Dynamic Reconstruction of Sea Clutter

Lim Teck Por

(B. Eng. (Hons), NUS)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2003

ACKNOWLEDGEMENT

I would like to thank God, my parents, family and friends for their support, encouragement, etc.

I am very grateful to Dr Sadasivan Puthusserypady for having faith in me to take up this project, which had been very enriching. Also, his insistence that I should take up a module on random signals (EE5306) benefited me greatly. I am highly thankful to Dr Kenneth Ong for lending me the Uninterruptible Power Supply (UPS), without which some simulations would not have been completed. Special thanks to Ms Agnes Ng and Ms Elaine Chua for providing access to some Pentium 4 PCs, which were crucial for the larger simulations.

Many thanks to Dr R. Hegger for kindly supplying recompiled TISEAN binaries that are faster than those on the TISEAN website. Thanks also to Dr D. G. Stork for supplying Figures 3.2, 3.3 and 3.4 from "Pattern Classification" by Duda, Hart and Stork. Thanks also go to the following researchers who had kindly taken the trouble to photocopy and mail papers which were unavailable in Singapore, which were quite necessary in order to form a good literature review: J. D. Farmer, A. T. Jessup, L. Kuncheva, D. Lowe and A. Wolf.

This work owes much to all who have offered negative feedback (including the examiners). Dr Sadasivan Puthusserypady, Gao Zhengfeng, Du Lin, Tey Eng Tian and Luo Huaien had commented on the thesis; Ajeesh P. Kurian and Budi Juswardy had provided assistance and discussions.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

AWGN	Additive White Gaussian Noise
DBF	Diagonal Basis Functions
EBF	Elliptical Basis Functions
EM	Expectation Maximization
FCM	Fuzzy *c*-means clustering
FLOPS	Floating-point Operations Per Second
GE	Generalization Error
HOP	Horizon of Predictability
int(x)	Integer part of x
KE	Kolmogorov Entropy
kRBF	RBF with *k*-means clustering to organize the centers
LOO	Leave-One-Out
MEM	Maximum Entropy Method
MLP	Multilayer Perceptron
NaN	Not a Number
NMSE	Normalized Mean Squared Error
OSL	Oseledec matrix
pdf	Probability Density Function
PRF	Pulse Repetition Frequency
RBF	Radial Basis Functions
SNR	Signal to Noise Ratio
SOM	Self Organizing Map
SVD	Singular Value Decomposition
sup	supremum

LIST OF SYMBOLS

b_{ij}	Membership of the *i*-th point in the *j*-th cluster
d_E	Embedding dimension (Section 2.3)
D_0	Box-counting dimension (Section 2.4.1)
D_2	Correlation dimension (Section 2.4.2)
D_{KY}	Kaplan-Yorke dimension (Section 2.4.4)
γ	Regularization parameter controlling smoothness of function approximation
γ_c	Regularization parameter for covariance matrix for Babuska's clustering method (Section 3.3)
λ	Lyapunov exponent (Section 2.4.3)
M	Number of centers in RBF after clustering
M_c	Number of centers in RBF before clustering
$\mathbf{\Psi}(n)$	Embedding vector $\mathbf{\Psi}(n) = \big(y(n), \quad y(n-\tau), \quad \cdots, \quad y\big(n-(d_E-1)\tau\big)\big)$
$\boldsymbol{\psi}(n)$	Input vector of RBF $\boldsymbol{\psi}(n) = \big(y(n), \quad y(n-1), \quad \cdots \quad y(n-(d_E\tau-1))\big)$
ρ_{ij}	Distance of the *i*-th point from the *j*-th center
$s^2(\mathbf{x})$	Sample variance of elements of vector **x**, as in Eq. (4.3).
τ	Embedding delay (Section 2.2)
$u(\bullet)$	Step function

SUMMARY

This thesis explores issues related to the modelling of sea clutter data using Radial Basis Function (RBF) networks and variants. Previous work had shown that sea clutter may be chaotic, and thus amenable to nonlinear time series analysis. Because RBF networks possess the property of universal approximation, it is possible to use them to model sea clutter data. This is a noisy, nonlinear problem; a large RBF network is usually required.

The prescriptions for choosing embedding delay are put on a sound theoretical basis. The standard procedure for estimating embedding dimension is improved. Clipping is introduced, as a simple, yet effective way to stabilize iterated predictions. A method is devised to speed up cross validation, which applies to variants of the Radial Basis Function (RBF) utilizing clustering techniques. Error variance is used for selecting models, rather than mean squared error. The RBF architecture is revised to account for empty clusters. A possible explanation is found for the puzzling phenomenon of empty clusters. It is suggested that non-deterministic behaviour of the clustering stage could affect RBF performance. Several types of data driven, non-radial basis functions are introduced, which may require less centers, thereby alleviating the curse of dimensionality. This stemmed from a desire to find a compromise between coping with high dimensionality, and yet using all available information as effectively as possible. Regularization is extended to non-radial basis functions.

The improved understanding and procedures were applied to model sea clutter using iterated prediction. One spin-off is that the significant computational savings from speeding up cross validation may tip the balance and encourage more applications to

employ the RBF, rather than the Multilayer Perceptron (MLP). It may also discourage certain regularization techniques which cannot be accelerated.

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

1.1 Motivation

Radar echo from the surface of sea is called sea clutter. The detection of small surface maritime targets by radar is limited by the presence of sea clutter. At low grazing angles (angle between sea surface and radar signal, see Figure 1.1.) and close to shore, large amplitude echoes (sea spikes) may cause increased false alarm rates. This requires the detection threshold to be raised and thus limits the size of detectable targets.

So far, it had been difficult to establish reliable relationships between sea clutter measurements and the environmental factors that determine the sea conditions [1]. It is apparent that improved understanding of sea clutter would result in improved radar detection. According to Haykin [2], a nonlinear predictive model could be used to cancel out sea clutter. Cancelling out the clutter helps to improve detection of small targets on the surface of the sea.

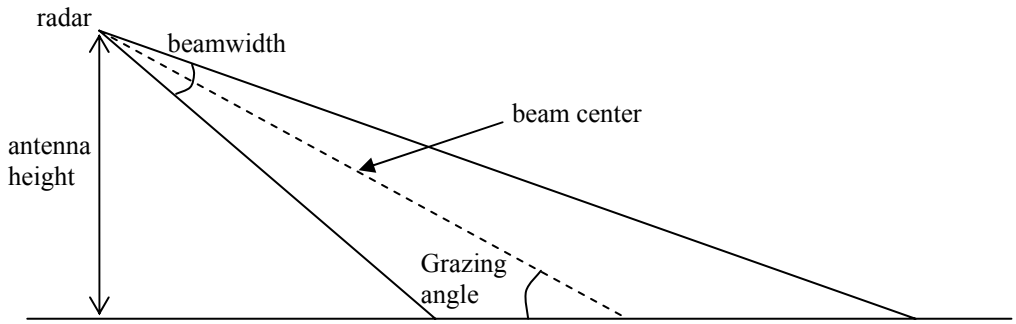

Figure 1.1 Illustration of beamwidth and grazing angle.

1.2 Literature Review

A model that is used to describe sea clutter at long radar wavelengths (High Frequency (HF) and Very High Frequency (VHF)) is Bragg scatter [1]. This is similar to the Bragg scattering observed in X-ray diffraction of crystals. A rough sea surface can be described by its vertical displacement from the mean, with a corresponding Fourier spectrum. Scattering from the sea surface can be characterized as scattering from a particular component of the surface spectrum resonant with radar frequency, resulting in constructive interference. The major scattering effect is due to the resonant component, and the other components of the spectrum can be neglected.

At higher frequencies, such as X-band (wavelength ≈ 3cm), the sea surface is often modelled as a composite surface with two scales of roughness (composite surface model). The resonant water waves of the classical Bragg model that might contribute to radar scatter have wavelengths of the order of centimetres. These short water waves (capillary waves) are said to ride on the higher amplitude long waves (gravity waves). Gravity waves are so named because their velocity of propagation is determined primarily by gravity. Capillary waves are small waves (less than about 1.73cm); their velocity is determined mainly by the surface tension of water [1], the velocity of which is determined mainly by the surface tension of water. Wetzel [3] noted that there are unresolved issues with the composite surface model, such as the assumption that sea surface displacements are small compared to the radar wavelength.

Because of the highly variable nature of clutter echoes, it is often described by probability distributions. Except for the Rayleigh distribution, there is no physical basis for the use of these distributions [1]. The most general clutter model at this time

is the Rayleigh mixture model (see Appendix A); it includes the K-distribution and Weibull distribution as special cases [4].

Note that the K-distribution and other compound distributions assume that there exists a large number of independent scatterers (see Appendix A). Despite the algebraic virtuosity of the derivations, this assumption may be questioned, considering that factors like wind velocity, temperature, etc, are approximately constant in a patch of sea surface. Furthermore, waves typically do not travel over the sea surface in completely random directions.

Also, there is the problem that the various statistical models are used because they fit some experimental data, and so they are not necessarily based on physical mechanisms [4]. Another problem is that a lot of data is required for calculating the higher moments because the long tails are problematic [5, 6]. This encourages one to consider possible alternatives. In the past decade, Haykin *et al.* had published a stream of research findings indicating that sea clutter may be chaotic [7-13]. Furthermore, this had also spawned a stream of research which applied chaos theory to sea clutter, of which [2, 14-25] are representative.

Besides the work in Ref. [13], there are also some independent results and theory, which may support the hypothesis that sea clutter may be chaotic:

- In Ref. [26, 27], it was shown that ocean waves exhibit some chaotic properties.
- At low grazing angle, sea clutter is dominated by sea spikes [1]. Churyumov and Kravtsov [28] showed that breaking waves are responsible for sea spikes. Jessup *et al.* [29] showed that the frequency of sea spikes was related to the Reynolds

number (an important quantity associated with turbulence in fluid dynamics). Hence, there may be a relationship between sea clutter and turbulence.

- Researchers [30-32] have shown that it is possible to model two dimensional fluid flows with chaos theory. The problem is that they have not extended their models to higher dimensions.
- A 5 degree of freedom chaotic model had been suggested by Lorenz [33, 34] for modelling large scale ocean models. This may possibly be useful for modelling the ocean, because Abarbanel *et al.* [35] had shown that some ocean measurements have an observed embedding dimension of 5.

On the other hand, in recent years, there had been some dissenting voices [36-38]. Gao and Yao [39] suggest that sea clutter is multifractal (see Glossary), but not chaotic. It may be reasonable to enquire if the chaotic hypothesis is also another curve fitting exercise, this time with respect to multi-dimensional manifolds.

Hence, it would be interesting to see if it is possible to model the dynamics of the sea clutter with a neural network. If iterated prediction of the network produces a sequence with similar properties as compared to the original data, then dynamic reconstruction has succeeded. Previous research, as in Ref. [40], had only examined the chaotic properties of successful reconstructions. Examining the failed reconstructions, instead of ignoring them, may yield some insights.

1.3 Contributions of this Thesis

This work may be of interest to researchers who are working in the areas of chaos and/or neural networks. The following contributions are briefly listed:

- A sound theoretical basis had been put forward to explain the rules for choosing embedding delay, which had previously been prescriptive (Section 2.2).
- Instead of using one algorithm for estimating embedding dimension, 2 algorithms are used; this is useful for double checking (Section 2.3).
- A method to speed up *k*-fold cross validation is proposed (Section 3.6.5).
- The standard architecture of the RBF is revised to include the possibility that the number of centers may be unequal to the number of weights in the linear layer, due to empty clusters (Section 3.2).
- A possible explanation is found for the puzzling phenomenon of empty clusters, which occasionally occur (Section 3.3.2). It is suggested that the non-deterministic behaviour of the clustering stage could sometimes affect the RBF (Section 3.3.1).
- Several types of data driven, non-radial basis functions are introduced (Section 3.4.2). Regularization is extended to non-radial basis functions (Section 3.6.4).
- It was suggested that instead of dealing with both real and complex components of the sea clutter signal, it may be sufficient to choose one, if they are related by the Hilbert transform (Section 4.2).
- It is suggested that sacrificing the bias in resolving the bias-variance dilemma may be useful in the context of dynamic reconstruction (Section 5.2).
- Alternative formulations of Generalization Error (GE) are given, *i.e.* voting instead of averaging (Section 5.1.2) and the use of variance instead of mean squared error for model selection (Section 5.2).
- It is demonstrated that sequences generated by kRBF models selected using error variance (Figure 5.17) can result in better dynamic reconstructions than kRBF models selected using mean squared error (Figure 5.12).

- Iterated prediction is performed using many different unique candidate starting points, unlike existing literature. Initializing iterated prediction with estimated values instead of the test set is suggested (Section 5.4.1). Clipping is introduced, as a simple, yet effective way to stabilize iterated predictions (Section 5.4.2).

1.4 Overview of the Thesis

Chapter 2 introduces some methods used in experimental chaos; Chapter 3 introduces the theory of RBF networks and variants. Chapter 4 outlines attempts to characterize the data, prior to running the simulations; the simulations results are recorded in Chapter 5. Conclusions and future work are discussed in Chapter 6. The appendices are provided for the convenience of the readers; some derivations may take a long time to produce without mathematical handbooks. The Glossary enables readers to check up technical terms; it owes much to Ref. [41] and also the glossaries of Ref. [42, 43].

CHAPTER 2

Phase Space Reconstruction

The set of all possible states of a system is called the phase space or state space of the system. Phase space reconstruction is defined as the identification of a mapping that provides a model for an unknown dynamical system. It provides a practical means for making physical sense of an experimental time series without knowledge of the underlying dynamics of the system. The workhorse of phase space reconstruction is Taken's Delay Embedding Theorem. Concepts relevant to phase space reconstruction are introduced in this chapter. Modifications to existing concepts and procedures are also discussed.

2.1 Taken's Delay Embedding Theorem

Essentially, the main idea behind Taken's Delay Embedding Theorem is that it is possible to reconstruct state space from a time series consisting of measurements of a chaotic system. Consider a time series with N^{total} samples; each n-th measurement is given as $y(n)$ (sampling rate is typically fixed). The delayed samples of the time series $\{y(n)\}_{n=1}^{N^{total}}$ are formed into the embedding vector $\mathbf{\Psi}(n)$:

$$\mathbf{\Psi}(n) \triangleq \left(y(n), \quad y(n-\tau), \quad \cdots, \quad y\left(n-(d_E-1)\tau\right)\right), \tag{2.1}$$

where the embedding dimension is $d_E \in \mathbb{Z}^+$ and the embedding delay is $\tau \in \mathbb{Z}^+$. From a time series with N^{total} samples, $N^{\mathbf{\Psi}} \triangleq N^{total} - (d_E - 1)\tau$ embedding vectors (each of dimension d_E) can be formed. These embedding vectors form a reconstructed attractor.

The reconstructed attractor preserves the topological properties of the original attractor, and hence the chaotic invariants (see Glossary or Section 2.4) estimated from the reconstructed attractor are equivalent to the chaotic invariants of the attractor itself. Note that normalizing the time series only scales the attractor, which does not affect the chaotic invariants.

Consider the Lorenz system [44] as the archetypical chaotic system; it is described by a set of differential equations:

$$\begin{aligned} \dot{x} &= \sigma(y - x) \\ \dot{y} &= -xz + rx - y \\ \dot{z} &= xy - bz, \end{aligned} \tag{2.2}$$

where $\sigma = 16$, $r = 45.92$, $b = 4$. The initial conditions $x = 1$, $y = 1$, $z = 1$ are fed into the Runge-Kutta ODE solver in MATLAB© to produce a time series for each component (x, y and z). Unless otherwise stated, the Lorenz data used throughout this work refers to the x-component of the Lorenz system (see Figure 2.1). The first 20,000 points are discarded to remove the transients. Then the data is pre-processed to have 0 mean and variance 1.

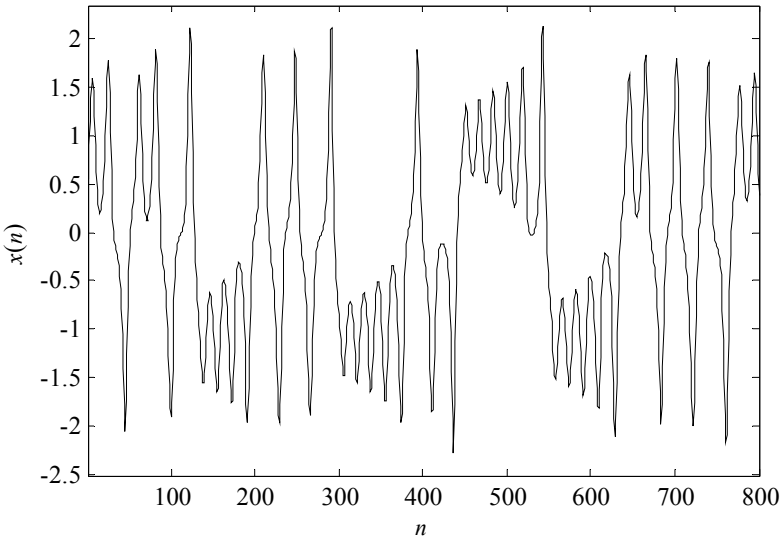

Figure 2.1 Plot of x-component of Lorenz time series.

Figure 2.2 illustrates the famous Lorenz attractor plotted from the x, y and z components of the Lorenz ODE defined by Eq. (2.2) and numerically solved using the Runge-Kutta method of order 4. Typically, ergodicity is assumed [45], *i.e.* time averages are the same as state space averages. In such a case, transients can be ignored, and it is only necessary to consider the long term behaviour of the system, *i.e.* the attractors. Observe that the Lorenz attractor is symmetrical, with 2 "lobes", and has a "fractal" structure.

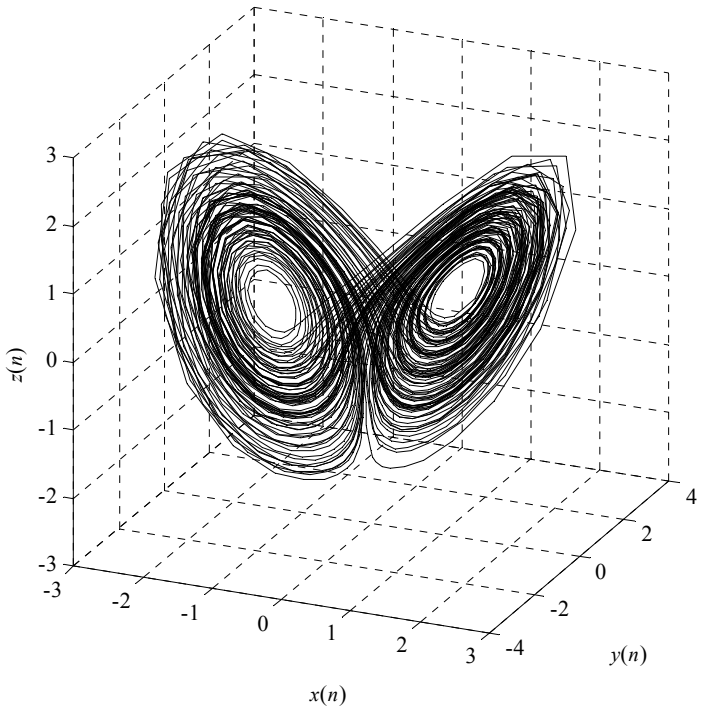

Figure 2.2 Three-dimensional plot of Lorenz attractor.

Figure 2.3 illustrates an attractor reconstructed from the x-component of the data, using Taken's Embedding Theorem. The reconstructed attractor looks like a warped version of Figure 2.2. One way to verify that the reconstruction is successful is to check that chaotic invariants (see Glossary or Section 2.4) of the reconstructed attractor match those of the original attractor.

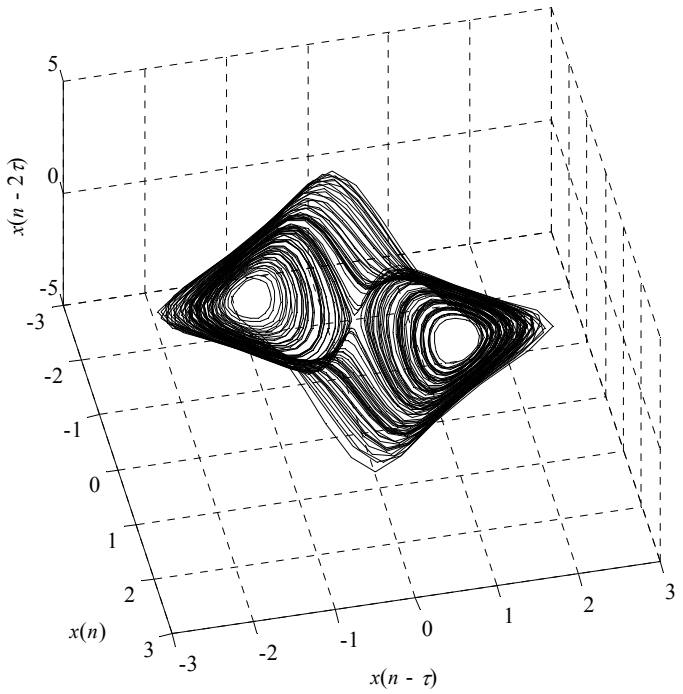

Figure 2.3 Three-dimensional plot of Lorenz attractor reconstructed from *x*-component of the data using Taken's Embedding Theorem.

It should be noted that Taken's Embedding Theorem only gives sufficient conditions, not necessary ones [46]. It applies only to generic systems, and there are examples where measuring a variable from a dynamical system will lead to a distorted phase space reconstruction [47]. For example, the *z*-component of the Lorenz system does not distinguish the 2 unstable foci associated with the 2 "lobes" of the attractor, due to the underlying symmetries of the Lorenz model [48]. The phase space reconstruction in Figure 2.4 is topologically different from that in Figure 2.2, since there is only one "lobe", rather than two.

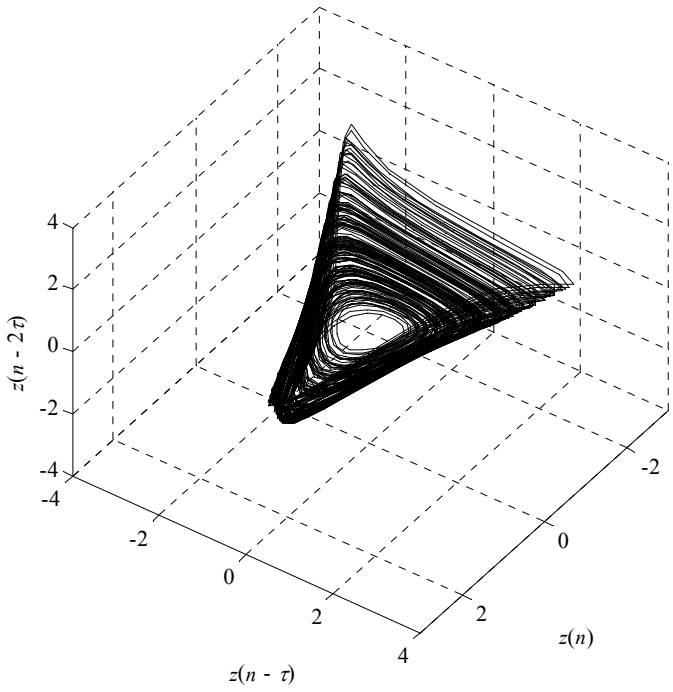

Figure 2.4 Three-dimensional plot of Lorenz attractor reconstructed from z-component of the data using Taken's Embedding Theorem.

2.2 Embedding Delay

Theoretically, for an infinite amount of infinitely accurate data, the choice of the lag, τ, is unimportant [43]. However, in practice, the quality of phase space reconstruction depends on the choice of τ. The prescription of Ref. [49] is to choose τ such that it corresponds to the first minimum of the of the mutual information between a time series $Y = \{y(n)\}_{n=1}^{N^{total}-\tau}$ and a delayed version $Y_\tau = \{y(n)\}_{n=\tau+1}^{N^{total}}$.

Treating Y as a discrete random variable, its entropy is defined as:

$$H(Y) \triangleq -\sum_{y \in \mathcal{Y}} p(y) \log\big(p(y)\big), \tag{2.3}$$

where $\mathcal{Y}$ is the alphabet (the set of possible symbols) of Y and $p(y) = p\{Y = y\}$, [50]. Mutual information between Y and Y_τ can be defined as

$$I(Y;Y_\tau) \triangleq H(Y) + H(Y_\tau) - H(Y,Y_\tau), \tag{2.4}$$

and the discrete joint entropy, $H(Y,Y_\tau)$, is defined as

$$H(Y,Y_\tau) \triangleq -\sum_{y\in\mathcal{Y}}\sum_{y\in\mathcal{Y}_\tau} p(y,y_\tau)\log\big(p(y,y_\tau)\big), \tag{2.5}$$

where $\mathcal{Y}_\tau$ is the alphabet of Y_τ.

The discrete formulation is used in the chaos literature; Fraser and Swinney [49] argue that the continuous case results in entropies which are coordinate dependent. Note that $I(Y;Y_\tau)$ is often called the Average Mutual Information in the chaos literature, as in Ref. [43].

It should be noted that if the time series is ergodic, and sufficiently long, then it is safe to assume that

$$H(Y) \approx H(Y_\tau), \tag{2.6}$$

and hence $H(Y_\tau)$ is essentially independent of the choice of τ. Finding the value of τ which results in the minimum value of $I(Y;Y_\tau)$ can be expressed as $\arg\min_\tau I(Y;Y_\tau)$. Substituting the approximation of (2.6) into Eq. (2.4), we get

$$\arg\min_\tau\big(H(Y) + H(Y_\tau) - H(Y,Y_\tau)\big) \approx \arg\min_\tau\big(-H(Y,Y_\tau)\big), \tag{2.7}$$

which implies that rather than finding the first minimum of the mutual information, it is sufficient to find the first minimum of $-H(Y,Y_\tau)$. This is in turn equivalent to searching for the first maximum of the joint entropy. This means that it is not necessary to calculate $H(Y)$ or $H(Y_\tau)$ for any value of τ. However, the

computational savings induced are relatively insignificant, since the computational complexity of $I(Y;Y_\tau)$ is dominated by the computational complexity of $H(Y,Y_\tau)$. Rather, the point is theoretical; finding the maximum of the joint entropy corresponds to the maximum entropy method (MEM) [51]. Essentially, MEM states that from a family of probability distributions, the probability distribution with the maximum entropy should be chosen, subject to the given constraints. MEM can be seen as a smoothness criterion [51].

For a concrete example, consider the Lorenz system. Figure 2.5 is a plot of the mutual information vs lag computed using `mutual.exe` from TISEAN [52]. The first local minima suggests that the embedding delay should be $\tau = 4$.

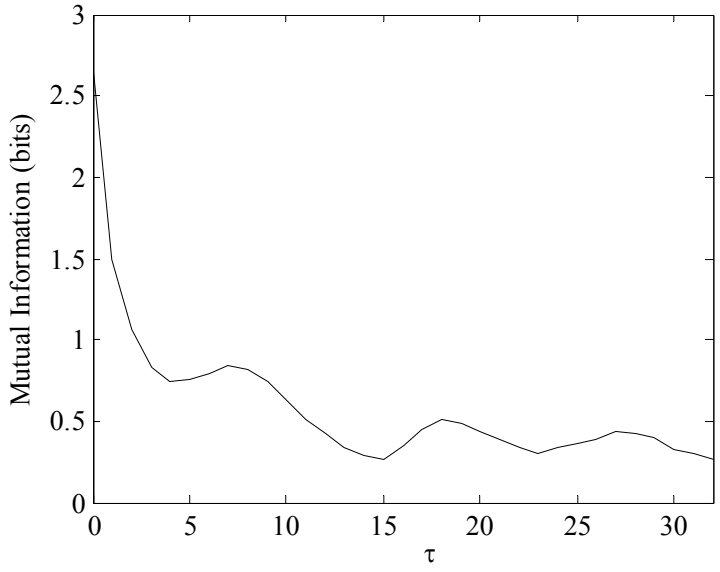

Figure 2.5 Plot of $I(Y;Y_\tau)$ vs τ for Lorenz system.

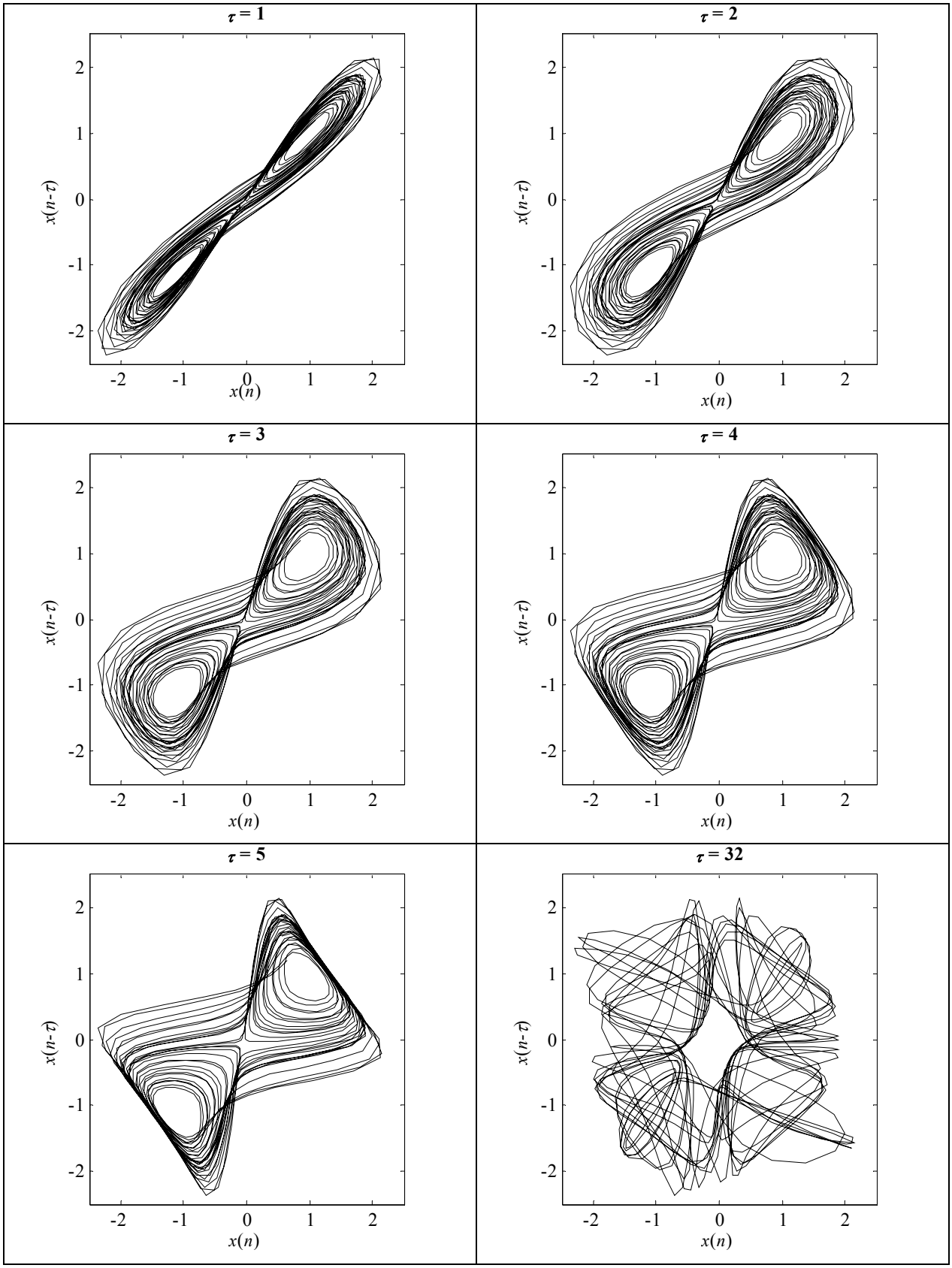

Figure 2.6 Reconstructed phase portraits of Lorenz data at varying time lags.

Figure 2.6 shows reconstructed phase portraits (plots of $x(n)$ vs $x(n - \tau)$ for various values of τ; it provides visual confirmation that near the first local minima of Figure 2.5, the attractor seems to be well unfolded. Hence, MEM provides a justification for the prescription to choose τ such that it corresponds to the first minimum of the plot of mutual information vs lag.

However, chaotic maps require a different way of choosing the embedding delay. As τ increases, the joint entropy increases and the reconstructed phase portrait progressively appears more "random". It had been suggested by Kalman in 1956 [53, 54], that chaotic maps are related to Markov chains. In such a case, it may be possible to model the relationship between Y, Y_τ and Y_{τ_2} using a Markov chain, where τ_2 is a time lag which is greater than τ. The data processing inequality [50] states that if $Y \to Y_\tau \to Y_{\tau_2}$ forms a Markov chain, then $I(Y;Y_\tau) \geq I(Y;Y_{\tau_2})$. This implies that the plot of mutual information (and also the plot of joint entropy) should be monotonically decreasing if the Markov chain model applies.

Consider the Ikeda map [43, 55], which is an example of a discrete map, defined by

$$z(n+1) = p + Bz(n)\exp\left\{i\left(\kappa - \frac{\alpha}{1+|z(n)|^2}\right)\right\}, \tag{2.8}$$

where p = 1.0, B = 0.9, κ = 0.4, and α = 6.0. Figure 2.7 is the corresponding plot of mutual information vs lag. Note that the plot is approximately monotonic; the minimum in the plot at large values of τ may be due to long range correlations, which ensure that Y_τ and Y_{τ_2} are not perfectly independent. Thus, the Markov Chain model is only approximate, but it appears to be a good model to use for discrete maps. Alternatively, the minimum may be an artefact due to the inadequacies of using a

histogram to represent a probability density function (pdf) in 2 dimensions or higher [56].

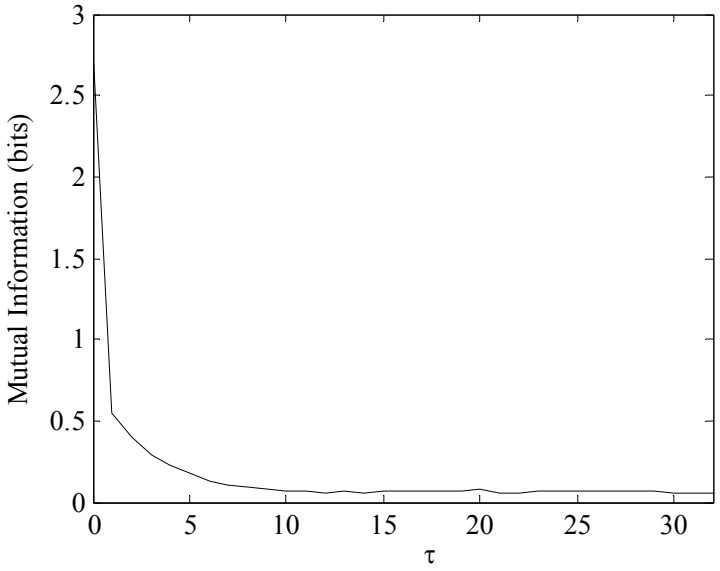

Figure 2.7 Plot of $I(Y;Y_\tau)$ vs τ for real component of Ikeda Map.

Figure 2.8 shows that when the lag is more than 1 or 2, the reconstructed phase portrait is too disordered. The prescription of Cao [57], which recommends a lag of 1 for discrete maps, makes more sense now. If the plot of mutual information decreases more or less monotonically; the reconstructed phase portrait would be too disordered at the minimum, if it exists.

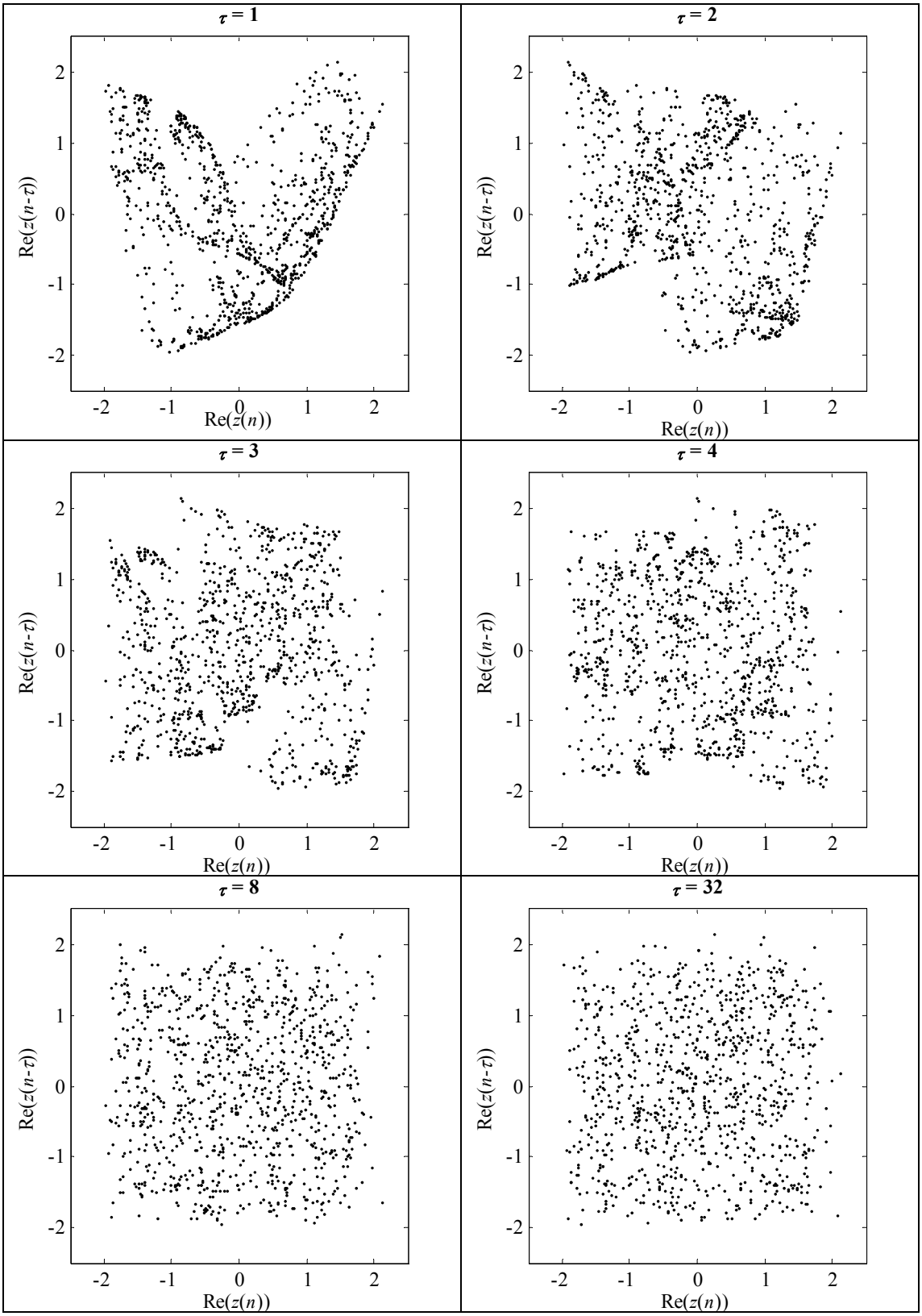

Figure 2.8 Reconstructed phase portraits of the real component of the Ikeda Map at varying time lags.

2.3 Embedding Dimension

The embedding dimension d_E determines the dimension of the reconstructed phase space that is required. In theory, the embedding dimension had been shown by Sauer *et al.* [58] to be

$$d_E > 2D_0, \tag{2.9}$$

where D_0 is the box-counting dimension (see Glossary or Section 2.4.1). However, note that inequality (2.9) provides a sufficient condition, but it may be possible to make do with a smaller dimension under particular circumstances. In fact, Ding *et al.* [59] showed that for the purpose of calculating correlation dimension (see Glossary or Section 2.4.2),

$$d_E = \text{int}(D_0) + 1 \tag{2.10}$$

suffices.

A practical algorithm to estimate d_E is the method of Global False Nearest Neighbours (GFNN) [60]. Essentially, the idea is that if the embedding dimension is too low, then the topology of the embedding may be distorted. An indication of the distortion is to estimate the number of points which are supposed to be far apart in phase space, and yet end up as neighbours because the embedding dimension is too low.

$$\begin{aligned} R_d^2(n, n_\eta) &= \left\| \mathbf{\Psi}_d(n) - \mathbf{\Psi}_d(n_\eta) \right\|^2 \\ &= \sum_{k=0}^{d-1} \left(y(n - k\tau) - y(n_\eta - k\tau) \right)^2 \end{aligned} \tag{2.11}$$

where $R_d^2(n, n_\eta)$ is the squared Euclidean distance between $\mathbf{\Psi}_d(n)$, the embedding vector at dimension d, and $\mathbf{\Psi}_d(n_\eta)$ is the nearest neighbour at dimension d (n_η is the index of the nearest neighbour). Then,

$$R_{d+1}^2(n,n_\eta) - R_d^2(n,n_\eta) = \left(y(n-d\tau) - y(n_\eta - d\tau)\right)^2 . \tag{2.12}$$

From Eq. (2.11) and Eq. (2.12), the following is obtained:

$$\sqrt{\frac{R_{d+1}^2(n,n_\eta) - R_d^2(n,n_\eta)}{R_d^2(n,n_\eta)}} = \frac{\left\| y(n-d\tau) - y(n_\eta - d\tau) \right\|}{\left\| \mathbf{\Psi}_d(n) - \mathbf{\Psi}_d(n_\eta) \right\|} . \tag{2.13}$$

The first criterion to determine a false nearest neighbour is that the distance is large when going from dimension d to $d + 1$, *i.e.*

$$\frac{\left\| y(n-d\tau) - y(n_\eta - d\tau) \right\|}{\left\| \mathbf{\Psi}_d(n) - \mathbf{\Psi}_d(n_\eta) \right\|} > R_{tol} . \tag{2.14}$$

The threshold R_{tol} is a constant such that $R_{tol} \geq 10$ [60] or $R_{tol} \approx 15$ [43].

The second criterion is

$$\frac{R_{d+1}(n,n_\eta)}{R_A} > A_{tol} \tag{2.15}$$

where R_A^2 is the sample variance of the time series and $A_{tol} \approx 2$ is an arbitrary threshold. If either criterion is true, a false nearest neighbour is declared. The dimension d_E is where the percentage of false nearest neighbours plateau off.

As early as 1995, researchers had reported flaws in the original algorithm [61]. One problem is that the number of false neighbours is underestimated when R_{tol} is large. Hegger and Kantz [62] suggested that rather than using a fixed value, R_{tol} should be pegged to the maximal Lyapunov exponent, λ_1 (see Glossary or Section 2.4.3), and the time delay:

$$R_{tol} > e^{\lambda_1 \tau} . \tag{2.16}$$

Also, pairs which are too far away are not really false neighbours, but are merely inappropriate candidates. Hence, points where

$$\frac{R_{d+1}(n, n_\eta)}{R_A} > \frac{1}{R_{tol}} \tag{2.17}$$

are disregarded.

Another problem that Hegger and Kantz [62] mentioned, was that with insufficient data and large *d*, the first criterion, (2.14), introduces false neighbours even for deterministic systems. According to Aggarwal *et al.* [63], there is poor discrimination between different neighbouring points as dimension increases. Hence, this problem cannot be solved by any false nearest neighbour method. The best solution is to have sufficient data.

The improved implementation of GFNN [52, 62] (`false_nearest.exe` in TISEAN) is used throughout this work. Additive White Gaussian Noise (AWGN) is added to Lorenz data, for various values of SNR. In this work, SNR refers to the ratio of signal power to noise power. Figure 2.9 shows that the performance of GFNN degrades gracefully as SNR decreases, except for low SNR (0dB and -10dB), where the GFNN curves are no longer monotonic. In fact, there is a sudden increase in the percent of false nearest neighbours at dimension 5. This sudden increase will not affect the algorithm adversely in practice, because at 0dB and -10dB, the noise level is so high that chaotic signal processing is meaningless anyway. Besides, it serves to differentiate low dimensional signals from high dimensional signals, such as noise.

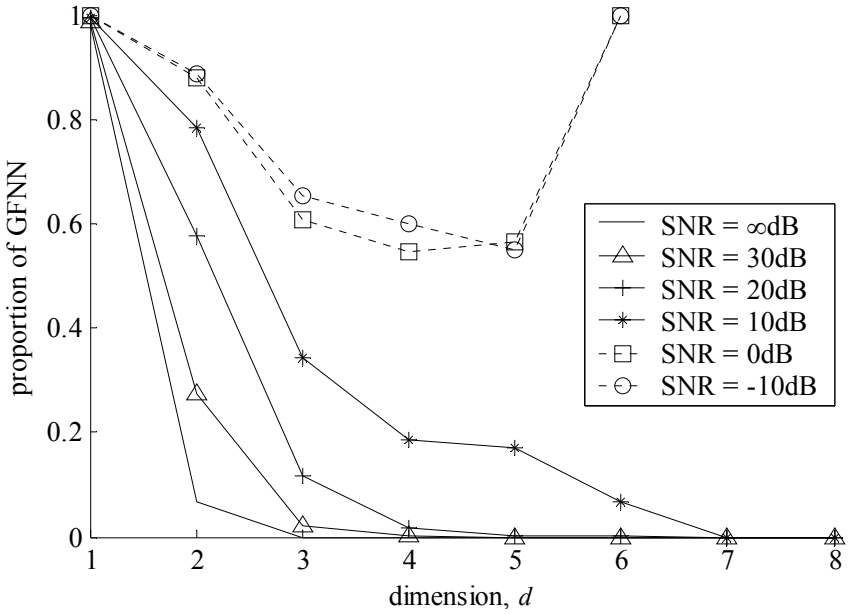

Figure 2.9 Performance of GFNN with different SNR levels for Lorenz data.

GFNN is further supplemented by Cao's method [57, 64]. Cao's method is similar to GFNN, except that it eliminates the use of arbitrary thresholds, and uses the maximum norm. Define a ratio

$$a_d(n) \triangleq \frac{\left\| \mathbf{\Psi}_{d+1}(n) - \mathbf{\Psi}_{d+1}(n_\eta) \right\|}{\left\| \mathbf{\Psi}_d(n) - \mathbf{\Psi}_d(n_\eta) \right\|}, \tag{2.18}$$

where $n \in [1, N - d\tau]$. The mean of this ratio is

$$E_d = \frac{1}{N - d\tau} \sum_{n=1}^{N-d\tau} a_d(n). \tag{2.19}$$

Based on Eq. (2.19), a ratio is defined:

$$E1_d \triangleq \frac{E_{d+1}}{E_d}. \tag{2.20}$$

$E1_d$ stops changing when $d \geq d_E$.

