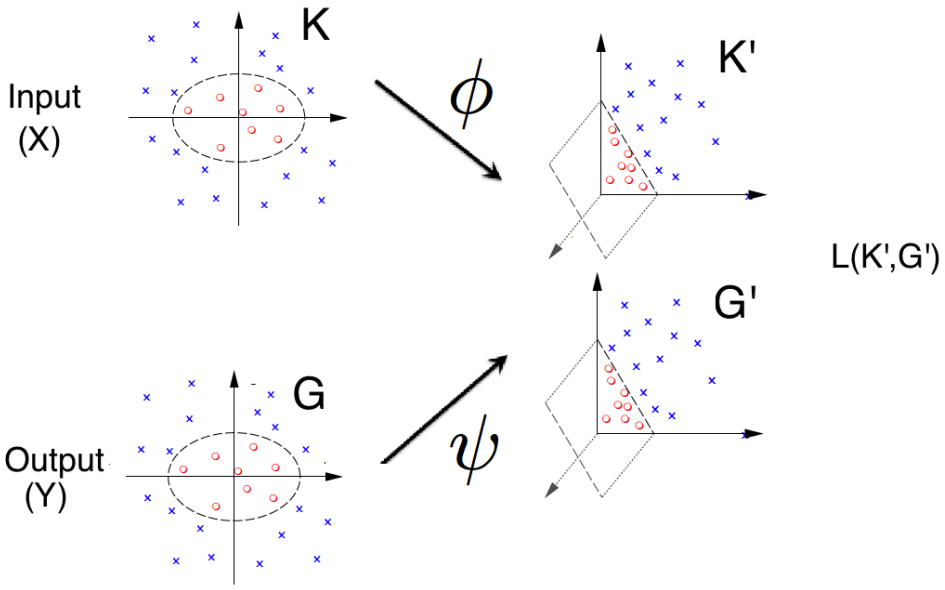


Figure 4.1: Twin Kernel Learning framework.

The constructed $\mathbf{C}$-matrix looks as follows,

$$\mathbf{C} = \begin{array}{c|cccc} & K^{(0)} & K^{(1)} & K^{(i)} & K^{(d_2)} \\ \hline G^{(0)} & C_{1,1} & C_{1,2} & \vdots & C_{d_2,1} \\ G^{(1)} & C_{1,1} & C_{1,2} & \vdots & C_{d_2,1} \\ G^{(0)} & C_{2,1} & C_{1,2} & \vdots & C_{2,d_2} \\ G^{(j)} & \dots & \dots & C_{i,j} & \dots \\ G^{(d_1)} & C_{d_1,1} & C_{d_1,2} & \vdots & C_{d_1,d_2} \end{array} \tag{4.6}$$

where $C_{i,j} = \overline{HSIC}(\mathbf{K}^{(i)}, \mathbf{G}^{(j)})$.

To further explain our intuition, every entry $[\mathbf{C}]_{i,j} = \overline{HSIC}(\mathbf{K}^{(i)}, \mathbf{G}^{(j)})$ represents higher order cross-correlations among the polynomial combination of features of order i and j between input and output, respectively. Hence by appropriately choosing coefficient's α_i and β_j we are maximizing these higher order cross-correlations in the kernel feature space. We approximate sum to finite degrees d_1 and d_2 which leads us to a finite dimensional problem with $deg(\phi) = d_1$ and $deg(\psi) = d_2$. This amounts to using all polynomial combinations of features up to degree d_1 for input and d_2 for output. So higher the degree we choose more is the dependence and

intuitively better is the prediction. The upper bound on the degree is computationally limited by the added non-linearity in the kernel based prediction algorithm and can lead to overfitting. So we choose these degrees empirically by cross-validation until we saturate the performance of the prediction algorithm.

We have the following theorem regarding the solution of optimization problem 4.5,

Theorem 24. *The solution* $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$ *to the optimization problem in 4.5 is given by the first left and right singular vector of the* $\mathbf{C}$*-matrix*

Proof. Using Perron-Frobenius theorem [72] for square non-negative matrices $\mathbf{C}^T\mathbf{C}$ and $\mathbf{C}\mathbf{C}^T$, we claim that both $\mathbf{C}^T\mathbf{C}$ and $\mathbf{C}\mathbf{C}^T$ have Perron vectors $\boldsymbol{\alpha}^*$ and $\boldsymbol{\beta}^*$, respectively. Both $\boldsymbol{\alpha}^*$ and $\boldsymbol{\beta}^*$ are the left and right singular vectors of $\mathbf{C}$ and also maximize Eq. 4.5. □

The above theorem gives us our required solution to the problem. Hence to solve for the unknown's α_i's and β_j's we do Singular Value Decomposition (SVD) of the $\mathbf{C}$-matrix and choose $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ to be the first left and right singular vectors Theorem 24. The non-negativity of the $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ vector is guaranteed due to non-negativity of the $\boldsymbol{C}$-matrix combined with Perron-Frobenius theoremm Chang et al. [72].

We also observe that $\alpha_0 = \beta_0 = 0$, so choosing $d_1 = 1$ corresponds to using the identity mapping $\phi(t) = t$ on the input kernel, this corresponds to using the initial kernel only, or equivalently, no mapping on input kernel. A similar argument also applies if we set $d_2 = 1$, then we have $\psi(t) = t$, and no mapping on output base kernel. We note that if we set $d_2 = 1$ and d_1 to be some arbitrary value greater than one, then solution $\boldsymbol{\alpha}^*$ is exactly where $\alpha_i^* \propto \overline{HSIC}(\mathbf{K}^{(i)}, \mathbf{G})$, which same as choosing coefficients based on kernel alignment as in Cortes et al. [70].

We also like to point out the similarity of our proposed objective to that of Kernel Canonical Correlation Analysis (KCCA) objective, which also uses HSIC [73]. In KCCA we find two nonlinear mappings $\phi(\cdot) \in \mathcal{K}$ and $\psi(\cdot) \in \mathcal{G}$ from their prespecified RKHS's maximizing statistical correlation. In our approach, we also looking for analytical kernel transformations $\phi(\cdot)$ and $\psi(\cdot)$ on initial kernel matrices to maximize the same objective.

4.3 Modified Twin Gaussian Processes

We use both HSIC and KL-Divergence criteria's of Twin Gaussian Processes (Section 2.5.2) and compare against the degree of mapping d_1 and d_2. This allows us to show how each information measure is affected as the mapping degrees d_1 and d_2 are increased. The relationship we observe is straightforward and direct, allowing the choice of d_1 and d_2 to be made easily. We refer to these new modified TGP's as Higher Order TGP with KL-Divergence (HOTGP) and Higher Order HSIC (HOHSIC) for TGP using HSIC.

Modified TGP with KL-Divergence: In this version of TGP we minimize the KL-divergence between the transformed kernels, $\phi(\mathbf{K})$ and $\psi(\mathbf{G})$, given the training data $X \times Y$, and test example x^*. The prediction function for HOTGP is,

$$\mathbf{y}^* = \arg\min_y D_{KL}((\psi(\boldsymbol{G}_{Y \cup y}) || \phi(\boldsymbol{K}_{X \cup x^*})) \tag{4.7}$$

Modified TGP with HSIC: For this version of TGP with HSIC criteria, the prediction function maximizes the HSIC between the transformed kernels $\phi(\mathbf{K})$ and $\psi(\mathbf{G})$ given the training data $X \times Y$, and test example x^*. The prediction function of HOHSIC is as follows,

$$\mathbf{y}^* = \arg\max_y \overline{HSIC}((\psi(\mathbf{G}_{Y \cup y}), \phi(\mathbf{K}_{X \cup x^*})) \tag{4.8}$$

4.4 Experiments

We show empirical results using Twin Gaussian Processes with KL-Divergence and HSIC, using both monomial and Gegenbaur transformations on two synthetic and two real-world datasets. To measure improvement in performance over the baseline we look at empirical reduction in error which we call *% Gain* defined as,

$$\%\ Gain = \left(1 - \frac{Error_{(mapping)}}{Error_{(no\ mapping)}}\right) \times 100.$$

In all of our experiments, we use the Gaussian kernels $k(x_i, x_j) = exp(-\gamma_x||x_i - x_j||^2)$ and $g(y_i, y_j) = exp(-\gamma_y||y_i - y_j||^2)$ as base kernels on input and output, respectively. The bandwidth parameters γ_x and γ_y were chosen using cross-validation using base kernel on the original dataset. The weight parameters were chosen to be fixed values $\lambda_1 = 0.51$ and $\lambda_2 = 0.52$ using rough estimates from expressions in Gottlieb et al. [61] and validated on validation set. For choice of expansion degree we increase d_1 and d_2 until the *% Gain* saturates on the cross-validation. We then learn the kernel transformations $\phi(\cdot)$ and $\psi(\cdot)$ using the proposed approach.

S Shape dataset The S-shape dataset from Bishop and Svensén [74] which is a simple 1D input/output regression problem. In this dataset, 500 values of inputs (x) are sampled uniformly in $(0, 1)$ and then evaluated for $r = x + 0.3sin(2\pi x) + \epsilon$, with ϵ drawn from a zero mean Gaussian noise with standard deviation $\sigma = 0.05$ (Figure 4.2). The goal here is to solve the inverse problem, which is to predict x, given r. The dataset is challenging in the sense that it is multivalued (in the middle of the S-shape), discontinuous (at the boundary of univalued and multivalued region) and noisy ($\epsilon = \mathcal{N}(0, \sigma)$). The error is metric is the mean absolute error (MAE).

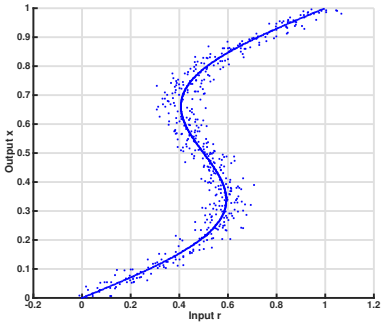


Figure 4.2: S-Shape dataset

4.4.1 Datasets

Synthetic Data

Poser dataset The Poser dataset contains synthetic images of human motion capture sequences from the Poser 7, [75]. The motion sequences includes 8 categories: walk, run, dance, fall, prone, sit, transitions and misc. There are 1927 training examples coming from different sequences of varying lengths and the test set is a continuous sequence of 418 time steps. The input feature vectors are 100d silhouette shape descriptors while the output feature vectors are 54d vectors with the x, y and z rotation of joint angles. The error metric is the mean absolute error (MAE) in mm.

Real-world Data

USPS handwritten digits reconstruction In USPS handwritten digit reconstruction dataset from Weston et al. [2] (Figure 4.3), the goal is to predict 16 pixel values in the center of an image, given the outer pixels. We use 7425 examples for training (without labels) and 2475 examples (roughly 1/4th for each digit) for testing. The error metric here is the reconstruction error measured using mean absolute error (MAE).

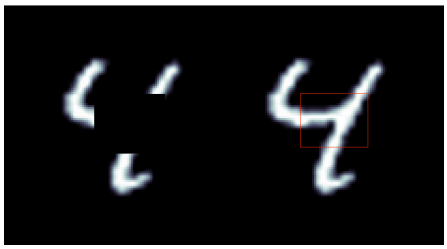


Figure 4.3: USPS digits reconstruction dataset from Weston et al. [2].

HumanEva-I pose dataset The HumanEva-I dataset from Sigal et al. [9] is a challenging dataset that contains real motion capture sequences from three different subjects (S1,S2,S3) performing five different actions (Walking, Jogging, Box, Throw/Catch, Gestures, Figure 4.4). We train models on all subjects and all actions. We have input images from three different cameras; C1,C2 and C3 and we use HoG features from Dalal and Triggs [76] on them. The output vectors are 60d with the x, y, z joint positions in mm. We report results using concatenated features from all three cameras (C1+C2+C2) and also individual features from each individual camera (C1,C2 or C3).

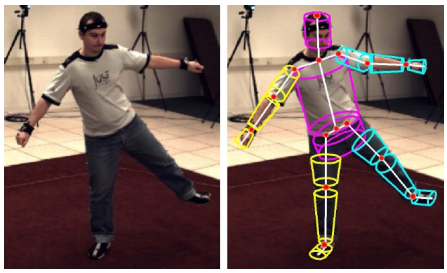
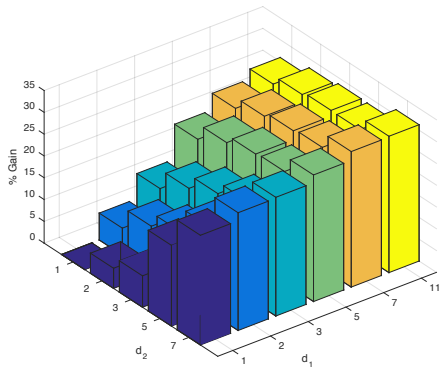


Figure 4.4: HumanEva-I dataset from Sigal et al. [9]

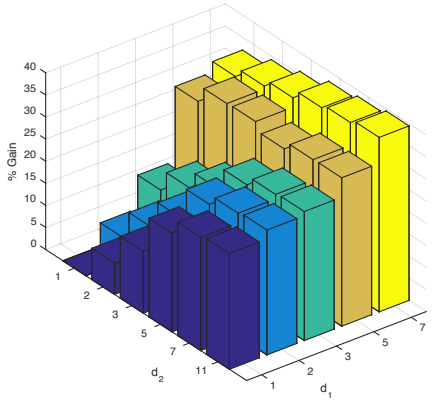
4.4.2 Results

Synthetic Data

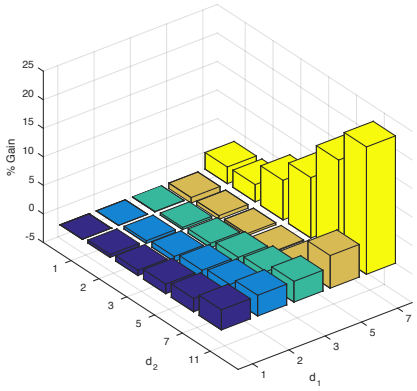
S-Shape data We choose bandwidth parameter to be $\gamma_x = 1$ and $\gamma_y = 1$ using cross-validation. To illustrate effect of increasing degrees d_1 and d_2, we run our results on a grid of degrees from the set $\{1, 2, 3, 5, 7, 11\}$. In figure 4.5 we plot the increase in the mapping degree as % Gain. Figures 4.5a and 4.5b show results for using KL-Divergence as the optimization criteria. Figures 4.5c and 4.5d show results for using HSIC as an the optimization criteria.



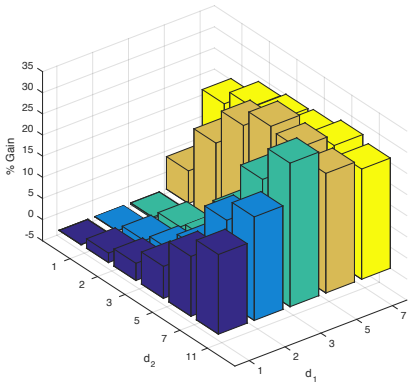
(a) TGP (KL-Div) with Monomial Basis



(b) TGP (KL-Div) with Gegenbaur Basis



(c) TGP (HSIC) with Monomial Basis



(d) TGP (HSIC) with Gegenbaur Basis

Figure 4.5: *% Gain* for TGP with KL-Div (Figures 4.5a and 4.5b) and HSIC (Figures 4.5c and 4.5d) using both monomial and Gegenbaur basis on left and right, respectively.

We observe that for both as we increase d_1 and d_1, % Gain increases i.e. mean absolute error (MAE) reduces. Also for each pair of figures, for each objective, changing from a monomial basis to Gegenbaur basis helps improve % Gain, from *31.27%* to *39.49%* for KL-Div, and *22.31%* to *26.06%* for HSIC.

Crit.	(d_1, d_2)	MAE (w/o map)	MAE (w/map)	Gain %
KL-Div	(1,11)	57.03	41.57	**27.25**%
HSIC	(1,11)	48.08	38.71	**19.48**%
KL-Div (Gegen.)	(1,11)	57.02	44.19	**22.50**%
HSIC (Gegen.)	(1,23)	48.08	35.55	**36.02**%

Table 4.2: Root Mean Absolute Error for Poser dataset for two criteria's of TGP using monomials and Gegenbaur transformations.

Poser dataset For the Poser dataset we choose bandwidth to be $\gamma_x = 10$ and $\gamma_y = 10^{-5}$ using cross-validation. The final results are shown in Table 4.2. In this case we observe that for both basis input degree turns out be $d_1 = 1$, and using Gegenbaur basis gives us better results of *57.02%* at $d_2 = 11$, when compared to monomial basis with *48.08%* at $d_2 = 23$. Here we distinctly observe benefits of using Gegenbaur basis in terms better accuracy and better numerical stability with a lower output degree d_2.

Real-world Data

Handwritten digit reconstruction We report our results with bandwidth parameters $\gamma_x = 2e10^{-7}$ and $\gamma_y = 2e10^{-2}$. The mapping degrees were chosen for HOTGP were $(d_1, d_2) = (11, 11)$ and for HOHSIC were $(d_1, d_2) = (23, 23)$, using cross validation on MAE criteria. Table 4.3 shows summary of results and compares our approach with other kernel-based structured prediction methods. We observe two lowest scores to be from Twin Gaussian process using HSIC with monomial basis, $(d_1, d_2) = (11, 11)$, and KL-Divergence with Gegenbaur basis, and $(d_1, d_2) = (23, 23)$. The % Gain shows that using Gegenbaur basis leads to better same results for lower degree over baseline with no-mapping. The best accuracy is obtained for both objective criteria.

