Learning in Hilbert Spaces

by

Nimit Kumar

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Bachelor of Technology
(Department of Electrical Engineering)
at Indian Institute of Technology, Kanpur
2004

Supervisor:

Dr. Laxmidhar Behera

ABSTRACT

Learning in Hilbert Spaces

by

Nimit Kumar

Committee Head: Dr. S.C. Srivastava

David Hilbert(1862-1943) was one of the most outstanding mathematicians of his generation. His twenty-three problems (and the recently discovered twenty-fourth problem reported in American Mathematical Monthly) have perplexed even the greatest geniuses and a solution to even one of them has given the concerned much deserving publicity. Amongst the numerous mathematical creations of David Hilbert is the idea of Hilbert Space. Hilbert's work in integral equations in about 1909 led directly to 20th-century research in functional analysis (the branch of mathematics in which functions are studied collectively). This work also established the basis for his work on infinite-dimensional space, later called Hilbert space, a concept that is useful in mathematical analysis and quantum mechanics. Hilbert's was motivated by the analogy of n-dimensional Euclidean Spaces, which he extended to be infinite dimensional spaces. Hilbert and Schmidt together gave the foundation of complex L_2 spaces and the notion of inner product and norm, orthogonality, closed sets, and linear subspaces. It was von Neumann who later developed the theory of linear operators in Hilbert Spaces. This work is dedicated to all such mathematicians behind the development of Hilbert spaces.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Dr. L. Behera, my project supervisor, for his inspiration and continuous encouragement during my endeavors. I am grateful for his support that he extended when I wanted to try some of the most challenging and novel things during my research. I must express my gratitude for Dr. P.K. Kalra, who has been guiding me through his vast experience especially when I was lost in the enormous "Hilbert Spaces". I thank all my friends, Mr. R.N. Yadav, Vrijendra, Yogesh, Gaurav, Shantanu and Santosh without whom my work would never have been as progressive as it was. I would like to dedicate this work of mine to the grace of the almighty god, his holiness Paramahamsa Swami Satyananda and my parents because of whom I am capable of doing this work.

TABLE OF CONTENTS

CHAPTER 1

Introduction

Hilbert Spaces are Inner Product Spaces, with rich geometric structures because of the orthogonality of vectors in the space. Hilbert space is a structure which combines the familiar ideas from vector spaces with the right ideas of analysis to form a useful context for handling mathematical problems of a wide variety. In this work, an elementary yet complete discussion of Hilbert space is presented with applications in Learning Theory, Control Systems, System Identification and Robotics.

A Hilbert space is a vector space with an inner product defined in it, such that every Cauchy Sequence converges to an element in it. This is called the completeness property of the vector space. An inner product space, which is not complete is referred to as a pre-Hilbert Space. We discuss the idea of Vector spaces, Normed spaces, Inner Product spaces and Hilbert spaces in Chapter 2, where a detailed interpretation of their properties is presented alongwith examples of popular Hilbert spaces. The relevance of the completeness property is presented, which confirms some of the popular properties of Hilbert spaces. Hilbert spaces may be infinite dimensional and are determined by the Hamel basis. The dimension of a vector space is the cardinality of the smallest Hamel basis for that space.

One of important ideas used in Learning Theory and Function Approximation

is the Reproducing Kernel Hilbert Spaces. A Reproducing Kernel Hilbert Space (RKHS) is a Hilbert space of functions on some parameter. A RKHS is a restricted, smooth space which is determined by a reproducing kernel which by the Riesz Representation Theorem is a positive definite function. The name RKHS originates from the fact that the kernel reproduces a given functional. The idea of reproducing kernel and RKHS is discussed in Chapter 3. In learning theory, another well known result is due to J. Mercer (1909), which determines the criteria for the selection of positive definite kernels for construction of reproducing kernel hilbert spaces.

Linear operators on a Hilbert space are functionals satisfying the rule of linear superposition. A linear operator is a mapping from one space into another space (here we are interested only in linear operators over Hilbert spaces). Linear Operator theory has been successfully applied to investigate the optimal control problem and stability of control systems. In Chapter 4, a brief discussion on Linear Operator is given with an application to optimal control. We derive the optimal control law for a Linear Quadratic Regulator using Linear Operator theory in Hilbert space. We also investigate the adjoint of a linear operator and some important properties associated with linear operators.

System Identification is the process of building a mathematical model of a dynamical system based on observed data obtained from the system. System Identification can be considered as a function approximation problem given a set of observations of the system. Observations from a system are always finite. Thus, model-based methods do not always give good results because of their inability to capture hidden relationships which might not be truly reflected through the parameters of the model. As discussed in Chapter 3 RKHS provide a framework for function approximation from a finite set of data. Each observation is associated with a kernel function, which

is considered as the evaluation functional at the given observation. In Chapter 5, we discuss how the RKHS provides a framework for function approximation over finite observations.

Clustering is an integral part of pattern recognition and is probably one of the most frequently used data-mining methods. Quantum Clustering is a method motivated by Parzen's scale-space clustering and associates every data point to a vector in Hilbert space. The mapping is essentially done through the Gaussian kernel function satisfying Mercer's Theorem such that the potential function obtained through the Schroedinger equation is minimized at the clusters centers. In Chapter 7, we discuss quantum clustering, a recently introduced method of clustering using higher-dimensional Hilbert spaces. Visual Motor Coordination is a popular problem in robotics, which aims at controlling the movement of the robot manipulator through the visual feedback obtained from two cameras locating the end-effector position. A recently proposed algorithm for visual motor coordination using a quantum clustering based neural control scheme is explained in Chapter 8. A comparison of this algorithm with the popularly used Kohonen's Self Organizing Map approach is also explained and shows better performance of the proposed method.

In Chapter 9 we discuss the foundations of Nonlinear and Chaotic Systems and demonstrate the control of two famous Chaotic Systems using a Chaotic Control Algorithm. In Chapter 10 a Backstepping control algorithm for a nonlinear system in strict-feedback form is proposed and demonstrated using a nonlinear dynamical system. We finally conclude the work by discussions and conclusions.

CHAPTER 2

Hilbert Spaces

Inner product spaces or pre-Hilbert spaces are vector spaces with an inner product defined in it [4], [3]. In this chapter, we discuss the foundations of the theory of Hilbert spaces and why the Hilbert space has important consequences. A Hilbert space is an infinite-dimensional Normed space which is also endowed with an inner product.

We begin by defining some elementary properties of a *Vector Spaces*.

Definition 2.1. A *vector space* $V = (\boldsymbol{X}, S)$ over the set of scalars S is a set of vectors X together with two operations, addition and scalar multiplication, such that the following properties hold:

1. $\boldsymbol{x}, \boldsymbol{y} \in X$ and $\alpha, \beta \in S \Rightarrow \alpha\boldsymbol{x} + \beta\boldsymbol{y} \in X$.

2. $\boldsymbol{x} + \boldsymbol{y} = \boldsymbol{y} + \boldsymbol{x}$.

3. $(\boldsymbol{x} + \boldsymbol{y}) + \boldsymbol{z} = \boldsymbol{x} + (\boldsymbol{y} + \boldsymbol{z})$.

4. There exits a zero vector $\mathbf{0} \in X$ such that $\boldsymbol{x} + \mathbf{0} = \boldsymbol{x} \forall \boldsymbol{x} \in \boldsymbol{X}$.

5. $\alpha(\boldsymbol{x} + \boldsymbol{y}) = \alpha\boldsymbol{x} + \alpha\boldsymbol{y}$

6. $(\alpha + \beta)\boldsymbol{x} = \alpha\boldsymbol{x} + \beta\boldsymbol{x}$

7. $(\alpha\beta)\boldsymbol{x} = \alpha(\beta\boldsymbol{x})$

8. There exists scalars $\mathbf{0}$ and $\mathbf{1}$ such that $\mathbf{0}\boldsymbol{x} = \mathbf{0}$ and $\mathbf{1}\boldsymbol{x} = \boldsymbol{x}$.

Some of the most popular vector spaces are the n-dimensional real and complex spaces, the set of measurable functions, including piecewise continuous functions and the set of complex infinite n-tuple sequences over the complex field. The *Function space* is similarly defined as a vector space of functions which has addition and scalar multiplication defined in the following natural way:

- $(f + g)(\boldsymbol{x}) = f(\boldsymbol{x}) + g(\boldsymbol{x})$,
- $(\lambda f)(\boldsymbol{x}) = \lambda f(\boldsymbol{x})$.

2.1 Normed Metric Spaces

Definition 2.2. A *metric* $d(\boldsymbol{x}, \boldsymbol{y})$ over a vector space V is a scalar which satisfies the following conditions:

1. $d(\boldsymbol{x}, \boldsymbol{y}) \geq 0$, and $d(\boldsymbol{x}, \boldsymbol{y}) = 0$ *iff* $\boldsymbol{x} = \boldsymbol{y}$. *(The Identity Axiom)*

2. $d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x})$. *(The Symmetry Axiom)*

3. $d(\boldsymbol{x}, \boldsymbol{y}) \leq d(\boldsymbol{x}, \boldsymbol{z}) + d(\boldsymbol{z}, \boldsymbol{y})$. *(The Triangle Axiom)*

A vector space with a metric defined on it is known as a *Metric Space*. The dimension of a vector space is determined by the cardinality of the smallest *Hamel Basis* defined below.

Definition 2.3. A set $\boldsymbol{B}$ in a vector space V is defined as the *Hamel Basis* for V if:

1. $\boldsymbol{B}$ is linearly independent,

2. $span(\boldsymbol{B}) = V$.

A Normed space is a vector space with a norm defined in it. The concept of a norm in a vector space is an abstract generalization of the length of a vector as defined below.

Definition 2.4. A real function $\| \bullet \|$ on a vector space is defined as a *norm* if:

1. $\|\boldsymbol{x}\| = 0$ *iff* $\boldsymbol{x} = 0$,

2. $\|\lambda \boldsymbol{x}\| = \lambda \|\boldsymbol{x}\| \forall \boldsymbol{x} \in \boldsymbol{X}$ and $\lambda \in S$,

3. $\|\boldsymbol{x} + \boldsymbol{y}\| \leq \|\boldsymbol{x}\| + \|\boldsymbol{y}\|$.

Convergence in a normed space is defined below and will be found useful in later discussions.

Definition 2.5. Let $V(\boldsymbol{X}, S, \| \bullet \|)$ be a normed vector space with a norm as defined in 2.4. A sequence $\{\boldsymbol{x}_n\}$ of elements of V *converges*to some $\boldsymbol{x} \in \boldsymbol{X}$, if for every $\epsilon > 0$ there exists a number M such that for every $n \geq M$ we have $\|\boldsymbol{x}_n - \boldsymbol{x}\| < \epsilon$. A convergent sequence has a unique limit and a convergent sequence in a normed space is known as convergent in normed sense.

2.2 Complete Inner Product Spaces (Hilbert Spaces)

A Hilbert Space is a scalar product space (popularly known as *inner product space*) for which the correspondence normed space is *complete*. We begin by defining the inner product space and the property of completeness.

Definition 2.6. A real linear space $\boldsymbol{X}$ is called an *inner product space* if for each pair of elements $\boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{X}$ there is defined a real scalar $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ having the following properties (for every $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} \in \boldsymbol{X}$ and $\alpha \in S$):

1. $\langle \boldsymbol{x}, \boldsymbol{y} \rangle \geq 0$ *(Positive Definitenss)*,

2. $\langle \boldsymbol{x}, \boldsymbol{x} \rangle = 0$ *iff* $\boldsymbol{x} = 0$,

3. $\langle \boldsymbol{x}, \boldsymbol{y} \rangle = \langle \boldsymbol{y}, \boldsymbol{x} \rangle$ *(Symmetry)*,

4. $\langle \alpha \boldsymbol{x}, \boldsymbol{y} \rangle = \alpha \langle \boldsymbol{x}, \boldsymbol{z} \rangle$

5. $\langle \boldsymbol{x} + \boldsymbol{y} +, \boldsymbol{z} \rangle = \langle \boldsymbol{y}, \boldsymbol{z} \rangle + \langle \boldsymbol{x}, \boldsymbol{z} \rangle$.

The function $\langle \bullet, \bullet \rangle$ on $\boldsymbol{X} \times \boldsymbol{X}$is known as the *inner product* on $\boldsymbol{X}$. Inner-product spaces are also known as *pre-Hilbert spaces*. Morover, every inner-product on a vector space V induces a norm on V defined as: $\|\boldsymbol{x}\| = \langle \boldsymbol{x}, \boldsymbol{x} \rangle$.

2.2.1 Cauchy Sequence and the idea of Completeness

The convergence of sequences in normed space has a natural fallout as the idea of Cauchy Sequence and completeness as defined below.

Definition 2.7. A sequence of vectors $\{\boldsymbol{x}_n\}$ in a normed space is called a *Cauchy Sequence* if for every $\epsilon > 0$ there exists a number M such that $\|\boldsymbol{x}_m - \boldsymbol{x}_n\| < \epsilon$ for all $m, n > M$.

In other words

$$lim_{m,n \to \infty} \|\boldsymbol{x}_m - \boldsymbol{x}_n\| = 0.$$

An inner product space is *complete* if every Cauchy sequence in $\boldsymbol{X}$ converges to a point in $\boldsymbol{X}$.

2.2.2 Hilbert Spaces – Definition and Examples

Definition 2.8. A complete inner product space is called a *Hilbert space*.

By the completeness of an inner product space V, we mean the completeness of V as a normed space as defined in 2.4. For the discussions that follow, we denote a Hilbert space by H. We review the concept of orthogonality in a Hilbert space.

Definition 2.9. Let V and W be subspaces of a Hilbert space H; then V is orthogonal to W, written as $V \perp W$, if for all $\boldsymbol{v} \in V$ and $\boldsymbol{w} \in W$, $\langle \boldsymbol{v}, \boldsymbol{w} \rangle = 0$.

Definition 2.10. If V is a subspace of a Hilbert space H; then the orthogonal component of V in H is the set $V^{\perp} = \{\boldsymbol{x} \in H : \forall \boldsymbol{v} \in V, \langle \boldsymbol{x}, \boldsymbol{v} \rangle = 0\}$.

Definition 2.11. If V and W are orthogonal subspaces of a Hilbert space H; then the orthogonal sum of V in W is $V \oplus W = \{\boldsymbol{x} \in H : \boldsymbol{x} = \boldsymbol{v} + \boldsymbol{w}, \boldsymbol{v} \in V, \boldsymbol{w} \in W\}$.

Example 2.12. Since the space of complex numbers $\boldsymbol{C}$ is complete, it is a Hilbert space. Similarly the n-dimensional space $\boldsymbol{C}^n$ is a Hilbert space.

Example 2.13. The space of all infinite sequences $\{\boldsymbol{z}_n\}$ of complex numbers such that

$$\sum_{n=1}^{\infty} \|\boldsymbol{z}_n\|^2 < \infty$$

is defined as a L^2 space. The space $L^2([a, b])$ is the space of all Lebesgue integrable functions on the interval $[a, b]$. It follows that $L^2([a, b])$ is a Hilbert space.

A popular notion is on equivalence of two Hilbert spaces. Two Hilbert spaces X and Y are called *equivalent* if there is a bijective mapping $U : X \to Y$ that preserves:

- linearity,
- the inner product, and
- the norm.

The following theorem is useful in drawing equivalence between Hilbert spaces.

Theorem 2.14. *Every Hilbert space $\boldsymbol{X} \neq 0$ is equivalent to the Hilbert space $L_2(I)$ for some index set I.*

CHAPTER 3

Reproducing Kernel Hilbert Spaces (RKHS)

Reproducing Kernel Hilbert Spaces (RKHS) can be used in a wide variety of curve fitting, function estimation, model description and model building applications [13], [16]. An RKHS is a Hilbert space in which all point evaluations are bounded linear functionals.

3.1 Reproducing Kernels

Let H be a Hilbert space of functions on some domain $\boldsymbol{X}$, such that for every $\boldsymbol{x} \in \boldsymbol{X}$ there exists an element $\Phi(\boldsymbol{x}) \in H$, such that

$$f(\boldsymbol{x}) = \langle \Phi(\boldsymbol{x}), f \rangle, \forall f \in H$$

where $\langle \bullet, \bullet \rangle$ is the inner product defined in H. Let $\boldsymbol{x}, \boldsymbol{y} \in \boldsymbol{X}$, we define the *kernel* function as

$$K(\boldsymbol{x}, \boldsymbol{y}) = \langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}) \rangle. \tag{3.1}$$

The function Φ is known as the feature map associated with the kernel K.

3.2 The idea of RKHS

A Hilbert space with continuous functionals is known as a *Reproducing Kernel Hilbert Spaces*(RKHS). The bounded linearity of the evaluation functionals is the

defining property for a RKHS.

Definition 3.1. A Hilbert space H is called a *Reproducing Kernel Hilbert Space* if the following conditions are satisfied:

1. the elements of H are complex or real-valued functions defined on any set $\boldsymbol{X}$;
2. for every $\boldsymbol{x} \in \boldsymbol{X}$ there exists $K_{\boldsymbol{x}} > 0$, such that $|f(\boldsymbol{x})| \leq K_{\boldsymbol{x}}\|f\|$, $f \in H$.