Unlike GFNN, the embedding dimension for Cao's method is decided not by the presence of a plateau, but by saturation of the curve, e.g., Figure 2.10.

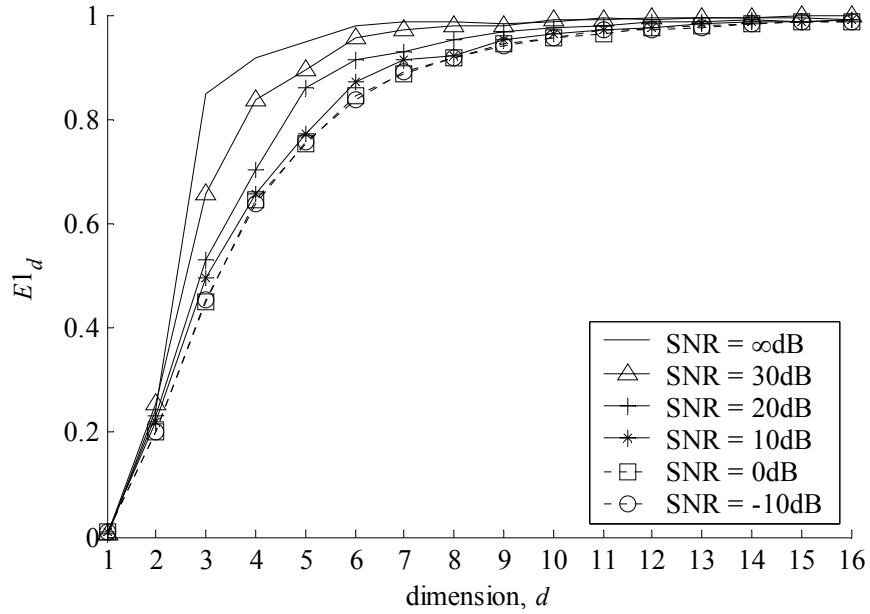

Figure 2.10 Plot of $E1_d$ with different SNR levels for Lorenz data.

Again, AWGN is added to the signal for various values of SNR. For no noise, the embedding dimension is 4. At lower SNR, the embedding dimension is 6 or 7. It can be seen that the algorithm also degrades gracefully with decreasing SNR, even when SNR is negative. However, at 0dB and -10dB, it is hard to discern the embedding dimension. The problem with GFNN is that for low SNR, the plateau is not easy to discern. On the other hand, Cao's method tends to give higher estimates, but these estimates are acceptable, considering condition (2.9). Cao's method can be used to verify the results obtained by GFNN; this may help the researcher to determine if the embedding dimension is wrong.

An interesting feature of Cao's method is that it also incorporates a test for determinism. Define

$$E_d^* \triangleq \frac{1}{N - d\tau} \sum_{n=1}^{N-d\tau} \left| y(n + d\tau) - y(n_\eta + d\tau) \right|. \tag{2.21}$$

Another ratio is defined, based on Eq. (2.21):

$$E2_d \triangleq \frac{E_{d+1}^*}{E_d^*}. \tag{2.22}$$

It appears that Eq. (2.22) could distinguish between random coloured noise and chaos. A stochastic process would produce a plot of $E2_d$ which fluctuates about 1, because E_d^* should be independent of d. The implementation of Cao's method (`cao.dll`) in TSTOOL [64] is used throughout this work.

2.4 Chaotic Invariants

Chaotic invariants are statistical quantities which can be used to characterize chaotic systems. The box-counting dimension (Section 2.4.1), correlation dimension (Section 2.4.2) and Kaplan-Yorke dimension (Section 2.4.4) quantify the dimensionality of the attractor. On the other hand, Lyapunov exponents (Section 2.4.3), Kolmogorov entropy (Section 2.4.5) and Horizon of Predictability (Section 2.4.6) quantify the dynamical aspects of the attractor.

Chaotic invariants are unchanged under smooth nonlinear changes of coordinate system. This invariance is important, because when measuring a variable, the recorded signal is often not the actual dynamical variable being characterized. For example, it might be an electrical signal from a temperature probe, though the actual variable of interest might be temperature. It is permissible to use the recorded values of the electrical signal to directly compute the chaotic invariants as long as the relationship between the electrical signal and the actual variable is one-to-one [65].

2.4.1 Box-counting Dimension

The box-counting dimension D_0 of a set U is defined as

$$D_0 \triangleq \lim_{\varepsilon \to 0^+} \sup \frac{\log N(\varepsilon)}{\log(1/\varepsilon)}, \tag{2.23}$$

where $N(\varepsilon)$ is the number of balls of diameter ε required to cover U, and the supremum (sup) is the least upper bound of a set. For sets such as points, line segments and surfaces, the box-counting dimension is 0, 1, and 2, respectively. Usually, D_0 is not an integer for chaotic attractors. For example, $D_0 \approx 2.06$ for the Lorenz attractor [66].

2.4.2 Correlation Dimension

Although the box-counting dimension is conceptually straight-forward, its application to actual data, especially for higher dimensional state spaces, is problematic [65]. The number of computations required for the box-counting procedure increases exponentially with the state space dimension. To provide a computationally simpler dimension, Grassberger and Procaccia [66] introduced a dimension based on the behaviour of the correlation sum. The correlation sum is defined as

$$C(r) \triangleq \frac{1}{N^{\boldsymbol{\Psi}}(N^{\boldsymbol{\Psi}}-1)} \sum_{n=1}^{N^{\boldsymbol{\Psi}}} \sum_{n_2=1, n \neq n_2}^{N^{\boldsymbol{\Psi}}} u\left(r - \left\|\boldsymbol{\Psi}(n) - \boldsymbol{\Psi}(n_2)\right\|\right), \tag{2.24}$$

where $r \in \mathbb{R}$, n and n_2 are dummy variables, $N^{\boldsymbol{\Psi}}$ is the number of embedding vectors and $u(\bullet)$ is the step function. Essentially, $u(\bullet)$ contributes 1 to the sum for each $\left\|\boldsymbol{\Psi}(n) - \boldsymbol{\Psi}(n_2)\right\|$ less than r. The denominator is $N^{\boldsymbol{\Psi}}(N^{\boldsymbol{\Psi}}-1)$ rather than $(N^{\boldsymbol{\Psi}})^2$, because of the restriction that $n \neq n_2$.

The correlation dimension is defined as:

$$D_2 \triangleq \lim_{r \to 0} \frac{\log\big(C(r)\big)}{\log(r)}. \tag{2.25}$$

Actually, $D_2 \leq D_0$, but numerical evidence shows that D_2 is very close to D_0 [66].

Note that any real data set has a finite number of points, and hence it is not possible to take the limit $r \to 0$. Hence, there is some minimum distance between the points, and when r is less than that, $D_2 = 0$. Also, enough data points should be available [67-69].

Also, it is necessary to exclude temporally correlated points, since the correlation sum should cover a random sample of points drawn independently (successive elements of a time series are not usually independent). If indices between points differ by less than a quantity w, they are ignored. The quantity w is called the Theiler window [52]. Note that if $N \gg w$, the loss of O(wN) points is not significant, considering that that O(N^2) points are used to compute Eq. (2.24).

2.4.3 Lyapunov Exponents

The Lyapunov exponent of a dynamic system is a quantity which specifies the sensitive dependence on initial conditions (see Glossary). For a one-dimensional nonlinear system, the separation of 2 adjacent points after ς steps can be expressed as

$$\varepsilon_a = \left| f^{\varsigma}(y_0 + \varepsilon) - f^{\varsigma}(y_0) \right|, \tag{2.26}$$

where $f^{\varsigma}(\bullet)$ is the mapping function $f(\bullet)$ iterated ς times, y_0 is the initial point and $y_0 + \varepsilon$ is the adjacent point. The Gronwall inequality (see Glossary) states that the separation between 2 neighbouring solutions to the same differential equation can

separate from each other at a rate greater than exponential [70]. Hence, to relate ε_a to the initial separation, an exponential scaling relation is introduced:

$$\varepsilon_a \triangleq e^{\varsigma\lambda}\varepsilon\,. \tag{2.27}$$

Hence,

$$\begin{aligned} \varepsilon e^{\varsigma\lambda} &= \left| f^{\varsigma}(y_0+\varepsilon) - f^{\varsigma}(y_0) \right| \\ \lambda &= \frac{1}{\varsigma}\ln\frac{\left| f^{\varsigma}(y_0+\varepsilon) - f^{\varsigma}(y_0) \right|}{|\varepsilon|}. \end{aligned} \tag{2.28}$$

Exponential divergence only applies to small amplitudes of ε, due to finite attractor size [71]. For $\varepsilon \to 0$:

$$\begin{aligned} \lambda &= \lim_{\varsigma\to\infty}\lim_{\varepsilon\to 0}\frac{1}{\varsigma}\ln\left|\frac{f^{\varsigma}(y_0+\varepsilon) - f^{\varsigma}(y_0)}{\varepsilon}\right| \\ &= \lim_{\varsigma\to\infty}\frac{1}{\varsigma}\ln\left|\left(\left.\frac{df^{\varsigma}(y)}{dy}\right|_{y=y_0}\right)\right|. \end{aligned} \tag{2.29}$$

Furthermore, using the chain rule, given as

$$\begin{aligned} \frac{d}{dy_0}f^{\varsigma}(y_0) &= f'(y_{\varsigma-1})\left(\left.\frac{df^{\varsigma-1}(y)}{dy}\right|_{y=y_0}\right) \\ &= \prod_{i=0}^{\varsigma-1}\left(\left.\frac{df(y)}{dy}\right|_{y=y_i}\right), \end{aligned} \tag{2.30}$$

where $y_i = f^i(y_0)$. Thus, (2.29) may be written as

$$\lambda = \lim_{\varsigma\to\infty}\frac{1}{\varsigma}\sum_{i=0}^{\varsigma-1}\ln\left|f'(y_i)\right|. \tag{2.31}$$

This shows that the Lyapunov exponent λ gives the stretching rate per iteration, averaged over the trajectory. The dependence of λ on the choice of y_0 may be removed if the system is ergodic. Note that a chaotic system must have at least one positive Lyapunov exponent.

For the case of multiple dimensions, consider a chaotic dynamical system with d_F degrees of freedom (see Glossary). It has the mapping **F**, such that

$$\boldsymbol{\Psi}(n+1) = \mathbf{F}\left(\boldsymbol{\Psi}(n)\right). \tag{2.32}$$

Define

$$\boldsymbol{\Delta}(n) \triangleq \boldsymbol{\Psi}(n+1) - \boldsymbol{\Psi}(n). \tag{2.33}$$

Using Eq. (2.32) and (2.33),

$$\begin{aligned} \mathbf{F}\left(\boldsymbol{\Psi}(n+1)\right) &= \boldsymbol{\Psi}(n+2) \\ &= \boldsymbol{\Psi}(n+1) + \boldsymbol{\Delta}(n+1). \end{aligned} \tag{2.34}$$

A Taylor series can be formed:

$$\mathbf{F}\left(\boldsymbol{\Psi}(n+1)\right) = \mathbf{F}\left(\boldsymbol{\Psi}(n)\right) + \mathbf{DF}\left(\boldsymbol{\Psi}(n)\right)\bullet\boldsymbol{\Delta}(n) + \cdots, \tag{2.35}$$

where **DF** is the $d_F \times d_F$ Jacobian matrix. Substituting Eq. (2.34) into Eq. (2.35),

$$\begin{aligned} \boldsymbol{\Psi}(n+1) + \boldsymbol{\Delta}(n+1) &= \boldsymbol{\Psi}(n+1) + \mathbf{DF}\left(\boldsymbol{\Psi}(n)\right)\bullet\boldsymbol{\Delta}(n) + \cdots \\ \boldsymbol{\Delta}(n+1) &\approx \mathbf{DF}\left(\boldsymbol{\Psi}(n)\right)\bullet\boldsymbol{\Delta}(n), \end{aligned} \tag{2.36}$$

assuming $\boldsymbol{\Delta}(n)$ is small and $\boldsymbol{\Delta}(n+1)$ stays small. Over multiple time steps ς,

$$\begin{aligned} \boldsymbol{\Delta}(n+\varsigma) &= \mathbf{DF}\left(\boldsymbol{\Psi}(n+\varsigma-1)\right)\bullet\mathbf{DF}\left(\boldsymbol{\Psi}(n+\varsigma-2)\right)\cdots\mathbf{DF}\left(\boldsymbol{\Psi}(n)\right)\bullet\boldsymbol{\Delta}(n) \\ &= \mathbf{DF}^{\varsigma}\left(\boldsymbol{\Psi}(n)\right)\bullet\boldsymbol{\Delta}(n). \end{aligned} \tag{2.37}$$

The square of the magnitude of the vector is given as:

$$\left\|\boldsymbol{\Delta}(n+\varsigma)\right\|^2 = \boldsymbol{\Delta}^T(n)\bullet\left[\mathbf{DF}^{\varsigma}\left(\boldsymbol{\Psi}(n)\right)\right]^T\mathbf{DF}^{\varsigma}\left(\boldsymbol{\Psi}(n)\right)\bullet\boldsymbol{\Delta}(n). \tag{2.38}$$

The essential quantity determining this is $\left[\mathbf{DF}^{\varsigma}\left(\boldsymbol{\Psi}(n)\right)\right]^T\bullet\mathbf{DF}^{\varsigma}\left(\boldsymbol{\Psi}(n)\right)$. Let the Oseledec matrix be

$$\mathbf{OSL}\left(\varsigma, \boldsymbol{\Psi}(n)\right) \triangleq \left\{\left[\mathbf{DF}^{\varsigma}\left(\boldsymbol{\Psi}(n)\right)\right]^T\bullet\mathbf{DF}^{\varsigma}\left(\boldsymbol{\Psi}(n)\right)\right\}^{(1/2)\varsigma}. \tag{2.39}$$

$\left[\mathbf{DF}^{\varsigma}\left(\boldsymbol{\Psi}(n)\right)\right]^T\bullet\mathbf{DF}^{\varsigma}\left(\boldsymbol{\Psi}(n)\right)$ is real and symmetric, and so the $(1/2)\varsigma$ power is well defined. Oseledec's multiplicative ergodic theorem [72] states that as $\varsigma \to \infty$, the

matrix **OSL** is independent of $\mathbf{\Psi}(n)$ for almost all $\mathbf{\Psi}(n)$, well defined, and has eigenvalues e^{λ_1}, e^{λ_2}, …, $e^{\lambda_{d_F}}$ for a system with d_F degrees of freedom [43]. The Lyapunov exponents are none other than λ_1, λ_2, ..., λ_{d_F}, which are the global Lyapunov exponents of the observed attractor of the dynamical system.

The $\varsigma \to \infty$ limit is the reason why the standard global Lyapunov exponents are indicative of long time phase space instability. Hence, they are relevant to the predictability of the system only on the average in the long time limit. To examine the finite time behaviour, the eigenvalues of the matrix $\mathbf{OSL}\left(\varsigma, \mathbf{\Psi}(n)\right)$, for finite ς, are examined. Each eigenvalue for the d-th degree of freedom is given as $e^{\lambda_d(\varsigma, \mathbf{\Psi})}$, and each $\lambda_d(\varsigma, \mathbf{\Psi})$ is the corresponding local Lyapunov exponent.

Abarbanel *et al.* [73] states that the predictability on a strange attractor depends on the local magnitude of the instability at the phase space point associated with the next time step. If the attractor is homogeneous, in the sense that the local Lyapunov exponents are the same in all parts of it, then the global exponents would be adequate for prediction. In Ref. [73], numerical evidence is given that local exponents vary significantly over the attractor. After all, in practice, ς cannot be infinite [74].

2.4.4 Kaplan-Yorke Dimension

Kaplan and Yorke [71] conjectured that a relationship exists between the Lyapunov exponents and the dimension of a strange attractor. An intuitive demonstration of this conjecture in 2 dimensions is given as follows.

Assume that the attractor lies inside a square, whose sides are normalized to unity. A chaotic mapping stretches one side by $L_1 > 1$, and the other by $L_2 < 1$. Since the system is dissipative, $L_1 L_2 < 1$, *i.e.* the state space is bounded. The mapped area fits back inside the unit square in the shape of a horseshoe; folding takes place, permitting the divergence of nearby orbits, given the constraint that the state space has to be bounded. The strange attractor must lie inside this horseshoe (see Figure 2.11).

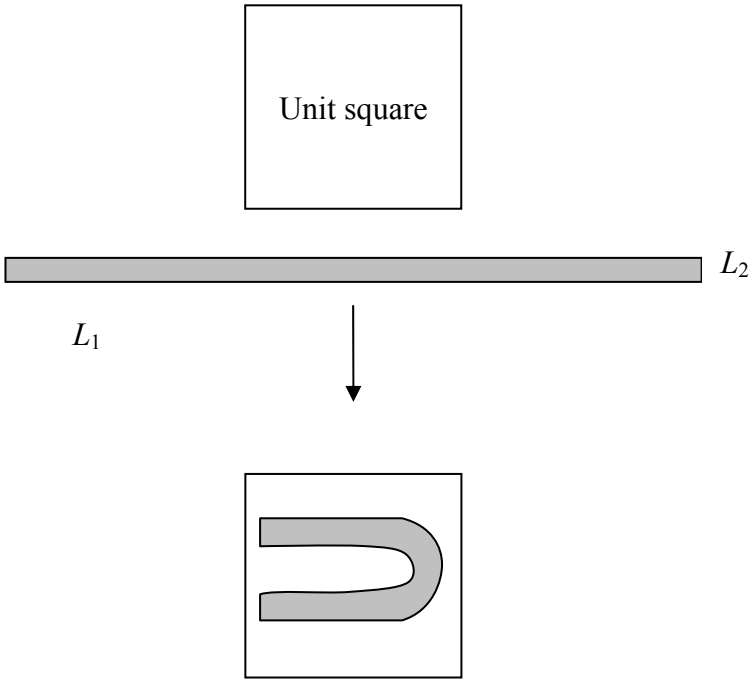

Figure 2.11 Stretching and folding induced by chaotic mapping in 2 dimensions.

The smallest number of squares that have sides L_2, needed to cover the horseshoe is given by a function $N(\bullet)$ of L_2, where

$$N(L_2) = \text{int}(L_1 / L_2) + 1 . \tag{2.40}$$

Suppose the process is repeated k times. It is now possible to cover the attractor with smaller squares of size L_2^k. For the k-th map, this gives

$$N(L_2^k) \approx (L_1 / L_2)^k . \tag{2.41}$$

The box counting dimension, D_0, is defined by

$$\lim_{\varepsilon \to 0} N(\varepsilon) = \varepsilon^{-D_0}, \tag{2.42}$$

where $N(\varepsilon)$ is the minimum number of ε squares which covers the set. For large k,

$$N(L_2^k) \approx (L_1 / L_2)^k = (L_2^k)^{-D_0}. \tag{2.43}$$

This gives

$$1 - D_0 = \frac{\ln L_1}{\ln L_2}. \tag{2.44}$$

Since $L_2 < 1 \Rightarrow \lambda_2 < 0$, substituting $\lambda_1 = \ln L_1$ and $-|\lambda_2| = \ln L_2$ for the Lyapunov exponents results in

$$D_0 = 1 + \frac{\lambda_1}{|\lambda_2|}. \tag{2.45}$$

It is easy to generalize this demonstration to multiple dimensions. This time, the expansion is a hypervolume, rather than just a length L_1:

$$\prod_{i=1}^{K_Y} L_i \geq 1, \tag{2.46}$$

where K_Y is the smallest integer such that the overall hypervolume is not contracting. Note that some of the dimensions can be contracting, as long as (2.46) is satisfied. The dimensions, L_i are arranged, such that the associated exponents are ordered in descending order $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_{d_F}$. The tiny hypercube required to cover the attractor has a length of $L_{K_Y+1} < 1$ and hypervolume

$$\left(L_{K_Y+1}\right)^{K_Y} < 1. \tag{2.47}$$

Dividing the expansion (2.46) by the hypervolume (2.47) and substituting into Eq. (2.43) gives

$$N(L^k_{K_Y+1}) \approx \left(\frac{\prod_{i=1}^{K_Y} L_i}{L^{K_Y}_{K_Y+1}} \right)^k = (L^k_{K_Y+1})^{-D_0}. \tag{2.48}$$

This leads to

$$\begin{aligned} \prod_{i=1}^{K_Y} L_i &= (L_{K_Y+1})^{-D_0+K_Y} \\ \sum_{i=1}^{K_Y} \ln L_i &= \left(-D_0 + K_Y\right) \ln(L_{K_Y+1}). \end{aligned} \tag{2.49}$$

Since $L_{K_Y+1} < 1 \Rightarrow \lambda_{K_Y+1} < 0$, substituting $L_i = e^{\lambda_i}$ and $-\left|\lambda_{K_Y+1}\right| = \ln L_{K_Y+1}$, we get

$$\begin{aligned} -D_0 &= -K_Y + \frac{\sum_{i=1}^{K_Y} \lambda_i}{\lambda_{K_Y+1}} \\ D_0 &= K_Y + \frac{\sum_{i=1}^{K_Y} \lambda_i}{\left|\lambda_{K_Y+1}\right|}, \end{aligned} \tag{2.50}$$

where K_Y is the largest integer whereby the following is true: $\sum_{i=1}^{K_Y} \lambda_i \geq 0$. Essentially, the Kaplan-Yorke conjecture states the Kaplan-Yorke dimension,

$$D_{KY} \triangleq K_Y + \frac{\sum_{i=1}^{K_Y} \lambda_i}{\left|\lambda_{K_Y+1}\right|}, \tag{2.51}$$

gives a good estimate of D_0. For physically realizable systems of dimension 3:

$$D_{KY} = 2 + \frac{\lambda_1}{\left|\lambda_3\right|}, \tag{2.52}$$

where λ_1 is the largest positive Lyapunov exponent and λ_3 is the negative Lyapunov exponent. The middle Lyapunov exponent, λ_2, is theoretically 0, otherwise the attractor is unstable [75]. In practice, it is hardly possible for any computed Lyapunov exponent to be exactly 0, due to noise and computational errors.

One problem is that for dimensions greater than 2, there are situations whereby D_{KY} overestimates D_0 [71]. Nonetheless, the weight of numerical evidence [76] supports the Kaplan-Yorke conjecture, and it can give good estimates for the usual systems encountered, such as the Lorenz system.

2.4.5 Kolmogorov Entropy

Sensitive dependence on initial conditions results in entropy increase, due to the loss of positional information with time. The Kolmogorov entropy (KE) of an attractor can be considered as the rate of information loss along the attractor, or as a measure of the degree of predictability of points along the attractor, for an arbitrary initial point. It can be computed from Pesin's identity [45], *i.e*. KE is equivalent to the sum of the positive Lyapunov exponents.

2.4.6 The Horizon of Predictability

The Horizon of Predictability (HOP) [13] is estimated as the average time required for trajectories that are within 1% of root mean square attractor size, to separate to 50% of root mean square attractor size. From the definition of the Lyapunov exponents, this can be written as

$$\frac{0.50}{0.01} = e^{\lambda_1 t}, \tag{2.53}$$

where λ_1 is the largest positive Lyapunov exponent. From Eq. (2.53), HOP can be estimated as

$$t = \frac{\ln(50)}{\lambda_1}. \tag{2.54}$$

2.5 Contributions of this Chapter

- Finding the first minimum in the average mutual information is related to the maximum entropy method (Section 2.2).
- The reason why a delay of 1 or 2 is prescribed for chaotic maps is explained. Thus, a sounder theoretical basis had been put forward to explain the rules for choosing embedding delay, which had previously been prescriptive (Section 2.2).
- Since the use of Taken's Embedding Theorem is necessary for dynamic reconstruction, it is vital to have a reliable method of extracting the embedding dimension, even in the presence of high level of noise. Instead of using one algorithm for estimating embedding dimension, GFNN and Cao's method are used; this is useful for double checking (Section 2.3). If the estimated embedding dimension is too large, it may result in spurious Lyapunov exponents being computed.

2.6 Summary

A brief introduction to chaos theory was given. The main focus was to explain how embedding delay, embedding dimension and chaotic invariants are obtained. These quantities will be crucial in setting up a RBF network to model a chaotic system. Comparing the estimated chaotic invariants of the observed data, and the estimated chaotic invariants of the predicted time series also provides an indication of the performance of the predictor.

CHAPTER 3

RBF Networks and Variants

Prediction of a chaotic time series is a nonlinear problem which can be handled by RBFs, because of their universal approximation capabilities [77]. In the presence of noise, the problem becomes ill-posed, and regularization (Section 3.6.1) is required. In this chapter, the RBF network and variants are introduced, in the context of time series prediction.

3.1 Predictive Modelling

Consider prediction of a scalar time series with N_y samples, using delay coordinates, as in Ref. [78]:

$$\hat{y}(n+T) = \hat{f}\left(y(n),\quad y(n-\tau),\quad \cdots,\quad y(n-d_E\tau+\tau)\right) \tag{3.1}$$

where $T \in \mathbb{Z}^+$ is the number of time steps ahead being predicted, $y(n)$ is the actual data at sample n, $\hat{f}(\bullet)$ is the estimated function of the actual system $f(\bullet)$, producing the estimate $\hat{y}(n+T)$ at sample $n+T$. The embedding dimension, d_E, is found using the method of Global False Nearest Neighbours (GFNN) [62] and Cao's method [57], while τ is the embedding time delay, which is found from the first minimum in the mutual information [49].

The training set and the test set are normalized by subtracting the mean of the training set (μ^{train}), and dividing by the standard deviation (σ^{train}) of the training set. This obviates the need for a bias, and also guards against numerical problems.

3.1.1 Information Preservation Rule

Next, consider Eq. (3.1) recast in a different form as in Ref. [22] where $T = 1$:

$$\hat{y}(n+1) = \hat{f}\left(y(n), \quad y(n-1), \quad \cdots, \quad y(n-d_E\tau+1)\right). \tag{3.2}$$

The rationale is the information preservation rule [79], *i.e.* all available information should be preserved optimally and used efficiently.

The difference in using Eq. (3.2) is that all information between lags is utilized, right up to and excluding the predicted value. Eq. (3.2) may appear strange initially, because it is equivalent to having $\tau = 1$, and with the embedding dimension

$$p = d_E\tau, \tag{3.3}$$

which is too high. However, it is the length of the time window which is important [80, 81]. The length of the time window is defined by

$$\tau_w = (d_E - 1)\tau. \tag{3.4}$$

3.1.2 Vector Notation

It appears natural to group the time delay coordinates in a vector

$$\boldsymbol{\psi}(n) = \left(y(n), \quad y(n-1), \quad \cdots, \quad y(n-(p-1))\right), \tag{3.5}$$

where p is also the number of nodes in the input layer of the predictor.

Vector notation is much more concise and powerful; an Autoregressive (AR) model can be expressed by:

$$y(n) = \mathbf{a}^H\boldsymbol{\psi}_I(n-1) + \eta(n), \tag{3.6}$$

where $\mathbf{a} \in \mathbb{C}^{p-1}$ contains the coefficients of the AR model of order $p-1$. H is the

Hermitian operator, $\boldsymbol{\psi}_l(n) = \big(y(n), \quad y(n-1), \quad \cdots \quad y(n-(p-2))\big)$ and $\eta(n)$ is a white noise process. The reason why AR processes are classified as linear stochastic processes now becomes clear: the Hermitian inner product is a linear operator.

It also suggests that a nonlinear one-step predictor can perform the same job as the linear one:

$$\hat{y}(n) = \hat{f}\left(\boldsymbol{\psi}_l(n-1)\right) + e(n). \tag{3.7}$$

The predictor $\hat{f}(\bullet)$ "predicts" a value $\hat{y}(n)$ with $e(n)$ as the model error. Here, $y(n)$ is the observed data, $\boldsymbol{\psi}_l(n-1)$ is the input vector. In fact, prediction of a chaotic system can be regarded as a nonlinear AR problem [82].

Embedding in phase space is a geometrical method, and the emphasis is on the manipulation of vectors and matrices. Depending on the context, y_i, $\hat{y}_i$ and $\boldsymbol{\psi}_i$ are used, instead of *y(n)*, $\hat{y}(n)$ and $\boldsymbol{\psi}(n)$, because the brackets can become unwieldy.

The collection of all $\boldsymbol{\psi}_i$ forms a matrix of dimension $N_\psi \times p$, while the collection of all y_i forms a vector of desired output **y**, of dimension N_ψ, where $N_\psi = N_y - p$. Note that N_ψ is the maximum number of $\boldsymbol{\psi}_i$ and y_i which can be formed from a single scalar time series with N_y samples. The use of different subscripts emphasises the difference between *n* and *i*, as $n \in [1, N_y]$ whilst $i \in [1, N_\psi]$.

In the context of neural networks, typically the data set is split into the training set (see Glossary) with N_ψ^{train} training examples, or design sets (see Glossary) with

N_{ψ}^{design} training examples. In this work, N refers to the number of examples available for training, be it in the training set or design set. Where it is necessary to be explicit, N_{ψ}^{train} or N_{ψ}^{design} will be used.

3.2 RBF Architecture

Typically, a RBF network has p input nodes, M_c centers, M linear weights, 1 bias, w_0, and 1 output node, $\hat{y}_{i+1}$, connected as in Figure 3.1. Each j-th center is associated with a nonlinear function called a basis function, $\phi_j(\bullet)$. This work focuses on RBF networks which utilize clustering methods to organize the centers. The clustering procedures may produce empty clusters (Section 3.3.2), which are dropped. Hence, $M_c \geq M$, with equality only when there are no empty clusters.

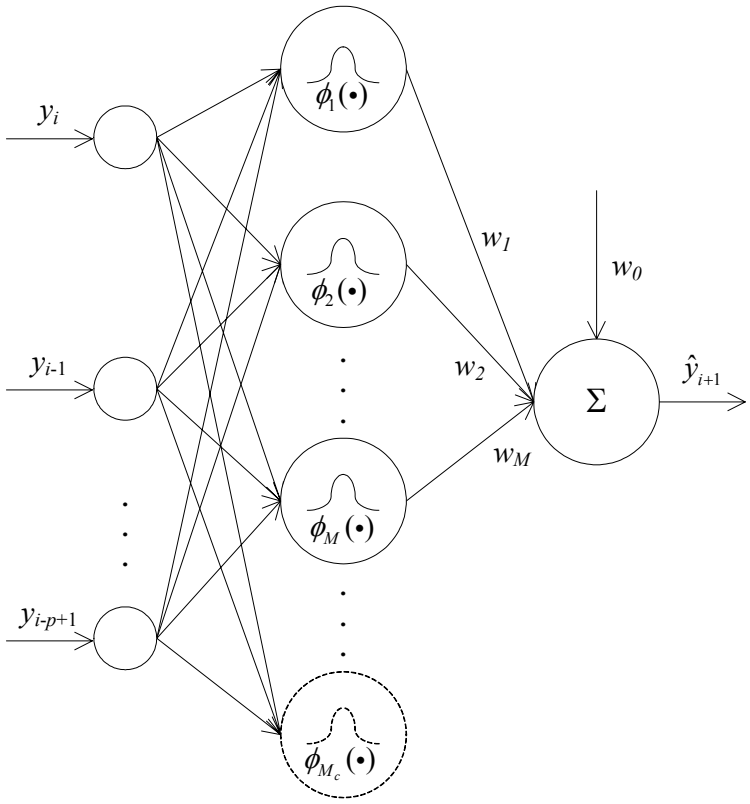

Figure 3.1 Schematic of a RBF network for time series prediction.

A RBF network can also be seen as a weighted sum of nonlinear functions:

$$\hat{y}_{i+1} = w_0 + \sum_{j=1}^{M} w_j \phi_j \left(\rho_{ij} \right), \tag{3.8}$$

where w_j is the j-th weight, ρ_{ij} is the norm (Euclidean norm, for ordinary RBFs) between the i-th point, $\boldsymbol{\psi}_i$, and the j-th center, $\boldsymbol{\mu}_j$, and $\phi_j(\bullet)$ is the j-th basis function.

Typically, large numbers of basis functions may be required for real life problems, *i.e*. Haykin *et al.* used 1500 centers to model sea clutter data [22]. Another issue is that the standard deviations of the basis functions are usually chosen in an ad-hoc manner. Orr [83] showed that better performance was achieved by having a single adjustable basis function width, rather than a similar RBF, where the number of centers, M_c, is adjustable, but with only one fixed width.

In a multi-dimensional setting, the most general way to adjust the width of a basis function is to modify its associated covariance matrix. This naturally results in Elliptical Basis Functions (EBFs). By sacrificing radial symmetry, it may be possible to approximate a function using less basis functions, resulting in a simpler model. This may be useful for practical data sets which are large and multivariate.

3.3 Clustering

Clustering is the unsupervised process of partitioning N data points into M_c sets [84]. It is useful for organizing the centers in the hidden layer [85, 86]. Instead of using N centers for N data points, RBF with clustering uses M centers (after subtracting empty clusters, if any), which is faster when $N \gg M$ (Section 3.5).

A brief outline of the *k*-means clustering algorithm and its relatives:

---

1. Randomly initialize the M_c groups, by selecting M_c data points and using them as the centers.
2. At each iteration of the clustering process a distance matrix is formed:

$$\mathbf{D}^{(i_t)} = \left(\rho_{ij}\right)_{N\times M_c}, \tag{3.9}$$

   where i_t is the iteration number and ρ_{ij} is the distance between the *i*-th point and the *j*-th center.
3. Each *i*-th point has a membership value, $b_{ij} \in [0,1]$, with respect to each *j*-th center. The membership value is assigned based on the value of ρ_{ij}. The membership value is binary in the case of *k*-means clustering, and fuzzy for fuzzy *c*-means (FCM) [87] and Gustafson-Kessel (GK) clustering [88]. For *k*-means clustering, this means that the points which are nearest to the *j*-th center are assigned $b_{ij} = 1$, and the other points are assigned $b_{ij} = 0$. Expectation Maximization (EM) (see Appendix D or Ref. [89]) works with probabilities, so it can also be considered to be a soft clustering method [90].
4. The means are recalculated.
5. Repeat steps 2 to 4, and terminate when memberships stop changing. Another possibility is to stop when the positions of centers stop changing. Alternatively, stop after a predetermined number of iterations.

In general, clustering algorithms typically minimize a functional J_m:

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{M_c} b_{ij}^m \rho_{ij}^2 \,, \tag{3.10}$$

where $b_{ij} \in [0,1]$ is the membership of point i in cluster j, and ρ_{ij} is the distance between $\boldsymbol{\psi}_i$ at point i and $\boldsymbol{\mu}_j$ (center of cluster j). The weighting exponent, m, has to be greater than 1 and is usually set as 2.

The use of EBFs (Section 3.4.1) naturally suggests the use of GK clustering, which is essentially the generalized form of FCM. It uses the distance

$$\rho_{ij} = \sqrt{(\boldsymbol{\psi}_i - \boldsymbol{\mu}_j)^T \left[v_j \left(\left| \mathbf{A}_j \right| \right)^{\frac{1}{p}} \mathbf{A}_j^{-1} \right] (\boldsymbol{\psi}_i - \boldsymbol{\mu}_j)} \,, \tag{3.11}$$

where v_j is a constant usually set to 1, and the fuzzy covariance matrix $\mathbf{A}_j$ is defined as

$$\mathbf{A}_j \triangleq \frac{\sum_{i=1}^{N} b_{ij}^m (\boldsymbol{\psi}_i - \boldsymbol{\mu}_j)(\boldsymbol{\psi}_i - \boldsymbol{\mu}_j)^T}{\sum_{i=1}^{N} b_{ij}^m} \,. \tag{3.12}$$

The method of Babuska *et al.* [91] implements GK clustering with two modifications to avoid numerical problems:

1. constraining the condition number (ratio of largest eigenvalue to smallest eigenvalue) of each covariance matrix.
2. regularization of each covariance matrix by

$$\mathbf{M}_j \triangleq (1-\gamma_c)\mathbf{A}_j + \gamma_c \left(\left| \mathbf{A}_0 \right| \right)^{\frac{1}{p}} \mathbf{I} \,, \tag{3.13}$$

where $\mathbf{A}_0$ is the covariance matrix of the whole data set, and $\gamma_c \in [0,1]$ is the regularization parameter. The value of γ_c determines the shape of the clusters, as the distance is now

$$\rho_{ij} = \sqrt{(\boldsymbol{\psi}_i - \boldsymbol{\mu}_j)^T \left[v_j \left(\left| \mathbf{M}_j \right| \right)^{\frac{1}{p}} \mathbf{M}_j^{-1} \right] (\boldsymbol{\psi}_i - \boldsymbol{\mu}_j)}, \tag{3.14}$$

which is similar to Eq. (3.11). When $\gamma_c = 1$, all the covariance matrices are equal, and the clusters are spherical. However, γ_c is effectively a hyperparameter, since it cannot be determined by training. Thus, one may have to perform cross validation in order to find the proper value of γ_c, but this would increase computational requirements.

3.3.1 Erraticity

Suppose some assumptions are made about the given data [87]:

1. The samples come from a known number of classes, $M_c \in \mathbb{Z}^+$.
2. The prior probabilities $P(\omega_j)$ for each class is known, where $j = 1, ..., M_c$.
3. The forms for the class-conditional probability densities $p(\boldsymbol{\psi} \mid \omega_j, \boldsymbol{\theta}_j)$ are known, where $j = 1, ..., M_c$.
4. The values for each of parameter vector $\boldsymbol{\theta}_j$ is unknown, where $j = 1, ..., M_c$.
5. It is unknown which data point belongs to which class.

With these assumptions, the pdf for the data samples can be given by a mixture density (sum of pdfs):

$$p(\boldsymbol{\psi} \mid \boldsymbol{\theta}) = \sum_{j_2=1}^{M_c} p(\boldsymbol{\psi} \mid \omega_{j_2}, \boldsymbol{\theta}_{j_2}) P(\omega_{j_2}), \tag{3.15}$$

where $\boldsymbol{\theta} = \begin{pmatrix} \boldsymbol{\theta}_1 \\ \vdots \\ \boldsymbol{\theta}_{M_c} \end{pmatrix}$ and j_2 is a dummy variable.

Consider a data set $\mathcal{D} = \{\boldsymbol{\psi}_1, \cdots \boldsymbol{\psi}_N\}$ of N samples drawn independently from the mixture density in Eq. (3.15), with $\boldsymbol{\theta}$ fixed but unknown. The likelihood of the observed samples is defined by

$$p(\mathcal{D} \mid \boldsymbol{\theta}) \triangleq \prod_{k=1}^{N} p(\boldsymbol{\psi}_k \mid \boldsymbol{\theta}) . \tag{3.16}$$

The corresponding log-likelihood is

$$l(\boldsymbol{\theta}) = \sum_{i=1}^{N} \ln p(\boldsymbol{\psi}_i \mid \boldsymbol{\theta}) . \tag{3.17}$$

The maximum-likelihood estimate $\hat{\boldsymbol{\theta}}$ is the value of $\boldsymbol{\theta}$ which maximizes $p(\mathcal{D} \mid \boldsymbol{\theta})$. If $p(\mathcal{D} \mid \boldsymbol{\theta})$ is assumed to be differentiable with respect to $\boldsymbol{\theta}$, then

$$\frac{\partial l(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_j} = \sum_{i=1}^{N} \frac{1}{p(\boldsymbol{\psi}_i \mid \boldsymbol{\theta})} \frac{\partial}{\partial \boldsymbol{\theta}_j} \left[p(\boldsymbol{\psi}_i \mid \boldsymbol{\theta}) \right]. \tag{3.18}$$

Substituting Eq. (3.15) in Eq. (3.18) results in

$$\begin{aligned} \frac{\partial l(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_j} &= \sum_{i=1}^{N} \frac{1}{p(\boldsymbol{\psi}_i \mid \boldsymbol{\theta})} \frac{\partial}{\partial \boldsymbol{\theta}_j} \left[\sum_{j_2=1}^{M_c} p(\boldsymbol{\psi}_i \mid \omega_{j_2}, \boldsymbol{\theta}_{j_2}) P(\omega_{j_2}) \right] \\ &= \sum_{i=1}^{N} \frac{1}{p(\boldsymbol{\psi}_i \mid \boldsymbol{\theta})} \frac{\partial}{\partial \boldsymbol{\theta}_j} \left[p(\boldsymbol{\psi}_i \mid \omega_j, \boldsymbol{\theta}_j) P(\omega_j) \right], \end{aligned} \tag{3.19}$$

assuming the partial derivative vanishes if $j \neq j_2$. Introducing the posterior probability:

$$P(\omega_j \mid \boldsymbol{\psi}_i, \boldsymbol{\theta}) = \frac{p(\boldsymbol{\psi}_i \mid \omega_j, \boldsymbol{\theta}_j) P(\omega_j)}{p(\boldsymbol{\psi}_i \mid \boldsymbol{\theta})}, \tag{3.20}$$

and substituting Eq. (3.20), $\dfrac{\partial l(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_j}$ can be rewritten as:

$$\begin{aligned}\frac{\partial l(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_j} &= \sum_{i=1}^{N} \frac{P(\omega_j)}{p(\boldsymbol{\psi}_i \mid \boldsymbol{\theta})} \frac{\partial}{\partial \boldsymbol{\theta}_j}\Big[p(\boldsymbol{\psi}_i \mid \omega_j, \boldsymbol{\theta}_j)\Big] \\ &= \sum_{i=1}^{N} \frac{P(\omega_j \mid \boldsymbol{\psi}_i, \boldsymbol{\theta})}{p(\boldsymbol{\psi}_i \mid \omega_j, \boldsymbol{\theta}_j)} \frac{\partial}{\partial \boldsymbol{\theta}_j}\Big[p(\boldsymbol{\psi}_i \mid \omega_j, \boldsymbol{\theta}_j)\Big] \\ &= \sum_{i=1}^{N} P(\omega_j \mid \boldsymbol{\psi}_i, \boldsymbol{\theta}) \frac{\partial}{\partial \boldsymbol{\theta}_j} \ln\Big[p(\boldsymbol{\psi}_i \mid \omega_j, \boldsymbol{\theta}_j)\Big].\end{aligned} \tag{3.21}$$

The maximum likelihood solution occurs when $\frac{\partial l(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_j} = \mathbf{0}$. Assuming mixture density is Gaussian with unknown mean vectors, $\boldsymbol{\theta}_j = \boldsymbol{\mu}_j$ and $\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_1 \\ \vdots \\ \boldsymbol{\mu}_{M_c} \end{pmatrix}$. Then

$$\ln\Big[p(\boldsymbol{\psi}_i \mid \omega_j, \boldsymbol{\mu}_j)\Big] = -\ln\Big[(2\pi)^{p/2} \left|\boldsymbol{\Sigma}_j\right|^{1/2}\Big] - \frac{1}{2}(\boldsymbol{\psi}_i - \boldsymbol{\mu}_j)^T \boldsymbol{\Sigma}_j^{-1}(\boldsymbol{\psi}_i - \boldsymbol{\mu}_j), \tag{3.22}$$

where $\boldsymbol{\Sigma}_j$ is the covariance matrix, and

$$\frac{\partial}{\partial \boldsymbol{\mu}_j} \ln\Big[p(\boldsymbol{\psi}_i \mid \omega_j, \boldsymbol{\mu}_j)\Big] = \boldsymbol{\Sigma}_j^{-1}(\boldsymbol{\psi}_i - \boldsymbol{\mu}_j). \tag{3.23}$$

Substituting Eq. (3.23) into Eq. (3.21),

$$\sum_{i=1}^{N} P(\omega_j \mid \boldsymbol{\psi}_i, \boldsymbol{\mu}) \boldsymbol{\Sigma}_j^{-1}(\boldsymbol{\psi}_i - \boldsymbol{\mu}_j) = 0 \tag{3.24}$$

is obtained. Multiplying Eq. (3.24) by $\boldsymbol{\Sigma}_j$ and rearranging terms,

$$\boldsymbol{\mu}_j = \frac{\sum_{i=1}^{N} P(\omega_j \mid \boldsymbol{\psi}_i, \boldsymbol{\mu}) \boldsymbol{\psi}_i}{\sum_{i=1}^{N} P(\omega_j \mid \boldsymbol{\psi}_i, \boldsymbol{\mu})}. \tag{3.25}$$

Eq. (3.25) suggests an iterative scheme for improving estimates of the mean [87]:

$$\hat{\boldsymbol{\mu}}_j(i_t + 1) = \frac{\sum_{i=1}^{N} P(\omega_j \mid \boldsymbol{\psi}_i, \hat{\boldsymbol{\mu}}(i_t)) \boldsymbol{\psi}_i}{\sum_{i=1}^{N} P(\omega_j \mid \boldsymbol{\psi}_i, \hat{\boldsymbol{\mu}}(i_t))}, \tag{3.26}$$

where $\hat{\boldsymbol{\mu}}_j$ is the estimate of $\boldsymbol{\mu}_j$, i_t is the iteration number and $\hat{\boldsymbol{\mu}} = \begin{pmatrix} \hat{\boldsymbol{\mu}}_1 \\ \vdots \\ \hat{\boldsymbol{\mu}}_{M_c} \end{pmatrix}$. This can be viewed as hill climbing for maximizing the log-likelihood function $l(\boldsymbol{\mu})$ [87], and like all hill climbing procedures, there is no guarantee of reaching the global maximum.

For example, consider a mixture model consisting of two univariate Gaussians as a function of their means, μ_1 and μ_2, as in Figure 3.2. Since the Gaussians are univariate, each $\boldsymbol{\mu}_j$ is actually a scalar and $\boldsymbol{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}$. Suppose there are 2 clusters with means at a and b. There are 2 possibilities for $\boldsymbol{\mu}$: $\boldsymbol{\mu}_a = \begin{pmatrix} a \\ b \end{pmatrix}$ and $\boldsymbol{\mu}_b = \begin{pmatrix} b \\ a \end{pmatrix}$. This results in 2 local maxima for $l(\boldsymbol{\mu})$ (which is a function of μ_1 and μ_2) in Figure 3.3.

