An Incremental Approach to Training Data Selection in Neural Networks

Sethu Vijayakumar

May 9, 1996

Contents

1 Introduction **7**

1.1 Background/framework 7

1.2 Least squares estimation 9

1.2.1 The least squares estimate 9

1.2.2 Minimum-Variance Unbiased Estimate (Gauss-Markov Estimate) 10

1.2.3 Minimum-Variance Estimate 11

1.2.4 Shortcomings of the least squares estimation model in NN learning 12

1.3 The What & Why of this research 13

1.4 Organization of the dissertation 14

2 Functional analytic approach to NN learning **15**

2.1 Introduction 15

2.2 Learning as an inverse problem 15

2.3 Analogy between conventional model and the functional analytic approach 18

2.3.1 Conventional neural network and learning 18

2.3.2 Learning in neural networks as used in the dissertation 20

2.4 Generalizing ability and optimization criteria 22

2.4.1 Memorization criterion 23

2.4.2 Wiener criterion 24

2.4.3 Projection criterion 25

2.5 Conclusion 26

3 Incremental Learning **29**

3.1 Introduction 29

3.2 Incremental learning 29

3.3 Incremental Wiener learning 33

3.4 Conclusion 37

4 Moore-Penrose Generalized Inverse **39**
4.1 Introduction . . . 39
4.2 Mathematical definition and characteristic properties . . . 39
4.3 Recursive computation of generalized inverse . . . 41
4.4 Conclusion . . . 44

5 Incorporating training data incrementally **45**
5.1 Introduction . . . 45
5.2 Incorporation of new training data . . . 45
5.3 Computational complexity analysis . . . 52
5.4 Data Sampling for reduced computational complexity . . . 53
5.5 Conclusion . . . 55

6 Training data selection for improved generalization **57**
6.1 Introduction . . . 57
6.2 Information measure . . . 57
6.3 Modified Wiener criterion based on incremental sampling . . . 58
6.4 Training data selection for improved generalization . . . 59
6.5 Computer simulations . . . 61
6.5.1 Results of simulation (I) . . . 62
6.5.2 Results of simulation (II) . . . 65
6.5.3 Interpretation of the simulation results . . . 67
6.6 Conclusion . . . 71

7 Conclusion **73**
7.1 Results . . . 73
7.2 Future objectives . . . 74

Acknowledgements **74**

Bibliography **75**

List of Figures

1.1 General neural network architecture 8
1.2 Training data, representative function $f(x_m) = y_m$ and learned function f_0. . . . 9

2.1 NN as a real valued function 16
2.2 NN learning as an inverse problem 17
2.3 Conventionally used NN architecture 18
2.4 nth hidden unit of the neural network 19
2.5 NN architecture used in this dissertation 20
2.6 Functional analytic approach to NN learning. 21
2.7 NN learning as an inverse problem 23
2.8 Projection learning 26

3.1 Neural network learning model. The two stages involved: (i) The selection of training data and hence, the sampling operator A, (ii) Evaluating the optimum learning operator X. 30
3.2 Incremental learning with additive training data 31
3.3 Geometrical interpretation of a Wiener learning. For an operator X satisfying the Wiener criterion, XA provides an oblique projection f_1 of $f \in \mathcal{R}(R)$ onto $\mathcal{R}(RA^*)$ along $\mathcal{R}(R) \cap \mathcal{N}(A)$. 35
3.4 Relationship between the Wiener and Projection learning operators. Wiener learning is a projection on $\mathcal{R}(RA^*)$ whereas Projection learning is a projection on $\mathcal{R}(A^*)$, both along $\mathcal{N}(A)$. 36

5.1 Operator T mapping $y^{(m)}$ from R^m into R^{m+1} space 46
5.2 Structure of the matrix U_{m+1} 49
5.3 Different operators and their relationship with respect to the original function space and the sampled spaces. 50
5.4 Structure of the final result for incremental computation of the Wiener learning operator. 51

6.1 Sampling for computational efficiency versus generalizing ability 60
6.2 Simulation result for original function $f \in \mathcal{R}(R)$ using three sample points 68
6.3 Simulation result for original function $f \in \mathcal{R}(R)$ using two sample points 68
6.4 Simulation result for original function $f \notin \mathcal{R}(R)$ using three sample points 69
6.5 Simulation result for original function $f \notin \mathcal{R}(R)$ using two sample points 69
6.6 Simulation results for original function $f \in \mathcal{R}(R)$ using three sample points . . . 70
6.7 Simulation results for original function $f \in \mathcal{R}(R)$ 70

Notation	Affectation
NN	Neural Network
x_m	Sample point
$\{x_m\}_{m=1}^{M}$	Training set
y_m	Sample value
y	Sample vector
$y^{(n)}$	Sample vector using n sample values
$\{x_m, y_m\}_{m=1}^{M}$	Training data
$\{e_m\}_{m=1}^{M}$	Natural basis
$\{e_n^{(m)}\}_{n=1}^{m}$	Natural basis of m dimensional vector space
H	Hilbert space of functions to be approximated
R^m	Sampled space of dimension m
A	Sampling operator
A_m	Sampling operator for m sample points
X	Learning operator
X_m	Learning operator corresponding to A_m
R	Correlation operator of functions to be approximated
$(.,.)$	Inner product
$\bar{.}$	complex conjugate
$. \otimes \bar{.}$	Schatten product
$tr(X)$	Trace of X
u	Vector u
u^T	Transpose of vector u
$A^\dagger$	Moore-Penrose generalized inverse of A

Notation	**Affectation**
$\mathcal{R}(A)$	Range of A
$\mathcal{N}(A)$	Null space of A
$rank(A)$	Rank of matrix A
$dim(S)$	Dimension of space S
$S^{\perp}$	Orthogonal complement of a subspace S
A^{*}	Adjoint operator of A
$\|A\|$	Norm of operator A
I	Identity matrix
$f(x)$	Original function to be approximated
$f_0(x)$	Learned function
$f_1(x), ...f_m(x), f_{m+1}(x)$	Learned function due to $1, ..., m, m+1$ sample points
J_M	Memorization learning criterion
J_W	Wiener learning criterion
J_P	Projection learning criterion

Chapter 1

Introduction

1.1 Background/framework

Neurocomputing is the technological discipline concerned with parallel, distributed, adaptive information processing systems that develop information processing capabilites in response to exposure to an information environment. Neurocomputing is a fundamentally new and different approach to information processing. It is the first alternative to *programmed computing*, which has dominated information processing for the past half century. The primary information processing structures of interest in neurocomputing are *neural networks*. The formal definition of a neural network follows.

A neural network *is a parallel, distributed information processing structure consisting of* processing elements, *which can possess a local memory and carry out localized information processing operations, interconnected via unidirectional signal channel called* connections. *Each processing element has a single output connection that branches or "fans out"into as many collateral connections as desired; each carries the same signal refered to as* the processing element output signal. *The processing element output signal can be of any mathematical type desired. The information processing that goes on within each processing element can be defined arbitrarily with the restriction that it has to be completely local, that is, it must depend only on the current values of the input signals arriving at the processing element via impinging connections and on values stored in the processing element's local memory.*

Looking briefly at the development of the field of neurocomputing, one of the early beginings can be traced back to Rosenblatt[1962], who proved the convergence of a *learning algorithm* for the simplest class of neural networks called the *perceptrons*. However, Minsky and Papert[1969] proved the disability of these networks to learn some simple tasks for networks with no hidden layers. Studies on networks with multiple layers including hidden layers were studied but a weight changing algortihm was not developed until later. Another theme was the *associative content-adressable memory*, in which different input patterns became associated with one an-

other if sufficiently similar. This was studied by Anderson[1968], Marr[1969], Kohonen[1974-89] etc. Another approach was using the statistical method and *energy functions* introduced by Hopfield[1982]. Statistical mechanics was then brought to bear on the stochastic network problem by Amit et al.[1989] using methods developed in theory of random magnetic systems called *spin glasses.*

In most, but not all, neural networks that have learning capabilities, the learning is accomplished through modification of connection weights. Various people have developed algorithms which works quite well for adjusting the weights connecting units in succesive layers of multilayer perceptrons. One of the most popular methods, known as *back-propagation*, appears to have been found first by Werbos[1974] and then independently rediscovered around 1985 by Rumelhart, Hinton and Williams. Much of the current activity is centered on back-propagation and its extensions.

There are many exciting implications of the possibility of *training* a network to do computations. Instead of having to specify every detail, we simply have to compile a training set of representative examples. This means that we can expect to solve problems where appropriate rules are very hard to know in advance, as in expert systems and robotics.

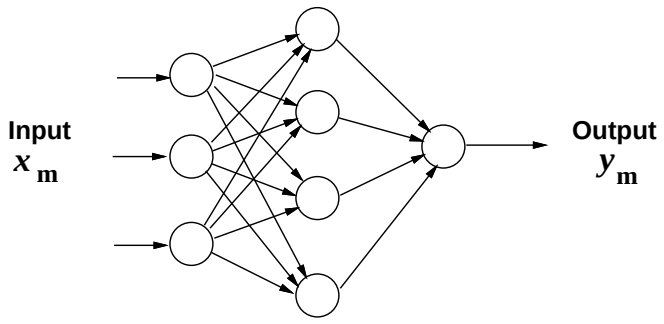

Figure 1.1: General neural network architecture

The basic architecture of a feed forward neural network is shown in fig.1.1. There is an input layer, a hidden layer and an output layer. We define the training set $\{x_m, y_m\}_{m=1}^{M}$ as the set of inputs x_m and the corresponding desired or optimal outputs y_m used to train the network.

Here, we define the concept of learning in a neural network. *The* learning problem *is to construct a neural network by using a training set* $\{x_m, y_m\}_{m=1}^{M}$ *so that the NN expresses the best approximation* $f_0(x)$ *to a desired function* $f(x)$ *under some given learning criterion.*

As is evident from fig.1.2, the learning problem becomes a problem of realizing an approximate function f_0 as close to the function f, where f is the function mapping the input and corresponding output of the training data, i.e. $f(x_m) = y_m$. This can also be thought of as the task of obtaining a learning operator X which learns the function f_0 from the training set.

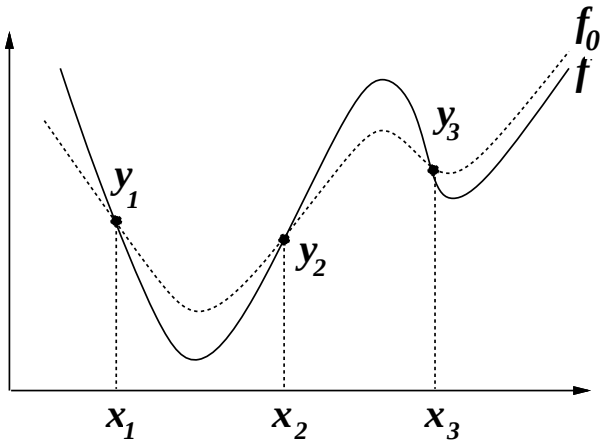

Figure 1.2: Training data, representative function $f(x_m) = y_m$ and learned function f_0.

1.2 Least squares estimation

Since the basic problem of learning, as shown in the previous section, is analogous to the function approximation problem, we will look at some existing least squares methods for function approximation in this section.

Perhaps the richest and the most exciting area of application of the least squares is the area of *statistical estimation*. It appears in virtually all the branches of science, engineering, and social science for the analysis of experimental data, for control of systems subject to random disturbances or for decision making based on incomplete information.

Least squares is, of course, only one of the several established approaches to estimation theory, the main alternatives being maximum likelihoods and Bayesian techniques. These later techniques usually require a complete statistical description of the problem variables in terms of joint probability distribution functions, whereas least squares requires only means, variances and covariances.

Basically three main types of linear estimation problems are explained below. The three problems are quite similar, differing only slightly in the choice of the optimality criterion and in the underlying statistical assumptions.

1.2.1 The least squares estimate

Suppose a quantity of data consisting of m real numbers has been collected and arranged as the m components of a vector y. The vector y is assumed to be a linear function of a few unknown parameters, which can be arranged as the components of n dimensional vector β, where $n < m$.

So we can write,

$$y = W\beta. \tag{1.1}$$

The $m \times n$ matrix W is assumed to be known experimentally or through apriori knowledge. The data vector y is known. The problem is to determine the vector β. Since $n < m$, however, it is generally not possible to find a vector β exactly satisfying eq.(1.1). A useful alternative is determining the value of β which best approximates a solution in the sense of minimizing the norm

$$\|y - W\beta\|. \tag{1.2}$$

If this norm is taken as the standard m dimensional space Euclidean norm, then, this approach leads to a simple least squares estimate.

Theorem 1.1 (Least Squares Estimate) *Suppose y is an m dimensional vector and W a $m \times n$ matrix with linearly independent columns. Then, there is a unique n vector $\hat{\beta}$ which minimizes eq.(1.2) over all β (the norm taken as the Euclidean m-space norm). The estimate $\hat{\beta}$ is given as*

$$\hat{\beta} = (W^T W)^{-1} W^T y. \tag{1.3}$$

When the matrix $W^T W$ does not have an inverse (is singular), then, we use the concept of pseudo inverse described in Chapter 4.

1.2.2 Minimum-Variance Unbiased Estimate (Gauss-Markov Estimate)

Now we consider an experiment that leads to an m-dimensional data vector y of the form

$$y = W\beta + \epsilon. \tag{1.4}$$

In this model, W is a known matrix, β is an n-dimensional unknown vector of parameters, and ϵ is an m-dimensional random vector with zero mean, i.e. $E_\epsilon(\epsilon) = 0$ and covariance $E_\epsilon(\epsilon\epsilon^T) = Q$ which we assume to be positive definite. Here, E_ϵ refers to the expectation taken over the entire set of ϵ. In this problem, we seek a linear estimate of β of the form

$$\hat{\beta} = Ky, \tag{1.5}$$

where K is a constant $n \times m$ matrix.
A natural criterion for the optimality of the estimation scheme is minimization of the norm of the error

$$\underset{\epsilon}{E}[\|\hat{\beta} - \beta\|^2]. \tag{1.6}$$

This can be expressed as

$$\begin{aligned} \underset{\epsilon}{E}[\|\hat{\beta}-\beta\|^2] &= \underset{\epsilon}{E}[\|Ky-\beta\|^2] = \underset{\epsilon}{E}[\|KW\beta+K\epsilon-\beta\|^2] \\ &= [\|KW\beta-\beta\|^2]+tr(KQK'). \end{aligned} \tag{1.7}$$

A practical estimation scheme for the above function cannot be derived since it is a function of the unknown variable or parameter β. However, we observe that if $KW = I$ or an identity matrix, then, the norm is independent of β. This suggests an alternative problem, which can be defined as
Find the estimate $\hat{\beta} = Ky$ minimizing

$$E[\|\hat{\beta}-\beta\|^2], \tag{1.8}$$

while satisfying

$$KW = I. \tag{1.9}$$

The solution to this problem is independent of β. Now, we give the classical theorem summarising the analysis and the resultant solution for this class of problems.

Theorem 1.2 (Gauss-Markov) *The linear minimum-variance unbiased estimate of β is*

$$\hat{\beta} = (W^TQ^{-1}W)^{-1}W^TQ^{-1}y, \tag{1.10}$$

with corresponding error covariance

$$\underset{\epsilon}{E}[(\hat{\beta}-\beta)(\hat{\beta}-\beta)'] = W^TQ^{-1}W^{-1}. \tag{1.11}$$

1.2.3 Minimum-Variance Estimate

In the previous two problems, vector β was considered to be a vector of unknown parameters. They could presumably have assumed any value from -∞ to +∞. In many situations, we do have prior information and then, we can consider the vector β as a random variable with known mean and covariances. This *a priori* statistical information can be exploited to produce an estimate of lower error variance than the minimum-variance unbiased estimate.

In view of this observation, we can consider the estimation of β from the measurements of the form

$$y = W\beta + \epsilon. \tag{1.12}$$

But in this case, both ϵ and β are random vectors. The criterion for optimization is the minimization of the following error term.

$$\underset{\epsilon,\beta}{E}[\|\hat{\beta}-\beta\|^2]. \tag{1.13}$$

where $E_{\epsilon,\beta}$ refers to the expectation over all ϵ and β and

$$\underset{\epsilon,\beta}{E}(\epsilon\epsilon^T) = Q, \tag{1.14}$$

$$\underset{\epsilon,\beta}{E}(\beta\beta^T) = R, \tag{1.15}$$

$$\underset{\epsilon,\beta}{E}(\epsilon\beta^T) = 0 \tag{1.16}$$

are known. Now, we state the analysis and the result for the problems of this class.

Theorem 1.3 (Minimum-Variance Estimate) *We assume that R and Q are positive definite matrices of appropriate size and that WRW^T+Q is non-singular. Then, the linear estimate $\hat{\beta}$ of β minimizing eq.(1.13) is*

$$\hat{\beta} = RW^T(WRW^T + Q)^{-1}y, \tag{1.17}$$

with error covariance

$$\underset{\epsilon,\beta}{E}[(\hat{\beta}-\beta)(\hat{\beta}-\beta)^T] = R - RW^T(WRW^T + Q)^{-1}WR. \tag{1.18}$$

A significant difference between the estimation problem of this sub-section and the previous is that the number of observations, m, need not be at least as large as the number of unknowns, n. Physically, the reason for this is that it is possible to form a meaningful estimate even with no measurement. Every new measurement simply provides additional information which may modify our original estimate.

1.2.4 Shortcomings of the least squares estimation model in NN learning

We have seen three least squares estimation problems and their solutions, which are widely used in the field of statistical estimation. However, one thing to be noted in these models are that they were defined for problems from a finite dimensional space to another finite dimensional space, with the original space being usually smaller than the sampled space.

In real world application problems in the learning of neural networks, we will see that the *original function space is usually larger than the sampled space; usually the original function space is infinite dimensional.* Under these circumstances, the linear estimation approach described here cannot be applied to the learning problem directly.

However, as will be described later in Section 2.4, we will use a modified estimation criterion. An analogy can be easily established, according to which the 'least squares estimate' of Section 1.2.1 corresponds to a special case of Memorization criterion in Section 2.4.1 and the 'Gauss Markov estimate'of Section 1.2.2 reduces to a special case of Projection criterion with noise in Section 2.4. On the other hand, the 'Minimum variance estimate' of Section 1.2.3 turns out to be the special case of the 'Wiener criterion'in Section 2.4.2.

In the next section, we will list the objectives of this research work.

1.3 The What & Why of this research

In the previous section, we discussed the background and framework of the present research work. As mentioned in Section 1.1, the learning is carried out by evaluating the optimal *learning operator* X under some optimization criterion, so that the learned function f_0 is as close to the original functions as possible. A lot of research work has been carried out [6],[8], [10], [11], [14], [24] in regard to obtaining the optimal learning operator or reconstruction operator, as it is called in the field of image reconstruction from the given set of training data. Solution for deriving the optimum learning operator X from a given training set, for different optimization criteria and is provided in Section 2.4.

However, a theoritical basis for the selection of an optimal training set that maximizes generalization ability of the neural network has not been established yet. Moreover, when looking at selecting training data, an obivious next step would be to devise a method of incorporating this newly selected training data into the learning process in real time using an incremental learning approach. Recomputing the learning operator from scratch every time a new training data or information becomes available obviously involves a lot of overhead and is inefficient in the sense that we are redoing a part of the work already done. Moreover, in real time applications, it is imperative to get fast results.

Therefore, with the above mentioned objectives in mind, in this dissertation we will look into the problem of devising an efficient method of incorporating new training data. Then, we will look at a sampling scheme to ensure better generalization in neural networks.

Summarising the major objectives, we have two main points

1. **Devise a method for efficiently incorporating a new training data and compute the new learning operator incrementally**. The objectives behind this aim are
    - It will be possible to compute the learning operator incrementally, hence, reducing computational cost.
    - It will possible to incorporate new training data in real time as and when they are available.
    - Intermediate results of training can be output without waiting for all the training data to arrive. This can be used to adapt or modify the sampling scheme based on the newly available information, also referred to as active sampling.
2. **Find the next optimal sample point x_{m+1} which minimizes the generalization error under some given learning criterion**. The objective behind this aim is
    - To ensure optimal training set which maximizes the generalizing ability of the neural network and hence, devise a theorical basis for the sampling schemes.