The RKHS is uniquely defined by a Reproducing Kernel defined above, withe the uniqueness determined by the Riesz Representation Theorem. Some of the useful properties for a Reproducing Kernel is its positive definiteness as defined below.

Definition 3.2. The complex or real-valued function $K = K(\boldsymbol{x}, \boldsymbol{y})$ on $\boldsymbol{X} \times \boldsymbol{X}$ is symmetric if $K(\boldsymbol{x}, \boldsymbol{y}) = \overline{K(\boldsymbol{y}, \boldsymbol{x})}$, and positive-definite if for any finite set $\{\boldsymbol{x}_i \in \boldsymbol{X}; i = 1, 2, \ldots\}$ and complex numbers $\lambda_i; i = 1, 2, \ldots,$

$$\sum_{i,j=1}^{n} \lambda_i \overline{\lambda_j} K(\boldsymbol{x}_i, \boldsymbol{x}_j) \geq 0.$$

3.3 Riesz Representation Theorem

The Riesz Representation Theorem discussed in this section provides a defining criteria for existence and uniqueness of a RKHS for a particular reproducing kernel. Before discussing the Riesz Representation Theorem, the following theorem provides a useful result on uniqueness of the reproducing kernel.

Theorem 3.3. (Moore-Aronszajn,1950) *To every positive definite function $K = K(\boldsymbol{x}, \boldsymbol{y})$ on $\boldsymbol{X} \times \boldsymbol{X}$, there corresponds a unique RKHS H_K of real valued functions on $\boldsymbol{X}$ and vice-versa.*

The Hilbert space asociated with K can be constructed as containing all finite linear combinations of the form $\sum_j a_j K(\boldsymbol{x}_j, \bullet)$, and their limits under the norm

induced by the inner product $\langle K(\boldsymbol{x}, \bullet), K(\bullet, \boldsymbol{y})\rangle = K(\boldsymbol{x}, \boldsymbol{y})$. Norm convergence implies pointwise convergence in a RKHS, as can be seen from the following:

$$|f_n(\boldsymbol{x}) - f_m(\boldsymbol{x})| = |K(\boldsymbol{x}, \bullet), f_n - f_m| \tag{3.2}$$

$$\leq K(\boldsymbol{x}, \boldsymbol{x})\|f_n - f_m\| \tag{3.3}$$

The poitive definiteness of the kernel function ensures that the Kernel function defined in 3.1 is an inner product. The function $K_{\boldsymbol{x}}(\bullet) = K(\boldsymbol{x}, \bullet)$ is known as the representer of eveluation at $\boldsymbol{x}$ in H_K.

Definition 3.4. If a bounded linear functional $K : \boldsymbol{X} \to \Re$ on an inner-product space $\boldsymbol{X}$ can be written $K_{\boldsymbol{x}}(y) = \langle \boldsymbol{x}, \boldsymbol{y}\rangle$, for a fixed $\boldsymbol{x} \in \boldsymbol{X}$ and all $\boldsymbol{y} \in \boldsymbol{X}$, then $\boldsymbol{x}$ is called a *representer* of K.

Theorem 3.5. (Riesz Representation Theorem) *A bounded linear functional on a Hilbert space has a unique representer in that space.*

In other words, let f be a bounded linear functional on a Hilbert space H. Then there exists exactly one $\boldsymbol{x}_0$ in H such that $f(\boldsymbol{x}) = \langle \boldsymbol{x}, \boldsymbol{x}_0\rangle$ for $\boldsymbol{x} \in H$. The representation theorem ensures the unqueness of the reproducing kernel. From Theorem 3.3, it follows that the reproducing kernel satisfies the following *reproducing property*:

$$\langle K(\boldsymbol{x}, \bullet), K(\bullet, \boldsymbol{y})\rangle = K(\boldsymbol{x}, \boldsymbol{y}).$$

3.4 The Kernel Trick

Given a kernel $K = K(\boldsymbol{x}, \boldsymbol{y})$, we define an RKHS:

$$H_K = \{f(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x})\} \tag{3.4}$$

with the following dot product:

$$\sum_i \sum_j \alpha_i \beta_j K(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

which implies the reproducing property:

$$\langle K(\boldsymbol{x}, \bullet), f) \rangle = f(\boldsymbol{x}) \tag{3.5}$$

$$\langle K(\bullet, \boldsymbol{x}), K(\bullet, \boldsymbol{y}) \rangle = K(\boldsymbol{x}, \boldsymbol{y}) \tag{3.6}$$

We can now comment on the useful kernel trick using the RKHS formalism, which is the building block of Kernel-based learning theory. The reproducing kernel map may be written as $\Phi(\boldsymbol{x}) : \boldsymbol{x} \rightarrow K(\bullet, \boldsymbol{x})$, which assigns to each observation $\boldsymbol{x}$ a kernel function $K(\bullet, \boldsymbol{x})$. From the reproducing property, we have:

$$\langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{y}) \rangle = \langle K(\bullet, \boldsymbol{x}), K(\bullet, \boldsymbol{y}) \rangle = K(\boldsymbol{x}, \boldsymbol{y}). \tag{3.7}$$

The use of this property lies in areas where the optimization (or learning) of parameters in a learning machine requires evaluation of inner product operations in a transformed feature space. It should be noted that using the kernel function does not require the transformation function Φ to be specified explicitly. Any positive definte function $K(\boldsymbol{x}, \boldsymbol{y})$ can be used in for this purpose. This property is used in learning algorithms such as Support Vector Machines, Kernel PCA, Kernel ICA and other Kernel Learning Machines.

3.5 Mercer's Theorem

Though the use of positive definite kernels is well-emphasized previously, it is important to identify a function as a positive-definite function before it can qualify as a reproducing kernel. An improtant result due to J. Mercer guarantees that a positive definite kernel function reproduces a unique RKHS by Theorem 3.3 and 3.5. We shall first give an elementary result for evaluating the positive definiteness of a kernel function over a finite set of observations.

Proposition 3.6. *Let* $\boldsymbol{X}$ *be a finite input space with* $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ *a symmetric function on* $\boldsymbol{X}$*. Then* $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ *is a kernel function* iff *the matrix*

$$\boldsymbol{K} = (K(\boldsymbol{x}_i, \boldsymbol{x}_j))_{i,j=1}^{n}$$

is positive definite (that is, has positive eigenvalues).

Theorem 3.7. (Mercer,1909) *For a contnuous symmetric function* $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ *in* $L_2(X)$ *having an expansion of the form*

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sum_{k=1}^{\infty} \lambda_k \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}_j)$$

with positive coefficients λ_k *(that is,* $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ *describes an inner product in some feature space), it is necessary and sufficient that the condition*

$$\int_X \int_X K(\boldsymbol{x}_i, \boldsymbol{x}_j) g(\boldsymbol{x}_i) g(\boldsymbol{x}_j) d\boldsymbol{x}_i d\boldsymbol{x}_j \geq 0$$

is valid for all $g \in L_2(X)$ *(*X *being a compact subset of* $\Re_n$*).*

Note that, given the above condition, we can expand the kernel function in its eigenfunctions:

$$K(\boldsymbol{x}, \boldsymbol{x}') = \sum_{k=1}^{\infty} \lambda_k \Psi(\boldsymbol{x}), \Psi(\boldsymbol{x}')$$

where

$$\int K(\boldsymbol{x}, \boldsymbol{x}') \Psi(\boldsymbol{x}') d\boldsymbol{x}' = \lambda_j \Psi_j(\boldsymbol{x});$$

that is, $\Psi_j(\boldsymbol{x})$ is an eigenfunction.

Some of the popular kernel functions include the Gaussian Radial Basis function, Polynomial kernels, String kernels (popularly used in bioinformatics and text classification) and the recently used Fuzzy kernel [10]. We conclude this chapter with a *kernel-based* definition of a RKHS.

Definition 3.8. *(Parzen,1961)*A Hilbert space H is said to be a reproducing kernel Hilbert space (RKHS). with reproducing kernel K, if the members of H are functions on some set $\boldsymbol{X}$, and if there is a kernel K on $\boldsymbol{X} \times \boldsymbol{X}$ having the following two properties; for every $\boldsymbol{x} \in \boldsymbol{X}$ and a particular $\boldsymbol{x}'$:

1. $K(\bullet, \boldsymbol{x}') \in H$, and
2. $f(\boldsymbol{x}') = \langle f, K(\bullet, \boldsymbol{x}') \rangle \forall f \in H$.

We can then associate with $K(\bullet, \boldsymbol{x}')$ a unique collection of functions of the form

$$f(\boldsymbol{x}) = \sum_{i=1}^{L} c_i K(\bullet, \boldsymbol{x}) \tag{3.8}$$

where L is the number of observations, and c_i's are real coefficients.

CHAPTER 4

System Identification using RKHS

The problem of System Identification (SI) is a challenging problem that demands function approximation, regression, model fitting and several statistical methods [14]. Traditional SI methods like NARMAX, ARMA, ARMAX and MA models, cannot sometimes capture the intrinsic system behaviour because of their deterministic approach especially when the observations or data are not benign. This motivated the use of adaptive and learning based methods, such as Neural Networks, that do not assume any prior information about the model. In this chapter, we discuss the elements of SI and the problems associated with it. We present some examples of dynamical systems that are used to demonstrate the use of RKHS based learning method for SI.

4.1 The System Identification Problem

In the discussions that follow, a system is always considered to be *dynamic*, that is the current output value depends not only on the current external stimuli but also on their earlier values. Sometimes when it is not feasible for the external stimuli to be observed, the outputs are in the form of a *time series*.

The System Identification procedure is made up of three basic entities:

1. The data or observations from the system

2. A set of candidate models

3. A rule by which candidate models can be assessed using the data

The data obtained from the system so that they are *maximally informative* and are those during the normal operation of the system. A set of candidate models is selected to compete for best suiting the system based on the data. In classical system identification methods, this is done by assuming possible system transfer function forms and then estimating their parameters. This is popularly known as a model-based approach. Another popular method is the *black-box* based approach, as used in Neural Networks, where no parametric assumptions are made. Model sets with adjustable parameters with physical interpretation are accordingly known as *gray-boxes*.

4.2 Approximation in Hilbert Spaces

Given a set of observations $\{\boldsymbol{z}_i\}_{i=1}^N$ from a metric space $\boldsymbol{Z}$ corresponding to inputs $\{\boldsymbol{x}_i\}_{i=1}^N$, the purpose of System Identification is to identify the underlying mapping between the two [5], [6]. Neglecting the effect of errors, the observations arise as follows

$$\boldsymbol{z}_i = L_i f \tag{4.1}$$

where $\{L_i\}_{i=1}^N$ is a set of linear evaluation functionals, defined on the Hilbert space H of functions which identify the mapping between the observations $\{\boldsymbol{z}_i\}_{i=1}^N$ and inputs $\{\boldsymbol{x}_i\}_{i=1}^N$. These linear functionals associate real numbers to the function f. By Riesz Representation Theorem, we cvan the represent the complete set of observations $\boldsymbol{z} = [\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_N]^T$ in vector form as follows:

$$\overline{\boldsymbol{z}} = \boldsymbol{L}f = \sum_{i=1}^{N} (L_i f) e_i \tag{4.2}$$

where $e_i \in \Re^N$ is the i^{th} standard basis vector. In other words, the SI problem involves identification of the mapping function f such that

$$\boldsymbol{z}_i = f(\boldsymbol{x}_i). \tag{4.3}$$

We can then define the problem of system identification as follows:
Given a function space F and a set of observations $\{\boldsymbol{z}_i\}_{i=1}^N$ of values of linear functionals $\{L_i\}_{i=1}^N$ defined on F, the task of the system identification process is to find in F a function f which satisfies Equation 4.1.

4.3 Use of RKHS in System Identification

RKHS is a Hilbert space of functions and is determined by an evaluation functional or the kernel. The linear evaluation functionals on an RKHS are bounded and continuous. By the Riesz Representation Theorem, we can express the observations as

$$L_i f = \langle f, \Psi_i \rangle_{\boldsymbol{F}}, i = 1, 2, \ldots, N \tag{4.4}$$

where $\langle \bullet, \bullet \rangle$ denotes the inner product in F and $\{\Psi_i\}_{i=1}^N$ are a set of functions each belonging to $\boldsymbol{F}$ and unquely determined by by the functionals $\{L_i\}_{i=1}^N$. Thus, given a RKHS, the set of functions $\{\Psi_i\}_{i=1}^N$ and the observations $\{\boldsymbol{z}_i\}_{i=1}^N$, the objective of the approximation problem is to find a function $f \in F$ such that Equation 4.4 is satisfied.

4.4 Normal and Regularized Solutions

Assume that the kernel function evaluations $K(\boldsymbol{x}_i, \bullet)$ are linearly independent and that they form a finite dimensional space, as there are only a finite number of observations. Considering the error free case, the problem of approximation of well-posed for infitely large observations. However, as there are only a finite number of

observations obtained from a system, there is no unique solution. Thus, combining Equations 4.3 and 3.8 leads to the following linear system of equations:

$$\boldsymbol{K}\boldsymbol{c} = \boldsymbol{z}^N \tag{4.5}$$

where $\boldsymbol{K}$ is the kernel Gram matrix as defined in Proposition 3.6. This solution is the *normal* solution, $\hat{f}$, and is guaranteed to exist and be uniqueas there will always be one of minimal distance from the null element of F. However, if the data are affected by errors (which is more often the case), the normal solution no longer exists. Instead, a solution can be obtained by minimizing the norm of the errors in Z, that is to find an $f \in F$ such that

$$Err[f] = \sum_{i=1}^{N} \|f(\boldsymbol{x}_i - \boldsymbol{z}_i)\|_Z = \text{minimum}. \tag{4.6}$$

To remove this possibly ill-conditioned optimization process, a regularized formulation of the problem is as follows:

$$J_{reg}[f] = \sum_{i=1}^{N} \|f(\boldsymbol{x}_i) - \boldsymbol{z}_i\|_Z^2 + \rho\|f\|_F^2 \tag{4.7}$$

where $\rho \in \Re^+$ is known as the regularization parameter. Considering $Z = L_2$, Equation 4.7 can be simplified as solving the following equation for c:

$$(\boldsymbol{K} + \rho I)\boldsymbol{c} = \boldsymbol{z}^N \tag{4.8}$$

and

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} c_i K(\boldsymbol{x}, \boldsymbol{x}_i). \tag{4.9}$$

The above two methods for approximation are also known as *static* methods, as they do not involve any iterative or sequential solutions.

4.5 Iterative Solutions

The above approach gives an appropriate solution when the observations are error free or have very little error. The choice of the regularization parameter is again an issue that neds careful consideration before the method can be succesfully used. To avoid these problems, an iterative learning method is proposed. The method arises due to the following theorem.

Theorem 4.1. *Let* $\{\gamma_n\}_{n=1}^{\infty}$ *satisfy:*

1. $0 < \gamma_n < \frac{2}{\lambda_{max}} \forall n$, *where* λ_{max} *is the largest eigenvalue of* $LL^* = \boldsymbol{K}$*; and*
2. $\sum_{n=1}^{\infty} \gamma_n = \infty$.

Define the iteration as $f^n = L^*\boldsymbol{c}^n = \sum_{i=1}^{N} c_i^n K(\boldsymbol{x}_i, \bullet)$ together with $f^0 \in F$ (i.e. $\boldsymbol{c}^0 \in \Re^n$) arbitrary and

$$\boldsymbol{c}^{n+1} = \boldsymbol{c}^n - \gamma_n \tilde{\boldsymbol{c}}^n, \tilde{\boldsymbol{c}}^n = LL^*\boldsymbol{c}^n - \boldsymbol{z}^n, \text{then} \tag{4.10}$$

$$\|\tilde{f}^n\|_F^2 = \|L^*\tilde{\boldsymbol{c}}^n\|_F^2 \to 0, \text{as} n \to \infty. \tag{4.11}$$

The above theorem ensures that an iterative method in the form explained above always yields a solution within an error tolerance. In the next section we demonstrate the use of the above two methods in system identification of two dynamical systems in both noise-free and noisy conditions

4.6 Results

As an example of the application of the static and iterative RKHS approach we consider the following discrete-time nonlinear dynamical system

$$y(t) = 0.5y(t-1)+0.3y(t-1)u(t-1)+0.2u(t-1)+0.05y^2(t-1)+0.6u^2(t-1). \tag{4.12}$$

The observations are generated as $z(t) = y(t) + \epsilon(t)$ where $\epsilon \sim N(0, 0.1)$. In identifying the system the data were generated from an initial condition of $y(1) = 0.1$ and the control input was sampled as $u(t) \sim N(0.2, 0.1)$. The RKHS approach was then applied to estimate a model of the form $y(t) = f(y(t-1), u(t-1))$. Th reproducing kernel was choosen as the Gaussian function, $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = exp(-\beta\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2)$ where $\beta \in \Re^+$. The training and testing sets were generated with 500 samples each. After a series of cross-validation, the value of β was decided as 0.1 and the regularization factor was chosen as 0.1, while γ_n was initialized with 0.002. The test results for static and iterative method is shown in Figures 4.1 and 4.2 respectively. A Mean Square Error of 0.0101 was reached in both the cases. On increasing the noise level, the error in static method goes high, demonstrating its inability to handle noisy data.