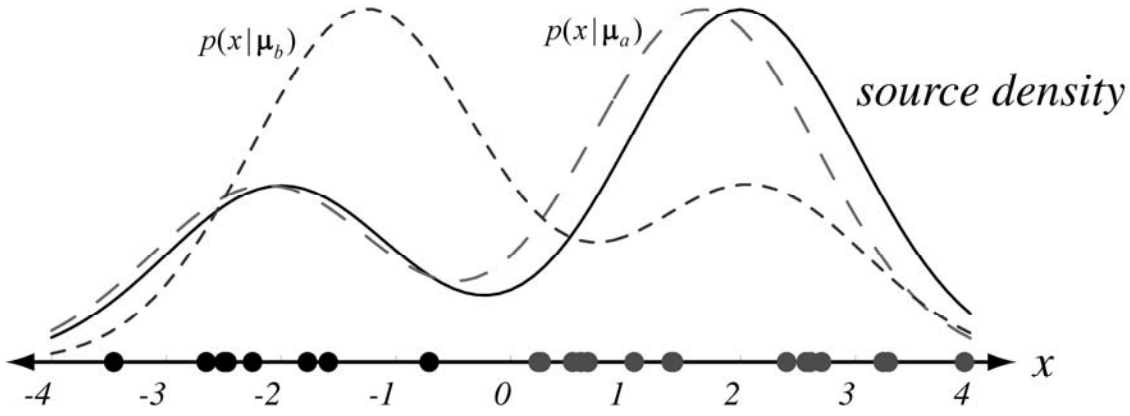

Figure 3.2 Mixture model consisting of two univariate Gaussians as a function of their means, μ_a and μ_b [87].

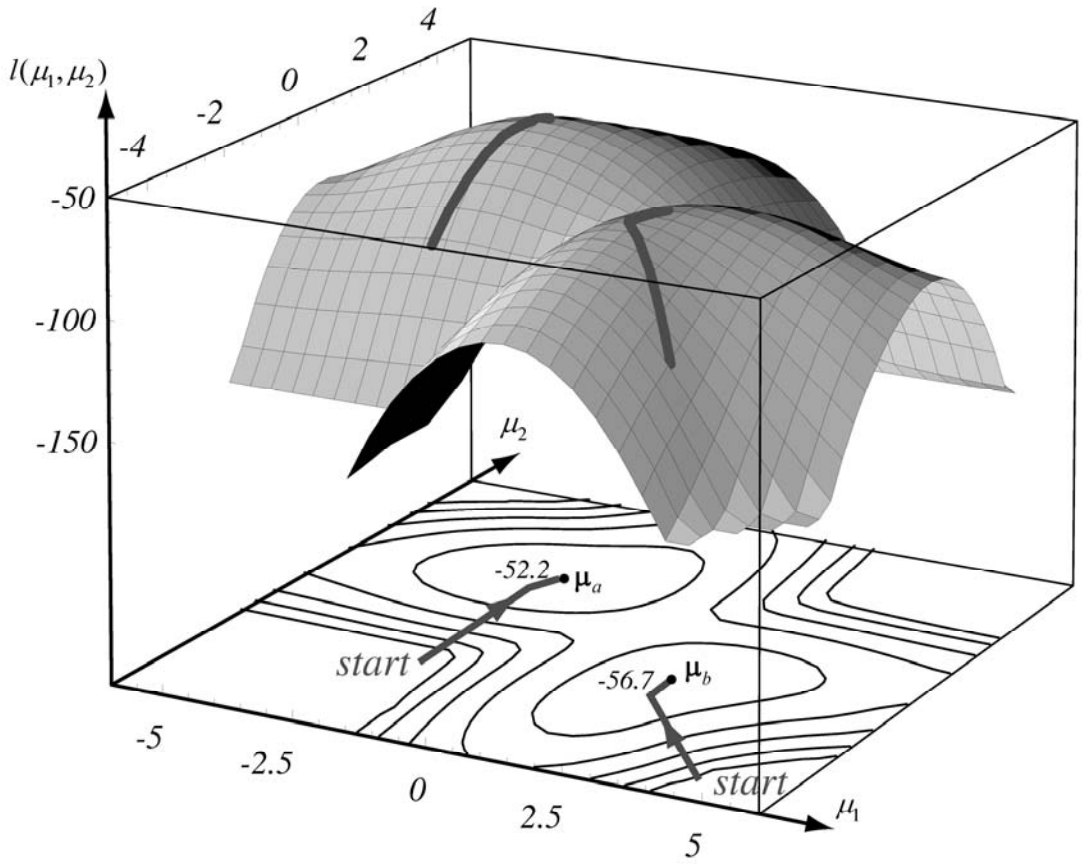

Figure 3.3 Trajectories on $l(\mu_1, \mu_2)$ for estimation of means using *k*-means [87].

The ordinary *k*-means algorithm can be regarded as an unsupervised clustering technique that estimates the means of the 2 Gaussians. It is an iterative process, a form of stochastic hill climbing in the log-likelihood function $l(\boldsymbol{\mu})$. A brief summary of the *k*-means algorithm is given below.

---

1. Randomly initialize the M_c centroids.
2. Classify the samples according to the nearest $\boldsymbol{\mu}_j$.
3. Recompute $\boldsymbol{\mu}_j$
4. Repeat steps 2 & 3 until there is no change in $\boldsymbol{\mu}_j$

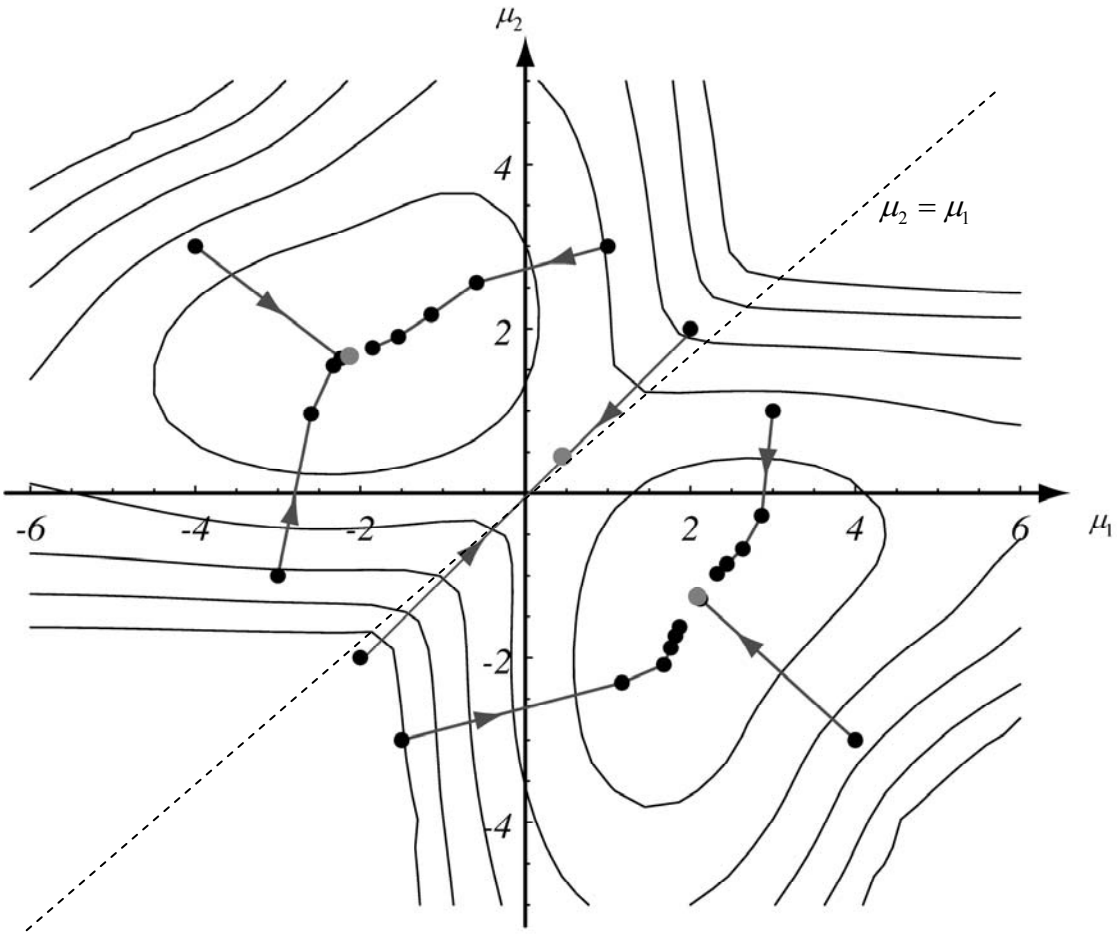

Figure 3.4 Trajectories for *k*-means clustering, adapted from Ref. [87].

The contours in Figure 3.4 represent equal values of the log-likelihood function $l(\boldsymbol{\mu})$. The dots indicate parameter values after successive iterations of the *k*-means algorithm. The trajectories on $l(\boldsymbol{\mu})$ in Figure 3.3 and Figure 3.4 illustrate that the *k*-means algorithm can be sensitive to initial starting conditions and converge to some local maxima [92]. Six of the starting points shown lead to local maxima, whereas two lead to a saddle point near $\boldsymbol{\mu} = \mathbf{0}$. In fact, it was shown by Selim and Ismail [93] that under certain conditions, the algorithm may fail to converge to a local optimum.

Since the initial centers are randomly placed, the position of each center and the number of members in each cluster could be different each time clustering is

performed, which ultimately affect the regression results. This means that the generalization errors estimated by cross validation for same M_c, but different regularization parameter γ (Section 3.6.1), could become "erratic", depending on the initialization. It is highly tempting to use the word "inconsistent" to describe this phenomenon, but it is already used in statistics. Assume that one possesses data known to be generated by a bimodal process where each mode is generated by uniform noise, and there are no overlaps between the 2 sets (in contrast to the Gaussians in Figure 3.2). Hence, *k*-means is a consistent estimator of the position of the 2 means. On the other hand, it is possible to design situations whereby "erraticity" is possible, regardless of the number of data points in the groups. Thus, the 2 concepts are distinct.

One way to deal with the problem of "erraticity" is to redo the clustering multiple times, but this increases computational complexity significantly, since clustering itself is time-consuming. It appears that caching the clustering results (Section 3.6.5) is the most practical solution. This not only sidesteps the "erraticity" problem, but also reduces the computational load.

One may perhaps worry that the intermediate results that are cached may nevertheless be subject to the idiosyncrasies of that particular run of the clustering algorithm. However, cross validation will not be affected much; the values of M_c are located sparsely, since M_c is chosen from a logarithmic scale (Section 3.6.3). Hence, the clustering results for each value of M_c should be quite different from those produced using other values of M_c.

3.3.2 Empty Clusters

Note that M and M_c may differ ($M \le M_c$); empty clusters may occur, especially when the number of members per cluster is low ($N / M_c \to 1$). Figure 3.4 illustrates the possibility of empty clusters occurring for *k*-means. Depending on the initial conditions, and how the clustering algorithm is initialized, it is possible to construct some examples whereby empty clusters may occur. If 2 centers are identical, members from both groups will be allocated to one cluster, and the other cluster will become empty.

A partition matrix at the i_t-th iteration is defined by $\mathbf{P}^{(i_t)} \triangleq \left(b_{ij}\right)_{N \times M_c}$, where b_{ij} indicates membership of the *i*-th point (*i*-th row) with respect to the *j*-th center (*j*-th column). If the initialization begins with the partition matrix, it is possible to assign memberships in such a way that identical centers result. For example, consider a one-dimensional data set [72 4, 5], to be clustered into 2 groups. Given

$$\mathbf{P}^{(1)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{pmatrix}, \tag{3.27}$$

two centers are formed, each at 3.

If the initialization begins with random, unique points selected as the centers, states with identical centers may appear to be unreachable. Consider an example: suppose it is necessary to cluster the set {3 - ε, 3, 3 + ε} into 2 groups. If ε is some arbitrary number, usually 2 groups will be formed. However, if ε is less than machine precision,

then 2 different centers would not be produced, and in fact an empty cluster occurs. As far as the machine is concerned, the 2 centers are at identical positions.

It may appear that this example is merely an academic exercise, since the occurrence of empty clusters appears to be unlikely. However, if the data set is long enough, different points in an embedding can be arbitrarily close due to transitivity (See Glossary). Hence, 2 clusters which are very close together may result in empty clusters, due to finite precision. There is also the rare possibility whereby the data is perturbed by observational noise such that a few of the points may come close enough.

In Figure 3.4, as long as any 2 centers, j_1 and j_2, coincide on the "critical line" $\mu_{j_1}(i_t) = \mu_{j_2}(i_t)$ at any iteration i_t, 2 identical centers will be formed. If there are M_c centers in the univariate problem, then the log-likelihood function becomes M_c-dimensional. Figure 3.4 can be regarded as a plot of the "state space" of the clustering algorithm, because the "state" of the algorithm is determined by the location of the means. In a M_c-dimensional "state space" plot there can be $N^{lines} = \sum_{i=2}^{M_c} {}^{M_c}C_i = 2^{M_c} - 1 - {}^{M_c}C_1$ "critical lines" which may result in empty clusters. Theoretically, the chance of any trajectory starting in these lines is of probability measure 0. However, the trajectory does not have to start on any of these lines; the trajectory just has to reach any of them. Besides the local maxima, the "critical lines" are also "attractors". It is not necessary for the trajectory to touch any of the lines; it is suffcient for any of the local maxima to attract the trajectory such that it touches a line. Furthermore, due to finite precision, the trajectory only needs to reach a distance of epsilon (limit of machine precision) away from the "critical line".

Symmetry of $l(\boldsymbol{\mu})$ in Figure 3.3 results from the fact that if a local maxima exists at $\boldsymbol{\mu} = \begin{pmatrix} \mu_a \\ \mu_b \end{pmatrix}$, then there should be another one near $\boldsymbol{\mu} = \begin{pmatrix} \mu_b \\ \mu_a \end{pmatrix}$. The two possibilities is a consequence of the fact that there are 2 different arrangements in the way the groups can be labelled, provided $\mu_a \neq \mu_b$. The first cluster can have mean μ_a and the second cluster can have mean μ_b or vice versa. If there are M_c centers, symmetry implies that there can be $M_c!$ local maxima. This also implies a lot of saddle points, and the possibility that clustering will not converge at any local maxima is relatively high. Nonetheless, the large number of local maxima means that there is a chance that one of them will be near enough to a "critical line" to attract a trajectory.

Note that for the case whereby each $\boldsymbol{\mu}_j$ is a p dimensional vector $\begin{pmatrix} \mu_j^1 & \cdots & \mu_j^p \end{pmatrix}^T$, it may be simpler to consider trajectories in each dimension. Since convergence is achieved only if the trajectories converge in each dimension, this means that convergence is achieved if the same trajectory converges in all of the p "state space" plots (each is M_c-dimensional).

Once an empty cluster is encountered, it may be dropped, *i.e*. the number of centers is reduced by 1, and clustering resumes. Since this does not happen too frequently, usually $M \approx M_c$. If Singular Value Decomposition (SVD) (Section 3.5.2) is used for solving the least squares stage, the issue of empty clusters can be side-stepped.

Alternatively, another way to deal with the empty cluster is to find a point which is farthest from its centroid, and use it to form the nucleus of a new cluster and continue

clustering from there. Thus, the number of clusters will be conserved. This method is implemented in some commercial implementations, such as in MATLAB©.

At first glance, it may seem that fuzzy clustering methods are immune to the occurrence of empty clusters. However, since it is simpler to have one covariance matrix associated with each point, the fuzzy partition matrix is usually converted into a hard partition matrix. The simplest way to do this is to assign each point to the group whereby the point has maximum membership. Unfortunately, this winner-takes-all approach may also result in empty clusters, since there may be groups which do not "win" any point. A simple solution is to reallocate points which have the highest memberships for that empty cluster, away from their actual groups, provided they do not empty their group in the process. A different perspective is to tolerate empty clusters, and to remove them after clustering is completed.

3.3.3 Hierarchical Clustering

Hierarchical Clustering [94] is an alternative to *k*-means clustering and related approaches. For the agglomerative approach, each cluster attracts new members and snowballs until the algorithm halts. Conversely, for the divisive approach, the whole data is divided into separate groups, and these groups are further subdivided, a little like an amoeba which is splitting. In this work, only the agglomerative approach is studied.

The criterion for deciding membership of points in any cluster is distance; it is possible to use the Euclidean norm and the Mahalanobis norm as before. For the

method of deciding membership called single linkage, the nearest neighbour between groups constitutes the distance between clusters.

Consider the distance matrix $\mathbf{D}^{(i_t)} \triangleq \left(\rho_{i_1 i_2}\right)_{N\times N}$, where i_t is the iteration number and $\rho_{i_1 i_2}$ is the distance between the i_1-th point and the i_2-th point. For example,

$$\mathbf{D}^{(1)} = \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{matrix} \begin{pmatrix} 0 & & & & \\ 2 & 0 & & & \\ 6 & 5 & 0 & & \\ 10 & 9 & 4 & 0 & \\ 9 & 8 & 5 & 3 & 0 \end{pmatrix}. \tag{3.28}$$

There are 5 data points ($N = 5$), with the corresponding labels at the left of the matrix. The smallest distance is that between points 1 and 2; hence these are joined to form a cluster. The distance between this cluster and point 3 is $\min\left(\mathbf{D}^{(1)}_{1,3}, \mathbf{D}^{(1)}_{2,3}\right) = 5$, where the subscripts of $\mathbf{D}^{(i_t)}_{i_1,i_2}$ refer to the i_1-th row and i_2-th column of $\mathbf{D}^{(1)}$. Similarly, the distance from this cluster to point 4 is 9, and the distance to point 5 is 8. A new distance matrix $\mathbf{D}^{(2)}$ is formed, consisting of distance between points and between point(s) and center(s):

$$\mathbf{D}^{(2)} = \begin{matrix} \{1,2\} \\ 3 \\ 4 \\ 5 \end{matrix} \begin{pmatrix} 0 & & & \\ 5 & 0 & & \\ 9 & 4 & 0 & \\ 8 & 5 & 3 & 0 \end{pmatrix}. \tag{3.29}$$

The next iteration results in:

$$\mathbf{D}^{(3)} = \begin{matrix} \{1,2\} \\ 3 \\ \{4,5\} \end{matrix} \begin{pmatrix} 0 & & \\ 5 & 0 & \\ 8 & 4 & 0 \end{pmatrix}. \tag{3.30}$$

Note that entries of $\mathbf{D}^{(i_t)}$ consist of distance(s) between points, distance(s) between point(s) and center(s), and distance(s) between centers. Individual points or groups are

gradually grouped together. The process is repeated until all the groups are finally grouped together, in a tree structure called a dendrogram (See Figure 3.5).

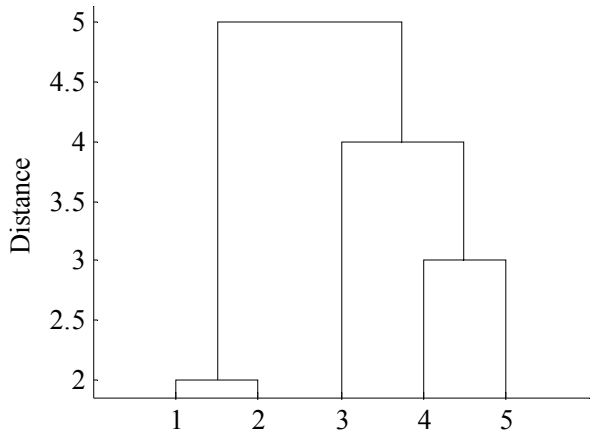

Figure 3.5 Dendrogram of clustering using single linkage.

Besides single linkage, another possibility of deciding membership is Ward's linkage, which is based on minimizing the increase in the total within cluster error sum of squares. Ward's method tends to find (or create) clusters of relatively equal sizes and shapes.

The objective during each iteration is to minimize $\sum_{j=1}^{M_c} ESS_j$, the increase in total within-cluster error sum of squares [94]. For each cluster,

$$ESS_j \triangleq \sum_{i=1}^{N_j} \left\| \boldsymbol{\psi}_i - \boldsymbol{\mu}_j \right\|^2, \tag{3.31}$$

where N_j is the number of points in the j-th cluster, and $\boldsymbol{\mu}_j$ is the centroid of the same cluster. Instead of Euclidean distance, the following quantity is used to indicate the distance between clusters:

$$\frac{N_{j_1} N_{j_2}}{N_{j_1} + N_{j_2}} \left\| \boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2} \right\|^2, \tag{3.32}$$

where the j_1-th cluster has N_{j_1} points with $\boldsymbol{\mu}_{j_1}$ as the centroid and the j_2-th cluster has N_{j_2} points with $\boldsymbol{\mu}_{j_2}$ as the centroid.

Hierarchical clustering offers a deterministic outcome, and is thus not subject to the idiosyncrasies of erraticity. It is also immune to the problem of empty clusters. However, the problem is that it is irrevocable; it is not possible to merge clusters once they are split, or to divide clusters once they are formed. Consider a set {-2.2, -2, -1.8, -0.1, 0.1, 1.8, 2, 2.2}. The data contains 3 obvious clusters, but the first split would form 2 separate clusters, {-2.2, -2, -1.8, -0.1} and {0.1, 1.8, 2, 2.2}. There is no way to form the set {-0.1, 0.1}. Nonetheless, hierarchical clustering appears to offer an interesting alternative.

3.3.4 Other Alternatives

The Expectation Maximization (EM) algorithm may also be used as a clustering method [95]. Actually, the EM and FCM algorithms are related [96]. Since the *k*-means algorithm is often employed to initialize the EM algorithm (see Appendix D), the outcome of the EM algorithm can be non-deterministic.

Yet another alternative is the Self Organizing Map (SOM) [97], which may also be used for the clustering stage. Since the SOM is usually initialized by the assignment of usually random weights, the outcome of the SOM can also be non-deterministic. Incidentally, an early version of *k*-means clustering is closely related to the SOM algorithm [98].

3.4 Basis Functions

Each center is assigned a basis function. Gaussian basis functions are used throughout the simulations:

$$\phi_j\left(\rho_{ij}\right) = e^{-h^{NF}\rho_{ij}^2}, \tag{3.33}$$

where ρ_{ij} is the norm between the *i*-th point, $\boldsymbol{\psi}_i$, and the *j*-th center, $\boldsymbol{\mu}_j$, and h^{NF} is a normalizing factor. Usually the normalizing factor for Gaussian kernels is $h^{NF} = \frac{1}{2}$. In this work, the prescription suggested by Haykin [98] is followed:

$$h^{NF} = \frac{M}{d_{\max}^2}, \tag{3.34}$$

where $d_{\max}$ is the maximum Euclidean distance between the centers. Eq. (3.34) ensures that the individual basis functions are not too peaked or too flat. The choice of normalizing factor deserves some mention, because it cannot be simply assumed that the change in the normalizing factor will be absorbed by the weights. When there are multiple centers, approximation (3.35) holds only when all other centers and weights can be ignored, *i.e.* the other centers are very far away, and the weights are of reasonable size (*i.e.* true with weight regularization):

$$\hat{y}_{i+1} \approx w_0 + w_j\left(e^{-\rho_{ij}^2}\right)^{h^{NF}}. \tag{3.35}$$

3.4.1 Choice of Norm for Inputs

Typically, the Euclidean norm is used for ordinary RBFs. Park and Sandberg [99] extended RBFs to the class of EBFs with diagonal norm-inducing matrices (called Diagonal norm-inducing matrix Basis Functions or DBFs in this work), and proved that they had universal approximation capabilities. The norm in this case is

$$\rho_{ij} = \sqrt{(\boldsymbol{\psi}_i - \boldsymbol{\mu}_j)^T \left[\operatorname{diag}\left(\sigma_{1j}^{-2} \quad \sigma_{2j}^{-2} \quad \cdots \quad \sigma_{pj}^{-2} \right) \right] (\boldsymbol{\psi}_i - \boldsymbol{\mu}_j)} \,, \tag{3.36}$$

where σ_{pj} is the "smoothing factor" in the p-th coordinate for the j-th kernel, and the diagonal matrix is $p \times p$.

A more general form of the norm ρ_{ij} would be

$$\rho_{ij} = \sqrt{(\boldsymbol{\psi}_i - \boldsymbol{\mu}_j)^T \boldsymbol{\Lambda}_j (\boldsymbol{\psi}_i - \boldsymbol{\mu}_j)} \,, \tag{3.37}$$

where $\boldsymbol{\Lambda}_j$ is the j-th norm-inducing matrix. According to Park [100], the issue of universal approximation for non-diagonal norm-inducing matrices appears to be an open problem. Nonetheless, it seems reasonable to conjecture that most of the theorems in Ref. [99] could be applied to positive definite norm-inducing matrices, especially since positive definite basis functions have been discussed in Ref. [101], for the interpolation problem. This suggests the Mahalanobis norm (Appendix C), where $\boldsymbol{\Lambda}_j = \mathbf{M}_j^{-1}$ and $\mathbf{M}$ is symmetric positive definite. The use of the Mahalanobis norm results in EBFs which are unconstrained in their orientations.

3.4.2 Data Driven Basis Functions

The universal approximation theorem [77, 99] states that a suitable RBF or DBF can approximate any function, but is silent about the methods required to find the parameters of the RBF or DBF. It is not feasible to tune all elements of all $\boldsymbol{\Lambda}_j$: $\{\boldsymbol{\Lambda}_j\}_{j=1}^{M}$; each $\boldsymbol{\Lambda}_j$ has $\frac{p(p+1)}{2}$ unique elements, resulting in a total of $\frac{Mp(p+1)}{2}$ elements to tune. If each $\boldsymbol{\Lambda}_j$ is constrained to be diagonal, there would still be Mp elements. Computational complexity would still be a problem in the case of chaotic

time series prediction; usually $M \geq 100$ and $p \geq 3$. Hence, some way to determine the elements of $\mathbf{\Lambda}_j$ cheaply is required.

At each i_t-th iteration of the clustering process, an $N \times M_c$ partition matrix $\mathbf{P}^{(i_t)} \triangleq \left(b_{ij}\right)_{N \times M_c}$ is available. Usually, the information in $\mathbf{P}^{(i_t)}$ is not used, because only the locations of the centers are of interest. Utilizing the information in $\mathbf{P}^{(i_t)}$ results in a "data-driven" approach, which allows one to estimate the statistical properties of each cluster, *i.e*. covariance and higher order statistics, and to use these properties to adjust the basis functions. It is very likely that the solution obtained by a data driven approach is sub-optimal. If data points are distributed randomly and uniformly, there is little advantage of clustering algorithms over mere random selection of centers for the RBF networks. However, this would not be true for a chaotic system; the presence of an attractor ensures that state space is not homogeneous. The clustering process would produce a variety of clusters of varying sizes. If $\left\|\boldsymbol{\psi}_i - \boldsymbol{\psi}_{i+1}\right\| \to 0$, this implies $\left\|y_i - y_{i+1}\right\| \to 0$, since $f(\bullet)$ is continuous for chaotic systems. Small, concentrated clusters correspond to clusters with low variance. If these clusters are associated with basis functions which decay rapidly with distance, they may be useful for modelling fine details in the function being approximated. On the other hand, basis functions which decay slowly with distance would be useful for regions which do not require much detail; otherwise too many basis functions would be required.

This suggests the use of basis functions with varying widths, by equating each $\mathbf{M}_j$ to a matrix $\mathbf{C}_j$ which is determined by the data. A possibility is to have

$$\mathbf{C}_j = \frac{\mathrm{tr}\left(\mathbf{S}_j\right)}{p}\mathbf{I}, \tag{3.38}$$

where $\mathbf{S}_j$ is the sample covariance matrix of the j-th cluster. This produces "spherical" basis functions; each basis function is associated with a different width. In the one-dimensional case, Eq. (3.38) reduces to a variance term. The networks which use (3.38) for the norm inducing matrix are called Trace Basis Functions (TBFs).

A natural extension is to consider basis functions with varying orientations (each cluster is associated with a different covariance matrix). One possibility is to use the full sample covariance matrix:

$$\mathbf{C}_j = \mathbf{S}_j. \tag{3.39}$$

Another possibility is

$$\mathbf{C}_j = \mathrm{diag}\left(\mathbf{S}_j\right). \tag{3.40}$$

The networks employing Eq. (3.40) result in DBFs.

3.4.3 Regularized Covariance Matrices

Positive definite matrices have positive determinants and are invertible; in contrast, each $\mathbf{S}_j$ is positive semidefinite (see Appendix B) and thereby possibly singular. If any resulting $\mathbf{\Lambda}_j$ is singular, it may confound the linear layer of the EBF. Hence, it is advisable to check if $\mathbf{M}_j^{-1}$ is singular. Since SVD can be used for checking the condition of a matrix and also for matrix inversion, one single call to SVD suffices for each $\mathbf{M}_j^{-1}$. Thus, checking for singularity is an extra burden which is negligible. Furthermore, there may be matrices where there are only a few points, which are automatically singular. If one simply substitutes an identity matrix, this obviates the

need for extra computations. Hence, there will be a combination of "hyperspheres" and "hyperellipsoids".

Nonetheless, it is also possible to regularize $\mathbf{M}_j$, similar to the way sample covariance matrices are regularized in regularized discriminant analysis by the addition of $\mathbf{I}$ multiplied by a constant $\gamma_b \in [0,1]$ [102]. Thus, $\mathbf{M}_j$ becomes:

$$\mathbf{M}_j = (1-\gamma_b)\mathbf{C}_j + \gamma_b \mathbf{I}\,. \tag{3.41}$$

Another way to regularize the matrices is suggested by the fact that since SVD can be used for inverting an ill-conditioned matrix, the inverted matrix is not necessary well conditioned. Thus it is also logical to regularize $\mathbf{M}_j^{-1}$, resulting in

$$\mathbf{\Lambda}_j = (1-\gamma_a)\mathbf{M}_j^{-1} + \gamma_a \mathbf{I}\,, \tag{3.42}$$

where $\gamma_a \in [0,1]$. The addition of $\gamma_a \mathbf{I}$ in Eq. (3.42) forces $\mathbf{\Lambda}_j$ to be positive definite (see Appendix B) and makes the "hyperellipsoids" more "spherical". Note that the combination

$$\mathbf{\Lambda}_j = (1-\gamma_a)\left((1-\gamma_b)\mathbf{C}_j + \gamma_b \mathbf{I}\right)^{-1} + \gamma_a \mathbf{I} \tag{3.43}$$

is excessive, since the inverse of a positive definite matrix is also a positive definite matrix (see Appendix B).

3.5 Linear Layer

The linear layer of the RBF or the EBF refers to the layer which performs a weighted sum of the outputs from the various basis functions and the bias. One could use Gaussian elimination, SVD [103], or conjugate gradient method as applied to linear systems [104, 105] to find the weights in the linear layer, *i.e.* solve the linear system:

$$\mathbf{w} = \left(\mathbf{G}^T\mathbf{G}\right)^{-1}\mathbf{G}^T\mathbf{y}\,, \tag{3.44}$$

where **w** is the weight vector and $\mathbf{G} \triangleq \left(\phi(\rho_{ij})\right)_{N\times M}$.

3.5.1 Computational Complexity

For the interpolation case, *i.e.* $N = M$, the linear layer has computational complexity of O(N^3). If $N \gg M$, clustering reduces the computational complexity of the linear layer to O(M^3). The computational complexity required is only $\left(M/N\right)^3$ of what it would be originally, provided the overhead induced by clustering is ignored. However, some clustering methods, such as hierarchical clustering, have computational complexity of O(N^3), but state of the art modifications can reduce it to O(N^2) [106]. Thus, it is not true that clustering always speeds up RBFs, since this is implementation specific.

Also, it is not always true that clustering will solve the problem of the curse of dimensionality (see Glossary), because clustering algorithms are themselves prone to it. For example, each iteration of the *k*-means algorithm requires O(M_cNp) multiplications to evaluate the squared Euclidean distance. The square root is unnecessary; see Eq. (3.87). The curse of dimensionality means that the number of

centers required to model a function increases exponentially as $M_c = m^p$, where $m \in \mathbb{R}^+$ is an arbitrary constant. The number of points required to sample the data also increases exponentially, $N = n^p$, where $n \in \mathbb{R}^+$ is an arbitrary constant. Hence, the actual complexity of each iteration is O($(mn)^p p$). Furthermore, the calculation of relative distances becomes error prone as dimensions increase, as there is poor discrimination between different neighbouring points [63].

Assuming that the computational complexity of the hidden layer is $\mathrm{O}(M^3)$, and since

$$(M-1)^3 = M^3 - 3M^2 + 3M - 1, \tag{3.45}$$

this implies that removing one neuron lightens the computational burden of the hidden layer by a O(M^2) term. This is overshadowed by the O(M^3) term, but it does suggest that pre-processing the input signal so that it is zero mean, not only simplifies the architecture, but also lightens the computational burden of the hidden layer slightly.

3.5.2 SVD

The Singular Value Decomposition (SVD) of a $M \times M_r$ matrix **A**, with rank M_r, is

$$\begin{aligned} \mathbf{A} &= \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T \\ &= \begin{pmatrix} \mathbf{U}_1 & \mathbf{U}_2 \end{pmatrix} \begin{pmatrix} \boldsymbol{\Sigma}_1 \\ \mathbf{0} \end{pmatrix} \mathbf{V}^T, \end{aligned} \tag{3.46}$$

where **U** is an $M \times M$ orthogonal matrix, **Σ** is an $M \times M_r$ matrix, **V** is an $M_r \times M_r$ orthogonal matrix and $\mathbf{U}_1$ is $M \times M_r$, such that the diagonal matrix

$$\boldsymbol{\Sigma}_1 = \begin{pmatrix} r_1 & & 0 \\ & \ddots & \\ 0 & & r_{M_r} \end{pmatrix} \tag{3.47}$$

is $M_r \times M_r$ and non-singular. The nonnegative square roots of the eigenvalues of $\mathbf{A}^T\mathbf{A}$ are the singular values. The singular values make up the diagonal entries of $\mathbf{\Sigma}_1$: r_1 to r_{M_r}. The columns of $\mathbf{U}$ are the orthonormal eigenvectors of $\mathbf{A}\mathbf{A}^T$ and the columns of $\mathbf{V}$ are the orthonormal eigenvectors of $\mathbf{A}^T\mathbf{A}$.

The solution to the linear system $\mathbf{Aw} = \mathbf{b}$ is given by

$$\mathbf{w} = \mathbf{V}\mathbf{\Sigma}_1^{-1}\mathbf{U}_1^T\mathbf{b}. \tag{3.48}$$

Usually, SVD is computed by some variant of QR iteration, and hence has $O(M^3)$ computational complexity [104]. Note that SVD was used by Broomhead and Lowe [107].

In versions of MATLAB© 5, the implementation of SVD is less robust than in version 6; on large data sets, SVD could fail for ill conditioned matrices. Thus, one possibility is to use conjugate gradient for linear systems.

3.5.3 Conjugate Gradient for Linear Systems

Define a quadratic function

$$\zeta(\mathbf{w}) \triangleq \frac{1}{2}\mathbf{w}^T\mathbf{A}\mathbf{w} - \mathbf{w}^T\mathbf{b}, \tag{3.49}$$

where $\mathbf{A}$ is a $M \times M$ symmetric positive definite matrix. Hence,

$$\frac{\partial\zeta(\mathbf{w})}{\partial\mathbf{w}} = \nabla\zeta(\mathbf{w}) = \mathbf{A}\mathbf{w} - \mathbf{b}. \tag{3.50}$$

The quadratic function $\zeta(\mathbf{w})$ attains a minimum precisely when $\frac{\partial\zeta(\mathbf{w})}{\partial\mathbf{w}} = 0$, resulting in $\mathbf{Aw} = \mathbf{b}$. It is possible to see $\zeta(\mathbf{w})$ as a scalar field, and thus the use of ∇ is natural.

Thus, it is possible to apply unconstrained optimization techniques to obtain a solution to the linear system **Aw** = **b**. Most multi-dimensional optimization methods progress from one iteration to the next by performing one-dimensional search along some search direction $\mathbf{s}_k$, so that

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha \mathbf{s}_k , \tag{3.51}$$

where α is a line search parameter that is chosen to minimize the objective function $\zeta(\mathbf{w}_k + \alpha \mathbf{s}_k)$ along $\mathbf{s}_k$.

Note some special features of such a quadratic optimization problem. Firstly, the negative gradient is just the residual vector:

$$-\nabla \zeta(\mathbf{w}) = \mathbf{b} - \mathbf{A}\mathbf{w} = \mathbf{r} . \tag{3.52}$$

Secondly, for any search direction, $\mathbf{s}_k$, it is unnecessary to perform a line search, because the optimal choice for α can be determined analytically. Specifically, the minimum over α occurs when the new residual is orthogonal to the search direction:

$$\begin{aligned} 0 &= \frac{d}{d\alpha} \zeta(\mathbf{w}_{k+1}) \\ &= \nabla \zeta(\mathbf{w}_{k+1})^T \frac{d}{d\alpha} \mathbf{w}_{k+1} \\ &= \left(\mathbf{A}\mathbf{w}_{k+1} - \mathbf{b}\right)^T \frac{d}{d\alpha} \left(\mathbf{w}_k + \alpha \mathbf{s}_k\right) \\ &= -\mathbf{r}_{k+1}^T \mathbf{s}_k . \end{aligned} \tag{3.53}$$

Since the new residual can be expressed in terms of the old residual and the search direction,

$$\begin{aligned} \mathbf{r}_{k+1} &= \mathbf{b} - \mathbf{A}\mathbf{w}_{k+1} \\ &= \mathbf{b} - \mathbf{A}(\mathbf{w}_k + \alpha \mathbf{s}_k) \\ &= (\mathbf{b} - \mathbf{A}\mathbf{w}_k) - \alpha \mathbf{A}\mathbf{s}_k \\ &= \mathbf{r}_k - \alpha \mathbf{A}\mathbf{s}_k , \end{aligned} \tag{3.54}$$

and subsituting Eq. (3.53) into Eq. (3.54), it is possible to solve for $\alpha = \frac{\mathbf{r}_k^T \mathbf{s}_k}{\mathbf{s}_k^T \mathbf{A}\mathbf{s}_k}$.

If these properties are used in specializing the conjugate gradient method for unconstrained optimization, the conjugate gradient method for solving symmetric positive definite linear systems is obtained.

It appears easy to apply the steepest descent method, using the negative gradient – in this case the residual – as search direction at each iteration. Unfortunately, the convergence rate of steepest descent is often very poor, due to repeated searches in the same directions. One possibility is to orthogonalize each new search direction against all of the previous ones (Gram Schmidt orthogonalization), leaving only components in "new" directions. However, this is prohibitively expensive computationally and also requires excessive storage to save all of the search directions. However, if the search directions are made mutually **A**-orthogonal (vectors **y** and **z** are **A**-orthogonal if $\mathbf{y}^T \mathbf{A}\mathbf{z} = 0$), instead of using the standard inner product, then it can be shown that the successive **A**-orthogonal search directions satisfy a three-term recurrence (this is the role played by β, which is defined on the next page). This short recurrence makes the computation very cheap, and also makes it unnecessary to save all of the previous gradients, only the most recent two. The algorithm is given below:

Set *maxit* to predefined value, *i.e*. 10000

Set *tol* to predefined value, *i.e*. limited by finite precision

$\mathbf{w}_0$ = initial guess, perhaps **b**

$\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{w}_0$

$\mathbf{s}_0 = \mathbf{r}_0$

$k = 0$

Repeat while $k < maxit$ and $\|\mathbf{r}_{k+1}\| > tol$

$$\alpha = \frac{\mathbf{r}_k^T \mathbf{s}_k}{\mathbf{s}_k^T \mathbf{A}\mathbf{s}_k} \text{ (compute search parameter)}$$

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha \mathbf{s}_k \text{ (update solution)}$$

$$\mathbf{r}_{k+1} = \mathbf{r}_k - \alpha \mathbf{A}\mathbf{s}_k$$

$$\beta_{k+1} = \frac{\mathbf{r}_{k+1}^T \mathbf{r}_{k+1}}{\mathbf{r}_k^T \mathbf{r}_k}$$

$$\mathbf{s}_{k+1} = \mathbf{r}_{k+1} + \beta_{k+1} \mathbf{s}_k \text{ (compute new search direction)}$$

Increment k by 1

end

Each iteration of the algorithm requires only a single matrix-vector multiplication, $\mathbf{A}\mathbf{s}_k$ (which is of complexity O(M^2); see Ref. [108]), plus a small number of inner products. The storage requirements are also very modest, since the vectors **w**, **r**, and **s** can be overwritten on successive iterations.

3.6 Bias Variance Dilemma

What is the bias-variance dilemma? The famous paper by Geman *et al.* [109] offers a detailed explanation.

Let Y be the random variable associated with the observed variable y_i, and X be the random variable associated with the input $\boldsymbol{\psi}_i$. For any $\boldsymbol{\psi}_i$ drawn from the training set, $E\left[Y \mid X = \boldsymbol{\psi}_i\right]$ is the conditional expectation of Y given $\boldsymbol{\psi}_i$, *i.e.* the average of Y taken with respect to the conditional distribution $p(Y \mid X)$. Assume the observational noise $\eta_i = y_i - \breve{y}_i$ is zero mean. Hence, define the true value $\breve{y}_i$ as

$$\breve{y}_i \triangleq E\left[Y \mid X = \boldsymbol{\psi}_i\right]. \tag{3.55}$$

Define the estimated value $\hat{y}_i$ to be a function of $\boldsymbol{\psi}_i$ and dependent upon the particular realization of the training set $\mathcal{D}$:

$$\hat{y}_i \triangleq \hat{f}\left(\boldsymbol{\psi}_i; \mathcal{D}\right). \tag{3.56}$$

The model error is given as $e_i = y_i - \hat{y}_i$. The expectation of e_i^2 can be described as

$$\begin{aligned} E_{\mathcal{D}}\left[\left(y_i - \hat{y}_i\right)^2\right] &= E_{\mathcal{D}}\left[\left(y_i - \breve{y}_i + \breve{y}_i - \hat{y}_i\right)^2\right] \\ &= E_{\mathcal{D}}\left[\left(y_i - \breve{y}_i\right)^2\right] + E_{\mathcal{D}}\left[\left(\breve{y}_i - \hat{y}_i\right)^2\right] + 2E_{\mathcal{D}}\left[\left(\breve{y}_i - \hat{y}_i\right)\left(y_i - \breve{y}_i\right)\right] \\ &= \mathrm{var}_{\mathcal{D}}\left[\eta_i\right] + E_{\mathcal{D}}\left[\left(\breve{y}_i - \hat{y}_i\right)^2\right] \\ &\quad + 2\left(E_{\mathcal{D}}\left[\breve{y}_i y_i\right] - E_{\mathcal{D}}\left[\breve{y}_i^2\right] - E_{\mathcal{D}}\left[\hat{y}_i y_i\right] + E_{\mathcal{D}}\left[\hat{y}_i \breve{y}_i\right]\right), \end{aligned} \tag{3.57}$$

where $E_{\mathcal{D}}\left[\bullet\right]$ represents expectation with respect to the training set $\mathcal{D}$, and similarly, $\mathrm{var}_{\mathcal{D}}\left[\bullet\right]$ represents variance with respect to the training set $\mathcal{D}$.

Note that $2\left(E_{\mathcal{D}}\left[\breve{y}_i y_i\right]-E_{\mathcal{D}}\left[\breve{y}_i^2\right]-E_{\mathcal{D}}\left[\hat{y}_i y_i\right]+E_{\mathcal{D}}\left[\hat{y}_i \breve{y}_i\right]\right)=0$, because the 1st and 2nd terms cancel each other out, and similarly for the 3rd and 4th terms.

$E_{\mathcal{D}}\left[\breve{y}_i y_i\right]=E_{\mathcal{D}}\left[\breve{y}_i(\breve{y}_i+\eta_i)\right]=E_{\mathcal{D}}\left[\breve{y}_i^2\right]$, since $\breve{y}_i$ should be independent of observational noise. Also, $E_{\mathcal{D}}\left[\hat{y}_i y_i\right]=E_{\mathcal{D}}\left[\hat{y}_i(\breve{y}_i+\eta_i)\right]=E_{\mathcal{D}}\left[\hat{y}_i \breve{y}_i+\hat{y}_i \eta_i\right]=E_{\mathcal{D}}\left[\hat{y}_i \breve{y}_i\right]$.

Note that since $\hat{y}_i$ should be independent of observational noise, $E_{\mathcal{D}}\left[\hat{y}_i \eta_i\right]=0$.

$$
\begin{aligned}
E_{\mathcal{D}}\left[\left(\breve{y}_i-\hat{y}_i\right)^2\right] &= E_{\mathcal{D}}\left[\left(\breve{y}_i-E_{\mathcal{D}}\left[\hat{y}_i\right]+E_{\mathcal{D}}\left[\hat{y}_i\right]-\hat{y}_i\right)^2\right] \\
&= E_{\mathcal{D}}\left[\left(\breve{y}_i-E_{\mathcal{D}}\left[\hat{y}_i\right]\right)^2\right]+E_{\mathcal{D}}\left[\left(E_{\mathcal{D}}\left[\hat{y}_i\right]-\hat{y}_i\right)^2\right] \\
&\quad +2E_{\mathcal{D}}\left[\left(\breve{y}_i-E_{\mathcal{D}}\left[\hat{y}_i\right]\right)\left(E_{\mathcal{D}}\left[\hat{y}_i\right]-\hat{y}_i\right)\right] \\
&= E_{\mathcal{D}}\left[\left(\breve{y}_i-E_{\mathcal{D}}\left[\hat{y}_i\right]\right)^2\right]+\mathrm{var}_{\mathcal{D}}\left[\hat{y}_i\right] \\
&\quad +2\left(E_{\mathcal{D}}\left[\breve{y}_i E_{\mathcal{D}}\left[\hat{y}_i\right]\right]-E_{\mathcal{D}}\left[\breve{y}_i \hat{y}_i\right]-E_{\mathcal{D}}\left[\left(E_{\mathcal{D}}\left[\hat{y}_i\right]\right)^2\right]+E_{\mathcal{D}}\left[\hat{y}_i E_{\mathcal{D}}\left[\hat{y}_i\right]\right]\right).
\end{aligned}
\tag{3.58}
$$

$2\left(E_{\mathcal{D}}\left[\breve{y}_i E_{\mathcal{D}}\left[\hat{y}_i\right]\right]-E_{\mathcal{D}}\left[\breve{y}_i \hat{y}_i\right]-E_{\mathcal{D}}\left[\left(E_{\mathcal{D}}\left[\hat{y}_i\right]\right)^2\right]+E_{\mathcal{D}}\left[\hat{y}_i E_{\mathcal{D}}\left[\hat{y}_i\right]\right]\right)=0$, because the 1st and 2nd terms cancel each other out, and similarly for the 3rd and 4th terms.