1.4 Organization of the dissertation

The chapters of the dissertation are organized as follows. Chapter 2 introduces the concept of learning in neural networks as an inverse problem. The corrspondence between the conventionally used NN model and the functional analytic approach is established. Different optimization criteria for learning and their solutions and characteristics are provided. In Chapter 3, the concept of incremental learning is formalised. Chapter 4 introduces the Moore-Penrose generalized inverse and an incremental method of computation for a special type of matrix is explained. This result is applied in Chapter 5 to devise an efficient method for incorporating training data incrementally. An analysis of the computational complexity of the learning operator is carried out and a training set selection scheme or sampling for reducing computational complexity is derived. Chapter 6 deals with the problem of data sampling for maximizing generalizing ability of the network. Although, the original problem of obtaining next best sample point has not been solved, the sampling scheme derived in Chapter 5 is compared with the aim of maximimizing generalizing ability. A computer simulation for the verification of the derived results follows. Conclusions and evaluations of the research are given in Chapter 7.

Chapter 2

Functional analytic approach to NN learning

2.1 Introduction

Conventionally, learning in neural networks is carried out by changing the weights in the connection of the network through some heuristic and hence, realising the desired function f. However, here we will use a functional analytic method for analysing the learning concept and give a correspondence between the two approaches. Functional analytic approach to learning in Neural Networks(NNs) has been proved to improve generalizing abilty because of their focus on the original function space rather than the sampled space [15], [16], [17]. The concept will be developed in steps in Section 2.2 and the correspondence to the conventional model will be established in Section 2.3. The closeness or the effectivenes of the learning has to be defined or measured in some way. To this effect, we have a number of optimization criteria for the learning process, which will be stated in Section 2.4. Their characteristics and solutions are elaborated on.

2.2 Learning as an inverse problem

Let us begin by considering a multi-layer feedforward network whose number of input and output units are L and 1, respectively. Let $\xi^{(1)}, \xi^{(2)}, ..., \xi^{(L)}$ represent the inputs to the input layer neurons. Let these inputs constitute the elements of an L-dimensional vector x, referred to as the 'input vector'. An input-output relation of the n-th hidden unit is denoted by $u_n(x)$ and called the basis function. Let w_n be the corresponding weight connecting the n-th hidden unit and the output unit. In this case, the network can be considered as a real valued function f_0 of L variables. The input-output relationship is shown in fig.2.1.

The *learning problem* is to construct a neural network by using a training set so that the NN

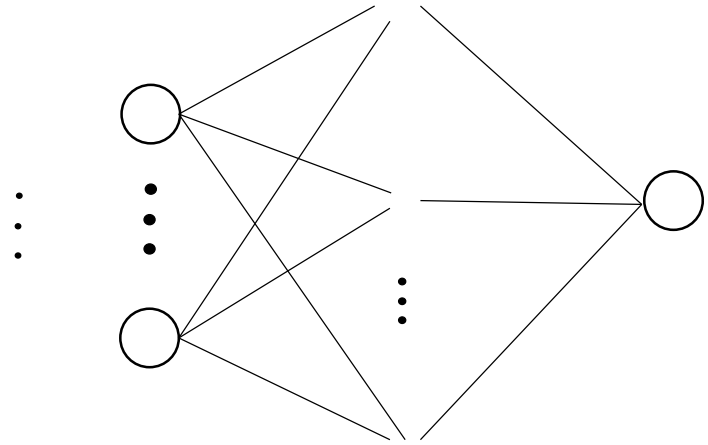

Figure 2.1: NN as a real valued function

expresses the best approximation $f_0(x)$ to a desired function $f(x)$ under some given learning criterion. Some of the notations used here are as defined below:

$\{x_m : 1 \leq m \leq M\}$**:** A *training set* given as a set of M input vectors.

$\{y_m : 1 \leq m \leq M\}$**:** The corresponding desired output values, where $y_m = f(x_m)$.

y**:** Vector consisting of elements $\{y_m : 1 \leq m \leq M\ \}$.

$\{x_m, y_m\}_{m=1}^{M}$**:** Training data.

The learning problem of neural networks can now be discussed from the point of view of function approximation. In other words, the learning problem can be regarded as the same problem as that of the general sampling theorem [12] and image restoration [11], [14]. In this context, x_m and y_m can be called the m-th *sample point* and the corresponding *sample value* of the desired function f, respectively.

Let H be the set of all functions f to be aproximated by the neural networks. Assume that H is a Hilbert space with a *reproducing kernel* $K(x, x')$. Let D be the domain of the basis functions, which is a subset of the L-dimensional Euclidean space R^L. The reproducing kernel $K(x, x')$ is a bivariate function defined on $D \times D$ which satisfies the following two conditions:

1. For any fixed x' in D, $K(x, x')$ is a function in H.

2. For any function f in H and for any x' in D, it holds that

$$(f(x), K(x, x')) = f(x'), \tag{2.1}$$

where the left hand side of eq.(2.1) denotes the inner product in H.

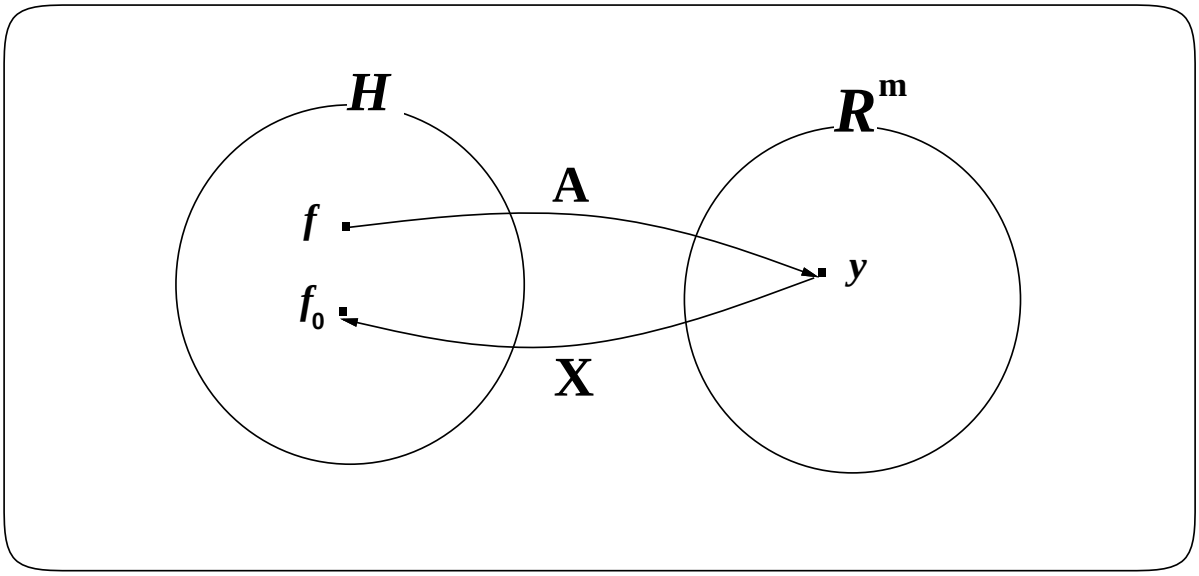

Figure 2.2: NN learning as an inverse problem

In the theory of Hilbert space, arguments are developed by regarding a function as a point in that space. Thus, things such as 'value of a function at a point' cannot be discussed under the general framework of Hilbert space. However, if the Hilbert space has a reproducing kernel, then it is possible to deal with the value of a function at a point. Indeed, if we define function $\psi_m(x)$ as

$$\psi_m(x) = K(x, x_m) : 1 \leq m \leq M, \tag{2.2}$$

then, the value of f at a sample point x_m is expressed in Hilbert space language as the inner product of f and ψ_m as

$$f(x_m) = (f, \psi_m). \tag{2.3}$$

Once the training set $\{x_m : 1 \leq m \leq M\}$ is fixed, the vector y is uniquely determined from f. So, we can introduce an operator A which transforms f to y:

$$y = Af. \tag{2.4}$$

The operator A, called the *sampling operator*, becomes a linear operator even when we are concerned with nonlinear neural networks. It is expressed by using the Schatten product as

$$A = \sum_{m=1}^{M} e_m \otimes \overline{\psi_m}, \tag{2.5}$$

where $\{e_m : 1 \leq m \leq M\}$ is the so-called natural basis [1]. The Schatten product denoted by $(. \otimes \bar{.})$ is defined by

[1] The vector e_m is the M-dimensional vector consisting of zero elements except the element m equal to 1.

$$(e_m \otimes \overline{\psi_m}) f = (f, \psi_m) e_m. \tag{2.6}$$

Hence, the learning problem is the problem of obtaining an estimate , say f_0, to f from y in the model (See fig.2.2). This can be considered as an *inverse problem* [16], [17] equivalent to obtaining an operator X which provides f_0 from y:

$$f_0 = Xy. \tag{2.7}$$

The operator X is called the *learning operator*. It is also referred to elsewhere in other fields by different names like *restoration operator, reconstruction operator, approximation operator*, etc. This operator X can be optimized based on different criteria. These criteria, known as the *learning criteria*, are of different types like the memorization criterion, the Wiener criterion, the projection criterion, etc. [17].

2.3 Analogy between conventional model and the functional analytic approach

2.3.1 Conventional neural network and learning

As mentioned before, the conventional neural networks learns through modification of weights in the connections. A simple model of an L-input, 1-output and N-hidden units neural network is shown in fig.2.3.

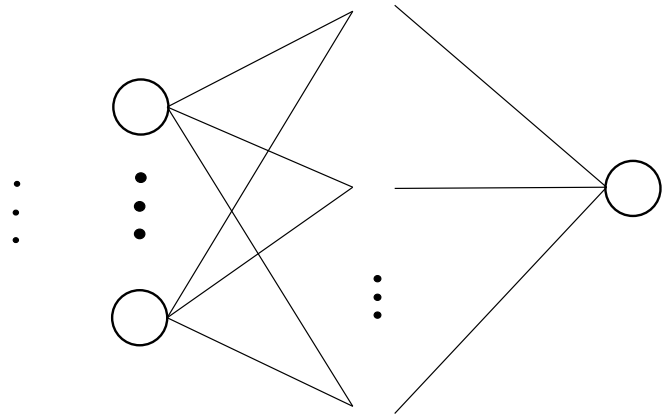

Figure 2.3: Conventionally used NN architecture

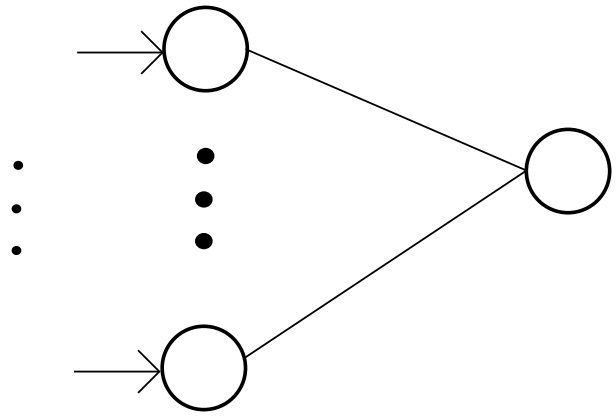

Figure 2.4: nth hidden unit of the neural network

In general, neural networks of the type shown in fig.2.3 is used widely. This network learns by changing the weight sets $\{w_{nl}\}_{l=1}^{L}{}_{n=1}^{N}$ and $\{w_n\}_{n=1}^{N}$. The summation of the inputs, each weighted by it's corresponding connecting weight w_{nl}, acts as the input to the hidden layers. This value is then processed using the activation function of the hidden unit, which is usually a sigmoidal function.

Based on the weights w_{nl}, we can assume that the hidden layer is in fact realising an input-output or basis function, denoted by

$$u_n(x) = \sigma(\sum_{l=1}^{L} w_{nl}\xi_l). \tag{2.8}$$

This relationship is graphically depicted in fig.2.4. This implies that by changing the weight set $\{w_{nl}\}_{l=1}^{L}{}_{n=1}^{N}$, we also change the basis function u_n of the hidden unit (fig.2.4). Hence, changing the weights $\{w_{nl}\}_{l=1}^{L}{}_{n=1}^{N}$ using learning algorithms in the above kind of networks corresponds to fixing the input-output relationship of the hidden units.

By adjusting both the weight sets $\{w_{nl}\}_{l=1}^{L}{}_{n=1}^{N}$ and $\{w_n\}_{n=1}^{N}$ using the training data $\{x_m, y_m\}_{m=1}^{M}$ based on some fixed learning algorithm like the back-propagation algorithm etc., we get the realization of the function f_0 such that $f_0(x_m) = y_m$.

Once again, summarising the conventional learning approach,

Using a fixed activation function (usually a sigmoidal function), adjust the weights w_{nl} and w_n (refer fig.2.3) using a weight adjustment algorithm to realize a particular function f. As an input to the training stage, the training data $\{x_m, y_m\}_{m=1}^{M}$ is used so as to realize a function f_0 such that $f_0(x_m) = y_m$ ($m = 1$ to M). There is no criterion specifying the outputs outside the sample points.

In the next subsection, we will describe the learning method used in this dissertation and establish an analogy between the two learning methods.

2.3.2 Learning in neural networks as used in the dissertation

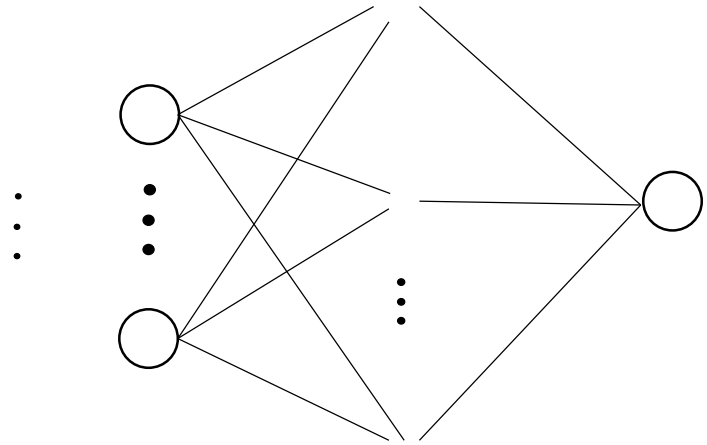

Figure 2.5: NN architecture used in this dissertation

Now, we will describe the neural network model being used in this dissertation. Refer to fig.2.5 for the architecture. As is evident from the fig.2.5, there are no weights in the connections between the input layer and the hidden layer. So, corresponding to the problem of changing the weights $\{w_{nl}\}_{l=1}^{L}{}_{n=1}^{N}$ in the conventional NN model shown in fig.2.3, we have the problem of selecting the basis functions u_n itself (refer eq.(2.8)). This ensures that the activation function need not necessarily be a sigmoidal activation function. The method of selection of the basis functions themselves will be dealt with later in this subsection.

Next, we explain the process of learning in NNs and the problem of actual realization by using a functional analytical method. Refer to fig.2.6 for the model. We divide our model into two parts, one part involves the training problem looked at from the mathematical point of view. The other part involves realization of this mathematically computed learning through weights in the conventional network model. Here, if we follow the figure in order of the operations being carried out, the sampling operator A, which is a function of the training set $\{x_m\}_{m=1}^{M}$, provides the sample value vector y from the original function f.

Now, the learning problem, looked at from the mathematical point of view involves computing an optimal learning operator X (refer to fig.2.6).

The other part involving realisation in a conventional neural network can be done by using the learning operator X obtained in the first part and using the following lemma [22].

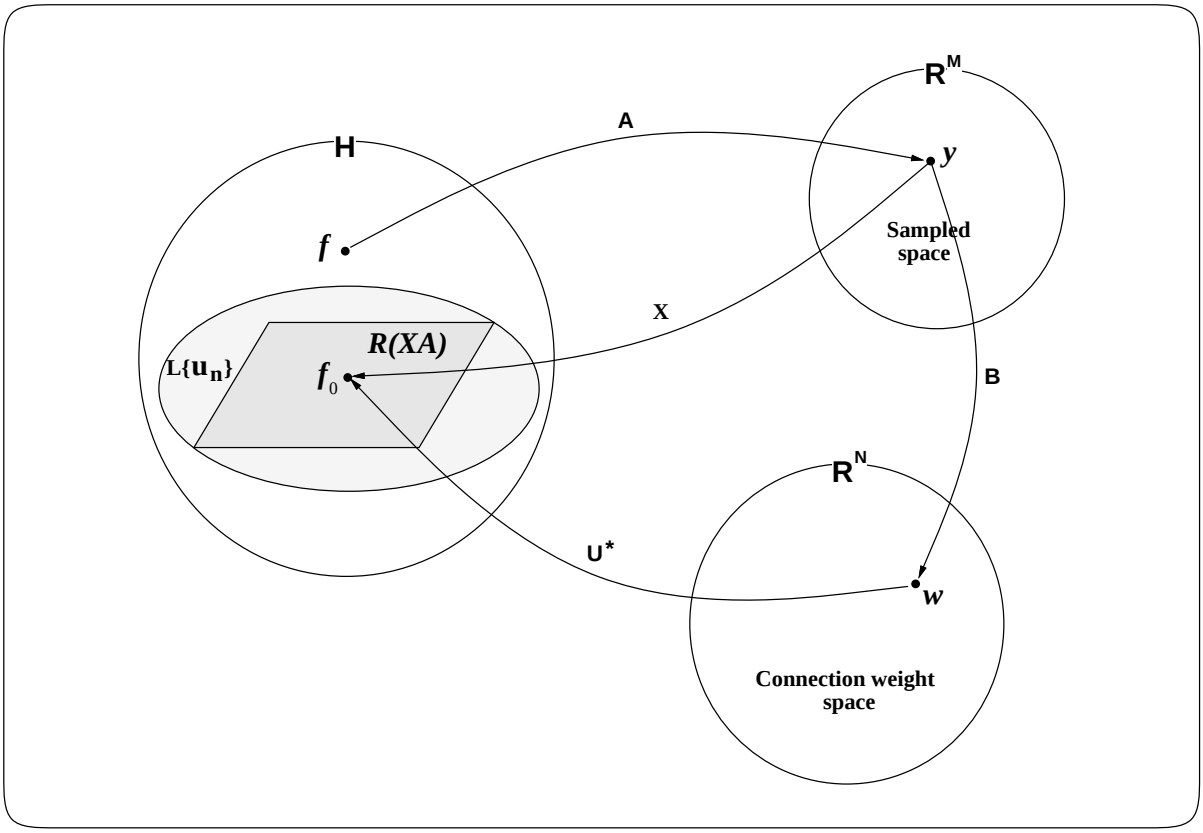

Figure 2.6: Functional analytic approach to NN learning.

Let U be defined as,

$$U \equiv \sum_{n=1}^{N} (e_n \otimes \overline{u_n}), \tag{2.9}$$

where u_n is the basis function determining the input-output relationship of the n-th hidden unit.

Lemma 2.1 *The optimally generalizing neural network based on the learning operator X can be realized practically by determining the number of hidden units N, the basis functions of the hidden units $\{u_n\}_{n=1}^N$ and the connection weights $\{w_n\}_{n=1}^N$ (refer fig.2.5) as follows:*

1. *Fix N such that*

$$N \geq dim(\mathcal{R}(XA)) = rank(XA). \tag{2.10}$$

2. *Fix $\{u_n\}_{n=1}^N$ such that*

$$L(\{u_n\}_{n=1}^N) \supset \mathcal{R}(XA). \tag{2.11}$$

3. *Compute $\{w_n\}_{n=1}^N$ by using the formula*

$$w_n = (VXy, e_n) \ \ n=1,\ldots,N, \tag{2.12}$$

where V satisfies the condition

$$U^*V = P_{L(\{u_n\}_{n=1}^N)}. \tag{2.13}$$

Hence, using this lemma, it is possible to realize the neural network through physical weights in one step after the learning operator X has been mathematically determined. Looking at the correspondence between the conventional NN model of Section 1.1.1 and the functional analytic model used in this dissertation, we see that

- Determining the weights $\{w_{nl}\}_{n=1\,l=1}^{N\quad L}$ in fig.2.3 corresponds to fixing the basis functions $\{u_n\}_{n=1}^N$ in our model. This also means that at this stage, the transformation U is defined using eq.(2.9).

- Determining the weights $\{w_n\}_{n=1}^N$ in fig.2.3 corresponds to determining the operator B in fig.2.6 which maps the sample values vector y to the weights.

Note: In the conventional BP approach the two sets of weights $\{w_{nl}\}_{n=1\,l=1}^{N\quad L}$ and $\{w_n\}_{n=1}^N$ are altered based on the algorithm without any apparent difference between the two sets. However, using the model used in this dissertation, we can clearly identify the difference between the two weights. The weight set $\{w_{nl}\}_{l=1\,n=1}^{L\quad N}$ corresponds to determining the range of the reproducible or learnable functions based on the *sampling scheme* alone; which corresponds to determining the basis functions $\{u_n\}_{n=1}^N$ in our model. Therefore, this uses only the *training set* and not the corresponding *sample values*. Evaluation of the weight set $\{w_n\}_{n=1}^N$ corresponds to fixing the learned function f_0 within the range $\mathcal{R}(XA)$, which actually depends on the sample values y_m.