Additionally, we perform experiments using Recurrent Neural Networks (RNN) from Guo et al. [77]. We randomly sampled 1100 images from the original data set, and used the first half of the image (128 pixels) to predict the second half (128 pixels). Table 4.4 presents RMS Errors of the RNN model which is 0.5591 and % Gain over best method (Monomial with KLDiv) is 56.77 %. We use mapping degrees of $d_1 = d_2 = 11$ in all cases.

HumanEva-I pose dataset We report results using concatenated features from all three cameras (C1 +C2+C2), and also features from individual camera (C1,C2 and C3). We use Gaussian kernel with $\gamma_x = \gamma_y = 10^{-4}$. For KL-divergence criteria we get $(d_1, d_2) = (1, 11)$ for monomial basis, and $(d_1, d_2) = (1, 5)$ for Gegenbaur basis. In case of HSIC criteria, we get $(d_1, d_2) = (11, 11)$ for both monomial and Gegenbaur basis. Table 4.5) shows complete set of results. The $\%\ Gain$ for each criteria is shown in bold. We observe in both subtables that using concatenated features (C1+C2+C3) gives us better results than using individual camera

Method	(d_1, d_2)	MAE	Method	(d_1, d_2)	MAE
NN	/	0.341	KRR	/	0.250
SVR	/	0.250	KDE	/	0.260
$SOAR_{krr}$	/	0.233	$SOAR_{svr}$	/	0.230
HSIC	(1,1)	0.3399	HSIC	(11,11)	*0.195* (**2.50%**)
KL-Div	(1,1)	0.2151	KL-Div	(11,11)	0.211 (2.01)%
KL-Div Gegen.	(1,1)	0.2101	KL-Div Gegen.	(11,11)	*0.19* (**7.01%**)
HSIC Gegen.	(1,1)	0.3399	HSIC Gegen.	(23,23)	*0.25442* (25.14%)

Table 4.3: Comparison with others models from Bo and Sminchisescu [1] for USPS digits reconstruction dataset. The two lowest errors are *emphasized* and their % *Gain* **bolded**. NN means nearest neighbor regression, KDE means kernel dependency estimation [2] with 16d latent space obtained by kernel principal component analysis. SOAR means Structured Output Associative regression [1]. % *Gain* shows reduction in error compared to no-mapping vs. using mapping for KL-Div and HSIC criteria with monomial and Gegenbaur basis.

TGP	Baseline	Monomial	Gegenbaur
KLDiv	0.2596	**0.2380 (8.32 %)**	0.2417 (6.90 %)
HSIC	0.3172	0.2649 (16.49 %)	0.3052 (3.78 %)

Table 4.4: RMS Error for USPS digits reconstruction dataset compared with RNN from HNSSO is 0.5591. % Gain over best method (Monomial with KLDiv) is 56.77 %. We use mapping degrees $d_1 = d_2 = 11$ in all cases.

features.

In subtable 4.5a, we see results with monomials basis for both HSIC KL-Divergence criterion. In general we observe KL-Divergence to be giving better results compared to HSIC. The best results for this case is with features (C1+C2+C3) with KL-Divergence and % *Gain* of 5.080%. In subtable 4.5b for the Gegenbaur basis we observe a significant reduction in error when using KL-divergence with % *Gain* of 99.95%. We show consistent improvement in performance with this expansion and much better results using Gegenbaur expansion. We provide detailed results on all subjects and actions in the appendix.

Features	Crit.	w/map	wo/map	% Gain
HoG (C1C2C3)	KL-Div	45.17	42.88	**5.08**%
	HSIC	172.66	172.59	**0.05**%
HoG (C1)	KL-Div	34.29	33.43	**2.51**%
	HSIC	172.66	173.88	**0.08**%
HoG (C2)	KL-Div	31.99	31.58	**1.29**%
	HSIC	172.66	173.88	**0.09**%
HoG (C3)	KL-Div	30.93	30.49	**1.41** %
	HSIC	172.66	172.59	**0.09**%

(a) Monomial transformation: KL-Div - $(d_1, d_2) = (1, 11)$, HSIC-$(d_1, d_2) = (11, 11)$

Features	Crit.	w/map	wo/map	% Gain
HoG (C1C2C3)	KL-Div	25.40	0.011	**99.95** %
	HSIC	172.66	172.29	**0.25**%
HoG (C1)	KL-Div	44.42	9.63	**77.46**%
	HSIC	172.66	172.28	**0.26**%
HoG (C2)	KL-Div	44.68	12.30	**71.71**%
	HSIC	172.66	172.27	**0.26**%
HoG (C3)	KL-Div	44.35	0.01	**99.97**%
	HSIC	172.66	172.28	**0.26**%

(b) Gegenbaur transformation: KL-Div- $(d_1, d_2) = (1, 5)$, HSIC-$(d_1, d_2) = (11, 11)$

Table 4.5: Mean Absolute Error for HumanEva-I dataset for two criteria's, KL-Div and HSIC, with and without mapping, using both monomial and Gegenbaur transformation.

4.5 Discussion

% Gain Criterion	S-shape	Poser	USPS Digits	HumaEva-I (C1+C2+C3)	HumanEva-I (C1,C2,C3)
KL-Div.	31.27 %.	6.39 %	1.97 %	5.08%	(2.51%, 1.29%, 1.40%)
HSIC	22.31 %.	1.26%	2.11 %	0.05%	(0.08%, 0.09%, 0.09%)
KL-Div. (Gegen.)	39.49 %.	14.31%	14.31 %	99.95 %	(77.46%,71.71% 99.97%)
HSIC (Gegen.)	26.06 %.	7.01%	7.01 %	0.25 %	(0.26%,0.26%, 0.26%)

Table 4.6: Summary of all datasets with both criteria, and using both monomial and Gegenbaur transformations.

Table 4.6 provides a complete summary of results for all datasets. It is clear from the experimental results that as we increase mapping degrees d_1 and d_2, we use higher order combination of polynomial kernel features to maximize dependence between input and output, and this leads to better regression. Reduction in prediction error as indicated by the *% Gain* metric. Choice of degree is done using cross-validation and kernel parameters are selected using the kernel median trick. In S-shape dataset increase in both d_1 and d_2 helps until the performance saturates, and later falls off due to numerical instability due to the added non-linearity and overfitting.

For the case of the Poser dataset (Table 4.2) and for HumanEva with KL-Divergence (Table 4.5a and 4.5b), we see that the best performance is for d_1 equal to one. This amounts

to choosing choosing the identity mapping/no-mapping on input, $\phi(t) = t$. As we described in section 4.2 this amounts to choosing coefficients proportional to kernel target alignment of Cortes et al. [70] score between initial input kernel and kernels obtained from basis expansion of initial output kernel $\mathbf{G}$.

The effect of using the two different objective criteria KL-Divergence versus HSIC, we see that in many cases, KL-Divergence does better or as well as HSIC (Table 4.3). In terms of ease of optimization of TGP with HSIC, it turns out to be easier criteria to optimize as there is no explicit training step for it Bo and Sminchisescu [20]), and it is relatively easier to compute objective than KL-Divergence.

In terms of choice of basis functions, it is clear that using Gegenbaur basis leads to better numerical stability and often times better results. In some cases like, HumaEva with KL-Divergence (Table 4.5b), it does lead to using lower degree values which leads to easier optimization in prediction phase.

4.6 Summary

We proposed a novel method for learning kernels by using polynomial expansions in terms of base kernels. We also empirically showed that maximizing statistical dependency (HSIC) between input and output kernel features leads to better performance for structured prediction. We also proposed an efficient, matrix-decomposition based algorithm to learn them and showed state-of-the-art empirical results on several synthetic and real-world datasets by covariance functions of Twin Gaussian Processes.

Chapter 5

Supervised Dimensionality Reduction via Distance Correlation Maximization

In this chapter, we focus on supervised dimensionality reduction in the regression setting, we consider the problem of predicting a univariate response $y_i \in \mathbb{R}$ from a vector of continuous covariates $\mathbf{x}_i \in \mathbb{R}^p$ for $i = 1$ to n. We propose a novel formulation for supervised dimensionality reduction that is based on a dependency criterion called Distance Correlation of Szekely et al. [54].

5.1 Related Work

Sliced Inverse Regression (SIR) of Li [78], Lue [79], Szretter and Yohai [80] is one of the earliest developed supervised dimensionality reduction techniques and is a seminal work that introduced the concept of a central subspace that we now describe. This technique aims to find a subspace given by the column space of a $p \times d$ matrix $\mathbf{B}$ with $d << p$ such that $\mathbf{y} \perp\!\!\!\perp \mathbf{X} | \mathbf{B}^T\mathbf{X}$ where $\perp\!\!\!\perp$ indicates statistical independence. Under mild conditions the intersection of all such dimension reducing subspaces is itself a dimension reducing subspace, and is called the central subspace [81]. SIR aims to estimate this central subspace. Sliced Average Variance Estimation (SAVE) of Shao et al. [82] and Shao et al. [83] is another early method that can be used to estimate the central subspace. SIR uses a sample version of the first conditional moment $\mathbf{E}\left[\mathbf{X} \mid Y\right]$ to construct an estimator of this subspace and SAVE uses the sample first and second conditional moments to estimate it. Likelihood Acquired Directions (LAD) of Cook and Forzani [84] is a technique that obtains the maximum likelihood estimator of the central subspace under assumptions of conditional normality of the predictors given the response. Like LAD, methods SIR and SAVE rely on elliptical distributional assumptions like Gaussianity of the data.

More recent methods that do not require any distributional assumptions on the marginal distribution of $\mathbf{x}$ or on the conditional distribution of y. The authors of Gradient Based Kernel Dimension Reduction (gKDR), Fukumizu and Leng [85], use an equivalent formulation of the conditional independence relation $\mathbf{y} \perp\!\!\!\perp \mathbf{X} | \mathbf{B}^T\mathbf{X}$ using conditional cross-covariance operators and aim to find a $\mathbf{B}$ that maximizes the mutual information $I(\mathbf{B^T X}, \mathbf{y})$. In this work, the authors estimate the conditional cross-covariance operators by using Gaussian kernels. gKDR instead uses kernels only to provide equivalent characterizations of conditional independence using sample estimators of cross-covariance operators.

Sufficient Component Analysis (SCA) of Yamada et al. [86] is another technique where the $\mathbf{B}$ is also learnt using a dependence criterion. SCA aims to maximize the least-squares mutual information given by $SMI(Z, Y) = \frac{1}{2} \int \int (\frac{p_{zy}(z,y)}{p_z(z)p_y(y)} - 1)^2 dzdy$ between the projected features $\mathbf{Z} = \mathbf{B}^T\mathbf{X}$ and the response. This is done under orthonormal constraints over $\mathbf{B}$, and the optimal solution is found by approximating $\frac{p_{zy}(z,y)}{p_z(z)p_y(y)}$ using method of density ratio estimation [87, 88], and also an analytical closed form solution for the minima is obtained. In the work of Suzuki and Sugiyama [89] (LSDR), the authors optimize this objective using a natural gradient based iterative solution on the Steifel manifold $\mathbb{S}_d^m(\mathbb{R})$ via a line search along the geodesic in the direction of the natural gradient [90, 91].

Furthermore, other works of Li et al. [92], Kong et al. [93], Berrendero et al. [94] have used Distance Correlation as a criterion for feature selection in a regression setting, but in our work we show benefits using Distance Correlation as a criterion for supervised dimensionality reduction.

5.2 Laplacian Formulation of Sample Distance Correlation

In this section, we propose a Laplacian formulation of sample distance covariance, and sample distance correlation, which we later use to propose our objective used for supervised dimensionality reduction (SDR).

A graph Laplacian version of sample distance correlation can be obtained as follows,

Lemma 25. *Given matrices of squared Euclidean distances* $\mathbf{E_X}$ *and* $\mathbf{E_Y}$*, and Laplacians* $\mathbf{L_X}$ *and* $\mathbf{L_Y}$ *formed over adjacency matrics* $\widehat{\mathbf{E}}_\mathbf{X}$ *and* $\widehat{\mathbf{E}}_\mathbf{Y}$*, the square of sample distance correlation*

$\hat{\rho}^2(\mathbf{X}, \mathbf{Y})$ *is given by*

$$\hat{\rho}^2(\mathbf{X}, \mathbf{Y}) = \frac{\mathbf{Tr}\left(\mathbf{X}^T\mathbf{L_Y}\mathbf{X}\right)}{\sqrt{\mathbf{Tr}\left(\mathbf{Y}^T\mathbf{L_Y}\mathbf{Y}\right)\mathbf{Tr}\left(\mathbf{X}^T\mathbf{L_X}\mathbf{X}\right)}}. \tag{5.1}$$

Proof. Given matrices $\widehat{\mathbf{E}}_\mathbf{X}$, $\widehat{\mathbf{E}}_\mathbf{Y}$, and column centered matrices $\widetilde{\mathbf{X}}$, $\widetilde{\mathbf{Y}}$, from result of Torgerson [95] we have that $\widehat{\mathbf{E}}_\mathbf{X} = -2\widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^T$ and $\widehat{\mathbf{E}}_\mathbf{Y} = -2\widetilde{\mathbf{Y}}\widetilde{\mathbf{Y}}^T$. In the problem of multidimensional scaling (MDS) [96], we know for a given adjacency matrix say $\mathbf{W}$ and a Laplacian matrix $\mathbf{L}$,

$$\mathbf{Tr}\left(\mathbf{X}^T\mathbf{L}\mathbf{X}\right) = \frac{1}{2}\sum_{i,j}[\mathbf{W}]_{ij}[\mathbf{E}_\mathbf{X}]_{i,j}. \tag{5.2}$$

Now for the Laplacian $\mathbf{L} = \mathbf{L_X}$ and adjacency matrix $\mathbf{W} = \widehat{\mathbf{E}}_\mathbf{Y}$ we can represent $\mathbf{Tr}\left(\mathbf{X}^T\mathbf{L_Y}\mathbf{X}\right)$ in terms of $\widehat{\mathbf{E}}_\mathbf{Y}$ as follows,

$$\mathbf{Tr}\left(\mathbf{X}^T\mathbf{L_Y}\mathbf{X}\right) = \frac{1}{2}\sum_{i,j=1}^{n}[\widehat{\mathbf{E}}_\mathbf{Y}]_{i,j}[\mathbf{E}_\mathbf{X}]_{i,j}.$$

From the fact $[\mathbf{E}_\mathbf{X}]_{i,j} = (\langle\widetilde{\mathbf{x}}_i, \widetilde{\mathbf{x}}_i\rangle + \langle\widetilde{\mathbf{x}}_j, \widetilde{\mathbf{x}}_j\rangle - 2\langle\widetilde{\mathbf{x}}_i, \widetilde{\mathbf{x}}_j\rangle)$, and also $\widehat{\mathbf{E}}_\mathbf{X} = -2\widetilde{\mathbf{X}}\widetilde{\mathbf{X}}^T$ we get

$$\begin{aligned}\mathbf{Tr}\left(\mathbf{X}^T\mathbf{L_Y}\mathbf{X}\right) &= -\frac{1}{4}\sum_{i,j=1}^{n}[\widehat{\mathbf{E}}_\mathbf{Y}]_{i,j}([\widehat{\mathbf{E}}_\mathbf{X}]_{i,i} + [\widehat{\mathbf{E}}_\mathbf{X}]_{j,j} - 2[\widehat{\mathbf{E}}_\mathbf{X}]_{i,j}) \\ &= \frac{1}{2}\sum_{i,j}[\widehat{\mathbf{E}}_\mathbf{X}]_{i,j}[\widehat{\mathbf{E}}_\mathbf{Y}]_{i,j} - \frac{1}{4}\sum_{j}^{n}[\widehat{\mathbf{E}}_\mathbf{X}]_{j,j}\sum_{i}^{n}[\widehat{\mathbf{E}}_\mathbf{Y}]_{i,j} - \frac{1}{4}\sum_{i}^{n}[\widehat{\mathbf{E}}_X]_{i,i}\sum_{j}^{n}[\widehat{\mathbf{E}}_\mathbf{Y}]_{i,j}\end{aligned}$$

Since $\widehat{\mathbf{E}}_\mathbf{X}$ and $\widehat{\mathbf{E}}_\mathbf{Y}$ are double centered matrices $\sum_{i=1}^{n}[\widehat{\mathbf{E}}_\mathbf{Y}]_{i,j} = \sum_{j=1}^{n}[\widehat{\mathbf{E}}_\mathbf{Y}]_{i,j} = 0$ it follows that

$$\mathbf{Tr}\left(\mathbf{X}^T\mathbf{L_Y}\mathbf{X}\right) = \frac{1}{2}\sum_{i,j}[\widehat{\mathbf{E}}_\mathbf{X}]_{i,j}[\widehat{\mathbf{E}}_\mathbf{Y}]_{i,j}.$$

It also follows that

$$\hat{\nu^2}(\mathbf{X}, \mathbf{Y}) = \frac{1}{n^2}\sum_{i,j=1}^{n}[\widehat{\mathbf{E}}_\mathbf{Y}]_{i,j}[\mathbf{E}_\mathbf{X}]_{i,j} = \frac{2}{n^2}\mathbf{Tr}\left(\mathbf{X}^T\mathbf{L_Y}\mathbf{X}\right)$$