$E_{\mathcal{D}}\left[\breve{y}_i E_{\mathcal{D}}\left[\hat{y}_i\right]\right]=\breve{y}_i E_{\mathcal{D}}\left[\hat{y}_i\right]$ since $\breve{y}_i$ is deterministic.

$E_{\mathcal{D}}\left[\breve{y}_i \hat{y}_i\right]=\breve{y}_i E_{\mathcal{D}}\left[\hat{y}_i\right]$, since $\breve{y}_i$ is deterministic.

$E_{\mathcal{D}}\left[\left(E_{\mathcal{D}}\left[\hat{y}_i\right]\right)^2\right]=\left(E_{\mathcal{D}}\left[\hat{y}_i\right]\right)^2$.

$E_{\mathcal{D}}\left[\hat{y}_i E_{\mathcal{D}}\left[\hat{y}_i\right]\right]=\left(E_{\mathcal{D}}\left[\hat{y}_i\right]\right)^2$.

Thus, the model error can be decomposed into

$$
E_{\mathcal{D}}\left[e_i^2\right]=\mathrm{var}_{\mathcal{D}}\left[\eta_i\right]+bias^2+\mathrm{var}_{\mathcal{D}}\left[\hat{y}_i\right], \tag{3.59}
$$

where $bias = \breve{y}_i - E_{\mathcal{D}}\left[\hat{y}_i\right]$. Note that the variance of the observational noise $\text{var}_{\mathcal{D}}\left[\eta_i\right]$, cannot be minimized, as it is independent of the neural network. If the output $\hat{y}_i$ was always a constant value, $\text{var}_{\mathcal{D}}\left[\hat{y}_i\right] = 0$, but the bias, *i.e.* the deviation from the ideal $\breve{y}_i$, would be enormous. The bias error is the part of the model error that is due to the restricted flexibility of the model; in reality, most processes are quite complex, and the class of models typically applied are not capable of representing the process exactly. In contrast, if the model is unbiased, *i.e.* *bias* = 0, the variance term can be large if the model is complicated. Hence, there is a trade-off between bias and variance.

It is interesting to note that a somewhat similar relationship exists in statistics, with respect to measurement error [110]. From the definition of variance:

$$E\left[Y^2\right] = \text{var}\left[Y\right] + \left(E\left[Y\right]\right)^2. \tag{3.60}$$

Hence,

$$E\left[\left(Y - y_0\right)^2\right] = \text{var}\left[Y - y_0\right] + \left(E\left[Y - y_0\right]\right)^2. \tag{3.61}$$

where y_0 is the actual value of the point being measured. Using the property:

$$\text{var}\left[aY + b\right] = a^2\,\text{var}\left[Y\right], \tag{3.62}$$

where a and b are arbitrary constants, the following result is obtained:

$$E\left[\left(Y - y_0\right)^2\right] = \text{var}\left[Y\right] + \left(E\left[Y - y_0\right]\right)^2. \tag{3.63}$$

This is similar to Eq. (3.59).

3.6.1 Regularization

Regularization can be used to deal with the bias-variance dilemma in the presence of noise. Essentially, a compromise is sought between model error and a constraint based on prior information. The resultant error functional ζ_γ^2 is given by:

$$\zeta_\gamma^2 = \frac{1}{2}\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2 + \frac{\gamma}{2}\left\|\mathbf{D}\hat{f}\right\|^2, \tag{3.64}$$

where **D** is a linear differential operator [98], $\hat{f}$ is the approximating function and γ is the regularization parameter. The term $\frac{1}{2}\left\|\mathbf{D}\hat{f}\right\|^2$ is the regularizing term which takes into account prior information about the form of the solution by penalizing model complexity. Choose $\left\|\mathbf{D}\hat{f}\right\|^2 = \mathbf{w}^T\mathbf{G}_\gamma\mathbf{w}$, where $\mathbf{G}_\gamma \triangleq \left(\phi(\rho_{j_1 j_2})\right)_{M\times M}$ and $\rho_{j_1 j_2}$ is the distance between the j_1-th center and the j_2-th center. Thus, the linear system in (3.44) becomes

$$\mathbf{w} = \left(\mathbf{G}^T\mathbf{G} + \gamma\mathbf{G}_\gamma\right)^{-1}\mathbf{G}^T\mathbf{y}. \tag{3.65}$$

Note that if the minimum distance between non-identical centers, $\min_{j_1\neq j_2}(\rho_{j_1 j_2}) \to \infty$, then $\mathbf{G}_\gamma \to \mathbf{I}$ for Gaussian basis functions. More generally, if the magnitude of the basis function decays with distance (e.g. L^1 functions), then $\mathbf{G}_\gamma \to c\mathbf{I}$, where c is an arbitrary constant, which means that $\mathbf{w}^T\mathbf{G}_\gamma\mathbf{w} \to c\mathbf{w}^T\mathbf{w}$, which is just weight decay. This approximation could be made, especially when M is small.

However, $\mathbf{G}_\gamma$ could be recycled for each value of γ, as it is independent of γ. Also, in the process of forming $\mathbf{G}_\gamma$, the value of $d_{\max}$ can be recovered as a by-product of calculating each $\rho_{j_1 j_2}$, so there seems little to gain from this approximation.

Nonetheless, note that $\mathbf{G}^T\mathbf{G} + \gamma c\mathbf{I}$ is positive definite and invertible because $\gamma c\mathbf{I}$ is positive definite (see Appendix B). There is no such guarantee for $\mathbf{G}^T\mathbf{G} + \gamma\mathbf{G}_\gamma$, as both $\mathbf{G}^T\mathbf{G}$ and $\gamma\mathbf{G}_\gamma$ might be positive semidefinite. Thus, the advantage of using weight decay is that it is numerically more robust.

It is possible to estimate the value of γ [98]. Define the average squared error over a given data set as

$$R(\gamma) = \frac{1}{N}\sum_{i=1}^{N}\left(f(\boldsymbol{\psi}_i) - F_\gamma(\boldsymbol{\psi}_i)\right)^2, \tag{3.66}$$

where

$$F_\gamma(\boldsymbol{\psi}_i) = \sum_{l=1}^{N} a_{il}(\gamma) y_l \,. \tag{3.67}$$

Here, $F_\gamma(\boldsymbol{\psi}_i)$ is a linear combination of the set of observables, and a function of γ; each $a_{il}(\gamma)$ is a coefficient. This can be expressed in matrix notation as:

$$\mathbf{F}_\gamma = \boldsymbol{\Gamma}(\gamma)\mathbf{y}\,, \tag{3.68}$$

where $\mathbf{F}_\gamma = \begin{bmatrix} F_\gamma(\boldsymbol{\psi}_1) \\ F_\gamma(\boldsymbol{\psi}_2) \\ \vdots \\ F_\gamma(\boldsymbol{\psi}_N) \end{bmatrix}$ and the matrix $\boldsymbol{\Gamma}(\gamma) = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix}$ is called the influence matrix. It is possible to estimate $E[R(\gamma)]$, using

$$\hat{R}(\gamma) = \frac{1}{N}\left\|\left(\mathbf{I} - \boldsymbol{\Gamma}(\gamma)\right)\mathbf{y}\right\|^2 + \frac{\mathrm{var}[\boldsymbol{\eta}]}{N}\mathrm{trace}\left(\boldsymbol{\Gamma}^2(\gamma)\right) - \frac{\mathrm{var}[\boldsymbol{\eta}]}{N}\mathrm{trace}\left(\mathbf{I} - \boldsymbol{\Gamma}^2(\gamma)\right), \tag{3.69}$$

where $\mathbf{A}$ is the influence matrix [98] and $\boldsymbol{\eta}$ is the observational noise of $\mathbf{y}$. Assume $\mathbf{A} = \mathbf{I}$, for the case where the data solely consists of white noise, and substitute into Eq. (3.69):

$$\hat{R}(\gamma) = \frac{\mathrm{var}[\boldsymbol{\eta}]N}{N} = \mathrm{var}[\boldsymbol{\eta}]. \tag{3.70}$$

Eq. (3.70) suggests that for negative SNR, error variance should be close to 1. This is borne out by observation in reality (Section 5.3.2), as the error variance for signals with negative SNR is found to be close to the variance of the noise.

Note that for $\hat{R}(\gamma)$ to be unbiased, the denominator should be N - N^{EF} [111, 112], where

$$N^{EF} = N - \mathrm{trace}(\mathbf{P}). \tag{3.71}$$

P is the projection matrix defined by:

$$\mathbf{Py} \triangleq \mathbf{y} - \mathbf{Gw} = \mathbf{e}. \tag{3.72}$$

Define

$$\mathbf{A} \triangleq \mathbf{G}^T\mathbf{G} + \gamma\mathbf{G}_\gamma. \tag{3.73}$$

Substituting Eq. (3.73) into Eq. (3.65),

$$\mathbf{w} = \mathbf{A}^{-1}\mathbf{G}^T\mathbf{y} \tag{3.74}$$

Substituting Eq. (3.74) into Eq. (3.72),

$$\begin{aligned} \mathbf{Py} &= \mathbf{y} - \mathbf{GA}^{-1}\mathbf{G}^T\mathbf{y} \\ \mathbf{P} &= \mathbf{I}_N - \mathbf{GA}^{-1}\mathbf{G}^T, \end{aligned} \tag{3.75}$$

where $\mathbf{I}_N$ is the identity matrix of size $N \times N$. Substituting Eq. (3.75) into Eq. (3.71),

$$\begin{aligned} N^{EF} &= N - \mathrm{trace}\left(\mathbf{I}_N - \mathbf{GA}^{-1}\mathbf{G}^T\right) \\ &= \mathrm{trace}\left(\mathbf{GA}^{-1}\mathbf{G}^T\right) \\ &= \mathrm{trace}\left(\mathbf{G}^T\mathbf{GA}^{-1}\right). \end{aligned} \tag{3.76}$$

Substituting Eq. (3.73) into Eq. (3.76) gives

$$
\begin{aligned}
N^{EF} &= \operatorname{trace}\left(\left(\mathbf{A}-\gamma\mathbf{G}_{\gamma}\right)\mathbf{A}^{-1}\right) \\
&= \operatorname{trace}\left(\mathbf{I}_{M}-\gamma\mathbf{G}_{\gamma}\mathbf{A}^{-1}\right) \\
&= M-\gamma\operatorname{trace}\left(\mathbf{G}_{\gamma}\mathbf{A}^{-1}\right),
\end{aligned}
\tag{3.77}
$$

where $\mathbf{I}_M$ is the identity matrix of size $M \times M$. In the absence of regularization, $N^{EF} = M$.

3.6.2 Cross Validation

One way to select the hyperparameters, M_c and γ, is to perform k-fold cross validation:

---

1. Split the data set (size N^{total}) into the training set ($\mathcal{D}$ of size N^{train}) and final test set ($\mathcal{T}$ of size N^{test}), *i.e.* $N^{total} = N^{train} + N^{test}$. Typically, $N^{train} = \frac{2}{3} N^{total}$.

2. Split $\mathcal{D}$ into k equal parts: $\left\{\mathcal{D}_{k_L}\right\}_{k_L=1}^{k}$, where $k_L = 1, \cdots, k$ (see Figure 3.6).

3. Train k sub-RBFs using the data other than those in the current $\mathcal{D}_{k_L}$. $\mathcal{D}_{k_L}$ serves as the test set, while the rest of the data is the training set. According to [113], the training set of each sub-RBF can also be called the design set. Each design set has N_ψ^k training examples:

$$N_\psi^k = \frac{k-1}{k} N_\psi^{train}, \tag{3.78}$$

where $N_\psi^{train} = N^{train} - p$.

4. Evaluate the error $GE_{k_L}^{val}$ of the k_L-th sub-RBF on the validation set $\mathcal{D}_{k_L}$; typically $GE_{k_L}^{val}$ is the mean squared error of the estimated function on $\mathcal{D}_{k_L}$.

5. Repeat steps 2-4 for all k_L = 1, …, k, and compute the estimate of the generalization error as:

$$GE \approx \frac{1}{k} \sum_{k_L=1}^{k} GE_{k_L}^{val}. \tag{3.79}$$

6. The optimal hyperparameters are chosen according to $\arg\min_{(M_c,\gamma)} \frac{1}{k} \sum_{k_L=1}^{k} GE_{k_L}^{val}$.

In this work, $k = 5$, unless stated otherwise. Note that Leave One Out (LOO) can be considered to be a type of k-fold cross validation, as it entails $k = N_{\psi}^{train}$, *i.e.* a validation set of just 1 element.

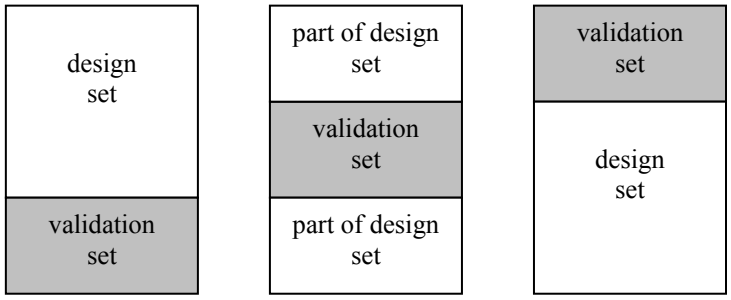

Figure 3.6 Illustration of k-fold cross validation where $k = 3$.

3.6.3 Choice of Hyperparameters

This section describes a rationale to choose hyperparameters from a logarithmic scale; typically, only prescriptions exist in the neural network literature. Some chaotic signals exhibit $1/f$ spectra. For example, a signal extracted from a Lorenz system demonstrates $1/f$ behaviour, as seen from Figure 3.7. AWGN is added to the same Lorenz signal at 25dB SNR, resulting in the spectrum in Figure 3.8. It is possible to apply the principle of superposition and to consider this as the addition of the $1/f$ spectrum and the noise spectrum. Thus, the spectrum slopes down to the noise floor, where the higher frequencies are dominated by noise. The $1/f$ spectrum is due to the presence of fractal structure in the attractor, where finer structure contributes to the higher frequencies, but at a lower magnitude.

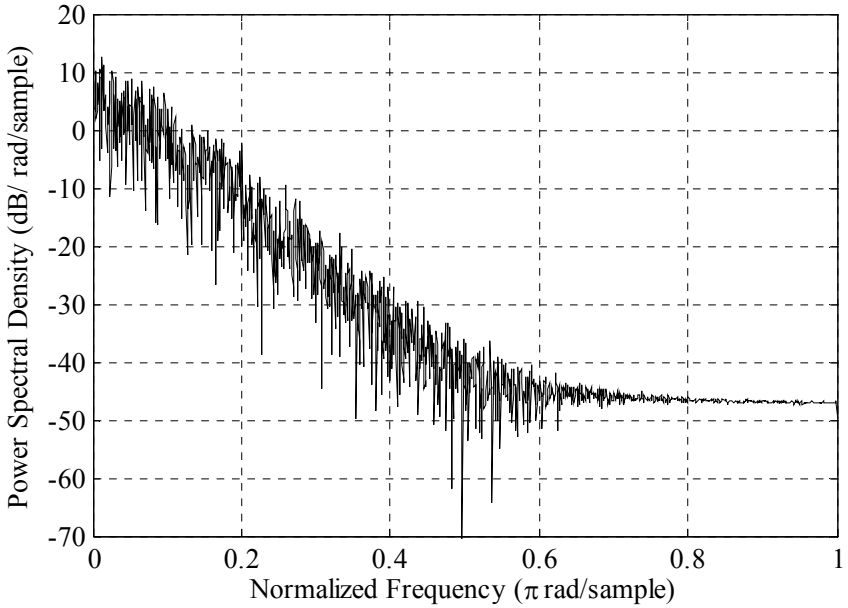

Figure 3.7 Power spectrum of noiseless Lorenz signal demonstrates 1/*f* behaviour.

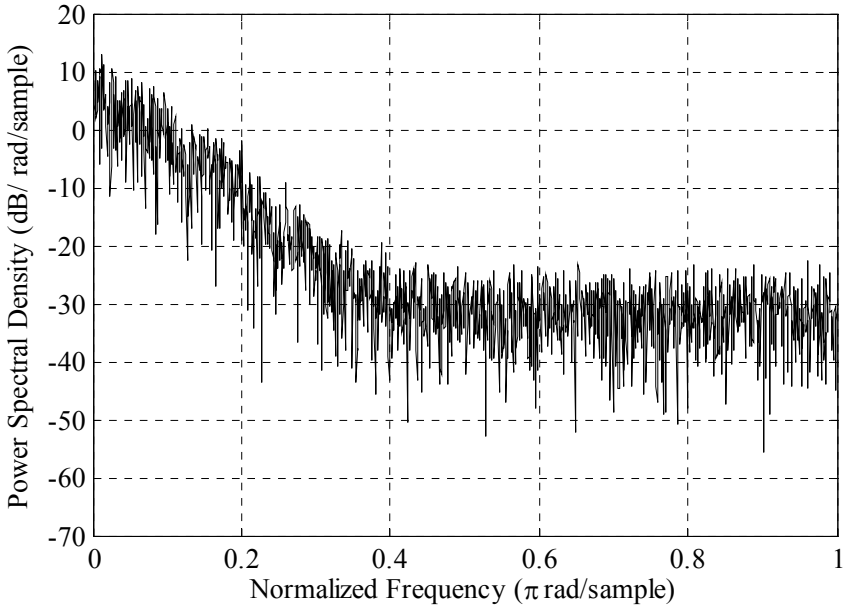

Figure 3.8 Power spectrum of Lorenz signal at 25dB SNR.

Thus, it appears that a principled way to determine the regularization parameter γ, is to choose it from a logarithmic scale. The right value of γ will help to suppress the noise and prevent overfitting, at the expense of recovering the finer details of the attractor. In any case, the fine structure is likely to be dominated by the noise anyway.

The fractal structure of the attractor also suggests that the number of centers, M_c, should be chosen from a logarithmic scale as well. Intuitively, as with the box-counting dimension, an exponentially increasing number of spheres of diameter ε are required to cover all the points in the data set, as $\varepsilon \to 0$.

RBF networks suffer from the curse of dimensionality [114]. This means that if the dimension of underlying data increases, the corresponding number of basis functions required also increase exponentially. The exact dimensionality of the problem is unknown, as some inputs may be correlated. Indeed, if Eq. (3.2) is used, then many of the inputs would be highly correlated. Ironically, the curse of dimensionality also provides the crucial insight that the candidates for M_c should be chosen from a logarithmic scale. After all, if this was an exact interpolation network, then $M_c = N$ would be increasing exponentially with increasing dimension.

3.6.4 Modification of Norm for Regularization

Calculating $\mathbf{G}_\gamma = \left(\phi(\rho_{j_1 j_2})\right)_{M\times M}$ means that it is necessary to find the distance between each center:

$$\rho_{j_1 j_2} = \sqrt{(\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2})^T \boldsymbol{\Lambda}_{j_1 j_2} (\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2})}, \tag{3.80}$$

where $\boldsymbol{\Lambda}_{j_1 j_2}$ is the norm-inducing matrix with respect to centers j_1 and j_2.

For the case of the EBF, the distance between a center and a point ρ_{ij}, is a function of $\mathbf{S}_j$. However, when finding the distance between 2 centers, how should information from each covariance matrix be incorporated?

One possibility is $\mathbf{\Lambda}_{j_1 j_2} = \mathbf{S}_{j_3}^{-1}$, where $\mathbf{S}_{j_3}$ is the sample covariance of the combined cluster formed from clusters j_1 and j_2. Since $\mathbf{G}_\gamma$ is a symmetric $M \times M$ matrix and all the elements on the main diagonal are equal to 1 when Gaussian basis functions are used, it is only necessary to form $\left\{\mathbf{S}_{j_3}\right\}_{j_3=1}^{\frac{M(M-1)}{2}}$ sample covariance matrices and to compute their corresponding inverses $\left\{\mathbf{S}_{j_3}^{-1}\right\}_{j_3=1}^{\frac{M(M-1)}{2}}$. Assuming matrix inversion to be O(p^3), this means that the complexity of computing $\mathbf{G}_\gamma$ is effectively O($p^3 M^2$), and represents a moderately significant computational load. If the covariance matrices are cached, then this also requires a significant amount of storage.

A natural alternative is $\mathbf{\Lambda}_{j_1 j_2} = \mathbf{S}_{j_1 j_2}^{-1}$:

$$\rho_{j_1 j_2}^{P} \triangleq \sqrt{(\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2})^T \mathbf{S}_{j_1 j_2}^{-1} (\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2})}, \tag{3.81}$$

where $\mathbf{S}_{j_1 j_2}$ is the pooled sample covariance matrix defined as

$$\mathbf{S}_{j_1 j_2} \triangleq \frac{(N_{j_1} - 1)\mathbf{S}_{j_1} + (N_{j_2} - 1)\mathbf{S}_{j_2}}{N_{j_1} + N_{j_2} - 2}. \tag{3.82}$$

$\mathbf{S}_{j_1 j_2}$ is an unbiased estimator of the common covariance of 2 populations of clusters j_1 and j_2 [115]. N_{j_1} is the number of elements in cluster j_1, and N_{j_2} is the number of elements in cluster j_2. For those who are well versed in pattern recognition, the Bhattacharyya distance, given as

$$\rho_{j_1 j_2}^{BD} \triangleq (\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2})^T \left(\frac{\mathbf{S}_{j_1} + \mathbf{S}_{j_2}}{2} \right)^{-1} (\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2}) + \frac{1}{2} \ln \frac{\left| \frac{\mathbf{S}_{j_1} + \mathbf{S}_{j_2}}{2} \right|}{\sqrt{\left|\mathbf{S}_{j_1}\right|} \sqrt{\left|\mathbf{S}_{j_2}\right|}} \tag{3.83}$$

may come to mind (Ref. [116], Chapter 3). If $\mathbf{S}_{j_1} = \mathbf{S}_{j_2}$, $\rho_{j_1 j_2} = \sqrt{\rho_{j_1 j_2}^{BD}}$.

Using $\mathbf{\Lambda}_{j_1 j_2} = \mathbf{S}_{j_1 j_2}^{-1}$ still requires computational complexity of O($p^3 M^2$) for computing $\mathbf{G}_\gamma$. However, it is no longer necessary to form $\frac{M(M-1)}{2}$ covariance matrices, provided the sample covariance matrices $\{\mathbf{S}_j\}_{j=1}^{M}$ are cached (see Section 3.6.5). This results in some computational and storage savings.

There is a problem; the results in Appendix B show that $\mathbf{S}_{j_1 j_2}$ is positive definite if either $\mathbf{S}_{j_1}$ or $\mathbf{S}_{j_2}$ is positive definite. $\mathbf{S}_{j_1 j_2}$ may be positive semidefinite (implying that $\mathbf{S}_{j_1 j_2}^{-1}$ may not exist) if both $\mathbf{S}_{j_1}$ and $\mathbf{S}_{j_2}$ are positive semidefinite. Also, the results in Appendix C require $\mathbf{S}_{j_1 j_2}$ to be positive definite if $\rho_{j_1 j_2}$ is to be a valid metric. If $\rho_{j_1 j_2}$ is not a valid metric, it could vary in an irregular fashion from center to center, and regularization might fail.

On the other hand, a weighted sum (positive weights) of 2 valid norms is still a valid norm, suggesting:

$$\begin{aligned}\rho_{j_1 j_2}^{N} \triangleq & \frac{(N_{j_1}-1)}{N_{j_1}+N_{j_2}-2}\sqrt{(\boldsymbol{\mu}_{j_1}-\boldsymbol{\mu}_{j_2})^T \mathbf{S}_{j_1}^{-1}(\boldsymbol{\mu}_{j_1}-\boldsymbol{\mu}_{j_2})} \\ & + \frac{(N_{j2}-1)}{N_{j_1}+N_{j_2}-2}\sqrt{(\boldsymbol{\mu}_{j_1}-\boldsymbol{\mu}_{j_2})^T \mathbf{S}_{j_2}^{-1}(\boldsymbol{\mu}_{j_1}-\boldsymbol{\mu}_{j_2})}.\end{aligned} \tag{3.84}$$

The advantage is that if the M inverse matrices $\{\mathbf{S}_j^{-1}\}_{j=1}^{M}$ are cached, then it is unnecessary to compute any matrix inverse. The disadvantage is that $\rho_{j_1 j_2}$, as defined using Eq. (3.84), is not guaranteed to be a valid norm, if either $\mathbf{S}_{j_1}$ or $\mathbf{S}_{j_2}$ is positive

semidefinite (implying that $\mathbf{S}_{j_1}^{-1}$ or $\mathbf{S}_{j_2}^{-1}$ may not exist). Thus, it might be more prone than to problems than Eq. (3.81).

Generalizing these results to regularized non-radial basis functions (Section 3.4.3), the two possibilities being advocated in this section are:

$$\rho_{j_1 j_2}^{P} \triangleq \sqrt{(\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2})^T \left(\frac{(N_{j_1} - 1)\mathbf{M}_{j_1} + (N_{j_2} - 1)\mathbf{M}_{j_2}}{N_{j_1} + N_{j_2} - 2} \right)^{-1} (\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2})}, \tag{3.85}$$

and

$$\begin{aligned} \rho_{j_1 j_2}^{N} \triangleq & \frac{(N_{j_1} - 1)}{N_{j_1} + N_{j_2} - 2} \sqrt{(\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2})^T \boldsymbol{\Lambda}_{j_1} (\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2})} \\ & + \frac{(N_{j2} - 1)}{N_{j_1} + N_{j_2} - 2} \sqrt{(\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2})^T \boldsymbol{\Lambda}_{j_2} (\boldsymbol{\mu}_{j_1} - \boldsymbol{\mu}_{j_2})}, \end{aligned} \tag{3.86}$$

where $\mathbf{M}_{j_1}$ is the $\mathbf{M}_j$ matrix and $\boldsymbol{\Lambda}_{j_1}$ is the $\boldsymbol{\Lambda}_j$ matrix associated with cluster j_1; $\mathbf{M}_{j_2}$ is the $\mathbf{M}_j$ matrix and $\boldsymbol{\Lambda}_{j_2}$ is the $\boldsymbol{\Lambda}_j$ matrix associated with cluster j_2. $\mathbf{M}_j$ is possibly regularized as in Eq. (3.41) and $\boldsymbol{\Lambda}_j$ is possibly regularized as in Eq. (3.42).

3.6.5 Speeding Up Cross Validation

It is observed that clustering is responsible for most of the running time, when the data set is high dimensional, provided $N \gg M$. Thus, clustering results are reused whenever possible, *i.e.* caching. Different algorithms are simulated, which differ only in the ways they utilize the information from the clustering stage. Thus, clustering results could be safely stored on disk, and reused whenever needed. On the other hand, if the order of M is close to that of N, then solving the least squares problem is also

very time consuming. This is because the computational complexity of matrix inversion is O(M^3) for SVD (Section 3.5.2) and for Gaussian elimination [117].

For regularized networks, a brute-force search is conducted in a two-dimensional space, for the optimal values of (M_c, γ). The search takes place in a $L_M \times L_\gamma$ grid, where L_M is the number of values of M_c (and also M) explored, and L_γ is the number of values of γ explored. In principle, it is possible to optimize other hyper-parameters, such as d_E, but it is necessary to consider the computational cost. Some calculations, such as those for finding τ and d_E are cached, so as to save computation time.

Observe that the value of M_c affects the clustering stage, but the value of γ only affects Eq. (3.65). Thus, the computational running time used by the clustering during k-fold cross validation can be reduced from O(L_M L_γ) to O(L_M) by having an inner loop vary γ for Eq. (3.65) L_γ times. Note that the computational complexity of the least squares stage remains as O(L_M L_γ).

In fact, the simulations are arranged such that loops which generate information which can be cached are in the outer layers. The innermost loop is the one which varies γ; only Eq. (3.65) is solved in the innermost loop. The algorithm for the simulations is described below:

Calculate τ and d_E based on data in training set

Form data into design sets and validation sets in preparation for k-fold cross validation

Repeat for $k_L = 1{:}k$

Repeat for $j_L = 1{:}L_M$

Perform clustering

Calculate covariance matrices and their inverses

Repeat for $a_L = 1{:}\ L_A$

Form $\mathbf{G}$ and $\mathbf{G}_\gamma$

Repeat for $\gamma_L = 1{:}\ L_\gamma$

Solve Eq. (3.65)

end

end

end

end

Note that k_L, j_L, a_L, and γ_L are dummy variables; L_A is the number of variants of RBF being tested, e.g. RBF, TBF, DBF, etc.

Another trick is to use the Euclidean norm squared instead of the Euclidean norm when performing clustering:

$$\sqrt{a} < x < \sqrt{b} \Rightarrow a < x^2 < b\,. \tag{3.87}$$

By omitting the square root, many clock cycles are saved, because the square root operation takes up many clock cycles on most computers. For any clustering algorithm, $N \times M$ norms need to be computed per iteration.

3.7 Contributions of this Chapter

- Research in neural networks had largely been about devising new algorithms and showing that they are superior in some sense. This had spawned a large variety of neural networks, with minor variants. However, no algorithm may be universally superior, due to the No Free Lunch Theorem [87]. Thus, rather than suggest an algorithm and claim that it is the best, the achievement here is to employ a trick to speed up *k*-fold cross validation (Section 3.6.5). It applies to both classification and regression, and to various variants of the Radial Basis Function (RBF), which may employ clustering techniques. This same trick could be employed for the Leave-One-Out (LOO) method as well. Currently, most neural network applications employ Multi Layer Perceptrons (MLPs). The computational savings introduced could tip the balance and encourage more applications to employ the RBF or its variants. Another implication is that it may discourage certain ways of regularizing RBFs, such as regularization by training with noise [118]. Unfortunately, the regularization parameter cannot be varied within the innermost loop, because the data is affected, and not just the least squares equations. Hence, training with noise cannot be accelerated using caching.
- The standard architecture of the RBF is revised (Section 3.2) to include the possibility that the number of centers may be unequal to the number of weights in the linear layer, due to empty clusters.

- Non-radial basis functions are introduced (Section 3.4.1) these may require less centers, thereby alleviating the curse of dimensionality. Numerical techniques and computational tricks are discussed.
- A possible explanation is found for the puzzling phenomenon of empty clusters, which occasionally occur (Section 3.3.2).
- It is suggested that the non-deterministic outcome of the clustering stage could sometimes affect the RBF. Since there are M_c centers, symmetry implies that there can be $M_c!$ local maxima the k-means algorithm can converge onto (Section 3.3.2). This can be a very large number in the context of time series prediction, because hundreds of centers may be used, not to mention the 1500 centers used to model sea clutter data in Ref. [22]. It appears that caching, *i.e*. the trick to speed up k-fold cross validation, is the most practical solution.
- The large number of parameters required to tune an EBF network suggested the concept of a data driven neural network (Section 3.4.2), whereby parameters are derived from the data, rather than adjusted during training. It is to be acknowledged that Roderick Murray-Smith [119] had a similar concept, except that each covariance matrix was estimated from a group of neighbouring clusters, rather than estimated from the cluster itself. In this work (conceived independently of Ref. [119]), regularized covariance matrices are suggested as a way of dealing with numerical issues (Section 3.4.3).
- The formulation of $\mathbf{G}_\gamma = \left(\phi(\rho_{j_1 j_2})\right)_{M \times M}$ is extended to non-radial basis functions (Section 3.6.4). The derivation for the effective number of parameters of ridge regression by Orr [112] is extended (Section 3.6.1).
- A rationale for choosing the hyperparameters from a logarithmic scale is given (Section 3.6.3).

3.8 Summary

A review of RBF networks and variants are given in this chapter. It is hoped that the use of generalized versions of RBF networks may require less centers, thereby alleviating the curse of dimensionality. This stemmed from a desire to find a compromise between coping with the curse of dimensionality, and yet using all available information as effectively as possible. Elliptical Basis Function (EBF) networks and other methods are discussed as ways of coping with the curse of dimensionality. Using these tools, it may be possible perform time series prediction more effectively.

CHAPTER 4

Data Characteristics

It is necessary to understand the data thoroughly before modelling it; throwing data into the neural network blindly only results in garbage in and garbage out. This chapter discusses how the data was obtained, and examines the data from different perspectives.

4.1 IPIX Radar

The data comes from a transportable radar called the Intelligent Pixel Processing (IPIX) radar [120]. The radar was situated on a cliff-top at a height of 30m above mean sea level at Osborne Head Gunnery Range, Dartmouth, Nova Scotia on the east coast of Canada (latitude 44°36.72'N and longitude 63°25.41'W). The radar was operated in dwelling mode, so that the dynamics of the sea clutter recorded by the radar would be entirely due to the motion of the ocean waves and the natural motion of the sea itself.

IPIX is an instrument quality X-band radar system. The actual operating frequency is 9.39 GHz, *i.e*. wavelength is approximately 3cm. It has 2 identical receivers, one connected to the vertically polarized antenna feed, and the other is connected to the horizontal antenna feed.

The data was downloaded from the McMaster IPIX website: http://soma.ece.mcmaster.ca/ipix/. Each data file consists of 131072 samples (vertical

polarization). The real and imaginary components of the data correspond to the in-phase and quadrature phase channels, respectively, because the radar is coherent.

The illuminated area of sea surface is influenced by the antenna beamwidth, antenna height above the sea surface and the grazing angle (see Figure 1.1). Air-sea interactions have an important impact on sea clutter [1]. Hence, it is necessary to record wind and wave observations. A common measure is significant wave height, which is defined as the average peak-to-trough height of the one-third highest waves. Table 4.1 describes the conditions under which the sea clutter data is collected.

Table 4.1 Details of sea clutter data files

Filename	`lo.dat`	`hi.dat`
Date	18 /11/1993	17/11/1993
Time	13:13:53	20:49:23
Pulse Repetition Frequency (PRF)	2000Hz	1000Hz
Pulse duration	200ns	200ns
Beamwidth	0.9°	0.9°
Antenna height	30m	30m
Grazing angle	1.4°	1.9°
Range resolution	30.0m	30.0m
Significant wave height	0.8 m	1.8m
Maximum wave height	1.3m	2.9m
Wind velocity	25km/h from 340°	22km/h from 218°

Note that the file `lo.dat` contains sea clutter data in low sea state, whilst the file `hi.dat` contains sea clutter data in high sea state. Sea state is a term used by mariners as a measure of wave height. Low sea state means that the sea surface is calm and the wave height is low; high sea state means that wave height is high [1]. Detailed discussions about the various parameters in Table 4.1 can be found in Ref. [13, 38].

4.2 Hilbert Transform

Takens' theorem applies to complex measurements in the trivial sense, by considering a complex embedding of dimension d_E to be equivalent to a real embedding of dimension $2d_E$, if no use is made of the complex structure [47]. Besides, there is the possibility of non-generic variables leading to distorted phase space reconstructions (see Figure 2.4 for an example of a distorted phase space reconstruction). Hence, it is good practice to examine the relationship between the real component and imaginary component.

By inspection, it appears that Re($z(n)$), the real component of sea clutter in low sea state (file `lo.dat`) is related to Im($z(n)$), the imaginary component via the Hilbert Transform (see Figure 4.1):

$$\mathrm{Re}(\mathbf{z}) = \mathcal{H}\left(\mathrm{Im}(\mathbf{z})\right), \tag{4.1}$$

where $\mathcal{H}(\bullet)$ is the Hilbert Transform and $\mathbf{z}$ is the vector containing $\{z(n)\}_{n=1}^{131072}$.

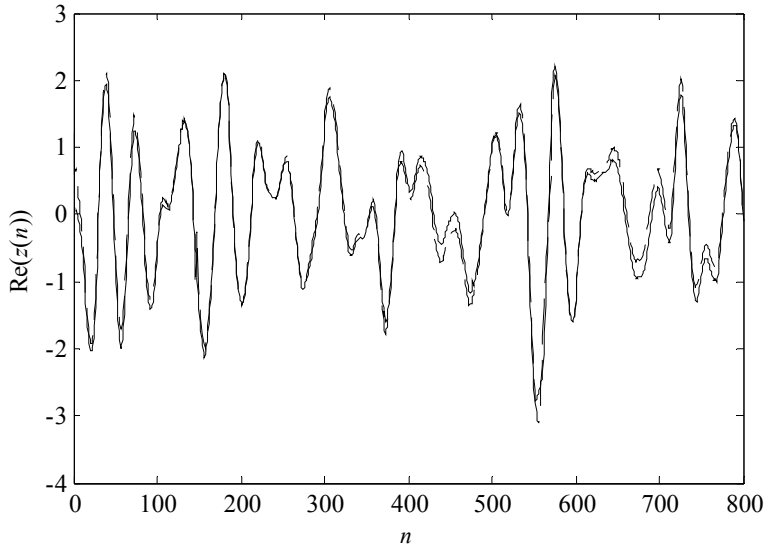

Figure 4.1 Plot of in-phase component (solid line) vs Hilbert Transform of quadrature component (dashed line) for sea clutter in low sea state.

Define the Normalized Error Variance as

$$\text{NEVH1} \triangleq \frac{\text{s}^2\left(\text{Re}(\mathbf{z}) - \mathcal{H}\left(\text{Im}(\mathbf{z})\right)\right)}{\text{s}^2(\text{Re}(\mathbf{z}))}, \tag{4.2}$$

where the sample variance is formulated in vector notation as

$$s^2(\mathbf{x}) \triangleq \frac{\mathbf{x}^H\mathbf{x}}{N_\mathbf{x} - 1} - \frac{(\mathbf{x}^H\mathbf{h})(\mathbf{h}^T\mathbf{x})}{N_\mathbf{x}(N_\mathbf{x} - 1)}, \tag{4.3}$$

and where the vector $\mathbf{x} \in \mathbb{C}^{N_\mathbf{x}}$ has $N_\mathbf{x} \in \mathbb{Z}^+$ elements and $\mathbf{h} \triangleq \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$ is a constant vector with $N_\mathbf{x}$ elements. NEVH1 is found to be 0.0293 over the entire data set of 131072 samples, which is very low, thereby verifying that Eq. (4.1) is true.

For sea clutter in high sea state (file `hi.dat`), the imaginary component also seems to be related to the real component via the Hilbert Transform (see Figure 4.2):

$$\text{Re}(\mathbf{z}) = -\mathcal{H}\left(\text{Im}(\mathbf{z})\right). \tag{4.4}$$

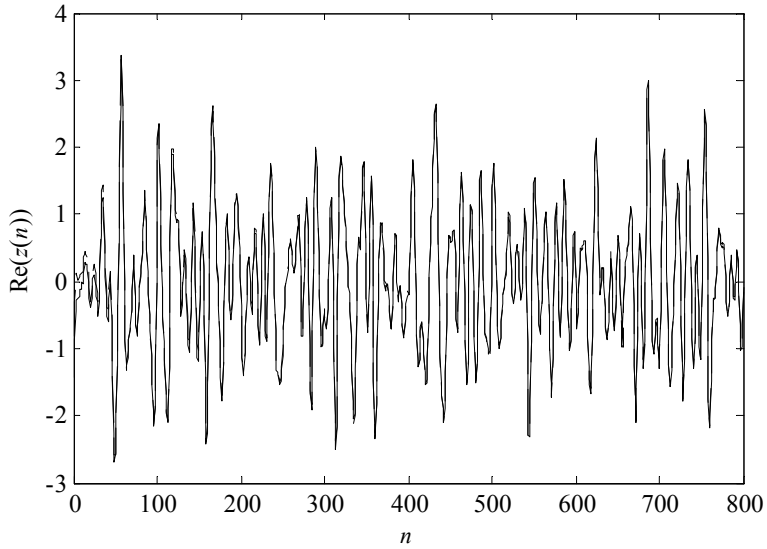

Figure 4.2 Plot of in-phase component (solid line) vs negative of Hilbert Transform of quadrature component (dashed line) for sea clutter in high sea state.

Define the Normalized Error Variance as

$$\mathrm{NEVH2} \triangleq \frac{\mathrm{s}^2\left(\mathrm{Re}(\mathbf{z}) + \mathcal{H}\left(\mathrm{Im}(\mathbf{z})\right)\right)}{\mathrm{s}^2(\mathrm{Re}(\mathbf{z}))}. \tag{4.5}$$

NEVH2 is found to be 0.00347 over the entire data set of 131072 samples, which is very low, thereby verifying that Eq. (4.4) is true.

Hence, it turns out that for the data sets studied, the real and imaginary components are not independent, since they are related via the Hilbert transform. This is possibly due to the action of quadrature modulators, which may be seen as phase shifters. Thus, instead of processing both components of the complex signal, it is sufficient to deal with either component.

4.3 Stationarity

If the data is non-stationary, it is meaningless to apply typical methods used in chaotic time series analysis, since the assumption of ergodicity is violated [121]. On the other hand, Ruelle [122] suggested that some of the nonlinear time series methods remain useful when the time dependence is assumed to be adiabatic (slow compared to the characteristic times of the other parameters of the system).

Hence, it is advisable to check if the data is stationary. This can be done via recurrence plots [13]. Consider $\mathbf{\Psi}(n)$ in phase space. The recurrence plot is an array of points in a $N^{\mathbf{\Psi}} \times N^{\mathbf{\Psi}}$ grid (where $N^{\mathbf{\Psi}}$ is the number of embedding vectors), where a dot is placed at (m, n) whenever $\mathbf{\Psi}(m)$ is sufficiently close to $\mathbf{\Psi}(n)$.

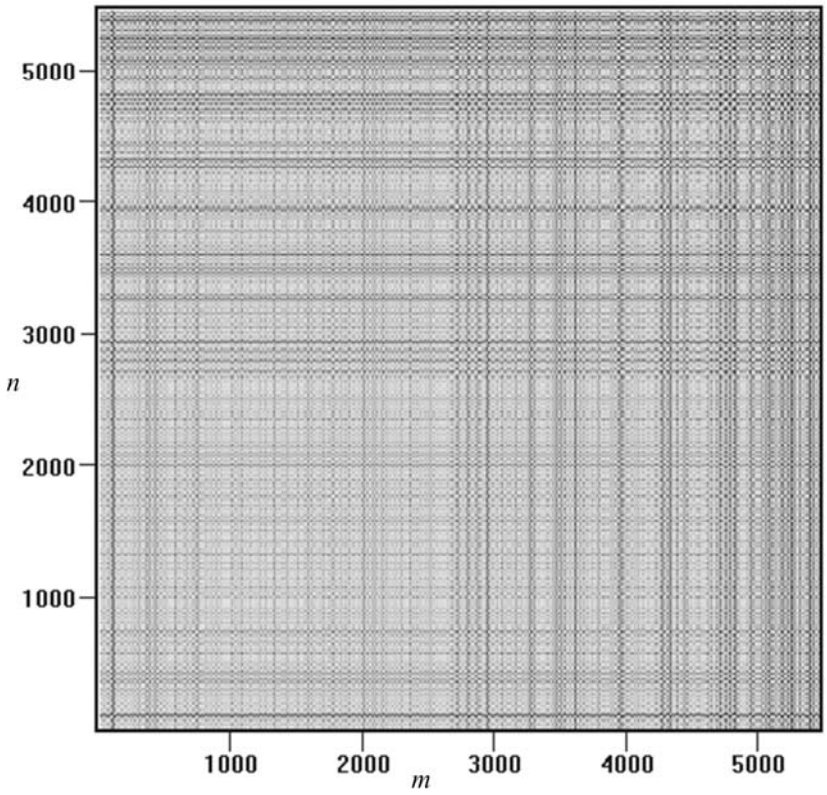

Figure 4.3 Recurrence plot for in-phase component of sea clutter in low sea state.

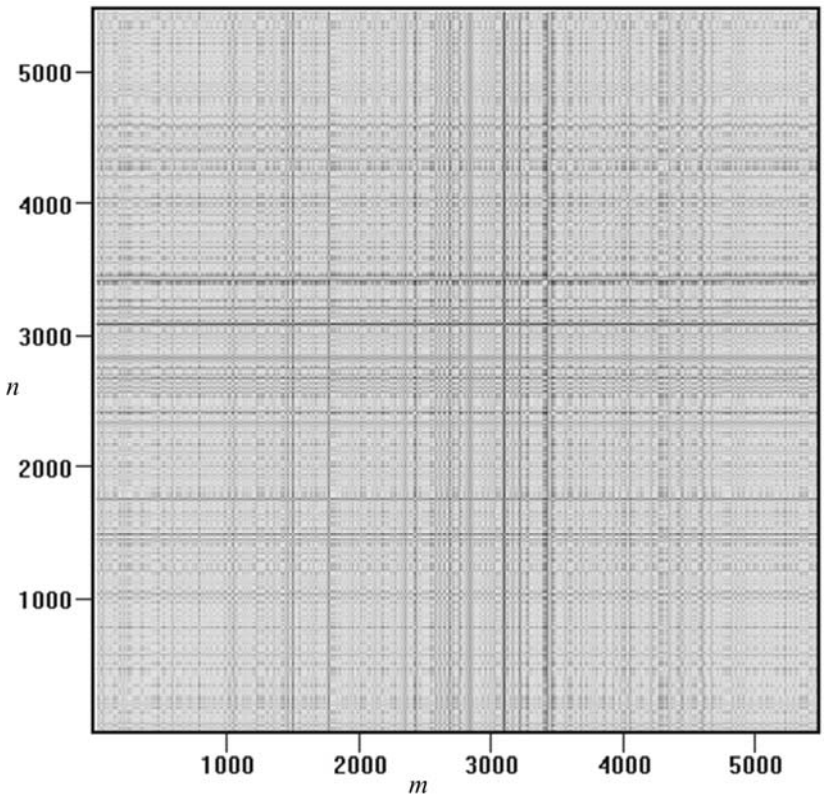

Figure 4.4 Recurrence plot for in-phase component of sea clutter in high sea state.

According to Haykin and Puthusserypady [13], nonstationarity can be detected from fading in the recurrence plot away from the main diagonal. Since no fading is discernable, stationarity is implied in Figure 4.3 and Figure 4.4.