2.4 Generalizing ability and optimization criteria

Multilayer feedforward neural networks are essentially approximators, mapping from a finite dimensional input space to an output space. One of the most desirable characteristics of such networks is their ability to learn from examples and to generalize from the training set to similar data not contained in the training examples.

Definition 2.1 *Generalizing abilty of a neural network is broadly defined as the ability of the NN to provide correct outputs, not only for the trained sample pairs $\{x_m, y_m\}_{m=1}^M$, but also regarding the untrained part of the desired function.*

The generalizing ability of the network basically depends on the optimization criterion chosen to measure the closeness between the learned function and the original function. For example, if the criterion checks for the closeness in the sampled space alone and not in the original function space, then, we cannot say anything about the novel samples i.e., points outside the

training set. Hence, the generalizing ability is not gauranteed. On the other hand, closeness in the original function space can also have different meanings like closeness with respect to a particular function or in an averaged sense etc;

As mentioned in Section 2.2, a variety of criteria have been used to measure the closeness between the original desired function and the learned function. In practice, a suitable criterion should be chosen based on the requirements of the problem. Examples of some criteria and their significance, shortcomings and interpretations have been provided in the following sub-sections.

2.4.1 Memorization criterion

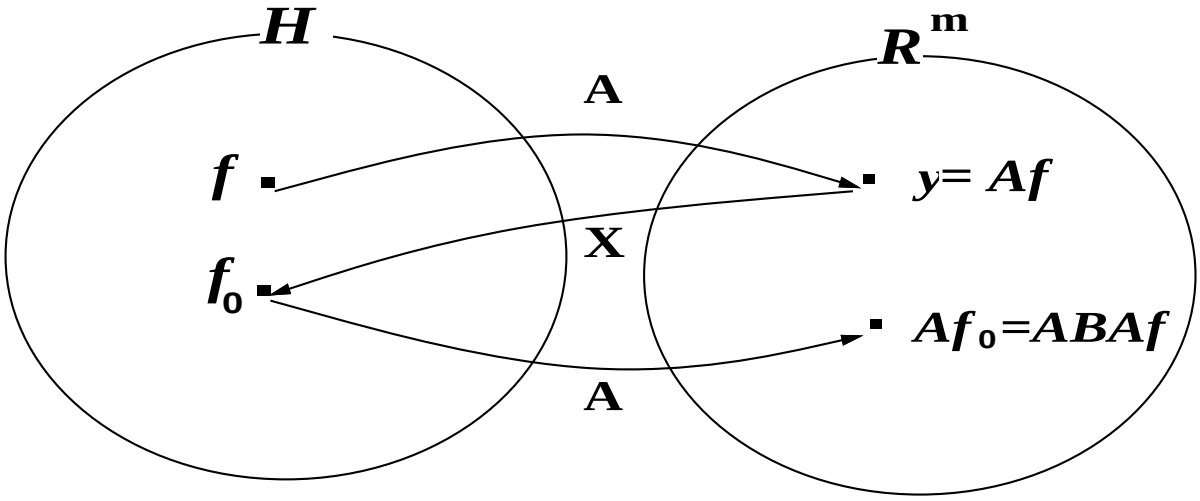

Figure 2.7: NN learning as an inverse problem

This is the criterion that is used in the popular Back Propagation learning algorithm in the field of neural networks. The formal definition of the criterion is as given below.

Definition 2.2 (Memorization criterionJ_M) *For all $y = Af$, minimize the following functional with respect to X:*

$$J_M[X] = \sum_{m=1}^{M} \{f_0(x_m) - y_m\}^2 = \|AXy - y\|^2, \tag{2.14}$$

where, $\|.\|$ is the norm in the sampled space R^M.

The graphical representation of the model is given in fig.2.7. This learning criterion J_M minimizes the error, in other words, asks for correct responses for the training inputs, but does not specify anyhting about the novel data or untrained data. Therefore, any algorithm based on J_M does not guarantee the generalization ability of the neural network in general. In this sense, we call J_M learning memorization learning. The suffix 'M' of J_M refers to 'memorization'.

Lemma 2.2 *[22] An operator X is a proper memorization learning if and only if X is a 1-inverse of the sampling operator A. In other words, X satisfies the equation*

$$AXA = A. \tag{2.15}$$

The operator X may be linear or non-linear. As for BP algorithm, X becomes non-linear in general.

2.4.2 Wiener criterion

Definition 2.3 (Wiener criterion J_W) *Minimize the following functional with respect to X:*

$$J_W[X] = E_f \|XAf - f\|^2, \tag{2.16}$$

where $\|.\|$ is the norm in the original function space H and E_f is the average of f over the original function space.

This criterion is often used in the field of image reconstruction. The criterion aims at reducing the difference between the original function f and the function f_0 reconstructed by using the learning operator X. The thing to be noted is that this minimization is done in the original function space H and in an *averaged sense* with respect to all the possible functions to be approximated.This is evident from eq.(2.16) and fig.2.7.

For the ease of calculation, we will look at only the linear solutions to the minimization criterion. Hence, we will consider only linear learning operators X as the possible solutions. However, limiting the learning operator X to a linear one does not imply that the model can represent or realise only linear functions f_0. A linear learning operator X can be used to realise a non-linear function f_0 in our learning model.

The Wiener criterion can be transformed into a more useful form. Let R be the *correlation operator* of all the functions to be approximated and be defined as

$$R \equiv E_f(f \otimes \bar{f}). \tag{2.17}$$

Lemma 2.3 *[24] A necessary and sufficient condition for the learning operator X to satisfy the Wiener criterion is that the following equation is satisfied.*

$$XARA^* = RA^*. \tag{2.18}$$

The eq.(2.18) can be interpreted in the following way. All learning operators satisfying the Wiener criterion can reconstruct the functions in the range $\mathcal{R}(RA^*)$ completely from the sample value vector $y = Af$ (Refer fig.2.7).

Corollary 2.1 (General solution for Wiener criterion) *A general form of the Wiener learning X is given as*

$$X = RA^*U^\dagger + W(I - UU^\dagger), \tag{2.19}$$

where

$$U = ARA^*. \tag{2.20}$$

Here, W is any operator from R^M to H and $\dagger$ represents the Moore-Penrose generalized inverse.

Once the training set is fixed, A can be calculated using eq.(2.5), and hence, corresponding to this sampling operator, a learning operator X satisfying the Wiener criterion can be obtained using eqs.(2.19) and (2.20).

We will be using the Wiener learning criterion as an example for implementing our theory on incremental learning in the present work. However, the concept is not limited to using this criterion, in other words, any optimization criterion can be used to suit the problem specification and requirements.

2.4.3 Projection criterion

The memorization criterion J_M does not gaurantee the generalization ability of a neural network. The Wiener criterion J_W provides the best approximation in the averaged sense with respect to the ensemble $\{f\}$. It does not imply the best approximation to any individual f. It may be sufficient for some cases. However, there are situtations where the best approximation f_0 to an individual f is needed. It is often said that we cannot obtain such an f_0 because we do not know the original function f. However, this is not true. The projection criterion J_P provides an answer to the problem.

Definition 2.4 (Projection criterion J_P) *For all $f \in H$, determine X such that*

$$f_0 = XAf \tag{2.21}$$

becomes the best approximation to the original function f in a subspace $\mathcal{R}(A^*)$.

The learned function $f_0 = XAf$ is always contained within $\mathcal{R}(X)$. Hence, while evaluating the nearest or the best approximation to f, we have to select the answer from within this subspace. When we use a linear operator X for learning f, the range of X becomes a closed subspace of H. Hence, the closest approximation of f in $\mathcal{R}(X)$, as shown in fig.2.8, is a projection of f on $\mathcal{R}(X)$. The following lemma shows that we can obtain the orthogonal projection of f onto $\mathcal{R}(X)$ from y even when we do not know f.

Lemma 2.4 *[17] Let S be a closed subspace of H. For any function f of H, its orthogonal projection onto S is computable from $y = Af$ if and only if S is included in the subspace $\mathcal{R}(A^*)$.*

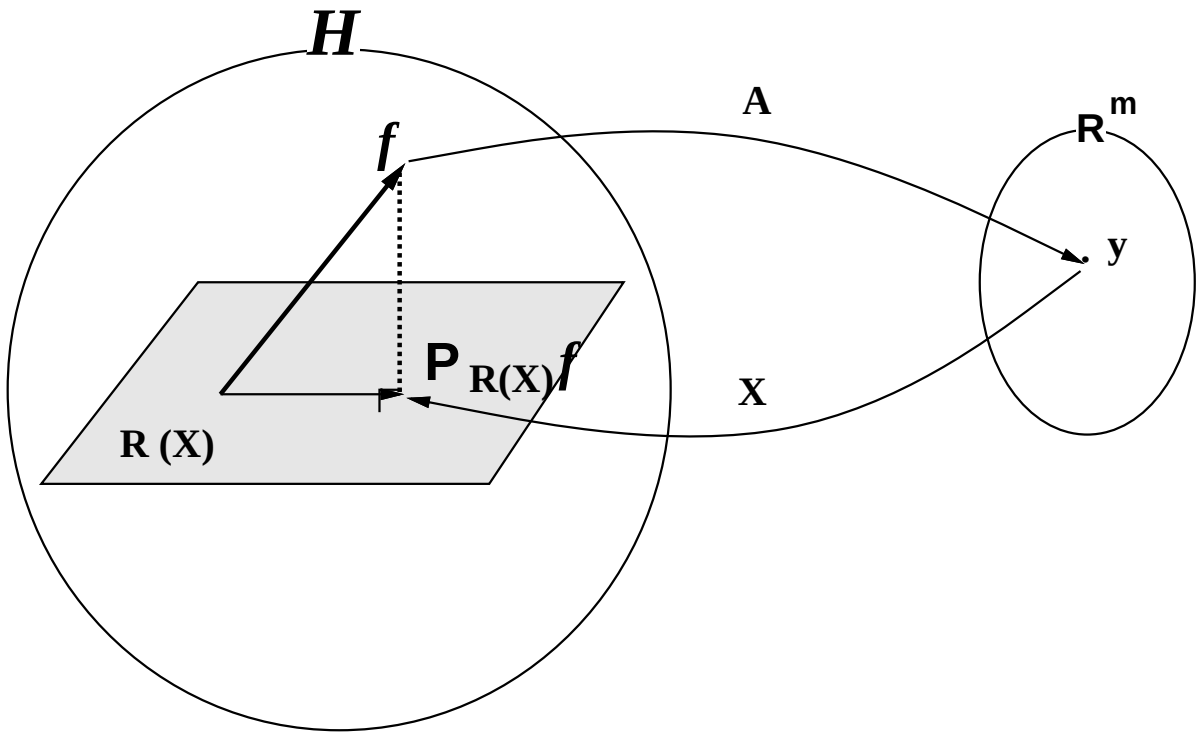

Figure 2.8: Projection learning

Based on this lemma, the condition for being able to reconstruct the function completely becomes

$$\mathcal{R}(X) \subseteq \mathcal{R}(A^*). \tag{2.22}$$

Since we are determining the orthogonal projection on the subspace $\mathcal{R}(X)$, it is better that this range be as large as possible for a better approximation. Hence, based on eq.(2.22), we will concentrate our attention on the projection on the range $\mathcal{R}(A^*)$. Hence, we can write the projection criterion as determining an X such that it satisfies

$$XAf = P_{R(A^*)}f : f \in H \tag{2.23}$$

It is for this reason that in Def.2.4, the learned function becomes the best approximation in the *subspace* $\mathcal{R}(A^*)$. This operator X is called a projection filter in the field of image reconstruction.

2.5 Conclusion

In this chapter, we started by explaining the neural network model and showed that learning in NNs can be represented as a function approximation problem. Furthermore, it was converted to an inverse problem using the functional analytical approach. Then, the correspondence with the conventional approach of training a NN by using connection weights was shown, hence,

proving that this approach can be used in the physical realization of the network. Finally, an alternative to the least squares estimation of Section 1.2 is provided in Section 2.4 to account for the real world problems involving infinite dimensional original function spaces.

Chapter 3

Incremental Learning

3.1 Introduction

So far, as described in Section 2.2, the learning problem has been discussed under the basic framework of finding an optimum learning or reconstruction operator from a given set of training data [1],[11],[17],[23],[24]. Under these circumstances, it was sufficient to construct a sampling operator A giving the sample value vector y and look at methods of finding an optimum reconstruction operator X, as discussed in Chapter 2. However, if we consider additional sample points, then, it is imperative to devise an efficient method to incorporate it into the learning scheme. To this effect, in Section 3.2, the concept of incremental learning is formalized. In Section 3.3, the Wiener criterion of Section 2.4.2 is represented in the context of incremental learning and some properties of incremental Wiener learning is proved.

3.2 Incremental learning

In a non-incremental or batch learning, sampling operator A is fixed based on the sampled training data using eqs.(2.2) and (2.5). With reference to fig.3.1, it implies that the sampling operator A is already fixed based on the given training data and hence, the first stage of operations, in other words, the selection of training data is no longer a variable. The problem, therefore, has been discussed under the objective of finding an optimum learning operator in order to reconstruct the function f_0, which is as close to the original function f under some optimization criterion.

However, there are problems in which one has the choice of selecting the training data. Basically, there are two methods of selecting training data. The first involves selecting the training set in one go, i.e., selecting the batch of training data all at once. The second involves incrementally selecting the training data one after the another. There has beeen a lot of experimental work done on the first approach [20]. However, a theoretical framework for selection of the

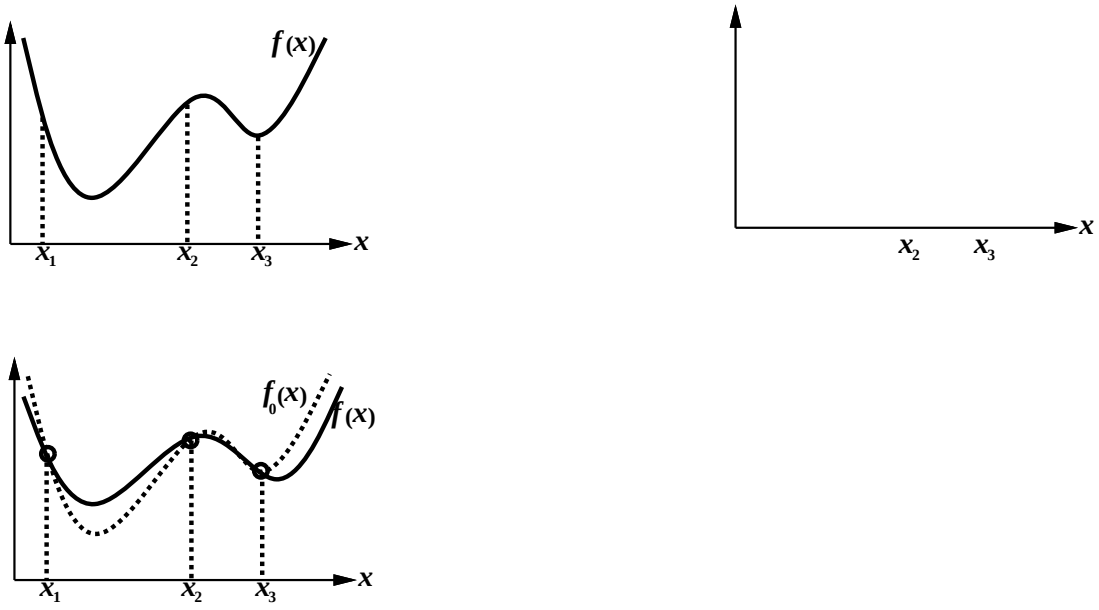

Figure 3.1: Neural network learning model. The two stages involved: (i) The selection of training data and hence, the sampling operator A, (ii) Evaluating the optimum learning operator X.

training data is yet to be established in either case. In this research work, we will try to look at the second approach and select the data incrementally. In such cases, it is necessary to formulate a basis for the selection of the optimal training data. Optimality may be measured in various ways. This topic of how to select optimal training set will be dealt with in Chapters 5 and 6.

When we look into the selection of training data, another very important process that has to be taken into account is the incorporation of the new training data. One obivious method of incorporating the new data is by recalculating the learning operator all over again every time a new data comes in. This, we will refer to as the 'non-incremental' method.

The problems involved in the non-incremental method is two-fold.

- Whenever a new training data is made available, this would involve recalculating the entire learning operator to incorporate this new information. This involves a lot of overhead.
- If the sampling itself is at a slow rate, then, we have to wait till all the training data samples have arrived before computing the learning operator.

With a view to solve these problems, we introduce a concept called the *incremental learning*. Incremental learning, here, refers to the successive or progressive learning with increase in the number of training data.

Since we will be dealing with a variable number of sample points and hence, a variable dimensional sampled space, we will use lower case letters to represent the dimension of the

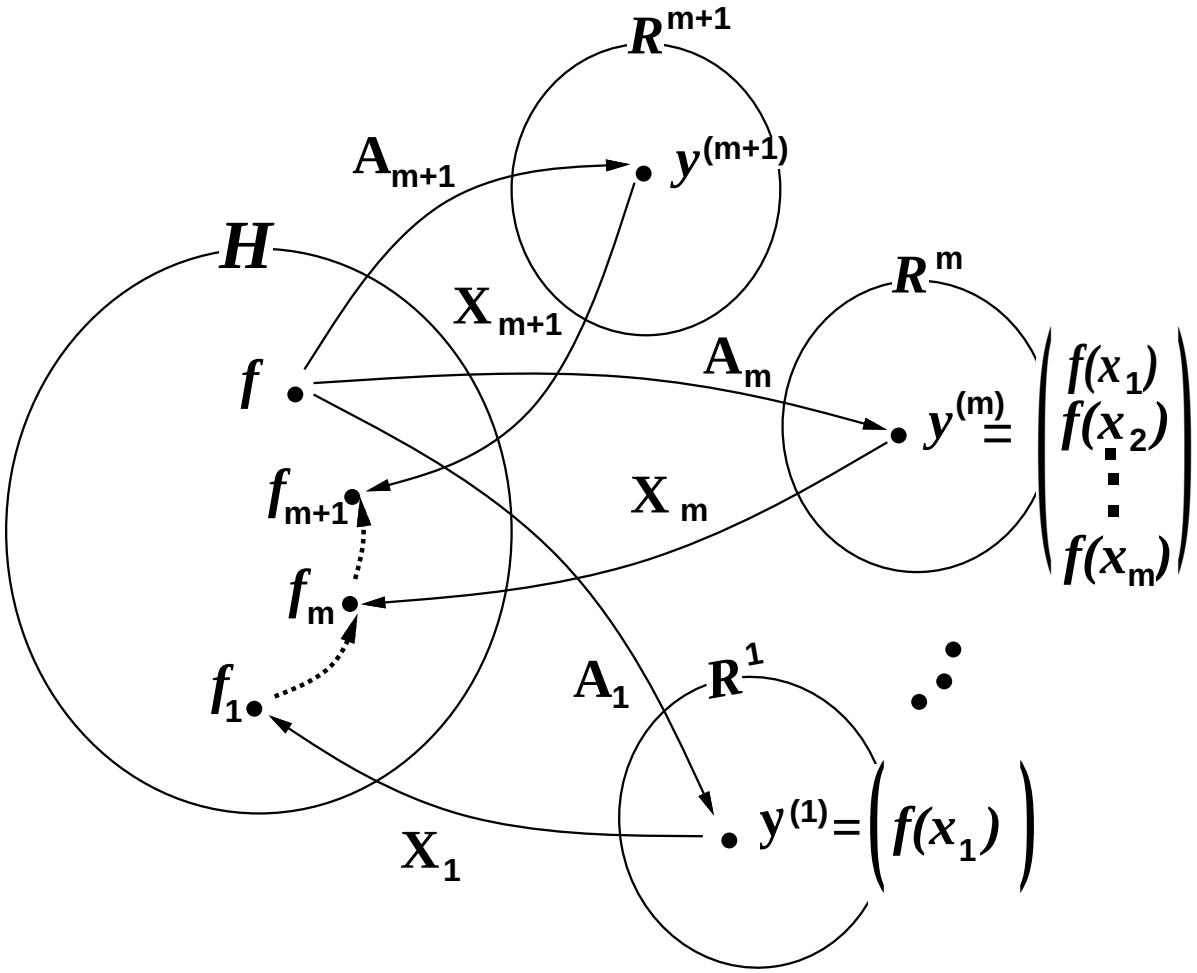

Figure 3.2: Incremental learning with additive training data

sampled space from now on. Let $\{e_n^{(m)} : 1 \leq n \leq m\}$ represent the natural basis of the sampled space R^m. Let A_m be the sampling operator from H to R^m while considering m sample points, defined as

$$A_m = \sum_{n=1}^{m} (e_n^{(m)} \otimes \overline{\psi_n}). \tag{3.1}$$

Let X_m be the corresponding learning operator for the m sample points derived using some optimization criterion. As shown in fig.3.2, with the increasing number of training data corresponding to increasing indexes of the sampling operator A_m, the sampled space R^m keeps expanding and the reconstructed function f_m approaches the original function f.