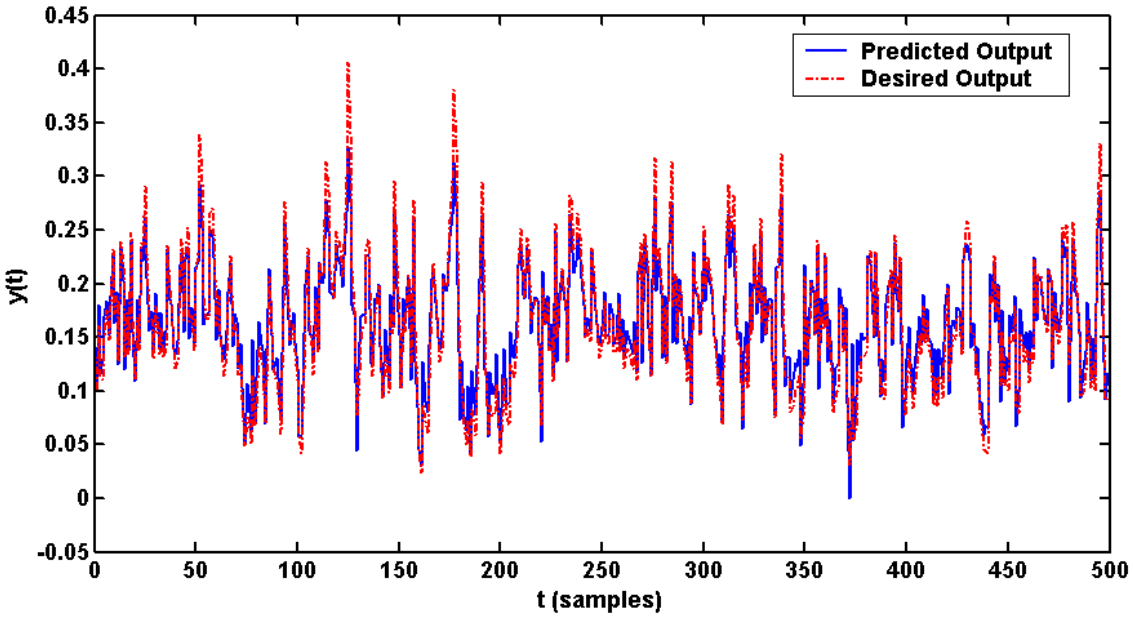

Figure 4.1: Test result of the dynamical system given in Equation 4.12 using the RKHS based static method.

We consider another benchmark problem as a nonlinear time series problem de-

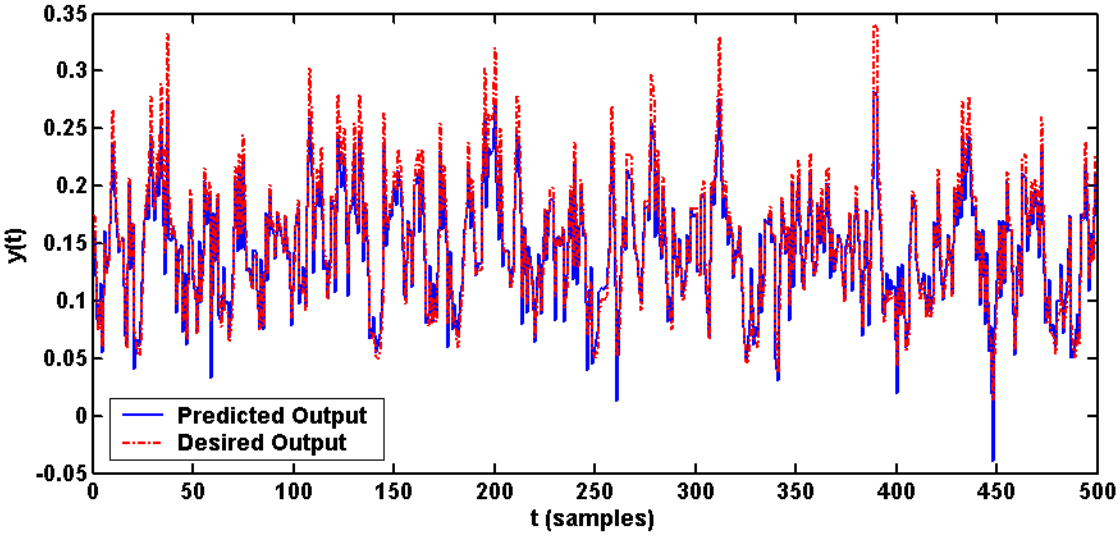

Figure 4.2: Test result of the dynamical system given in Equation 4.12 using the RKHS based iterative method.

scribed by the following difference equation

(4.13)

$$y(t) = (0.8-0.5exp(-y^2(t-1)))y(t-1)-(0.3+0.9exp(-y^2(t-1)))y(t-2)+0.1sin(\pi y(t-1)).$$

This provides a simple second order time series which is highly nonlinear. 200 data points were generated by iterating the equation from the initial point [0.10.1]. Similar to the previous methods, a noise, $N(0, 0.1)$, was added to the output.

Gaussian kernel function was chosen and β was initialized to be 0.37 after a series of cross-validation tests. The prediction was tested on 100 future steps. The results for both the static and iterative methods are shown in Figures 4.3, 4.4, 4.5 and 4.6. The static and iterative method show comparable accuracy in a noise-free case when both of them reach a MSE of 0.0001. However on adding a noise $N(0, 0.3)$, the MSE in the static case is 0.0045, while that using iterative method reaches an MSE of 0.0040. Thus, in situations where the observations are noisy, iterative methods are better to use.

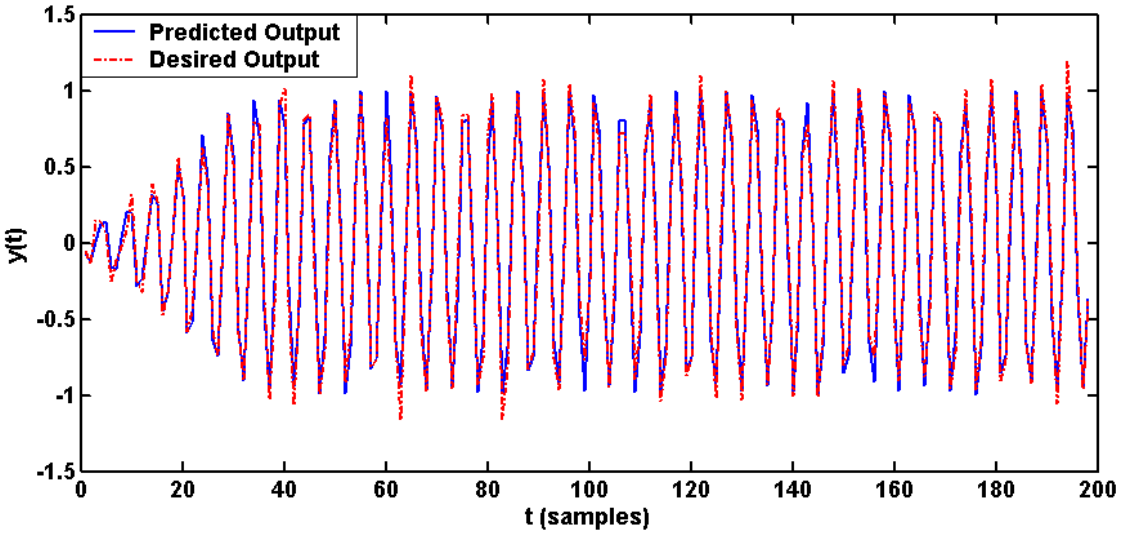

Figure 4.3: Training result of the dynamical system given in Equation 4.13 using the RKHS based static method.

RKHS based method for system identification provides a non-parametric and model-free method which is capable of identifying the system characteristics as a functional dependence on the observed values. This is particularly useful in noisy environments as demonstrated.

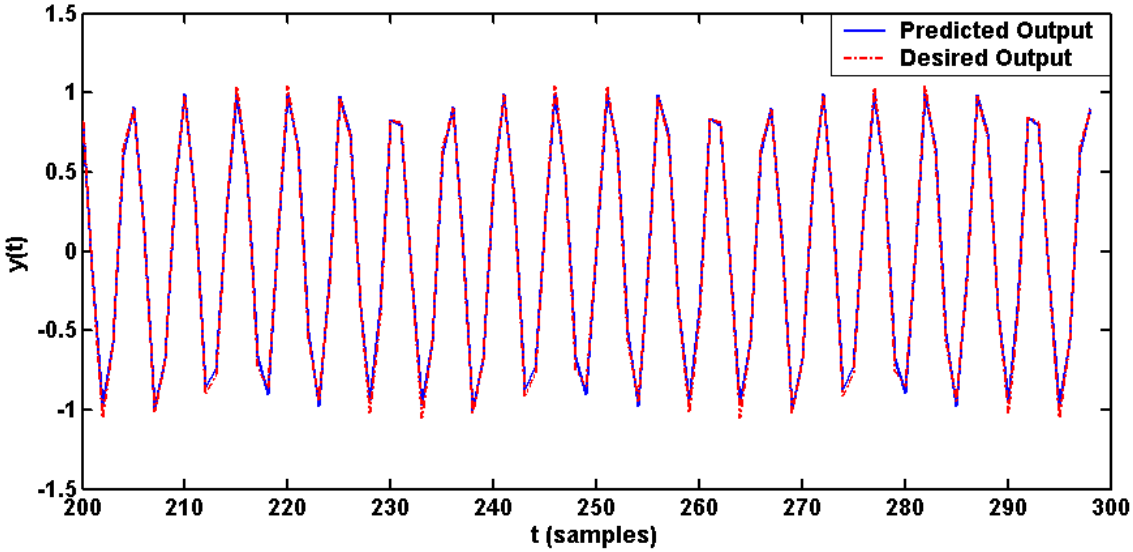

Figure 4.4: Test result of the dynamical system given in Equation 4.13 using the RKHS based static method.

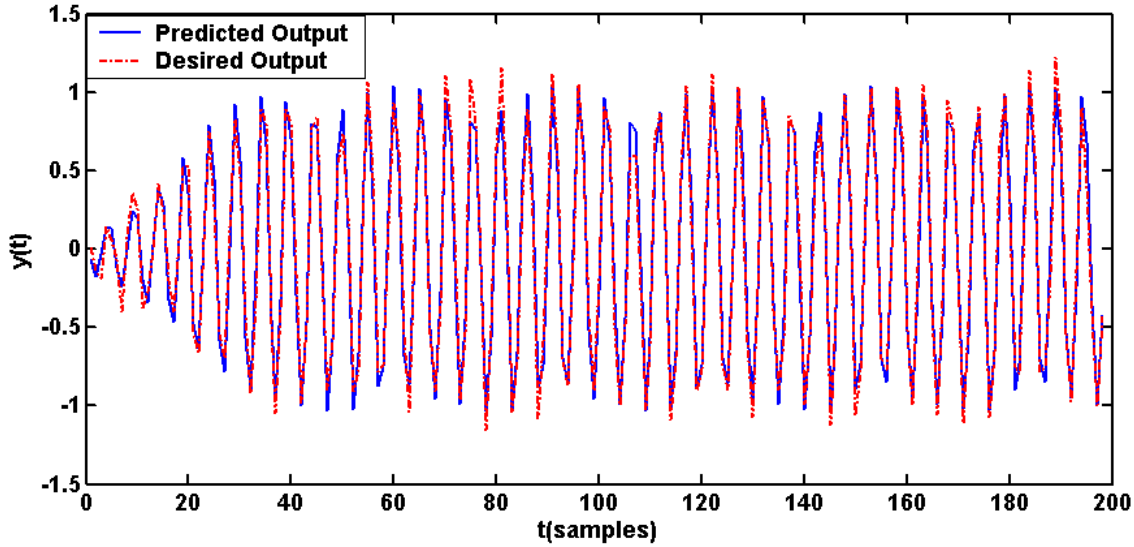

Figure 4.5: Training result of the dynamical system given in Equation 4.13 using the RKHS based iterative method.

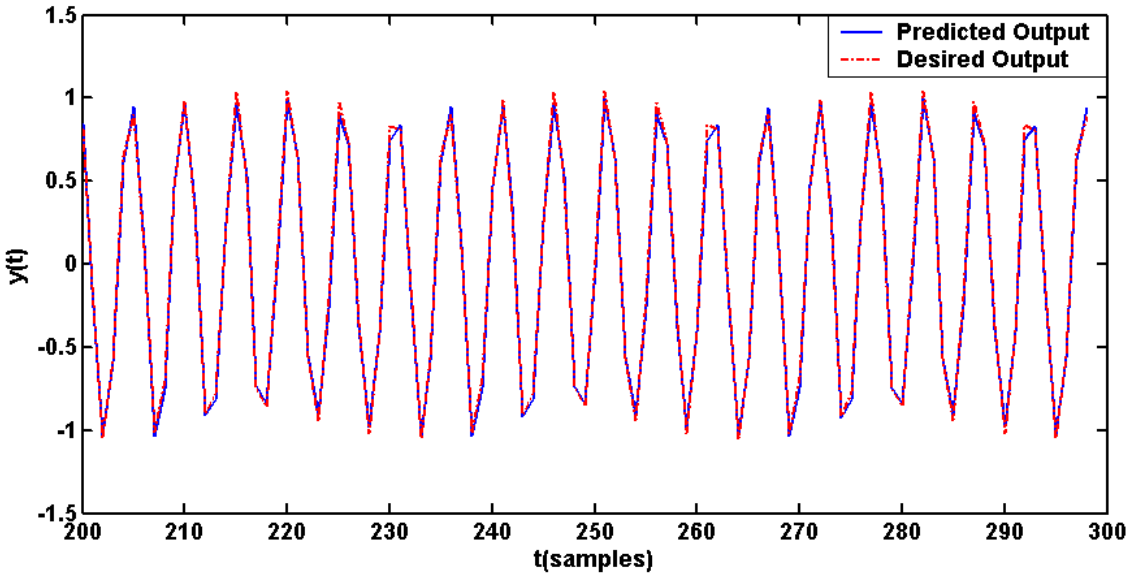

Figure 4.6: Test result of the dynamical system given in Equation 4.13 using the RKHS based iterative method.

CHAPTER 5

Linear Operator Theory

The concept of an operator (or transformation) on a normed vector space is a natural generalization of the idea of a function of a real variable. Linear operators on a normed vector space are widely used to represent physical quantities, and hence their importance is further enhanced in applied mathematics and mathematical physics [1], [23]. Some of the popular operators include differential, integral and matrix operators. In this chapter we introduce some useful properties of linear operators in Hilbert spaces with special attention to those that are useful in Control Theort as will be seen in the next chapter.

5.1 Linear Operators in Hilbert Spaces

Let $\boldsymbol{H}_1$ and $\boldsymbol{H}_2$ be vector spaces (in our case, we will limit our discussion to Hilbert spaces) over $\boldsymbol{X}$. A linear operator $\boldsymbol{A} : \boldsymbol{H}_1 \rightarrow \boldsymbol{H}_2$ is said to be linear if for all $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \boldsymbol{H}_1$ and $\alpha, \beta \in S$, where S is the field associated with $\boldsymbol{H}_1$

$$\boldsymbol{A}(\alpha \boldsymbol{x}_1 + \beta \boldsymbol{x}_2) = \alpha \boldsymbol{A}\boldsymbol{x}_1 + \beta \boldsymbol{A}\boldsymbol{x}_2. \tag{5.1}$$

Thus, a linear operator satisfies both additivity and homogeneity. It is immediate that all matrices are linear operators over the concerned field and so is the covolution operation. A linear operator in a Hilbert space inherits the special geometric

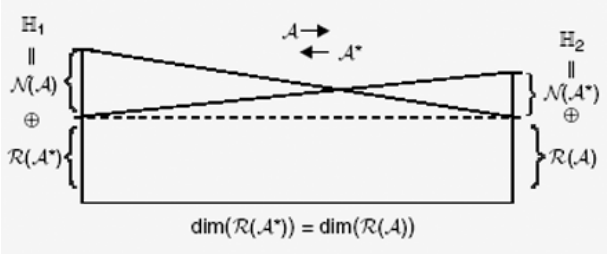

Figure 5.1: Fundamental relationship between an operator and its adjoint as given by Definition 5.1.

properties of the space.

5.2 Properties of Linear Operators

We discuss some relevant and important concepts and results arising out of Linear Operator theory. A fundamental concept in linear algebra and functional analysis is the Hilbert adjoint operator.

Definition 5.1. Given a linear operator $\boldsymbol{A} : \boldsymbol{H}_1 \to \boldsymbol{H}_2$, the *adjoint* of $\boldsymbol{A}$, denoted by $\boldsymbol{A}^*$, is an operator from $\boldsymbol{H}_1 \to \boldsymbol{H}_2$ that satisfies

$$\langle \boldsymbol{A}\boldsymbol{x}, \boldsymbol{y} \rangle_{\boldsymbol{H}_2} = \langle \boldsymbol{x}, \boldsymbol{A}^* \boldsymbol{y} \rangle_{\boldsymbol{H}_1} \tag{5.2}$$

for all $\boldsymbol{x} \in \boldsymbol{H}_1$ and $\boldsymbol{y} \in \boldsymbol{H}_2$.