4.4 Frequency Spectrum

One of the first things to do when analyzing time series data, is to examine its frequency spectrum. Figure 4.5 illustrates the frequency spectrum of the in-phase component of `lo.dat`. The frequency spectrum could perhaps be decomposed into a portion with white noise and a portion with 1/*f* noise, as with Figure 3.8. 1/*f* noise could be generated by dissipative dynamical systems in the transition to turbulence [65]. On the other hand, it should be stressed that even if 1/*f* noise is present, it could be caused by other mechanisms besides chaos.

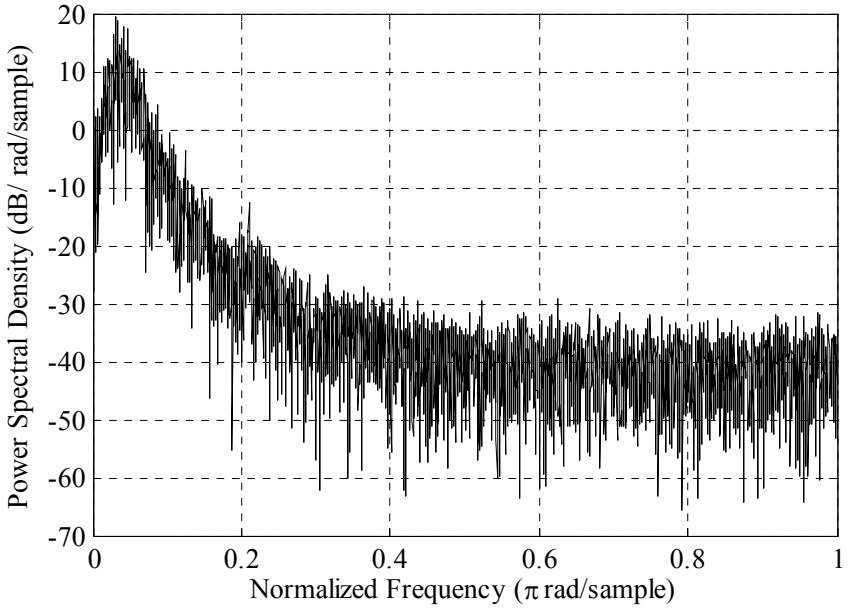

Figure 4.5 Power spectrum of in-phase component of sea clutter in low sea state.

The frequency spectrum of sea clutter in high sea state is similar (see Figure 4.6), except that more of the power is in the higher frequencies. Also, the lower frequencies are attenuated, resulting in a peak in the spectrum at about 0.2π rad/sample.

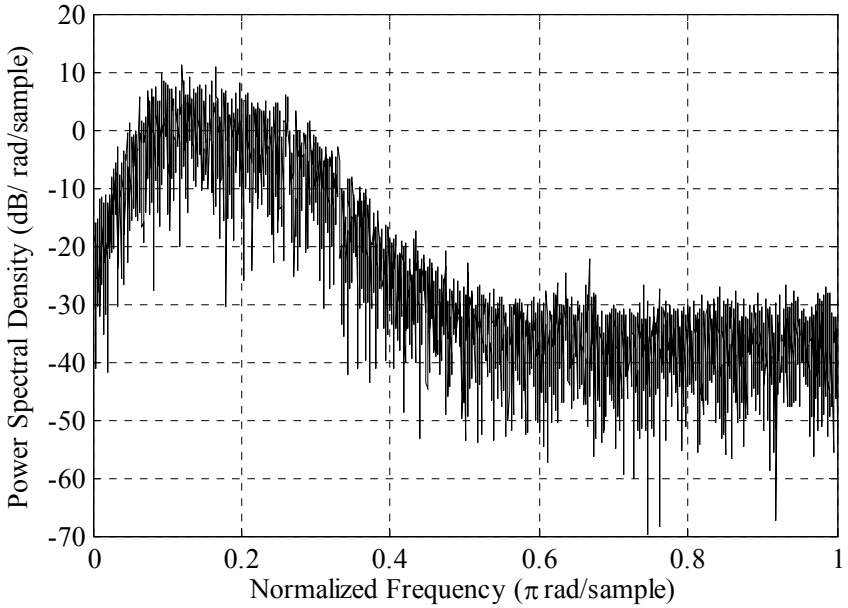

Figure 4.6 Power spectrum of in-phase component of sea clutter in high sea state.

4.5 Chaotic Invariants

The techniques of Chapter 2 can be used to determine the chaotic invariants of sea clutter data. However, it is good practice to determine the results for known chaotic systems, as a way of checking that the algorithms are properly implemented.

4.5.1 Chaotic Invariants of Known Systems

Experiments were carried out on the *x*-component of the Lorenz system as a benchmark. The time series has 2000 samples and is corrupted with AWGN for various values of SNR. In Table 4.2, HOP stands for Horizon of Predictability, KE stands for Kolmogorov Entropy and NaN means Not a Number. The correlation

dimension D_2 is computed using `takens_estimator.dll`, as implemented in TSTOOL [64]. The Lyapunov exponents are calculated using `lyap_spec.exe`, as implemented in TISEAN [52]. Both HOP and KE are estimated from the Lyapunov exponents.

Table 4.2 Computed chaotic invariants of Lorenz data at varying SNR

SNR (dB)	d_E	τ	D_2	D_{KY}	HOP	KE	Lyapunov exponents
-5	5	5	4.78	NaN	NaN	NaN	no positive exponents
10	4	5	4.26	1.74	187	2.09E-02	2.09E-02, -2.83E-02, -6.88E-02, -1.46E-01
20	4	5	3.64	2.01	113	3.47E-02	3.47E-02, -3.38E-02, -7.21E-02, -1.69E-01
25	3	4	2.47	2.03	69.6	5.62E-02	5.62E-02, -4.77E-02, -2.92E-01
30	3	4	2.22	2.06	61.6	6.35E-02	6.35E-02, -4.45E-02, -3.19E-01
∞	3	4	1.98	2.3	41.1	1.06E-01	9.51E-02, 1.13E-02, -3.49E-01

The estimates of D_2 and D_{KY} appear to be satisfactory, because the theoretical value of the Lorenz system is about 2.06 [123]. It is expected that both D_2 and D_{KY} should become more unreliable as SNR is increased. In this respect, D_{KY} appears to be a more robust estimate in the presence of AWGN. It appears that the values of HOP and KE are unreliable, because HOP and KE should not be increasing as SNR decreases. This is caused by underestimation of the positive Lyapunov exponents as SNR is decreased.

For a benchmark derived from experimental data, rather than differential equations, the data set used is the laser time series (Figure 4.7), data set A from the Santa Fe Time Series Competition [123]. Hereafter, the data set would be referred to as SFA. SFA can be modelled by the same equations as the Lorenz system, using the Haken-Lorenz model [123]; hence the chaotic invariants computed for SFA should be similar to those for the Lorenz system. Indeed, the results in Table 4.3 are similar to those for the row of Table 4.2 corresponding to SNR of ∞dB.

Table 4.3 also indicates that the computed chaotic invariants did not vary much as the number of samples were increased. During the Santa Fe Time Series Competition, a data set of 1100 samples was made available to the contestants; this corresponds to a training set of 730 samples. After the contest, the full data set of 10093 samples was made available. In the rest of this work, SFA refers to the original data set of 1100 samples unless otherwise specified.

Table 4.3 Computed chaotic invariants of SFA

samples	d_E	τ	D_2	D_{KY}	HOP	KE	Lyapunov exponents
730	3	2	1.96	2.14	31.9	1.23E-01	1.23E-01, -6.09E-02, -4.29E-01
10093	3	2	2.13	2.21	42.7	9.17E-02	9.17E-02, -2.91E-02, -2.73E-01

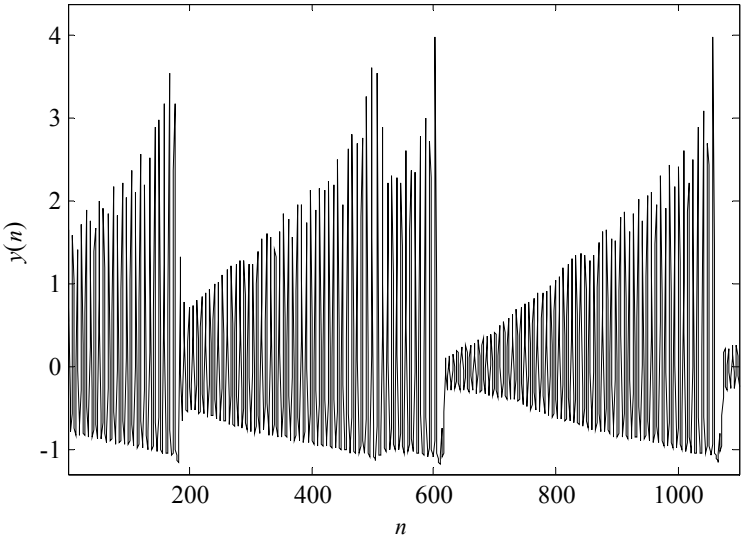

Figure 4.7 Time series of data set A from Santa Fe Time Series Competition (SFA).

Incidentally, the phase space reconstruction for SFA in Figure 4.8 looks similar to Figure 2.4.

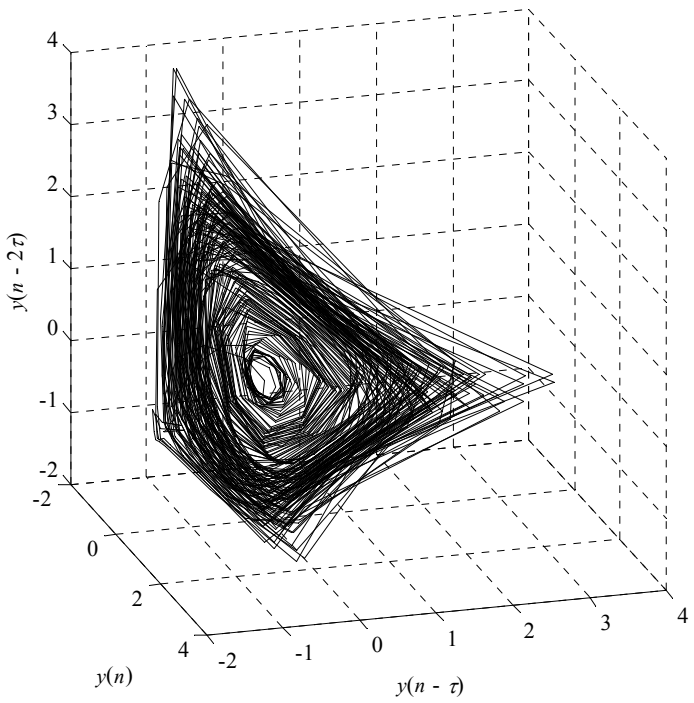

Figure 4.8 Phase space reconstruction for SFA.

4.5.2 Chaotic Invariants of Sea Clutter Data

Chaotic invariants were also computed for sea clutter data in low sea state and high sea state; the results are recorded in Table 4.4 and Table 4.5. The embedding dimension d_E, was found to be about 5 for both low sea state and high sea state. This tallies with the results in Ref. [13]. For sea clutter in low sea state, it is estimated to have a fractal dimension between 4 and 5.

Table 4.4 Chaotic invariants calculated for in-phase component of sea clutter data (low sea state)

samples	d_E	τ	D_2	D_{KY}	HOP	KE	Lyapunov exponents
5460	5	12	4.88	4.02	206	2.92E-02	1.90E-02, 1.03E-02, -3.99E-03, -2.03E-02, -5.99E-02
131072	5	11	4.9	4.18	130	4.25E-02	3.00E-02, 1.24E-02, -2.98E-03, -1.96E-02, -6.07E-02

Table 4.5 Chaotic invariants calculated for in-phase component of sea clutter data (high sea state)

samples	d_E	τ	D_2	D_{KY}	HOP	KE	Lyapunov exponents
5460	5	3	4.87	2.01	153	2.56E-02	2.56E-02, -9.24E-03, -4.49E-02, -1.05E-01, -2.61E-01
131072	5	3	4.66	3.72	80.2	6.46E-02	4.88E-02, 1.59E-02, -2.34E-02, -8.27E-02, -2.24E-01

Unfortunately, the full data set of 131072 samples requires a very long time to train, despite caching. Also, there is the possibility that a long training sequence may induce oscillations in network training [82]. Hence the training set is much shorter than the full data set and has only 5460 samples. Nonetheless, the chaotic invariants for sea clutter in low sea state remained approximately the same, despite data size. Thus, the training set is reasonably reflective of the full data set. This seems to confirm that sea clutter in low sea state is stationary, for `lo.dat`.

On the other hand, D_{KY} varied somewhat for sea clutter in high sea state as data size was changed. Besides, D_{KY} is significantly lower than D_2 in high sea state, suggesting that the Lyapunov exponents could not be accurately measured. This could be due to the effect of noise. Since the largest Lyapunov exponent λ_1, is larger for sea clutter data in high sea state, HOP is lower, suggesting that prediction would be tougher. This also suggests that modelling of sea clutter data in high sea state is likely to be problematic, because the training set is not entirely reflective of the full data set. This also implies that there is some degree of non-stationarity in sea clutter in high sea state; increasing the size of the training set is insufficient to deal with non-stationarity.

4.6 Contributions of this Chapter

- It was suggested that instead of dealing with both real and complex components of the sea clutter signal, it may be sufficient to choose one. This is because for the available data sets, the in-phase and quadrature phase components are not independent, as they are related by the Hilbert transform. Thus, it is not necessary to consider phase space reconstructions of dimension $2d_E$.

CHAPTER 5

Results and Discussions

The ideas discussed in the preceding chapters are tested in this chapter. A suitable algorithm is chosen for modelling sea clutter.

5.1 Caching the Loops

5.1.1 Timing Results

The first test is to verify that caching the loops result in significant computational savings. Thus it is necessary to compare the performance of an algorithm which is cached with the performance of the same algorithm which is not cached.

The x-component of Lorenz data is generated as in Section 2.1 using Eq. (2.2); AWGN is added at 25dB SNR. An ordinary RBF was used, coupled with k-means for the clustering stage; this combination of k-means and RBF will be called the kRBF. The embedding dimension, d_E, is found using the method of Global False Nearest Neighbours (GFNN) [62], while τ is the embedding time delay, which is found from the first minimum in the mutual information [49].

Conjugate gradient method for linear systems is used to solve the least squares system in the linear layer, because MATLAB© 5 has a less stable implementation of SVD. MATLAB© 6 has a more stable implementation of SVD, but it does not have a command to count the number of FLOPS required to perform the simulations (*i.e.* the

`flops` command). Note that in subsequent simulations throughout this work, SVD is used, since the rest of the simulations use compiled code developed on MATLAB© 6.

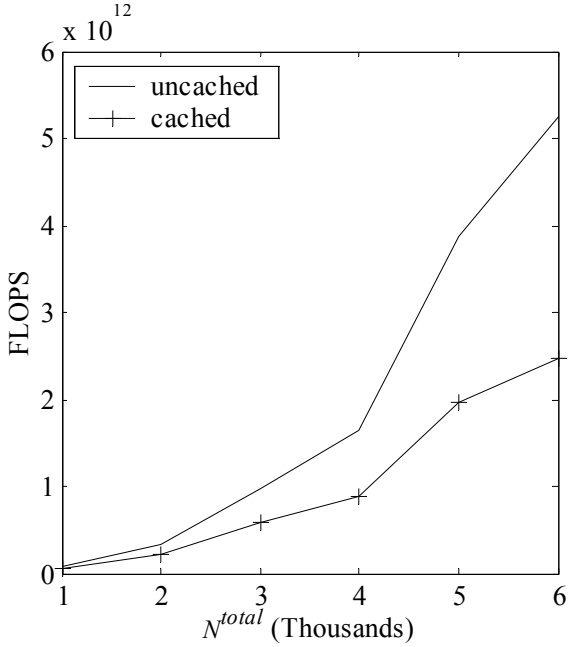

Figure 5.1 FLOPS required by kRBFs to model Lorenz datasets of different sizes.

Figure 5.1 shows the FLOPS required by simulations performed on Lorenz data (data sets of varying sizes), with caching and without caching. It appears that the cached algorithm is about twice as fast as the uncached version. This could be because the clustering stage is almost L_γ times faster, but the linear layer, *i.e*. solving Eq. (3.65), cannot be speeded up. However, caching is helpful for larger data sets, because the clustering stage is prone to the curse of dimensionality (See Section 3.5.1). Note that the FLOPS count included FLOPS spent on cross validation, and on the final training set, because both sets of algorithms would eventually need to spend time on the final training set, which cannot be cached. Actually, it is possible to use the cached clustering results to initialize the clustering stage of the final training set, in order to

achieve some computational savings. However, the results may differ somewhat, due to erraticity. Note that the optimum number of hidden units is given as

$$M \propto N^{1/3}, \tag{5.1}$$

according to Ref. [98]. On the other hand, the computational complexity of Gaussian Elimination or SVD is O(M^3). This implies that the computational complexity of the linear layer is effectively O(N), and seems to suggest that RBF networks are potentially scalable with respect to N, provided the clustering stage has negligible computational complexity, and cross validation is unnecessary. However, caching the clustering results does allow one to approach these two assumptions more closely.

5.1.2 Empty Clusters

To illustrate the phenomenon of empty clusters, a kRBF with no caching was used on SFA (N^{train} = 730), and d_E = 3 was found using GFNN.

Table 5.1 Centers remaining after clustering, M, for kRBF on SFA {M_c = 10, 25}

M_c	γ	M (k_L = 1)	M (k_L = 2)	M (k_L = 3)	M (k_L = 4)	M (k_L = 5)
10	0.0E+00	10	10	10	10	10
10	1.0E-04	10	10	10	10	10
10	3.0E-04	10	10	10	10	10
10	1.0E-03	10	10	10	10	10
10	3.0E-03	10	10	10	10	10
10	1.0E-02	10	10	10	10	10
10	3.0E-02	10	10	10	10	10
10	1.0E-01	10	10	10	10	10
10	3.0E-01	10	10	10	10	10
25	0.0E+00	25	25	25	25	25
25	1.0E-04	25	25	25	25	25
25	3.0E-04	25	25	25	25	25
25	1.0E-03	25	25	25	25	25
25	3.0E-03	25	25	25	25	25
25	1.0E-02	25	25	25	25	25
25	3.0E-02	25	25	25	25	25
25	1.0E-01	25	25	25	25	25
25	3.0E-01	25	25	25	25	25

Table 5.2 Centers remaining after clustering, M, for kRBF on SFA $\{M_c = 50, 100, 200, 400, 500\}$

M_c	γ	M ($k_L = 1$)	M ($k_L = 2$)	M ($k_L = 3$)	M ($k_L = 4$)	M ($k_L = 5$)
50	0.0E+00	50	50	50	50	50
50	1.0E-04	50	50	50	50	50
50	3.0E-04	50	50	50	50	50
50	1.0E-03	50	50	50	50	50
50	3.0E-03	50	50	50	50	50
50	1.0E-02	50	50	50	50	50
50	3.0E-02	50	50	50	50	50
50	1.0E-01	50	50	50	50	50
50	3.0E-01	50	50	50	50	50
100	0.0E+00	100	100	100	100	100
100	1.0E-04	100	100	100	100	100
100	3.0E-04	100	100	100	100	100
100	1.0E-03	100	100	100	100	100
100	3.0E-03	100	100	100	100	100
100	1.0E-02	100	100	100	100	100
100	3.0E-02	100	100	100	100	100
100	1.0E-01	100	100	100	100	100
100	3.0E-01	100	99	100	100	100
200	0.0E+00	200	200	200	200	200
200	1.0E-04	200	200	200	200	200
200	3.0E-04	200	200	200	200	200
200	1.0E-03	200	199	200	200	200
200	3.0E-03	200	200	200	200	200
200	1.0E-02	200	200	200	200	200
200	3.0E-02	200	200	200	200	200
200	1.0E-01	200	200	200	199	200
200	3.0E-01	200	200	200	200	200
400	0.0E+00	400	400	400	399	399
400	1.0E-04	400	400	400	400	400
400	3.0E-04	400	400	400	400	400
400	1.0E-03	400	400	400	400	400
400	3.0E-03	400	400	400	400	400
400	1.0E-02	400	400	400	400	400
400	3.0E-02	400	400	400	400	400
400	1.0E-01	400	400	400	400	400
400	3.0E-01	400	400	400	400	400
500	0.0E+00	500	500	500	500	500
500	1.0E-04	500	500	500	500	500
500	3.0E-04	500	500	500	500	500
500	1.0E-03	500	500	500	500	500
500	3.0E-03	500	500	500	500	500
500	1.0E-02	500	500	500	500	500
500	3.0E-02	500	500	500	500	500
500	1.0E-01	500	500	500	500	500
500	3.0E-01	500	500	500	500	500

The results in Table 5.1 and Table 5.2 show that M can vary with γ and k, except when $\{M_c = 10, 25, 50, 500\}$. It seems that when the number of clusters is very small, or when the number of points per cluster is very small, the issue of empty clusters is usually minor. Nonetheless, the problem of having M which varies with γ and k may affect the cross validation results, since the RBF models being compared have different M when they should have had the same M, *i.e.* erraticity. In any case, there is no reason for M to vary with γ, since γ is independent of the data. Thus, caching the clustering result should alleviate the problem of erraticity.

Consider the case of Lorenz data of 10dB SNR ($N^{train} = 2000$). The clustering method used was Babuska's method (Section 3.3) with $\gamma_c = 0.1$; ordinary RBF was used. The outcome of the clustering stage was cached, and hence the value of γ is irrelevant. Table 5.3 illustrates a particularly severe case of empty clusters. For $M_c = 1200$, and $\{k = 1,2,4,5\}$, more than half the centers were dropped. The problem of M varying with k persists, but at least it is reasonable that clustering results should vary with k, since the design sets (see Glossary) involved are different.

Table 5.3 Centers remaining after clustering, M, for Babuska's method of clustering ($\gamma_c = 0.1$) on Lorenz data at 10dB SNR

M_c	M ($k_L = 1$)	M ($k_L = 2$)	M ($k_L = 3$)	M ($k_L = 4$)	M ($k_L = 5$)
10	10	10	10	10	10
25	25	25	25	25	25
50	50	50	50	50	50
100	98	100	98	99	100
200	193	194	190	187	184
400	315	323	319	324	322
800	484	488	473	485	480
1200	580	598	620	576	598

Perhaps one possibility is to change the criterion for choosing the optimal hyperparameters from $\arg\min_{(M_c,\gamma)} \frac{1}{k}\sum_{k_L=1}^{k} GE_{k_L}^{val}$ to $\text{mode}\left(\left\{\arg\min_{(M_c,\gamma)} GE_{k_L}^{val}\right\}_{k_L=1}^{k}\right)$, which is essentially a voting criterion to choose the set of hyperparameters which occur most frequently over the k design sets. These two criteria are equivalent when the optimal set of hyperparameters do not change as k_L changes.

5.2 Error Criteria for Cross Validation

Typically, the error criterion for selecting models $GE_{k_L}^{val} = \text{MSE}_{k_L}^{val}$, where

$$\text{MSE}_{k_L}^{val} \triangleq \frac{(\mathbf{e}_{k_L}^{val})^H \mathbf{e}_{k_L}^{val}}{N_{k_L}^{val}}, \tag{5.2}$$

and $\mathbf{e}_{k_L}^{val}$ is the vector containing the estimation error $e_i = y_i - \hat{y}_i$ on the k_L-th validation set with $N_{k_L}^{val}$ elements. This work explores the alternative of $GE_{k_L}^{val} = \text{s}^2(\mathbf{e}_{k_L}^{val})$, where $\text{s}^2(\bullet)$ is the sample variance defined in Eq. (4.3). Since the bias does not affect the chaotic invariants of the estimated sequence, sacrificing the bias may enable one to obtain a more favourable resolution of the bias-variance dilemma (Section 3.6). Note that only the cross validation criterion is changed, and the bias neuron of the RBF is not removed.

Define Normalized Mean Squared Error (NMSE) [124] as

$$\text{NMSE} \triangleq \frac{(\mathbf{e}^{test})^H \mathbf{e}^{test}}{N^{test}\text{s}^2(\mathbf{y}^{test})}, \tag{5.3}$$

where $\mathbf{e}^{test}$ is the vector containing the error on test set and $\mathbf{y}^{test}$ is the vector containing the test set with N^{test} elements. Define the Normalized Error Variance (NEV) as

$$\text{NEV} \triangleq \frac{\text{s}^2(\mathbf{e}^{test})}{\text{s}^2(\mathbf{y}^{test})}. \tag{5.4}$$

Note that $\text{s}^2(\mathbf{y}^{test})$ is a normalizing factor in both Eq. (5.3) and Eq. (5.4); NEV = 1 corresponds to predicting the average; typically this happens when the data set is white noise. Similarly, NMSE = 1 corresponds to predicting the average of white noise, provided the noise is zero mean. The presence of the normalizing factor $\text{s}^2(\mathbf{y}^{test})$ makes human interpretation more convenient, since NMSE and NEV are used to gauge performance on the test set.

Incidentally, $\text{s}^2(\mathbf{e}^{test})$ is bounded below by the Cramer Rao Lower Bound (CRLB) [125] and bounded above by $\text{s}^2(\mathbf{y}^{test})$.

Table 5.4 Performance of kRBF using different error criteria.

Error criterion	d_{E}	M_c	γ	NEV	NMSE
$\text{MSE}_{k_L}^{val}$	3	100	1.0E-04	1.63E-02	1.63E-02
$\text{MSE}_{k_L}^{val}$	4	100	3.0E-04	5.25E-03	5.33E-03
$\text{MSE}_{k_L}^{val}$	5	100	0.0E+00	9.15E-03	9.20E-03
$\text{s}^2(\mathbf{e}_{k_L}^{val})$	3	100	1.0E-04	1.64E-02	1.65E-02
$\text{s}^2(\mathbf{e}_{k_L}^{val})$	4	100	3.0E-04	5.13E-03	5.19E-03
$\text{s}^2(\mathbf{e}_{k_L}^{val})$	5	100	0.0E+00	6.24E-03	6.23E-03

Table 5.4 shows the performance of kRBF on one-step prediction using $\text{MSE}_{k_L}^{val}$ or $\text{s}^2(\mathbf{e}_{k_L}^{val})$ for model selection. The values of M_c and γ were found using k-fold cross validation. Caching was used and there were no empty clusters generated. It appears that despite sacrificing bias, the use of $\text{s}^2(\mathbf{e}_{k_L}^{val})$ for model selection does not affect NMSE and NEV much. In fact, NEV is only higher for the case of $d_{\text{E}} = 3$, and is

significantly lower for the case of $d_E = 5$. Thus, $s^2(\mathbf{e}_{k_L}^{val})$, rather than $MSE_{k_L}^{val}$, will be used for model selection in cross validation in this work.

Note that the embedding dimension d_E is varied in Table 5.4, although $d_E = 3$ for SFA was found using GFNN in Table 4.3. The reason is that the choice of d_E can be ambiguous in the presence of high levels of noise (see Figure 2.9 and Figure 2.10). From the practical point of view, the choice of d_E should be one which gives the best results for one's application [126]. Thus, the result of varying d_E from 3 to 5 was tested. Note that $d_E = 5$ corresponds to the requirement that $d_E > 2D_0$ (Section 2.3).

5.3 Choosing the Algorithm

In Chapter 3, many variants of the basis function networks were discussed. So, which should be used for modelling sea clutter? This choice should be made after understanding the behaviour of the various networks. For the sake of computational tractability, the various networks are tested on SFA ($N^{train} = 730$), in order to find a candidate which can model data satisfactorily with a moderate number of centers. Nonetheless, modelling data using a small training set can be a challenging problem.

In Table 5.5, two clustering methods were used: k stands for *k*-means, f stands for FCM. RBF stands for the ordinary RBF algorithm. TBF, DBF and EBF are data driven basis functions, and had been explained in Section 3.4.2. The suffix P indicates the use of Eq. (3.85), whilst the suffix N indicates the use of Eq. (3.86); for example, TBFP stands for TBF using Eq. (3.85). Thus, there are 14 combinations of algorithms: 2 kinds of clustering algorithms {k, f} and 7 kinds of basis functions {RBF, TBFP,

DBFP, EBFP, TBFN, DBFN, EBFN} were used. The error criterion used was $s^2(\mathbf{e}_{k_L}^{val})$, and GE is defined as in Eq. (3.79). The values of M_c and γ were found using k-fold cross validation; no empty clusters generated.

However, it can be intimidating to look at the tables. An alternative is to obtain a scatter-plot of M_c, γ and NEV (Figure 5.2). Note that NMSE is tabulated for reference, and it is not necessary to include it in the scatter-plot, because NEV and NMSE are usually highly correlated. In this work, at least 7 variants of clustering and 7 kinds of basis functions are discussed. Together, this implies at least 49 combinations. Thus, a systematic way of labelling the algorithms is required. In the following figures, the string indicating the clustering method (lower case), and the string indicating the variant of RBF used (upper case) are concatenated to form the text label describing the point. A subscript (optional) is added to indicate the embedding dimension d_E used for training the RBF or variant. For example, "kRBF$_3$" indicates k-means clustering combined with the standard RBF algorithm, using $d_E = 3$.

The clustering of points makes the plot hard to see in two dimensions. In front of a computer, it is possible to use a mouse to rotate the plot to help to discern the three-dimensional structure of the plot. Note that because points with lower NEV are of higher interest, NEV is plotted on a logarithmic scale. Otherwise, it is very hard to resolve those clusters with low NEV. It is apparent that there are several clusters of points in Figure 5.2. One way to understand the figure is to split up the plot into 2 plots, as in Figure 5.3. For $M_c \geq 200$, it appears to be necessary to subdivide the plot as in Figure 5.4, in order to discern the various clusters more effectively.

Table 5.5 Simulation results using *k*-means and FCM

clustering stage	basis function	d_E	M_c	γ	GE	NEV	NMSE
f	DBFP	5	50	0.0E+00	3.74E-02	1.36E-02	1.39E-02
k	DBFP	5	50	0.0E+00	4.95E-02	1.89E-02	1.89E-02
f	DBFP	3	50	0.0E+00	4.28E-02	1.95E-02	1.97E-02
k	DBFP	3	50	0.0E+00	4.45E-02	2.00E-02	2.01E-02
k	RBF	4	100	1.0E-04	2.73E-02	5.99E-03	6.00E-03
f	RBF	4	100	1.0E-04	2.91E-02	7.06E-03	7.06E-03
k	RBF	5	100	1.0E-03	5.22E-02	9.50E-03	9.61E-03
f	RBF	5	100	1.0E-04	4.14E-02	1.01E-02	1.03E-02
k	RBF	3	100	1.0E-04	4.08E-02	1.64E-02	1.65E-02
f	RBF	3	100	1.0E-03	4.22E-02	1.99E-02	2.00E-02
f	TBFP	3	100	0.0E+00	4.76E-02	2.49E-02	2.50E-02
f	EBFP	3	100	0.0E+00	4.76E-02	3.32E-02	3.31E-02
f	DBFP	4	400	3.0E-03	3.62E-02	2.98E-03	2.97E-03
f	TBFN	4	400	3.0E-03	3.62E-02	3.03E-03	3.02E-03
f	TBFP	4	400	3.0E-03	3.62E-02	3.05E-03	3.05E-03
f	DBFN	4	400	3.0E-03	3.62E-02	3.14E-03	3.14E-03
k	TBFN	4	400	0.0E+00	3.23E-02	4.61E-03	4.79E-03
k	DBFP	4	400	0.0E+00	3.23E-02	4.73E-03	4.83E-03
k	TBFP	4	400	0.0E+00	3.23E-02	4.99E-03	5.13E-03
k	DBFN	4	400	0.0E+00	3.23E-02	5.13E-03	5.32E-03
k	TBFN	5	400	1.0E-02	5.29E-02	5.45E-03	5.52E-03
k	EBFN	5	400	1.0E-02	5.29E-02	5.61E-03	5.68E-03
k	EBFP	4	400	0.0E+00	3.23E-02	5.73E-03	5.83E-03
k	DBFN	5	400	1.0E-02	5.29E-02	5.77E-03	5.84E-03
k	TBFP	5	400	1.0E-02	5.29E-02	6.09E-03	6.15E-03
k	EBFP	5	400	1.0E-02	5.29E-02	6.10E-03	6.19E-03
k	EBFN	4	400	0.0E+00	3.23E-02	6.40E-03	6.57E-03
f	EBFN	5	400	3.0E-02	6.10E-02	9.74E-03	9.94E-03
f	TBFN	5	400	3.0E-02	6.10E-02	9.80E-03	9.99E-03
f	DBFN	5	400	3.0E-02	6.10E-02	9.97E-03	1.02E-02
f	TBFP	5	400	3.0E-02	6.10E-02	1.00E-02	1.02E-02
f	EBFP	5	400	3.0E-02	6.10E-02	1.00E-02	1.02E-02
f	TBFN	3	400	1.0E-02	4.75E-02	1.61E-02	1.61E-02
k	TBFP	3	400	1.0E-02	4.70E-02	1.62E-02	1.62E-02
k	EBFP	3	400	1.0E-02	4.70E-02	1.63E-02	1.63E-02
k	TBFN	3	400	1.0E-02	4.70E-02	1.63E-02	1.63E-02
k	EBFN	3	400	1.0E-02	4.70E-02	1.64E-02	1.64E-02
f	EBFP	4	400	3.0E-03	3.62E-02	6.50E-02	6.50E-02
k	DBFN	3	400	1.0E-02	4.70E-02	9.35E-01	9.32E-01
f	EBFN	3	400	1.0E-02	4.75E-02	1.00E+00	9.97E-01
f	EBFN	4	400	3.0E-03	3.62E-02	1.00E+00	9.97E-01
f	DBFN	3	500	3.0E-02	4.96E-02	1.79E-02	1.79E-02

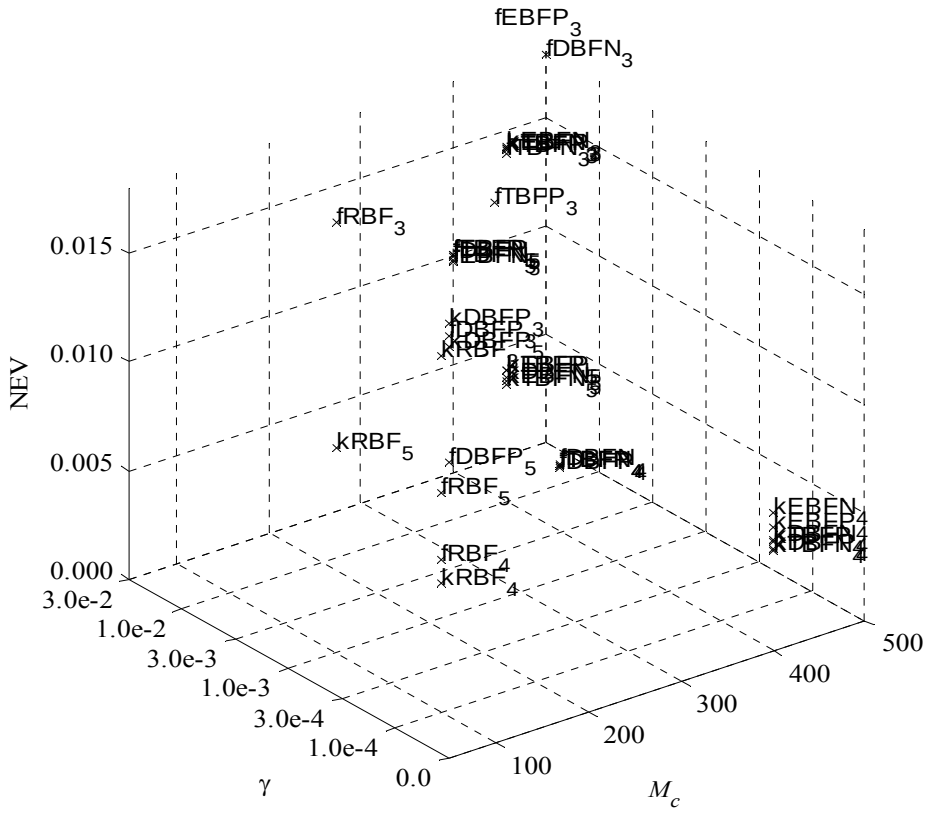

Figure 5.2 Scatter-plot of results using *k*-means clustering and FCM.

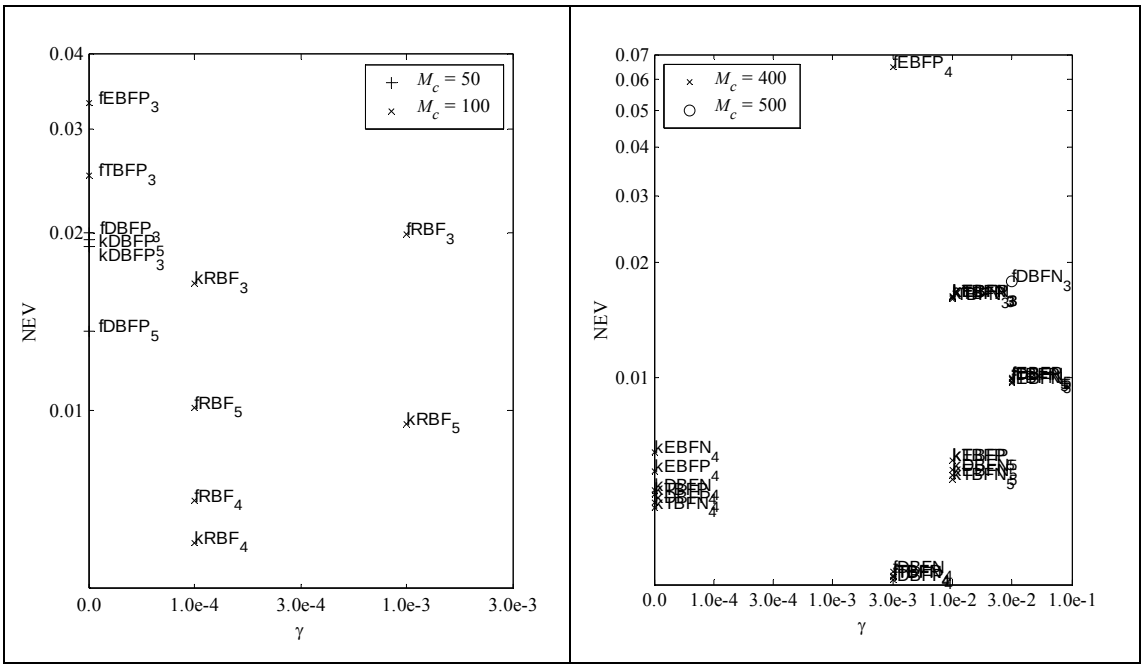

Figure 5.3 Plots of results in Figure 5.2 for $M_c \leq 100$, and $M_c \geq 200$, respectively.

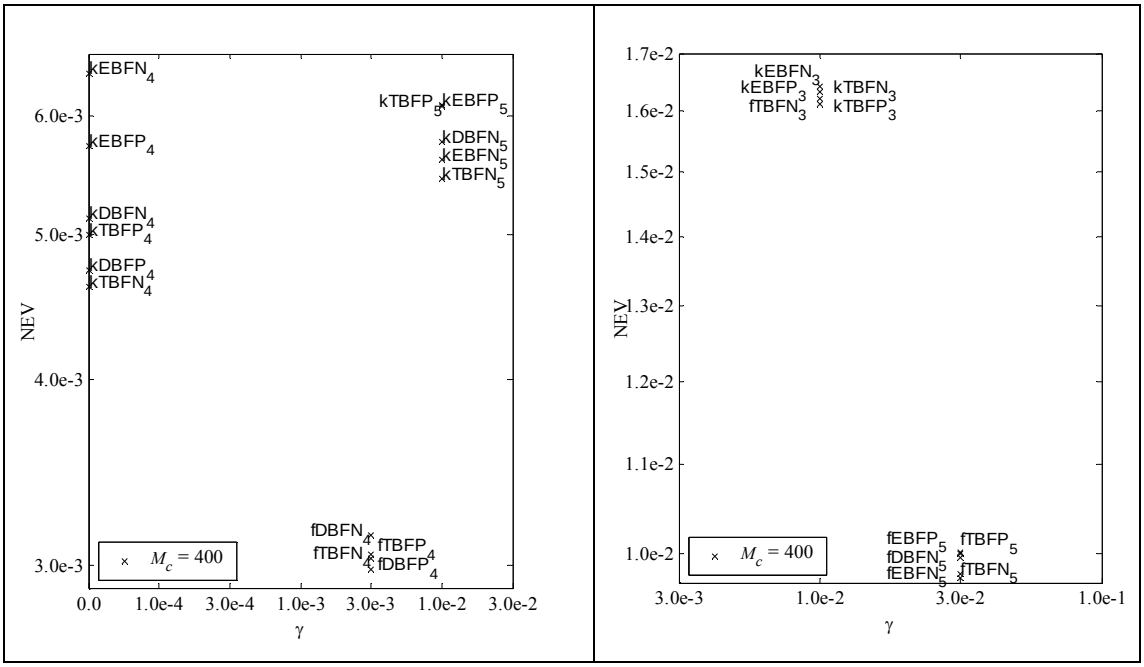

Figure 5.4 Plots of results in Figure 5.2 for $M_c \geq 200$ for different range of values of NEV.

It seems that for any given value of M_c, RBF and DBF seem to do well. If the criterion for deciding on the most suitable candidate to use is a good compromise between the number of centers used and the NEV, then it turns out that the best candidate is kRBF or fRBF. The use of non radial basis functions may result in smaller errors, but this is at the cost of using more centers. This is not a big problem here. However, the running time can be prohibitive if M_c is large (*i.e.* on large data sets), since the complexity of the linear layer is O(M^3).

On the other hand, the use of non-radial basis functions may occasionally result in very few centers required. This would result in little or no regularization required by the data driven algorithms. However, NEV would often be much higher. This suggests that when computing resources are scarce, using the data driven algorithms may occasionally result in the use of less centers, which makes the least squares stage

cheaper. Furthermore, since little or no regularization was required at times, further savings may result by omitting regularization altogether. The savings from this step can be quite significant for large matrices, as the complexity of matrix inversion is O(M^3), and thus it scales quite rapidly with size of the data set. However, it has to be noted that the error is much larger, which makes it unsuitable for iterated prediction, because the errors will grow exponentially. Note also that the TBF, DBF and EBF often perform worse when paired with *k*-means clustering. This is understandable, since it is more natural to pair them with clustering methods which utilize the Mahalanobis norm.

The simulations were repeated using hierarchical clustering (Table 5.6 and Table 5.7). Four variants were explored: Single linkage with Euclidean norm (hes), Single linkage with Mahalanobis norm (hms), Ward's method with Euclidean norm (hew), and Ward's method with Mahalanobis norm (hmw). The basis functions {RBF, TBFN, DBFN, EBFN, TBFP, DBFP, EBFP} were used. The corresponding scatter plot is Figure 5.5.

Compared to Table 5.5, using hierarchical clustering methods resulted in slightly less NEV when $M_c \leq 200$. The use of Mahalanobis norm and Ward's linkage (Table 5.7) for clustering, and coupled with the ordinary RBF algorithm appears to do quite well here; it achieves low values of NEV without requiring the use of too many centers. From Figure 5.6, the hewRBF trained using embedding dimension is $d_E = 4$ has the lowest NEV, which is similar to results obtained with $kRBF_4$ and $fRBF_4$ in Table 5.5. It appears that hierarchical clustering only confers marginal benefits, since the difference in performance between $kRBF_4$ and $hewRBF_4$ is slight.