However, the point to be noted is that in the non-incremental method, the new learning operator X_{m+1} is computed directly from $\{x_n, y_n : 1 \leq n \leq m+1\}$ (Refer to fig.3.2). This involves a lot of overhead and every time a new data comes in, the entire learning operator has to be computed all over again. Suppose we are in the learning process, and every data point

means a change in the best fit. We will fall hopelessly behind by trying every time to recalculate the new learning operator from scratch. What is needed is a method which looks at the changes in the learning operator produced by the new sampling operator and the new sampled value. Hence, in incremental approach, the new learning operator is computed from the previously derived learning operator X_m using m sample points, the next sampling location [1] x_{m+1} and A_m. This can be represented as

$$X_{m+1} = F(X_m, x_{m+1}, A_m), \tag{3.2}$$

where F is some function.

To show the necessity for such an approach, we present an example.

Assume a situation in which we are fitting a straight line through points that come from the radar. At any moment, we can stop and compute the best fit line. When the next point misses that line, the estimate has to be changed.

This kind of a problem is analogous to the learning problem in the neural networks. We are trying to reconstruct the original function f from the given set of training data. As and when the new data arrives, the region of uncertainity grows smaller and the fit gets better and better.

Summarising the advantages of the *incremental approach using functional analytic method*,

1. The minimization of the criterion is done in the original function space and not in the sampled space, leading to better generalization ability. This has already been dealt with in detail in [15],[16],[17]. However, one should emphasize that this advantage is not due to the *incremental approach* but rather due to the use of *functional analytic method*.

2. It is not necessary to recalculate the entire learning operator when a new training data is made available. Instead, results of the previous stage can be used for the computation, leading to a saving in the computational cost.

3. It is possible to incorporate new training data as and when they are available. This is advantageous in cases when the sampling is slow or expensive.

4. Intermediate results such as in f_m are available. This information based on intermediate reconstruction may be used to improve sampling efficiency or optimize reconstruction by adaptive optimization criterions [7],[9].

In the following Section, we shall formulate the Wiener criterion from the incremental learning viewpoint.

[1] The next sampling location is derived based on some heuristic or optimization criterion with a view to maximize the generalization ability for a minimized training set. This will be dealt with in detail in Chapter 6.

3.3 Incremental Wiener learning

As an example of the optimization criterion for incremental learning, we shall use the Wiener criterion in this dissertation. As has been stated before, this choice is purely coincidental and the concept of incremental learning can also be extended to other optimization criteria.

The Wiener criterion, described in Section 2.4.2, can be represented in the context of incremental learning as follows.

Definition 3.1 *Minimize the following functional with respect to* X_{m+1}*:*

$$J_W[X_{m+1}] = E_f \|X_{m+1} A_{m+1} f - f\|^2, \tag{3.3}$$

where, $\|.\|$ *is the norm in the original function space* H *and* E_f *is the average of* f *over the original function space.*

Now, we present a result to formally prove that there is a better approximation or learning with additve training samples in cases without noise. To prove this, we present a couple of lemmas to begin with.

Lemma 3.1 *A necessary and sufficient condition for an operator* X *to satisfy Wiener criterion is*

$$XAf = f : f \in \mathcal{R}(RA^*). \tag{3.4}$$

Proof

($\Rightarrow$)

Let the operator X satisfy the Wiener criterion. Then, from eq.(2.18) it follows that

$$XARA^* = RA^*. \tag{3.5}$$

For all $f \in \mathcal{R}(RA^*)$, we have

$$XAf = XARA^* f. \tag{3.6}$$

Using eq.(3.5), we have

$$XAf = RA^* f = f. \tag{3.7}$$

($\Leftarrow$)

Assume for all $f \in \mathcal{R}(RA^*)$,

$$XAf = f. \tag{3.8}$$

Since this condition is for $f \in \mathcal{R}(RA^*)$, we can write that the following equation holds for all f', where $f = RA^* f'$.

$$XARA^* f' = RA^* f'. \tag{3.9}$$

Hence, the lemma is proved ■

So, we can now use this alternate condition for the future proofs.

It is known [23] that all learnable functions in case of Wiener learning belongs to the range $\mathcal{R}(R)$. The image of a subspace L due to an operator A is defined as

$$AL = \{Af | f \in L\}. \tag{3.10}$$

Lemma 3.2 *[23] For any closed subspaces L and any non-negative operator R, it holds that*

$$L + RL^{\perp} \supset \mathcal{R}(R). \tag{3.11}$$

Since $\mathcal{R}(RA^*)$ is finite dimensional, it is a closed subspace. Substituting $L = \mathcal{N}(A)$ in Lemma 3.2, we have

$$\mathcal{N}(A) + R\mathcal{N}(A)^{\perp} \supset \mathcal{R}(R). \tag{3.12}$$

Since $R\mathcal{N}(A)^{\perp} = \mathcal{R}(RA^*)$, it follows from eq.(3.12)that

$$\mathcal{N}(A) + \mathcal{R}(RA^*) \supset \mathcal{R}(R). \tag{3.13}$$

Lemma 3.3 *[23] It holds that*

$$\mathcal{R}(R) = \mathcal{R}(RA^*) + (\mathcal{R}(R) \cap \mathcal{N}(A)). \tag{3.14}$$

Lemma 3.3 shows that it is possible to express the original function space for Wiener learning $\mathcal{R}(R)$ as the sum of the subspaces $\mathcal{R}(RA^*)$ of Lemma 3.1 and $\mathcal{R}(R) \cap \mathcal{N}(A)$. Also, all the functions belonging to $\mathcal{R}(R) \cap \mathcal{N}(A)$ yields zero on being sampled by the sampling operator A. Hence, the reconstructed function also becomes zero for the original function belonging to $\mathcal{R}(R) \cap \mathcal{N}(A)$. Then we have

Lemma 3.4 *The necessary and sufficient condition for an operator X to satisfy Wiener criterion within subspace $\mathcal{R}(R)$ is that XA is a projection on $\mathcal{R}(RA^*)$ along $\mathcal{R}(R) \cap \mathcal{N}(A)$.*

By using the result of eq.(3.13), we can extend the subspace of the original function in Lemma 3.4 to $\mathcal{R}(RA^*) + \mathcal{N}(A)$. Hence we have

Lemma 3.5 *Wiener learning operator is a projection on $\mathcal{R}(RA^*)$ along $\mathcal{N}(A)$ within the subspace $(\mathcal{R}(RA^*) + \mathcal{N}(A)$.*

In other words, the Wiener learning operator can be called as a projection onto the subspace $\mathcal{R}(RA^*)$ along $\mathcal{R}(R) \cap \mathcal{N}(A)$ within the subspace $(\mathcal{R}(RA^*) + \mathcal{N}(A))$ which is a superset of $\mathcal{R}(R)$. However, we assume that the possible set of original functions defined by the Hilbert space H are already contained in the $\mathcal{R}(R)$ because of the correlation operators.

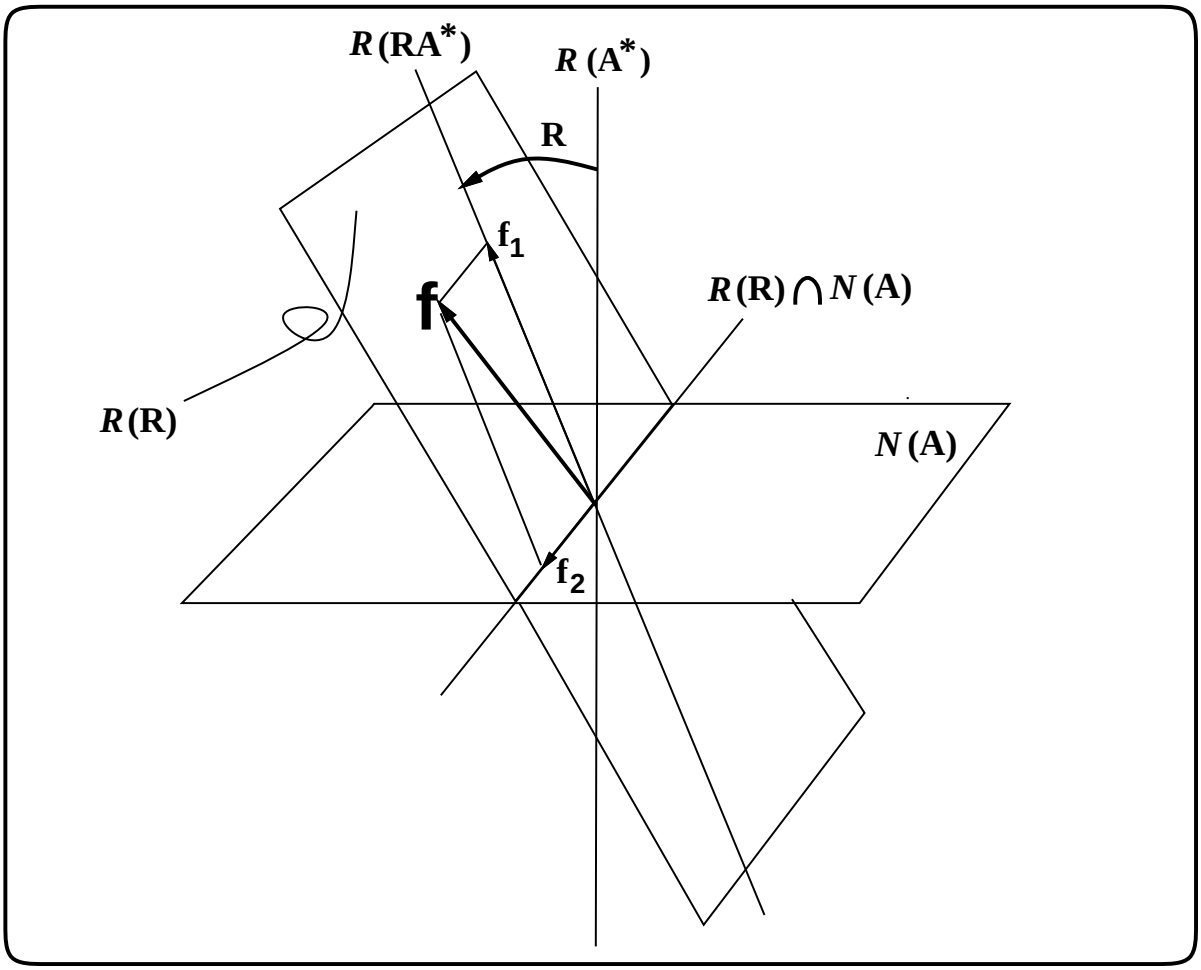

Figure 3.3: Geometrical interpretation of a Wiener learning. For an operator X satisfying the Wiener criterion, XA provides an oblique projection f_1 of $f \in \mathcal{R}(R)$ onto $\mathcal{R}(RA^*)$ along $\mathcal{R}(R) \cap \mathcal{N}(A)$.

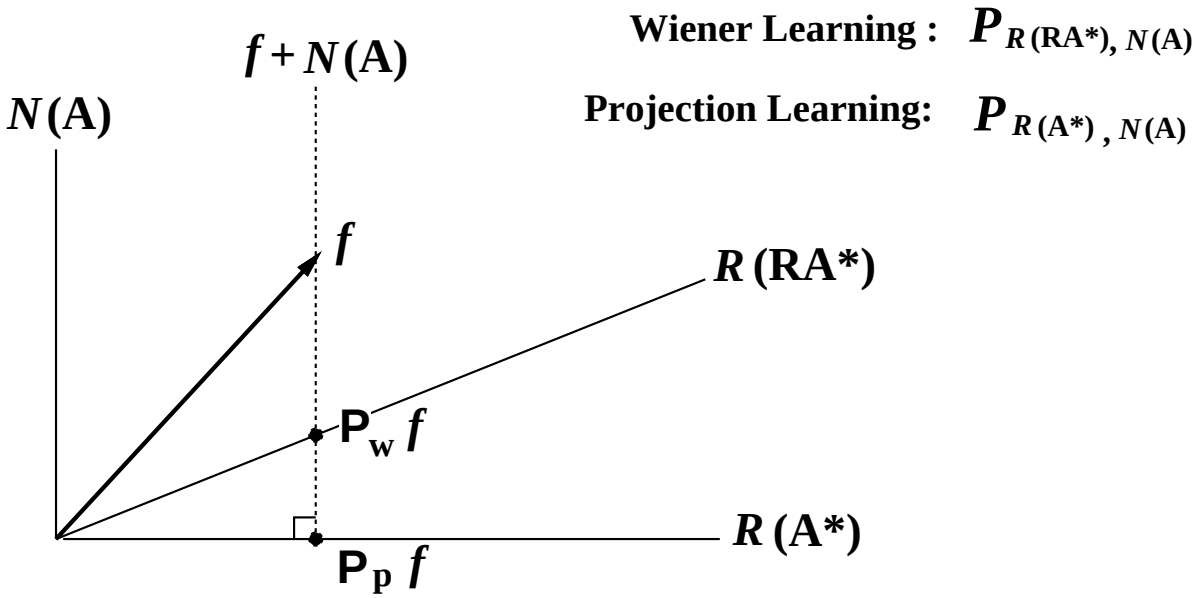

Figure 3.4: Relationship between the Wiener and Projection learning operators. Wiener learning is a projection on $\mathcal{R}(RA^*)$ whereas Projection learning is a projection on $\mathcal{R}(A^*)$, both along $\mathcal{N}(A)$.

Theorem 3.1 *Additive training samples ensure better approximation in Wiener learning; that is, it holds for any f in $\mathcal{R}(R)$ that*

$$\min_{X_{m+1}} \|X_{m+1}A_{m+1}f - f\|^2 \leq \min_{X_m} \|X_m A_m f - f\|^2. \tag{3.15}$$

Proof

Based on Lemma 3.5, we can prove an equivalent statement instead of eq.(3.15). The equivalent equation to be proved is stated as below.

$$\|P_{\mathcal{R}(RA^*_{m+1}),\mathcal{N}(A_{m+1})}f - f\|^2 \leq \|P_{\mathcal{R}(RA^*_m),\mathcal{N}(A_m)}f - f\|^2, \tag{3.16}$$

where $f \in \mathcal{R}(R)$.

We have, from the basic theory of subspaces and due to the non-negativeness of the operator R,

$$\mathcal{R}(RA^*_{m+1}) = \mathcal{R}(A_{m+1}RA^*_{m+1}). \tag{3.17}$$

Since the Wiener criterion involves the minimization of the error terms, it is enough to consider the worst case analysis for the left hand side term of eq.(3.16). From the analysis of the structure of the term $A_{m+1}RA^*_{m+1}$, it is easy to see that the worst case i.e. when the additional terms

are linearly dependent on the previous set, for the comparison of $A_{m+1}RA^*_{m+1}$ and $A_mRA^*_m$ is the case when

$$\mathcal{R}(A_{m+1}RA^*_{m+1}) = \mathcal{R}(A_mRA^*_m). \tag{3.18}$$

Using eqs.(3.17)and (3.18), it is easy to see that the worst case analysis yields the equality in eq.(3.16). This implies eq.(3.16). ■

As shown in fig. 3.2 and proved using Theorem3.1, with the increasing number of samples the learned function f_m, f_{m+1},...,etc; gets closer to the original function f with respect to some error measuring criterion.

3.4 Conclusion

In this chapter, the concept of incremental learning using functional analytic approach is formalised in order to devise a scheme for carrrying out the learning process incrementally. As an example of the learning optimization criteria, we have represented the Wiener criterion in the context of the incremental learning. This will be used later on in our scheme for incorporating new additions to the training data incrementally.

Chapter 4

Moore-Penrose Generalized Inverse

4.1 Introduction

In this chapter, we will provide a mathematical definition for the Moore-Penrose generalised inverse and provide some of its essential properties. Then, we will state a method for incremental computation of the Moore-Penrose generalised inverse of a matrix of special type. This result will be used in further chapters to implement our theory of incremental learning for Wiener criterion, which has already been formalised in the previous chapter.

In recent years, need has been felt in numerous areas of applied mathematics for some kind of partial inverse of a matrix that is singular or even rectangular . By a generalized inverse of a given matrix A, we shall mean a matrix X associated in some way with A such that

- It exists for a class of matrices larger than the class of non-singular matrices.
- Has some of the properties of the usual inverse.
- Reduces to the usual inverse when A is non-singular.

There are different types of generalized inverses and in general, they are not unique for a given matrix. In this chapter, we shall deal with a special type of generalized inverse called the 'Moore-Penrose generalised inverse', also referred to at places by the name psuedoinverse.

4.2 Mathematical definition and characteristic properties

A matrix has an inverse only if it is square, and even then only if it is non-singular, or in other words, if its columns (or rows) are linearly independent. However, even for singular and rectangular matrices, it is essential to compute an approximate inverse. To this effect, in 1955 Penrose showed that, for a finite matrix A (square or rectangular) of real or complex elements,

there is a unique matrix X satisfying the four equations

$$AXA = A, \tag{4.1}$$

$$XAX = X, \tag{4.2}$$

$$(AX)^* = AX, \tag{4.3}$$

$$(XA)^* = XA, \tag{4.4}$$

where A^* denotes the conjugate transpose of A. This unique generalized inverse had previously been studied by E.H.Moore. Hence, it is commonly known as *Moore-Penrose inverse* and is often denoted by $A^\dagger$.

Various useful properties of the generalized inverses are summarised in the following lemma. Let I designate the $n \times n$ identity matrix. Then, we have, for arbitrary matices,

Lemma 4.1 *[3]*

(a) *$A = 0(m \times n)$ implies $A^\dagger = 0(n \times m)$.*

(b) *$A^{\dagger\dagger} = A$.*

(c) *$A^{\dagger *} = A^{*\dagger}$.*

(d) *If A has full column rank, $A^\dagger = (A^*A)^{-1}A^*$.*

(e) *If A is non-singular, $A^\dagger = A^{-1}$.*

(f) *If U and V are unitary, $(UAV)^\dagger = V^*A^\dagger U^*$.*

(g) *If A is hermitian and idempotent, $A^\dagger = A$.*

(h) *$A^\dagger A, AA^\dagger, I - A^\dagger A$ and $I - AA^\dagger$ are all projection operators.*

(i) *$EF = 0$ implies $F^\dagger E^\dagger = 0$.*

(j) *If P is hermitian and idempotent, $(PQ)^\dagger = Q^\dagger P$ whenever either $PQ = Q$ or P commutes with $Q, Q^\dagger Q$ and $QQ^\dagger$.*

(k) *The equation $AX = B$ has a solution if and only if*

$$AA^\dagger B = B, \tag{4.5}$$

in which case X is a solution if and only if there is a matrix Y such that

$$X = A^\dagger B + (I - A^\dagger A)Y. \tag{4.6}$$

The solution is unique if and only if the null space of A is void, i.e., $(I - A^\dagger A = 0)$.

4.3 Recursive computation of generalized inverse

We shall present a general result for the recursive computation of the generalized inverse for a special kind of matrix which will be used in our incremental computation of the learning operator. Before the result for incremental computation, we shall provide a couple of lemmas which will be used in the proof.

Lemma 4.2 *For any symmetric matrix M, the following conditions are sufficient and necessary conditions for* **non-negative definiteness***[3].*

1. $(Mf, f) \geq 0$ *for any function* f.
2. $M = HH^T$ *for some matrix* H.
3. $x^T Mx \geq$ *for all vectors* x.
4. *The eigen values of* M *are all non-negative.*
5. *There is a symmetric matrix* R, *such that* $R^2 = M$.

Non-negative defineteness of M *is expressed by* $M \geq 0$.