The relationship between an operator and its adjoint is explained in Figure 5.1. A linear operator $\boldsymbol{A} : \boldsymbol{H}_1 \to \boldsymbol{H}_2$ is referred to as *compact* (or *completely continuous*) if it transforms any bounded set in $\boldsymbol{H}_1$ into a relatively compact set in $\boldsymbol{H}_2$. An operator is called *finite dimensional* if its range is of finite dimension.

Definition 5.2. A symmetric operator $\boldsymbol{A} : \boldsymbol{H}_1 \to \boldsymbol{H}_2$ is referred to as *positive definite* if there exists a constant $c > 0$, such that $\langle \boldsymbol{A}\boldsymbol{x}, \boldsymbol{x} \rangle_{\boldsymbol{H}} = c \langle \boldsymbol{x}, \boldsymbol{x} \rangle_{\boldsymbol{H}}, \forall \boldsymbol{x} \in \boldsymbol{H}$.

We close this chapter by stating two important theorems on compactness of a linear operator and its adjoint.

Theorem 5.3. *Finite dimensional bounded operators are compact operators.*

Theorem 5.4. *Let* A *be a compact operator on a Hilbert space* H *and let* B *be a bounded operator on* H*. Then* AB *and* BA *are compact.*

CHAPTER 6

Optimal Control using Linear Operators in Hilbert Spaces

We discuss the problem of Optimal Control, in particular the Linear Quadratic Regulator, and an approach for its solution using Linear Operator Theory. Unlike the Algebraic Riccati Equation, which is not time varying, we present a time varying situation in the form of the matrix Riccati Equation.

6.1 Linear Quadratic Optimal Control

Given a dynamical system

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t), \boldsymbol{x}(0) = \boldsymbol{x}_0, \tag{6.1}$$

$$\boldsymbol{y}_t = \boldsymbol{C}\boldsymbol{x}(t) \tag{6.2}$$

on a fixed interval $[0, T]$. The idea is to minimize the performance function defined below:

$$J(\boldsymbol{u}) = \int_0^T \boldsymbol{x}^{\mathrm{T}}\boldsymbol{x}dt + \alpha \int_0^T \boldsymbol{u}^{\mathrm{T}}\boldsymbol{u}dt, \text{where } \alpha > 0. \tag{6.3}$$

We shall derive the optimal control law using Linear Operator Theory and compare it with one obtained using the solution of the Algebraic Riccati Equation. The goal of the minimization process is the construction of a stabilizing linear state feedback controller of the form $\boldsymbol{u} = -\boldsymbol{k}\boldsymbol{x}$, that minimizes $J(\boldsymbol{u})$.

Consider the class of vector-valued functions defined as:

$$\boldsymbol{L}_M^2([0,T]) = \{\boldsymbol{u} : \boldsymbol{u}^T = (\boldsymbol{u}_1, \boldsymbol{u}_2, \ldots \boldsymbol{u}_M), \boldsymbol{u}_k \in \boldsymbol{L}^2([0,T]), k = 1, 2, \ldots, M\}. \tag{6.4}$$

An inner product in $\boldsymbol{L}_M^2([0,T])$ is defined as:

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \int_0^T \boldsymbol{u}^T \boldsymbol{v} dt. \tag{6.5}$$

It can be seen that 6.5 satisfies all the conditions for an inner product as given by definition 2.6.

Claim 6.1. *$\boldsymbol{L}_M^2([0,T])$ is a Hilbert space.*

If every element of the vector $\boldsymbol{u}(t)$ is a continuous function of t, then the solution of the dynamical system given above is unique and is of the form:

$$\boldsymbol{x}(t) = \Phi(t,0)\boldsymbol{x}_0 + \int_0^t \Phi(t,\tau)\boldsymbol{B}\boldsymbol{u}(t)d\tau \tag{6.6}$$

where $\Phi(t,\tau)$ is the state transition matrix and continuous on $[0,T]\times[0,T]$. Consider the following linear operator $L : \boldsymbol{L}_M^2([0,T]) \to \boldsymbol{L}_N^2([0,T])$, that is from the space of $\boldsymbol{u}(t)$ to the space of $\boldsymbol{x}(t)$:

$$L[\boldsymbol{u}(t)] = \int_0^t \Phi(t,\tau)\boldsymbol{B}\boldsymbol{u}(t)d\tau. \tag{6.7}$$

Claim 6.2. *L is a compact operator.*

Proof. Since, $L[\boldsymbol{u}(t)]$ is a bounded and finite dimensional operator, it follows from the theorem 5.3 that $L[\boldsymbol{u}(t)]$ is compact. □

Let $v = -\Phi(t,0)\boldsymbol{x}_0$, we can then write $\boldsymbol{x}(t) = \boldsymbol{L}\boldsymbol{u} - v$. Thus, $J(\boldsymbol{u})$ can be written as

$$J(\boldsymbol{u}) = \langle \boldsymbol{L}\boldsymbol{u} - v, \boldsymbol{L}\boldsymbol{u} - v \rangle + \alpha\langle \boldsymbol{u}, \boldsymbol{u} \rangle = \|\boldsymbol{L}\boldsymbol{u} - v\|^2 + \alpha\|\boldsymbol{u}\|^2. \tag{6.8}$$

6.2 LQR Control using Matrix Riccati Equation

The idea of LQR is to determine the $\boldsymbol{u}$ that minimizes $J(\boldsymbol{u})$ as given by 6.8. This can be obtained from the following general result:

Theorem 6.3. *Suppose $\boldsymbol{H}$ and $\boldsymbol{K}$ are real Hilbert spaces and $L : \boldsymbol{H} \to \boldsymbol{K}$ is a compact operator with L^* as its adjoint. Let v be a fixed element in $\boldsymbol{K}$ and let $\alpha \in \Re$. Define a functional $J : \boldsymbol{H} \to \Re$ as in 6.8. If $\alpha > 0$, then there exists a unique $\boldsymbol{u}_0 \in \boldsymbol{H}$ such that $J(\boldsymbol{u}_0) \leq J(\boldsymbol{u}) \forall \boldsymbol{u} \in \boldsymbol{H}$. Moreover, $\boldsymbol{u}_0$) is a solution of the equation*

$$LL^*\boldsymbol{u}_0 + \alpha\boldsymbol{u}_0 = L^*\boldsymbol{v}. \tag{6.9}$$

Proof. Let $\boldsymbol{A} = LL^*$. Since both L and L^* are positive compact operators, then by theorem 5.4, $\boldsymbol{A}$ is a positive compact operator. Therefore $\boldsymbol{A}$ cannot have a negative eigenvalue and hence $-\alpha$ is not an eigenvalue of $\boldsymbol{A}$. Moreover 6.9 has a unique solution whenever α is not an eigenvalue of $\boldsymbol{A}$ and $\alpha \neq 0$. Suppose $\boldsymbol{u}_0$) is the unique solution of 6.9, then for an arbitrary $\boldsymbol{h}$ it follows from 6.8 that

$$\begin{aligned} J(\boldsymbol{u}_0 + \boldsymbol{h}) = & \quad \langle \boldsymbol{L}\boldsymbol{u}_0 - v, \boldsymbol{L}\boldsymbol{u}_0 - v\rangle + \alpha\langle \boldsymbol{u}_0, \boldsymbol{u}_0\rangle \\ & + \langle \boldsymbol{v}, \boldsymbol{v}\rangle + 2\alpha\langle \boldsymbol{u}_0, \boldsymbol{h}\rangle + 2\langle \boldsymbol{L}\boldsymbol{h}, \boldsymbol{L}\boldsymbol{u}_0 - v\rangle + \alpha\langle \boldsymbol{h}, \boldsymbol{h}\rangle \\ & = \|\boldsymbol{L}\boldsymbol{u} - v\|^2 + \alpha\|\boldsymbol{u}\|^2 + \alpha\|\boldsymbol{h}\|^2 + \|\boldsymbol{v}\|^2 \\ & \geq J(\boldsymbol{u}_0). \end{aligned}$$

This concludes the proof. □

The above theorem ensures the minimization of the performance index, but the solution in the present form is not feasible and hence one needs the following theorem.

Theorem 6.4. *Suppose $J(\boldsymbol{u})$ is given as in 6.8 with $\alpha > 0$ and that satisfies the dynamical system given by equations 6.1. If the control vector*

$$\boldsymbol{u}(t) = -\frac{1}{\alpha}\boldsymbol{B}^T\boldsymbol{P}(t)\boldsymbol{x}(t) \tag{6.10}$$

for all $t \in [0,T]$, and $\boldsymbol{P}(t)$ is the solution of the Matrix Riccati Equation

$$\dot{\boldsymbol{P}}(t) + \boldsymbol{A}^T\boldsymbol{P}(t) + \boldsymbol{P}(t)\boldsymbol{A} - \frac{1}{\alpha}\boldsymbol{P}(t)\boldsymbol{B}\boldsymbol{B}^T\boldsymbol{P}(t) + \boldsymbol{I} = \boldsymbol{0}, \tag{6.11}$$

with $\boldsymbol{P}(T) = 0$, then $\boldsymbol{u}(t)$ minimizes $J(\boldsymbol{u})$ as given by equation 6.8.

Proof. We will show that $\boldsymbol{u}(t)$ satisfies 6.9, where $L\boldsymbol{u}$ is given by 6.7. If $\boldsymbol{u}(t)$ satisfies 6.9, then

$$\boldsymbol{u} = -\frac{1}{\alpha}L^*(L\boldsymbol{u} - \boldsymbol{v}) = -\frac{1}{\alpha}L^*\boldsymbol{x}. \tag{6.12}$$

We next find a formula for evaluating $L^*\boldsymbol{w}$ for an arbitrary $\boldsymbol{w} \in \boldsymbol{L}^2_M([0,T])$. To do this, we calculate

$$\begin{aligned}
\langle L\boldsymbol{u}, \boldsymbol{w}\rangle &= \textstyle\int_0^T [\int_0^S \Phi(s,t)\boldsymbol{B}\boldsymbol{u}(t)\,dt]^T\boldsymbol{w}(s)\,ds \\
&= \textstyle\int_0^T \int_0^S \boldsymbol{u}^T(t)\boldsymbol{B}^T\Phi^T(s,t)\boldsymbol{w}(s)\,dt\,ds \\
&= \textstyle\int_0^T \boldsymbol{u}^T(t)[\int_0^S \boldsymbol{B}^T\Phi^T(s,t)\boldsymbol{w}\,ds]\,dt \\
&= \langle \boldsymbol{u}, L^*\boldsymbol{w}\rangle
\end{aligned}$$

Consequently the adjoint operator can be written as

$$[L^*\boldsymbol{w}](t) = \int_0^S \boldsymbol{B}^T\Phi^T(s,t)\boldsymbol{w}\,ds \;\forall t \in [0,T]. \tag{6.13}$$

We next assume that there exists a matrix $\boldsymbol{P}(t)$ such that

$$\boldsymbol{P}(t)\boldsymbol{x}(t) = \int_t^T \Phi^T(s,t)\boldsymbol{x}(s)\,ds \text{ with } \boldsymbol{P}(T) = 0. \tag{6.14}$$

It is necessary to find conditions for existence of such a matrix $\boldsymbol{P}(t)$. Differentiating 6.14 with respect to t and using the equality

$$\dot{\boldsymbol{P}}^T(s,t) = -\boldsymbol{A}^T\Phi(s,t),$$

we obtain

$$\begin{aligned}\dot{\boldsymbol{P}}(t)\boldsymbol{x}(t) + \boldsymbol{P}(t)\dot{\boldsymbol{x}}(t) &= -\boldsymbol{x}(t) - \boldsymbol{A}^T\textstyle\int_t^T \Phi^T(s,t)\boldsymbol{x}(s)\,ds \\ &= -\boldsymbol{x}(t) - \boldsymbol{A}^T\boldsymbol{P}(t)\boldsymbol{x}(t).\end{aligned}$$

In view of 6.1, this reduces to

$$\dot{\boldsymbol{P}}(t)\boldsymbol{x}(t) + \boldsymbol{P}(t)[\boldsymbol{A}\boldsymbol{x}(t) + \boldsymbol{B}\boldsymbol{u}(t)] = -\boldsymbol{x}(t) - \boldsymbol{A}^T\boldsymbol{P}(t)\boldsymbol{x}(t). \tag{6.15}$$

Using 6.12to replace $\boldsymbol{u}(t)$, so that the above equation becomes

$$\dot{\boldsymbol{P}}(t) + \boldsymbol{A}^T\boldsymbol{P}(t) + \boldsymbol{P}(t)\boldsymbol{A} - \frac{1}{\alpha}\boldsymbol{P}(t)\boldsymbol{B}\boldsymbol{B}^T\boldsymbol{P}(t) + \boldsymbol{I} = \boldsymbol{0} \tag{6.16}$$

and is popularly known as the *Matrix Riccati Equation*. Clearly $\boldsymbol{P}(t)$ satisfies 6.11 with $\boldsymbol{P}(T) = 0$. If

$$\boldsymbol{u}(t) = -\frac{1}{\alpha}\boldsymbol{B}^T\boldsymbol{P}(t)\boldsymbol{x}(t),$$

it turns out that $\boldsymbol{u}(t)$ satisfies

$$LL^*\boldsymbol{u}_0 + \alpha\boldsymbol{u}_0 = L^*\boldsymbol{v},$$

where $v = -\Phi(t,0)\boldsymbol{x}_0$, and hence by Theorem 6.3, $\boldsymbol{u}_t$ minimzes the cost functional $J(\boldsymbol{u})$. This completes the proof. □

The matrix riccati equation is often known as the state equation of the nlinear-quadratic control problem and has been shown to have a unique solution for all $t < T$.

CHAPTER 7

Clustering in Hilbert Spaces

In this chapter we discuss a recently introduced non-parametrc clustering method known as Quantum Clustering [8], [9]. Most clustering methods assume an underlying density function for the data, which is not always reasonable to do so. All of the classical parametric densities are unimodal (have a single local maximum), whereas many practical problems involve multimodal densities. We begin by discussing the basics of non-parametric clustering that form the building block of the Quantum Clustering algorithm.

Scale-space clustering gives a non-parametric estimate of the probability density function of $\boldsymbol{x} \in S$, where S is the space of observations, as the weighted combination of a set of basis functions (or kernels), f_i, evaluated at each observation $\boldsymbol{x}_i$. The Parzen-window approach to estimating densities can be introduced by temporarily assuming that the region $\Re_n$ is a d-dimensional hypercube. On considering the Parzen-windows approach, where the weight, w_i corresponding to each observation, is independent of the position, and then modulating the above function with an optimal filter with scale s, $k_s(\boldsymbol{x})$, we get the following estimate for the probability density function.

$$\hat{p}_s(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \, f_i(\boldsymbol{x}) * k_s(\boldsymbol{x}) \; = w \sum_{i=1}^{N} f_i(\boldsymbol{x}) * k_s(\boldsymbol{x}) \; = w \sum_{i=1}^{N} \Psi_i(\boldsymbol{x}) \tag{7.1}$$

For the probability density function to be valid, the function Ψ must be positive definite, hence leading to a kernel-based approach for the problem. A popular choice for this is the Gaussian function, that is also used in Quantum Clustering.

7.1 Quantum Clustering

Quantum clustering is a non-parametric clustering technique [17] based on the scale-space clustering algorithm [18]. It uses the Gaussian kernel to generate the Parzen-window estimator [7]. Given a set of N data points, $\mathbf{x}_1, \ldots, \mathbf{x}_N$, the task is to estimate the probability density function $\Psi(\mathbf{x})$ using the Parzen-window estimator. The estimator is constructed by associating with each of the N data points a Gaussian defined as in Eq. (1). In this sense it belongs to a more general class of non-parametric method - the one using Kernels, here a gaussian kernel.

$$\Psi(\mathbf{x}) = \sum_{i=1}^{N} e^{-(\mathbf{x}-\mathbf{x}_i)/2\sigma^2} \tag{7.2}$$

The maxima of the function $\Psi(\mathbf{x})$ have been shown to occur at the cluster centers [18], where $1/2\sigma^2$ is the scale of the probability estimator. The function $\Psi(\mathbf{x})$ is the Schroedinger Wave Function and performs a nonlinear transformation of the input space into a Hilbert Space. It can be seen that as $\Psi(\mathbf{x})$ is a Gaussian kernel, it is positive definite and hence a standard inner product defined in the transformed space is also positive definite. That is, by Mercer's Theorem [19] , the nonlinear transformation $\Psi(\mathbf{x})$ defines a Hilbert Space.

7.1.1 Quantum Clustering – the Algorithm

The motivation is to search for a Hamiltonian for which Ψ is an eigenstate and a ground state of the operator H.

$$H\Psi = (-\frac{1}{2\sigma^2}\nabla^2 + V(\mathbf{x}))\Psi = E\Psi \tag{7.3}$$

Eq. (2) is a rescaled form of the Schrodinger Eq. in Quantum Mechanics, with Ψ denoting the eigenstate, H the Hamiltonian, V the potential energy and E is the energy eigenvalue. It must be noted that σ is the only parameter in the equation. σ deduces the correct clustering of the space as it controls the width of the Parzen-window [18]. The parameter σ can be controlled in a way such that the technique yields the relevant number of clusters.