Similarly, we can express the sample distance covariance using Laplacians $\mathbf{L_X}$ and $\mathbf{L_Y}$ as

$$\hat{\nu^2}(\mathbf{X}, \mathbf{Y}) = \left(\frac{2}{n^2}\right) \mathbf{Tr}\left(\mathbf{X}^T\mathbf{L_Y}\mathbf{X}\right) = \left(\frac{2}{n^2}\right) \mathbf{Tr}\left(\mathbf{Y}^T\mathbf{L_X}\mathbf{Y}\right).$$

The sample distance variances can be expressed as $\hat{\nu}^2(\mathbf{X}, \mathbf{X}) = \left(\frac{2}{n^2}\right) \mathbf{Tr}\left(\mathbf{X}^T\mathbf{L_X}\mathbf{X}\right)$ and $\hat{\nu}^2(\mathbf{Y}, \mathbf{Y}) = \left(\frac{2}{n^2}\right) \mathbf{Tr}\left(\mathbf{Y}^T\mathbf{L_Y}\mathbf{Y}\right)$ substituting back into expression of sample distance correlation above we get Equation 5.1. □

5.3 Framework

5.3.1 Problem Formulation

The goal in supervised dimensionality reduction (SDR) is to learn a low dimensional representation $\mathbf{Z} \in \mathbb{R}^{n\times p}$ of input features $\mathbf{X} \in \mathbb{R}^{n\times d}$, so as to predict the respone vector $\mathbf{y} \in \mathbb{R}$ from $\mathbf{Z}$. The intuition being that $\mathbf{Z}$ captures all information relevant to predict $\mathbf{y}$. Also, during testing, for out-of-sample prediction, for a new data point $\mathbf{x}^*$, we estimate $\mathbf{z}^*$ assuming that it is predictable from $\mathbf{x}^*$. In our proposed formulation, we use aforementioned Laplacian based sample distance correlation to measure dependencies between variables. We propose maximize dependencies between the low dimensional features $\mathbf{Z}$ and response vector $\mathbf{y}$, and also low dimensional features $\mathbf{Z}$ with input features $\mathbf{X}$. Our objective is to maximize the sum of squares of these two sample distance correlations which is given by,

$$f(\mathbf{Z}) = \hat{\rho}^2(\mathbf{X}, \mathbf{Z}) + \hat{\rho}^2(\mathbf{Z}, \mathbf{y}) \tag{5.3}$$

$$f(\mathbf{Z}) = \frac{\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L_X}\mathbf{Z}\right)}{\sqrt{\mathbf{Tr}\left(\mathbf{X}^T\mathbf{L_X}\mathbf{X}\right)\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L_Z}\mathbf{Z}\right)}} + \frac{\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L_y}\mathbf{Z}\right)}{\sqrt{\mathbf{Tr}\left(\mathbf{y}^T\mathbf{L_y}\mathbf{y}\right)\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L_Z}\mathbf{Z}\right)}}. \tag{5.4}$$

On simplification we get the following optimization problem which we refer to as **Problem (P)**.

$$\max_{\mathbf{Z}} \qquad f(\mathbf{Z}) = \frac{\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{S_{X,y}}\mathbf{Z}\right)}{\sqrt{\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L_Z}\mathbf{Z}\right)}} \qquad \textbf{Problem (P)}$$

where $k_X = \frac{1}{\sqrt{\mathbf{Tr}(\mathbf{X}^T\mathbf{L_X}\mathbf{X})}}$, $k_Y = \frac{1}{\sqrt{\mathbf{Tr}(\mathbf{y}^T\mathbf{L_y}\mathbf{y})}}$ are constants, and $\mathbf{S_{X,y}} = k_X\mathbf{L_X} + k_Y\mathbf{L_y}$.

5.3.2 Algorithm

In the proposed problem (**Problem (P)**), we observe that numerator of our objective is convex while denominator is non-convex due the presence of a square root and a nonlinear Laplacian term $\mathbf{L_Z}$ on $\mathbf{Z}$. Hence, this makes direct optimization of this objective practically infeasible. So to optimize **Problem (P)**, we present a surrogate objective **Problem (Q)** which lower bounds our proposed original objective. We maximize this lower bound with respect to $\mathbf{Z}$ and show that optimizing this surrogate objective **Problem (Q)** (lower bound), also maximizes the proposed objective in **Problem (P)**. We do so by utlizing the Generalized Minorization-Maximization (G-MM) framework of Parizi et al. [24].

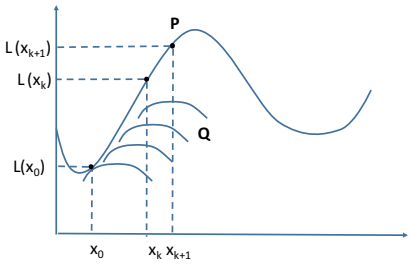


Figure 5.1: Generalized MM algorithm.

The G-MM framework of Parizi et al. [24] is an extension of the well known MM framework of Lange et al. [97] (Figure 5.1). It removes the equality constraint between both objectives at every iteration $\mathbf{Z}_k$, except at initialization step $\mathbf{Z}_0$. This allows the use a broader class of surrogates that avoid maximization iterations being trapped at sharp local maxima, and also makes the problem less sensitive to problem initializations.

The surrogate lower bound objective is as follows,

$$\max_{\mathbf{Z}} \qquad g(\mathbf{Z}, \mathbf{M}) = \frac{\mathbf{Tr}\left(\mathbf{Z}^T \mathbf{S}_{\mathbf{X},\mathbf{y}} \mathbf{Z}\right)}{\mathbf{Tr}\left(\mathbf{Z}^T \mathbf{L}_{\mathbf{M}} \mathbf{Z}\right)} \qquad \textbf{Problem (Q)}$$

where $\mathbf{M} \in \mathbb{R}^{n \times d}$ belongs to the set of column-centered matrices.

The surrogate problem (**Problem (Q)**) is convex in both its numerator and denominator for a fixed auxiliary variable $\mathbf{M}$. Theorem 26 provides the required justification that under certain conditions, maximizing the surrogate **Problem (Q)** also maximizes the proposed objective **Problem (P)** .

An outline of the strategy for optimization is as follows:

a) **Initialize**: Initialize $\mathbf{Z}_0 = \left[c\mathbf{J}_d, \mathbf{0}^T_{(n-d)\times d}\right]^T$, a column-centered matrix where $c = \frac{1}{\sqrt[4]{2(d-1)}}$

and $\mathbf{J}_d \in \mathbb{R}^{d\times d}$ is a centering matrix. This is motivated by statement 1) in proof of Theorem 26.

b) **Optimize**: Maximize the surrogate lower bound $\mathbf{Z}_{k+1} = \arg\max g(\mathbf{Z}, \mathbf{Z}_k)$ (Section 5.4).

c) **Rescaling**: Rescale $\mathbf{Z}_{k+1} \leftarrow \kappa\mathbf{Z}_{k+1}$ such that $\mathbf{Tr}\left(\mathbf{Z}_{k+1}\mathbf{L}_{\mathbf{Z}_{k+1}}\mathbf{Z}_{k+1}\right)$ is greater than one. This is motivated by proof of statement 3) of Theorem 26, and also the fact that $g(\mathbf{Z}, \mathbf{M}) = g(\kappa\mathbf{Z}, \mathbf{M})$ and $f(\mathbf{Z}) = f(\kappa\mathbf{Z})$ for any scalar κ.

d) Repeat step b and c above until convergence.

Theorem 26. *Under above strategy, maximizing the surrogate* ***Problem Q*** *also maximizes* ***Problem P****.*

Proof. For convergence it is enough for us to show the following, [24]:

1. $f(\mathbf{Z}_0) = g(\mathbf{Z}_0, \mathbf{Z}_0)$ for $\mathbf{Z}_0 = \left[c\mathbf{J}_d, \mathbf{0}^T_{(n-d)\times d}\right]^T$ and $c = \frac{1}{\sqrt[4]{2(d-1)}}$,
2. $g(\mathbf{Z}_{k+1}, \mathbf{Z}_k) \geq g(\mathbf{Z}_k, \mathbf{Z}_k)$ and,
3. $f(\mathbf{Z}_{k+1}) \geq g(\mathbf{Z}_{k+1}, \mathbf{Z}_k)$

To prove statement 1, for $\mathbf{Z}_0 = \left[c\mathbf{J}_d, \mathbf{0}^T_{(n-d)\times d}\right]^T$, we observe that $\mathbf{Z}_0$ column-centered, $\mathbf{L}_{\mathbf{Z}_0} = 2\mathbf{Z}_0\mathbf{Z}_0^T$ and $\mathbf{Z}_0^T\mathbf{Z}_0 = c^2\mathbf{J}_d$. Hence we get $\mathbf{Tr}\left(\mathbf{Z}_0^T\mathbf{Z}_0\mathbf{L}_{\mathbf{Z}_0}\mathbf{Z}_0\right) = c^4\mathbf{Tr}\left(2\mathbf{J}_d\right) = c^4 2(d-1) = 1$. This proves the required statement $f(\mathbf{Z}_0) = g(\mathbf{Z}_0, \mathbf{Z}_0) = \mathbf{Tr}\left(\mathbf{Z}_0^T\mathbf{L}_{\mathbf{Z}_0}\mathbf{Z}_0^T\right)$.

Statement 2 follows from the optimization $\mathbf{Z}_{k+1} = \arg\max g(\mathbf{Z}, \mathbf{Z}_k)$. To prove statement 3 we have to show that

$$\frac{\mathbf{Tr}\left(\mathbf{Z}_{k+1}^T\mathbf{S}_{\mathbf{X},\mathbf{y}}\mathbf{Z}_{k+1}\right)}{\sqrt{\mathbf{Tr}\left(\mathbf{Z}_{k+1}^T\mathbf{L}_{\mathbf{Z}_{k+1}}\mathbf{Z}_{k+1}\right)}} \geq \frac{\mathbf{Tr}\left(\mathbf{Z}_{k+1}^T\mathbf{S}_{\mathbf{X},\mathbf{y}}\mathbf{Z}_{k+1}\right)}{\mathbf{Tr}\left(\mathbf{Z}_{k+1}^T\mathbf{L}_{\mathbf{Z}_k}\mathbf{Z}_{k+1}\right)}.$$

Since numerators on both sides are equal, it is enough for us to show that

$$\sqrt{\mathbf{Tr}\left(\mathbf{Z}_{k+1}^T\mathbf{L}_{\mathbf{Z}_{k+1}}\mathbf{Z}_{k+1}\right)} \leq \mathbf{Tr}\left(\mathbf{Z}_{k+1}^T\mathbf{L}_{\mathbf{Z}_k}\mathbf{Z}_{k+1}\right).$$

Now from Lemma 33 we have $\mathbf{Tr}\left(\mathbf{Z}_{k+1}^T\mathbf{L}_{\mathbf{Z}_{k+1}}\mathbf{Z}_{k+1}\right) \leq \mathbf{Tr}\left(\mathbf{Z}_{k+1}^T\mathbf{L}_{\mathbf{Z}_k}\mathbf{Z}_{k+1}\right)$. It follows from the rescaling step (step c) of the optimization strategy that the left hand side $\mathbf{Tr}\left(\mathbf{Z}_{t+1}\mathbf{L}_{\mathbf{Z}_{t+1}}\mathbf{Z}_{t+1}\right)$

is always greater that one, and so taking square root of it implies $\sqrt{\mathbf{Tr}\left(\mathbf{Z}_{t+1}\mathbf{L}_{\mathbf{Z}_{t+1}}\mathbf{Z}_{t+1}\right)} \leq \mathbf{Tr}\left(\mathbf{Z}_{t+1}^T\mathbf{L}_{\mathbf{Z}_t}\mathbf{Z}_{t+1}\right)$. □

We summarize all of the above steps in Algorithm 5.3.1 below and section 5.4 further describes optimization algorithm to solve **Problem (Q)** required by it.

Algorithm 5.3.1 DISCOMAX

Require: Initialize $\mathbf{Z}_0 = \left[c\mathbf{J}_d, \mathbf{0}_{(n-d)\times d}^T\right]^T$, a column-centered matrix where $c = \frac{1}{\sqrt[4]{2(d-1)}}$, $k \leftarrow 0$

Ensure: $\mathbf{Z}^* = \arg\max_{\mathbf{Z}} f(\mathbf{Z})$

1: **repeat**

2: Solve,

$$\mathbf{Z}_{k+1} = \arg\max_{\mathbf{Z}} g(\mathbf{Z}, \mathbf{Z}_k) \qquad \textbf{Problem (Q)}$$

3: Rescale $\mathbf{Z}_{k+1} \leftarrow \kappa\mathbf{Z}_{k+1}$ such that $\mathbf{Tr}\left(\mathbf{Z}_{k+1}^T\mathbf{L}_{\mathbf{Z}_{k+1}}\mathbf{Z}_{k+1}\right) \geq 1$

4: $k = k + 1$

5: **until** $\|\mathbf{Z}_{k+1} - \mathbf{Z}_k\|^2 < \epsilon$

6: $\mathbf{Z}^* = \mathbf{Z}_{k+1}$

7: **return** $\mathbf{Z}^*$

5.4 Optimization

In this section, we propose a framework for optimizing the surrogate objective $g(\mathbf{Z}, \mathbf{M})$, referred to as **Problem (Q)**, for a fixed $\mathbf{M} = \mathbf{Z}_k$. We observe that for a given value of $\mathbf{M}$, $g(\mathbf{Z}, \mathbf{M})$ is a ratio of two convex functions. To solve this, we convert this maximization problem to an equivalent minimization problem $h(\mathbf{Z}, \mathbf{M})$, by taking its reciprocal [98]. This allows us to utilize the Quadratic Fractional Programming Problem (QFPP) framework of Dinkelbach [99] and Zhang [100] to minimize $h(\mathbf{Z}, \mathbf{M})$. We refer to this new minimization problem as **Problem (R)**. It is stated below.

$$\min_{\mathbf{Z}} \qquad h(\mathbf{Z}, \mathbf{M}) = \frac{\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L}_{\mathbf{M}}\mathbf{Z}\right)}{\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{S}_{\mathbf{X},\mathbf{y}}\mathbf{Z}\right)} \qquad \textbf{Problem (R)} \tag{5.5}$$

where $\mathbf{M} = \mathbf{Z}_k$.

In his seminal work Dinkelbach [99] and later Zhang [100] proposed a novel framework to solve constrained QFP problems by converting it to an equivalent parametric optimization

problem, by introducing a scalar parameter $\alpha \in \mathbb{R}$. We utilize this equivalence proposed to defined new parametric problem, **Problem (S)**. The solution involves a search over the scalar parameter α while repeatedly solving **Problem (S)** to get the required solution $\mathbf{Z}_{k+1}$. This search process continues until values of α converge.

In a nutshell, Dinkelbach [99] and Zhang [100] frameworks suggest the following optimizations are equivalent:

Problem (R)		**Problem (S)**
$\underset{\mathbf{z}\in\mathbb{R}^d}{\text{minimize}}\ h(\mathbf{z}) = \frac{f_1(\mathbf{z})}{f_2(\mathbf{z})}$	$\Longleftrightarrow$	$\underset{\mathbf{z}\in\mathbb{R}^d}{\text{minimize}}\ H(\mathbf{z};\alpha^*) = f_1(\mathbf{z}) - \alpha^* f_2(\mathbf{z})$ for some $\alpha^* \in \mathbb{R}$

where $f_i(\mathbf{z}) := \mathbf{z}_i^T \mathbf{A}_i \mathbf{z} - 2\mathbf{b}_i \mathbf{z} + c_i$ with $\mathbf{A}_1, \mathbf{A}_2 \in \mathbb{R}^{n\times n}$, $\mathbf{b}_1, \mathbf{b}_2 \in \mathbf{R}^n$, and $c_1, c_2 \in \mathbb{R}$. $\mathbf{A}_1$ and $\mathbf{A}_2$ are symmetric with $f_2(\mathbf{x}) > 0$ over some $\mathbf{z} \in \mathcal{Z}$.

To see the equivalence of $h(\mathbf{Z}, \mathbf{M})$ in **Problem (R)** to $h(\mathbf{z})$ above we observe that: $\mathbf{A}_1 = \mathbf{I}_n \otimes \mathbf{L}_\mathbf{M}$, $\mathbf{A}_2 = \mathbf{I}_n \otimes \mathbf{S}_{\mathbf{X},\mathbf{y}}$, $c_i = c_2 = 0$, and $\mathbf{b}_1 = \mathbf{b}_2 = \mathbf{0}$. Also, due to positive definiteness of $\mathbf{A}_i$, $f_i(\mathbf{z})$ is positive[1], and $f(\mathbf{z}_i) > 0$. Using this setup for $h(\mathbf{Z}, \mathbf{M})$ we get,[2]

$$\min_{\mathbf{Z}} \qquad h(\mathbf{Z}, \mathbf{M}) = \frac{\mathbf{vec}\,(\mathbf{Z})^T\,(\mathbf{I}_n \otimes \mathbf{L}_\mathbf{M})\mathbf{vec}\,(\mathbf{Z})}{\mathbf{vec}\,(\mathbf{Z})^T\,(\mathbf{I}_n \otimes \mathbf{S}_{\mathbf{X},\mathbf{y}})\mathbf{vec}\,(\mathbf{Z})} \tag{5.6}$$

In subsection 5.4.1 we propose a Golden Section Search [101] based algorithm (Algorithm 5.4.1) which utilizes concavity property of $H(\mathbf{Z};\alpha)$ with respect to α to locate the best α^*. During this search we repeatedly solve **Problem (S)** starting with an intial interval $0 = \alpha_l \leq \alpha \leq \alpha_u = \lambda_{min}(\mathbf{L}_\mathbf{M}, \mathbf{S}_{\mathbf{X},\mathbf{y}})$ for a fixed $\mathbf{M}$, then at each step shorten the search interval by moving upper and lower limits closer to each other. We continue until convergence to α^*. The choice of the upper limit of $\alpha_u = \lambda_{min}(\mathbf{L}_\mathbf{M}, \mathbf{S}_{\mathbf{X},\mathbf{y}})$ is motivated by proof of Lemma 31.