Table 5.6 Training results using variants of hierarchical clustering (Euclidean norm)

clustering stage	basis function	d_E	M_c	γ	GE	NEV	NMSE
hes	TBFN	3	25	1.0E-04	2.25E-01	1.46E-01	1.46E-01
hes	EBFN	3	25	1.0E-04	2.25E-01	1.46E-01	1.46E-01
hew	DBFP	4	50	0.0E+00	4.79E-02	1.81E-02	1.82E-02
hew	DBFP	3	50	0.0E+00	3.77E-02	1.86E-02	1.88E-02
hew	DBFP	5	50	0.0E+00	4.37E-02	3.09E-02	3.09E-02
hew	TBFP	4	50	0.0E+00	6.67E-02	3.57E-02	3.66E-02
hew	EBFP	4	50	0.0E+00	6.67E-02	3.57E-02	3.66E-02
hes	TBFN	4	50	0.0E+00	2.87E-01	1.72E-01	1.74E-01
hes	EBFN	4	50	0.0E+00	2.87E-01	1.72E-01	1.74E-01
hes	TBFN	5	50	0.0E+00	2.63E-01	2.08E-01	2.12E-01
hes	EBFN	5	50	0.0E+00	2.63E-01	2.08E-01	2.12E-01
hew	RBF	4	100	3.0E-04	2.70E-02	5.59E-03	5.62E-03
hew	RBF	5	100	1.0E-03	5.10E-02	8.85E-03	8.92E-03
hew	RBF	3	100	1.0E-03	4.17E-02	1.98E-02	1.99E-02
hew	TBFP	3	100	0.0E+00	5.31E-02	2.27E-02	2.26E-02
hew	EBFP	3	100	0.0E+00	5.31E-02	2.27E-02	2.26E-02
hes	RBF	3	100	1.0E-03	4.75E-02	3.01E-02	3.12E-02
hes	TBFP	4	100	0.0E+00	1.82E-01	1.33E-01	1.35E-01
hes	EBFP	4	100	0.0E+00	1.82E-01	1.33E-01	1.35E-01
hes	DBFN	4	100	0.0E+00	1.69E-01	1.72E-01	1.72E-01
hes	DBFN	5	100	0.0E+00	1.69E-01	2.08E-01	2.08E-01
hes	DBFP	4	200	0.0E+00	8.62E-02	2.67E-02	2.66E-02
hes	RBF	4	200	3.0E-03	6.14E-02	3.66E-02	3.70E-02
hes	RBF	5	200	1.0E-04	5.86E-02	4.12E-02	4.14E-02
hes	DBFP	5	200	1.0E-04	9.82E-02	4.62E-02	4.61E-02
hes	DBFP	3	200	0.0E+00	6.45E-02	4.76E-02	4.76E-02
hes	TBFP	3	200	1.0E-04	1.75E-01	6.33E-02	6.37E-02
hes	EBFP	3	200	1.0E-04	1.75E-01	6.33E-02	6.37E-02
hew	TBFP	5	200	3.0E-03	6.98E-02	1.78E-01	1.78E-01
hew	EBFP	5	200	3.0E-03	6.98E-02	1.78E-01	1.78E-01
hew	TBFN	4	400	1.0E-02	7.39E-02	1.47E-02	1.47E-02
hew	DBFN	4	400	1.0E-02	7.39E-02	1.47E-02	1.47E-02
hew	EBFN	4	400	1.0E-02	7.39E-02	1.47E-02	1.47E-02
hew	TBFN	5	400	1.0E-02	8.38E-02	1.79E-02	1.78E-02
hew	DBFN	5	400	1.0E-02	8.38E-02	1.79E-02	1.78E-02
hew	EBFN	5	400	1.0E-02	8.38E-02	1.79E-02	1.78E-02
hew	TBFN	3	400	1.0E-01	6.39E-02	2.03E-02	2.03E-02
hew	DBFN	3	400	1.0E-01	6.39E-02	2.03E-02	2.03E-02
hew	EBFN	3	400	1.0E-01	6.39E-02	2.03E-02	2.03E-02
hes	DBFN	3	400	0.0E+00	9.65E-02	6.70E-02	6.68E-02
hes	TBFP	5	400	1.0E-03	2.29E-01	1.58E-01	1.58E-01
hes	EBFP	5	400	1.0E-03	2.29E-01	1.58E-01	1.58E-01

Table 5.7 Training results using variants of hierarchical clustering (Mahalanobis norm)

clustering stage	basis function	d_E	M_c	γ	GE	NEV	NMSE
hms	TBFN	3	10	1.0E-04	3.65E-01	3.35E-01	3.35E-01
hms	EBFN	3	10	1.0E-04	3.65E-01	3.35E-01	3.35E-01
hmw	DBFP	4	50	0.0E+00	4.87E-02	1.35E-02	1.35E-02
hmw	RBF	3	50	1.0E-04	4.09E-02	2.04E-02	2.05E-02
hmw	TBFP	3	50	0.0E+00	5.26E-02	2.66E-02	2.65E-02
hmw	EBFP	3	50	0.0E+00	5.26E-02	2.66E-02	2.65E-02
hmw	TBFP	4	50	0.0E+00	6.39E-02	3.95E-02	4.03E-02
hmw	EBFP	4	50	0.0E+00	6.39E-02	3.95E-02	4.03E-02
hms	TBFN	4	50	0.0E+00	1.99E-01	1.24E-01	1.30E-01
hms	EBFN	4	50	0.0E+00	1.99E-01	1.24E-01	1.30E-01
hms	TBFP	3	50	0.0E+00	1.30E-01	2.15E-01	2.15E-01
hms	EBFP	3	50	0.0E+00	1.30E-01	2.15E-01	2.15E-01
hmw	RBF	4	100	1.0E-04	2.52E-02	6.11E-03	6.10E-03
hmw	RBF	5	100	3.0E-04	4.48E-02	7.25E-03	7.27E-03
hmw	DBFP	5	100	0.0E+00	5.27E-02	1.83E-02	1.82E-02
hmw	DBFP	3	100	0.0E+00	4.18E-02	2.56E-02	2.58E-02
hms	TBFN	5	100	1.0E-04	3.10E-01	3.71E-01	3.70E-01
hms	EBFN	5	100	1.0E-04	3.10E-01	3.71E-01	3.70E-01
hms	RBF	3	200	3.0E-02	5.20E-02	2.23E-02	2.23E-02
hms	RBF	4	200	1.0E-04	8.99E-02	9.10E-02	9.16E-02
hmw	DBFN	3	200	1.0E-04	1.09E-01	9.11E-02	9.08E-02
hms	DBFP	5	200	0.0E+00	9.62E-02	1.29E-01	1.29E-01
hms	RBF	5	200	3.0E-03	1.09E-01	1.48E-01	1.49E-01
hms	DBFN	3	200	1.0E-04	1.30E-01	2.30E-01	2.29E-01
hmw	TBFN	4	400	3.0E-02	9.41E-02	2.51E-02	2.51E-02
hmw	DBFN	4	400	3.0E-02	9.41E-02	2.51E-02	2.51E-02
hmw	EBFN	4	400	3.0E-02	9.41E-02	2.51E-02	2.51E-02
hmw	TBFP	5	400	3.0E-02	9.66E-02	2.84E-02	2.84E-02
hmw	EBFP	5	400	3.0E-02	9.66E-02	2.84E-02	2.84E-02
hmw	TBFN	5	400	3.0E-02	9.66E-02	2.84E-02	2.84E-02
hmw	DBFN	5	400	3.0E-02	9.66E-02	2.84E-02	2.84E-02
hmw	EBFN	5	400	3.0E-02	9.66E-02	2.84E-02	2.84E-02
hms	DBFP	4	400	1.0E-04	9.96E-02	6.58E-02	6.68E-02
hms	DBFN	4	400	3.0E-04	1.62E-01	1.35E-01	1.35E-01
hms	TBFP	4	400	0.0E+00	1.65E-01	1.47E-01	1.47E-01
hms	EBFP	4	400	0.0E+00	1.65E-01	1.47E-01	1.47E-01
hms	DBFN	5	400	0.0E+00	1.69E-01	1.48E-01	1.48E-01
hms	TBFP	5	400	3.0E-04	2.14E-01	1.56E-01	1.55E-01
hms	EBFP	5	400	3.0E-04	2.14E-01	1.56E-01	1.55E-01
hmw	TBFN	3	400	3.0E-04	5.00E-02	9.96E-01	9.93E-01
hmw	EBFN	3	400	3.0E-04	5.00E-02	9.96E-01	9.93E-01
hms	DBFP	3	500	0.0E+00	7.52E-02	9.35E-02	9.34E-02

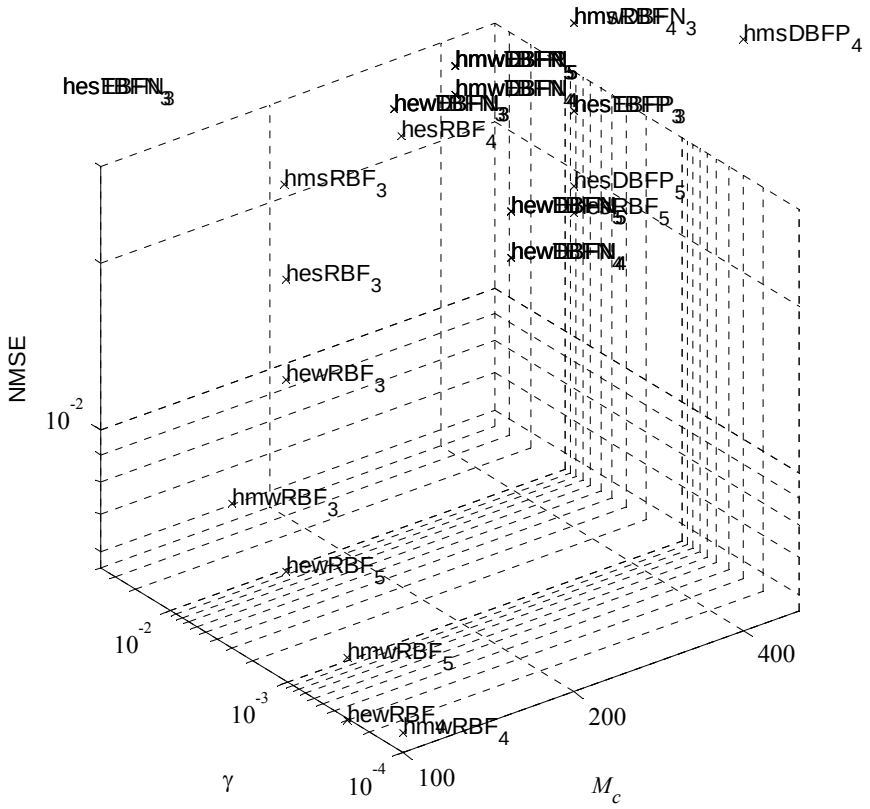

Figure 5.5 Scatter-plot of results using hierarchical clustering.

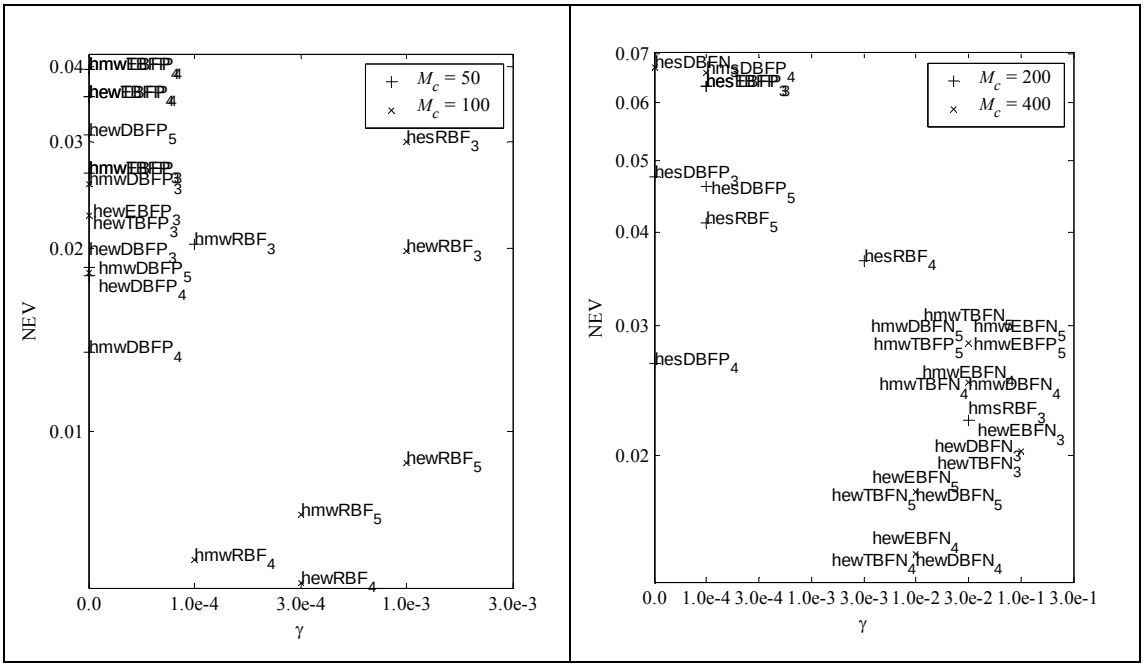

Figure 5.6 Plots of results in Figure 5.5 for $M_c \leq 100$, and $M_c \geq 200$, respectively.

The use of hew coupled with non-radial basis functions appeared to do well. It seems counterintuitive, but the use of hmw combined with non-radial basis functions did not appear to do well. There were also several clusters in Figure 5.6, whereby the use of identical clustering algorithms and values of d_E, coupled with different variants of the basis functions resulted in virtually identical results. One such cluster was hewTBFN$_4$, hewEBFN$_4$ and hewDBFN$_4$. The text labels had been manually rearranged for the sake of readability.

One conclusion that can be drawn was the single linkage did not appear to be very useful. The results in Figure 5.6 indicate that Ward's method does better than single linkage, indicating that clusters of relatively equal sizes and shapes are favoured.

Perhaps, the important issue is the balance between spherical and elliptical clusters, and one way to investigate this is to explore the use of Babuska's method of clustering which also obtains ellipsoidal clusters (Section 3.3), with the shape of the clusters determined by γ_c. The results are recorded in Table 5.8 and Table 5.9, with corresponding scatter-plots in Figure 5.7 and Figure 5.8. The effect of changing the clustering regularization parameter γ_c was explored; $\gamma_c = 0.05$ (bh), $\gamma_c = 0.1$ (b1) and $\gamma_c = 0.2$ (b2). The basis functions {RBF, TBFN, DBFN, EBFN, TBFP, DBFP, EBFP} were used. Interestingly, the RBFs in Table 5.8 end up with $M_c = 100$, and performance which is slightly better than in Table 5.5. Thus, the pairing of non-Euclidean clustering methods with the ordinary RBF can be successful.

It seems that $\gamma_c = 0.2$ does better than $\gamma_c = 0.05$ when M_c is small, but the situation is reversed when M_c is large. Since higher values of γ_c indicates more spherical clusters,

this suggests that ellipsoidal clusters do better when M_c is large. However, in such a situation, many of the clusters are effectively singletons (with covariance matrix set to **I**), since the ratio of points to centers is close to 1. The results for hierarchical clustering, and for Babuska's method suggest that non-Euclidean methods of clustering do not seem to confer significant benefits.

Table 5.8 Training results using Babuska's method of clustering ($M_c \leq 100$)

clustering stage	basis function	d_E	M_c	γ	GE	NEV	NMSE
b2	TBFN	3	25	0.0E+00	7.04E-02	4.96E-02	4.94E-02
bh	EBFN	3	25	0.0E+00	7.75E-02	5.08E-02	5.13E-02
bh	TBFN	3	25	0.0E+00	7.75E-02	5.10E-02	5.09E-02
b1	TBFN	3	25	0.0E+00	8.32E-02	5.20E-02	5.18E-02
b1	DBFN	3	25	0.0E+00	8.74E-02	5.47E-02	5.46E-02
b1	EBFN	3	25	0.0E+00	8.32E-02	5.94E-02	5.92E-02
b2	EBFN	3	25	0.0E+00	7.04E-02	6.27E-02	6.27E-02
b2	DBFN	3	25	0.0E+00	8.57E-02	6.85E-02	6.83E-02
bh	DBFN	3	25	0.0E+00	8.11E-02	7.89E-02	7.87E-02
b1	DBFP	4	50	0.0E+00	3.79E-02	1.10E-02	1.10E-02
b2	DBFP	4	50	0.0E+00	3.46E-02	1.14E-02	1.14E-02
bh	DBFP	5	50	0.0E+00	4.50E-02	1.41E-02	1.41E-02
b1	DBFP	5	50	0.0E+00	4.74E-02	1.64E-02	1.66E-02
b1	RBF	3	50	0.0E+00	4.25E-02	1.86E-02	1.87E-02
b2	DBFP	5	50	0.0E+00	4.23E-02	2.07E-02	2.08E-02
bh	DBFP	3	50	0.0E+00	4.43E-02	2.13E-02	2.13E-02
bh	EBFP	3	50	0.0E+00	5.18E-02	2.19E-02	2.21E-02
bh	TBFP	3	50	0.0E+00	5.18E-02	2.22E-02	2.22E-02
b2	TBFP	3	50	0.0E+00	4.70E-02	2.41E-02	2.43E-02
b2	EBFP	3	50	0.0E+00	4.70E-02	2.48E-02	2.48E-02
b2	DBFP	3	50	0.0E+00	3.70E-02	2.50E-02	2.50E-02
b1	RBF	4	100	1.0E-03	2.98E-02	7.66E-03	7.70E-03
bh	RBF	4	100	3.0E-04	2.79E-02	7.99E-03	8.03E-03
b2	RBF	4	100	3.0E-04	2.97E-02	8.33E-03	8.44E-03
bh	RBF	5	100	1.0E-03	4.49E-02	9.49E-03	9.55E-03
b2	RBF	5	100	1.0E-03	4.35E-02	1.13E-02	1.14E-02
b1	RBF	5	100	1.0E-03	4.33E-02	1.27E-02	1.28E-02
bh	RBF	3	100	1.0E-04	3.94E-02	1.64E-02	1.65E-02
b2	RBF	3	100	1.0E-03	4.08E-02	1.65E-02	1.65E-02

The non-radial basis functions do appear to provide a good alternative when small numbers of centers are required, or when NEV has to be as small as possible.

However, the performance of the non-radial basis functions compared to the ordinary RBF did not appear to be markedly better. One possible reason is that non-radial basis functions are prone to numerical problems.

Table 5.9 Training results using Babuska's method of clustering ($M_c \geq 200$)

clustering stage	basis function	d_E	M_c	γ	GE	NEV	NMSE
b1	TBFN	5	200	1.0E-02	5.35E-02	1.28E-02	1.28E-02
b1	EBFN	5	200	1.0E-02	5.35E-02	1.00E+00	9.97E-01
b2	TBFN	5	200	1.0E-02	5.36E-02	1.00E+00	9.97E-01
b2	EBFN	5	200	1.0E-02	5.36E-02	1.00E+00	9.97E-01
bh	TBFP	5	400	1.0E-02	5.44E-02	7.96E-03	7.98E-03
bh	TBFN	5	400	1.0E-02	5.44E-02	8.20E-03	8.21E-03
bh	EBFP	5	400	1.0E-02	5.44E-02	8.53E-03	8.54E-03
b1	TBFP	5	400	1.0E-02	5.58E-02	9.21E-03	9.20E-03
bh	DBFN	5	400	1.0E-02	5.44E-02	9.44E-03	9.43E-03
b1	DBFN	5	400	1.0E-02	5.58E-02	9.72E-03	9.71E-03
b1	EBFP	5	400	1.0E-02	5.58E-02	9.75E-03	9.74E-03
bh	EBFN	5	400	1.0E-02	5.44E-02	1.06E-02	1.06E-02
b2	TBFP	5	400	3.0E-02	5.68E-02	1.48E-02	1.48E-02
b2	DBFN	5	400	3.0E-02	5.68E-02	1.63E-02	1.63E-02
b2	EBFP	5	400	3.0E-02	5.68E-02	1.02E-01	1.02E-01
bh	DBFP	4	500	1.0E-02	4.04E-02	5.37E-03	5.36E-03
bh	TBFP	4	500	1.0E-02	4.04E-02	5.52E-03	5.52E-03
bh	DBFN	4	500	1.0E-02	4.04E-02	5.55E-03	5.54E-03
b2	TBFP	4	500	1.0E-02	4.14E-02	6.00E-03	6.01E-03
b1	TBFN	4	500	1.0E-02	4.16E-02	6.21E-03	6.22E-03
b1	EBFN	4	500	1.0E-02	4.16E-02	6.33E-03	6.33E-03
bh	EBFN	4	500	1.0E-02	4.04E-02	6.42E-03	6.41E-03
bh	TBFN	4	500	1.0E-02	4.04E-02	6.81E-03	6.79E-03
b1	DBFN	4	500	1.0E-02	4.16E-02	6.91E-03	6.89E-03
bh	EBFP	4	500	1.0E-02	4.04E-02	7.43E-03	7.42E-03
b1	TBFP	4	500	1.0E-02	4.16E-02	7.61E-03	7.59E-03
b1	EBFP	4	500	1.0E-02	4.16E-02	7.68E-03	7.66E-03
b2	EBFP	4	500	1.0E-02	4.14E-02	8.05E-03	8.03E-03
b1	DBFP	3	500	1.0E-04	3.31E-02	7.95E-02	7.93E-02
b2	DBFN	4	500	1.0E-02	4.14E-02	2.04E-01	2.03E-01
b2	EBFN	4	500	1.0E-02	4.14E-02	2.15E-01	2.15E-01
b1	EBFP	3	500	1.0E-04	3.99E-02	2.63E-01	2.63E-01
b1	TBFP	3	500	1.0E-04	3.99E-02	3.02E-01	3.01E-01
b2	TBFN	4	500	1.0E-02	4.14E-02	9.99E-01	9.97E-01

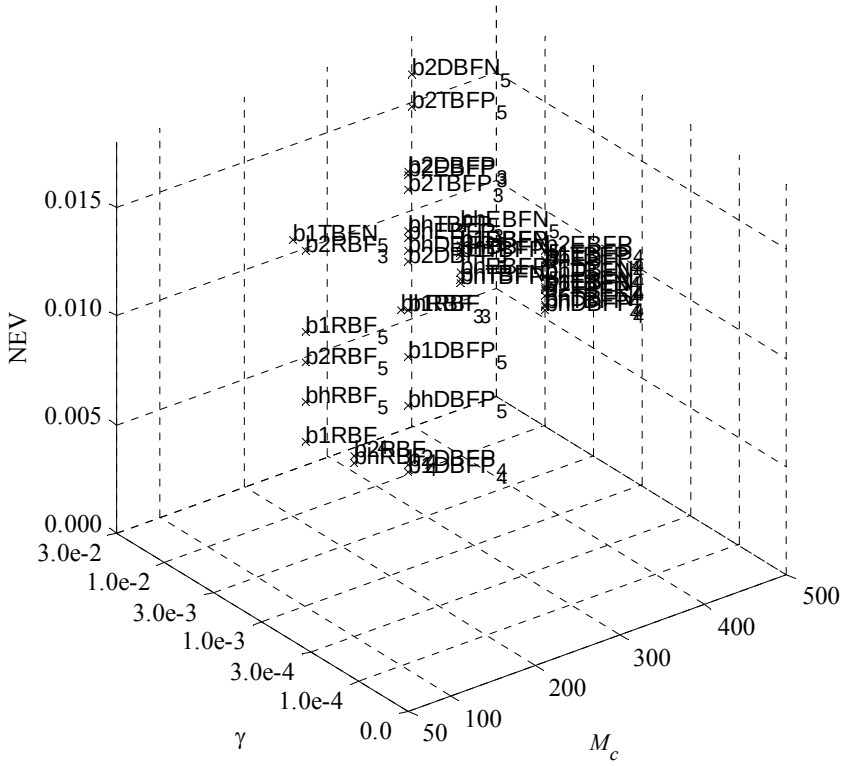

Figure 5.7 Scatter-plot of results using Babuska's method of clustering.

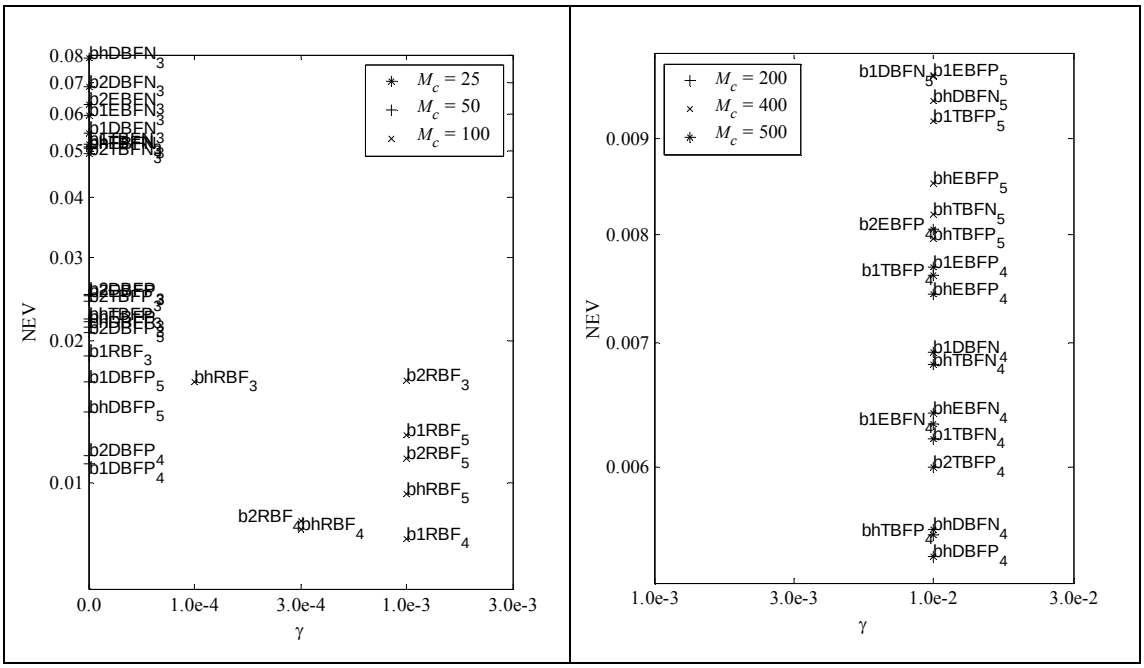

Figure 5.8 Plots of results in Figure 5.7 for $M_c \leq 100$, and $M_c \geq 200$, respectively.

One way to check how important numerical issues are, is to examine the performance of EBFs when the covariance matrices are regularized, as in Eq. (3.41) and Eq. (3.42).

Simulation results are recorded in Table 5.10 and Table 5.11, with corresponding scatter-plots in Figure 5.9 and Figure 5.10. Due to the large number of clustering methods tested, {k, f, hes, hms, hew, hmw, bh, b1, b2}, only the EBFP and EBFN are tested. The regularization parameters γ_a and γ_b are varied. These are represented by the superscripts $\gamma_a = 0.05$ (ah), $\gamma_a = 0.1$ (a1), $\gamma_a = 0.2$ (a2), $\gamma_b = 0.05$ (bh), $\gamma_b = 0.1$ (b1) and $\gamma_b = 0.2$ (b2). As usual, the subscripts refer to the embedding dimension. For example, $\text{b1EBFP}_3^{\text{b2}}$ refers to the combination of Babuska's method of clustering with $\gamma_c = 0.1$, using EBFP, with $\gamma_a = 0$, $\gamma_b = 0.2$, and $d_E = 3$. The total of 324 combinations had been reduced to a more manageable 90 combinations by displaying only the top 15 performers (low NEV) for each value of M_c in Table 5.10 and Table 5.11.

Figure 5.10 illustrates that though regularized covariance matrices do not result in significantly better performance compared to kRBF, they do allow EBFP and EBFN to perform better than what the results in Table 5.5 to Table 5.9 suggest. Hence, it is likely that numerical issues are the main reason why the non-radial basis functions do not seem to perform better than the ordinary RBF in Table 5.5 to Table 5.9. Incidentally, Figure 5.10 shows that many EBFs have $\gamma = 0$. This suggests that when regularized covariance matrices are used, it may be possible to do away with regularization, in order to reduce computational load.

Table 5.10 Training results using regularized covariance matrices ($M_c \leq 100$)

clustering stage	basis function	γ_b	γ_a	d_E	M_c	γ	GE	NEV	NMSE
hms	EBFN	0	0.05	3	10	1.0E-04	3.72E-01	3.35E-01	3.35E-01
hms	EBFN	0	0.1	3	10	1.0E-04	3.79E-01	3.35E-01	3.35E-01
hms	EBFN	0	0.2	3	10	1.0E-04	3.96E-01	3.35E-01	3.35E-01
bh	EBFN	0	0.2	3	25	0.0E+00	7.46E-02	4.87E-02	4.87E-02
b2	EBFN	0	0.1	3	25	0.0E+00	7.00E-02	4.95E-02	4.93E-02
b1	EBFN	0	0.2	3	25	0.0E+00	8.24E-02	5.00E-02	4.99E-02
bh	EBFN	0	0.05	3	25	0.0E+00	7.55E-02	5.09E-02	5.08E-02
b1	EBFN	0	0.1	3	25	0.0E+00	8.32E-02	5.16E-02	5.16E-02
b1	EBFN	0	0.05	3	25	0.0E+00	8.32E-02	5.20E-02	5.18E-02
bh	EBFN	0	0.1	3	25	0.0E+00	7.47E-02	5.37E-02	5.38E-02
b2	EBFN	0	0.05	3	25	0.0E+00	7.04E-02	6.54E-02	6.59E-02
b2	EBFN	0	0.2	3	25	0.0E+00	7.04E-02	1.27E-01	1.27E-01
hes	EBFN	0	0.05	3	25	1.0E-04	2.25E-01	1.46E-01	1.46E-01
hes	EBFN	0	0.1	3	25	1.0E-04	2.25E-01	1.46E-01	1.46E-01
hes	EBFN	0	0.2	3	25	1.0E-04	2.25E-01	1.46E-01	1.46E-01
b1	EBFN	0.1	0	4	50	0.0E+00	4.07E-02	1.26E-02	1.27E-02
f	EBFN	0.1	0	4	50	0.0E+00	3.59E-02	1.40E-02	1.40E-02
b2	EBFP	0.05	0	4	50	0.0E+00	3.98E-02	1.40E-02	1.43E-02
f	EBFP	0.1	0	4	50	0.0E+00	3.59E-02	1.42E-02	1.42E-02
b1	EBFN	0.2	0	4	50	0.0E+00	3.27E-02	1.44E-02	1.45E-02
b1	EBFP	0.05	0	4	50	0.0E+00	3.98E-02	1.53E-02	1.54E-02
hew	EBFN	0.05	0	4	50	0.0E+00	3.59E-02	1.72E-02	1.78E-02
b1	EBFP	0.2	0	4	50	0.0E+00	3.27E-02	1.77E-02	1.78E-02
hmw	EBFN	0.1	0	4	50	0.0E+00	3.81E-02	1.78E-02	1.79E-02
b1	EBFN	0.05	0	4	50	0.0E+00	4.02E-02	1.91E-02	1.97E-02
f	EBFP	0.05	0	3	50	0.0E+00	4.52E-02	2.07E-02	2.07E-02
bh	EBFP	0.05	0	3	50	0.0E+00	4.26E-02	2.21E-02	2.21E-02
f	EBFN	0.05	0	3	50	0.0E+00	4.17E-02	2.25E-02	2.25E-02
k	EBFN	0.05	0	3	50	0.0E+00	4.51E-02	2.26E-02	2.26E-02
b2	EBFN	0.05	0	4	50	0.0E+00	4.08E-02	2.32E-02	2.33E-02
k	EBFP	0.2	0	4	100	0.0E+00	2.72E-02	8.34E-03	8.33E-03
hew	EBFP	0.2	0	4	100	0.0E+00	2.79E-02	8.62E-03	8.69E-03
hmw	EBFP	0.2	0	4	100	0.0E+00	3.41E-02	9.53E-03	9.50E-03
hew	EBFN	0.2	0	4	100	0.0E+00	3.40E-02	9.80E-03	9.89E-03
hmw	EBFN	0.2	0	4	100	0.0E+00	3.37E-02	1.07E-02	1.07E-02
bh	EBFP	0.2	0	4	100	0.0E+00	3.25E-02	1.13E-02	1.13E-02
bh	EBFN	0.2	0	4	100	0.0E+00	3.25E-02	1.15E-02	1.15E-02
k	EBFN	0.2	0	4	100	0.0E+00	3.18E-02	1.27E-02	1.28E-02
hmw	EBFN	0.05	0	4	100	0.0E+00	4.17E-02	1.62E-02	1.64E-02
k	EBFP	0.1	0	3	100	0.0E+00	4.50E-02	1.74E-02	1.75E-02
bh	EBFN	0.2	0	3	100	0.0E+00	4.22E-02	1.75E-02	1.75E-02
f	EBFP	0.1	0	3	100	0.0E+00	4.16E-02	1.76E-02	1.77E-02
hmw	EBFP	0.05	0	3	100	0.0E+00	3.99E-02	1.77E-02	1.80E-02
f	EBFN	0.1	0	3	100	0.0E+00	4.41E-02	1.80E-02	1.80E-02
k	EBFP	0.05	0	3	100	0.0E+00	4.57E-02	1.80E-02	1.83E-02

Table 5.11 Training results using regularized covariance matrices ($M_c \geq 200$)

clustering stage	basis function	γ_b	γ_a	d_E	M_c	γ	GE	NEV	NMSE
k	EBFN	0.2	0	5	200	3.0E-04	5.06E-02	3.60E-03	3.61E-03
b1	EBFN	0	0.05	5	200	1.0E-02	5.35E-02	1.09E-02	1.11E-02
b1	EBFN	0	0.1	5	200	1.0E-02	5.35E-02	1.30E-02	1.31E-02
k	EBFP	0.2	0	5	200	3.0E-04	4.86E-02	1.31E-02	1.34E-02
hew	EBFP	0.1	0	4	200	3.0E-04	3.19E-02	1.35E-02	1.39E-02
b2	EBFP	0.2	0	5	200	3.0E-04	5.39E-02	1.46E-02	1.50E-02
f	EBFP	0.2	0	4	200	3.0E-04	3.17E-02	1.54E-02	1.60E-02
hmw	EBFP	0.2	0	5	200	3.0E-04	3.93E-02	1.57E-02	1.61E-02
b2	EBFN	0.2	0	4	200	1.0E-04	3.72E-02	1.57E-02	1.71E-02
hew	EBFP	0.2	0	3	200	0.0E+00	4.06E-02	1.79E-02	1.78E-02
hew	EBFP	0.05	0	4	200	0.0E+00	3.17E-02	1.86E-02	1.86E-02
hmw	EBFP	0.05	0	4	200	1.0E-04	3.43E-02	1.91E-02	1.92E-02
f	EBFP	0.2	0	3	200	0.0E+00	4.23E-02	1.92E-02	1.92E-02
f	EBFN	0.2	0	3	200	0.0E+00	3.86E-02	2.04E-02	2.04E-02
hmw	EBFP	0.1	0	4	200	3.0E-04	3.07E-02	2.08E-02	2.12E-02
f	EBFN	0	0.2	4	400	3.0E-03	3.62E-02	2.98E-03	2.98E-03
f	EBFN	0.05	0	4	400	3.0E-03	3.62E-02	3.00E-03	2.99E-03
f	EBFP	0	0.2	4	400	3.0E-03	3.62E-02	3.03E-03	3.02E-03
k	EBFN	0.05	0	4	400	0.0E+00	3.23E-02	4.64E-03	4.67E-03
k	EBFN	0.1	0	4	400	0.0E+00	3.23E-02	4.91E-03	4.96E-03
k	EBFN	0.05	0	5	400	1.0E-02	5.29E-02	5.40E-03	5.47E-03
k	EBFN	0.1	0	5	400	1.0E-02	5.29E-02	5.49E-03	5.56E-03
k	EBFP	0	0.2	5	400	1.0E-02	5.29E-02	5.51E-03	5.58E-03
k	EBFP	0	0.05	5	400	1.0E-02	5.29E-02	5.55E-03	5.62E-03
k	EBFN	0	0.05	5	400	1.0E-02	5.29E-02	5.55E-03	5.63E-03
k	EBFP	0	0.1	5	400	1.0E-02	5.29E-02	5.57E-03	5.64E-03
k	EBFP	0	0.2	4	400	0.0E+00	3.23E-02	5.57E-03	5.80E-03
k	EBFP	0	0.1	4	400	0.0E+00	3.23E-02	5.61E-03	5.83E-03
k	EBFN	0	0.05	4	400	0.0E+00	3.23E-02	5.78E-03	5.90E-03
k	EBFN	0	0.2	4	400	0.0E+00	3.23E-02	5.80E-03	5.91E-03
bh	EBFP	0	0.2	4	500	1.0E-02	4.04E-02	3.56E-03	3.55E-03
bh	EBFN	0	0.2	4	500	1.0E-02	4.04E-02	3.63E-03	3.63E-03
bh	EBFP	0.05	0	4	500	1.0E-02	4.04E-02	3.73E-03	3.73E-03
bh	EBFN	0.1	0	4	500	1.0E-02	4.04E-02	3.88E-03	3.87E-03
bh	EBFN	0	0.1	4	500	1.0E-02	4.04E-02	3.96E-03	3.95E-03
bh	EBFP	0	0.1	4	500	1.0E-02	4.04E-02	3.98E-03	3.97E-03
bh	EBFN	0	0.05	4	500	1.0E-02	4.04E-02	4.00E-03	4.01E-03
bh	EBFP	0	0.05	4	500	1.0E-02	4.04E-02	4.17E-03	4.15E-03
b1	EBFN	0	0.1	4	500	1.0E-02	4.16E-02	5.28E-03	5.28E-03
b1	EBFP	0	0.1	4	500	1.0E-02	4.16E-02	6.25E-03	6.24E-03
b1	EBFP	0	0.05	4	500	1.0E-02	4.16E-02	6.32E-03	6.31E-03
b1	EBFP	0	0.2	4	500	1.0E-02	4.16E-02	6.37E-03	6.36E-03
bh	EBFN	0.05	0	4	500	1.0E-02	4.04E-02	6.37E-03	6.36E-03
b1	EBFN	0	0.2	4	500	1.0E-02	4.16E-02	7.03E-03	7.02E-03
b1	EBFN	0	0.05	4	500	1.0E-02	4.16E-02	7.24E-03	7.22E-03

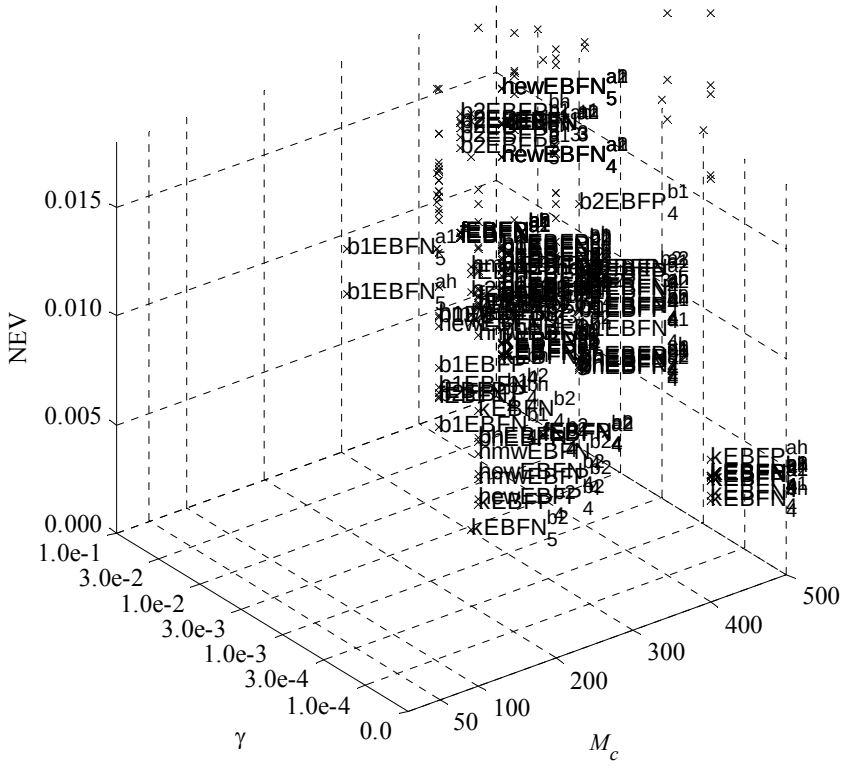

Figure 5.9 Scatter-plot of results using regularized covariance matrices.

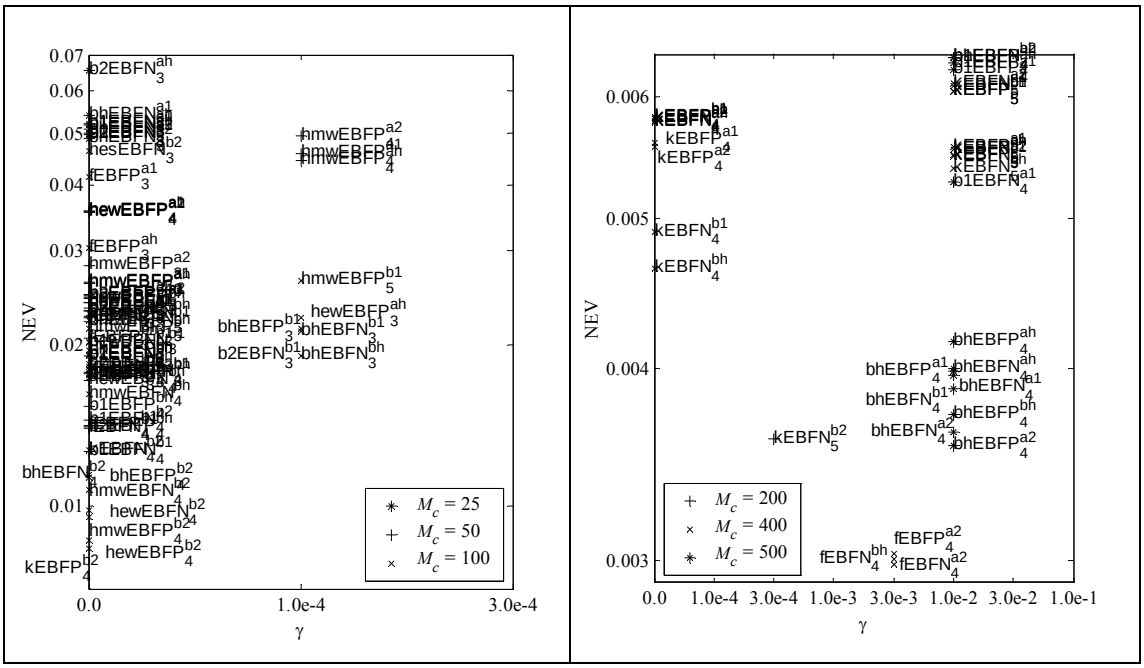

Figure 5.10 Plots of results in Figure 5.9 for $M_c \leq 100$, and $M_c \geq 200$, respectively.

The main conclusion which can be drawn from this section is that the kRBF appears to offer a good compromise between the number of centers used, and performance on the test set, as measured by NEV or NMSE (both NEV and NMSE are highly correlated in Table 5.5 to Table 5.9).

5.3.1 Committee machine

Table 5.12 shows GE obtained for Babuska's method of clustering with M_c fixed at 50 centers, $d_E = 3$, and $\gamma_c = 0.1$, on SFA for both EBFP and EBFN. It demonstrates the possibility that either EBFP or EBFN will fail (due to normalization of data to mean 0 and variance 1, GE ≈ 1 is close to performance obtained when predicting white noise), because of the inadequacies in Eq. (3.85) or Eq. (3.86). The failure was not due to clustering, because all GE values were obtained from the same set of clustering results. When either algorithm fails, the other algorithm may perform better. Thus, it may be advisable to choose either EBFP or EBFN, depending on which has less generalization error. This is like a "committee machine", which combines the outputs of several neural networks [98]. Caching the clustering stage results in reduced computational demands if "committee machines" are used.

Table 5.12 Generalization Errors (GE) using Babuska's algorithm

γ	GE (EBFP)	GE (EBFN)
0.0E+00	5.51E-02	5.21E-01
1.0E-04	5.95E-02	5.08E-01
3.0E-04	6.36E-02	7.08E-01
1.0E-03	5.00E-01	1.08E+00
3.0E-03	9.15E-02	1.03E+00
1.0E-02	3.09E-01	1.01E+00
3.0E-02	2.90E-01	1.01E+00
1.0E-01	1.69E+00	1.01E+00
3.0E-01	5.36E+01	1.01E+00

5.3.2 Effect of Varying SNR

Simulations were run for Lorenz data with AWGN of varying SNR. The Lorenz system is isomorphic to the laser system from which SFA was derived, and it would be interesting to compare the results. It turns out that Lorenz data requires more centers to model. The Lorenz attractor has two "lobes", whilst the SFA attractor has only one "lobe", possibly accounting for the fact that more centers are required to model the Lorenz data. It seems unavoidable that the complexity of the neural network model corresponds to the complexity of the attractor is in state space, rather than the complexity of the underlying equations.

Table 5.13 Variation of GE and NEV of kRBF with SNR for Lorenz data

SNR	M_c	γ	GE	NEV
-5	25	3.0E-01	8.82E-01	8.57E-01
10	400	3.0E-01	1.22E-01	1.26E-01
20	400	1.0E-01	1.36E-02	1.44E-02
25	800	3.0E-02	4.64E-03	4.77E-03
30	800	1.0E-02	1.52E-03	1.57E-03
999	1200	0.0E+00	7.16E-07	1.63E-06

Table 5.13, shows that SNR influences the number of centers required M_c; M_c decreased with decreasing SNR. Consider Figure 3.8; if the noise floor is higher, less detail will be recoverable. This in turn means that fewer centers will be required. Interestingly, even for negative SNR, the NEV remained at less than 1. This meant that some learning still took place, despite the limited data length, and the overpowering presence of noise.

5.3.3 Sea Clutter

The same procedures were applied to sea clutter data in low sea state (`lo.dat`), in-phase component, where N^{total} = 8192. The kRBF with caching was used. The embedding dimension was varied, because the embedding dimension is approximately 5, for both low and high sea state. GE was very low, if compared to Table 5.5 and Table 5.13.

Table 5.14 Training Results for Sea Clutter (Low Sea State)

d_E	M_c	γ	GE	NEV	NMSE
4	400	0.00E+00	2.16E-03	2.45E-03	2.45E-03
5	400	3.00E-04	2.45E-03	3.01E-03	3.02E-03
6	400	0.00E+00	2.65E-03	3.19E-03	3.19E-03

Table 5.15 Training Results for Sea Clutter (High Sea State)

d_E	M_c	γ	GE	NEV	NMSE
4	50	1.00E-03	2.06E-02	2.75E-02	2.75E-02
5	100	3.00E-03	2.36E-02	2.71E-02	2.71E-02
6	100	3.00E-04	2.62E-02	2.63E-02	2.63E-02

Similarly, the same procedures were applied to sea clutter data in high sea state (`hi.dat`), in-phase component, for N^{total} = 8192. The kRBF with caching was used. Curiously, less centers were required; this probably resulted in higher values of NEV and NMSE. As expected, the NMSE is higher, since high sea state corresponds to a rough sea. Interestingly, the number of centers required were much less, than for low sea state. This is reasonable, as overfitting to the noise may occur otherwise.

5.4 Dynamic Reconstruction

In phase space, a noiseless chaotic system has the mapping $\mathbf{F}(\bullet)$ such that

$$\mathbf{\Psi}(n+1) = \mathbf{F}\left(\mathbf{\Psi}(n)\right). \tag{5.5}$$

Equivalently, the time series has a mapping $y_{i+1} = f(\boldsymbol{\psi}_i)$. Iterated prediction is the use of the neural network that was trained with one-step prediction to make multi-step predictions by feeding its output into its input. This results in

$$\hat{y}_{i+1}^{iter} = \hat{f}\left(\boldsymbol{\psi}_i^{iter}\right), \tag{5.6}$$

where $\hat{y}_{i+1}^{iter}$ is the output from iterated prediction, $\hat{f}(\bullet)$ is the mapping learnt from 1-step prediction, and $\boldsymbol{\psi}_i^{iter} = \left(\hat{y}_i^{iter}, \quad \hat{y}_{i-1}^{iter}, \quad \cdots \quad \hat{y}_{i-(p-1)}^{iter}\right)$. From the computer science point of view, a recursive function means a function which calls itself repeatedly, and term "recursive prediction" is also used in the literature [127].