Given Ψ, one can solve Eq. (2) for V:

$$V(\mathbf{x}) = E + \frac{(\frac{1}{2\sigma^2})\nabla^2\Psi}{\Psi} \tag{7.4}$$

If V is positive definite, that is $V \geq 0$, E maybe defined as in Eq. (4).

$$E = -min\frac{(\frac{1}{2\sigma^2})\nabla^2\Psi}{\Psi} \tag{7.5}$$

As Ψ is positive definite, it follows that it has no node and hence E is the minimal eigenvalue of V. The lowest possible eigenvalue occurs for the harmonic potential in which case $E = \frac{d}{2}$. This leads to the inequality (5).

$$0 < E \leq \frac{d}{2} \tag{7.6}$$

Quantum Clustering, thus works by determining the cluster centers, with the data points assigned to a cluster based on the Euclidean Distance. In the problem of visual-motor coordination, we are interested in locating the cluster centers and hence the use of QC is preferred.

7.2 Why use Quantum Clustering?

In Quantum Clustering, the cluster centers are obtained while searching for the minima of the potential functions. In statistical clustering methods, the distribution function is maximized. In QC the wave function is interpreted as the probability

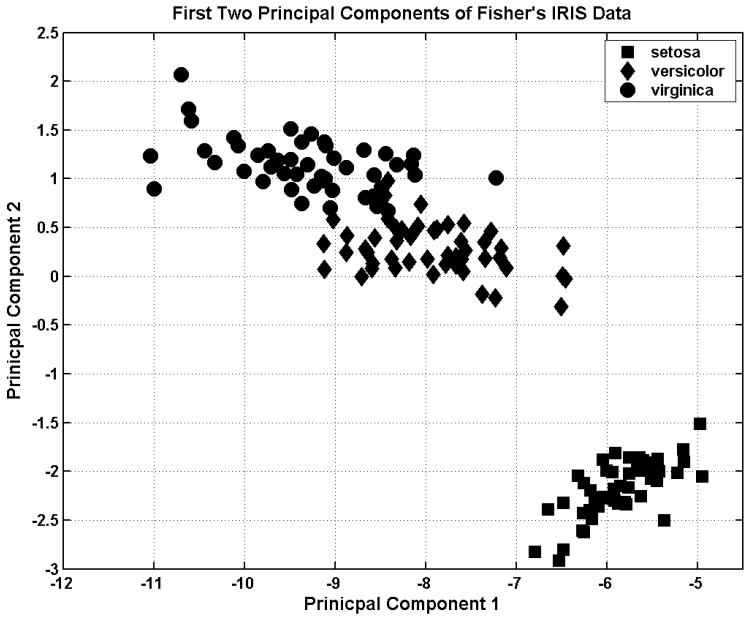

Figure 7.1: Plot of the first two principal components of the IRIS Data.

density function itself. However, determining the potential minima is more robust than locating the maxima of the distribution or wave function. It has been observed that the minima of the potential is more well-behaved than the maxima of the wave function. We illustrate clustering using Quantum Clustering method on the historic IRIS dataset. IRIS is a three class four-dimensional problem, with each class denoting a flower type - Virgnica, Versicolor & Setosa. While Setosa is linearly separable from the other two, the remaining two classes are mutually non-separable. In Figure 7.1 plot the data using the first two principal components of the dataset. [htbd] Figure 7.2 shows the plot of normalized potential function, V/E, for the IRIS Dataset, with the threshold on deciding the minima as a dotted line. The Plot of the distribution function is similarly shown in Figure 7.3. It is evident that the maxima of the distribution function is difficult to find and does not occur in all the three regions. However, the minima of the potential function can be found in every cluster. Thus minima of potential is easily obtained for every cluster than the maxima of the wave

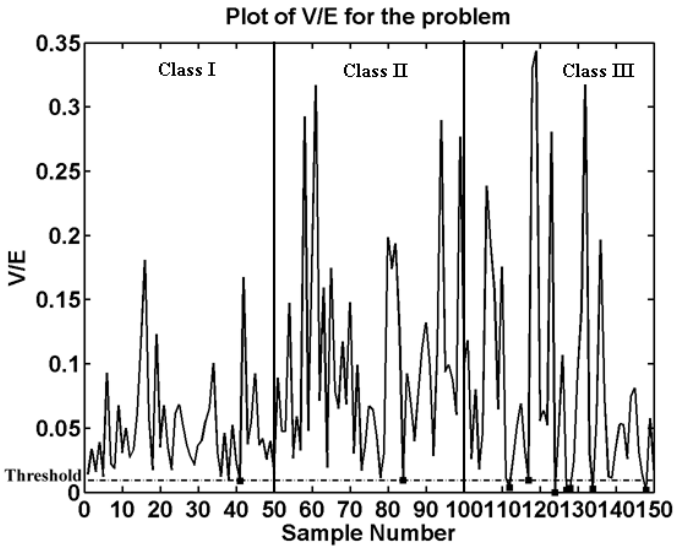

Figure 7.2: Plot of the Potentials at the datapoints of the IRIS Data show that all clusters have at least one point as the minima.

function.

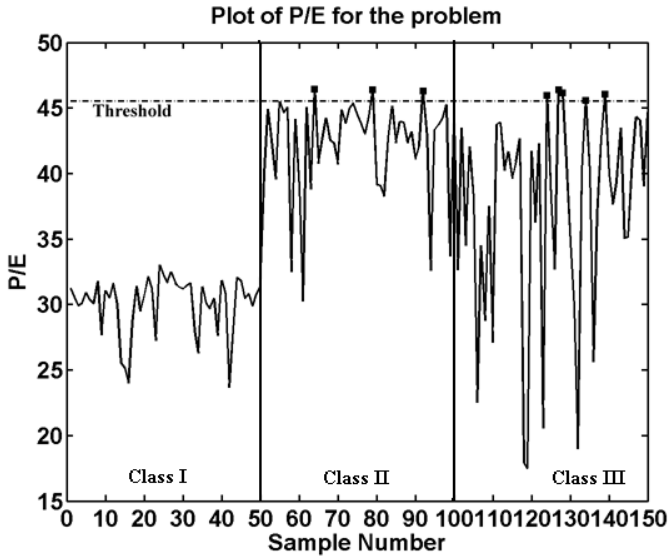

Figure 7.3: Plot of the Wave Functions at the datapoints of the IRIS Data show that the maxima are not well defined for all the clusters.

CHAPTER 8

Visual-Motor Coordination

Visual-Motor Coordination is a problem considered analogous to the hand-eye coordination in biological systems. In this chapter we propose a novel approach to this problem using Quantum Clustering and an extended Kohonen's Self Organizing Feature Map (K-SOFM) [11]. This facilitates the use of the method in varying workspaces by considering the joint angles of the robot arm. Unlike previous work, where a fixed topology for the input space is considered, the proposed approach determines a topology as the workspace varies. Quantum Clustering is a method which constructs a scale-space probability function and uses the Schroedinger Eq. and its lowest eigenstate to obtain a potential whose minimum gives the cluster centers. It transforms the input space into a Hilbert space, where it searches for its minimum. The motivation of this work is to identify the implicit relationship existing between the end-effector positions and the joint angles through Quantum Clustering and Neural Network methods to fine-tune the system to correctly identify the mapping.

8.1 The Problem Statement

A pair of two-dimensional 'retinal' coordinates u_1, u_2 are obtained from the two cameras respectively as shown in Figure 8.1. The objective is to identify the transfor-

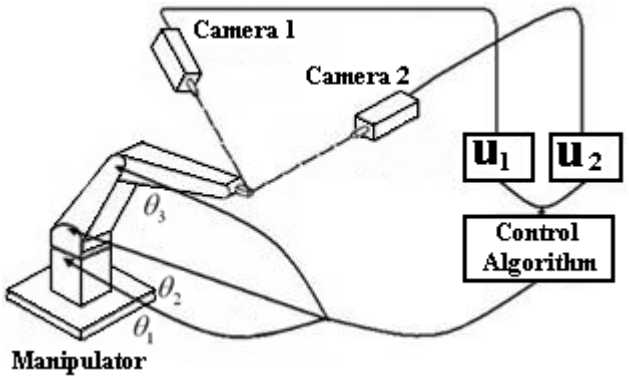

Figure 8.1: Schematic Diagram of the Visuo-Motor Setup.

mation $\Theta(\mathbf{u}_{target})$ from the retinal coordinates to the joint angles θ of the three-joint arm. The vector $\mathbf{u}_{target}$ is a 4-dimensional vector obtained by grouping the retinal coordinates $\mathbf{u}_1$and $\mathbf{u}_2$. In this work, we consider $\mathbf{u}_{target}$ as a 3-dimensional vector by taking into account the actual 3-D location of the end-effector in the workspace rather than through the transformed camera-plane. That is, we do not consider the camera transformation and work on the original end-effector position in the workspace. The first order Taylor expansion of $\Theta(\mathbf{u}_{target})$ is given by Eq. (6).

$$\Theta(\mathbf{u}_{target}) = \theta_s + \mathbf{A}_s(\mathbf{u}_{target} - \mathbf{w}_s) \tag{8.1}$$

The idea is to discretize the workspace $\mathbf{U}$, into non-overlapping regions $\mathbf{F}_r$, such that $\mathbf{w}_r \in \mathbf{F}_r$ is the reference or weight vector, θ_r is the zero-order term and $\mathbf{A}_r$ is the Jacobian Matrix, which determines the first order term. Eq. (6) gives the local approximation of each neuron for the joint angle values of the target point $\mathbf{u}_{target}$.

8.2 Visuo–motor Coordination using Quantum–Clustering based Neural Control Scheme

8.2.1 A Joint Input-Output Space Partitioning Scheme

We simulate the workspace using the inverse kinematics relationship between the end-effector position in the 3-dimensional world space and the joint angles. This

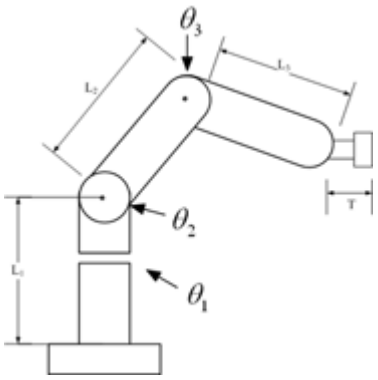

Figure 8.2: Schematic Diagram of the Manipulator which can be used to generate the inverse kinematics relationship.

inverse kinematics relationship is easily obtained using the geometry of the robot manipulator. A typical manipulator is shown in Figure 8.2. Given a set of end-effector positions $\mathbf{u}$ and their corresponding joint-angles θ from the workspace, we consider a six dimensional vector $\mathbf{z}$ by grouping $\mathbf{u}$ and θ. The motivation for the proposed approach can be understood if we observe from Figure 8.3 that the output space formed by the joint angles is clustered into four well defined regions. Thus, the vector $\mathbf{z}$ also forms a clustered space. This implicit relationship between the workspace and the joint-angle space when considered when initializing and training the network, greatly improves the capability of the network while decreasing the complexity of the network topology. It also follows from the kinematics of the manipulator that not all joint angle values will be allowed when it is expected to work in a given workspace due to dimensional constraints.

Using Quantum Clustering, we obtain the cluster centers in the space represented by $\mathbf{z}$. The initial values for $\mathbf{w}_r$ and θ_r are then extracted back from the six dimensional cluster centers. It should be noted that though a given joint angle determines a unique end-effector position, it is not true otherwise. An end-effector position maybe realized by more than one joint-angle values. This implicit information must

be accounted for while preparing the network. This forms one of the motivation for resorting to the proposed clustering method. It has been observed that such a selection needs a small fine-tuning of the cluster centers and hence has a very fast learning time. In addition, by an adaptive selection of clusters from the workspace, we have reduced the number of neurons significantly.

8.2.2 The Neural Learning Algorithm

The Learning Algorithm adopted in the present work is motivated by the extended Kohonen's Self-Organizing Feature Maps introduced by Walter and Schulten in [22]. Given an end-effector position target $\mathbf{u}_{target}$, a winner neuron μ is selected, based on the Euclidean distance metric in the workspace. The neuron whose reference vector is closest to the target is declared winner. The arm is given a coarse movement, θ_0^{out} by the initial output of the network determined by Eq. (7) resulting in the end-effector to move to a position $\mathbf{v}_0$. This if followed by a fine movement determined by θ_1^{out}, using Eq. (8), giving a correcting movement to the arm and the end-effector reaching the final position of $\mathbf{v}_1$. It has been shown that a series of similar corrective actions will converge to the correct end-effector position. However, we use only one corrective fine movement and achieve a considerable amount of accuracy.

The collective averaged output is evaluated using Eq. (7).

$$\theta_0^{out} = s^{-1} \cdot \sum_k h_k^{mix}(\theta_k + \mathbf{A}_k(\mathbf{u}_{target} - \mathbf{w}_k)) \tag{8.2}$$

where $s = \sum_k h_k^{mix}$ and $h_k^{mix} = exp(-\frac{\|\mu - k\|}{2\sigma_{mix}^2})$, with μ representing the winner neuron such that $\mathbf{w}_\mu$ is closest to the vector $\mathbf{u}_{target}$. θ_0^{out} gives a coarse movement to the arm such that the end-effector reaches a position $\mathbf{v}_0$. The correcting fine movement is evaluated using Eq. (8) resulting in a final movement of the end-effector to $\mathbf{v}_1$.

$$\theta_1^{out} = \theta_0^{out} + s^{-1} \cdot \sum_k h_k^{mix} \mathbf{A}_k(\mathbf{u}_{target} - \mathbf{v}_0) \tag{8.3}$$

The learning scheme used in the present work can be grouped as under:

$$\Delta \mathbf{v} = \mathbf{v}_1 - \mathbf{v}_0 \tag{8.4}$$

$$\Delta \theta^{out} = \theta_1^{out} - \theta_0^{out} \tag{8.5}$$

$$\Delta \theta_k = \theta_0^{out} - \theta_k^{out} - \mathbf{A}_k(\mathbf{v}_0 - \mathbf{w}_k) \tag{8.6}$$

$$\Delta \mathbf{A}_\mu = \|\Delta \mathbf{v}\|^{-2} \cdot (\Delta \theta^{out} - \mathbf{A}_\mu \cdot \Delta \mathbf{v}) \Delta \mathbf{v}^T \tag{8.7}$$

$$\mathbf{w}_k \leftarrow \mathbf{w}_k + \epsilon \cdot h_{\mu k} \cdot (\mathbf{u}_{target} - \mathbf{w}_k) \tag{8.8}$$

$$\theta_k \leftarrow \theta_k + \epsilon' \cdot h'_{\mu k} \cdot \Delta \theta_k \tag{8.9}$$

$$\mathbf{A}_k \leftarrow \mathbf{A}_k + \epsilon' \cdot h'_{\mu k} \cdot \Delta \mathbf{A}_k \tag{8.10}$$

The functions $h_{\mu k}$ and $h'_{\mu k}$ are defined as:

$$h_{\mu k} = exp(-\frac{\|\mu - k\|}{2\sigma^2}) \tag{8.11}$$

$$h'_{\mu k} = exp(-\frac{\|\mu - k\|}{2\sigma'^2}) \tag{8.12}$$

with μ and k representing the three dimensional indices associated with the corresponding neuron. The parameters ϵ, ϵ', σ, σ' and σ_{mix} vary during the training time depending on the current iteration and can be expressed using the general expression as below.

$$\eta = \eta_{initial}(\frac{\eta_{final}}{\eta_{initial}})^{(\frac{t}{t_{max}})} \tag{8.13}$$

In Eq. (18), $\eta \in \{\epsilon, \epsilon', \sigma, \sigma', \sigma_{mix}\}$, t is the current iteration and t_{max} is the total number of iterations to be performed by the network.

8.2.3 Indexing scheme for the adaptive topology of extended KSOM

In this work, we use an adaptive scheme for determining the topology of the workspace using Quantum Clustering of the joint space formed by grouping the end-effector positions and the corresponding joint angles. After the clustering step, a set

of points, $\mathbf{w}_1 \ldots \mathbf{w}_N$, in the workspace are obtained, which are the potential cluster centers and hence denote the reference or weight vector for the neurons. The index of each neuron is then obtained using a normalized and scaled version of the corresponding weight vector. The normalization is carried out by dividing each dimension by the corresposnding linear dimension of the workspace. It should be noted that in the proposed approach, the index values are real values and not necessarily integers, unlike the method in [15] and [22]. The motivation for the proposed indexing scheme is that it represents both the actual and the relative location of neurons in the workspace.

8.3 Quantum Clustering vs Kohonen's Self-Organizing Maps for Visual Motor Coordination

In the Kohonen's Self-organizing Feature Map based method, a fixed topology for the space is used by creating a 3-dimensional lattice of neurons. These neurons then organize themselves homogeneously so that the workspace is well spanned and the Eq. (6) can be used to identify the joint angles for a given end-effector position. This has two major disadvantages:

- The number of neurons have to be pre-decided and fixed, thus allowing little flexibility.
- The topology of the workspace is fixed once the lattice parameters are specified.