To solve **Problem (S)** for a given α, we propose an iterative algorithm in subsection 5.4.2 (Algorithm 5.4.2). It uses the classical Majorization-Minimization framework of Lange [102].

[1] If $\mathbf{A}_i$ is semi-definite then we regularize by adding $\mathbf{A}_i + \epsilon\mathbf{I}$ so that $\mathbf{A}_i \succ 0$.

[2] $\otimes$ indicates kronecker product. $\mathbf{vec}\,(\mathbf{Z})$ denotes column vectorization of matrix $\mathbf{Z}$.

5.4.1 Golden Section Search

Dinkelbach [99] and Zhang [100] showed the following properties of the objective[3] $H(\alpha)$ with respect to α, for a fixed $\mathbf{Z}$.

Theorem 27. *Let $G\colon \mathbb{R} \to \mathbb{R}$ be defined as*

$$G(\alpha) = \min_{\mathbf{Z}} H(\mathbf{Z}; \alpha) = \min_{\mathbf{Z}} \left\{ \mathbf{Tr}\left(\mathbf{Z}^T \mathbf{L_M} \mathbf{Z}\right) - \alpha \mathbf{Tr}\left(\mathbf{Z}^T \mathbf{S_{X,y}} \mathbf{Z}\right) \right\}$$

*as derived from **Problem (S)**, then following statements hold true.*

1. *G is continuous at any $\alpha \in \mathbb{R}$.*
2. *G is concave over $\alpha \in \mathbb{R}$.*
3. *$G(\alpha) = 0$, has a unique solution α^*.*

Algorithm 5.4.1 exploits the concavity property of $G(\alpha)$ to perform a Golden Section Search over α. Subsection 5.4.2 provides an iterative Majorization-Minimization algorithm (Algorithm 5.4.2) to solve this minimization problem **Problem (S)**.

5.4.2 Distance Correlation Maximization

Algorithm 5.4.2 gives a iterative fixed point algorithm which solves **Problem (S)**. Theorem 28 provides a fixed point iterate used to minimize $H(\mathbf{Z}, \alpha)$ with respect to $\mathbf{Z}$ for a given α. The fixed point iterate[4] $\mathbf{Z}_{t+1} = \mathbf{H}\mathbf{Z}_t$ minimizes **Problem (S)** and a monotonic convergence is assured by the Majorization-Minimization result of Lange [102]. Theorem 28 below derives the fixed point iterate used in Algorithm 5.4.2.

Theorem 28. *For a fixed γ^2 (Lemma 30), some α (Lemma 31) and*

$$\mathbf{H} = \left(\gamma^2 \mathbf{D}_X - \alpha \mathbf{S_{X,y}}\right)^{\dagger} \left(\gamma^2 \mathbf{D}_X - \mathbf{L_M}\right)$$

[3] For a fixed $\mathbf{Z}$ and variable argument α we denote $H(\mathbf{Z}; \alpha)$ as $H(\alpha)$.

[4] We use the subscript t to indicate fixed point iteration of $\mathbf{Z}_t$.

the iterate $\mathbf{Z}_t = \mathbf{H}\mathbf{Z}_{t-1}$ *monotonically minimizes the objective,*

$$F(\mathbf{Z};\alpha) = \mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L_M}\mathbf{Z}\right) - \alpha\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{S_{X,y}}\mathbf{Z}\right) \tag{5.7}$$

Proof. From Lemma 30 we know that, $(\gamma^2\mathbf{D_X} - \mathbf{L_M}) \succeq 0$. Hence the following would hold true for any real matrix $\mathbf{N}$,

$$\mathbf{Tr}\left((\mathbf{Z}-\mathbf{N})^T(\gamma^2\mathbf{D}_X - \mathbf{L_M})(\mathbf{Z}-\mathbf{N})\right) \geq 0$$

Rearranging the terms we get the following inequality over $\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L_M}\mathbf{Z}\right)$,

$$\begin{aligned}
&\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L_M}\mathbf{Z}\right) + \mathbf{Tr}\left(\mathbf{N}^T(\gamma^2\mathbf{D}_X - \mathbf{L_M})\mathbf{Z}\right) - \mathbf{Tr}\left(\mathbf{N}^T(\gamma^2\mathbf{D}_X - \mathbf{L_M})\mathbf{N}\right) \\
&\qquad \leq \mathbf{Tr}\left(\mathbf{Z}^T\gamma^2\mathbf{D}_X\mathbf{Z}\right) - \mathbf{Tr}\left(\mathbf{Z}^T(\gamma^2(\mathbf{D}_X - \mathbf{L_M})\mathbf{N}\right) \\
&\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L_M}\mathbf{Z}\right) \leq \mathbf{Tr}\left(\mathbf{Z}^T\gamma^2\mathbf{D}_X\mathbf{Z}\right) - 2\mathbf{Tr}\left(\mathbf{Z}^T(\gamma^2\mathbf{D}_X - \mathbf{L_M})\mathbf{N}\right) + \mathbf{Tr}\left(\mathbf{N}^T(\gamma^2\mathbf{D}_X - \mathbf{L_M})\mathbf{N}\right) \\
&\qquad = l(\mathbf{Z},\mathbf{N})
\end{aligned}$$

If $\mathbf{N} = \mathbf{Z}$ then $l(\mathbf{Z},\mathbf{Z}) = \mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L_M}\mathbf{Z}\right)$. Hence $l(\mathbf{Z},\mathbf{N})$ majorizes $\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L_M}\mathbf{Z}\right)$. It also follows that the surrogate function $l(\mathbf{Z},\mathbf{N}) - \alpha\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{S_{X,y}}\mathbf{Z}\right)$ majorizes our desired objective function $H(\mathbf{Z};\alpha)$. To optimize this surrogate loss we equate its gradient to zero and rearrange the terms to obtain

$$\begin{aligned}
(\gamma^2\mathbf{D}_X - \alpha\mathbf{S_{X,y}})\mathbf{Z} &= (\gamma^2\mathbf{D}_X - \mathbf{L_M})\mathbf{N} \\
\mathbf{Z} &= (\gamma^2\mathbf{D}_X - \alpha\mathbf{S_{X,y}})^\dagger(\gamma^2\mathbf{D}_X - \mathbf{L_M})\mathbf{N},
\end{aligned}$$

which gives us the update equation $\mathbf{Z}_{t+1} = \mathbf{H}\mathbf{Z}_t$ where $\mathbf{H}$ is given by,

$$\mathbf{H} = (\gamma^2\mathbf{D}_X - \alpha\mathbf{S_{X,y}})^\dagger(\gamma^2\mathbf{D}_X - \mathbf{L_M}). \tag{5.8}$$

Hence it follows from framework of Lange [102] that above update equation monotonically minimizes $H(\mathbf{Z};\alpha)$. □

Algorithm 5.4.2 summarizes the steps of an iterative Majorization-Minimization approach to solve **Problem (S)**.

Algorithm 5.4.1 Golden Section Search for $\alpha \in [\alpha_l, \alpha_u]$ for a fixed $\mathbf{M} = \mathbf{Z}_k$.

Require: ϵ, $\eta = \frac{1+\sqrt{5}}{2}$, $\alpha_l = 0$, $\mathbf{S}_{\mathbf{X},\mathbf{y}}$, $\mathbf{L}_{\mathbf{X}}$, $\mathbf{L}_{\mathbf{y}}$, $\mathbf{M} = \mathbf{Z}_k$.
Ensure: $\mathbf{Z}_{k+1} = \arg\min_{\mathbf{Z}} g(\mathbf{Z}, \mathbf{Z}_{k+1})$

$\mathbf{D}_X \leftarrow \mathbf{diag}(\mathbf{L}_{\mathbf{X}})$
$\mathbf{L}_{\mathbf{M}} \leftarrow 2\mathbf{M}^T\mathbf{M}$
$\alpha_u \leftarrow \lambda_{max}(\mathbf{L}_{\mathbf{M}}, \mathbf{S}_{\mathbf{X},\mathbf{y}})$ (Lemma 30)
$\beta \leftarrow \alpha_u + \eta(\alpha_l - \alpha_u)$
$\delta \leftarrow \alpha_l + \eta(\alpha_u - \alpha_l)$
repeat
    $H(\beta) \leftarrow \underset{\mathbf{Z}\in\mathbb{R}^d}{\text{minimize}} \left(\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L}_{\mathbf{M}}\mathbf{Z}\right) - \beta\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{S}_{\mathbf{X},\mathbf{y}}\mathbf{Z}\right)\right)$ (**Problem (S)**)
    $H(\delta) \leftarrow \underset{\mathbf{Z}\in\mathbb{R}^d}{\text{minimize}} \left(\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L}_{\mathbf{M}}\mathbf{Z}\right) - \delta\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{S}_{\mathbf{X},\mathbf{y}}\mathbf{Z}\right)\right)$ (**Problem (S)**)
    if $(H(\beta) > H(\delta))$ **then**
        $\alpha_u \leftarrow \delta$, $\delta \leftarrow \beta$
        $\beta \leftarrow \alpha_u + \eta(\alpha_l - \alpha_u)$
    else
        $\alpha_l \leftarrow \beta$, $\beta \leftarrow \delta$
        $\delta \leftarrow \alpha_l + \eta(\alpha_u - \alpha_l)$
    end if
until $(|\alpha_u - \alpha_l| < \epsilon)$
$\alpha^* \leftarrow \frac{\alpha_u + \alpha_u}{2}$
$\mathbf{Z}_{k+1} \leftarrow \arg\min_{\mathbf{Z}\in\mathbb{R}^d} \left(\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L}_{\mathbf{M}}\mathbf{Z}\right) - \alpha^*\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{S}_{\mathbf{X},\mathbf{y}}\mathbf{Z}\right)\right)$ (**Problem (S)**)
return α^*, $\mathbf{Z}_{k+1}$

Algorithm 5.4.2 Distance Correlation Maximization for a given α

Require: γ^2 (Theorem 30), α, $\mathbf{M} = \mathbf{Z}_k$, $\mathbf{S_{X,y}}$, $\mathbf{L_M}$, $\mathbf{D_X}$
Ensure: $H(\mathbf{Z};\alpha) = \underset{\mathbf{Z}\in\mathbb{R}^d}{\text{minimize}} \left(\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{L_M}\mathbf{Z}\right) - \alpha\mathbf{Tr}\left(\mathbf{Z}^T\mathbf{S_{X,y}}\mathbf{Z}\right)\right)$

$t \leftarrow 0$
$\mathbf{Z}_t = \mathbf{Z}_k$
$H(\mathbf{Z}_t;\alpha) \leftarrow \left(\mathbf{Tr}\left(\mathbf{Z}_t^T\mathbf{L_M}\mathbf{Z}_t\right) - \alpha\mathbf{Tr}\left(\mathbf{Z}_t^T\mathbf{S_{X,y}}\mathbf{Z}_t\right)\right)$
$\mathbf{H} = \left(\gamma^2\mathbf{D_X} - \alpha\mathbf{S_{X,y}}\right)^{\dagger}\left(\gamma^2\mathbf{D_X} - \mathbf{L_M}\right)$
repeat
    $\mathbf{Z}_{t+1} = \mathbf{H}\mathbf{Z}_t$
    $H(\mathbf{Z}_{t+1};\alpha) \leftarrow \left(\mathbf{Tr}\left(\mathbf{Z}_t^T\mathbf{L_M}\mathbf{Z}_t\right) - \alpha\mathbf{Tr}\left(\mathbf{Z}_t^T\mathbf{S_{X,y}}\mathbf{Z}_t\right)\right)$
    $t \leftarrow t + 1$
until $(|H(\mathbf{Z}_{t+1};\alpha) - H(\mathbf{Z}_t;\alpha)| < \epsilon)$ **or** $(t \geq T_{\max})$
$F(\alpha) \leftarrow H(\mathbf{Z}_t;\alpha)$
$\mathbf{Z}^* \leftarrow \mathbf{Z}_t$
return $F(\alpha), \mathbf{Z}^*$

5.5 Experiments

In this section we present experimental results that compare our proposed method with several state-of-the-art supervised dimensionality reduction techniques on a regression task.

5.5.1 Methodology

Methodology we use for our experiments is as follows:

(i) We run our proposed algorithm on the training set $\mathbf{X}_{\texttt{Train}}$ to learn low-dimensional features $\mathbf{Z}_{\texttt{Train}}$.

(ii) We learn the map $\psi\colon \mathbf{z} \mapsto y$ using Support Vector Regression on $\mathbf{Z}_{\texttt{Train}}$ and $\mathbf{Y}_{\texttt{Train}}$.

(iii) We learn mappings $\phi_i\colon \mathbf{x} \mapsto z_i$, $i = 1$ to d for each dimension of $\mathbf{z}$ using Support Vector Regression on $\mathbf{X}_{\texttt{Train}}$ and $\mathbf{Z}_{\texttt{Train}}$.

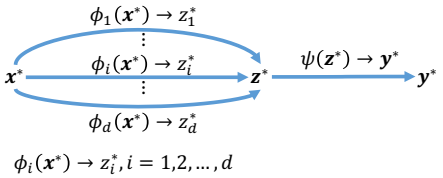


Figure 5.2: Out-of-sample prediction

During testing/out-of-sample phase, given a test input $\mathbf{x}^*$, we use maps $\phi_i\colon \mathbf{x} \mapsto z_i$ for $i = 1$ to d and generate $\mathbf{z}^*$. We then utilize maps $\psi\colon \mathbf{z} \mapsto y$ on $\mathbf{z}^*$ to get the predicted response y^*. Figure 5.2 illustrates the testing phase of our methodology.

5.5.2 Datasets

In our results we report the Root Mean Squared (RMS) errors on five datasets from the UCI-Machine Learning Repository [103] in Tables 5.1 to 5.5. We use the following datasets in our experiments.

(a) **Boston Housing** [3]: This dataset contains information collected by the U.S Census Service concerning housing in the area of Boston Mass. This dataset has been used extensively throughout the vast regression literature to benchmark algorithms. The response variable to be predicted is the median value of owner-occupied homes.

(b) **Relative Location of Computed Tomography (CT) Slices** [4]: This dataset consists of 385 features extracted from computed tomography (CT) images. Each CT slice is described by two histograms in polar space that are concatenated to form the final feature vector. The response variable to be predicted is the relative location of an image on the axial axis. The ground truth of responses in this dataset was constructed by manually annotating up to 10 distinct landmarks in each CT Volume with a known location. This response takes values in the range $[0, 180]$ where 0 denotes the top of the head and 180 denotes the the soles of the feet.

(c) **BlogFeedback** [5]: This dataset originates from a set of raw HTML documents of blog posts that were crawled and processed. The task associated with this data is to predict the number of comments in the upcoming 24 hours. In order to simulate this situation, the dataset was curated by choosing a base time (in the past) and selecting the blog posts that were published at most 72 hours before the selected base date/time. Then a set of 281 features of the selected blog posts were computed from the information that was available at the basetime. The target is to predict the number of comments that the blog post received in the next 24 hours, relative to the basetime. In the training data, the base times were in the years 2010 and 2011. In the test data the base times were in February and March 2012.

(d) **Geographical Origin of Music** [6]: Instances in this dataset contain audio features extracted from 1059 wave files covering 33 countries/areas. The task associated with the data is to predict the geographical origin of music. The program MARSYAS was used to extract

68 audio features from the wave files. These were appended with 48 chromatic attributes that describe the notes of the scale bringing the total number of features to 116.

(e) **UJI Indoor Localization** [7]: The UJIIndoorLoc is a Multi-Building Multi-Floor indoor localization database that relies on WLAN/WiFi fingerprinting technology. Automatic user localization consists of estimating the position of the user (latitude, longitude and altitude) by using an electronic device, usually a mobile phone. The task is to predict the actual longitude and latitude. The database consists of 19937 training/reference records and 1111 validation/test records. The 529 features contain the WiFi fingerprint, the coordinates where it was taken, and other useful information. Given that this paper focusses on the setting of univariate responses, we only aim to predict the 'Longitude'.

5.5.3 Results

We perform five-fold cross validation on each of these datasets and report the average Root Mean Square (RMS) error on the hold-out test sets. Tables 5.1 to 5.5 present the cross-validated RMS error of our proposed method (DisCoMax), and six other supervised dimensionality reduction techniques namely; LSDR [89], gKDR [85], SCA [86], LAD [84], SAVE [82] and [83] and SIR [78].