Lemma 4.3 *[4] The Moore-Penrose generalized inverse of a partitioned matrix is given as below. If* H *is a* $m \times m$ *matrix and* h *is a* m*-dimensional vector, then*

$$\begin{pmatrix} H \\ h \end{pmatrix}^{\dagger} = \begin{pmatrix} [I - Kh^t]H^{\dagger} & K \end{pmatrix}, \tag{4.7}$$

$$\text{where } K = \begin{cases} \frac{1}{(h^t Ch)} Ch & \text{if } \alpha_m > 0, \\ \frac{1}{(1+h^t Bh)} Bh & \text{if } \alpha_m = 0. \end{cases}$$

Let S_m be an $m \times m$ non-negative definite matrix i.e., $S_m \geq 0$. Let S_{m+1} be represented in the form

$$S_{m+1} = \begin{pmatrix} S_m & s_m \\ s_m^T & \sigma(m+1) \end{pmatrix}, \tag{4.8}$$

where s_m and $\sigma(m+1)$ represent an m-dimensional vector and scalar, respectively. We assume that $S_0 = 0$, $s_0 = 0$ and $S_m^{\dagger}$ is known. Let t_m, α_m, β_m and T_m be defined as

$$\begin{aligned} t_m &= S_m^{\dagger} s_m, & \alpha_m &= \sigma(m+1) - (s_m, t_m), \\ \beta_m &= 1 + \|t_m\|^2, & T_m &= I - \tfrac{1}{\beta_m}(t_m \otimes \overline{t_m}). \end{aligned}$$

Lemma 4.4 *[3] The Moore-Penrose generalized inverse for S_{m+1} is given as*

$$S_{m+1}^{\dagger} = \begin{cases} \begin{bmatrix} S_m^{\dagger} + \frac{1}{\alpha_m}(t_m \otimes \overline{t_m}) & -\frac{1}{\alpha_m} t_m \\ -\frac{1}{\alpha_m} t_m^T & \frac{1}{\alpha_m} \end{bmatrix} & if\ \alpha_m > 0, \\ \\ \begin{bmatrix} T_m S_m^{\dagger} T_m & \frac{1}{\beta_m}(T_m S_m^{\dagger} t_m) \\ \frac{1}{\beta_m}(T_m S_m^{\dagger} t_m)^T & \frac{1}{\beta_m{}^2}(t_m, S_m^{\dagger} t_m) \end{bmatrix} & if\ \alpha_m = 0. \end{cases} \tag{4.9}$$

Proof Since we will use the intermediate stage of the proof of the above lemma in our later chapters, we present one of the methods of proving it here.

$$S_{m+1} = \begin{pmatrix} S_m & s_m \\ s_m^T & \sigma(m+1) \end{pmatrix}.$$

Since S_{m+1} is a non-negative matrix, it is possible to decompose it as

$$S_{m+1} = \begin{pmatrix} H \\ h^t \end{pmatrix} \begin{pmatrix} H^T & h \end{pmatrix}, \tag{4.10}$$

where H is a $m \times m$ matrix and h is a m-dimensional vector. Let $B = (H^T H)^{\dagger}$ and $C = I - H^{\dagger} H$. From eqs.(4.3) and (4.10),

$$Hh = s_m. \tag{4.11}$$

Solving for h, we have fro some vector u

$$h = H^{\dagger} s_m + Cu. \tag{4.12}$$

Since $Cu \perp H^{\dagger} s_m$, it follows from eqs.(4.3) and (4.10) that

$$\sigma(m+1) = h^t h = \|H^{\dagger} s_m\|^2 + \|Cu\|^2. \tag{4.13}$$

We also have

$$Ch = C(H^{\dagger} s_m + Cu) = C^2 u = Cu. \tag{4.14}$$

Eqs.(4.13) and(4.14) yield

$$\|Ch\|^2 = \sigma(m+1) - \|H^{\dagger} s_m\|^2 \tag{4.15}$$

$$= \sigma(m+1) - s_m^t H^{\dagger T} H^{\dagger} s_m, \tag{4.16}$$

$$\|Ch\|^2 = \sigma(m+1) - s_m^t S_m^{\dagger} s_m = \alpha_m. \tag{4.17}$$

Using Lemma 4.3, we have

$$\begin{pmatrix} H \\ h \end{pmatrix}^{\dagger} = \begin{pmatrix} [H^{\dagger} - K\tau_m^{'}] & K \end{pmatrix}, \tag{4.18}$$

where $\tau_m = S_m^\dagger s_m$.
Also, we have

$$\begin{aligned} S_{m+1}^\dagger &= \begin{pmatrix} S_m & s_m \\ s_m^T & \sigma(m+1) \end{pmatrix} \\ &= \left[\begin{pmatrix} H \\ h \end{pmatrix} \quad \begin{pmatrix} H \\ h \end{pmatrix}^T \right]^\dagger \\ &= \begin{pmatrix} H \\ h \end{pmatrix}^{\dagger T} \begin{pmatrix} H \\ h \end{pmatrix}^\dagger \end{aligned} \tag{4.19}$$

From eq.(4.18) and eq.(4.19), we have

$$S_{m+1}^\dagger = \begin{pmatrix} (S_m^\dagger - H^{\dagger T} K \tau_m^T \\ -\tau_m K^T H^\dagger + \tau_m K^{'} K \tau_m^{'}) & H^{\dagger T} K - \tau_m K^{'} K \\ \left(H^{\dagger T} K - \tau_m K^{'} K\right)^T & K^{'} K \end{pmatrix} \tag{4.20}$$

Since C is a projection operator, we have

$$C^2 = C, \tag{4.21}$$

$$H^{\dagger T} C = 0 \tag{4.22}$$

We also have

$$H^{\dagger T} B h = S_m^\dagger \tau_m, \tag{4.23}$$

$$B h = H^\dagger \tau_m, \tag{4.24}$$

$$h^T B h = \|\tau_m\|^2 \tag{4.25}$$

$$\tag{4.26}$$

Hence, we see that

$$H^{\dagger T} K = \begin{cases} 0 & \text{if } \alpha_m > 0 \\ S_m^\dagger \tau_m / b_m & \text{if } \alpha_m = 0 \end{cases} \tag{4.27}$$

and,

$$K^{'} K = \begin{cases} \frac{1}{\alpha_m} & \text{if } \alpha_m > 0 \\ \tau_m^{'} S_m^\dagger \tau_m / {b_m}^2 & \text{if } \alpha_m = 0 \end{cases} \tag{4.28}$$

Thus,

$$S_m^\dagger - H^{\dagger T} K \tau_m^T - \tau_m K^T H^\dagger + \tau_m K^{'} K \tau_K^{'} = \begin{cases} S_m^\dagger + \tau_m \tau_m^{'} / \alpha_m & \text{if } \alpha_m > 0 \\ T_m S_m^\dagger T_m & \text{if } \alpha_m = 0 \end{cases} \tag{4.29}$$

Substituting eqs.(4.27),(4.28) and (4.29) in eq.(4.20), we have eq.(4.9). ■

4.4 Conclusion

This chapter introduced an extension of the concept of inverses of matrices to provide inverses for singular and rectangular matrices as well, namely, the Moore-Penrose generalized inverse. A recursive method of computing the generalized inverse for a particular type of matrix is shown and proved. This result will be used in the next chapter for computing the learning operator incrementally and hence, incorporating training data incrementally.

Chapter 5

Incorporating training data incrementally

5.1 Introduction

So far, we have formed a framework for discussing the learning problem and established a necessity for looking into incremental type of learning. To this effect, in Chapter 3, the incremental learning concept has been formalised and then, we have derived a result for the incremental computation of the generalized inverse in Chapter 4. In this chapter, we shall deal with the actual problem of incorporating an additional training data efficiently in Section 5.2 and use the tools and models that have been defined for the purpose in earlier chapters. Finally, a computational complexity analysis and a data sampling scheme for reduced complexity is provided in Sections 5.3 and 5.4, respectively.

5.2 Incorporation of new training data

In this section, we shall formulate a method of incorporating the newly available training data to the already computed learning operator from the previous data. This will involve using the incremental approach, already defined in Chapter 3. As mentioned Chapter 3, we shall use incremental Wiener learning defined in Section 3.3 as an example for implementation of this incremental data sampling scheme.

From eqs.(2.19) and (2.20), considering a particular solution for the Wiener learning, i.e., taking $W = 0$,we have

$$X_{m+1} = RA^*_{m+1}U^\dagger_{m+1}. \tag{5.1}$$

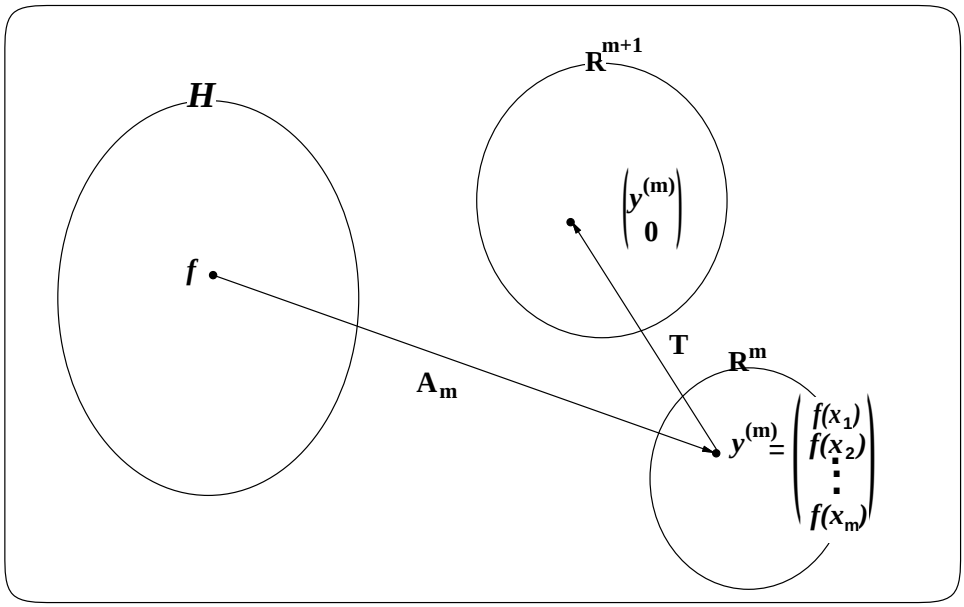

Figure 5.1: Operator T mapping $y^{(m)}$ from R^m into R^{m+1} space

where

$$U_{m+1} = A_{m+1} R A_{m+1}^*. \tag{5.2}$$

Since R is assumed to be non-negative definite, the next lemma holds.

Lemma 5.1 *The matrix U_{m+1} is a non-negative definite matrix.*

Now, consider only the generalized inverse part $U_{m+1}^\dagger$ in eq.(5.1). It follows from eqs.(2.5) and (5.2) that

$$\begin{aligned} U_{m+1} &= A_{m+1} R A_{m+1}^* \\ &= \sum_{n=1}^{m+1} (e_n^{(m+1)} \otimes \overline{\psi}_n) R \sum_{l=1}^{m+1} (\psi_l \otimes \overline{e_l^{(m+1)}}) \\ &= \sum_{n,l=1}^{m+1} (e_n^{(m+1)} \otimes \overline{\psi}_n)(R\psi_l \otimes \overline{e_l^{(m+1)}}) \\ &= \sum_{n,l=1}^{m+1} (R\psi_l, \psi_n)(e_n^{(m+1)} \otimes \overline{e_l^{(m+1)}}). \end{aligned} \tag{5.3}$$

Eq.(5.3) implies that the n,l-th element of the matrix U_{m+1} is $(R\psi_l, \psi_n)$.

Now, we would like to express the new sampling operator A_{m+1} in terms of the old sampling operator A_m. However, it is to be noted that the sampling operator in case of the increased samples is an operator into an higher dimensional space than the previous one. Therefore, to extend the sample vector y from R^m to R^{m+1}, we now introduce an operator T from R^m to R^{m+1} (refer fig.5.1) such that

$$T \equiv \sum_{n=1}^{m} (e_n^{(m+1)} \otimes \overline{e_n^{(m)}}). \tag{5.4}$$

Eq.(5.4) yields

$$Te_p^{(m)} = e_p^{(m+1)} : 1 \leq p \leq m. \tag{5.5}$$

The following is a direct result of eq.(5.5).

Lemma 5.2 *Let u be an m-dimensional vector, i.e. $u \in R^m$ with elements $u = (u_1 u_2 ... u_m)^T$. Then, $Tu \in R^{m+1}$ is always of the form $Tu = (u_1 u_2 ... u_m 0)^T$.*

Lemma 5.3 *Let V be an $m \times m$ matrix with elements $V = (v_{pq} : 1 \leq p, q \leq m)$. Then, TVT^* represents an $(m+1) \times (m+1)$ matrix V' with elements (v'_{pq}) defined as*

$$\begin{aligned} v'_{pq} &= v_{pq} : 1 \leq p, q \leq m, \\ &= 0 \ : \ Otherwise. \end{aligned}$$

Proof Since V is expressed as

$$V = \sum_{p,,q=1}^{m} v_{pq}(e_p^{(m)} \otimes \overline{e_q^{(m)}})$$

, it follows from eq.(5.5) that

$$\begin{aligned} TVT^* &= T[\sum_{p,q=1}^{m} v_{pq}(e_p^{(m)} \otimes \overline{e_q^{(m)}})]T^*, \\ &= \sum_{p,q=1}^{m} v_{pq}[(Te_p^{(m)} \otimes \overline{Te_q^{(m)}})], \\ &= \sum_{p,q=1}^{m} v_{pq}(e_p^{(m+1)} \otimes \overline{e_q^{(m+1)}}). \end{aligned}$$

This implies the lemma. ■

Then, we can write the new sampling operator A_{m+1} as

$$A_{m+1} = TA_m + (e_{m+1}^{(m+1)} \otimes \overline{\psi_{m+1}}). \tag{5.6}$$

Let s_m be the m-dimensional vector defined as

$$s_m = A_m R\psi_{m+1}. \tag{5.7}$$

It follows from eqs.(5.7) and (5.2) that

$$\begin{aligned} s_m &= A_m R\psi_{m+1} \\ &= \sum_{l=1}^{m} (e_l^{(m)} \otimes \psi_l) R\psi_{m+1} \\ &= \sum_{l=1}^{m} (R\psi_{m+1}, \psi_l) e_l^{(m)}, \end{aligned}$$

and hence

$$s_m = \sum_{l=1}^{m} (R\psi_{m+1}, \psi_l) e_l^{(m)}. \tag{5.8}$$

This implies that the l-th element of vector s_m is equal to $(R\psi_{m+1}, \overline{\psi_l})$.

Lemma 5.4 *The matrix U_{m+1} is expressed as*

$$\begin{aligned} U_{m+1} &= TU_mT^* + (Ts_m \otimes \overline{e_{m+1}^{(m+1)}}) + (e_{m+1}^{(m+1)} \otimes \overline{Ts_m}) \\ &\quad + (R\psi_{m+1}, \psi_{m+1})(e_{m+1}^{(m+1)} \otimes \overline{\psi_{m+1}}), \end{aligned} \tag{5.9}$$

which has the structure shown in fig.5.2.

Proof It follows from eqs.(5.6) and (5.2) that

$$\begin{aligned} U_{m+1} &= A_{m+1} R A_{m+1}^* \\ &= [TA_m + (e_{m+1}^{(m+1)} \otimes \overline{\psi_{m+1}})] R [TA_m + (e_{m+1}^{(m+1)} \otimes \overline{\psi_{m+1}})]^* \\ &= TA_m R A_m^* T^* + (TA_m R\psi_{m+1} \otimes \overline{e_{m+1}^{(m+1)}}) + (e_{m+1}^{(m+1)} \otimes \overline{TA_m R\psi_{m+1}}) \\ &\quad + (e_{m+1}^{(m+1)} \otimes \overline{\psi_{m+1}}) R (\psi_{m+1} \otimes \overline{e_{m+1}^{(m+1)}}) \\ &= TS_mT^* + (Ts_m \otimes \overline{e_{m+1}^{(m+1)}}) + (e_{m+1}^{(m+1)} \otimes \overline{Ts_m}) \\ &\quad + (R\psi_{m+1}, \psi_{m+1})(e_{m+1}^{(m+1)} \otimes \overline{e_{m+1}^{(m+1)}}) \end{aligned}$$

Hence, eq.(5.4) holds. Looking at different terms of the expansion, we make the following observations:

- The matrix represented by the term TU_mT^* is an $(m+1) \times (m+1)$ matrix with the $m+1$st column and the $m+1$st row having all zeroes and the remaining $(m \times m)$ sub-matrix having the same terms as U_m. This is clear from Lemma 5.3. Hence, this term corresponds to the top left hand sub-matrix of U_{m+1} as shown in fig.5.2 since no other terms contribute to the above sub-matrix.

- Ts_m, by applying Lemma 5.2, is an $(m+1)$ dimensional vector with the $m+1$st element equal to 0 and the other elements same as vector s_m. Since a Schatten product of a $m+1$ dimensional vector with $e_{m+1}^{(m+1)}$ gives the $m+1$st column of the $(m+1) \times (m+1)$ matrix, the term $(Ts_m \otimes \overline{e_{m+1}^{(m+1)}})$ corresponds to the vector s_m in fig.5.2 since no other terms contribute to the above vector .

- Using similar reasoning, the term $(e_{m+1}^{(m+1)} \otimes \overline{Ts_m})$ corresponds to the $m+1$st row of the $(m+1) \times (m+1)$ matrix U_{m+1} and has the values of the m dimensional vector s_m^T since the last element is zero and no other terms contribute to the above vector.

- The remaining term $(R\psi_{m+1}, \psi_{m+1})(e_{m+1}^{(m+1)} \otimes \overline{e_{m+1}^{(m+1)}})$ corresponds to the $(m+1, m+1)$ element of the matrix U_{m+1} and has the value $\sigma(m+1) = (R\psi_{m+1}, \psi_{m+1})$.

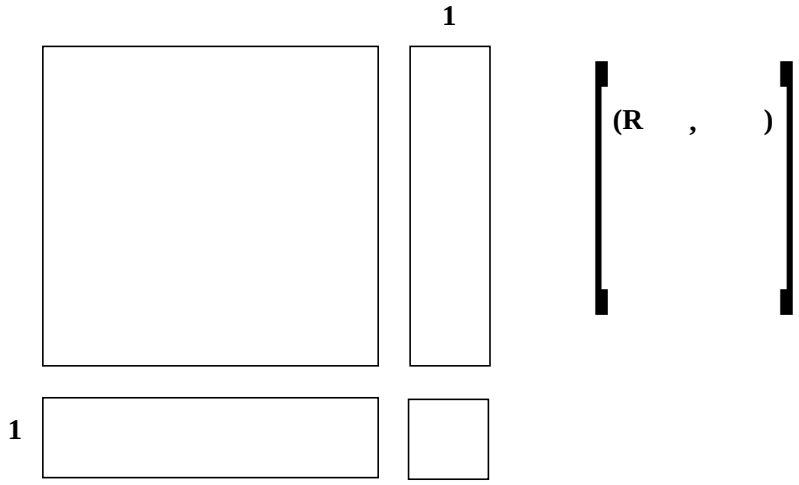

Figure 5.2: Structure of the matrix U_{m+1}

Hence, the matrix U_{m+1} can be written in the form as shown in fig.5.2, which also corresponds to the structure of eq.(4.8). Therefore, the lemma is proved. ■

In the following theorem, many notations will be introduced. They are illustrated in fig.5.3.