A fixed topology of the workspace is not necessary because that constraints the neurons to occur uniformly. The receptive field dimensions of all neurons in the visual system is not same. The proposed method has the same efficiency using almost half the neurons used in the K-SOM based approach and a flexible topology. Figure 8.3 shows the distribution of points in a workspace and the corresponding points in the

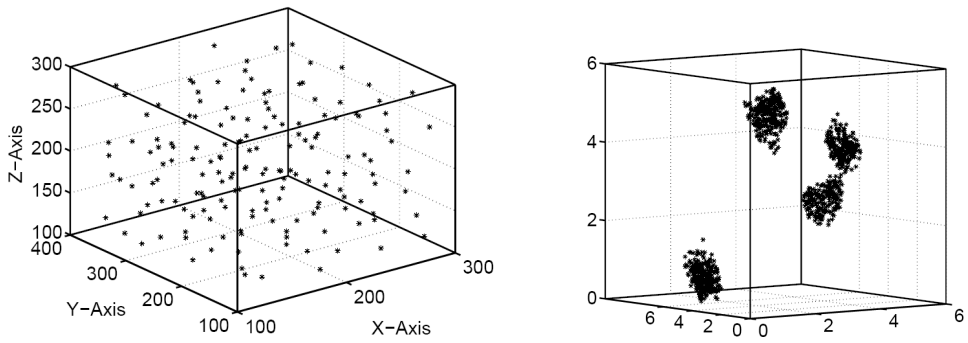

Figure 8.3: Plot of the reference centers of different neurons and corresponding joint angles obtained using QC-based method.

joint-angle space.

8.4 Results & Discussions

8.4.1 Parameters and Initialization

The workspace is defined by a region 200mm X 300mm X 200mm. The arm lengths are all equal to 254mm and the manipulator has a rigid wrist portion having a length equal to 50mm as used in [2]. The initialization of parameters was similar to that in [2]. The values were chosen as $\epsilon_i = 1.0$, $\epsilon_f = 0.05$, $\epsilon_i' = 0.9$, $\epsilon_i' = 0.9$, $\sigma_i = 2.5$, $\sigma_f = 0.01$, $\sigma_i' = \sigma_{mix_i} = 2.5$, $\sigma_f' = \sigma_{mix_f} = 0.01$. The subscript i and f denote the initial and final values of the corresponding parameter respectively.

8.4.2 Performance and Results

The topology of the workspace is determined by the number of clusters obtained. Unlike the Kohonen–SOM based approach where the workspace topology in the form of a 3–Dimensional map has to be pre–specified after a heuristic approach mostly based on hi–and–trial, the QC-based approach has only one free parameters in form of the cluster width parameter $q = \frac{1}{2\sigma^2}$, which controls the number of clusters generated. Each cluster denotes the receptive field of a neuron.

The proposed algorithm is tested on 200 Random Points in the workspace, a circle

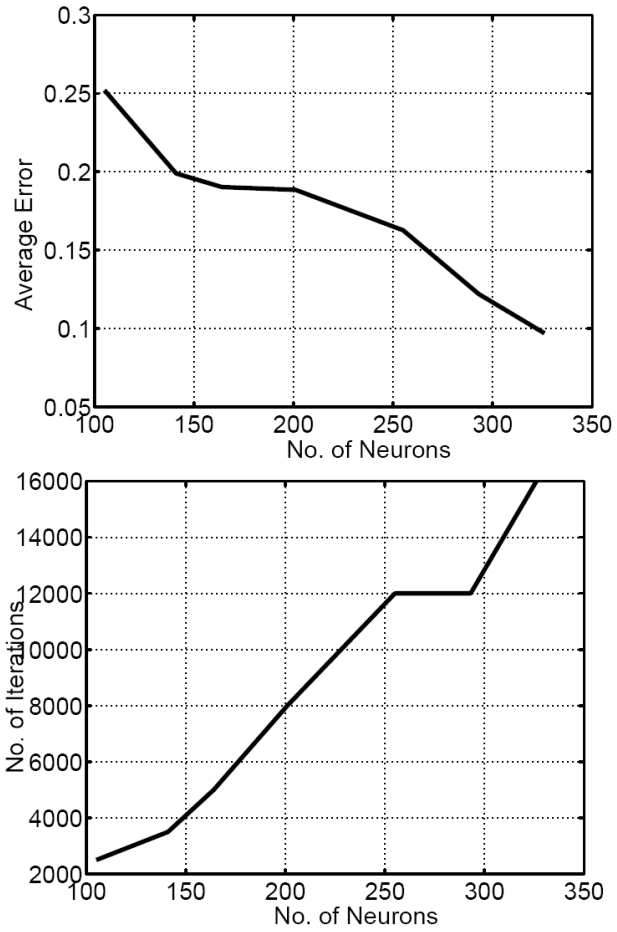

Figure 8.4: Plot of the Average Error and No. of iterations required as the number of neurons varies.

Table 8.1: Performance (in terms of the mean normed error) of Extended K-SOM based method when 5000 iterations were used to train the system.

SOM Topology	**No. of Neurons**	**200 Random Points**	**Circle**	**Sphere**
12x7x4	336	0.1438	0.0722	0.0711
8x7x5	280	0.1551	0.0750	0.0715
6x6x5	180	0.1931	0.1162	0.1133
6x5x5	150	0.2122	0.1001	0.0924

Table 8.2: Performance (in terms of the mean normed error) of Quantum Clustering based method when 5000 iterations are used.

$q = \frac{1}{2\sigma^2}$	**No. of Neurons**	**200 Random Points**	**Circle**	**Sphere**
0.14	255	0.1481	0.0803	0.0968
0.125	181	0.1804	0.10	0.11
0.12	164	0.1814	0.09	0.13
0.11	141	0.2291	0.1088	0.1334

in a 2–Dimensional plane and a Sphere. The error measured is reported using the Mean Normalized Error (MNE). MNE for a set of dataset with **N** points in the workspace is defined by Eq. (19).

$$\text{MNE} = \frac{1}{\mathbf{N}} \sum_{i=1}^{\mathbf{N}} \|\mathbf{u}_{target} - \mathbf{u}_{predicted}\| \tag{8.14}$$

We begin by evaluating the number of neurons which are necessary for representing the workspace appropriately. We observe that larger number of neurons give better results, but also require more training. This is evident from Figure 8.4. Figure 8.4 shows that increasing the number of neurons shows a decline in the MNE. However, for faster training, it is important to have only a sufficient number of neurons.

Our simulations lead us to the conclusion that the adaptive toplogy based scheme proposed in this work needs only 164 neurons for the workspace in use. This number can be compared with the work in [2], where 12x7x4 (=336) neurons are used.The result for three situations using different SOM Topology and also with different cluster widths is demonstrated in Tables 8.1 and 8.2. They show that the Quantum Clustering based method achieves better results with fewer neuron units, mainly because of the fact that, the topology of the workspace is not fixed but is determined

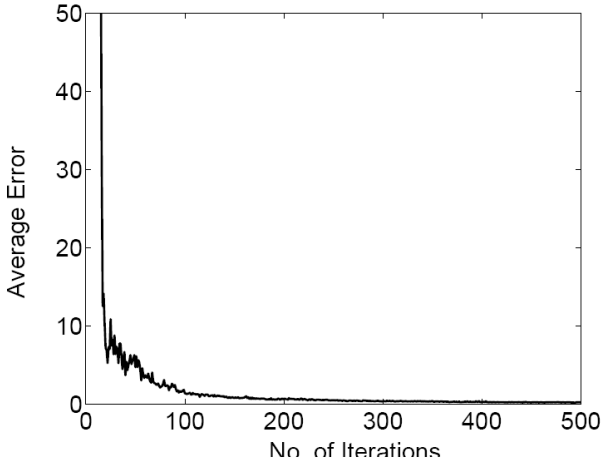

Figure 8.5: Plot of the Average Error with the number of iterations for 164 neurons.

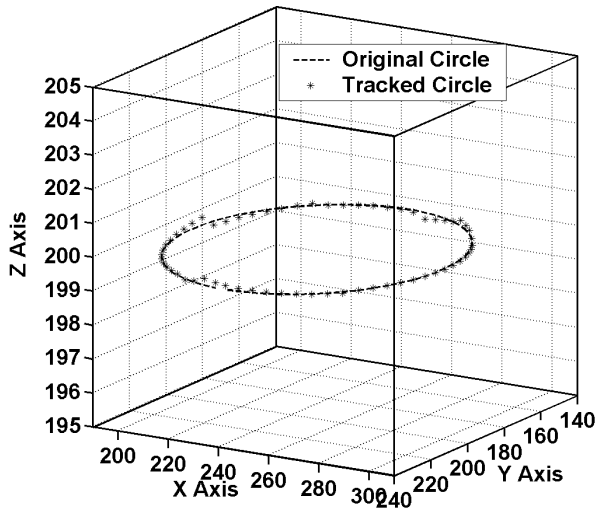

Figure 8.6: Plot of the tracked circle using 164 neurons.

through a joint clustering process.

The error vs. iterations graph is shown in Figure 8.5 for an adaptive scheme of 164 neurons. Figure 8.6 shows a circle being tracked in the workspace with an average error of 0.09mm. These results are comparable to that obtained using KSOM-based method where a much higher number of neurons were used.

CHAPTER 9

Nonlinear & Chaotic Systems

A trajectory of a chaotic system behaves in a random-like fashion. The set on which the chaotic motion is taking place attracts trajectories starting from a set of initial conditions. This attracting set is called a *chaotic attractor*. A nonlinear dynamical system with a chaotic attractor will produce motion on the attractor which has random-like properties.In particular, it is not possible to predict when a certain point or a small neigborhood of a point on the attractor will be visited by the system, except to say that it will be reached by the system in finite time. This property of a chaotic system is referred to as *ergodicity*. The task of the control algorithm is to use this ergodicity property and to bring the chaotic system's trajectory to a prespecified state in the system's state space [20].

9.1 Controlling Chaotic Systems

Consider the folowing two classes of single-output dynamical systems - one using nonlinear difference equation and the other using a nonlinear differential equation.

$$\boldsymbol{x}(k+1) = \boldsymbol{f}(\boldsymbol{x}(k), u(k)), \tag{9.1}$$

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), u(t)). \tag{9.2}$$

The control u, in both the cases, is bounded. The control u is either a specified function of time or a specified function of the system statem. The following assumptions are made regarding the system:

- There exists an open-loop control such that the system has a chaotic attractor.
- For a specified constant control $u = u_e$ there is a corresponding equilibrium state located in a vicinity of the chaotic attractor.

The equilibrium state satisfies the equation

$$\boldsymbol{x}_e = \boldsymbol{f}(\boldsymbol{x}_e, u_e)$$

for the discrete-time system, and it satisfies the relation

$$\mathbf{0} = \boldsymbol{f}(\boldsymbol{x}_e, u_e)$$

for the continuous-time system. We now describe the way the controllable target is obtained for the systems of interest.

The linearized discrete-time system is given by

$$\Delta\boldsymbol{x}(k+1) = \boldsymbol{A}\Delta\boldsymbol{x}(k+1) + \boldsymbol{b}\Delta u(k),$$

where $\Delta\boldsymbol{x}(k) = \boldsymbol{x}(k) - \boldsymbol{x}_e$ and $\Delta u(k) = u(k) - u_e$. Then the LQR method constructs the state variable feedback controller of the form $\Delta u(\Delta\boldsymbol{x}) = -\boldsymbol{k}\Delta\boldsymbol{x}$, that guarantees the origin of the linearized system,

$$\Delta\boldsymbol{x}(k+1) = (\boldsymbol{A} - \boldsymbol{b}\boldsymbol{k})\Delta\boldsymbol{x}(k), \tag{9.3}$$

to be asymptotically stable. Consider a Lyapunov function of the form $V(\Delta\boldsymbol{x}) = \Delta\boldsymbol{x}^T\boldsymbol{P}_l\Delta\boldsymbol{x}$, where $\boldsymbol{P}_L$ is obtained by solving the corresponding Lyapunov equation

for 9.3.The controllable target for the nonlinear system driven by a bounded controller of the form

$$u = -\text{sat}(\boldsymbol{k}\Delta\boldsymbol{x}) + u_e, \tag{9.4}$$

where

$$\text{sat}(v) = \begin{cases} u_{max} - u_e & \text{if} \quad v > u_{max} - u_e, \\ v & \text{if} \quad u_{min} - u_e \leq v \leq u_{max} - u_e, \\ u_{min} - u_e & \text{if} \quad v < u_{min} - u_e, \end{cases}$$

and the bounds u_{min} and u_{max} are specified by a designer. Using the above control law, we seek the largest level curve $V(\boldsymbol{x}) = V_{max}$ such that for the points $\{\boldsymbol{x} : V(\boldsymbol{x}) \leq V_{max}\}$, for the discrete-time system we have $\Delta V(k) = V(\boldsymbol{x}(k+1)) - V(\boldsymbol{x}(k)) < 0$ and for the continuous-time system we get $\dot{V(t)} = 2(\boldsymbol{x} - \boldsymbol{x}_e)^T \dot{\boldsymbol{P}}_l \dot{\boldsymbol{x}} < 0$.

The Chaotic Control Algorithm is then given by the following algorithm:

IF $V(\boldsymbol{x}) > V_{max}$

THEN u = open-loop control to keep the system in the chaotic attractor

ELSE u = $-\text{sat}(\boldsymbol{k}\Delta\boldsymbol{x}) + u_e$

END

We illustrate the chaotic algorithm when applied to the Hénon map and the Bouncing Ball map.

9.2 The Henon Map

The dynamical system known as the Hénon map is described by the difference equations

$$x_1(k+1) = -1.4x_1^2(k) + x_2(k) + 1 \tag{9.5}$$

$$x_2(k+1) = 0.3x_1(k) \tag{9.6}$$

Applying a control input to the first equation of 9.5 gives us the following set of equations:

$$x_1(k+1) = -1.4x_1^2(k) + x_2(k) + 1 + u(k) \tag{9.7}$$

$$x_2(k+1) = 0.3x_1(k) \tag{9.8}$$

The orbit of $\boldsymbol{x}(0) = [1\,0]^T$ of the Hénon map is shown in Figure 9.1.

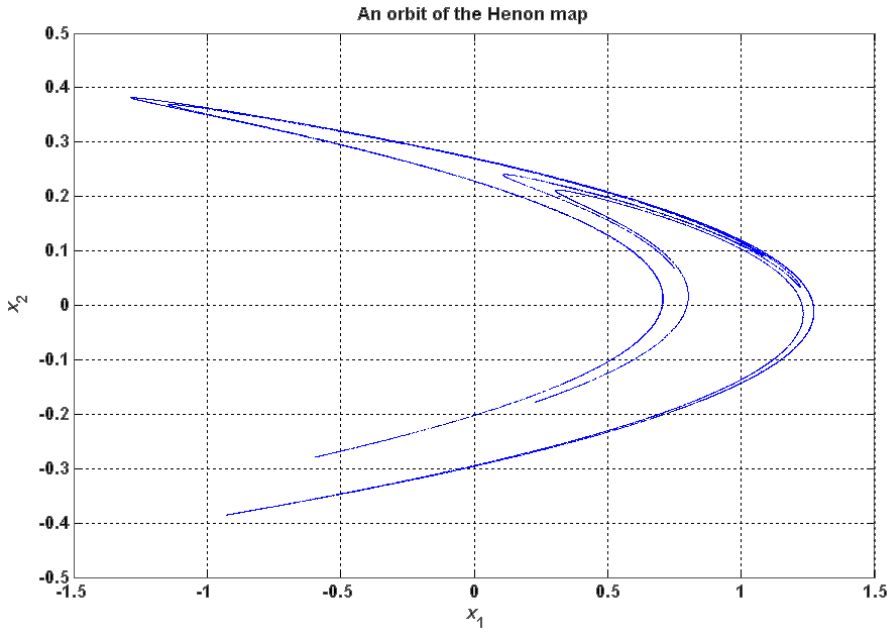

Figure 9.1: An orbit of the Hénon map.

9.2.1 Chaotic Control Algorithm

The two fixed states for the system are

$$[0.6314\,0.1894]^T \text{and} [-1.1314\ -0.3394]^T.$$

We use the first of the above fixed states and linearize the system. The control law

$$u = -0.3\,\text{sat}(\boldsymbol{k}(\boldsymbol{x}(t) - \boldsymbol{x}_e)/0.3) \tag{9.9}$$

is obtained by solving the LQR problem. Solving the Lyapunov equation gives us $V_{max} = 0.0109$. Plots of x_1 and x_2 versus time for the closed-loop system in 9.7 using

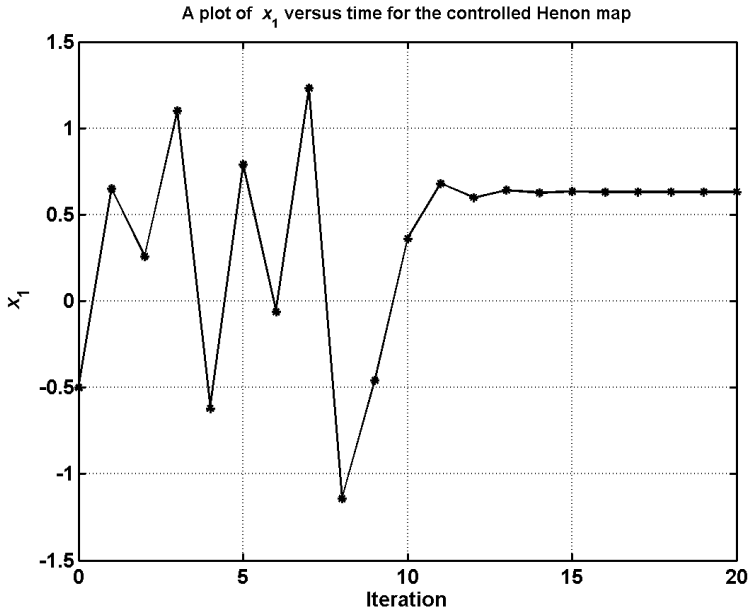

Figure 9.2: Plot of x_1 versus time for the controlled Hénon map.

the control law in 9.9 are shown in Figures 9.2 and 9.3 respectively. The orbit of $\boldsymbol{x}(0) = [-0.5\,0]^T$ of the controlled Hénon map is shown in Figure 9.4, such that the controllable target is shown as the set of points inside the ellipse $V(\boldsymbol{x} - \boldsymbol{x}_e) = V_{max} = 0.0109$.