In case of DisCoMax, we use the methodology described in sub-section 5.5.1. For other methods we used in our evaluation, these techniques generate explicit maps to obtain the low-dimensional representations. As in the case of the methodology for DisCoMax, we use these

Method/dimension	3	5	7	9	11
DisCoMax	**0.1559**	**0.1493**	**0.1327**	**0.1311**	**0.1297**
LSDR [89]	0.1978	0.1963	0.1892	0.1886	0.1873
gKDR [85]	0.1997	0.1813	0.1762	0.1738	0.1719
SCA [86]	0.1875	0.1796	0.1708	0.1637	0.1602
LAD [84]	0.2019	0.1964	0.1932	0.1917	0.1903
SAVE [82]	0.2045	0.1983	0.1967	0.1952	0.1947
SIR [78]	0.2261	0.2193	0.2086	0.2076	0.2068

Table 5.1: Boston Housing [3]: U.S Census Service concerning housing in the area of Boston Mass to predict median value of owner-occupied homes. Baseline results of SVR RMS error of 0.1719.

Method/d	3	5	7	9	11
DisCoMax	**19.19**	**18.67**	**18.14**	**17.94**	**17.81**
LSDR [89]	23.63	22.31	22.09	21.93	21.82
gKDR [85]	24.06	23.39	22.76	22.52	22.50
SCA [86]	23.17	24.96	24.21	23.34	23.06
LAD [84]	26.74	25.57	24.39	24.26	24.20
SAVE [82]	28.18	27.82	27.62	27.53	27.50
SIR [78]	29.92	29.46	29.18	28.86	28.63

Table 5.2: Geographical Origin of Music [4]: Input contains audio features extracted from 1059 wave files covering 33 countries/areas. The task associated with this data is to predict geographical origin of music.

Method/d	3	5	7	9	11
DisCoMax	**25.82**	**24.69**	**24.33**	**23.90**	**23.62**
LSDR [89]	30.36	28.16	27.39	27.24	27.18
gKDR [85]	29.72	27.62	27.29	26.91	26.81
SCA [86]	28.53	27.31	26.60	26.32	26.30
LAD [84]	30.42	30.39	30.20	30.04	29.99
SAVE [82]	31.93	31.27	30.72	30.53	30.31
SIR [78]	33.63	32.65	31.39	31.16	30.83

Table 5.3: BlogFeedback [5]: Data contains features computed from raw HTML documents of blog posts. The task associated with this data is to predict the number of comments in the upcoming 24 hours.

Method/d	3	5	7	9	11
DisCoMax	**12.29**	**11.11**	**10.19**	**9.73**	**9.66**
LSDR [89]	14.38	13.14	12.87	12.73	12.69
gKDR [85]	13.65	12.86	12.67	12.35	12.05
SCA [86]	14.19	13.64	12.94	12.12	11.73
LAD [84]	17.70	17.62	17.34	17.15	16.89
SAVE [82]	19.32	18.74	18.62	17.76	17.21
SIR [78]	21.53	21.23	20.97	20.77	20.64

Table 5.4: Relative location of CT slices [6]: Dataset consists of 385 features extracted from CT images. Features are concatenation of two histograms in polar space. The response variable is relative location of an image on the axial axis.

Method/d	3	5	7	9	11
DisCoMax	**12.28**	**11.10**	**10.19**	**9.73**	**9.65**
LSDR [89]	14.38	13.14	12.86	12.73	12.69
gKDR [85]	13.65	12.86	12.67	12.34	12.05
SCA [86]	14.18	13.63	12.94	12.12	11.73
LAD [84]	17.69	17.62	17.34	17.15	16.89
SAVE [82]	19.32	18.74	18.61	17.75	17.20
SIR [78]	21.53	21.23	20.97	20.77	20.63

Table 5.5: UJI Indoor Localization [7]: Multi-Building Multi-Floor indoor localization database. Task is to predict the actual longitude and latitude. The 529 attributes contain WiFi fingerprints and coordinates of where they were taken. Database consists of around 20k training/reference records and 11k validation/test records.

explicit maps and Support Vector Regression (with a RBF kernel) to generate cross-validated RMS errors on the responses.

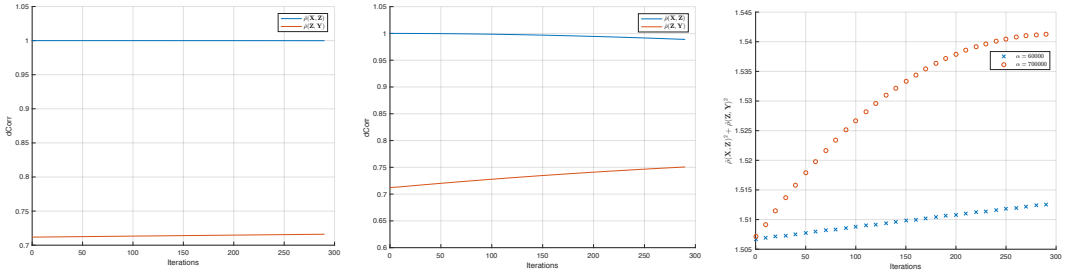
We fix folds across the seven techniques presented within each of the tables (Tables 5.1 to 5.5). We also compute RMS errors for increasing dimensions $d = 3, 5, 7, 9$ and 11. We note the significant improvement in the predictive performance (smaller error) of DisCoMax learnt features across for all cases with different dimensionality, and also gradual increase performance (smaller error) as we increase dimensionality learnt features.

For baseline comparison purposes, in case of the Boston Housing dataset, we observe a RMS error of 0.1719 using Support Vector Regression without any dimensionality reduction ($d = 13$). This when compared to DisCoMax RMS errors which ranged between 0.1559 ($d = 3$) and 0.1297 ($d = 11$) always did worse. We bold errors for DisCoMax for cases where errors were significantly better when compared with their corresponding standard deviations taken into account.

5.6 Discussion

In this section, we discuss effects of choice of α in the optimization of **Problem (S)** (Algorithm 5.4.2). We also empirically show optimization of **Problem (P)** using Algorithm 5.3.1, which optimizes a lower bound in **Problem (Q)**. We use the Boston Housing dataset for our analysis.

Figures 5.3a and 5.3b show gradual increase in sample distance correlations $\hat{\rho}(\mathbf{X}, \mathbf{Z}_t)$

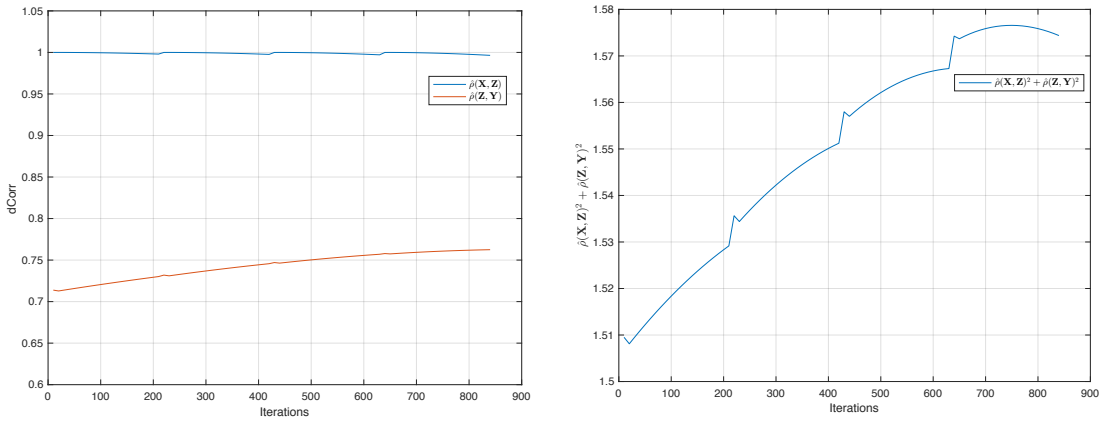


(a) $\hat{\rho}(\mathbf{X}, \mathbf{Z})$ (Blue) and $\hat{\rho}(\mathbf{Z}, \mathbf{y})$ (Red) vs. fixed point iterations (t) for $\alpha = 6 \times 10^4$ (b) $\hat{\rho}(\mathbf{X}, \mathbf{Z})$ (Blue) and $\hat{\rho}(\mathbf{Z}, \mathbf{y})$ (Red) vs. fixed point iterations (t) for $\alpha = 70 \times 10^4$ (c) $f(\mathbf{Z})$ vs. fixed point iterations for $\alpha = 6 \times 10^4$ (Red) for $\alpha = 70 \times 10^4$ (Blue).

Figure 5.3: Effect of α values on growth of proposed objective in Algorithm 5.4.2. Figures show slower (faster) growth of distance correlations for smaller (larger) α.

(Blue) and $\hat{\rho}(\mathbf{Z}_t, \mathbf{y})$ (Red) with respect the number of fixed point t for two different choices of $\alpha = 6 \times 10^4$ and $\alpha = 70 \times 10^4$. We clearly observe that the choice of α has a strong effect on rate of increase/decrease of individual distance correlations $\hat{\rho}^2(\mathbf{X}, \mathbf{Z}_t)$ and $\hat{\rho}^2(\mathbf{Z}_t, \mathbf{y})$ as iterations progress. This is because the α value positively weighs the term $\mathbf{Tr}\left(\mathbf{Z}^T \mathbf{S}_{\mathbf{X},\mathbf{y}} \mathbf{Z}\right)$ over $\mathbf{Tr}\left(\mathbf{Z}^T \mathbf{L}_{\mathbf{M}} \mathbf{Z}\right)$ in **Problem (S)**. Figure 5.3c shows the rate of change of objective function $f(\mathbf{Z})$ with respect to the fixed point iterations t for two choices of α. The figure clearly shows the slower (faster) rate of increase of $f(\mathbf{Z})$ for smaller (larger) α.

Figure 5.4a and 5.4b repectively show the overall growth of distance correlations ($\hat{\rho}(\mathbf{X}, \mathbf{Z})$, $\hat{\rho}(\mathbf{Z}, \mathbf{y})$) and $f(\mathbf{Z})$, with respect to the fixed point iterations (t), for $\alpha^* = 800 \times 10^4$. We periodically observe a sharp increases in $f(\mathbf{Z})$ and distance correlations after each DisCoMax subproblem of 220 fixed point iterations. The figures show four such G-MM iterations of Algorithm 5.3.1. These sharp increases are due to the resubstitution of $\mathbf{M} = \mathbf{Z}_k$ in Step 2 of Algorithm 5.3.1. This clearly shows us that we are able to maximized are original proposed objective in **Problem (P)**.



(a) $\hat{\rho}(\mathbf{X},\mathbf{Z})$ (Blue) and $\hat{\rho}(\mathbf{Z},\mathbf{y})$ (Red) vs overall Iterations.

(b) $f(\mathbf{Z}) = \hat{\rho}(\mathbf{X},\mathbf{Z})^2 + \hat{\rho}(\mathbf{Z},\mathbf{y})^2$ vs overall Iterations.

Figure 5.4: Overall gradual increase in $f(\mathbf{Z})$ (Figure 5.4a) and distance correlations (Figure 5.4b) for $\alpha^* = 800 \times 10^4$. Plots show increase in both for each DisCoMax subproblem of (Algorithm 5.4.2) and four outer G-MM iterations of Algorithm 5.3.1.

5.7 Summary

In this chapter we proposed a novel framework to perform supervised dimensionality reduction. Our framework aims to maximize the dependency measure of called statistical distance correlation. We also proposed a novel algorithm to optimize our proposed objective using the Generalized Minorization-Maximization approach of Parizi et al. [24]. Finally, we showed superior empirical performance of our method on several regression problems in comparison to existing state-of-the-art methods.

Chapter 6

Conclusion

In this dissertation, we proposed two frameworks for learning feature representation by maximizing nonlinear dependency measures for structured data.

In the first framework, we propose a novel method for learning kernels by using polynomial expansions in terms of base kernels. We proposed a novel objective of maximizing dependency between input and outputs measured by Hilbert Schmidt Independence Criterion. We use this objective to learn these polynomial expansions and show that maximizing dependence between input and output kernel features leads to better performance for structured prediction.

Moreover, the algorithm we propose is an efficient matrix-decomposition based algorithm to learn these kernel transformations. We use Twin Gaussian Processes as a prototypical structured prediction example, and using it show state-of-the-art empirical results on several synthetic and real-world datasets.

In the second framework, we use a dependency measure which is a special case of Hilbert Schmidt Independence Criterion to perform supervised dimensionality reduction. Our framework maximizes a statistical measure of dependence called Distance Correlation of Székely et al. [14]. Our formulation is based on learning a low-dimensional feature representation $\mathbf{z}$, which maximizes the squared sum of Distance Correlations between low dimensional features $\mathbf{z}$ and response y, and also between features $\mathbf{z}$ and covariates $\mathbf{x}$.

Furthermore, we propose a novel algorithm to optimize our proposed objective using the Generalized Minorization-Maximization approach of Parizi et al. [24]. Finally, we show a superior empirical performance of our method on several regression problems in comparison to existing state-of-the-art methods.

Future Work In case of first framework, it is worth investigating the following; 1) automated learning of kernel expansion parameters like degree (d_1, d_2) and γ, by using distributional priors on them and optimizing for log-likelihood, 2) extending this framework to multiple kernels for multi-modal and/or multi-task prediction, and 3) joint learning of kernels along with prediction.

In case of second framework, it is possible to easily extend it to handle structured objects, as Distance Correlation is also applicable to general metric spaces with a known distance metric [53]. Our proposed framework is practically applicable on relatively small datasets, as it involves repeatedly solving multiple optimization subproblems. Hence it is worth scaling this approach so that it is tractable for larger size datasets. In this framework we currently tackle the out-of-sample issue by learning multiple SVR's, i.e. one for each dimension of low dimensional feature $\mathbf{z}$, so extending it to learn explicit out-of-sample mappings from $\mathbf{x}$ to $\mathbf{z}$ is also desirable.

References

[1] L. Bo and C. Sminchisescu, "Structured output-associative regression," in *IEEE Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, Jun. 2009, pp. 2403–2410. viii, 44

[2] J. Weston, O. Chapelle, V. Vapnik, A. Elisseeff, and B. Schölkopf, "Kernel dependency estimation," pp. 873–880, 2002. viii, xi, 21, 41, 44

[3] D. Harrison and D. L. Rubinfeld, "Hedonic housing prices and the demand for clean air," *Journal of environmental economics and management*, vol. 5, no. 1, pp. 81–102, 1978. ix, 59, 60

[4] F. Graf, H.-P. Kriegel, M. Schubert, S. Pölsterl, and A. Cavallaro, "2D Image Registration in CT Images Using Radial Image Descriptors." *MICCAI*, vol. 6892, no. Chapter 74, pp. 607–614, 2011. ix, 59, 61

[5] K. Buza, "Feedback prediction for blogs," in *Data analysis, machine learning and knowledge discovery*. Springer, 2014, pp. 145–152. ix, 59, 61

[6] F. Zhou, Q. Claire, and R. D. King, "Predicting the Geographical Origin of Music," in *IEEE International Conference on Data Mining*. IEEE, 2014, pp. 1115–1120. ix, 59, 61

[7] J. Torres-Sospedra, R. Montoliu, A. Martınez-Usó, J. P. Avariento, T. J. Arnau, M. Benedito-Bordonau, and J. Huerta, "UJIIndoorLoc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems," in *Proceedings of the fifth conference on indoor positioning and indoor navigation*, 2014. ix, 60, 62

[8] M. Yamada, G. Niu, and J. Takagi, "Computationally efficient sufficient dimension reduction via squared-loss mutual information," *Asian Conference on Machine Learning*, 2011. xi, 26

[9] L. Sigal, A. O. Balan, and M. J. Black, "HumanEva: Synchronized Video and Motion Capture Dataset and Baseline Algorithm for Evaluation of Articulated Human Motion." *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 87, no. 1-2, pp. 4–27, 2010. xi, 41

[10] J. James. (2014, Apr.) Data never sleeps. [Online]. Available: https://www.domo.com/blog/2014/04/data-never-sleeps-2-0/ 1

[11] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7, pp. 436–444, May 2015. 2

[12] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge University Press, Jun. 2004. 2, 19

[13] A. Gretton, O. Bousquet, A. J. Smola, and B. Schölkopf, "Measuring Statistical Dependence with Hilbert-Schmidt Norms." *ALT*, vol. 3734, no. Chapter 7, pp. 63–77, 2005. 3, 23, 24

[14] G. J. Székely, M. L. Rizzo, and N. K. Bakirov, "Measuring and Testing Dependence by Correlation of Distances," *The annals of statistics*, vol. 35, no. 6, pp. 2769–2794, Dec. 2007. 3, 6, 26, 27, 65

[15] B. Schölkopf, A. J. Smola, B. Taskar, and S. Vishwanathan, "Predicting Structured Data," Tech. Rep., 2006. 3, 11, 21

[16] S. Nowozin and C. H. Lampert, "Structured Learning and Prediction in Computer Vision." *Foundations and Trends in Computer Graphics and Vision*, vol. 6, no. 3-4, pp. 185–365, 2011. 3, 21

[17] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel Methods in Machine Learning," *The annals of statistics*, vol. 36, no. 3, pp. 1171–1220, Jun. 2008. 3

[18] C. Tonde and A. Elgammal, "Simultaneous Twin Kernel Learning using Polynomial Transformations for Structured Prediction," in *IEEE Conference Computer Vision and Pattern Recognition*. IEEE, 2014, pp. 995–1002. 4

[19] ——, "Learning Kernels for Structured Prediction using Polynomial Kernel Transformations," *arXiv.org*, p. arXiv:1601.01411, Jan. 2016. 4