Theorem 5.1 *Let s_m, t_m, and α_m be defined as*

$$s_m = A_m R\psi_{m+1}, \tag{5.10}$$

$$t_m = U_m^{\dagger} s_m, \tag{5.11}$$

$$\alpha_m = (R\psi_{m+1}, \psi_{m+1}) - (s_m, t_m). \tag{5.12}$$

Then, the new learning operator X_{m+1} can be computed incrementally from the previous learning operator X_m as

$$X_{m+1} = (X_m + Y_m)T^* + (v_m \otimes \overline{e_{m+1}^{(m+1)}}), \tag{5.13}$$

where in case of $\alpha_m \neq 0$

$$v_m = \frac{1}{\alpha_m}(R\psi_{m+1} - RA_m^* t_m), \tag{5.14}$$

$$Y_m = -(v_m \otimes \overline{t_m}), \tag{5.15}$$

and in case of $\alpha_m = 0$

$$\beta_m = 1 + \|t_m\|^2, \tag{5.16}$$

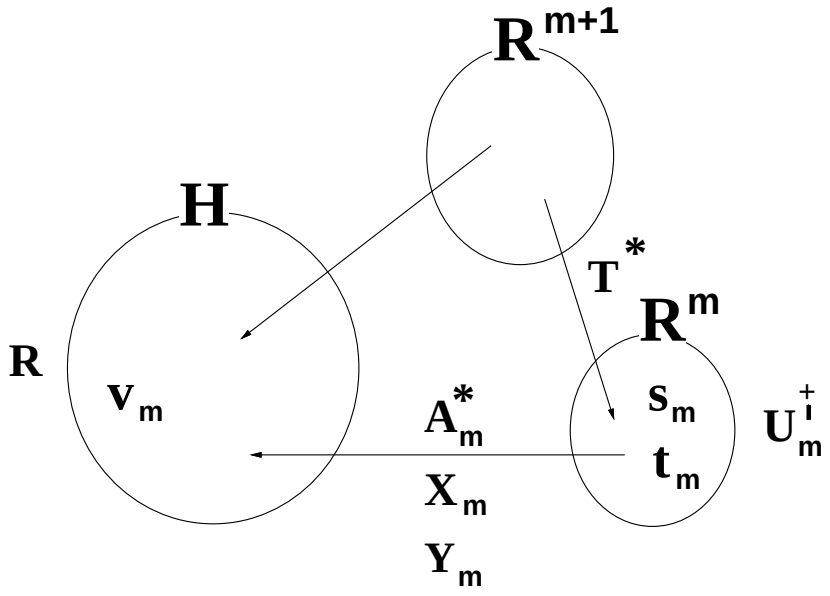

Figure 5.3: Different operators and their relationship with respect to the original function space and the sampled spaces.

$$Z_m = \frac{1}{\beta_m}(t_m \otimes \overline{t_m}), \tag{5.17}$$

$$K_m = (I - Z_m)U_m^\dagger t_m, \tag{5.18}$$

$$v_m = \frac{1}{\beta_m}(RA_m^* K_m + \frac{(t_m, U_m^\dagger t_m)}{\beta_m} R\psi_{m+1}), \tag{5.19}$$

$$Y_m = RA_m^*(Z_m U_m^\dagger Z_m - U_m^\dagger Z_m - Z_m U_m^\dagger) + \frac{1}{\beta_m}(R\psi_{m+1} \otimes \overline{K_m}). \tag{5.20}$$

Proof In case of $\alpha_m \neq 0$, it follows from eqs.(5.1) and (5.6) that

$$X_{m+1} = R[TA_m + (e_{m+1}^{(m+1)} \otimes \overline{\psi_{m+1}})]^* U_{m+1}^\dagger. \tag{5.21}$$

Since U_{m+1} in eq.(5.21) is non-negative definite and has the structure as shown in fig.5.2, we can apply Lemma 4.4 to eq.(5.21). Hence, we have

$$X_{m+1} = R[TA_m + (e_{m+1}^{(m+1)} \otimes \overline{\psi_{m+1}})]^* \begin{bmatrix} U_m^\dagger + \mathcal{A} & \mathcal{B} \\ \mathcal{B}^T & \mathcal{C} \end{bmatrix}, \tag{5.22}$$

where $\mathcal{A}, \mathcal{B}, \mathcal{C}$ are defined as below.

$$\mathcal{A} = \frac{1}{\alpha_m}(t_m \otimes \overline{t_m}), \tag{5.23}$$

$$\mathcal{B} = -\frac{1}{\alpha_m} t_m, \tag{5.24}$$

$$\mathcal{C} = \frac{1}{\alpha_m}. \tag{5.25}$$

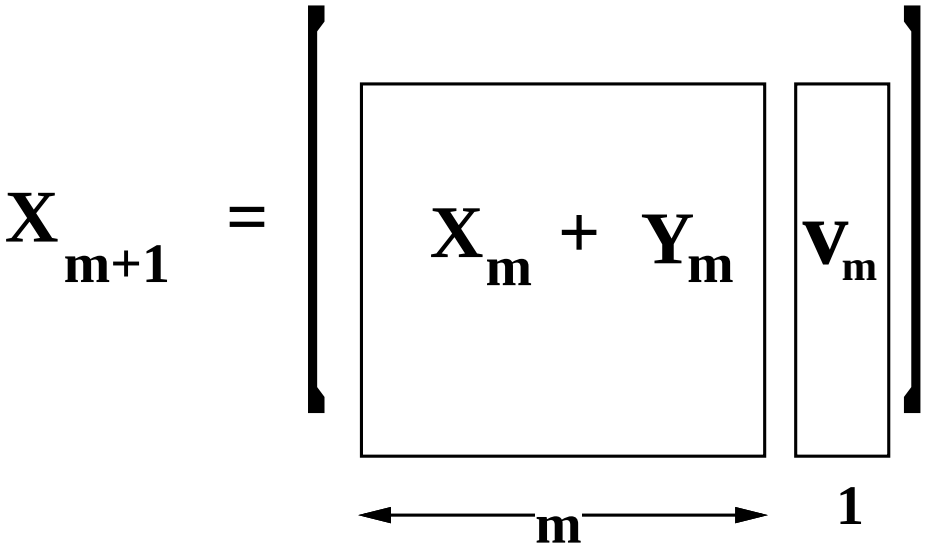

Figure 5.4: Structure of the final result for incremental computation of the Wiener learning operator.

Applying Lemma 5.2 and Lemma 5.3 to eq.(5.22), we have

$$\begin{aligned} X_{m+1} &= R[A_m^* T^* + (\psi_{m+1} \otimes \overline{\psi_{m+1}})][T(U_m^\dagger + \mathcal{A})T^* + (T\mathcal{B} \otimes \overline{e_{m+1}^{(m+1)}}) + (e_{m+1}^{(m+1)} \otimes \overline{T\mathcal{B}}) \\ &\quad + \mathcal{C}(e_{m+1}^{(m+1)} \otimes \overline{e_{m+1}^{(m+1)}})]. \end{aligned} \tag{5.26}$$

Since $T^*T = I_m$ and $T^* e_{m+1}^{(m+1)} = 0$, it follows from eqs.(5.26), (5.1), (5.14), and (5.15) that

$$\begin{aligned} X_{m+1} &= RA_m^*(U_m^\dagger + \mathcal{A})T^* + (RA_m^*\mathcal{B} \otimes \overline{e_{m+1}^{(m+1)}}) + R(\psi_{m+1} \otimes \overline{T\mathcal{B}}) + (R\psi_{m+1}\mathcal{C} \otimes \overline{e_{m+1}^{(m+1)}}) \\ &= (X_m + RA_m^*\mathcal{A} + R(\psi_{m+1} \otimes \overline{\mathcal{B}}))T^* + [(RA_m^*\mathcal{B} + R\psi_{m+1}\mathcal{C}) \otimes \overline{e_{m+1}^{(m+1)}}]. \end{aligned} \tag{5.27}$$

Substituting the values of $\mathcal{A}, \mathcal{B}, \mathcal{C}$ into eq.(5.27), we have

$$\begin{aligned} X_{m+1} &= [X_m + \frac{1}{\alpha_m}(RA_m^*(t_m \otimes \overline{t_m}) - R(\psi_{m+1} \otimes \overline{t_m}))]T^* \\ &\quad + [\frac{-1}{\alpha_m}(RA_m^* t_m - R\psi_{m+1}) \otimes \overline{e_{m+1}^{(m+1)}}], && (5.28) \\ &= [X_m - (v_m \otimes \overline{t_m})]T^* + (v_m \otimes \overline{e_{m+1}^{(m+1)}}), && (5.29) \\ &= (X_m + Y_m)T^* + (v_m \otimes \overline{e_{m+1}^{(m+1)}}). && (5.30) \end{aligned}$$

Hence, we proved Case I of Theorem 5.1.

In case of $\alpha_m = 0$ by similar analysis as Case I, and substituting the values for $\mathcal{A}, \mathcal{B}, \mathcal{C}$ based on Lemma 4.4, we can derive the result for Case II. This completes the proof of the Theorem. ■

As is evident from the result of Theorem 5.1, it is possible to compute the new learning operator incrementally by using the intermediate results being output at the previous stage. This is a big saving over the case when we have to compute the learning operator from scratch, every time a new training data or information comes in. However, there is a difference in the computational complexity for the cases $\alpha_m \neq 0$ and $\alpha_m = 0$. The consequences and the solution for this condition is dealt with in the next section.

A graphical representation of the result for incremental computation of the new learning operator X_{m+1} is shown in fig.5.4. When we transform the sample vector $y^{(m+1)}$ using the learning operator X_{m+1}, from the analysis of the structure the next corollary follows.

Corollary 5.1 *The learned function due to $m+1$ sample points f_{m+1} can be computed incrementally from the previous learned function f_m and the new sample value $f(x_{m+1})$ as*

$$f_{m+1} = f_m + Y_m y^{(m)} + f(x_{m+1}) v_m. \tag{5.31}$$

where Y_m and v_m are given by eqs.(5.15) and (5.14) for $\alpha_m \neq 0$ and by eqs.(5.20) and (5.19) for $\alpha_m = 0$.

Proof From the neural network model of Chapter 2, we have

$$f_{m+1} = X_{m+1} y^{(m+1)}. \tag{5.32}$$

Applying the result of Theorem 5.1 to the above equation, we have

$$f_{m+1} = (X_m + Y_m) T^* y^{(m+1)} + (v_m \otimes \overline{e_{m+1}^{(m+1)}}) y^{(m+1)}. \tag{5.33}$$

Since $T^* y^{(m+1)} = y^{(m)}$, we can write the above equation as

$$f_{m+1} = (X_m + Y_m) y^{(m)} + (y^{(m+1)}, e_{m+1}^{(m+1)}) v_m. \tag{5.34}$$

Since $X_m y^{(m)} = f_m$ and $(y^{(m+1)}, e_{m+1}^{(m+1)}) = f(x_{m+1})$, eq.(5.34) yields eq.(5.31). ■

5.3 Computational complexity analysis

In this section, we shall evaluate the computational complexity involved in computing the Wiener learning operator incrementally. As is evident from the result of the previous section, the incremental computation algorithm has two possible methods of computation based on the value of the variable α_m.

The reconstruction operator X_{m+1} can be computed incrementally, whatever be the value of the variable α_m. However, there is a difference in the computational cost involved in both cases. Here, we will analyse the cost involved in terms of the scalar multiplications for the two cases of Theorem (5.1).

Assume a N dimensional original function space. Then, we can compute the computational cost as under.

(i) When $\alpha_m \neq 0$: The computational complexity involved works out to be of the order of $m^2 + (N^2 + 3N + 1)m + (N^2 + 2N)$ scalar multiplications.

(ii) When $\alpha_m = 0$: The computational complexity involved works out to be of the order of $3m^3 + (5 + N)m^2 + (N^2 + 4N + 3)m + (N^2 + 2N + 1)$ scalar multiplications.

The same kind of analysis is also possible for infinite dimensional original function space, which also provides the results on the same lines.

Hence, to *reduce computational complexity*, it is essential to ensure that the *condition* $\alpha_m \neq 0$ *holds for the next sample point* as far as possible. In the next section, we will analyse the condition for reducing the computational complexity and derive a result for ensuring reduced computational complexity.

5.4 Data Sampling for reduced computational complexity

In this section, we shall derive the necessary and sufficient condition for $\alpha_m \neq 0$ and interpret it in terms for Wiener learning.

We once again refer to eq.(4.9). During the derivation of the result, we had encountered the conditions on α_m which resulted in the two subcases being output as the result. Based on this derivation we shall prove a lemma which gives the necessary and sufficient condition for $\alpha_m \neq 0$.

Lemma 5.5 *It holds that $\alpha_m \neq 0$ if and only if*

$$rank(U_{m+1}) = rank(U_m) + 1. \tag{5.35}$$

Proof From the proof of the iterative algorithm in Section 4.4, we have

$$U_{m+1} = \begin{pmatrix} H \\ h^t \end{pmatrix} \begin{pmatrix} H^T & h \end{pmatrix}, \tag{5.36}$$

$$\alpha_m = \|Ch\|^2 = \|(I - HH^T)h\|^2, \tag{5.37}$$

where $HH^T = U_m$. Hence, from eq.(5.37), $\alpha_m = 0$ if and only if $(I - HH^T)h = 0$. In other words, $\alpha_m = 0$ if and only if there exists x such that $H^T x = h$. Therefore, $\alpha_m = 0$ if and only if

$$rank(\begin{pmatrix} H^T & h \end{pmatrix}) = rank(H^T). \tag{5.38}$$

We also have

$$rank(U_m) = rank(HH^T) = rank(H^T). \tag{5.39}$$

$$rank(U_{m+1}) = rank(\begin{pmatrix} H^T & h \end{pmatrix}). \tag{5.40}$$

From eqs.(5.38),(5.39), and (5.40) , we have

$$rank(U_{m+1}) = rank(H^T) + \begin{cases} 0 & \text{if there exists } x \text{ such that } H^T x = h \\ 1 & \text{otherwise} \end{cases} \quad (5.41)$$

$$= rank(U_m) + sgn(\alpha_m) \quad (5.42)$$

Hence the lemma is proved. ■

Based on the above Lemma 5.5, we will now present a result on sampling for reduced computational complexity under Wiener learning criterion. Considering the meaning of the matrix U_m in terms of the Wiener learning criterion, we know that the matrix represented here corresponds to the generalized inverse term of the Wiener learning operator. Hence,

$$U_m = A_m R A_m^* = A_m R^{\frac{1}{2}} (A_m R^{\frac{1}{2}})^*. \quad (5.43)$$

Since $rank(.) = dim(\mathcal{R}(.))$, we have

$$\begin{aligned} rank(U_m) &= dim(\mathcal{R}(A_m R^{\frac{1}{2}} (A_m R^{\frac{1}{2}})^*)) \\ &= dim(\mathcal{R}((A_m R^{\frac{1}{2}})^*)) \\ rank(U_m) &= dim(\mathcal{R}(R A_m^*)) \end{aligned} \quad (5.44)$$

Hence, it follows from eqs.(5.44) and (5.35) that

Theorem 5.2 *It holds that $\alpha_m \neq 0$ if and only if*

$$dim(\mathcal{R}(R A_{m+1}^*)) - dim(\mathcal{R}(R A_m^*)) = 1. \quad (5.45)$$

Based on this result, if the training data or sampled data is selected in such a way so as to satisfy eq.(5.45), then, we can ensure reduced computational complexity in the incremental computation of the learning operator.

When the sampling satisfies the condition for reduced compuatational complexity, in other words, the next sample point x_{m+1} is selected in such a way that eq.(5.2) is satisfied, we have the following corollary.

Corollary 5.2 *The new learned function f_{m+1} using $m+1$ sample points can be expressed as the sum of the old learned function f_m and another function belonging to the original function space H as*

$$f_{m+1} = f_m + \alpha v_m, \quad (5.46)$$

where α is a scalar defined as

$$\alpha = f(x_{m+1}) - (y^{(m)}, t_m). \quad (5.47)$$

Proof Using eqs.(5.31) and (5.15), we have

$$f_{m+1} = f_m + (-v_m \otimes \overline{t_m})y^{(m)} + f(x_{m+1})v_m \quad (5.48)$$

$$= f_m - (y^{(m)}, t_m)v_m + f(x_{m+1})v_m \quad (5.49)$$

$$= f_m + (f(x_{m+1}) - (y^{(m)}, t_m))v_m. \quad (5.50)$$

Hence, the corollary is proved. ■

5.5 Conclusion

We have derived, in this chapter, a method of expressing the new learning operator X_{m+1} due to $m+1$ sample points as a function of the previous learning operator A_m and the new sample point x_{m+1}. Hence, it is possible to compute the learning operator incrementally, leading to a reduction in the computational cost. The incremental computation had two different computation schemes based on the value of a particular variable α_m given by eq.(5.12). A study of the computational cost in either cases was carried out and a sampling scheme or selection of training set for ensuring further reduction in the computational cost was established. We also derived an incremental method of computing the new learned function f_{m+1} from the previously learned function f_m, which was a direct result of the incremental result of the learning operator. In the next chapter, we will see how this sampling scheme compares to the main objective of optimal sampling for maximal generalization ability.

Chapter 6

Training data selection for improved generalization

6.1 Introduction

As has been mentioned in Section 3.2, the theoretical basis for the selection of training data has not been established yet. However, there has been a lot of of experimental work on ways and methods of choosing a training set [7], [9], [20], [22]. Especially in the area of pattern classification, there has been a lot of experimental work on how to choose a border pattern or classifying pattern. Analogous to that, in the learning in neural networks, we would like to select training data which results in maximum generalizing ability.

Selection of training data can be done in broadly two ways. First, all the data can be selected in one go under *batch selection*. The other is to incrementally select the training data. In the incremental selection, again, we can broadly classify the criterion for selection as 'active', which implies the value of the previously selected sample point will affect the selection of the next training set and 'non-active' or plain incremental. In this chapter, we shall look into a non-active incremental sampling scheme for ensuring maximal generalization ability. Section 6.3 provides the modified Wiener criterion for incremental sampling. We will compare the result on sampling for reduced computational cost obtained in the previous chapter applied to the goal of improving generalization ability in Section 6.4. Finally, computer simulations with different training data is carried out to verify our results.

6.2 Information measure

When the problem of deciding which training data to select next comes up, then, the question of comparing the available data options obiviously arise. Our criterion for how informative a new datum is will depend on what we are interested in.

Several alternatives that can be considered are :

1. We might want the new data to be maximally informative in a global sense; i.e. reduce the overall error term to the maximum.

2. Alternatively, we might not be interested in getting a globally informative next data; instead, we might want to get a better approximation around the region where we may not be able to sample directly.

3. Lastly, we might want data that maximally helps in distinguishing between the possible options; for example, in a probabilistic model, one which helps in eliminating the possibilty of some model while confirming the other.

The aim would be to find a utility as a function of the next sample location x_{m+1}. This utility is often referred to as the *information measure* of the new data. There is a possibility of addressing a more complex task of selecting *multiple* new data points but here, we will consider only the next data point.

As for the evaluation of the information measure is concerned, a lot of work has been done in different fields [7], [9], [19], [20]. In this thesis, we will use the Wiener optimization criterion to serve as a measure of the information gained due to the newly sampled data, since the *reduction of error* over the new sample points can be viewed as *new information gained*. Therefore, this can be interpreted as looking for globally informative data which reduces the error in an averaged sense over the original function space H. In the next section, we shall formalize the Wiener criterion based on incremental sampling in order to find a framework for selecting optimal training data.

6.3 Modified Wiener criterion based on incremental sampling

In this section, we shall present a modified Wiener learning criterion which is based on a optimal selection of training points. We shall assume an incremental method of choosing, which is the basic theme that has been developed over this thesis. Hence, we shall consider going from a stage of m sample points to $m+1$ sample points. Eq.(2.16) representing the Wiener optimization can be formulated in view of *incremental learning* and *optimal sampling* theory as:

$$\text{minimize over } x_{m+1} : J^{(0)}_{Wm+1}[x_{m+1}], \tag{6.1}$$

where

$$J^{(0)}_{Wm+1}[x_{m+1}] \equiv \min_{X_{m+1}} J_{Wm+1}[X_{m+1}], \tag{6.2}$$

$$J_{Wm+1}[X_{m+1}] \equiv E_f \|X_{m+1}A_{m+1}f - f\|^2. \tag{6.3}$$

The modified optimization criterion now involves a two level optimization. This involves finding the most optimal next sample point x_{m+1} which minimizes the error term $J^{(0)}_{Wm+1}[x_{m+1}]$; where $J^{(0)}_{Wm+1}$ is the optimization on the learning operator X_{m+1} for a given training set.

6.4 Training data selection for improved generalization

In this section, we will look into one of our major goals of training data selection for improved generalization. Ideally speaking, to obtain the optimum next sample point one must solve the eq.(6.1). In this dissertation, we will not attempt to solve this directly. Instead, we will compare the results on sampling obtained in Section 5.4, which was based on reducing computational complexity, to this original goal.

Based on result on sampling for reducing computational complexity of Theorem 5.2, we can basically divide the whole set of possible next sample points into two categories.

(a) Those sample points that satisfy the condition for $\alpha_m \neq 0$ (Optimal for reducing computational complexity).

(b) Those sample points that do not satisfy the condition for $\alpha_m \neq 0$ (Non-optimal for reducing computational complexity).

Theorem 6.1 *The most optimally generalizing sample point(s) lie within the category which satisfy the condition of $\alpha_m \neq 0$ (optimal for computational complexity).*

Proof

In general, for $R > 0$,

$$\mathcal{R}(RA^*_{m+1} \supset \mathcal{R}(RA^*_m). \tag{6.4}$$

For the non-optimal category, i.e. when $\alpha_m = 0$, from the proof of Lemma 5.5, we have

$$dim\mathcal{R}(RA^*_{m+1}) - dim\mathcal{R}(RA^*_m) = 0. \tag{6.5}$$

From eqs.(6.4) and (6.5), since we are considering finite dimensional spaces, we can say that

$$\mathcal{R}(RA^*_{m+1}) = \mathcal{R}(RA^*_m) \text{ for the category with } \alpha_m = 0. \tag{6.6}$$

Now, we can make two general observations.

1. From Lemma 3.5, it is known that Wiener learning is a projection on $\mathcal{R}(RA^*)$ along $\mathcal{N}(A)$. So for the case $\alpha_m \neq 0$, the Wiener learning is a projection on the subspace $\mathcal{R}(RA^*_{m+1})$ whereas for the case $\alpha_m = 0$, it is a projection onto $\mathcal{R}(RA^*_m) \subset \mathcal{R}(RA^*_{m+1})$, both along $\mathcal{N}(A_{m+1})$.