9.3 The Bouncing Ball

The bouncing ball on a vibrating plane is a classical example of a chaotic system [21]. For certain frequencies of the ball it is possible to get periodic motion for any integer period. Some of these periodic orbits are stable, while some are not. In this work, we simulate the motion of a bouncing ball and try to control it using a chaotic algorithm. The ball can then be controlled to undergo two kinds of motion:

1. It can bounce to a fixed height for every n cycles of the plate

2. It can bounce to m different heights for $n \times m$ cycles of the plate

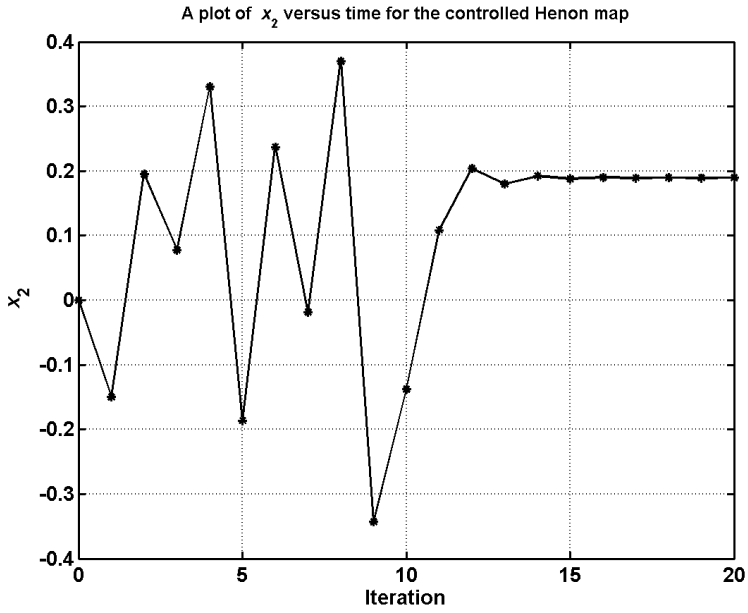

Figure 9.3: Plot of x_2 versus time for the controlled Hénon map.

Let

$$\phi(k) = \omega t(k) \text{and} \psi(k) = 2\omega V(k)/g,$$

where $\phi(k)$ is the current phase angle of the plate and $\psi(k)$ is the phase angle change between the current bounce of the ball and the next. Also we define

$$a_1 = \frac{1+e}{1+M} \text{ and } a_2 = \frac{M-e}{1+M},$$

where M and e are the mass ration of the ball to the plate and coefficient of restituation respectively. Assume the plate moves at a nominal frequency $\overline{\omega}$. Then the *high-bounce* ball map is given by

$$\phi(k+1) = \phi(k) + \frac{\omega_k}{\overline{\omega}}\psi(k), \tag{9.10}$$

$$\psi(k+1) = -a_2\psi(k) + \hat{a_1}\omega^2\, cos(\phi(k+1)), \tag{9.11}$$

where $\hat{a_1} = 2Aa_1/g$. Note that 9.10 is evaluated *modulo* 2π since ϕ refers to a physical position of the plate. The second equation is not evaluated *modulo* 2π since the plate may go through more than one cycle before the next bounce. The height of the ball

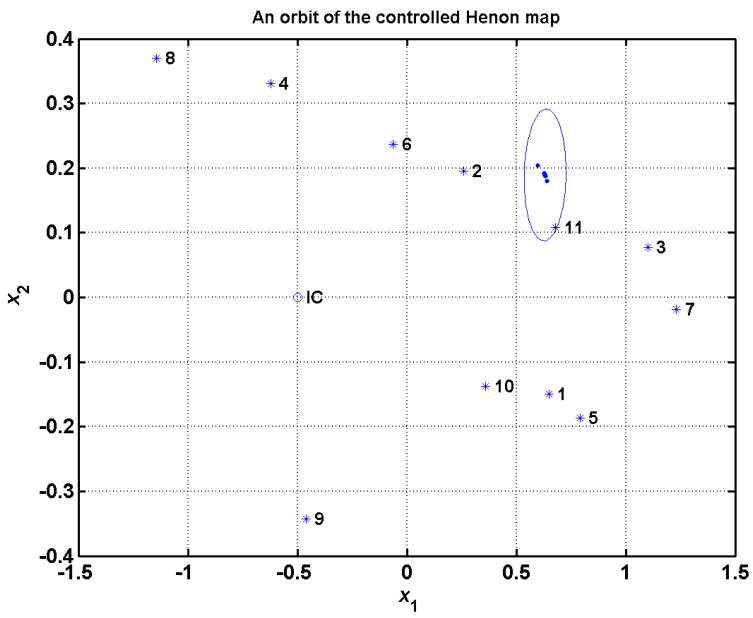

Figure 9.4: An orbit of the controlled Hénon map. The initial condition is denoted as 'o 'and labeled 'IC '. The itertions in the controllable target are denoted as '. '.

can then be calculated using the laws of motion and the maximum height is given by

$$h_{max} - h_k = \frac{g}{8\overline{\omega}^2}\phi_k^2, \tag{9.12}$$

where h_k is the height of the ball when the last bounce takes place. In the high-bounce map, we set $h_k = 0$. The following parameters are used for simulating the high-bounce map

$$\begin{aligned} M &= \tfrac{1}{26} \\ e &= 0.8 \\ A &= 0.013\text{meters} \end{aligned}$$

so that

$$\begin{aligned} a_1 &= 1.73333 \\ a_2 &= -0.73333 \end{aligned}$$

$$\hat{a}_1 \quad = 0.004594\text{sec}^2$$

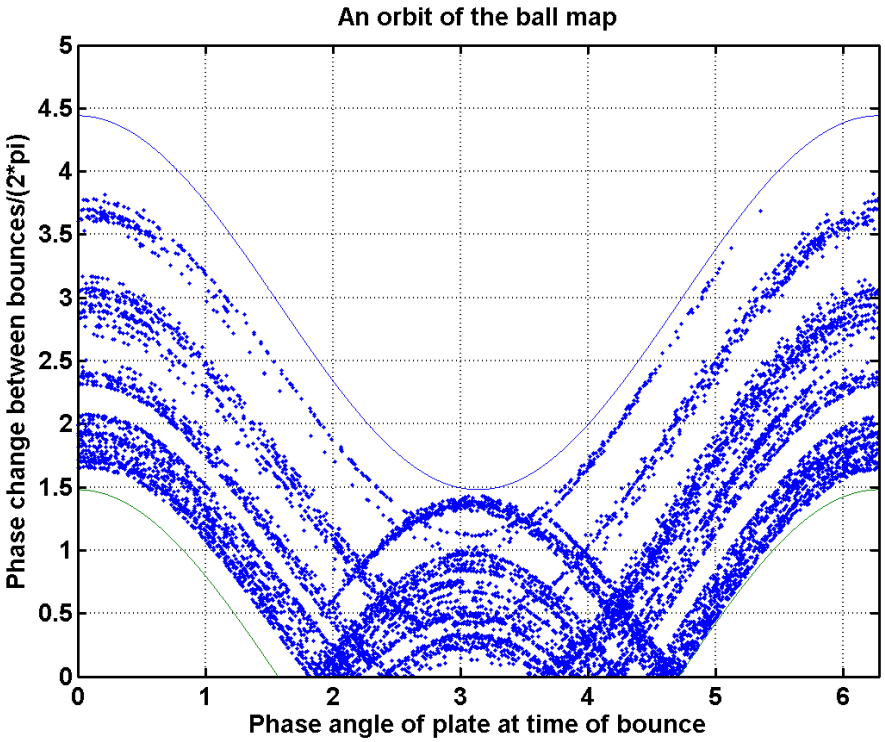

Figure 9.5: An orbit of the bouncing ball map.

9.3.1 Chaotic Control Algorithm

The task is to control the ball at a particular height. In our discussions, we use $\omega = 45\text{rad/sec}$ and V_{max} is evaluated to be 5.006. Figure 9.5 shows the orbit of the bouncing ball map. The folowing control algorithm is adopted for the bouncing ball

IF $V(\phi - \hat{\phi}) > V_{max}$

THEN $\omega = 45\text{rad/sec}$

ELSE $\omega = 22\text{rad/sec}$

END

An orbit of the controlled map is shown in Figure 9.6. A plot of $\phi(k)$ and $\psi(k)$ versus time is shown in Figure 9.7 and 9.8 respectively. The height of the controlled ball is

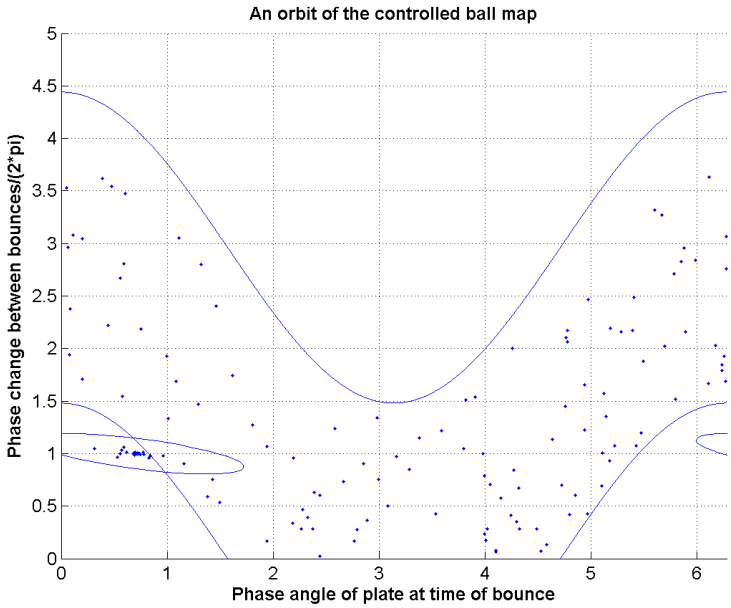

Figure 9.6: An orbit of the controlled modified ball map.

shown in Figure 9.9.

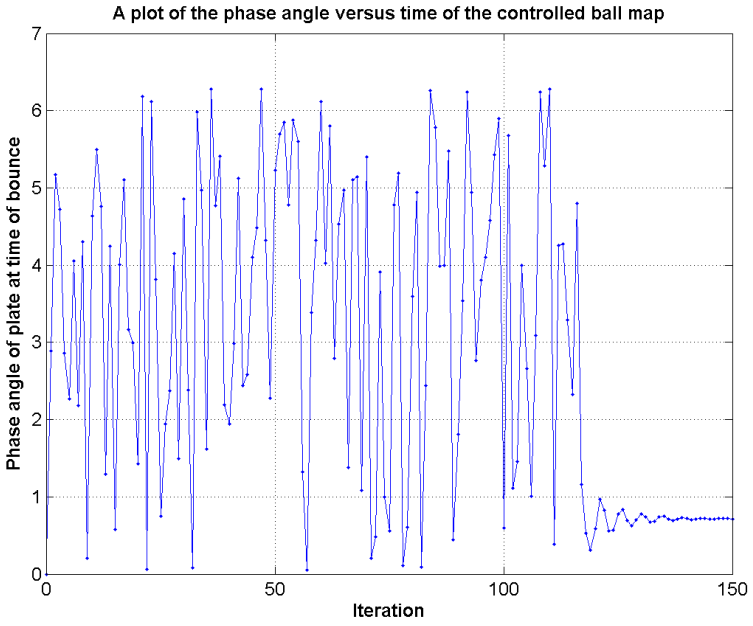

Figure 9.7: A plot of $\phi(k)$ versus time of the controlled modified ball map.

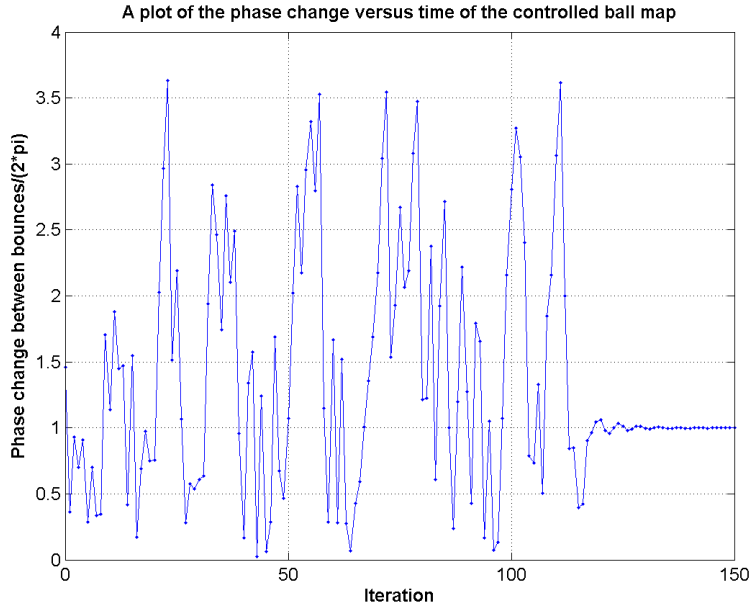

Figure 9.8: A plot of $\psi(k)$ versus time of the controlled modified ball map.

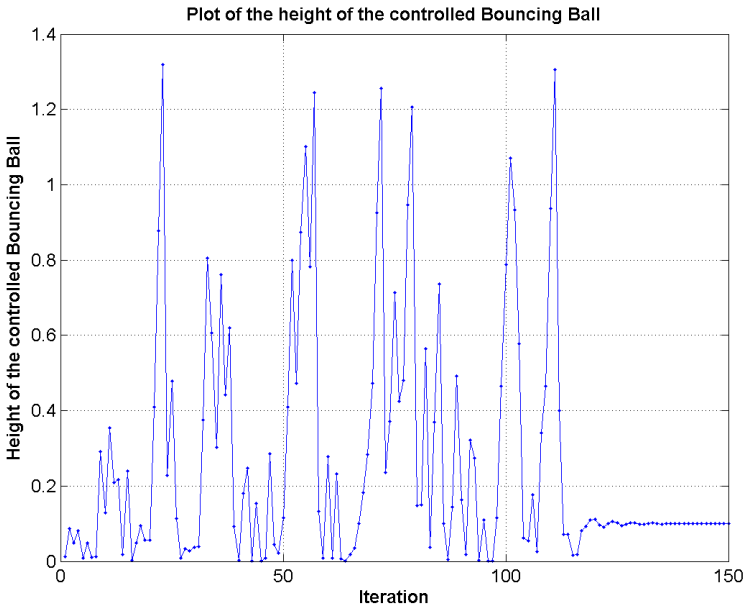

Figure 9.9: A plot of the height of the ball versus time of the controlled modified ball map.

CHAPTER 10

Backstepping Control of Nonlinear Systems

Robust control os nonlinear systems with uncertainties is of prime importance in various applications. The model of many nonlinear systems can be expressed in a special state-space form

$$\begin{aligned}
\dot{x}_1 &= F_1(x_1) + G_1(x_1)x_2 \\
\dot{x}_2 &= F_2(x_1, x_2) + G_1(x_1, x_2)x_3 \\
&\dots \\
\dot{x}_m &= F_m(x_1, x_2, \dots, x_m) + G_1(x_1, x2, \dots, x_m)u
\end{aligned}$$

where $x_i \in \Re^n, i = 1, 2, \dots, m$ denote the states of the system, $u \in \Re^n$ is the vector of control inputs, $F_i, G_i \in \Re^{n \times n}, i = 1, 2, \dots, m$ are nonlinear functions that contain both parametric and nonpartametric uncertainties, and G_i's are known and invertible [12].

10.1 A Neural Backstepping Control of Nonlinear Systems

The Neural Network based design of a nonlinear systemin the strict-feedback form as discussed above can be summarized as in Figure 10.1. The design of the controller structure is a three step process and and the design aims to track a desired signal.