[20] L. Bo and C. Sminchisescu, "Twin Gaussian Processes for Structured Prediction." *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 87, no. 1-2, pp. 28–52, 2010. 4, 21, 23, 34, 46

[21] G. J. Székely, M. L. Rizzo, and N. K. Bakirov, "Measuring and testing dependence by correlation of distances," *arXiv.org*, p. 4101, Mar. 2008. 4

[22] P. Vepakomma, C. Tonde, and A. Elgammal, "Supervised Dimensionality Reduction via Distance Correlation Maximization," *arXiv.org*, p. arXiv:1601.00236, Jan. 2016. 4

[23] I. J. Schoenberg, "Metric Spaces and Completely Monotone Functions," *The Annals of Mathematics*, vol. 39, no. 4, p. 811, Oct. 1938. 6, 30, 31, 32

[24] S. N. Parizi, K. He, S. Sclaroff, and P. F. Felzenszwalb, "Generalized Majorization-Minimization." *CoRR abs/1506.07613*, vol. 1506, 2015. 6, 51, 52, 64, 65

[25] F. Chung, "Lecture notes on spectral graph theory," *AMS Publications*, 1997. 8, 87

[26] P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe, "Convexity, Classification, and Risk Bounds," *Journal of the American Statistical Association*, vol. 101, no. 473, pp. 138–156, Mar. 2006. 9

[27] V. N. Vapnik, *The nature of statistical learning theory*, 2nd ed., ser. Statistics for Engineering and Information Science. New York, NY: Springer-Verlag, New York, 2000. 9, 11

[28] K. Pelckmans, J. A. K. Suykens, and B. De Moor, "Morozov, Ivanov and Tikhonov Regularization Based LS-SVMs," in *Neural Information Processing*. Berlin, Heidelberg: Springer Berlin Heidelberg, Nov. 2004, pp. 1216–1222. 9

[29] A. Globerson, T. Roughgarden, D. Sontag, and C. Yildirim, "How Hard is Inference for Structured Prediction?" *International Conference on Machine Learning*, pp. 2181–2190, 2015. 11

[30] V. Vapnik, *Estimation of dependences based on empirical data*, ser. Information Science and Statistics. Springer, New York, 2006. 11

[31] C. Cortes and V. Vapnik, "Support-Vector Networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, Sep. 1995. 11

[32] O. Chapelle, "Training a Support Vector Machine in the Primal." *Neural Computation*, vol. 19, no. 5, pp. 1155–1178, 2007. 12

[33] C. van den Berg, J. P. R. Christensen, and P. Ressel, *Harmonic Analysis on Semigroups*, ser. Theory of Positive Definite and Related Functions. Springer Science & Business Media, Dec. 2012. 14, 16

[34] N. Aronszajn, "Theory of Reproducing Kernels," vol. 68, no. 3, pp. 337–404, May 1950. 15, 16

[35] J. Mercer, "Functions of positive and negative type, and their connection with the theory of integral equations," *Philosophical transactions of the royal society of . . .*, 1909. 15, 16

[36] G. Wahba, *Spline models for observational data*, ser. CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1990, vol. 59. 18

[37] B. Schölkopf, R. Herbrich, and A. J. Smola, "A Generalized Representer Theorem." *COLT/EuroCOLT*, vol. 2111, no. Chapter 27, pp. 416–426, 2001. 18

[38] B. Schölkopf, K. Tsuda, and J.-P. Vert, *Kernel Methods in Computational Biology*. MIT Press, 2004. 19

[39] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning," 2008. 19

[40] G. S. Kimeldorf and G. Wahba, "A correspondence between Bayesian estimation on stochastic processes and smoothing by splines," *The Annals of Mathematical Statistics*, 1970. 19

[41] I. Steinwart, "On the influence of the kernel on the consistency of support vector machines," *The Journal of Machine Learning Research*, 2002. 19

[42] A. Christmann and I. Steinwart, "Consistency and robustness of kernel-based regression in convex risk minimization," *arXiv.org*, Sep. 2007. 19

[43] ——, "Consistency and Robustness of Kernel-Based Regression in Convex Risk Minimization," *Bernoulli*, vol. 13, no. 3, pp. 799–819, Aug. 2007. 19

[44] C. A. Micchelli, Y. Xu, and H. Zhang, "Universal Kernels," *The Journal of Machine Learning Research*, vol. 7, pp. 2651–2667, Dec. 2006. 19, 20

[45] B. K. Sriperumbudur, K. Fukumizu, and G. R. G. Lanckriet, "Universality, Characteristic Kernels and RKHS Embedding of Measures." *Journal of Machine Learning Research*, vol. 12, pp. 2389–2410, 2011. 20

[46] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. J. Smola, "A Kernel Method for the Two-Sample-Problem." *Advances in neural information processing systems*, pp. 513–520, 2006. 20

[47] B. Taskar and C. Guestrin, "Max-Margin Markov Networks," in *Advances in neural information processing systems*, 2003. 21

[48] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun, "Support vector machine learning for interdependent and structured output spaces." *ICML 2004*, p. 104, 2004. 21

[49] S. Bratieres, N. Quadrianto, and Z. Ghahramani, "Bayesian Structured Prediction Using Gaussian Processes," Jul. 2013. 21

[50] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning*, 2nd ed., ser. Springer Series in Statistics. New York, NY: Springer, New York, 2009. 22

[51] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh, "Clustering with Bregman divergences," vol. 6, pp. 1705–1749, 2005. 23

[52] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A Kernel two-sample Test," *The Journal of Machine Learning Research*, vol. 13, pp. 723–773, Mar. 2012. 25

[53] D. Sejdinovic, B. Sriperumbudur, A. Gretton, and K. Fukumizu, "Equivalence of distance-based and RKHS-based statistics in hypothesis testing," *The annals of statistics*, vol. 41, no. 5, pp. 2263–2291, Oct. 2013. 26, 28, 66

[54] J. G. Szekely, L. M. Rizzo, and K. N. Bakirov, "Measuring and testing dependence by correlation of distances," *Annals of Statistics*, vol. 35, pp. 2769–2794, 2007. 26, 47

[55] G. J. Székely, M. L. Rizzo, and others, "Brownian distance covariance," *The annals of applied statistics*, vol. 3, no. 4, pp. 1236–1265, 2009. 26

[56] G. J. Székely and M. L. Rizzo, "On the uniqueness of distance covariance," *Statistics and Probability Letters*, vol. 82, no. 12, pp. 2278–2282, 2012. 26

[57] ——, "The distance correlation t-test of independence in high dimension," *Journal of Multivariate Analysis*, vol. 117, pp. 193–213, 2013. 26

[58] S. Bochner, *Lectures on Fourier Integrals*. Princeton University Press, 1959. 30

[59] A. J. Smola, Z. L. Óvári, and R. C. Williamson, "Regularization with Dot-Product Kernels." *Advances in neural information processing systems*, pp. 308–314, 2000. 30, 35

[60] G. Szegö, *Orthogonal Polynomials*. American Mathematical Soc., 1939. 31

[61] D. Gottlieb, C.-W. Shu, A. Solomonoff, and H. Vandeven, "On the Gibbs phenomenon. I. Recovering exponential accuracy from the Fourier partial sum of a nonperiodic analytic function," *Journal of Computational and Applied Mathematics*, vol. 43, no. 1-2, pp. 81–98, 1992. 31, 33, 35, 40

[62] C. A. Micchelli and M. Pontil, "Learning the Kernel Function via Regularization." pp. 1099–1125, 2005. 34

[63] A. Argyriou, C. A. Micchelli, and M. Pontil, "Learning Convex Combinations of Continuously Parameterized Basic Kernels," *Conference on Learning Theory*, vol. 3559, no. 23, pp. 338–352, 2005. 34, 35

[64] A. Argyriou, R. Hauser, C. A. Micchelli, and M. Pontil, "A DC-programming algorithm for kernel selection." *International Conference on Machine Learning*, pp. 41–48, 2006. 34, 35

[65] A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet, "SimpleMKL," *The Journal of Machine Learning Research*, vol. 9, pp. 2491–2521, 2008. 35

[66] P. Gehler and S. Nowozin, "Infinite kernel learning," *NIPS Workshop on Kernel Learning Automatic Selection of Optimal Kernels*, 2008. 35

[67] S. K. G. Shirazi, R. Safabakhsh, and M. Shamsi, "Learning Translation Invariant Kernels for Classification." *Journal of Machine Learning Research*, vol. 11, pp. 1353–1390, 2010. 34, 35

[68] S. Özögür-Akyüz and G.-W. Weber, "On numerical optimization theory of infinite kernel learning." *Journal of Global Optimization*, vol. 48, no. 2, pp. 215–239, 2010. 34

[69] T. Suzuki and M. Sugiyama, "Multiple Kernel Learning Algorithms." *Journal of Machine Learning Research*, pp. 2211–2268, 2011. 35

[70] C. Cortes, M. Mohri, and A. Rostamizadeh, "Algorithms for Learning Kernels Based on Centered Alignment," no. 1, pp. 550–828, Mar. 2012. 35, 36, 38, 46

[71] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, and J. S. Kandola, "On Kernel-Target Alignment." *Advances in neural information processing systems*, pp. 367–373, 2001. 35

[72] K. C. Chang, K. Pearson, and T. Zhang, "Perron-Frobenius theorem for nonnegative tensors," *Communications in Mathematical Sciences*, vol. 6, no. 2, pp. 507–520, Jun. 2008. 38, 77

[73] B. Chang, U. Kruger, R. Kustra, and J. Zhang, "Canonical Correlation Analysis based on Hilbert-Schmidt Independence Criterion and Centered Kernel Target Alignment," pp. 316–324, 2013. 38

[74] C. M. Bishop and M. Svensén, "Bayesian Hierarchical Mixtures of Experts." *UAI*, pp. 57–64, 2003. 40

[75] SmithMicro Software. Poser 3D Animation & Character Creation Software . [Online]. Available: http://poser.smithmicro.com/ 40

[76] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection." *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 886–893, 2005. 41

[77] H. Guo, X. Zhu, and M. R. Min, "A Deep Learning Model for Structured Outputs with High-order Interaction." *CoRR abs/1506.07613*, vol. cs.LG, 2015. 43

[78] K.-C. Li, "Sliced inverse regression for dimension reduction," *Journal of the American Statistical Association*, vol. 86, no. 414, pp. 316–327, 1991. 47, 60, 61, 62

[79] H. H. Lue, "Sliced inverse regression for multivariate response regression," *Journal of Statistical Planning and Inference*, vol. 139, pp. 2656–2664, 2009. 47

[80] M. E. Szretter and V. J. Yohai, "The sliced inverse regression algorithm as a maximum likelihood procedure," *Journal of Statistical Planning and Inference*, vol. 139, pp. 3570–3578, 2009. 47

[81] R. D. Cook, "Graphics for regressions with a binary response," *Journal of the American Statistical Association*, vol. 91, no. 435, pp. 983–992, 1996. 47

[82] Y. Shao, R. D. Cook, and S. Weisberg, "Partial central subspace and sliced average variance estimation," *Journal of Statistical Planning and Inference*, vol. 139, pp. 952–961, 2009. 47, 60, 61, 62

[83] ——, "Marginal Tests with Sliced Average Variance Estimation," *Biometrika*, vol. 94, pp. 285–296, 2007. 47, 60

[84] R. D. Cook and L. Forzani, "Likelihood Based Sufficient Dimension Reduction," *Journal of the American Statistical Association*, vol. 104, pp. 197–208, 2009. 47, 60, 61, 62

[85] K. Fukumizu and C. Leng, "Gradient-based kernel dimension reduction for regression," *Journal of the American Statistical Association*, vol. 109, no. 505, pp. 359–370, 2014. 48, 60, 61, 62

[86] M. Yamada, G. Niu, J. Takagi, and M. Sugiyama, "Sufficient Component Analysis for Supervised Dimension Reduction," *arXiv preprint arXiv:1103.4998*, 2011. 48, 60, 61, 62

[87] M. Sugiyama, T. Suzuki, and T. Kanamori, *Density Ratio Estimation in Machine Learning*, 1st ed. New York, NY, USA: Cambridge University Press, 2012. 48

[88] V. Vapnik, I. Braga, and R. Izmailov, "Constructive setting for problems of density ratio estimation," *Statistical Analysis and Data Mining: The ASA Data Science Journal*, vol. 8, no. 3, pp. 137–146, 2015. 48

[89] T. Suzuki and M. Sugiyama, "Sufficient dimension reduction via squared-loss mutual information estimation," *Neural Computation*, vol. 25, no. 3, pp. 725–758, 2013. 48, 60, 61, 62

[90] S.-I. Amari, "Natural gradient works efficiently in learning," *Neural Computation*, vol. 10, no. 2, pp. 251–276, 1998. 48

[91] Y. Nishimori and S. Akaho, "Learning algorithms utilizing quasi-geodesic flows on the Stiefel manifold." *Neurocomputing*, vol. 67, pp. 106–135, 2005. 48

[92] R. Li, W. Zhong, and L. Zhu, "Feature screening via distance correlation learning," *Journal of the American Statistical Association*, vol. 107, no. 499, pp. 1129–1139, 2012. 48

[93] J. Kong, S. Wang, and G. Wahba, "Using distance covariance for improved variable selection with application to learning genetic risk models," *Statistics in medicine*, vol. 34, no. 10, pp. 1708–1720, 2015. 48

[94] J. R. Berrendero, A. Cuevas, and J. L. Torrecilla, "Variable selection in functional data classification: a maxima-hunting proposal," *Statistica Sinica*, 2014. 48

[95] W. S. Torgerson, "Multidimensional scaling: I. Theory and method," *Psychometrika*, vol. 17, no. 4, pp. 401–419, 1952. 49, 87

[96] I. Borg and P. J. F. Groenen, *Modern multidimensional scaling: Theory and applications*. Springer Science and Business Media, 2005. 49

[97] K. Lange, D. R. Hunter, and I. Yang, "Optimization Transfer Using Surrogate Objective Functions," *Journal of Computational and Graphical Statistics*, vol. 9, no. 1, p. 1, Mar. 2000. 51

[98] S. Schaible, "Minimization of ratios," *Journal of Optimization Theory and Applications*, vol. 19, no. 2, pp. 347–352, 1976. 53

[99] W. Dinkelbach, "On nonlinear fractional programming," *Management Science. Journal of the Institute of Management Science. Application and Theory Series*, vol. 13, no. 7, pp. 492–498, 1967. 53, 54, 55

[100] A. Zhang, "Quadratic Fractional Programming Problems with Quadratic Constraints," Ph.D. dissertation, Kyoto University, 2008. 53, 54, 55

[101] J. Kiefer, "Sequential minimax search for a maximum," *Proceedings of the American Mathematical Society*, vol. 4, no. 3, pp. 502–506, 1953. 54

[102] K. Lange, "The MM Algorithm," in *Optimization*. Springer New York, 2013, pp. 185–219. 54, 55, 56

[103] M. Lichman. (2013) UCI Machine Learning Repository. [Online]. Available: http://archive.ics.uci.edu/ml 59

[104] Y. Zhang, R. Tapia, and L. Velazquez, "On convergence of minimization methods: attraction, repulsion, and selection," *Journal of Optimization Theory and Applications*, vol. 107, no. 3, pp. 529–546, 2000. 86, 87

Appendices

Appendix A

Learning Polynomial Kernel Transformations for Structured Prediction

A.1 Mathematical Results

Theorem 29 (Perron-Frobenius [72])**.** *For* $\mathbf{A}_{n\times n} \geq 0$*, with spectral radius* $r = \rho(\mathbf{A})$*, the following statements are true.*

1. $r \in \sigma(\mathbf{A})$ *and* $r > 0$
2. r *is unique and it the spectral radius of* $\mathbf{A}$
3. $\mathbf{Az} = r\mathbf{z}$ *for some* $\mathbf{z} \in \mathbf{\Delta}^n = \{\mathbf{x}|\mathbf{x} \geq 0$ *with* $\mathbf{x} \neq \mathbf{0}\}$
4. *There is unique vector defined by*

$$\mathbf{Ap} = r\mathbf{p}, \mathbf{p} > 0, \text{ and } \|\mathbf{p}\|_1 = 1, \tag{A.1}$$

is called Perron vector of $\mathbf{A}$, and there are no other nonnegative vectors except for positive multiples of $\mathbf{p}$, regardless of eigenvalue.