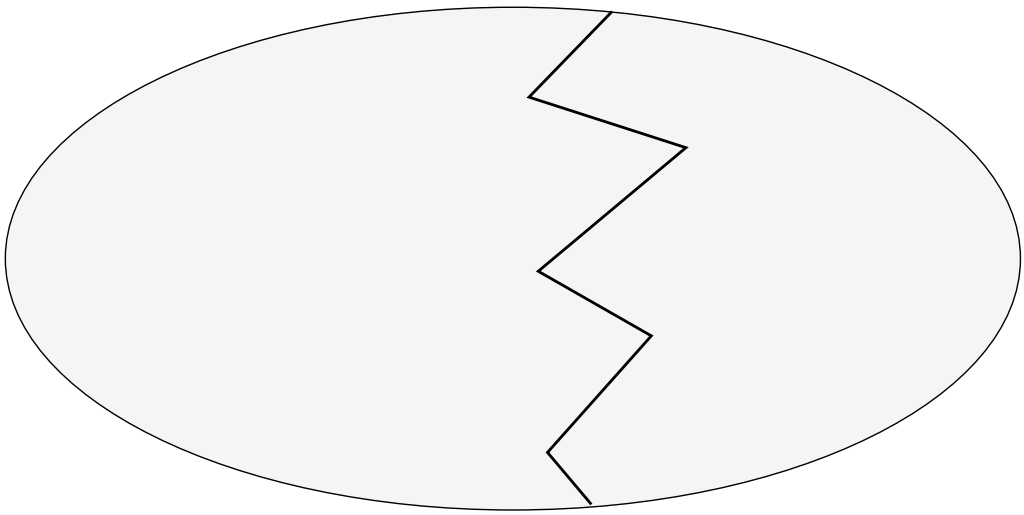

Figure 6.1: Sampling for computational efficiency versus generalizing ability

2. From equation 3.1, we know that for Wiener learning in noiseless case, additive training samples ensure the same or better approximation.

Therefore, from these two facts, it is obivious that the category with $\alpha_m = 0$ is the *lower bound* or worst case of the additive samples possible. Hence. the optimal sample(s) with optimal generalizing ability surely lies in the category with $\alpha_m \neq 0$, which corresponds to the category for sampling to ensure reduced computational complexity.

Hence, Theorem 6.1 is proved. ■

This can be interpreted in another way. This means that the goal of reducing computational complexity and the one of obtaining optimally generalizing sampling scheme which involves determining the optimal samples are compatible with each other, i.e. the goals can be satisfied simultaneously.

In the next section, we will carry out computer simulations of a learning problem in 4-dimensional space and try and verify the various results obtained so far with regard to sampling for optimal generalization and reduced complexity.

6.5 Computer simulations

For the computer simulation and verification of the results on sampling of training data obtained so far, we define the parameters as below.

Let H be the 4-dimensional linear space which is defined as the space containing all the linear combinations of the vector set

$$\{1, \sin\, x,\ \sin\, 2x,\ \sin\, 3x\} \tag{6.7}$$

The inner product in this space is defined as

$$(f, g) = \int_{-\pi}^{+\pi} f(x)g(x)dx. \tag{6.8}$$

The orthonormal bases(ONB) of the space H is as below.

$$\{\frac{1}{\sqrt{2\pi}}, \frac{1}{\sqrt{\pi}} \sin\, x, \frac{1}{\sqrt{\pi}} \sin\, 2x, \frac{1}{\sqrt{\pi}} \sin\, 3x\}. \tag{6.9}$$

The reproducing kernel of this space H is determined using the formula

$$K(x, x') = \sum_{m=1}^{L} \varphi_m(x)\varphi_m(x'). \tag{6.10}$$

where L is the dimension of the original function space and also equal to the number of orthonormal bases.

The reproducing kernel for the above space works out to be

$$K(x, x') = \frac{1}{2\pi} + \frac{1}{\pi}(\ \sin\, x\ \sin\, x' +\ \sin\, 2x\ \sin\, 2x',\ \sin\, 3x\ \sin\, 3x'). \tag{6.11}$$

For the sampling operator, we will consider the cases with two sample points and three sample points, respectively. The sampling operators A_2 amd A_3 are defined as below.

$$A_2 = \frac{1}{\sqrt{2\pi}} \begin{pmatrix} 1 & \sqrt{2}\sin\, x_1 & \sqrt{2}\sin\, 2x_1 & \sqrt{2}\sin\, 3x_1 \\ 1 & \sqrt{2}\sin\, x_2 & \sqrt{2}\sin\, 2x_2 & \sqrt{2}\sin\, 3x_2 \end{pmatrix}. \tag{6.12}$$

$$A_3 = \frac{1}{\sqrt{2\pi}} \begin{pmatrix} 1 & \sqrt{2}\sin\, x_1 & \sqrt{2}\sin\, 2x_1 & \sqrt{2}\sin\, 3x_1 \\ 1 & \sqrt{2}\sin\, x_2 & \sqrt{2}\sin\, 2x_2 & \sqrt{2}\sin\, 3x_2 \\ 1 & \sqrt{2}\sin\, x_3 & \sqrt{2}\sin\, 2x_3 & \sqrt{2}\sin\, 3x_3 \end{pmatrix}. \tag{6.13}$$

For Wiener learning, we have to have a probabilistic model of the possiblities of the occurance of the functions in the function space H. It is through this that we determine the correlation operator R. However, in actual practice, even though the individual functions in the space H are not known, we assume certain properties and either fix a particular correlation matrix or determine it through some statistical means.

For the simulation purpose here, we assume a limited set of 4 different functions, with the corresponding probabilities as shown in the table below.

function	probabilty of occurance
$\frac{1}{\sqrt{2}}$	0.25
$\sin x + \sin 2x$	0.25
$\sin 3x + \sin 2x$	0.25
$\sin 3x - \sin x$	0.25

Based on this, if we compute the correlation operator matrix using the formula

$$R \equiv E_f(f \otimes \overline{f}). \tag{6.14}$$

we have,

$$R = \begin{bmatrix} 0.7854 & 0 & 0 & 0 \\ 0 & 1.5708 & 0.7854 & -0.7854 \\ 0 & 0.7854 & 1.5708 & 0.7854 \\ 0 & -0.7854 & 0.7854 & 1.5708 \end{bmatrix}.$$

Before actually presenting the simulation results, let us try and understand the significance of the various sample points that will be selected.

Based on the theorem 5.2, if we satisfy the condition

$$dim\{\mathcal{R}(RA_{m+1}^*)\} - dim\{\mathcal{R}(RA_m^*)\} = 1. \tag{6.15}$$

then, it ensures reduced computational complexity. Also, in Chapter 6, it has been shown theoretically that training samples which satisfy the condition of theorem 5.2 also generalize better than those which do not satisfy this condition. We will try and verify this fact in the simulations.

Another point to be noted is that the condition of eq.(6.15) can be also stated as

$$rank(A_{m+1}RA_{m+1}^*) - rank(A_mRA_m^*) = 1. \tag{6.16}$$

Hence, it is sufficient to test the rank of the matrix $A_{m+1}RA_{m+1}^*$ to find out if the condition is satisfied or not.

6.5.1 Results of simulation (I)

Using this above framework, we will test the learning ability (and the generalization) for different cases. The different cases for which the simulation was carried out is listed below.

Case	*sample points*	*Remarks*
Case 1	$(0, \frac{\pi}{2}, \pi)$	$dim\mathcal{R}(RA_3^*) - dim\mathcal{R}(RA_2^*) = 0$
Case 2	$(0, \frac{\pi}{2}, \frac{\pi}{4})$	$dim\mathcal{R}(RA_3^*) - dim\mathcal{R}(RA_2^*) = 1$
Case 3	$(0, \frac{\pi}{4})$	$rank(A_2RA_2^*) = 2$
Case 4	$(0, \frac{\pi}{2})$	$rank(A_2RA_2^*) = 2$

Refering to the cases taken up in the simulation, case 1 corresponds to 3 sample points which does not satisfy the condition as stated in the remark, whereas case 2 corresponds to the case when they satisfy the condition. Case 3 and 4 correspond to learning with 2 sample points, each of then satisfying the full rank condition of eq.(6.16). We will carry out the simulation for the learning of functions both within and outside $\mathcal{R}(R)$.

Example (A):

Function to be reconstructed :	$f = \frac{1}{\sqrt{2}} + \sin x + 3 \sin 2x + 2 \sin 3x$
Vector representation :	$f = \begin{bmatrix} 1.7725 \\ 1.7725 \\ 5.3174 \\ 3.5449 \end{bmatrix}$.
Property of function:	$f \in \mathcal{R}(R)$

Case 1:

Training data : $\{(x_m, y_m)\}_{m=1}^M = \{(0, \frac{1}{\sqrt{2}}), (\frac{\pi}{2}, \frac{1}{\sqrt{2}} - 2), (\pi, \frac{1}{\sqrt{2}})\}$.

Property: $dim\mathcal{R}(RA_3^*) - dim\mathcal{R}(RA_2^*) = 0$

Reconstructed function:

$$f_0 = \frac{1}{\sqrt{2}} + \frac{1}{2}(\sin 3x - \sin x). \tag{6.17}$$

Case 2:

Training set : $\{(x_m, y_m\}_{m=1}^M = \{(0, \frac{1}{\sqrt{2}}), (\frac{\pi}{2}, \frac{1}{\sqrt{2}} - 1), (\frac{\pi}{4}, 3 + 2\sqrt{2})\}$.

Property: $dim\mathcal{R}(RA_3^*) - dim\mathcal{R}(RA_2^*) = 1$

Reconstructed function:

$$f_0 = \frac{1}{\sqrt{2}} + \sin x + 3 \sin 2x + 2 \sin 3x. \tag{6.18}$$

Case 3:

Training data : $\{(x_m, y_m)\}_{m=1}^{M} = \{(0, \frac{1}{\sqrt{2}}), (\frac{\pi}{4}, 3 + 2\sqrt{2})\}$.

Property: $rank(A_2 R A_2^*) = 2$

Reconstructed function:

$$f_0 = \frac{1}{\sqrt{2}} + 3 \sin 2x + \frac{2.6587}{\sqrt{\pi}}(\sin x + \sin 3x). \tag{6.19}$$

Case 4:

Training data : $\{(x_m, y_m)\}_{m=1}^{M} = \{(0, \frac{1}{\sqrt{2}}), (\frac{\pi}{2}, \frac{1}{\sqrt{2}} - 2)\}$.

Property: $rank(A_2 R A_2^*) = 2$

Reconstructed function:

$$f_0 = \frac{1}{\sqrt{2}} + \frac{1}{2}(\sin 3x - \sin x). \tag{6.20}$$

Example (B):

Function to be reconstructed :	$f = \sin x - \sin 2x$
Vector representation :	$f = \begin{bmatrix} 0 \\ 1.7725 \\ -1.7725 \\ 0 \end{bmatrix}$.
Property of function:	$f \notin \mathcal{R}(R)$

Case 1:

Training data : $\{(x_m, y_m)\}_{m=1}^{M} = \{(0, \frac{1}{\sqrt{2}}), (\frac{\pi}{2}, \frac{1}{\sqrt{2}} - 2), (\pi, \frac{1}{\sqrt{2}})\}$.

Property: $dim\mathcal{R}(RA_3^*) - dim\mathcal{R}(RA_2^*) = 0$

Reconstructed function:

$$f_0 = \frac{1}{2}(\sin x - \sin 3x). \tag{6.21}$$

Case 2:

Training set : $\{(x_m, y_m\}_{m=1}^{M} = \{(0, \frac{1}{\sqrt{2}}), (\frac{\pi}{2}, \frac{1}{\sqrt{2}} - 1), (\frac{\pi}{4}, 3 + 2\sqrt{2})\}$.

Property: $dim\mathcal{R}(RA_3^*) - dim\mathcal{R}(RA_2^*) = 1$

Reconstructed function:

$$f_0 = \frac{1}{\sqrt{\pi}}(0.7342 \sin x - 0.3041 \sin 2x - 1.0383 \sin 3x). \tag{6.22}$$

Case 3:

Training data : $\{(x_m, y_m)\}_{m=1}^{M} = \{(0, \frac{1}{\sqrt{2}}), (\frac{\pi}{4}, 3 + 2\sqrt{2})\}$.

Property: $rank(A_2 R A_2^*) = 2$

Reconstructed function:

$$f_0 = \frac{-0.1521}{\sqrt{\pi}}(\ \sin\ x + 2\ \sin\ 2x + \ \sin\ 3x). \tag{6.23}$$

Case 4:

Training data : $\{(x_m, y_m)\}_{m=1}^{M} = \{(0, \frac{1}{\sqrt{2}}), (\frac{\pi}{2}, \frac{1}{\sqrt{2}} - 2)\}$.

Property: $rank(A_2 R A_2^*) = 2$

Reconstructed function:

$$f_0 = \frac{1}{2}(\ \sin\ x - \ \sin\ 3x). \tag{6.24}$$

6.5.2 Results of simulation (II)

We carried out another simulation with a 5 dimensional original function space H covered using the vectors

$$\{1, \sin\ x,\ \sin\ 2x,\ \sin\ 3x,\ \sin\ 4x\} \tag{6.25}$$

The orthonormal bases(ONB) of the space H is as under:

$$\{\frac{1}{\sqrt{2\pi}}, \frac{1}{\sqrt{\pi}}\ \sin\ x, \frac{1}{\sqrt{\pi}}\ \sin\ 2x, \frac{1}{\sqrt{\pi}}\ \sin\ 3x, \frac{1}{\sqrt{\pi}}\ \sin\ x\}. \tag{6.26}$$

The functions on the basis of which the correlation operator R is computed is as given below.

function	probabilty of occurance
$\frac{1}{\sqrt{2}} + \ \sin\ 4x$	0.25
$\sin\ x + \ \sin\ 2x$	0.25
$\sin\ 2x + \ \sin\ 3x$	0.25
$\sin\ 4x - \ \sin\ 2x + \ \sin\ x$	0.25

The different cases for which simulation was carried out is shown in the table below.

Case	*sample points*	*Remarks*
Case 1	$(0, \frac{\pi}{2}, \pi)$	$dim\mathcal{R}(RA_3^*) - dim\mathcal{R}(RA_2^*) = 0$
Case 2	$(0, \frac{\pi}{2}, \frac{\pi}{4})$	$dim\mathcal{R}(RA_3^*) - dim\mathcal{R}(RA_2^*) = 1$
Case 3	$(0, \frac{\pi}{4}, \frac{\pi}{3})$	$dim\mathcal{R}(RA_3^*) - dim\mathcal{R}(RA_2^*) = 1$
Case 4	$(0, \frac{\pi}{2})$	$rank(A_2RA_2^*) = 2$

Example (C):

Function to be reconstructed :	$f = \frac{1}{\sqrt{2}} + 6 \sin x - 7 \sin 2x - 3 \sin 3x + 6 \sin 4x$
Vector representation :	$f = \begin{bmatrix} 0.7071 \\ 4.2426 \\ -4.9497 \\ -2.1213 \\ 4.2426 \end{bmatrix}$.
Property of function:	$f \in \mathcal{R}(R)$

Case 1:

Training set : $\{x_m\}_{m=1}^3 = \{0, \frac{\pi}{2}, \pi\}$.

Property: $dim\mathcal{R}(RA_3^*) - dim\mathcal{R}(RA_2^*) = 0$

Reconstructed Function:

$$f_0 = \frac{1}{\sqrt{2}} + 6 \sin x - 3 \sin 2x - 3 \sin 3x + 4 \sin 4x. \tag{6.27}$$

Case 2:

Training set : $\{x_m\}_{m=1}^3 = \{0, \frac{\pi}{2}, \frac{\pi}{4}\}$.

Property: $dim\mathcal{R}(RA_3^*) - dim\mathcal{R}(RA_2^*) = 1$

Reconstructed function:

$$f_0 = \frac{1}{\sqrt{2}} + 4.9062 \sin x - 5.4531 \sin 2x - 4.0938 \sin 3x + 4.1327 \sin 4x. \tag{6.28}$$

Case 3:

Training set : $\{x_m\}_{m=1}^{3} = \{0, \frac{\pi}{2}, \frac{\pi}{6}\}$.

Reconstructed function:

$$f_0 = \frac{1}{\sqrt{2}} + 5.4226 \sin x - 3.8453 \sin 2x - 3.5774 \sin 3x + 3.8453 \sin 4x. \tag{6.29}$$

Case 4:

Training set : $\{x_m\}_{m=1}^{2} = \{0, \frac{\pi}{2}\}$.

Property: $rank(A_2 R A_2^*) = 2$

Reconstructed function:

$$f_0 = \frac{1}{\sqrt{2}} + 6(\sin x - 3 \sin 2x - 3 \sin 3x + 4 \sin 4x). \tag{6.30}$$

6.5.3 Interpretation of the simulation results

In this section, we will analyse the results of the simulation and present the interpretations.

Analysis of Example (C)

To begin with, let us start with Example (C). This is a learning in 5 dimensional original functional space with a maximum of three sample points. In Case I, II and III, we consider 3 sample points, with only the third sample point being different. As mentioned in the simulation, Case I does not satisfy the condition for optimal sampling of Theorem 5.2, while Case II and III satisfy it.

Comparing case I and case II (refer fig.6.6), it is clear that the learning in case of Case II is better in the overall sense compared to Case I, which supports the result that we can ensure better generalization by choosing sample points satisfying the condition for reduction of computational cost.

Also looking at fig.6.7, comparing case III with the above two cases, we see that case III (optimal sampling) gives a better result that Case I (non-optimal sampling) but is worse than Case II (optimal sampling). Hence, this proves the fact that sampling which satisfies the Theorem 5.2 is not unique and even within this group, we can find samples which generalize better than the other (refer Theorem 6.1). However, all such next sample which satisfy Theorem 5.2 necessarily give equal or better performance than the sample which does not satisfy this result.