From a sequence of N^{iter} successive values of $\hat{y}_{i+1}^{iter}$, it is possible to generate the embedding

$$\mathbf{\Psi}_{i+\tau}^{iter} = \hat{\mathbf{F}}\left(\mathbf{\Psi}_i^{iter}\right), \tag{5.7}$$

whereby $\mathbf{\Psi}_i^{iter} = \left(\hat{y}_i^{iter}, \quad \hat{y}_{i-\tau}^{iter}, \quad \cdots \quad \hat{y}_{i-(d_E-1)\tau}^{iter}\right)^T$ is treated as the input vector and the estimated mapping is $\hat{\mathbf{F}}\left(\mathbf{\Psi}_i^{iter}\right) = \left(\hat{f}\left(\hat{y}_i^{iter}\right), \quad \hat{f}\left(\hat{y}_{i-\tau}^{iter}\right), \quad \cdots \quad \hat{f}\left(\hat{y}_{i-(d_E-1)\tau}^{iter}\right)\right)^T$. If the reconstructed phase space in Eq. (5.5) and the reconstructed phase space in Eq. (5.7) have similar properties, then dynamic reconstruction is considered to have succeeded. Typically, the chaotic invariants produced from each sequence of iterated prediction

$\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ are compared with the chaotic invariants produced from $\left\{y_{i+1}\right\}_{i=1}^{N^{iter}}$, which is the observed data .

It is reasonable to enquire if the kRBF used for iterated prediction should be chosen using $\text{MSE}_{k_L}^{val}$. An example of a typical iterated sequence $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ produced by kRBF chosen using $\text{MSE}_{k_L}^{val}$ is given in Figure 5.11, and the corresponding delay embedding is in Figure 5.12.

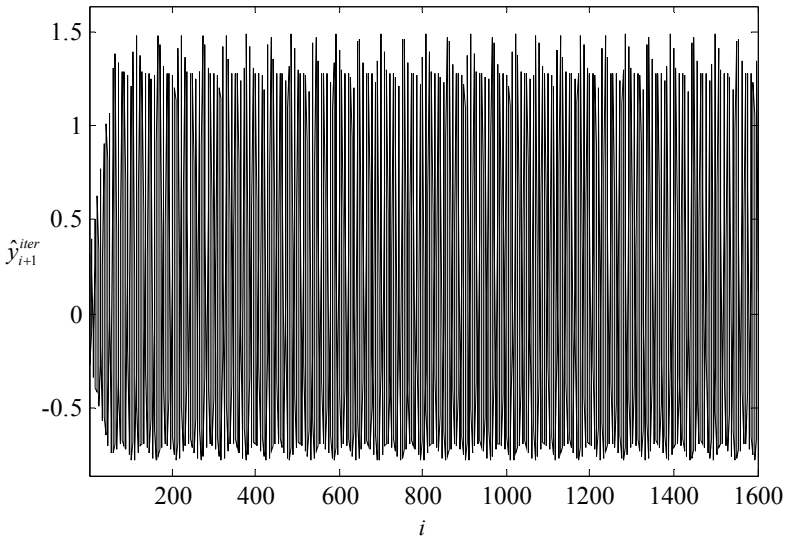

Figure 5.11 Iterated prediction on SFA using kRBF chosen by $\text{MSE}_{k_L}^{val}$.

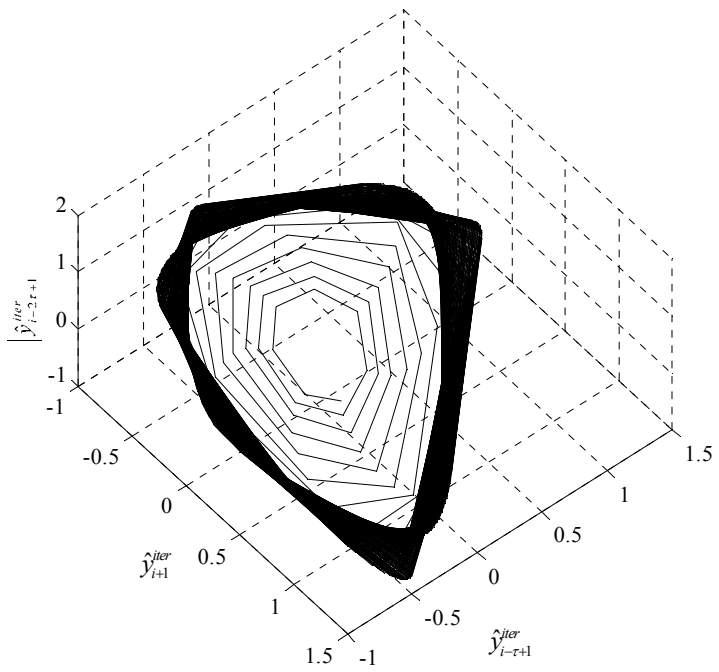

Figure 5.12 Delay embedding of $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ in Figure 5.11.

Observe that the time series in Figure 5.11 differs markedly from the time series in Figure 4.7, since bursts of steadily increasing amplitude, followed by small periods of quiescent behaviour are absent. The delay embedding in Figure 5.12 appears to capture the general shape of the attractor in Figure 4.8. However, the texture of the attractor is markedly different from that in Figure 5.13. Since $\text{MSE}_{k_L}^{val}$ appears to give qualitatively unsatisfactory results, $\text{s}^2(\mathbf{e}_{k_L}^{val})$ is used in the rest of this work.

5.4.1 Choice of Initialization

Actually, the data in the test set can be modelled as the sum of an ideal signal and observational noise: $y_i = \breve{y}_i + \eta_i$. Hence, each "ideal" embedding vector can be represented as:

$$\breve{\boldsymbol{\psi}}_i = \boldsymbol{\psi}_i - \boldsymbol{\eta}_{\psi_i}, \tag{5.8}$$

where $\boldsymbol{\eta}_{\psi_i}$ is the vector of observational noise associated with $\boldsymbol{\psi}_i$. This means that the full equation for iterated prediction is:

$$y_{i+1}^{iter} = \hat{f}\left(\breve{\boldsymbol{\psi}}_i + \boldsymbol{\eta}_{\psi_i} + \mathbf{e}_{\psi_i}\right). \tag{5.9}$$

Thus the actual perturbation of the input vector comes from both $\boldsymbol{\eta}_{\psi_i}$ and $\mathbf{e}_{\psi_i}$, and to minimize the effect of $\boldsymbol{\eta}_{\psi_i} + \mathbf{e}_{\psi_i}$, it is necessary to minimize the magnitude of $\eta_i + e_i$. If the observed time series has low SNR, this suggests that initializing the iterated prediction with estimated values (one-step prediction),

$$\boldsymbol{\psi}_i^{iter} = \left(\hat{y}_i, \quad \hat{y}_{i-1}, \quad \cdots \quad \hat{y}_{i-(p-1)}\right), \tag{5.10}$$

may sometimes be more effective than using values from the test set,

$$\boldsymbol{\psi}_i^{iter} = \left(y_i, \quad y_{i-1}, \quad \cdots \quad y_{i-(p-1)}\right). \tag{5.11}$$

To test the effect of using Eq. (5.10) versus Eq. (5.11), simulations were performed on SFA, based on the kRBF which was trained as in Table 5.5. The column "seed" designates the seeding method: "e" corresponds to Eq. (5.10), whilst "t" corresponds to Eq. (5.11). The column "d_E" indicates embedding dimension under which the kRBF is trained, i.e. 3 if $kRBF_3$ is used. The error criterion is $s^2(\mathbf{e}_{k_L}^{val})$ rather than $MSE_{k_L}^{val}$. Random starting points from the test set were chosen and the number of time steps of iterated prediction (N^{iter}) was 1600 samples, which was longer than the training set itself.

Table 5.16 Iterated prediction of SFA

d_E	seed	FIP	$\hat{d}_E$	SIP	D_2 (μ ± s)	D_{KY} (μ ± s)	HOP (μ ± s)	KE (μ ± s)	λ_1 (μ ± s)
3	e	99	3	20	1.22 ± 0.175	2.36 ± 0.182	34.3 ± 11.2	0.128 ± 0.0415	0.127 ± 0.0426
-	-	-	4	10	1.14 ± 0.259	3.15 ± 0.139	26.7 ± 2.68	0.177 ± 0.0371	0.148 ± 0.0156
3	t	106	3	21	1.24 ± 0.147	2.28 ± 0.126	35.9 ± 11.1	0.122 ± 0.0417	0.121 ± 0.0421
-	-	-	4	9	1.33 ± 0.304	3.14 ± 0.105	25.5 ± 7.86	0.203 ± 0.0585	0.163 ± 0.0375
4	e	0	3	18	1.62 ± 0.427	2.08 ± 0.0324	50.0 ± 13.5	0.0833 ± 0.0203	0.0833 ± 0.0203
-	-	-	4	12	1.67 ± 0.358	2.59 ± 0.125	52.5 ± 7.03	0.0826 ± 0.0143	0.0758 ± 0.0108
4	t	0	3	15	1.62 ± 0.355	2.07 ± 0.0299	48.4 ± 13.9	0.0861 ± 0.0198	0.0861 ± 0.0198
-	-	-	4	15	1.59 ± 0.439	2.59 ± 0.123	54.6 ± 9.42	0.0812 ± 0.0148	0.0738 ± 0.0133
5	e	0	3	30	1.95 ± 0.129	2.12 ± 0.0365	44.3 ± 8.38	0.0908 ± 0.0141	0.0908 ± 0.0141
5	t	0	3	30	1.94 ± 0.124	2.13 ± 0.0379	44.0 ± 6.19	0.0905 ± 0.0117	0.0905 ± 0.0117

The presence of local Lyapunov exponents means that different starting points may have very different behaviour. Thus, sequences $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ are generated until 30 successful sequences are found. SIP is the number of successful sequences, while FIP is the number of sequences which failed. A successful sequence is one with at least 1 positive Lyapunov exponent which can be numerically verified. An example of a successful sequence is given in Figure 5.13. Note that it captures the main qualitative features of Figure 4.7, *i.e.* bursts of steadily increasing amplitude, followed by small periods of quiescent behaviour.

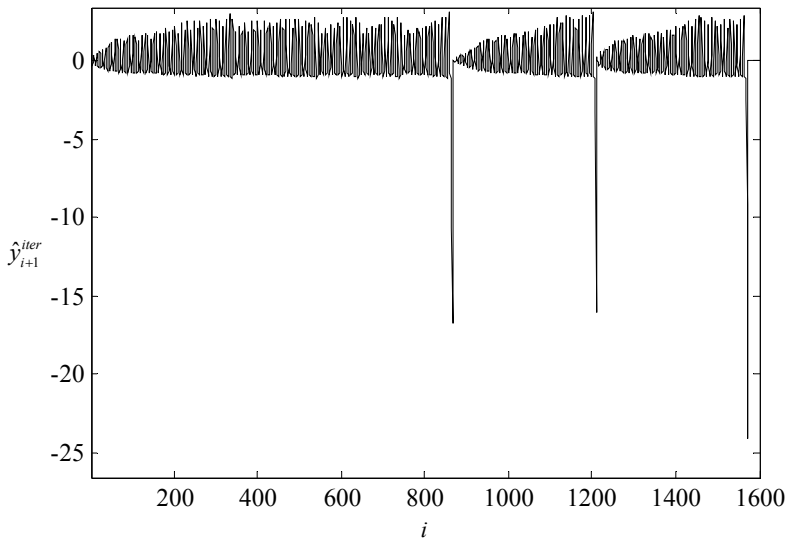

Figure 5.13 Example of successful iterated prediction for SFA.

In Table 5.16, the embedding dimension of each iterated prediction sequence $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ is $\hat{d}_E$, estimated using both GFNN [62] and Cao's method [57]. The sequences are sorted into various groups, according to their values of $\hat{d}_E$; (μ ± s) refers to the mean and sample standard deviation of the quantity under consideration. The quantities D_2, D_{KY}, HOP, KE and λ_1 are measured for each $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ in a group.

For example, D_2 is measured for each $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ corresponding to $\hat{d}_E = 3$, and the mean and sample standard deviation are recorded. Usually, the most positive Lyapunov exponent is more important, and so λ_1 is tabulated, instead of the entire Lyapunov spectrum.

The results in Table 5.16 suggest that the use of seeding method e reduced the number of failures for the case of $kRBF_3$ (106 reduced to 99). However, for $kRBF_4$ and $kRBF_5$, the benefits were marginal. This is because SFA is not noisy enough for the benefits of using seeding method e to become apparent.

Note that Figure 5.13 is distorted by the presence of spikes, causing the reconstructed attractor in Figure 5.14 to be more thinly spread out in phase space, resulting in underestimation of D_2, but not D_{KY}. Thus, the values of $\langle D_2 \rangle$ are significantly lower than the values of $\langle D_{KY} \rangle$ in Table 5.16.

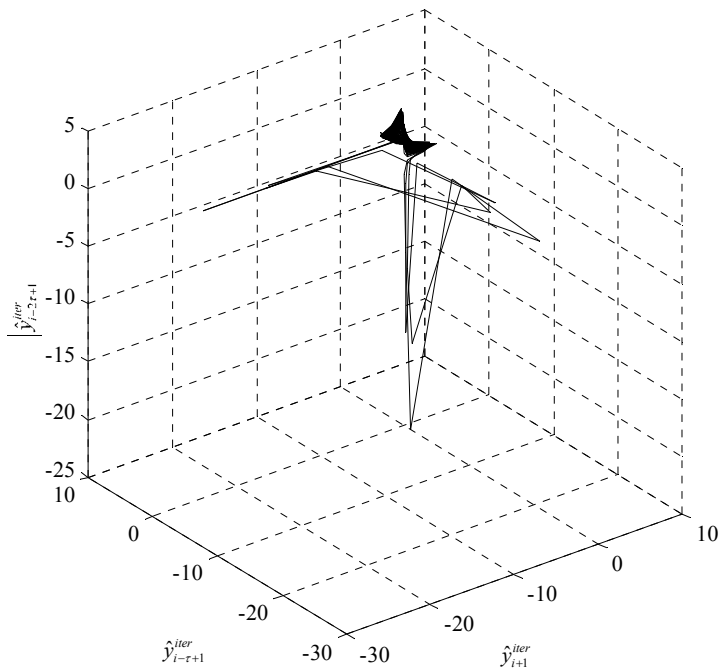

Figure 5.14 Delay embedding of $\left\{\hat{y}^{iter}_{i+1}\right\}_{i=1}^{N^{iter}}$ in Figure 5.13.

It is worthwhile to clarify the distinction between successful and failed sequences. As it stands, iterated prediction is rather problematic. Several examples of failed attempts are catalogued in Ref. [128]. They are: output becomes constant (see Figure 5.18), output becomes periodic, or output breaks down and output diverges and becomes unstable. This is actually equivalent to 2 possibilities: output converges to a lower dimensional attractor, or output becomes unstable. A fixed point is an attractor of dimension 1 whilst a limit cycle is an attractor of dimension 2.

Another cause of failure is numerical. When `lyap_spec.exe`, the program to estimate Lyapunov exponents fails due to numerical reasons, this is counted as a failure. Figure 5.15 illustrates an example of a failed reconstruction. Visually, it is not too different from Figure 5.13. However, one of the spikes is longer than any of the

spikes in Figure 5.15, and the quiescent period is also longer. The presence of sufficiently long spikes may have made the estimation of Lyapunov exponents difficult, and caused numerical difficulties. In any case, Figure 5.14 illustrates that spikes can cause distortions to the embedding, and hence, if the spikes are serious enough to interfere with the measurement of Lyapunov exponents, classifying this as failure is not unreasonable. Besides, a necessary condition for chaos is the presence of a positive Lyapunov exponent.

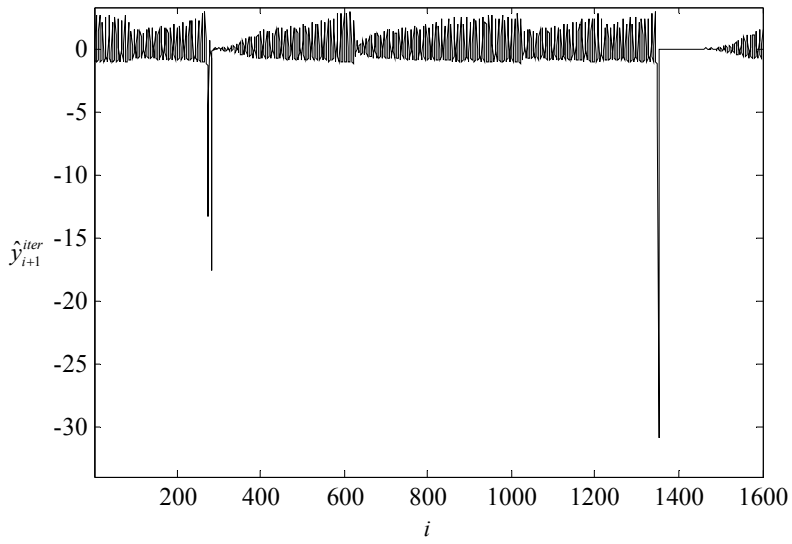

Figure 5.15 Example of failure for SFA.

5.4.2 Prior Information

Since $\boldsymbol{\psi}_i^{iter}$ is not the same as $\boldsymbol{\psi}_i$, supplying the feedback of the output to the input is actually equivalent to supplying dynamical noise:

$$\hat{y}_{i+1}^{iter} = \hat{f}\left(\boldsymbol{\psi}_i + \mathbf{e}_{\boldsymbol{\psi}_i}\right). \tag{5.12}$$

Here $\mathbf{e}_{\boldsymbol{\psi}_i}$ is the perturbation of the input vector $\boldsymbol{\psi}_i$ due to previous errors in the prediction output. A small amount of dynamical noise is sufficient to cause trouble for prediction [129]; this explains why iterated prediction is so difficult. As dynamical

noise could result in intermittency [130, 131], this also naturally explains why periodicity may suddenly appear (*i.e*. bifurcations).

A trick to improve iterated prediction is to utilize prior information. For example, the largest Lyapunov exponent can be used to improve the performance of iterated prediction [127]. However, a simple way to utilize prior information is to clip values of each $\hat{y}_{i+1}^{iter}$ to the maximum and minimum of the training set. This should not be confused with clipping of $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ after iterated prediction is performed. Rather, at each time step, a rule determines if the predicted point exceeds the maximum or the minimum of the training set. The value of the predicted point is then clipped. This means that the input to the next time step $\boldsymbol{\psi}_i^{iter}$ is corrected and thus the instability does not propagate throughout the iterated prediction. Otherwise, the phase space point could have escaped from the basin of attraction (see Glossary).

An example of the effect of clipping is illustrated in Figure 5.16. The range of values of $\hat{y}_{i+1}^{iter}$ is limited to $-1.17 \le \hat{y}_{i+1}^{iter} \le 3.99$, based on the maximum and minimum of the training set. The effect of clipping is most pronounced for i between 1000 to 1200, where a lot of points hover close to the lower limits but never go below it. The sequence $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ exhibits the property of bursts of steadily increasing amplitude, followed by small periods of quiescent behaviour, and is reasonably similar to Figure 4.7. In phase space, Figure 5.17 is reasonably similar to Figure 4.8, which suggests that clipping is very effective for SFA.

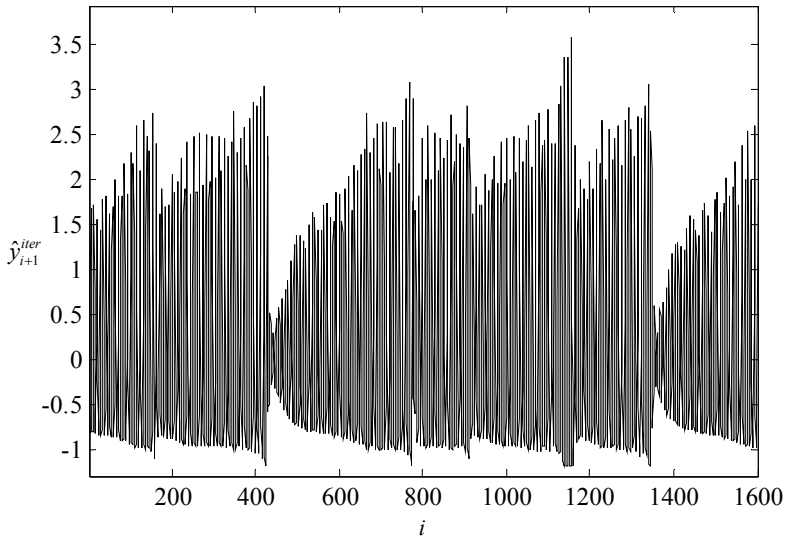

Figure 5.16 Illustration of the effect of clipping.

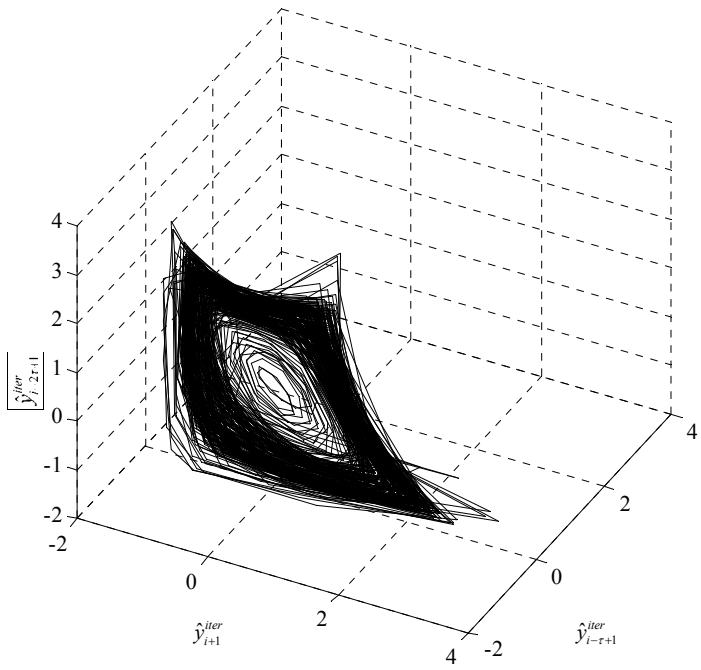

Figure 5.17 Delay embedding of $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ in Figure 5.16.

Experiments similar to those in Table 5.16 are repeated, with clipping and $N^{iter} = 1600$; results are recorded in Table 5.17. Excellent results are obtained when $kRBF_5$ is used, regardless of the seeding method; the mean values of D_2 and D_{KY} are remarkably

close to the theoretical values of 2.06. In fact, the results were comparable to state of the art neural network implementation by Patel and Haykin [128]. Furthermore, Ref. [128] showed only the outcome of a few selected starting points, whilst this work considered many arbitrary starting points.

Table 5.17 Iterated prediction of SFA (with clipping)

d_E	seed	FIP	$\hat{d}_E$	SIP	D_2 (μ ± s)	D_{KY} (μ ± s)	HOP (μ ± s)	KE (μ ± s)	λ_1 (μ ± s)
3	e	0	3	6	1.82 ± 0.103	2.22 ± 0.0343	28.2 ± 2.91	0.140 ± 0.0149	0.140 ± 0.0149
-	-	-	4	24	1.88 ± 0.106	3.08 ± 0.0403	20.7 ± 1.17	0.241 ± 0.0163	0.189 ± 0.0105
3	t	1	3	11	1.85 ± 0.0622	2.22 ± 0.0398	30.6 ± 4.40	0.131 ± 0.0191	0.130 ± 0.0180
-	-	-	4	19	1.89 ± 0.107	3.09 ± 0.0430	20.6 ± 1.88	0.247 ± 0.0212	0.192 ± 0.0174
4	e	0	3	29	1.85 ± 0.455	2.08 ± 0.117	32.0 ± 14.0	0.138 ± 0.0402	0.138 ± 0.0406
-	-	-	4	1	1.83 ± 0.00	2.67 ± 0.00	40.9 ± 0.00	0.113 ± 0.00	0.0955 ± 0.00
4	t	0	3	29	1.87 ± 0.388	2.08 ± 0.0908	31.0 ± 10.6	0.136 ± 0.0335	0.136 ± 0.0335
-	-	-	4	1	1.96 ± 0.00	2.50 ± 0.00	43.7 ± 0.00	0.0896 ± 0.00	0.0896 ± 0.00
5	e	0	3	30	2.09 ± 0.137	2.09 ± 0.0442	40.7 ± 5.29	0.0979 ± 0.0133	0.0979 ± 0.0133
5	t	0	3	30	2.05 ± 0.114	2.09 ± 0.0379	39.7 ± 4.13	0.0996 ± 0.0107	0.0996 ± 0.0107

5.4.3 Sea Clutter

The ideas in the previous sections are applied to the iterated prediction of in-phase component of sea clutter data in low sea state. N^{iter} = 1600 and $kRBF_5$ was used, based on d_E = 5 in Table 4.4; results are recorded in Table 5.18. The use of clipping resulted in 2673 failures and no SIP, and so the corresponding rows are not tabulated.

Table 5.18 Iterated prediction results for sea clutter in low sea state

seed	FIP	$\hat{d}_E$	SIP	D_2 (μ ± s)	D_{KY} (μ ± s)	HOP (μ ± s)	KE (μ ± s)	λ_1 (μ ± s)
e	2663	3	10	2.69 ± 0.143	1.40 ± 0.343	1470 ± 1050	0.00501 ± 0.00417	0.00501 ± 0.00417
t	2666	3	7	2.62 ± 0.103	1.73 ± 0.352	776 ± 570	0.00759 ± 0.00481	0.00759 ± 0.00481

The entire test set was used, and only a handful of successful reconstructions were found. The failures were all due to convergence to a fixed point; see Figure 5.18 for

an example. The use of seeding method e resulted in slightly more SIP than seeding method t.

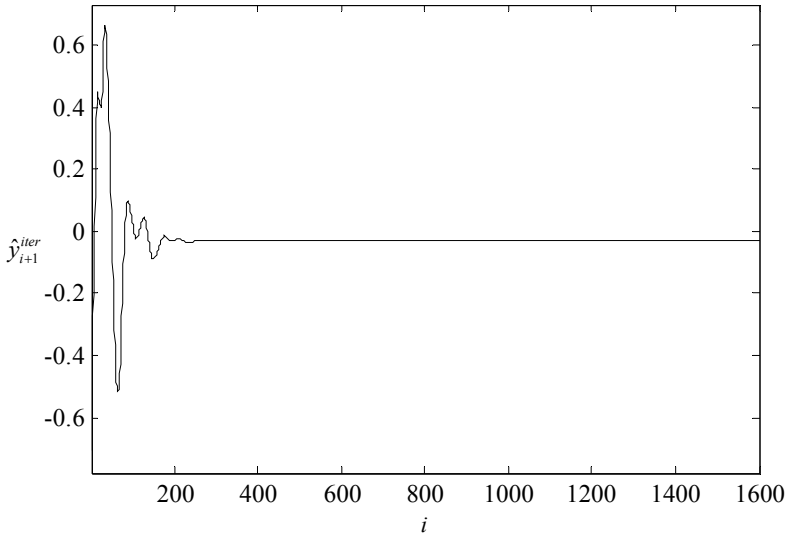

Figure 5.18 Example of convergence onto fixed point; an example of failed dynamic reconstruction of in-phase component of sea clutter in low sea state.

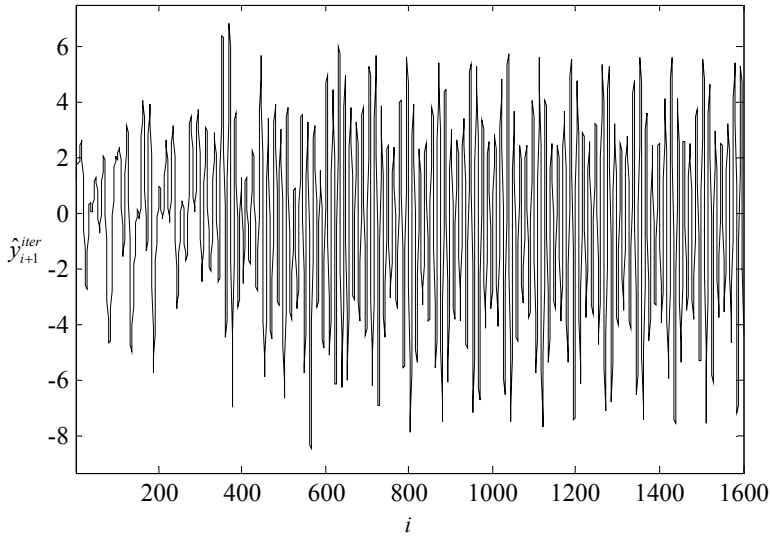

Figure 5.19 Iterated prediction of in-phase component of sea clutter in low sea state.

The estimated embedding dimension $\hat{d}_E$ of each $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ is 3, but the largest Lyapunov exponent of each $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ is so close to 0, that the chaotic nature of each

sequence is in doubt. Besides, the regularity of $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=801}^{1600}$ in Figure 5.19 is highly suspicious. Thus, it is imperative to examine the delay embedding of $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{N^{iter}}$ in Figure 5.21 and compare it with the delay embedding of the original time series of in-phase component of sea clutter in low sea state. Incidentally, Figure 5.20 is a projection of the actual delay embedding from dimension 5 onto dimension 3. Nonetheless, it is possible to extract useful information by regarding this as a kind of phase portrait. It appears that the outer part of the attractor is shaped like a torus, with an interior region (possibly higher dimensional) which is densely packed, as if it is noise (noise is infinite dimensional and appears ellipsoidal when embedded). It is possible that the toroidal portion could be due to the presence of 2 dominant frequencies, one fast and one slow. Peculiarly, part of the attractor appears squarish.

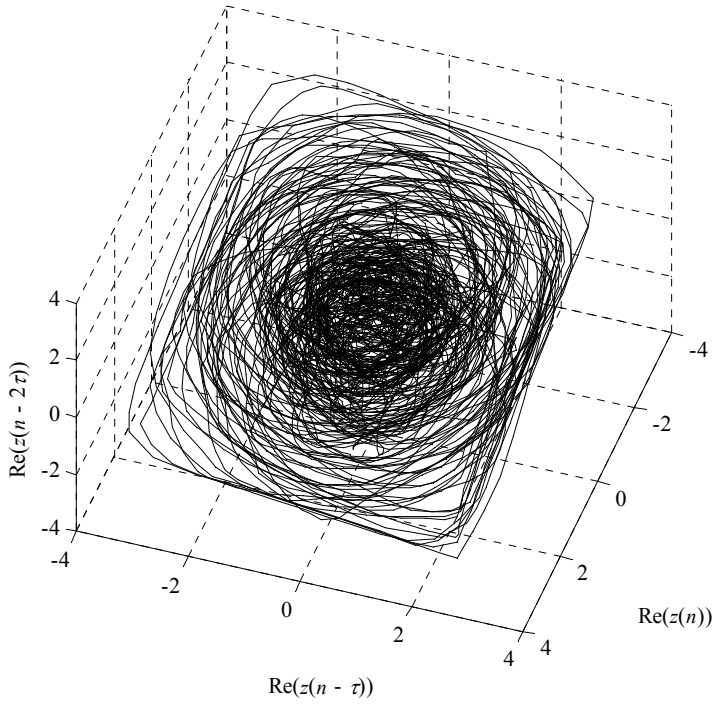

Figure 5.20 Delay embedding of in-phase component of sea clutter data (low sea state) in only 3 dimensions.

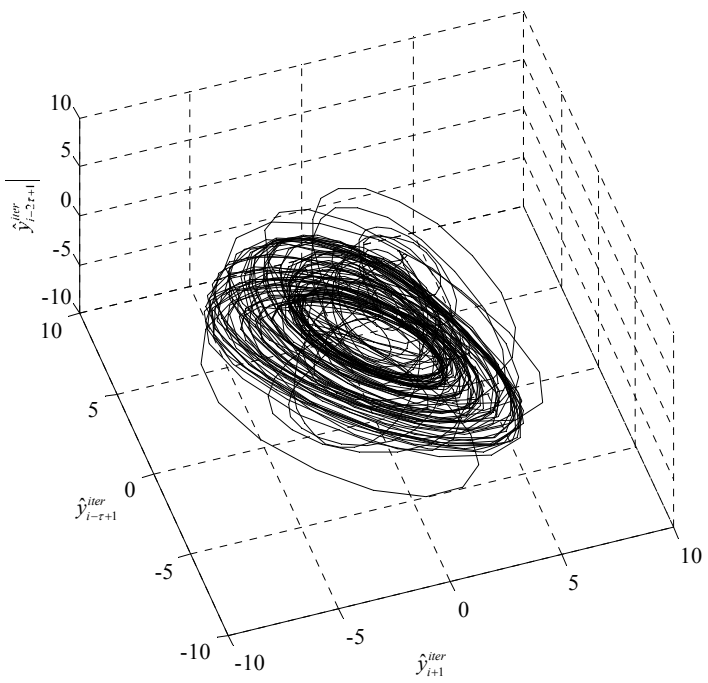

Figure 5.21 Delay embedding of $\left\{\hat{y}^{iter}_{i+1}\right\}_{i=1}^{1600}$ from time series in Figure 5.19.

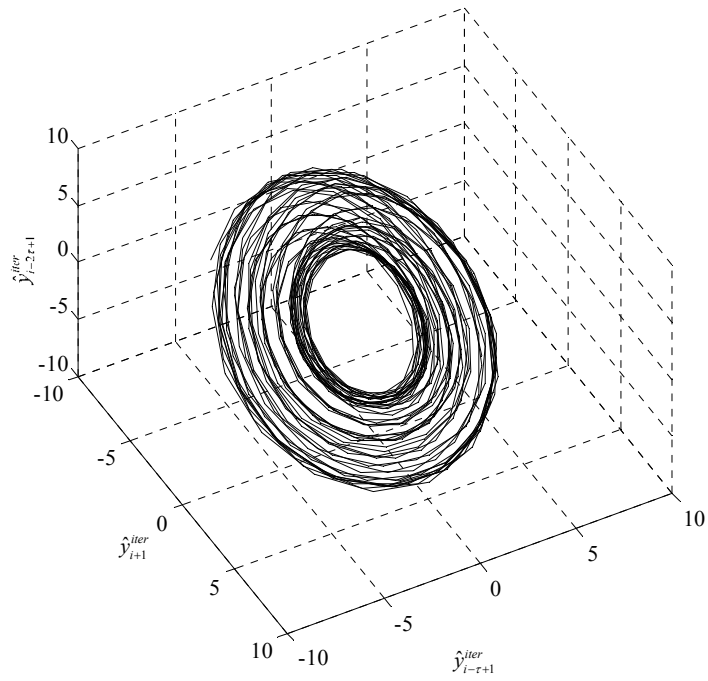

Figure 5.22 Delay embedding of $\left\{\hat{y}^{iter}_{i+1}\right\}_{i=801}^{1600}$ from time series in Figure 5.19.

Figure 5.22 is the delay embedding of the latter half of the time series in Figure 5.19, and clearly shows convergence onto the surface of a torus. This confirms that quasiperiodicity occurs in Figure 5.19. The values of λ_1 in Table 5.18 should be 0 for a torus; the tiny positive values could be due to numerical inaccuracy. Studying all the Lyapunov exponents in Table 5.19 (listed separately due to space constraints, it can be concluded that λ_1 and λ_2 are close to 0, which is consistent with the existence of a torus; again, the tiny negative values could be due to numerical inaccuracy.

Table 5.19 Lyapunov exponents from $\left\{\hat{y}_{i+1}^{iter}\right\}_{i=1}^{1600}$ for sea clutter in low sea state

seed	$\hat{d}_E$	SIP	λ_1 ($\mu \pm s$)	λ_2 ($\mu \pm s$)	λ_3 ($\mu \pm s$)
e	3	10	5.01E-03 ± 4.17E-03	-1.26E-02 ± 2.85E-03	-1.89E-01 ± 2.44E-02
t	3	7	7.59E-03 ± 4.81E-03	-8.19E-03 ± 4.13E-03	-1.85E-01 ± 2.27E-02

Apparently, iterated prediction has failed in the sense that most sequences converge onto a fixed point, and for those that do not, converge onto a torus which should have no positive Lyapunov exponent. On the other hand, it seems that Figure 5.20 could be crudely approximated as the union of a torus and a dense sphere. Thus, convergence onto a torus may show that the kRBF had successfully modelled part of the state space. If it is crudely assumed that union can be approximated by addition, *i.e.* the union of 2 attractors in state space can be approximated by the addition of 2 attractors in state space, in turn equivalent to the addition of 2 signals in time domain. In such a case, the dense spherical portion of the attractor could be regarded as white noise, which is impossible to model. Alternatively, it could be assumed that below a certain magnitude in state space, the system is so dominated by noise, that it is effectively producing only white noise, whereas above that magnitude, the system is quasiperiodic (converges onto a torus).

Iterated prediction was also performed on in-phase component of sea clutter data in high sea state. N^{iter} = 1600 and kRBF was used. Results are recorded in Table 5.20. The use of clipping resulted in completely identical results, and so it is not tabulated. Despite varying d_E, and the seeding method, no successful sequences were found; all sequences failed by converging onto a fixed point (see Figure 5.23).

Table 5.20 Chaotic invariants of time Series in Figure 5.21

d_E	seed	failures	SIP
4	e	2721	0
4	t	2721	0
5	e	2718	0
5	t	2718	0
6	e	2715	0
6	t	2715	0

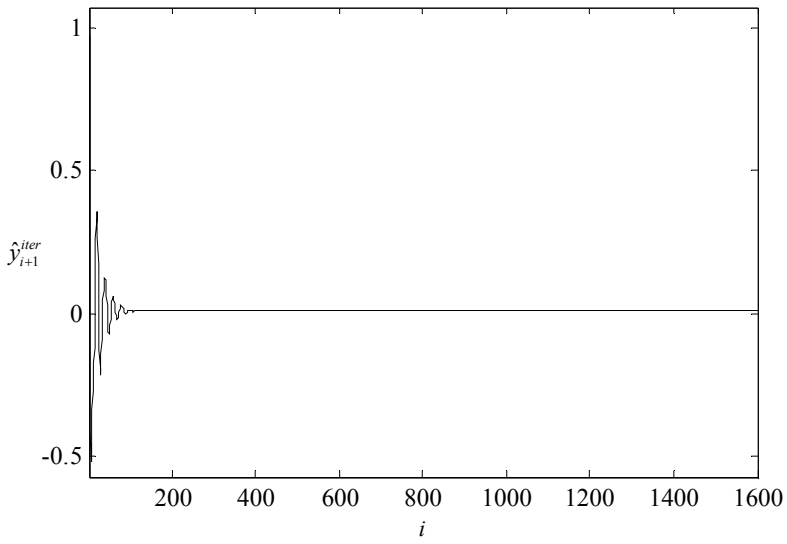

Figure 5.23 An example of failed dynamic reconstruction of in-phase component of sea clutter in high sea state.

One pertinent question is whether 400 centers (Table 5.14) is sufficient to learn an attractor of dimension 5 or 6, due to the curse of dimensionality. However, Table 5.14

showed that the generalization error is low, even lower than for SFA, which is a very much smaller system, so it seems that the kRBF had been successfully trained.

On the other hand, GE is an order of magnitude higher for sea clutter in high sea state, but still reasonably low (Table 5.15). Thus, it seems possible to model sea clutter in high sea state. However, the number of centers used is either 50 or 100, depending on d_E. This suggests that sea clutter data in high sea state may be so noisy that fewer centers are necessary, as with low SNR Lorenz data (see Table 5.13).

Consider the projection of the actual delay embedding of sea clutter in high sea state from dimension 5 onto dimension 3. Due to the impossibility of visualizing 5 dimensional embeddings, it is necessary to make do with 3 dimensions in Figure 5.24. It seems that there are no prominent features in state space, and the embedding resembles one which would be obtained with correlated noise (infinite dimensional elllipsoid). Thus, the case of sea clutter in high sea state is relatively ambiguous, compared to sea clutter in low sea state, whereby some structure can still be discerned in 3 dimensions. Thus, it seems difficult to model sea clutter in high sea state using state space information. In fact, Ref. [38] argues that sea clutter should be modelled with AR processes.

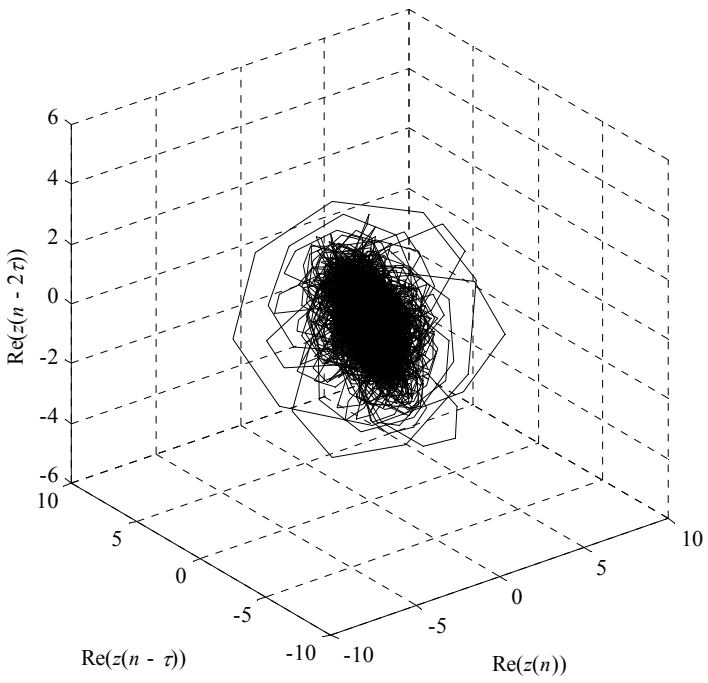

Figure 5.24 Delay embedding of in-phase component of sea clutter data (high sea state) in only 3 dimensions.

5.5 Contributions of this Chapter

- The problem of finding the optimal hyperparameters from cross-validation is reformulated into a voting scheme to deal with the problem of erraticity (Section 5.1.2).
- Error variance is used, instead of mean squared error, because bias does not affect the chaotic invariants of the estimated sequence (Section 5.2). It is demonstrated that sequences generated by kRBF models selected using error variance (Figure 5.17) can result in better dynamic reconstructions than kRBF models selected using mean squared error (Figure 5.12).
- Besides speeding up cross-validation, caching also deals with the problem of erraticity. Hence, it is possible to compare different algorithms, without fear that

clustering results may affect the outcome. Caching also reduces the computational complexity, when various algorithms are combined in a committee machine (Section 5.3.1).

- Iterated prediction is performed using many different unique candidate starting points; this guards against possible aberrations caused by local Lyapunov exponents. Initializing iterated prediction with estimated values instead of the test set is suggested (Section 5.4.1).
- Clipping is introduced, as a simple, yet effective way to stabilize iterated predictions (Section 5.4.2).
- Examples of iterated prediction converging onto torus are documented (Section 5.4.3).

CHAPTER 6

Conclusions and Future Work

This thesis is concerned with the problem of using RBF and variants to perform dynamic reconstruction of sea clutter data. In Chapters 2-5, many ideas were discussed and tested. In this chapter, a short summary is given, together with pointers some for future work.

6.1 Conclusions

It is clearly seen that the proposed method of speeding up cross validation results in significant savings in computational load, compared to the ordinary method of performing cross validation on regularized RBF networks. Since this trick requires the recognition that only one equation needs to be computed for the innermost loop, it may not be possible to speed up cross validation for MLPs, unless drastic approximations are performed. Since cross validation is necessary to prevent overfitting, and most neural network applications currently employ MLPs, this could encourage many more applications to employ the RBF or its variants.

It is observed that the use of data driven basis functions sometimes resulted in fewer centers and lower levels of regularization. In fact, no regularization is required for some of the variants at times. It may be possible that the Minimum Description Length [132] of the data driven basis functions are effectively the same as that of the ordinary RBF, for the same number of centers, because the calculation of the covariance matrix is data driven, and not derived by tuning external hyperparameters. Thus, data driven basis functions (using few centers) could be useful for crude

approximations. The problem is that for any given basis function, it is difficult to gauge how many centers it would require, and how the error performance would be like. It is not possible to have a simple rule describing the relative performance of the various algorithms, because there is no clear discernible trend of superiority of one model across all data sets and choices of embedding dimensions. Nonetheless, the kRBF appears to give a good compromise between the number of centers required, and performance as measured by NEV and NMSE. Note that the number of centers required is a crude measure of the complexity of the attractor in phase space, but does not necessarily reflect the underlying complexity of the system, since SFA and Lorenz data require significantly different number of centers to model.

Disappointingly, iterated prediction was remarkably unsuccessful for sea clutter, compared to the results obtainable with SFA; most sequences converged onto a fixed point. One possibility is that the RBF failed to learn the dynamics, because the training set was too small. Nonetheless, it could not be too big to ensure that simulation time is manageable, and also to avoid the danger of oscillations [82]. The other possibility is that sea clutter is intrinsically a 3 or 4 dimensional process, with the extra dimensionality possibly due to the presence of noise. After all, Figure 5.20 shows that part of the attractor is shaped like a torus.

Consider a dynamical system consisting of c independent oscillators, each with a limit cycle. If the fundamental periods of coupled oscillators are irrationally related, then the attractor of the combined system is the c-dimensional surface of a torus [133]. Thus, the 2-torus in Figure 5.22 seems to suggest that gravity waves and capillary waves may be produced by 2 coupled oscillators. However, since a dynamical system

in the ocean should be dissipative (see Glossary), the torus is not satisfactory as a model since it is a Hamiltonian system (see Glossary).

The Poincaré section of a c-dimensional torus is a torus of dimension c-1 [133]. Consider a system of c independent oscillators; if $c \geq 3$, a weak nonlinear coupling may produce a "turbulent" behaviour [134] (note that this is not true for $c \leq 2$). This suggests that an oceanic system comprising weakly coupled capillary waves, gravity waves and Rossby waves (waves with very large wave lengths) could be chaotic. In fact, Ref [33, 34] mention chaotic models of atmospheric systems which include Rossby waves and are incidentally of dimension 5.

In Ref. [13], the largest Lyapunov exponent was correlated with wave height, whilst wind was stated to be the most important environmental factor influencing sea clutter [1]. Perhaps the effect of wind could be modelled as a stochastic perturbation which increases the intrinsic dimensionality of the underlying dynamics. However, it should be noted that the effect of wind is contrary to the assumption of stationarity, *i.e.* transients can be ignored for chaotic systems [45].