A.2 Kernel Gradients

For data points $X = \{x_1, x_2, \dots x_m\}$ and test data point x.

$$\frac{\partial \phi(K(x_1, x))}{\partial x^{(d)}} = \frac{\partial \phi(t)}{\partial t}|_{t=K(x_1,x)} \frac{\partial K(x_1, x)}{\partial x^{(d)}}$$

$$\frac{\partial \phi(t)}{\partial t} = \sum_{i=0}^{d_1} \alpha_i H_i^{\gamma}(t)$$

$$H_0^{\gamma}(t) = 0, H_1^{\gamma}(t) = 2\gamma, \tag{A.2}$$

$$H_{i+1}^{\gamma}(t) = \left(\frac{2(\gamma + i)}{i+1}\right)(tH_i^{\gamma}(t) + G_i^{\gamma}(t)) - \left(\frac{2\gamma + i - 1}{i+1}\right) H_{i-1}^{\gamma}(t) \tag{A.3}$$

$$\frac{\partial \phi(K(X, x))}{\partial x^{(d)}} = \begin{bmatrix} \frac{\partial \phi(t)}{\partial t}|_{t=K(x_1,x)} \frac{\partial K(x_1,x)}{\partial x^{(d)}} \\ \frac{\partial \phi(t)}{\partial t}|_{t=K(x_2,x)} \frac{\partial K(x_2,x)}{\partial x^{(d)}} \\ \vdots \\ \frac{\partial \phi(t)}{\partial t}|_{t=K(x_m,x)} \frac{\partial K(x_m,x)}{\partial x^{(d)}} \end{bmatrix}$$

RBF kernel

$$K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$$

$$\frac{\partial K(X, x)}{\partial x^{(d)}} = \begin{bmatrix} -2\gamma(-x_1^{(d)} + x^{(d)})K(x, x_1) \\ -2\gamma(-x_2^{(d)} + x^{(d)})K(x, x_2) \\ \vdots \\ -2\gamma(-x_m^{(d)} + x^{(d)})K(x, x_m) \end{bmatrix}$$

$$\frac{\partial K(x_1, x)}{\partial x^{(d)}} = -2\gamma(-x_1^{(d)} + x^{(d)})K(x, x_1)$$

Linear kernel

$$K(x_i, x_j) = \gamma \langle x_i, x_j \rangle$$

$$\frac{\partial K(X,x)}{\partial x^{(d)}} = \begin{bmatrix} \gamma x_1^{(d)}) \\ \gamma x_2^{(d)} \\ \vdots \\ \gamma x_m^{(d)} \end{bmatrix}$$

Spectral kernel $\mathbf{x} \in \mathbb{R}^D$ and $\tau_k = |x - x^k| \in \mathbb{R}$

$$k(\tau) = \prod_{d=1}^{D} \exp\left(-2\pi^2 \tau_d^2 \sigma_d^2\right) \cos\left(2\pi \mu_d \tau_d\right) \tag{A.4}$$

- The inverse means $1/\mu_d$ represent the component periods
- The inverse standard deviations $1/\sqrt{\sigma_d}$ represent length scales

$$\frac{\partial k(\tau)}{\partial x_k} = \exp\left(-2\pi^2 \left(\sum_{d=1}^{D} \tau_d^2 \sigma_d^2\right)\right) \left((-4\pi^2 \sigma_k^2 \tau_k) \prod_{d=1}^{D} \cos\left(2\pi \mu_d \tau_d\right) - 2\pi \mu_k \tau_k \sin\left(2\pi \mu_k \tau_k\right) \prod_{d=1, d\neq k}^{D} \cos\left(2\pi \mu_d \tau_d\right)\right) \tag{A.5}$$

$$\frac{\partial k(\tau)}{\partial x_k} = (2\pi \tau_k) \exp\left(-2\pi^2 \left(\sum_{d=1}^{D} \tau_d^2 \sigma_d^2\right)\right) \left((2\pi \sigma_k^2) \prod_{d=1}^{D} \cos\left(2\pi \mu_d \tau_d\right) + \mu_k \sin\left(2\pi \mu_k \tau_k\right) \prod_{d=1, d\neq k}^{D} \cos\left(2\pi \mu_d \tau_d\right)\right) \tag{A.6}$$

Spectral mixture kernel Mixture of Q spectral kernels.

$$k(\tau) = \sum_{i=1}^{Q} w_i \prod_{d=1}^{D} \exp\left(-2\pi^2 \tau_d^2 \sigma_d^2\right) \cos\left(2\pi \mu_d \tau_d\right) \tag{A.7}$$

A.3 Additional Experimental Results

Crit. / Mean Abs. Er	(no mapping)	(mapping)	Gain %
KL-Div (11,11)	0.2151	0.21078	**2.008** %
HSIC (11,11)	0.3399	0.19536	**2.5007** %
KL-Div (Gegen.) (11,11)	0.2101	0.19536	**7.0096** %
HSIC (Gegen.) (23,23)	0.3399	0.25442	**25.1441%** %

Table A.1: Mean Absolute Error for USPS Handwritten digits dataset for two criteria's with and without mapping.

Features	Motions	Subject 1			Subject 2			Subject 3		
		TGP	HOTGP	% Gain	TGP	HOTGP	% Gain	TGP	HOTGP	% Gain
HoG (C1C2C3)	Walking	41.0491	38.0943	7.1984	27.3651	25.1213	8.1998	49.5457	46.8701	5.4003
	Jog	48.3995	47.1128	2.6585	37.7046	35.4643	5.9417	39.8883	37.3239	6.4288
	Gestures	16.3853	14.5804	11.0155	46.6891	44.3624	4.9833	60.9779	59.2251	2.8745
	Box	39.2996	37.2232	5.2837	48.1626	45.4181	5.6983	44.3865	41.5619	6.3637
	ThrowCatch	84.0748	81.9438	2.5346	48.492	45.9944	5.1504	/	/	/
	Average	45.8417	43.7909	5.7381	41.6827	39.2721	5.9947	48.6996	46.2453	5.2668
HoG (C1)	Walking	30.7786	29.7702	3.2765	20.2406	19.4853	3.7317	38.259	37.2519	2.6324
	Jog	35.2833	34.3592	2.6191	27.5999	26.6086	3.5918	26.9919	26.0356	3.5429
	Gestures	5.3718	5.2444	2.3714	39.9867	38.8397	2.8685	43.4735	42.8049	1.5378
	Box	26.9904	26.3226	2.4743	40.9108	39.764	2.8032	32.2198	31.3732	2.6276
	ThrowCatch	70.661	69.4619	1.6971	41.2717	40.6461	1.5159	/	/	/
	Average	33.817	33.0317	2.4876	34.0019	33.0687	2.9022	35.236	34.3664	2.5852
HoG (C2)	Walking	25.6649	24.9178	2.9112	17.4669	16.883	3.3428	32.0536	31.4839	1.7774
	Jog	36.1562	35.8948	0.7229	25.0174	24.4919	2.1002	22.5577	22.0753	2.1384
	Gestures	8.2509	7.8924	4.345	41.0384	40.7589	0.6812	37.409	37.5382	-0.3456
	Box	25.952	25.4622	1.8874	33.5624	33.103	1.3689	35.4483	34.9741	1.3379
	ThrowCatch	69.2615	68.8563	0.5849	38.06	37.7767	0.7443	/	/	/
	Average	33.0571	32.6047	2.0903	31.029	30.6027	1.6475	31.8672	31.5179	1.227
HoG (C3)	Walking	26.4685	25.6706	3.0143	16.2316	15.5688	4.0835	33.3515	32.5203	2.4922
	Jog	34.4813	34.0262	1.3199	26.0008	25.2859	2.7493	22.5078	22.0039	2.2388
	Gestures	9.5843	9.2134	3.8701	36.7243	36.3498	1.0198	35.8515	37.286	-4.0013
	Box	25.9197	25.425	1.9087	29.6339	29.4589	0.5906	30.8706	30.269	1.949
	ThrowCatch	67.1347	65.7422	2.0743	38.2297	38.0795	0.3929	/	/	/
	Average	32.7177	32.0155	2.4375	29.3641	28.9486	1.7672	30.6453	30.5198	0.6696
Total (% Gain)		HoG (C1+C2+C3) : **5.0796**			HoG (C1) : **2.5147**			HoG (C2) : **1.2928**	HoG (C3) : **1.4067**	

Table A.2: Evaluation using HoG features on HumanEva-I. Positive *% Gain* for each subject is shown in **bold**, and in red otherwise. Columns TGP and HOTGP indicate the mean absolute error while the *% Gain* column indicates the percentage reduction on error.

Features	Motions	Subject 1			Subject 2			Subject 3		
		HSIC	HOHSIC	% Gain	HSIC	HOHSIC	% Gain	HSIC	HOHSIC	% Gain
HoG (C1C2C3)	Walking	159.6824	159.406	0.1730	175.6263	175.6051	0.0120	200.1092	200.1722	-0.0315
	Jog	182.5472	182.6827	-0.0742	196.0942	196.2373	-0.0729	204.6334	204.7445	-0.0542
	Gestures	121.9071	120.8305	0.8831	128.1372	127.128	0.78759	161.8518	160.9014	0.5872
	Box	150.7509	150.3133	0.29026	190.5383	189.7172	0.4309	194.1369	193.4358	0.3611
	ThrowCatch	188.2296	188.4557	-0.1201	145.4743	144.7525	0.4961	/	/	/
	Average	160.6234	160.5714	0.0839	167.1740	167.0846	0.0808	190.1828	190.1161	0.0888
HoG (C1)	Walking	159.6824	159.4031	0.17489	175.6263	175.6204	0.0033	200.1092	200.1728	-0.0318
	Jog	182.5472	182.7127	-0.0906	196.0942	196.2703	-0.0898	204.6334	204.7865	-0.0748
	Gestures	121.9071	120.829	0.88431	128.1372	127.0738	0.82991	161.8518	160.9037	0.5857
	Box	150.7509	150.3093	0.29293	190.5383	189.7167	0.4312	194.1369	193.4387	0.3596
	ThrowCatch	188.2296	188.3374	-0.0572	145.4743	144.6998	0.53246	/	/	/
	Average	160.6234	160.5714	0.0839	167.1740	167.0846	0.0808	190.1828	190.1161	0.0888
HoG (C2)	Walking	159.6824	159.4017	0.17574	175.6263	175.6094	0.0096	200.1092	200.1662	-0.0285
	Jog	182.5472	182.6946	-0.0807	196.0942	196.2758	-0.0926	204.6334	204.8042	-0.0834
	Gestures	121.9071	120.8318	0.88202	128.1372	127.0976	0.81134	161.8518	160.8771	0.6022
	Box	150.7509	150.3304	0.27895	190.5383	189.7168	0.43115	194.1369	193.4401	0.3589
	ThrowCatch	188.2296	188.3753	-0.0774	145.4743	144.6909	0.53856	/	/	/
	Average	160.6234	160.3267	0.2357	167.1740	166.6781	0.3396	190.1828	189.8219	0.2123
HoG (C3)	Walking	159.6824	159.4267	0.16009	175.6263	175.6623	-0.0204	200.1092	200.2031	-0.0469
	Jog	182.5472	182.744	-0.1078	196.0942	196.2974	-0.10361	204.6334	204.8073	-0.0849
	Gestures	121.9071	120.8289	0.88443	128.1372	127.0813	0.82406	161.8518	160.9025	0.5865
	Box	150.7509	150.3118	0.29129	190.5383	189.716	0.43159	194.1369	193.4405	0.3587
	ThrowCatch	188.2296	188.3351	-0.0560	145.4743	144.7051	0.52877	/	/	/
	Average	160.6234	160.3293	0.2343	167.1741	166.6924	0.3321	190.1828	189.838	0.2033
Total (% Gain)		HoG (C1+C2+C3) : **0.2566**			HoG (C1) : **0.2589**			HoG (C2) : **0.2589**		HoG (C3) : **0.2625**

Table A.3: Evaluation using HoG features on HumanEva-I. Positive *% Gain* for each subject is shown in **bold**, and in red otherwise. Columns HSIC and HOHSIC (Gegen.) indicate the mean absolute error while the *% Gain* column indicates the percentage reduction on error.

Features	Motions	Subject 1			Subject 2			Subject 3		
		HSIC	HOHSIC	% Gain	HSIC	HOHSIC	% Gain	HSIC	HOHSIC	% Gain
HoG (C1C2C3)	Walking	159.6824	159.6333	0.03069	175.6263	175.6285	-0.0012	200.1092	200.1266	-0.0087
	Jog	182.5472	182.5801	-0.0180	196.0942	196.1282	-0.0174	204.6334	204.6587	-0.0123
	Gestures	121.9071	121.7042	0.1664	128.1372	127.9456	0.14957	161.8518	161.6735	0.11017
	Box	150.7509	150.6684	0.0547	190.5383	190.3837	0.0811	194.1369	194.0057	0.0676
	ThrowCatch	188.2296	188.2713	-0.0221	145.4743	145.3374	0.0941	/	/	/
	Average	160.6234	160.5714	0.0839	167.1740	167.0846	0.0808	190.1828	190.1161	0.0888
HoG (C1)	Walking	159.6824	159.6326	0.0311	175.6263	175.6299	-0.0020	200.1092	200.1262,	-0.0085
	Jog	121.9071	121.704	0.1665	128.1372	127.9364	0.15667	161.8518	161.6735	0.11017
	Gestures	182.5472	182.5845	-0.0204	196.0942	196.1347	-0.0206	204.6334	204.6659	-0.0158
	Box	150.7509	150.6677	0.0551	190.5383	190.3838	0.08113	194.1369	194.0059	0.0674
	ThrowCatch	188.2296	188.2513	-0.0115	145.4743	145.3287	0.1001	/	/	/
	Average	160.6234	160.5680	0.0843	167.1740	167.0827	0.1126	190.1828	194.0059	0.0888
HoG (C2)	Walking	159.6824	159.6331	0.0308	175.6263	175.6278	-0.0009	200.1092	200.1257	-0.0082
	Jog	121.9071	121.7045	0.1661	128.1372	127.9402	0.15374	161.8518	161.6687	0.11317
	Gestures	182.5472	182.581	-0.0184	196.0942	196.1343	-0.0204	204.6334	204.6689	-0.0173
	Box	150.7509	150.6713	0.0527	190.5383	190.3839	0.0810	194.1369	194.0059	0.0674
	ThrowCatch	188.2296	188.2578	-0.0149	145.4743	145.3272	0.1011	/	/	/
	Average	160.6234	160.5695	0.0832	167.1740	167.0826	0.1120	190.1828	190.1173	0.0903
HoG (C3)	Walking	159.6824	159.6374	0.0281	175.6263	175.6382	-0.0067	200.1092	200.1319	-0.0113
	Jog	121.9071	121.704	0.1666	128.1372	127.9402	0.15374	161.8518	161.6735	0.11017
	Gestures	182.5472	182.5894	-0.0231	196.0942	196.1389	-0.0227	204.6334	204.6695	-0.0176
	Box	150.7509	150.6681	0.0548	190.5383	190.3837	0.08115	194.1369	194.0059	0.0674
	ThrowCatch	188.2296	188.2506	-0.0111	145.4743	145.3293	0.0997	/	/	/
	Average	160.6234	160.5699	0.0430	167.1740	167.0860	0.0610	190.1828	190.1202	0.0372
Total (% Gain)		HoG (C1+C2+C3) : **0.0471**			HoG (C1) : **0.0846**			HoG (C2) : **0.0952**		HoG (C3) : **0.0952**

Table A.4: Evaluation using HoG features on HumanEva-I. Positive *% Gain* for each subject is shown in **bold**, and in red otherwise. Columns HSIC and HOHSIC indicate the mean absolute error while the *% Gain* column indicates the percentage reduction on error.

Features	Motions	Subject 1			Subject 2			Subject 3		
		TGP	HOTGP	% Gain	TGP	HOTGP	% Gain	TGP	HOTGP	% Gain
HoG (C1C2C3)	Walking	37.4366	9.5482	74.4951	21.7661	6.0762	72.084	44.4345	9.7506	78.0562
	Jog	8.6656	2.3434	72.957	60.314	11.9035	80.2642	48.5744	9.0458	81.3775
	Gestures	49.8854	10.5647	78.822	52.4706	10.9267	79.1756	35.704	9.1127	74.4771
	Box	38.3083	9.5265	75.1321	52.3877	11.4781	78.0901	48.8305	11.0568	77.3567
	ThrowCatch	63.9985	11.78	81.5933	59.1893	11.6716	80.2808	/	/	/
	Average	39.6588	8.7525	76.5999	49.2255	10.4112	77.9789	44.3858	9.7414	77.8168
HoG (C1)	Walking	36.9087	9.7608	73.5541	21.6479	9.931	54.125	44.895	10.94	75.6321
	Jog	9.4207	2.5032	73.4284	60.7753	11.8101	80.5676	51.9379	32.982	36.4973
	Gestures	49.1918	10.6199	78.4112	52.5459	10.9897	79.0856	35.9421	9.2451	74.2779
	Box	38.042	9.5071	75.0089	52.8592	11.6581	77.945	48.8239	13.3002	72.7589
	ThrowCatch	63.0999	11.7392	81.3959	58.7804	12.8523	78.1351	/	/	/
	Average	39.3326	8.82604	76.3597	49.3217	11.4482	73.9716	45.3997	16.6168	64.7915
HoG (C2)	Walking	37.8241	0.0109	99.9711	20.1582	0.0097	99.9518	42.5441	0.0115	99.9729
	Jog	8.8519	0.0068	99.9231	59.6685	0.0108	99.9817	51.9699	0.0114	99.978
	Gestures	48.517	0.0118	99.9756	53.1385	0.0100	99.9811	35.9633	0.0112	99.9687
	Box	36.7858	0.0098	99.9733	59.3451	0.0110	99.9813	49.0821	0.0108	99.9779
	ThrowCatch	64.2616	0.0141	99.9779	52.2494	0.0097	99.9813	/	/	/
	Average	39.2480	0.0107	99.9642	48.9119	0.0103	99.9754	44.8898	0.0112	99.9743
HoG (C3)	Walking	21.4906	0.0108	99.9497	12.4445	0.0095	99.9236	25.0639	0.0118	99.9528
	Jog	4.3108	0.0061	99.8604	34.9702	0.0103	99.9703	29.1524	0.0115	99.9604
	Gestures	28.7849	0.0119	99.9587	29.9167	0.0097	99.9674	19.6575	0.0112	99.9429
	Box	22.2626	0.0094	99.9578	30.5953	0.0101	99.967	28.0667	0.0099	99.9645
	ThrowCatch	35.3846	0.0142	99.9599	33.3681	0.0103	99.969	/	/	/
	Average	22.4467	0.0104	99.9373	28.2589	0.0100	99.9594	25.4851	0.0111	99.9551
Total (% Gain)		HoG (C1+C2+C3) : **99.9506**			HoG (C1) : **77.4652**			HoG (C2) : **71.7076**		HoG (C3):**99.9713**

Table A.5: Evaluation using HoG features on HumanEva-I. Positive *% Gain* for each subject is shown in **bold**, and in red otherwise. Columns TGP and HOTGP (Gegen-(1,5)) indicate the mean absolute error while the *% Gain* column indicates the percentage reduction on error.