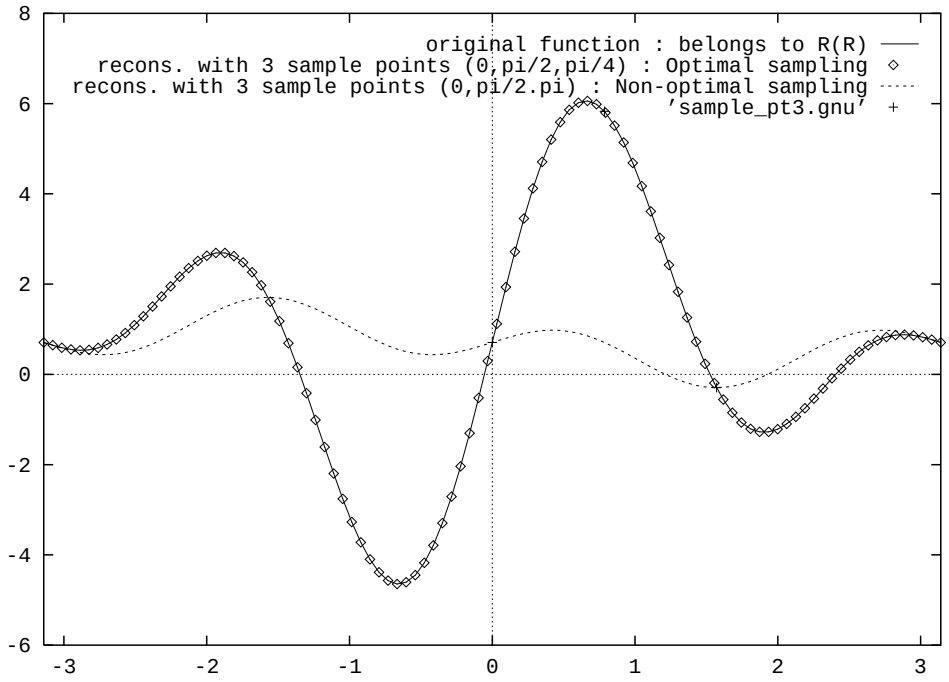

Figure 6.2: Simulation result for original function $f \in \mathcal{R}(R)$ using three sample points

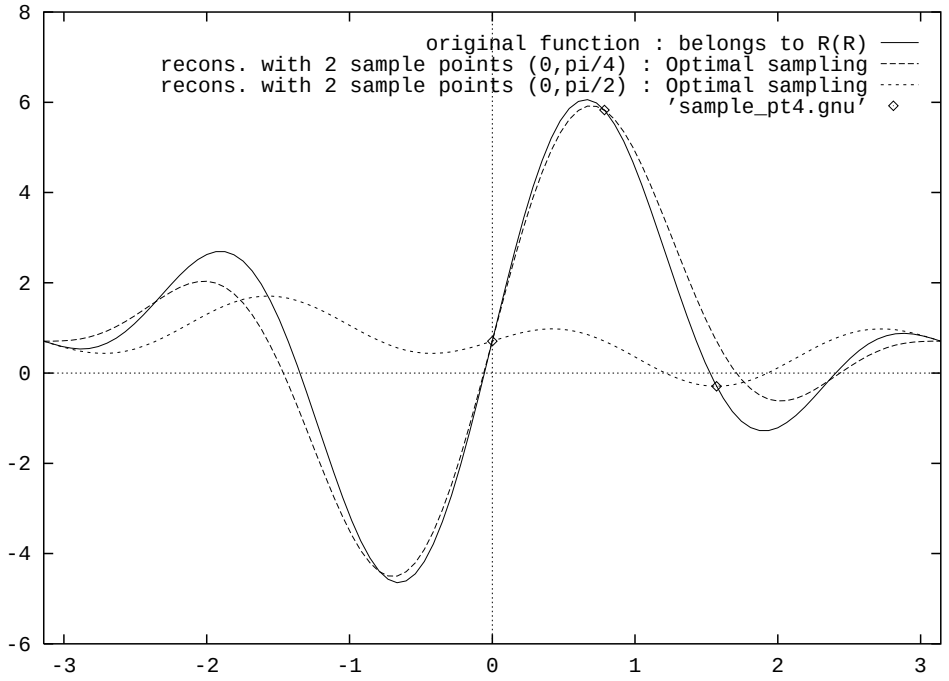

Figure 6.3: Simulation result for original function $f \in \mathcal{R}(R)$ using two sample points

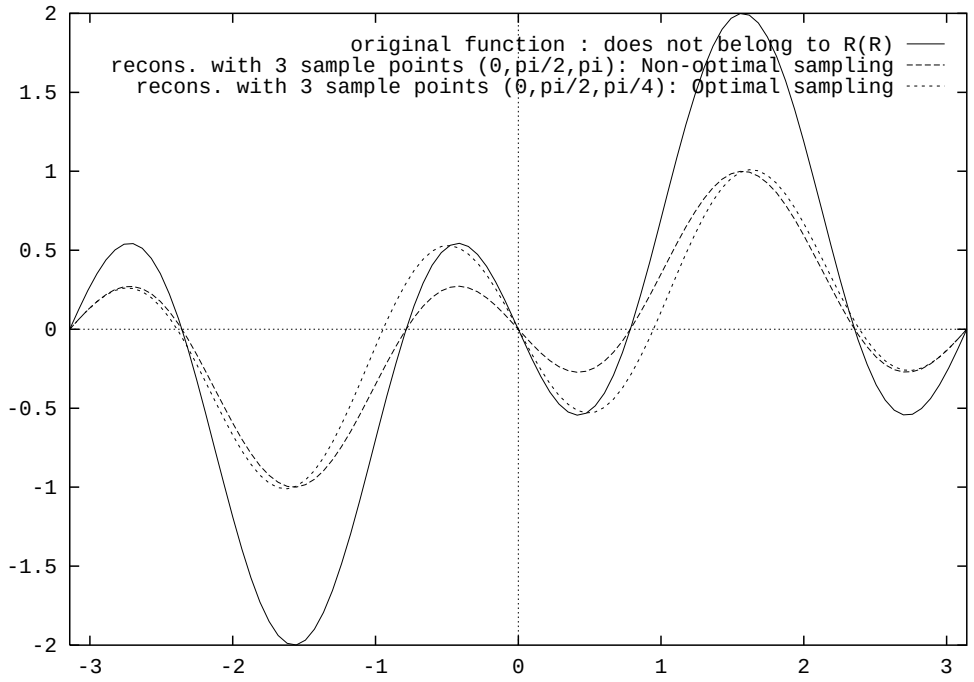

Figure 6.4: Simulation result for original function $f \notin \mathcal{R}(R)$ using three sample points

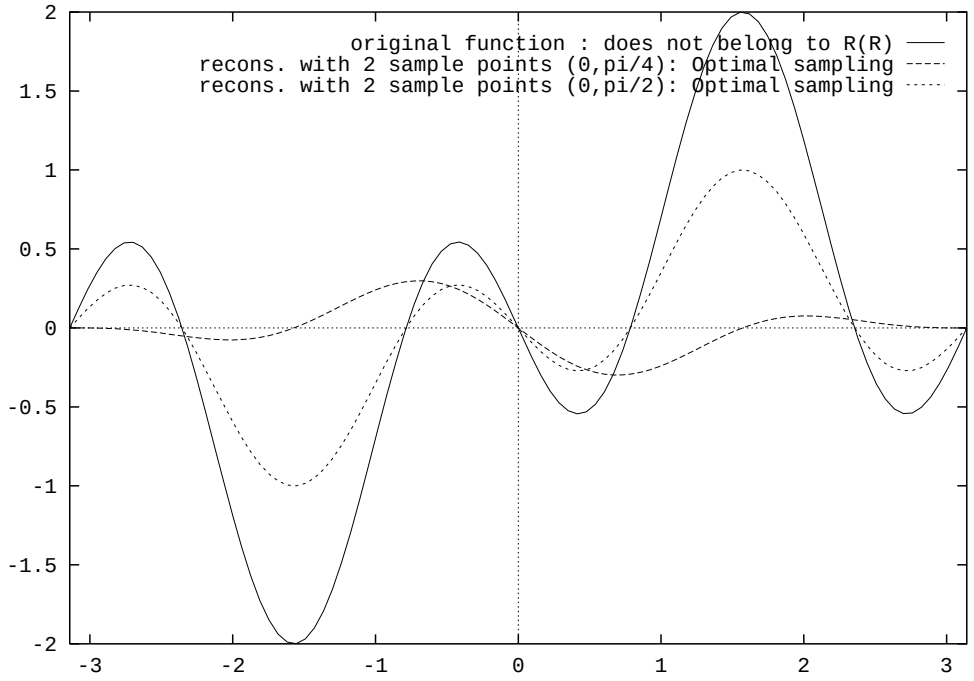

Figure 6.5: Simulation result for original function $f \notin \mathcal{R}(R)$ using two sample points

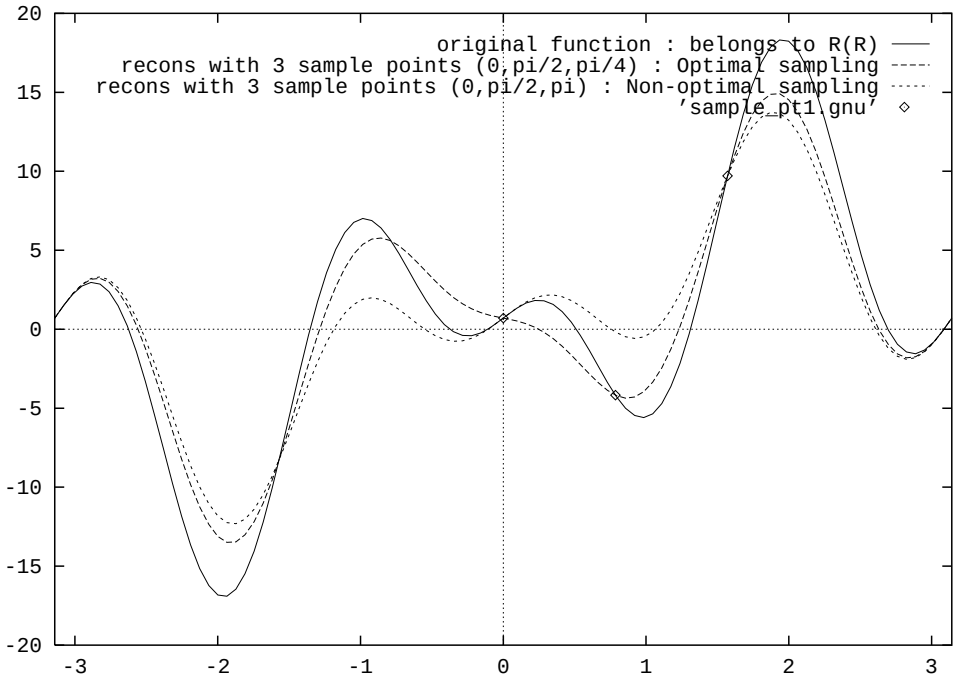

Figure 6.6: Simulation results for original function $f \in \mathcal{R}(R)$ using three sample points

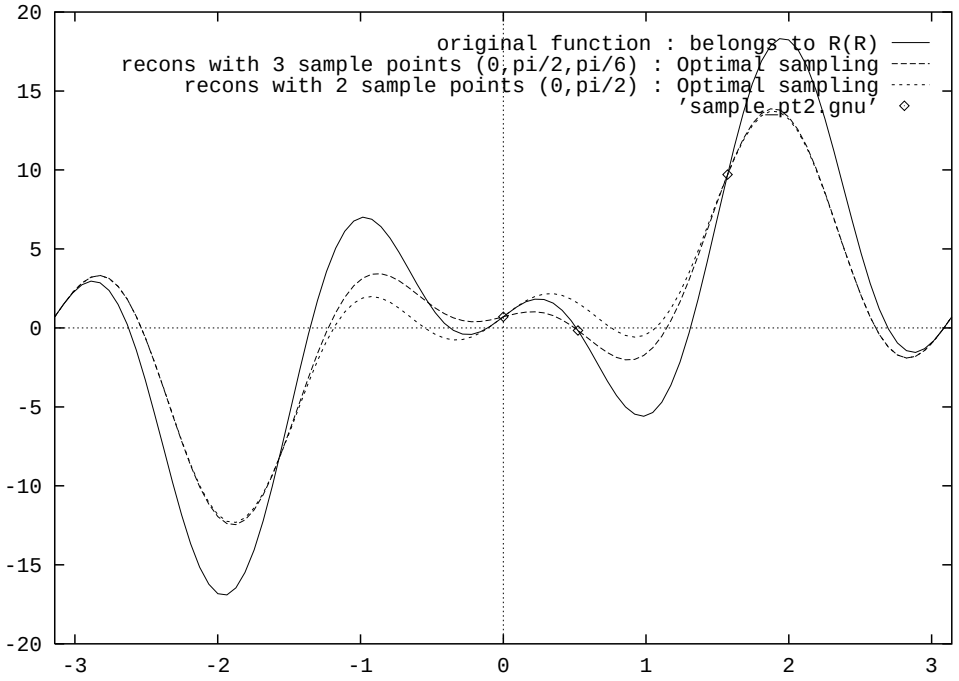

Figure 6.7: Simulation results for original function $f \in \mathcal{R}(R)$

Looking at case IV, which involves learning through only 2 samples which satisfy the optimality condition, we see that it gives the same performance as the learning with 3 sample points which does not satisfy the optimality condition (case I). Hence, this supports the idea that samples which do not satisfy the optimality criterion are redundant and does not improve the generalization abilty of the net.

Analysis of Example (A) and (B)

Example (A) and (B) is a simulation using 4 dimensional original function space with (A) learning a function within the function subspace $\mathcal{R}(R)$ while (B) learning one outside the subspace $\mathcal{R}(R)$. Case I an d II for either examples considers 3 sample points with Case II satisfying the optimality criterion whereas Case I not satisfying the criterion. Case III and IV both involve 2 sample points satisfying the optimality criterion.

Looking at the results (refer fig.6.2), we see that as shown in Example (C), the learning in case of Case II is better in the overall sense compared to Case I, again supporting the result that the generalization is better if the optimality condition is satisfied.

Looking at fig. 6.3, we see that sampling with 2 points, both satisfying the optimality criterion provides learned functions with different degrees of generalization. However, comparing case Case I and Case III, as we an see, the learned functions are the same. Hence, this also is in support of the result of Lemma 3.1 according to which increased sample in case of noiseless Wiener learning always gives better or same generalization.

We also carried out simulations for the functions outside the subspace $\mathcal{R}(R)$ in Example (C). However, simulation results for the same do not, in general, comply with the results we have obtained because the original function is asssumed to belong to $\mathcal{R}(R)$ due to the incorporation of the covariance of the function ensemble R. This is because the Wiener learning does not guarantee optimization for functions outside $\mathcal{R}(R)$.

6.6 Conclusion

sIn this chapter, we formulated the Wiener optimization criterion for incremental sampling. Solving this would yield the next best sample point in the context of maximizing generalizing ability in the Wiener sense. However, in this dissertation, we do not directly solve for this criterion. Instead, the sampling using result of Theorem 5.2 is compared with the goal of maximizing generalization. It is found that the goals of reducing computational cost and optimizing generalization are compatible with each other. Simulations result verify this observation as well.

Chapter 7

Conclusion

7.1 Results

In this dissertation, we formalized the concept of incremental learning and on the basis of that, formulated an efficient method for incorporating additive training data incrementally in real time. Furthermore, we devised a framework for choosing the training data for the neural networks(NNs) incrementally under specified optimality criterion.

In Chapter 2, we introduced a model of the neural networks [17] that was used here and went on to show how the 'learning' in NNs could be considered as a function approximation problem. Then, we compared the current model with the conventionally existing model and brought out an analogy between the two methods[22], proving that results obtained here can be directly incorporated for conventional weight adjusting NN architectures. Finally, we described three learning optimization criteria [14], [15], [16], [23] which overcame the shortcomings of the least squares estimation of Chapter 1 with regard to our learning model.

In Chapter 3, we have described the motivation for considering an incremental approach and formalized the concept of incremental learning. Since we used the Wiener criterion for optimizing the learning in our model, here, we extended this criterion to account for an incremental optimization.

Chapter 4 provided the mathematical concept of Moore-Penrose generalized inverse and some of its properties used later in the thesis. Then, we have provided a recursive computation of the generalized inverse of a special kind of matrix, which we will used later in our implementation of the incremental learning.

In Chapter 5, we dealt with one of the major aims of this dissertation and provided a method for incrementally computing the new learning operator X_{m+1} due to $m+1$ sample points by computing the change in the old learning operator, which is a function of the new sampling location x_{m+1}. Over and above, we deduced a sampling scheme for selecting the training data in order to further reduce the computational cost of the incremental algorithm.

Chapter 6 dealt with the problem of training data selection for improved generalization of the neural network. To this effect, a general introduction to the concept of information measure has been provided. Then, the Wiener criterion is redefined under the framework of incremental sampling to serve as the information measure optimizing functional. In this dissertation, we have not solved this optimization problem to derive the optimal sampling scheme from the generalization point of view. However, based on the result derived in Chapter 5, it is possible to divide the whole set of possible sample points into two categories. We were able to prove that the samples which improve the generalizing ability of the network always lie within the category of samples that reduces the compuational cost of the incremental algorithm. In other words, the aim of obtaining training samples which optimize computational cost and that which maximizes the generalizing ability of the network are compatible which each other. This result has also been verified through computer simulation of function approximation problems using the incremental Wiener optimization criterion.

7.2 Future objectives

As has been mentioned in Chapter 6, we have not yet solved the problem of determining the optimal training data selection from the generalization point of view. In continuing with this research approach, we would aim at completely solving the above mentioned problem.

Through the results of this present work, it has become possible to incorporate data incrementally and output intermediate learning results without waiting for the completion of the sampling process. This opens up a possiblity of using the information gained through the new sampling point and the intermediate learned results to adapt our sampling schemes for better performance. This is referred to in some fields as 'active learning'.

Acknowledgements

I would like to extend my heartfelt gratitude to Prof. Hidemitsu Ogawa, who supervised this work, for accepting me in his laboratory, for providing an excellent working environment and above all, for his valued guidance and encouragement despite his extremely busy schedule.

I am grateful to Dr. Itsuo Kumazawa, Assoc. Professor, for his constructive suggestions and advice. I also wish to thank Mr. Yukihiko Yamashita for the valuable discussions. I extends my thanks to Mrs. Kyoko Saitoh and Mrs. Shinagawa for their permanent kindness and help. I would also like to extend a special 'thanks' to all the students of Ogawa laboratory for the wonderful time, academically as well as in sharing those lighter moments, which went a long way in motivating this work.

I would like to express my gratitude to Mombusho, the Ministry of Education and Culture, Japan, for enabling me to come to Japan and providing the necessary financial support for this research. Last but certainly not the least, I would like to thank my parents and sister, who have been a constant source of inspiration throughout my life and have egged me on whenever I faltered.

Bibliography

[1] Amari,S. "Mathematical foundations of neurocomputing", Proceedings of the IEEE, vol.78, no.9, pp.1443-1463 (Sept. 1990) .

[2] Amari,S. "A universal theorem on learning curves", Neural Networks, vol.6, pp.161-166 (1993) .

[3] Albert,A. "Regression and the Moore-Penrose Pseudoinverse", Academic Press, New York and London, (1972) .

[4] Ben-Israel,A. and Greville,T.N.E "Generalised Inverses: Theory and Applications", John Wiley and Sons, New York, (1974) .

[5] Cline,R.E. "Representations for the generalized inverse of a partioned matrix", J. Soc. Indust. Appl. Math, vol.12, no.3, pp.588-601 (Sep. 1964) .

[6] Andrews,H.C. and Hunt,B.R. "Digital Image Restoration", Prentice-Hall, New Jersey, (1977) .

[7] Luttrell,S.P. "The use of transinformation in the design of data sampling schemes for inverse problems", Inverse Problems, vol.1, no.1, pp.199-218 (Feb. 1985) .

[8] Maeda,J. and Murata,K. "Restoration of band-limited images by an iterative regularized pseudoinverse method", J. Optical Society of America, vol.1, no.1, pp.28-34 (Jan. 1984) .

[9] MacKay,D.J.C. "Information based objective functions for active data selection", Neural Computation, vol.4, pp.590-604 (1992) .

[10] Ogawa,H. and Oja.E. "Projection filter, Wiener filter and Karhunen-Loeve subspaces in digital image restoration", J.Math. Anal. and Appl., vol.114, no.1, pp.37-51 (Feb. 1986) .

[11] Ogawa,H. "Optimum digital image restoration", in D.D. Majumdar (Ed.), Advances in Information Sciences and Technology: Pattern Recognition and Digital Techniques, Indian Statistical Istitute, Calcutta, vol.1, pp.290-297 (1984) .

[12] Ogawa,H. "A unified approach to generalized sampling theorems", Proceedings IEEE-IEICE-ASJ Int. Joint Conf. on ASSP, Tokyo, pp.1657-1660 (Apr. 7-11,1986) .

[13] Ogawa,H. "Projection filter regularization of ill-conditioned problem", SPIE Inverse problems in optics, vol.808, pp.189-196 (1987) .

[14] Ogawa,H. "Tutorial reviews: Image and signal restoration [1-4]", J. IEICE, Japan, vol.71, no.5-8, pp.491-497, 593-601, 739-748, 828-835 (1988) (In Japanese) .

[15] Ogawa,H. "Inverse problem and neural networks", Proceedings of IEICE 2nd Karuizawa Workshop on Circuits and Systems, pp.262-268 (1989) .

[16] Ogawa,H. "Neural network theory as an inverse problem", J. IEICE, Japan, vol.73, no.7, pp.690-695 (July 1990) .

[17] Ogawa,H. "Neural network learning, generalization and over-learning", Proc. ICIIPS'92 (Beijing), vol.2, pp.1-6 (Oct.30 - Nov.1,1992) .

[18] Strang,G. "Introduction to Applied Mathematics", Wellesley-Cambridge Press, Wellesley, (1986) .

[19] Valiant,L.G. "A theory of learnable", Communication of the ACM, vol.27, no.11, pp.1134-1142 (Nov. 1984) .

[20] Wann,M., Hediger,T. and Greenbaun,N.N. "The influence of training sets on generalization in feed-forward neural networks", Proc.Int. Joint Conf. on Neural Networks (San Diego), vol.3, pp.3.137-3.142 (June 1990) .

[21] "Functional approximation by feedforwar networks : A least squares approach to generalization", IEEE trans. on Neural Networks, vol.5, no.3, pp.363-371 (May 1994) .

[22] Yamasaki,K. "Generalizing ability and over-learning in the supervised learning of neural networks", Ph.D. thesis, Tokyo Institute of Technology, Japan, (1993) (In Japanese) .

[23] Yamashita,Y. and Ogawa,H. "Properties of averaged projection filter for image restoration ", J.IEICE,Japan, vol.J74-D-II, no.2, pp.142-149 (Feb. 1991) (In Japanese) .

[24] Yamashita,Y. and Ogawa,H. "Optimum image restoration filters and generalized inverses of operators", J.IEICE,Japan, vol.J75-D-II, no.5, pp.883-889 (May 1992) (In Japanese) .

[25] Zhang,B. T. "An incremental learning algorithm that optimizes network size and sample size in one trial", Proc. of ICNN'94(Orlando, Florida), pp.215-220 (June 28 - July 2, 1994) .

[26] Zlobec,S. "An explicit form of the Moore-Penrose inverse of an arbitrary complex matrix", SIAM Review, vol.12, no.1, (Jan. 1970) pp.132-134 .