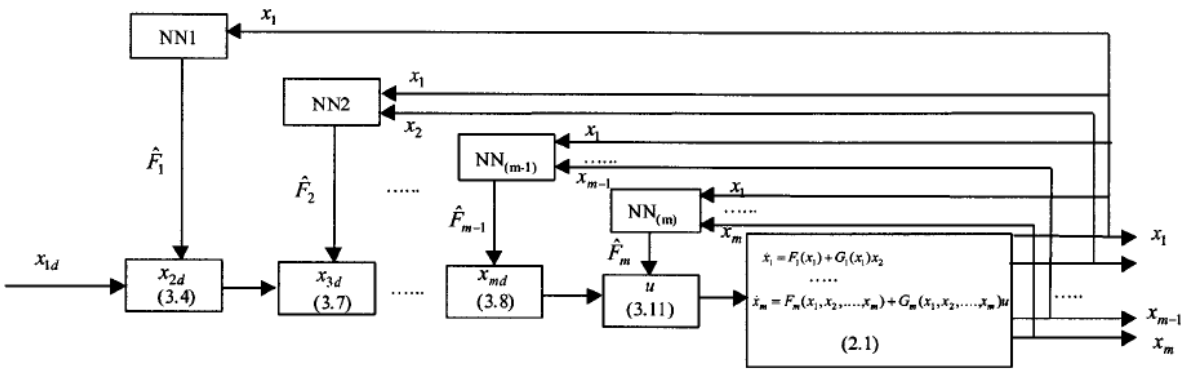

Figure 10.1: The backstepping control Neural Network control algorithm for nonlinear systems in the strict-feedback form.

Given a desired trajectory x_{1d}, the task is to make x_1 follow this. The algorithm can be summarized as below.

- *Step 1–Design Fictitious Controllers for* $x_2, x3, \ldots, x_m$*:* Consider the following fictitious controller for x_{2d}

$$x_{2d} = G_1^{-1}(-\hat{F}_1 + \dot{x}_{1d} - K_1 e_1) \tag{10.1}$$

with $K_1 > 0$ a design parameter, $\hat{F}_1$ the estimate of F_1. This yields the error dynamics given by

$$\dot{e}_1 = F_1 - \hat{F}_1 - K_1 e_1 + G_1 e_2. \tag{10.2}$$

The fictitious controller for x_{3d} is then of the form

$$x_{3d} = G_2^{-1}(-\hat{F}_2 + \dot{x}_{2d} - K_2 e_2 - G_1^T e_1), \tag{10.3}$$

where K_2 is a design parameter and $\hat{F}_2$ the estimate of F_2. Similarly, the fictitious controller for x_{md} is of the form

$$x_{md} = G_{m-1}^{-1}(-\hat{F}_{m-1} + \dot{x}_{(m-1)d} - K_{m-1} e_{m-1} - G_{m-2}^T e_{m-2} + G_{m-1} e_m), \tag{10.4}$$

where $K_{m-1} > 0$ is a design parameter and $\hat{F}_{m-1}$ the estimate of F_{m-1}.

- *Step 2–Design of Actual Control u:* The actual control u is of the form

$$u = G_m^{-1}(-\hat{F}_m + \dot{x}_{md} - K_m e_m - G_{m-1}^T e_{m-1}) \tag{10.5}$$

leading to the following error dynamics for error e_m

$$\dot{e}_m = F_m - \hat{F}_m - K_m e_m + G_{m-1}^T e_{m-1}, \tag{10.6}$$

with $K_m > 0$ is a design parameter and $\hat{F}_m$ the estimate of F_m.

- *Step 3–Closed-loop stability and performance analysis of NN weight tuning algorithm:* All Neural Networks used in the algorithm are Radial Basis Function Networks with a Gaussian basis function. The overall network is a CMAC (Cerebellar Model Arithmetic Computer) neural network.

10.2 Results

We demonstrate the algorithm on a nonlinear dynamical system discussed below.

10.2.1 A Dynamical System Control

We use the Robust Backstepping Control Algorithm for establishing control to a dynamical system given by the following set of equations:

$$\dot{x}_1 = x_2, \tag{10.7}$$

$$\dot{x}_2 = (-10sin(x_1) - x_2) + x_3, \tag{10.8}$$

$$\dot{x}_3 = 20(-10x_2 - 0.5x_3 + u), \tag{10.9}$$

where u is the control input. The desired trajectory was a sinusoid for the state variable x_1. The tracking error is plotted in Figure 10.2, while 10.3 demonstrates the control input for the system with varying time. The three state variables are tracked in Figures 10.4, 10.5 and 10.6.

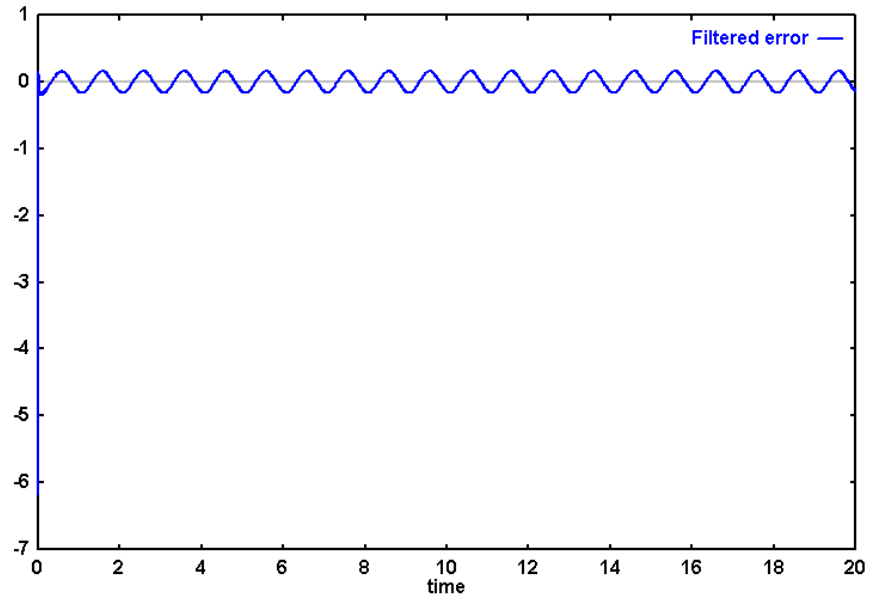

Figure 10.2: The plot of filtered error as obtained using the Backstepping algorithm for the dynamical system.

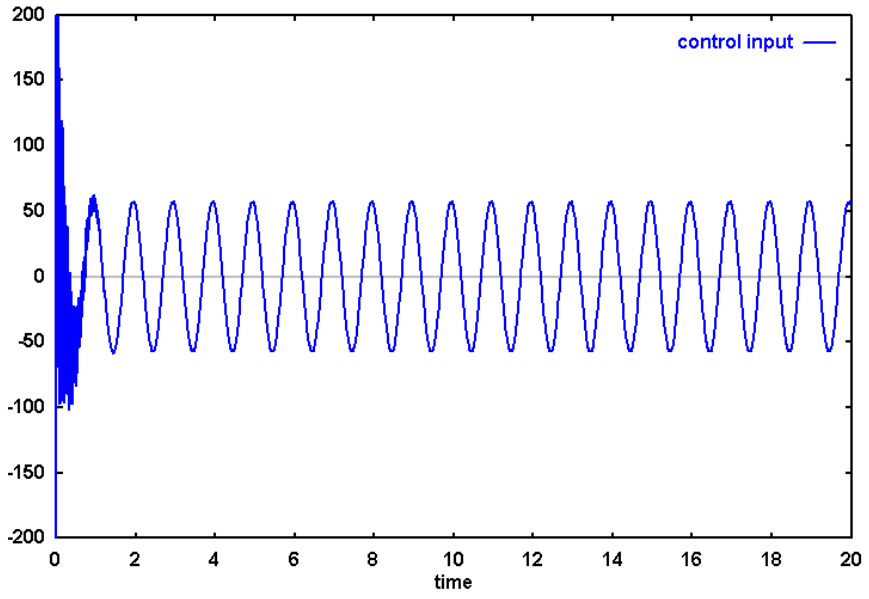

Figure 10.3: Plot of the control input to the dynamical system.

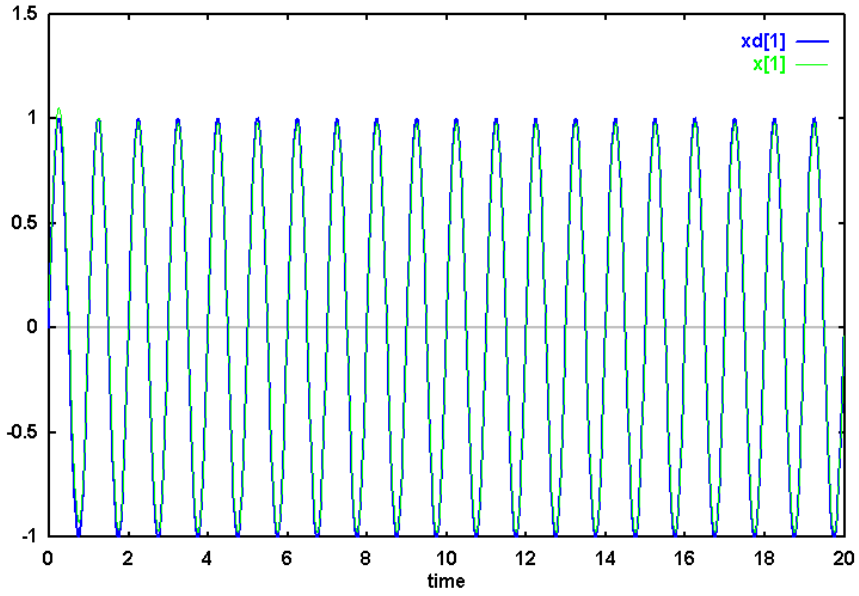

Figure 10.4: The plot of desired versus actual value of the state variable x_1.

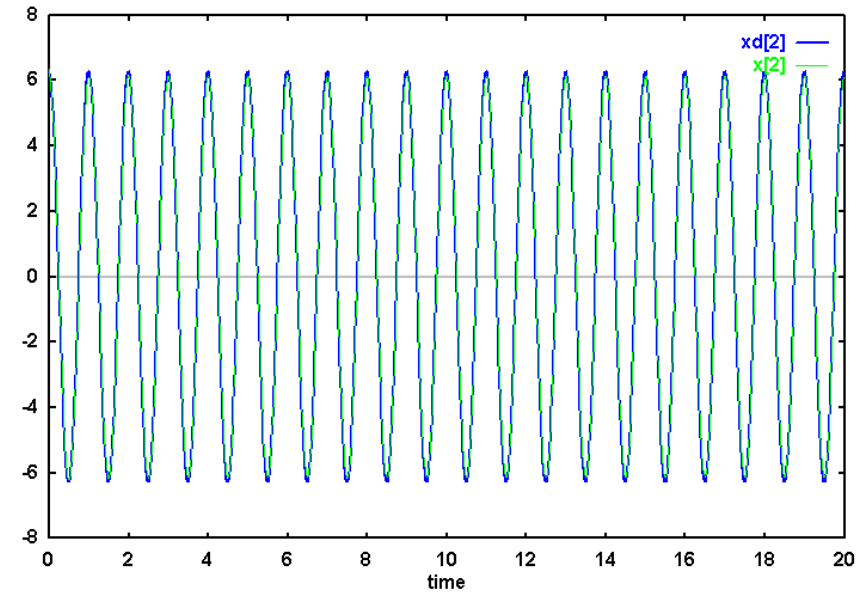

Figure 10.5: The plot of desired versus actual value of the state variable x_2.

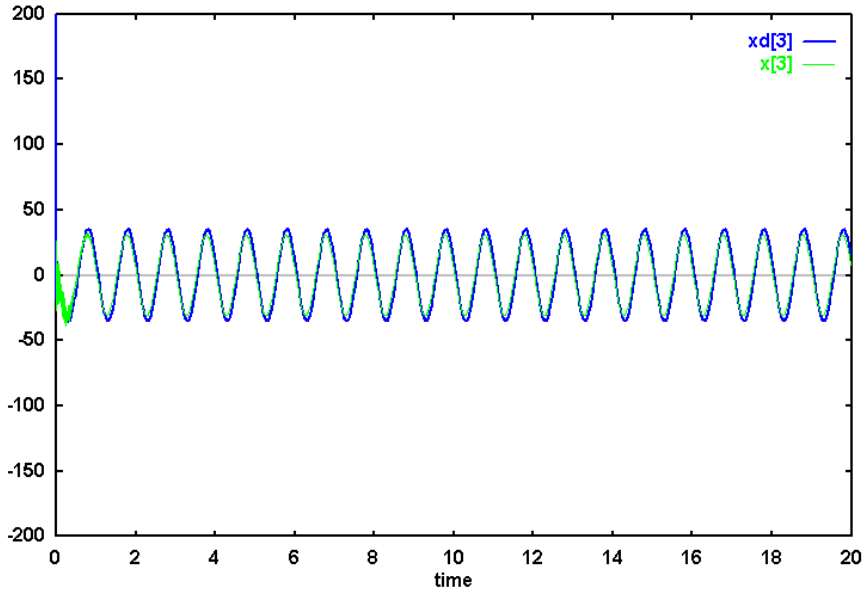

Figure 10.6: The plot of desired versus actual value of the state variable x_3.

The neural network based backstepping control algorithm is a robust and faster method to control a nonlinear system in strict-feedback form.

CHAPTER 11

Conclusions

An attempt to study the nature of Hilbert spaces and its use in Optimal Control, System Identification, Information Characterization and Learning methods is investigated in this work. Hilbert spaces provide special geometrical properties, some of which are discussed and demonstrated in this work. Linear Operators in Hilbert spaces provide a simple solution to the time varying optimal control problem. The study in this context was done by deriving the Matrix Riccati Equation for the LQR problem.

Function Aprroximation using RKHS was discussed and the simulations using the static and iterative methods were demonstrated. The iterative regularized method provide a better technique to handle noisy observations. The problem of Clustering using Hilbert spaces is discussed using recently proposed Quantum Clustering technique. We also use this method for a joint partitioning scheme for visual-motor coordination of a robot manipulator. This gives us a faster method for training and better accuracy on existing techniques.

The world of Hilbert spaces is much more than what has been discussed in this work. However, an attempt to highlight some advances in the area has been made.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] R.W. Beard. Linear operator equations with applications in control and signal processing. *IEEE Control System Magazine*, pages 69–79, 2002.

[2] L. Behera and N. Kirubanandan. A hybrid neural control scheme for visual-motor coordination. *IEEE Control Systems Magazine*, 19:34–41, 1999.

[3] J. Capon. Hilbert space methods for detection theory and pattern recognition. *IEEE Transactions on Information Theory*, 11:247–259, 1965.

[4] L. Debnath and P. Mikusinski. *Introduction to Hilbert Spaces with Applications*. Academic Press Inc., San Diego, USA, 1990.

[5] T.J. Dodd and R.F. Harrison. Iterative solution to approximation in repreducing kernel hilbert spaces. In *CD-ROm Proc. of 2002 IFAC World Congress*, 2002.

[6] T.J. Dodd, V. Kadirkamanathan, and R.F. Harrison. Function estimation in hilbert spaces using sequential projections. In *Proc. of the IFAC World Congress on Intelligent Control Systems and Signal Processing (ICONS 2003)*, pages 113–118, 2003.

[7] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification*. Wiley, New York, 2 edition, 2001.

[8] D. Horn. Clustering via hilbert space. *Physica A*, 302:70–79, 2001.

[9] D. Horn and A. Gottlieb. The method of quantum clustering. In *NIPS 2001*, pages 769–776, 2001.

[10] K.R.Muller, S. Mika, G. Ratsch, K. Tsuda, and B. Scholkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12:181–201, 1999.

[11] N. Kumar and L. Behera. A quantum clustering based neural control scheme for visual motor coordination. In *Proc. of the Fifth International Conference on Pattern Recognition (ICAPR 2003)*, pages 118–121, 2003.

[12] C. Kwan and F.L. Lewis. Robust backstepping control of nonlinear systems using neural networks. *IEEE Transactions on Systems, Man and Cybernetics – Part A: Systems and Humans*, 2000.

[13] V.I. Lebedev. *An Introduction to Functional Analysis and Computational Mathematics*. Birkhauser, Boston, 1996.

[14] L. Ljung. *System Identification: Theory for the User*. Prentice-hall PTR, New Jersey, USA, 1987.

[15] T.M. Martinetz, H.J. Ritter, and K. Schulten. Three dimensional neural network for learning visuomotor coordination of a robot arm. *IEEE Transactions on Neural Networks*, 1:131–136, 1990.

[16] L. Mate. *Hilbert Space methods in Science and Engineering*. Adam Hilger, Hungary, 1989.

[17] B.D. Ripley. *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge, UK, 1996.

[18] S.J. Roberts. Non-parametric unsupervised cluster analysis. *Pattern Recognition*, 30:261–272, 1997.

[19] V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, Inc., New York, 1998.

[20] T.L. Vincent. Control using chaos. *IEEE Control System Magazine*, pages 65–76, 1997.

[21] T.L. Vincent and A.I. Mees. Controlling a bouncing ball. *International Journal of Bifurcation and Chaos*, 10:579–592, 2000.

[22] J.A. Walter and K.J. Schulten. Implementation of self-organizing neural networks for visuomotor control of an industrial robot. *IEEE Transactions on Neural Networks*, 4:86–95, 1990.

[23] J. Weidmann. *Linear Operators in Hilbert Spaces*. Springer Verlag, New York, USA, 1980.