Appendix B

Supervised Dimensionality Reduction via Distance Correlation Maximization

B.1 Spectral Radius of Fixed-Point Iterate $T(\mathbf{Z}_t)$

To prove Lemma 33, required for proving convergence in Theorem 26, we need to show that the spectral radius $\lambda_{max}(\mathbf{H}) < 1$. We show this in Theorem 32 and proceed to prove it by first by proving two required lemmas below.

Lemma 30. *For any choice of* $\gamma^2 > \lambda_{max}(\mathbf{D_X}, \mathbf{L_M})$ *and* $\mathbf{P} := \left(\gamma^2\mathbf{D_X} - \mathbf{L_M}\right)$*, we have* $\mathbf{P} \succeq 0$.

Proof. To show $\mathbf{z}^T(\gamma^2\mathbf{D_X} - \mathbf{L_M})\mathbf{z} \geq 0$ for all $\mathbf{z}$, we require that $\gamma^2 \geq \frac{\mathbf{z}^T\mathbf{L_M}\mathbf{z}}{\mathbf{z}^T\mathbf{D_X}\mathbf{z}}$ for all $\mathbf{z}$. This is always true for all values of $\gamma^2 \geq \lambda_{max}(\mathbf{D_X}, \mathbf{L_M})$. □

Lemma 31. *If* $0 = \alpha_l \leq \alpha \leq \alpha_u = \lambda_{min}(\mathbf{L_M}, \mathbf{S_{X,y}})$ *and* $\mathbf{Q} := (\mathbf{L_M} - \alpha\mathbf{S_{X,y}})$*, then we have* $\mathbf{Q} \succeq 0$.

Proof. To show $\mathbf{z}^T(\mathbf{L_M} - \alpha\mathbf{S_{X,y}})\mathbf{z} \geq 0$ for all $\mathbf{z}$, we require that $\alpha \leq \frac{\mathbf{z}^T\mathbf{L_M}\mathbf{z}}{\mathbf{z}^T\mathbf{S_{X,y}}\mathbf{z}}$ for all $\mathbf{z}$. This is always true if all values of $\alpha \leq \min_{\mathbf{Z}} \frac{\mathbf{z}^T\mathbf{L_M}\mathbf{z}}{\mathbf{z}^T\mathbf{S_{X,y}}\mathbf{z}} = \lambda_{min}(\mathbf{L_M}, \mathbf{S_{X,y}})$ which is true by our choice of α. □

We now utilize the above to results to prove $\lambda_{max}(\mathbf{H}) \leq 1$ about the fixed point iterate $\mathbf{Z}_{t+1} = \mathbf{H}\mathbf{Z}_t$.

Theorem 32. *For the update equation* $\mathbf{Z}_{t+1} = \mathbf{H}\mathbf{Z}_t$ *with*

$$\mathbf{H} = \left(\gamma^2\mathbf{D_X} - \alpha\mathbf{S_{X,y}}\right)^{\dagger}\left(\gamma^2\mathbf{D_X} - \mathbf{L_M}\right),$$

we have $\lambda_{max}(\mathbf{H}) \leq 1$.

Proof. The update equation looks as follows

$$\mathbf{Z}_{t+1} = \left(\gamma^2\mathbf{D}_{\mathbf{X}} - \alpha\mathbf{S}_{\mathbf{X},\mathbf{y}}\right)^{\dagger}\left(\gamma^2\mathbf{D}_{\mathbf{X}} - \mathbf{L}_{\mathbf{M}}\right)\mathbf{Z}_t.$$

For sake of simplicity assume $\mathbf{P} = \left(\gamma^2\mathbf{D}_{\mathbf{X}} - \mathbf{L}_{\mathbf{M}}\right)$ and $\mathbf{Q} = (\mathbf{L}_{\mathbf{M}} - \alpha\mathbf{S}_{\mathbf{X},\mathbf{y}})$.

$$\mathbf{Z}_{t+1} = (\mathbf{P}+\mathbf{Q})^{-1}\mathbf{P}\mathbf{Z}_t$$

Using the Woodbury matrix identity $(\mathbf{A}+\mathbf{UBV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{B}^{-1}+\mathbf{VA}^{-1}\mathbf{U})^{-1}\mathbf{VA}^{-1}$, and setting $\mathbf{U} = \mathbf{I}$ and $\mathbf{V} = \mathbf{I}$, we get, $(\mathbf{A}+\mathbf{B})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}(\mathbf{B}^{-1}+\mathbf{A}^{-1})^{-1}\mathbf{A}^{-1}$. Applying this to the previous equation we get

$$\begin{aligned}\mathbf{Z}_{t+1} &= (\mathbf{P}^{-1} - \mathbf{P}^{-1}(\mathbf{P}^{-1}+\mathbf{Q}^{-1})^{-1}\mathbf{P}^{-1})\mathbf{P}\mathbf{Z}_t = \mathbf{I} - \mathbf{P}^{-1}(\mathbf{P}^{-1}+\mathbf{Q}^{-1})^{-1}\mathbf{Z}_t \\ &= \mathbf{I} - \mathbf{P}^{-1}\left((\mathbf{P}^{-1}+\mathbf{Q}^{-1})^{-1}\mathbf{Q}^{-1}\right)\mathbf{Q}\mathbf{Z}_t\end{aligned}$$

Using the positive definite identity $(\mathbf{P}^{-1}+\mathbf{B}^T\mathbf{Q}^{-1}\mathbf{B})^{-1}\mathbf{B}^T\mathbf{Q}^{-1} = \mathbf{P}\mathbf{B}^T(\mathbf{B}\mathbf{P}\mathbf{B}^T+\mathbf{Q})^{-1}$ for $\mathbf{B} = \mathbf{I}$ we get, $(\mathbf{P}^{-1}+\mathbf{Q}^{-1})^{-1}\mathbf{Q}^{-1} = \mathbf{P}(\mathbf{P}+\mathbf{Q})^{-1}$, which simplifies the term in the brackets as,

$$\mathbf{Z}_{t+1} = \mathbf{I} - \mathbf{P}^{-1}\left(\mathbf{P}(\mathbf{P}+\mathbf{Q})^{-1}\right)\mathbf{Q}\mathbf{Z}_t = \mathbf{I} - (\mathbf{P}+\mathbf{Q})^{-1}\mathbf{Q}\mathbf{Z}_t$$

If we compare the above equation with a the general update equation from Zhang et al. [104], which is of the form

$$T(\mathbf{Z}_{t+1}) = \mathbf{Z}_t - \beta(\mathbf{Z}_t)\mathbf{B}(\mathbf{Z}_t)^{-1}\nabla f(\mathbf{Z}_t)$$

where $\nabla f(\mathbf{Z}_t)$ is the gradient of the objective function $f(\mathbf{Z})$ we get,

$$\beta(\mathbf{Z}_t) = \frac{1}{2}, \qquad \mathbf{B}(\mathbf{Z}_t) = \mathbf{P}+\mathbf{Q}, \qquad \nabla f(\mathbf{Z}_t) = 2\mathbf{Q}\mathbf{Z}_t$$

Now from Theorem 30 we conclude that $\mathbf{B}(\mathbf{Z}) \succeq 0$, We also check the following condition

from Zhang et al. [104] that

$$0 \preceq \nabla^2 f(\mathbf{Z}) \preceq \frac{2\mathbf{B}}{\beta}.$$

or equivalently, as in our case $0 \preceq 2\mathbf{Q} \preceq 4(\mathbf{Q}+\mathbf{P})$, which is indeed true. Hence it follows that $\lambda_{max}(T'(\mathbf{Z})) \leq 1$ which implies $\lambda_{max}(\mathbf{H}) \leq 1$. □

We now proceed to show that at end of every $(t+1)$ fixed point iterations we have

$$\mathbf{Tr}\left(\mathbf{Z}_{t+1}^T \mathbf{L}_{\mathbf{Z}_{t+1}} \mathbf{Z}_{t+1}\right) \leq \mathbf{Tr}\left(\mathbf{Z}_{t+1} \mathbf{L}_{\mathbf{Z}_0} \mathbf{Z}_{t+1}\right).$$

Lemma 33. *For fixed point iteration $\mathbf{Z}_{t+1} = \mathbf{H}\mathbf{Z}_t$ for optimization of $\mathbf{Z}_{k+1} = \arg\max_{\mathbf{Z}} g(\mathbf{Z}, \mathbf{Z}_k)$, we have,* $\mathbf{Tr}\left(\mathbf{Z}_{k+1}^T \mathbf{L}_{\mathbf{Z}_{k+1}} \mathbf{Z}_{k+1}\right) \leq \mathbf{Tr}\left(\mathbf{Z}_{k+1} \mathbf{L}_{\mathbf{Z}_k} \mathbf{Z}_{k+1}\right)$.

Proof. Laplacian for a weighted adjacency matrix $\mathbf{W}$ (with self loops) is defined as $\mathbf{L} = \mathbf{D} - \mathbf{W}$ where $\mathbf{D}$ is a diagonal degree matrix with diagonal elements $[\mathbf{D}]_{i,i} = \sum_j [\mathbf{W}]_{i,j}$ and zero off-diagonal entries [25]. For adjacency matrix $\widehat{\mathbf{E}}_{\mathbf{Z}}$ we have $\widehat{\mathbf{E}}_{\mathbf{Z}} = \mathbf{J}\mathbf{E}_{\mathbf{Z}}\mathbf{J} = -2\widetilde{\mathbf{Z}}\widetilde{\mathbf{Z}}^T$ [95]. We have Laplacian as $\mathbf{L}_{\mathbf{Z}} = \mathbf{D}_{\mathbf{Z}} - \widehat{\mathbf{E}}_{\mathbf{Z}}$ with $\mathbf{D}_{\mathbf{Z}} = 0$. This gives us for $\mathbf{Z}_{t+1}$ the Laplacian $\mathbf{L}_{\mathbf{Z}_{t+1}} = 2\mathbf{Z}_{t+1}\mathbf{Z}_{t+1}^T$. It also follows from the fact that since we choose our intialization $\mathbf{Z}_0$ as column-centered matrix, and $\mathbf{Z}_{t+1} = \mathbf{H}\mathbf{Z}_t$ are also successively column-centered for all $t > 0$. Hence, $\mathbf{L}_{\mathbf{Z}_{t+1}} = 2\widehat{\mathbf{Z}}_{t+1}\widehat{\mathbf{Z}}_{t+1}^T$. Now substituting $\mathbf{Z}_{t+1} = \mathbf{H}\mathbf{Z}_t$ in Laplacian equation $\mathbf{L}_{\mathbf{Z}_{t+1}}$ we get,

$$\mathbf{L}_{\mathbf{Z}_{t+1}} = 2(\mathbf{H}\mathbf{Z}_t)(\mathbf{H}\mathbf{Z}_t)^T = 2\mathbf{H}\mathbf{Z}_t\mathbf{Z}_t^T\mathbf{H}^T = \mathbf{H}\mathbf{L}_{\mathbf{Z}_t}\mathbf{H}^T. \tag{B.1}$$

Substituting above equation into right hand side of the statement to be proved gives us,

$$\mathbf{Tr}\left(\mathbf{Z}_{t+1}^T \mathbf{L}_{\mathbf{Z}_{t+1}} \mathbf{Z}_{t+1}\right) = \mathbf{Tr}\left(\mathbf{Z}_{t+1}^T \mathbf{H}\mathbf{L}_{\mathbf{Z}_t}\mathbf{H}^T \mathbf{Z}_{t+1}\right).$$

Substituting eigen decomposition of $\mathbf{H} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$ where $\mathbf{\Lambda}$ is a diagonal eigenvalues matrix

with values less than one (Theorem 32) we get,

$$\mathbf{Tr}\left(\mathbf{Z}_{t+1}\mathbf{L}_{\mathbf{Z}_{t+1}}\mathbf{Z}_{t+1}\right) = \mathbf{Tr}\left(\mathbf{Z}_{t+1}^{T}(\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{T})\mathbf{L}_{\mathbf{Z}_t}(\mathbf{Q}^{T}\mathbf{\Lambda}\mathbf{Q})\mathbf{Z}_{t+1}\right).$$

For $\mathbf{\Lambda} = \mathbf{I}$ (identity matrix) gives us,

$$\mathbf{Tr}\left(\mathbf{Z}_{t+1}\mathbf{L}_{\mathbf{Z}_{t+1}}\mathbf{Z}_{t+1}\right) \leq \mathbf{Tr}\left(\mathbf{Z}_{t+1}^{T}(\mathbf{Q}\mathbf{I}\mathbf{Q}^{T})\mathbf{L}_{\mathbf{Z}_t}(\mathbf{Q}^{T}\mathbf{I}\mathbf{Q})\mathbf{Z}_{t+1}\right) \leq \mathbf{Tr}\left(\mathbf{Z}_{t+1}^{T}\mathbf{L}_{\mathbf{Z}_t}\mathbf{Z}_{t+1}\right).$$

Repeating the above process until $t = 0$ we get $\mathbf{Tr}\left(\mathbf{Z}_{t+1}\mathbf{L}_{\mathbf{Z}_{t+1}}\mathbf{Z}_{t+1}\right) \leq \mathbf{Tr}\left(\mathbf{Z}_{t+1}^{T}\mathbf{L}_{\mathbf{Z}_0}\mathbf{Z}_{t+1}\right)$. Now, for the initialisation $\mathbf{Z}_t = \mathbf{Z}_k$ at $t = 0$, and given that $\mathbf{Z}_{k+1} = \arg\max_{\mathbf{Z}} g(\mathbf{Z}, \mathbf{Z}_k)$ we have,

$$\mathbf{Tr}\left(\mathbf{Z}_{k+1}\mathbf{L}_{\mathbf{Z}_{k+1}}\mathbf{Z}_{k+1}\right) \leq \mathbf{Tr}\left(\mathbf{Z}_{k+1}^{T}\mathbf{L}_{\mathbf{Z}_k}\mathbf{Z}_{k+1}\right).$$

□

Lemma 33 above allows us to show the following corollary:

Corollary 34. *For fixed point iteration* $\mathbf{Z}_{t+1} = \mathbf{H}\mathbf{Z}_t$ *optimization of* $\mathbf{Z}_{k+1} = \arg\max_{\mathbf{Z}} g(\mathbf{Z}, \mathbf{Z}_k)$, *we have* $\mathbf{Tr}\left(\mathbf{Z}_{k+1}\mathbf{L}_{\mathbf{Z}_{k+1}}\mathbf{Z}_{k+1}\right) \leq \mathbf{Tr}\left(\mathbf{Z}_{k}^{T}\mathbf{L}_{\mathbf{Z}_k}\mathbf{Z}_{k}\right)$.

Proof. From Lemma 33 we have

$$\mathbf{Tr}\left(\mathbf{Z}_{k+1}\mathbf{L}_{\mathbf{Z}_{k+1}}\mathbf{Z}_{k+1}\right) \leq \mathbf{Tr}\left(\mathbf{Z}_{k+1}^{T}\mathbf{L}_{\mathbf{Z}_k}\mathbf{Z}_{k+1}\right) \leq \mathbf{Tr}\left(\mathbf{Z}_{k}^{T}\mathbf{H}^{T}\mathbf{L}_{\mathbf{Z}_k}\mathbf{H}\mathbf{Z}_{k}\right)$$

Following approach similar to proof of Lemma 33 above by substituting eigen decomposition of $\mathbf{H} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^{T}$ into equation above we get,

$$\mathbf{Tr}\left(\mathbf{Z}_{k+1}\mathbf{L}_{\mathbf{Z}_{k+1}}\mathbf{Z}_{k+1}\right) \leq \mathbf{Tr}\left(\mathbf{Z}_{k}^{T}((\mathbf{Q}^{T}\mathbf{I}\mathbf{Q})^{T})\mathbf{L}_{\mathbf{Z}_k}(\mathbf{Q}^{T}\mathbf{I}\mathbf{Q})\mathbf{Z}_{k}\right) \leq \mathbf{Tr}\left(\mathbf{Z}_{k}^{T}\mathbf{L}_{\mathbf{Z}_k}\mathbf{Z}_{k}\right)$$

□

Acknowledgement of Previous Publications

This dissertation work is primarily based on author's previous works presented in the following papers of the author:

1. C. Tonde, A. Elgammal, Simultaneous Twin Kernel Learning using Polynomial Transformations for Structured Prediction, *IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

2. C. Tonde, A. Elgammal, Learning Polynomial Kernels Transformations for Structured Prediction, *arxiv:1601.01411 (cs.LG)*, 2016.

3. P. Vepakomma[1], C. Tonde[1], Supervised Dimensionality reduction via Distance Correlation Maximization, *arXiv:1601.00236 (cs.LG)*, 2016.

[1] Authors contributed equally.