It also has to be acknowledged that the presence of local Lyapunov exponents may cause two neural network models to have different performance with respect to iterated prediction. This is because a "successful" sequence generated by iterated prediction requires that the neural network models the phase space closely, and also requires the local positive Lyapunov exponents to be low, so that the iterated prediction sequence will not diverge too much. Perhaps this is why existing literature

on iterated prediction, such as [128], only use a few selected starting points for iterated prediction.

Another problem is that the topology of the attractor could be too complicated, with many basins of attraction, and the neural network had only managed to learn only part of the phase space corresponding to one basin of attraction. Since fractal dimension is a global property, it is unable to distinguish between a network which had successfully learnt an entire attractor, and between a neural network which had failed to learn part of the attractor. Thus, it is necessary to inspect the reconstructed attractor visually. However, this is a course of action which is unavailable to high dimensional attractors.

It should also be noted that the whole process of dynamical reconstruction is itself fraught with limitations. Consider that as the modelling error approaches 0, the error in the estimated chaotic invariants should also approach 0. On the other hand, the converse is not necessarily true, *i.e.* a good match of the chaotic invariants of the system and the chaotic invariants of the reconstructed system is a necessary, but not sufficient, criterion for a good reconstruction. Hence, there is no guarantee that a good match between the chaotic invariants of the original system and the chaotic invariants of the reconstructed system implies that the reconstruction is successful. If dynamical reconstruction is successful, it may be useful for helping to decide if the original process is chaotic or stochastic. On the other hand, if the reconstruction is unsuccessful, it is difficult to draw any conclusion.

It may also be necessary to consider the fact that there are alternative routes to turbulence besides chaos [135]. There is the possibility that the underlying process behind the generation of sea clutter could be non-chaotic. It may very well be that under many conditions, such as high sea state, sea clutter data could be better modelled using stochastic processes, as in Ref. [38].

6.2 Future Work and Recommendations

- The techniques discussed in this thesis should be applied to sea clutter data obtained under different wind and wave conditions, from different places.
- Thus far, this work has only dealt with stationary data. In order to cope with non-stationarity, Kalman filters or Extended Kalman Filters (EKFs) should be incorporated. Indeed, this was already done in Ref. [40]. A combination of ideas would perhaps represent the state of the art in RBF networks and variants, in future.
- Brizzotti and Carvalho [84] had compared the effect of different clustering algorithms on RBF generalization in the context of pattern classification. It was claimed that some relatives of the *k*-means algorithm had improved performance. It may be interesting to use some of the clustering techniques employed in Ref. [84] and examine their performance in the context of regression.
- In addition, nonlinear signal processing methods for nonstationary data should be considered, such as Ref. [136, 137].
- The IPIX radar resolution is about 3cm, but capillary waves are of the order of 2 cm or less [11]. In fact, Skolnik [1] had stated that the assumption that sea surface displacements are small compared to radar wavelength is usually not satisfied. It

seems pertinent to ask if capillary waves are relevant to clutter of X-band radar. Thus it may be advisable to perform experiments with radar clutter of varying wavelengths to verify that the composite surface model is valid.

- It is clear that similar concepts can be utilized for clutter analysis for Light Detection and Ranging (LIDAR) systems. So far, the K-distribution appears to fit the data well. On the other hand, turbulence may imply chaos. It may be possible to use the same techniques on LIDAR clutter.

REFERENCES

[1] M. I. Skolnik, Introduction to radar systems, 3rd ed. Boston: McGraw Hill, 2001.

[2] S. Haykin, "Chaotic characterization of sea clutter: new experimental results and novel applications," presented at Conference Record of the Twenty-Ninth Asilomar Conference on Signals, Systems and Computers, pp. 1076-1080 vol.2, 1995.

[3] L. B. Wetzel, "Sea Clutter," in Radar handbook, M. I. Skolnik, Ed., 2nd ed. New York: McGraw-Hill, 1990, pp. 13.1-13.40.

[4] P. Z. Peebles, Radar principles. New York: Wiley, 1998.

[5] P. R. Tapster, A. R. Weeks, P. N. Pusey, and E. Jakeman, "Analysis of probability-density functions for laser scintillations in a turbulent atmosphere," Journal of the Optical Society of America A (Optics and Image Science), vol. 6, pp. 782-785, 1989.

[6] E. Jakeman, "K-distributed noise," Journal of Optics A: Pure and Applied Optics, vol. 1, pp. 781-789, 1999.

[7] H. Leung and S. Haykin, "Is there a radar clutter attractor?," Applied Physics Letters, vol. 56, pp. 593-595, 1990.

[8] S. Haykin and H. Leung, "Model reconstruction of chaotic dynamics: first preliminary radar results," presented at IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-92, pp. 125-128 vol.4, 1992.

[9] N. He and S. Haykin, "Chaotic modelling of sea clutter," Electronics Letters, vol. 28, pp. 2076-2077, 1992.

[10] S. Haykin and S. Puthusserypady, "Chaotic dynamics of sea clutter: an experimental study," Radar 97 (Conf. Publ. No. 449), pp. 75-79, 1997.

[11] S. Haykin and S. Puthusserypady, "Chaotic dynamics of sea clutter," Chaos, vol. 7, pp. 777-802, 1997.

[12] S. Haykin, "Radar clutter attractor: implications for physics, signal processing and control," IEE Proceedings - Radar, Sonar and Navigation, vol. 146, pp. 177-188, 1999.

[13] S. Haykin and S. Puthusserypady, Chaotic dynamics of sea clutter. New York: John Wiley, 1999.

[14] H. Leung and T. Lo, "Chaotic radar signal processing over the sea," IEEE Journal of Oceanic Engineering, vol. 18, pp. 287-295, 1993.

[15] H. Leung and S. Haykin, "Neural network modeling of radar backscatter from an ocean surface using chaos theory," Proceedings of the SPIE - The International Society for Optical Engineering, vol. 1565, pp. 279-286, 1991.

[16] S. Haykin and H. Leung, "Neural network modeling of radar backscatter from an ocean surface using chaos theory," IEEE Conference on Neural Networks for Ocean Engineering, pp. 215-222, 1991.

[17] B. X. Li and S. Haykin, "Chaotic detection of small target in sea clutter," presented at IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-93, pp. 237-240 vol.1, 1993.

[18] H. Leung, "Applying chaos to radar detection in an ocean environment: an experimental study," IEEE Journal of Oceanic Engineering, vol. 20, pp. 56-64, 1995.

[19] S. Haykin and S. Puthusserypady, "Chaos, sea clutter, and neural networks," presented at Conference Record of the Thirty-First Asilomar Conference on Signals, Systems & Computers, pp. 1224-1227 vol.2, 1997.

[20] H. Leung and X. Huang, "Parameter estimation in chaotic noise," IEEE Transactions on Signal Processing, vol. 44, pp. 2456-2463, 1996.

[21] H. Leung, G. Hennessey, and A. Drosopoulos, "Target detection in an oceanic environment using spatial temporal chaos," presented at IEEE International Conference on Systems, Man, and Cybernetics, 1997. 'Computational Cybernetics and Simulation'., 1997, pp. 3517-3521 vol.4, 1997.

[22] S. Haykin, S. Puthusserypady, and P. Yee, "Dynamic reconstruction of sea clutter using regularized RBF networks," presented at Conference Record of the Thirty-Second Asilomar Conference on Signals, Systems & Computers, pp. 19-23 vol.1, 1998.

[23] S. Haykin, Kalman filtering and neural networks. New York: Wiley, 2001.

[24] G. Hennessey, H. Leung, and A. Y. Drosopoulos, P.C., "Sea-clutter modeling using a radial-basis-function neural network," IEEE Journal of Oceanic Engineering, vol. 26, pp. 358-372, 2001.

[25] H. Leung, N. Dubash, and N. Xie, "Detection of small objects in clutter using a GA-RBF neural network," Aerospace and Electronic Systems, IEEE Transactions on, vol. 38, pp. 98-118, 2002.

[26] A. R. Osborne, M. Serio, L. Bergamasco, and L. Cavaleri, "Are Ocean Surface Waves Chaotic?," presented at Proceedings of the 2nd Experimental Chaos Conference, Arlington, Virginia, pp. 356-362, 1993.

[27] T. W. Frison and H. D. I. Abarbanel, "Ocean gravity waves: A nonlinear analysis of observations," Journal Of Geophysical Research, vol. 102, pp. 1051-1060, 1997.

[28] A. N. Churyumov and Y. A. Kravtsov, "Microwave backscatter from mesoscale breaking waves on the sea surface," Waves in Random Media, vol. 10, p. 1, 2000.

[29] A. T. Jessup, W. K. Melville, and W. C. Keller, "Breaking waves affecting microwave backscatter. 2. Dependence on wind and wave conditions," Journal Of Geophysical Research, vol. 96, pp. 20561-9, 1991.

[30] C. Jung, T. Tel, and E. Ziemniak, "Application of scattering chaos to particle transport in a hydrodynamical flow," Chaos, vol. 3, pp. 555-568, 1993.

[31] J. C. Sommerer, K. Hwar-Ching, and H. E. Gilreath, "Experimental evidence for chaotic scattering in a fluid wake," Physical Review Letters, vol. 77, pp. 5055-5058, 1996.

[32] J. C. Sommerer, E. Ott, and T. Tel, "Modeling two-dimensional fluid flows with chaos theory," Johns Hopkins APL Technical Digest, vol. 18, pp. 193-203, 1997.

[33] P. Birtea, M. Puta, T. S. Ratiu, and R. Tudoran, "A short proof of chaos in an atmospheric system," Physics Letters A, vol. 300, pp. 189-191, 2002.

[34] E. N. Lorenz, "On the existence of a slow manifold," Journal of the Atmospheric Sciences, vol. 43, pp. 1547-1557, 1986.

[35] H. D. I. Abarbanel, T. W. Frison, and L. S. Tsimring, "Obtaining order in a world of chaos [signal processing]," IEEE Signal Processing Magazine, vol. 15, pp. 49-65, 1998.

[36] C. P. Unsworth, M. R. Cowper, S. McLaughlin, and B. Mulgrew, "False detection of chaotic behaviour in the stochastic compound k-distribution model of radar sea clutter," presented at Proceedings of the Tenth IEEE Workshop on Statistical Signal and Array Processing, 2000., pp. 296-300, 2000.

[37] C. P. Unsworth, M. R. Cowper, S. McLaughlin, and B. Mulgrew, "Re-examining the nature of radar sea clutter," in IEE Proceedings - Radar, Sonar and Navigation, vol. 149, 2002, pp. 105-114.

[38] S. Haykin, R. Bakker, and B. W. Currie, "Uncovering nonlinear dynamics-the case study of sea clutter," Proceedings of the IEEE, vol. 90, pp. 860-881, 2002.

[39] J. Gao and K. Yao, "Multifractal features of sea clutter," presented at Proceedings of the IEEE Radar Conference, 2002., pp. 500-505, 2002.

[40] S. Roweis and Z. Ghahramani, "An EM algorithm for identification of nonlinear dynamical systems," in Kalman filtering and neural networks, S. S. Haykin, Ed. New York: Wiley, 2001, pp. xiii, 284.

[41] T. Kapitaniak and S. R. Bishop, The illustrated dictionary of nonlinear dynamics and chaos. New York: John Wiley, 1998.

[42] B. D. Ripley, Pattern recognition and neural networks. Cambridge, New York: Cambridge University Press, 1996.

[43] H. D. I. Abarbanel, Analysis of Observed Chaotic Data. New York Berlin Heidelberg: Springer-Verlag, 1996.

[44] E. N. Lorenz, "Deterministic Nonperiodic Flow," Journal of the Atmospheric Sciences, vol. 20, pp. 130-148, 1963.

[45] J. P. Eckmann and D. Ruelle, "Ergodic Theory of Chaos and Strange Attractors," Reviews of Modern Physics, vol. 57, pp. 617-656, 1985.

[46] J. Stark, Personal Communication, 19 May 2004.

[47] J. P. Huke, Personal Communication, 20 May 2004.

[48] D. S. Broomhead and G. P. King, "On the qualitative analysis of experimental dynamical systems," in Nonlinear phenomena and chaos, S. Sarkar, Ed. Bristol; Boston: Adam Hilger Ltd, 1986, pp. 113-144.

[49] A. M. Fraser and H. L. Swinney, "Independent coordinates for strange attractors from mutual information," Physical Review A, vol. 33, pp. 1134-1140, 1986.

[50] T. M. Cover and J. A. Thomas, Elements of information theory. New York: Wiley, 1991.

[51] A. Papoulis, Probability, random variables, and stochastic processes, 3rd ed. Singapore: McGraw-Hill, 1991.

[52] R. Hegger, H. Kantz, and T. Schreiber, "Practical implementation of nonlinear time series methods: The TISEAN package," Chaos, vol. 9, pp. 413-435, 1999.

[53] R. E. Kalman, "Nonlinear aspects of sampled-data control systems," presented at Proceedings of the Symposium on Nonlinear Circuit Analysis VI, 1956.

[54] T. Kohda and H. Fujisaki, "Kalman's recognition of chaotic dynamics in designing Markov information sources," IEICE Transactions A: on Fundamentals of Electronics, Communications and Computer Sciences, vol. E82A, pp. 1747-1753, 1999.

[55] K. Ikeda, "Multiple-valued stationary state and its instability of the transmitted light by a ring cavity system," Optics Communications, vol. 30, pp. 257-261, 1979.

[56] B. W. Silverman, Density estimation for statistics and data analysis. London ; New York: Chapman and Hall, 1986.

[57] L. Cao, "Practical method for determining the minimum embedding dimension of a scalar time series," Physica D: Nonlinear Phenomena, vol. 110, pp. 43-50, 1997.

[58] T. Sauer, J. A. Yorke, and M. Casdagli, "Embedology," Journal of Statistical Physics, vol. 65, pp. 579-616, 1991.

[59] M. Ding, C. Grebogi, E. Ott, T. Sauer, and J. A. Yorke, "Estimating correlation dimension from a chaotic time series: when does plateau onset occur?," Physica D: Nonlinear Phenomena, vol. 69, pp. 404-424, 1993.

[60] M. B. Kennel, R. Brown, and H. D. I. Abarbanel, "Determining embedding dimension for phase-space reconstruction using a geometrical construction," Physical Review A, vol. 45, pp. 3403-3411, 1992.

[61] D. R. Fredkin and J. A. Rice, "Method of false nearest neighbors: a cautionary note," Physical Review E, vol. 51, pp. 2950-2954, 1995.

[62] R. Hegger and H. Kantz, "Improved false nearest neighbor method to detect determinism in time series data," Physical Review E, vol. 60, pp. 4970-4973, 1999.

[63] C. C. Aggarwal, A. Hinneburg, and D. A. Keim, "On the surprising behavior of distance metrics in high dimensional space," Database Theory - ICDT 2001. 8th International Conference. Proceedings (Lecture Notes in Computer Science Vol.1973), pp. 420-434, 2001.

[64] C. Merkwirth, U. Parlitz, and W. Lauterborn, "TSTOOL - A software package for nonlinear time series analysis," presented at Proceedings of the International Workshop on Advanced Black-Box Techniques for Nonlinear Modeling, Katholieke Universiteit, Leuven, Belgium, pp. 144-146, 1998.

[65] R. C. Hilborn, Chaos and nonlinear dynamics : an introduction for scientists and engineers, 2nd ed. Oxford, New York: Oxford University Press, 2000.

[66] P. Grassberger and I. Procaccia, "Characterisation of strange attractors," Physical Review Letters, vol. 50, pp. 346-349, 1983.

[67] D. Ruelle, "Deterministic chaos: the science and the fiction," Proceedings of the Royal Society of London, Series A, vol. 427, pp. 241-248, 1990.

[68] M. A. H. Nerenberg and C. Essex, "Correlation dimension and systematic geometric effects," Physical Review A (Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics), vol. 42, pp. 7065-7074, 1990.

[69] C. Essex and M. A. H. Nerenberg, "Comments on 'Deterministic chaos: The science and the fiction' by D. Ruelle," Proceedings of the Royal Society of London, Series A, vol. 435, pp. 287-292, 1991.

[70] K. T. Alligood, T. Sauer, and J. A. Yorke, Chaos : an introduction to dynamical systems. New York: Springer, 1997.

[71] E. A. Jackson, Perspectives of nonlinear dynamics. Cambridge Cambridgeshire, New York: Cambridge University Press, 1989.

[72] V. I. Oseledec, "A multiplicative ergodic theorem. Characteristic Lyapunov exponents of dynamical systems," Trudy Mosk. Obsch., vol. 19, pp. 179-210, 1968.

[73] H. D. I. Abarbanel, R. Brown, and M. B. Kennel, "Variation of Lyapunov Exponents on a Strange Attractor," Journal of Nonlinear Science, vol. 1, pp. 175-199, 1991.

[74] H. D. I. Abarbanel, R. Brown, and M. B. Kennel, "Local Lyapunov Exponents from Observed Data," Journal of Nonlinear Science, vol. 2, pp. 343-365, 1992.

[75] H. Haken, "At least one Lyapunov exponent vanishes if the trajectory of an attractor does not contain a fixed point," Physics Letters A, vol. 94, pp. 71-72, 1983.

[76] D. A. Russell, J. D. Hanson, and E. Ott, "Dimension of strange attractors," Physical Review Letters, vol. 45, pp. 1175-1178, 1980.

[77] J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function networks," Neural Computation, vol. 3, pp. 246-257, 1991.

[78] J. D. Farmer and J. J. Sidorowich, "Predicting chaotic time series," Physical Review Letters, vol. 59, pp. 845-848, 1987.

[79] S. Haykin, "Neural networks expand SP's horizons," IEEE Signal Processing Magazine, vol. 13, pp. 24-49, 1996.

[80] M. T. Rosenstein, J. J. Collins, and C. J. De Luca, "Reconstruction expansion as a geometry-based framework for choosing proper delay times," Physica D: Nonlinear Phenomena, vol. 73, pp. 82-98, 1994.

[81] D. Kugiumtzis, "State space reconstruction parameters in the analysis of chaotic time series -- the role of the time window length," Physica D: Nonlinear Phenomena, vol. 95, pp. 13-28, 1996.

[82] J.-M. Kuo and J. C. Principe, "Reconstructed dynamics and chaotic signal modeling," presented at IEEE World Congress on Neural Networks, IEEE International Conference on Computational Intelligence, 1994, pp. 3131-3136 vol.5, 1994.

[83] M. Orr, "Optimising the widths of radial basis functions," presented at Vth Brazilian Symposium on Neural Networks, 1998. Proceedings., pp. 26-29, 1998.

[84] M. M. Brizzotti and A. C. P. L. F. Carvalho, "The influence of clustering techniques in the RBF networks generalization," presented at Seventh International Conference on Image Processing and Its Applications (Conf. Publ. No. 465), pp. 87-92 vol.1, 1999.

[85] J. Moody and C. J. Darken, "Fast learning in networks of locally-tuned processing units," Neural Computation, vol. 1, pp. 281-294, 1989.

[86] J. Moody and C. J. Darken, "Learning with localized receptive fields," in Proceedings of the 1988 Connectionist Models Summer Scool, T. e. al., Ed.: Morgan-Kauffman, 1988, pp. 133-143.

[87] R. O. Duda, P. E. Hart, and D. G. Stork, Pattern classification, 2nd ed. New York: Wiley, 2000.

[88] D. E. Gustafson and W. C. Kessel, "Fuzzy Clustering with a Fuzzy Covariance Matrix," presented at Proceedings of the IEEE Conference on Decision and Control, San Diego, pp. 761-766, 1979.

[89] G. J. McLachlan and T. Krishnan, The EM algorithm and extensions. New York: Wiley, 1996.

[90] M. Kearns, Y. Mansour, and A. Ng, "An Information-Theoretic Analysis of Hard and Soft Assignment Methods for Clustering," presented at Proceedings of the 13th Annual Conference on Uncertainty in Artificial Intelligence (UAI-97), San Francisco, CA, pp. 282-293, 1997.

[91] R. Babuska, P. J. van der Veen, and U. Kaymak, "Improved covariance estimation for Gustafson-Kessel clustering," presented at Proceedings of the 2002 IEEE International Conference on Fuzzy Systems, 2002. FUZZ-IEEE'02., pp. 1081-1085, 2002.

[92] P. S. Bradley and U. M. Fayyad, "Refining initial points for K-Means clustering," Proceedings of the 15th International Conference on Machine Learning, pp. 91-99, 1998.

[93] S. Z. Selim and M. A. Ismail, "K-means-type algorithms: a generalized convergence theorem and characterization of local optimality," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-6, pp. 81-87, 1984.

[94] B. Everitt, S. Landau, and M. Leese, Cluster analysis, 4th ed. London: Arnold, 2001.

[95] C. Ordonez and E. Omiecinski, "FREM: Fast and robust EM clustering for large data sets," International Conference on Information and Knowledge Management, Proceedings, pp. 590-599, 2002.

[96] M. Dang and G. Govaert, "Spatial Fuzzy Clustering using EM and Markov Random Fields," Systems Research and Information Systems, vol. 8, pp. 183-202, 1998.

[97] T. Kohonen, Self-organizing maps, 3rd ed. New York: Springer, 2000.

[98] S. Haykin, Neural networks : a comprehensive foundation, 2nd ed. Upper Saddle River, NJ: Prentice Hall, 1999.

[99] J. Park and I. W. Sandberg, "Nonlinear approximations using elliptic basis function networks," Circuits, Systems, and Signal Processing, vol. 13, pp. 99-113, 1994.

[100] J. Park, Personal Communication, 13 March 2003.

[101] T. Poggio and F. Girosi, "A theory of networks for approximation and learning," MIT AI Lab, Cambridge, Massachusetts, Technical Report 1140, 1989.

[102] J. H. Friedman, "Regularized Discriminant Analysis," Journal of the American Statistical Association, vol. 84, pp. 165-175, 1989.

[103] W. H. Press, S. T. Teukolsky, W. T. Vetterling, and B. P. Flannery, Numerical Recipes in C: the art of scientific computing. Cambridge, England: Cambridge University Press, 1992.

[104] M. T. Heath, Scientific computing : an introductory survey, 2nd ed. Dubuque, Iowa: McGraw-Hill, 2001.

[105] A. Greenbaum, Iterative methods for solving linear systems. Philadelphia, PA: Society for Industrial and Applied Mathematics, 1997.

[106] F. Murtagh, "Comments on `Parallel algorithms for hierarchical clustering and cluster validity'," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, pp. 1056-1057, 1992.

[107] D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks," Complex Systems, vol. 2, pp. 321-355, 1988.

[108] G. H. Golub and C. F. Van Loan, Matrix computations, 2nd ed. Baltimore, Md.: Johns Hopkins University Press, 1989.

[109] S. Geman, E. Bienenstock, and R. Doursat, "Neural networks and the bias/valance dilemma," Neural Computation, vol. 4, pp. 1-58, 1992.

[110] J. A. Rice, Mathematical statistics and data analysis, 2nd ed. Belmont, CA: Duxbury Press, 1995.

[111] D. J. C. MacKay, "Bayesian Interpolation," Neural Computation, vol. 4, pp. 415-447, 1992.

[112] M. Orr, "Introduction to radial basis function networks," Center for Cognitive Science, University of Edinburgh 1996.

[113] W. S. Sarle, "Neural Network FAQ, part 1 of 7: Introduction", periodic posting to the Usenet newsgroup comp.ai.neural-nets, Available: ftp://ftp.sas.com/pub/neural/FAQ.html, 1997.

[114] O. Nelles, Nonlinear system identification : from classical approaches to neural networks and fuzzy models. Berlin; New York: Springer, 2001.

[115] B. Flury, A first course in multivariate statistics. New York: Springer, 1997.

[116] K. Fukunaga, Introduction to statistical pattern recognition. New York: Academic Press, 1972.

[117] H. Anton, Elementary linear algebra, 6th ed. New York: Wiley, 1991.

[118] C. M. Bishop, "Training with Noise is Equivalent to Tikhonov Regularization," Neural Computation, vol. 7, pp. 108-116, 1995.

[119] R. Murray-Smith, "A Local Model Network Approach to Nonlinear Modelling," PhD Thesis, University of Strathclyde, Computer Science Department, Nov. 1994.

[120] S. Haykin, C. Krasnor, T. J. Nohara, B. W. Currie, and D. Hamburger, "A coherent dual-polarized radar for studying the ocean environment," IEEE Transactions on Geoscience and Remote Sensing, vol. 29, pp. 189-191, 1991.

[121] H. Kantz and T. Schreiber, Nonlinear Time Series Analysis. Cambridge, UK: Cambridge Univ. Press, 1997.

[122] D. Ruelle, "Diagnosis of dynamical systems with fluctuating parameters," in Proceedings of the Royal Society of London, Series A, vol. 413. Princeton, NJ, 1987, pp. 5-8.

[123] U. Hübner, C.-O. Weiss, N. B. Abraham, and D. Tang, "Lorenz-Like Chaos in NH3-FIR Lasers (Data Set A)," in Time series prediction: forecasting the future and understanding the past : proceedings of the NATO Advanced Research Workshop on Comparative Time Series Analysis, held in Santa Fe, New Mexico, May 14-17, 1992, A. S. Weigend and N. A. Gershenfeld, Eds. Reading, MA: Addison-Wesley Pub. Co., 1994, pp. 73-104.

[124] N. A. Gershenfeld and A. S. Weigend, "The Future of Time Series: Learning and Understanding," in Time series prediction: forecasting the future and understanding the past : proceedings of the NATO Advanced Research Workshop on Comparative Time Series Analysis, held in Santa Fe, New Mexico, May 14-17, 1992, A. S. Weigend and N. A. Gershenfeld, Eds. Reading, MA: Addison-Wesley Pub. Co., 1994, pp. 1-70.

[125] D. G. Manolakis, V. K. Ingle, and S. M. Kogon, Statistical and adaptive signal processing : spectral estimation, signal modeling, adaptive filtering, and array processing. Boston: McGraw-Hill, 2000.

[126] L. Cao, Personal Communication, 16 Jan 2003.

[127] S. Haykin and J. Principe, "Making sense of a complex world," IEEE Signal Processing Magazine, vol. 15, pp. 66-68, 1998.

[128] G. S. Patel and S. Haykin, "Chaotic Dynamics," in Kalman filtering and neural networks, S. Haykin, Ed. New York: Wiley, 2001, pp. xiii, 284.

[129] T. Schreiber and H. Kantz, "Observing and predicting chaotic signals: is 2% noise too much?," Predictability of complex dynamical systems, pp. 43-65, 1996.

[130] J. B. Gao, "Detecting nonstationarity and state transitions in a time series," Physical Review E, vol. 63, p. 066202, 2001.

[131] J. B. Gao, W. W. Tung, and N. Rao, "Noise-induced Hopf-bifurcation-type sequence and transition to chaos in the lorenz equations," Physical Review Letters, vol. 89, p. 254101, 2002.

[132] K. Judd, M. Small, and A. I. Mees, "Achieving Good Nonlinear Models: Keep It Simple, Vary the Embedding, and Get the Dynamics Right," in Nonlinear dynamics and statistics, A. I. Mees, Ed. Boston: Birkhäuser, 2001, pp. 65-80.

[133] S. Eubank and J. Doyne Farmer, "Introduction to dynamical systems," in Introduction to nonlinear physics, L. Lam, Ed. New York: Springer, 1997, pp. 55-105.

[134] S. E. Newhouse, D. Ruelle, and F. Takens, "Occurrence of strange axiom A attractors near quasi-periodic flows on T^m, ($m \geq 3$)," Communications in Mathematical Physics, vol. 64, pp. 35-40, 1978.

[135] J. P. Crutchfield and K. Kaneko, "Are Attractors Relevant to Turbulence?," Physical Review Letters, vol. 60, pp. 2715-2718, 1988.

[136] J. W. Havstad and C. L. Ehlers, "Attractor dimension of non-stationary dynamical sytems from small data sets," Physical Review A, vol. 39, pp. 845-853, 1989.

[137] T. W. Frison and H. D. I. Abarbanel, "Identification and quantification of nonstationary chaotic behavior," presented at IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-97, pp. 2393-2396 vol.3, 1997.

[138] E. Jakeman and P. N. Pusey, "A model for non-Rayleigh sea echo," IEEE Transactions on Antennas and Propagation, vol. AP24, pp. 806-814, 1976.

[139] K. D. Ward, "Compound representation of high resolution sea clutter," Electronics Letters, vol. 17, p. 561, 1981.

[140] I. S. Gradshteyn, I. M. Ryzhik, and A. Jeffrey, Table of integrals, series, and products, 4th ed. New York: Academic Press, 1965.

[141] E. Jakeman and P. N. Pusey, "Significance of K distributions in scattering experiments," Physical Review Letters, vol. 40, p. 546, 1978.

[142] E. Jakeman, "On the statistics of K-distributed noise," Journal of Physics A (Mathematical and General), vol. 13, p. 31, 1980.

[143] E. Jakeman, "Enhanced backscattering," in IEE Colloquium on Radar Clutter and Multipath Propagation, 1989, pp. 5/1-5/5.

[144] E. Jakeman and P. N. Pusey, "Light scattering from electrohydrodynamic turbulence in liquid crystals," Physics Letters A, vol. 44A, pp. 456-458, 1973.

[145] E. Jakeman and P. N. Pusey, "The statistics of light scattered by a random phase screen," Journal of Physics A (Mathematical and General), vol. 6, pp. L88-L92, 1973.

[146] E. Jakeman and P. N. Pusey, "Non-Gaussian fluctuations in electromagnetic radiation scattered by random phase screen. I. Theory," Journal of Physics A (Mathematical and General), vol. 8, pp. 369-391, 1975.

[147] I. Nabney, NETLAB : algorithms for pattern recognitions, 2nd printing, with corrections. ed. London ; New York: Springer, 2002.

[148] R. E. Bellman, Adaptive control processes : a guided tour. Princeton, N.J.: Princeton University Press, 1961.

[149] S. Elaydi, Discrete chaos. Boca Raton: Chapman & Hall/CRC, 2000.

APPENDIX A
Derivation of K-distribution

The most general clutter model at this time is the Rayleigh mixture model [4]. It is assumed that in each patch of ocean surface, the received signal is dominated by a small number of independent scatterers (this is reminiscent of Rayleigh fading). The probability density of r_c, the clutter voltage envelope, is then given as

$$f(r_c) = \int_0^\infty p(r_c \mid \sigma) q(\sigma) d\sigma = \int_0^\infty p(r_c \mid \sigma) dQ(\sigma), \tag{A.1}$$

where

$$p(r_c \mid \sigma) = \frac{r_c}{\sigma} e^{-\frac{r_c^2}{2\sigma}} u(\sigma) \tag{A.2}$$

is Rayleigh distributed, and $u(\bullet)$ is the unit step function. The random variable σ describes clutter power, and is distributed according to some probability density distribution $q(\sigma)$ (with associated cumulative distribution function $Q(\sigma)$).

Jakeman and Pusey [138] argued that when radar illuminates a large area of the sea, the envelope of the return signal can be well approximated by a Rayleigh distribution. This is a consequence of the central limit theorem, since the signal can be thought of as being the vector sum of randomly phased components from a large number of independent scatterers.

The K-distribution was introduced by Jakeman and Pusey in 1976 [138], but it was Ward who first showed the K-distribution to be the closed form solution of a Gamma distribution modulated by a Rayleigh distribution in reference [139]. The Gamma distribution can be derived by taking a random walk where the number of steps, n, obey the negative binomial distribution. In the limit as $n \to \infty$, *i.e.* the radar patch

size is large compared to the density of scatterers, yet small compared to the characteristic bunching size, the negative binomial distribution approaches the gamma distribution [139]. According to Peebles [4], the names Chi and Gamma are used interchangeably in the literature.

The K-distribution can be derived from (A.1) by substituting a Gamma distribution:

$$f(\sigma) = \frac{b}{\Gamma(b)\bar{\sigma}}\left(\frac{b\sigma}{\bar{\sigma}}\right)^{b-1} e^{-\frac{b\sigma}{\bar{\sigma}}} u(\sigma) \tag{A.3}$$

where $b > 0$, is a scale parameter that relates only to the mean of the clutter, $\Gamma(\cdot)$ is the gamma function, and $\bar{\sigma}$ is the average power in the bandpass clutter signal having the voltage amplitude r_c. Substituting (A.2) and (A.3) into (A.1), we obtain:

$$\begin{aligned} f(r_c) &= \int_0^\infty \frac{r_c}{\sigma} e^{-\frac{r_c^2}{2\sigma}} \frac{b}{\Gamma(b)\bar{\sigma}}\left(\frac{b\sigma}{\bar{\sigma}}\right)^{b-1} e^{-\frac{b\sigma}{\bar{\sigma}}} u(\sigma) d\sigma \\ &= \frac{r_c b}{\Gamma(b)\bar{\sigma}}\left(\frac{b}{\bar{\sigma}}\right)^{b-1} \int_0^\infty \sigma^{b-2} e^{-\frac{r_c^2}{2\sigma} - \frac{b}{\bar{\sigma}}\sigma} u(\sigma) d\sigma \\ &= \frac{r_c}{\Gamma(b)}\left(\frac{b}{\bar{\sigma}}\right)^{b} \int_0^\infty \sigma^{b-2} e^{-\frac{r_c^2}{2\sigma} - \frac{b}{\bar{\sigma}}\sigma} u(\sigma) d\sigma. \end{aligned} \tag{A.4}$$

Given the following relationship:

$$\int_0^\infty \sigma^{v-1} e^{-\gamma\sigma - \frac{\beta}{\sigma}} d\sigma = 2\left(\frac{\beta}{\gamma}\right)^{\frac{v}{2}} B_v\left(2\sqrt{\gamma\beta}\right) \tag{A.5}$$

where $\mathrm{Re}(\gamma) > 0$, $\mathrm{Re}(\beta) > 0$ and $B_v(\cdot)$ is the modified Bessel function of order v [140], we substitute $v = b - 1$, $\gamma = \frac{b}{\bar{\sigma}}$, and $\beta = \frac{r_c^2}{2}$ to get:

$$
\begin{aligned}
f(r_c) &= \frac{r_c}{\Gamma(b)}\left(\frac{b}{\bar{\sigma}}\right)^b 2\left(\frac{\beta}{\gamma}\right)^{\frac{v}{2}} B_v\left(2\sqrt{\gamma\beta}\right) \\
&= \frac{2r_c}{\Gamma(b)}\left(\frac{b}{\bar{\sigma}}\right)^b \left(\frac{r_c^2\bar{\sigma}}{2b}\right)^{\frac{b-1}{2}} B_{b-1}\left(2\sqrt{\frac{b}{\bar{\sigma}}\frac{r_c^2}{2}}\right) \\
&= \frac{4r_c}{\Gamma(b)}\frac{1}{2}\left(\frac{b}{\bar{\sigma}}\right)^b \left(\frac{1}{2}\right)^{\frac{b-1}{2}} \left(\frac{b}{\bar{\sigma}}\right)^{\frac{1-b}{2}} r_c^{b-1} B_{b-1}\left(r_c\sqrt{\frac{2b}{\bar{\sigma}}}\right) \\
&= \frac{4}{\Gamma(b)}\left(\frac{b}{2\bar{\sigma}}\right)^{\frac{b+1}{2}} r_c^b B_{b-1}\left(r_c\sqrt{\frac{2b}{\bar{\sigma}}}\right).
\end{aligned}
\tag{A.6}
$$

This is equivalent to the K-distribution, which is defined as:

$$
f(r_c) = \frac{4}{\Gamma(b)}\left(\frac{b}{2\bar{\sigma}}\right)^{\frac{b+1}{2}} r_c^b B_{b-1}\left(r_c\sqrt{\frac{2b}{\bar{\sigma}}}\right)u(r_c). \tag{A.7}
$$

One interesting property of the K-distribution is that the sum of a finite number, *N*, of vectors whose amplitudes are K-distributed, is also K-distributed, albeit with a scaled shape parameter, *bN* [6].

Besides sea clutter, the K-distribution can be used to model scattering of laser light in a turbulent layer of air, [141-143], and also scattering of laser light by a turbulent layer of crystal [144-146].

APPENDIX B
Positive Definiteness of $(A+B)^{-1}$

Theorem B.1 The covariance matrix is positive semidefinite

Let the covariance matrix be $\boldsymbol{\Sigma} \triangleq E\left[(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^H\right]$, where $\mathbf{x}$ is a random vector and $\boldsymbol{\mu}$ is the mean, such that $\mathbf{x}, \boldsymbol{\mu} \in \mathbb{C}^D$, where $D \in \mathbb{Z}^+$. Let $\mathbf{a}$ be any vector $\mathbf{a} \in \mathbb{C}^D$. Define $y \triangleq \mathbf{a}^H(\mathbf{x}-\boldsymbol{\mu})$. Substitute $\boldsymbol{\Sigma}$ into $E\left[y^H y\right] \geq 0$,

$$\begin{aligned} E\left[y^H y\right] &= E\left[\mathbf{a}^H(\mathbf{x}-\boldsymbol{\mu})(\mathbf{x}-\boldsymbol{\mu})^H\mathbf{a}\right] \\ &= \mathbf{a}^H\boldsymbol{\Sigma}\mathbf{a} \geq 0. \end{aligned} \tag{B.1}$$

Since $\mathbf{a}$ is any vector, this means that $\boldsymbol{\Sigma}$ is positive semidefinite.

Theorem B.2 Sum of positive definite matrix and positive semidefinite matrix is positive definite

Consider a positive definite matrix $\mathbf{A} \in \mathbb{R}^{D\times D}$, a positive semidefinite matrix $\mathbf{B} \in \mathbb{R}^{D\times D}$, and any nonzero vector $\mathbf{x} \in \mathbb{C}^D$, where $D \in \mathbb{Z}^+$. The following relationship holds:

$$\mathbf{x}^H\left(\mathbf{A}+\mathbf{B}\right)\mathbf{x} = \mathbf{x}^H\mathbf{A}\mathbf{x} + \mathbf{x}^H\mathbf{B}\mathbf{x}. \tag{B.2}$$

As $\mathbf{x}^H\mathbf{A}\mathbf{x} > 0$ and $\mathbf{x}^H\mathbf{B}\mathbf{x} \geq 0$, this means that $\mathbf{x}^H\mathbf{A}\mathbf{x} + \mathbf{x}^H\mathbf{B}\mathbf{x} > 0$. Thus, the sum of a positive definite and positive semidefinite matrix is positive definite.

Theorem B.3 The inverse of a positive definite matrix is positive definite

From the definition of the eigenvalue and eigenvector,

$$\begin{aligned} \mathbf{A}\mathbf{x} &= \lambda\mathbf{x} \\ \frac{1}{\lambda}\mathbf{x} &= \mathbf{A}^{-1}\mathbf{x}, \end{aligned} \tag{B.3}$$

where $\mathbf{A} \in \mathbb{R}^{D\times D}$, $\mathbf{x} \in \mathbb{R}^D$ and $\lambda \in \mathbb{R}$. Eq. (B.3) shows that if A has the eigenvalue λ, then the corresponding eigenvalue of $\mathbf{A}^{-1}$ would be $1/\lambda$.

Consider a positive definite matrix $\mathbf{C} \in \mathbb{R}^{D \times D}$, where $D \in \mathbb{Z}^{+}$; each eigenvalue of **C** is positive. From Eq. (B.3), each eigenvalue of $\mathbf{C}^{-1}$ is also positive. This implies that $\mathbf{C}^{-1}$ is also positive definite.

From Theorems B.1, B.2 and B.3, $\left(\mathbf{A}+\mathbf{B}\right)^{-1}$ is positive definite, where **A** is positive definite, and **B** is positive semidefinite.

APPENDIX C
Proof that Mahalanobis Norm is a Valid Metric

The Mahalanobis norm is defined by

$$d(\mathbf{x},\mathbf{y}) \triangleq \sqrt{(\mathbf{x}-\mathbf{y})^H \mathbf{M}^{-1}(\mathbf{x}-\mathbf{y})}\,, \tag{C.1}$$

where $\mathbf{x},\mathbf{y} \in \mathbb{C}^D$, $\mathbf{M} \in \mathbb{R}^{D\times D}$ is symmetric positive definite, and $D \in \mathbb{Z}^+$. Consider a finite dimensional complex vector space, and associate it with the distance function $d(\mathbf{x},\mathbf{y}) : \mathbb{C}^D \times \mathbb{C}^D \to \mathbb{R}$. This is a valid metric; it has 4 properties (see Glossary):

1. $d(\mathbf{x},\mathbf{y}) \geq 0$, $\forall \mathbf{x},\mathbf{y} \in \mathbb{C}^D$, $D \in \mathbb{Z}^+$.

Inverse of any symmetric positive definite matrix $\mathbf{M} \in \mathbb{R}^{D\times D}$ is positive definite (Theorem B.3 of Appendix B). If $\mathbf{x} \neq \mathbf{y}$,

$$\begin{aligned} (\mathbf{x}-\mathbf{y})^H \mathbf{M}^{-1}(\mathbf{x}-\mathbf{y}) &> 0 \\ \sqrt{(\mathbf{x}-\mathbf{y})^H \mathbf{M}^{-1}(\mathbf{x}-\mathbf{y})} &> 0. \end{aligned} \tag{C.2}$$

If $\mathbf{x} = \mathbf{y}$, $d(\mathbf{x},\mathbf{y}) = 0$. An intuitive way to see this is that the square root of any value of a quadratic function has to be nonnegative. Furthermore, this property also implies another property:

2. $d(\mathbf{x},\mathbf{y}) = 0$ if and only if $\mathbf{x} = \mathbf{y}$, $\forall \mathbf{x},\mathbf{y} \in \mathbb{C}^D$, $D \in \mathbb{Z}^+$.

Clearly, if $\mathbf{x} \neq \mathbf{y}$, then $d(\mathbf{x},\mathbf{y}) > 0$.

3. $d(\mathbf{x},\mathbf{y}) = d(\mathbf{y},\mathbf{x})$, $\forall \mathbf{x},\mathbf{y} \in \mathbb{C}^D$, $D \in \mathbb{Z}^+$.

$$d(\mathbf{y},\mathbf{x}) = \sqrt{\left(-(\mathbf{x}-\mathbf{y})\right)^H \mathbf{M}^{-1}\left(-(\mathbf{x}-\mathbf{y})\right)} = d(\mathbf{x},\mathbf{y}) \tag{C.3}$$

An intuitive way to see this is that the positive square root of a quadratic function is symmetrical.

4. $d(\mathbf{x},\mathbf{y})+d(\mathbf{y},\mathbf{z})\geq d(\mathbf{x},\mathbf{z})$, $\forall \mathbf{x},\mathbf{y},\mathbf{z}\in\mathbb{C}^D$, $D\in\mathbb{Z}^+$.

This is the triangle equality, which is usually proved using the Cauchy-Schwartz inequality. The trivial case $\mathbf{x} = \mathbf{y} = \mathbf{z}$ results in equality, and hence the $\geq$ sign. The triangle inequality can be reformulated by substituting $\mathbf{x}-\mathbf{y}=\mathbf{w}$ and $\mathbf{y}-\mathbf{z}=\mathbf{v}$:

$$\begin{aligned} d(\mathbf{x},\mathbf{y})+d(\mathbf{y},\mathbf{z}) &\geq d(\mathbf{x},\mathbf{z}) \\ \sqrt{(\mathbf{x}-\mathbf{y})^H\mathbf{M}^{-1}(\mathbf{x}-\mathbf{y})}+\sqrt{(\mathbf{y}-\mathbf{z})^H\mathbf{M}^{-1}(\mathbf{y}-\mathbf{z})} &\geq \sqrt{(\mathbf{x}-\mathbf{z})^H\mathbf{M}^{-1}(\mathbf{x}-\mathbf{z})} \\ \sqrt{\mathbf{w}^H\mathbf{M}^{-1}\mathbf{w}}+\sqrt{\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v}} &\geq \sqrt{(\mathbf{w}+\mathbf{v})^H\mathbf{M}^{-1}(\mathbf{w}+\mathbf{v})}. \end{aligned} \tag{C.4}$$

Using the extended Cauchy-Schwartz inequality, (C.9):

$$\begin{aligned} \left|\mathbf{u}^H\mathbf{v}\right|^2 &\leq \mathbf{u}^H\mathbf{M}\mathbf{u}\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v} \\ 2\sqrt{\mathbf{u}^H\mathbf{M}\mathbf{u}\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v}} &\geq \left|\mathbf{u}^H\mathbf{v}\right|+\left|\mathbf{v}^H\mathbf{u}\right| \\ 2\sqrt{\mathbf{u}^H\mathbf{M}^H\mathbf{M}^{-1}\mathbf{M}\mathbf{u}\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v}} &\geq \left|\mathbf{v}^H\mathbf{M}^{-1}\mathbf{M}\mathbf{u}\right|+\left|\mathbf{u}^H\mathbf{M}^H\mathbf{M}^{-1}\mathbf{v}\right|. \end{aligned} \tag{C.5}$$

Substituting $\mathbf{w} = \mathbf{M}\mathbf{u}$ into (C.5),

$$\begin{aligned} 2\sqrt{\mathbf{w}^H\mathbf{M}^{-1}\mathbf{w}\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v}} &\geq \left|\mathbf{v}^H\mathbf{M}^{-1}\mathbf{w}\right|+\left|\mathbf{w}^H\mathbf{M}^{-1}\mathbf{v}\right| \\ \left|\mathbf{w}^H\mathbf{M}^{-1}\mathbf{w}\right|+2\sqrt{\mathbf{w}^H\mathbf{M}^{-1}\mathbf{w}\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v}}+\left|\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v}\right| &\geq \left|\mathbf{v}^H\mathbf{M}^{-1}\mathbf{w}\right|+\left|\mathbf{w}^H\mathbf{M}^{-1}\mathbf{v}\right| \\ &\quad +\left|\mathbf{w}^H\mathbf{M}^{-1}\mathbf{w}\right|+\left|\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v}\right|. \end{aligned} \tag{C.6}$$

Using the triangle inequality for complex numbers, *i.e.* $|z_1|+|z_2|\geq|z_3|$, whereby $z_1, z_2, z_3 \in \mathbb{C}$, and substituting into (C.6),

$$\begin{aligned} \left(\sqrt{\left|\mathbf{w}^H\mathbf{M}^{-1}\mathbf{w}\right|}+\sqrt{\left|\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v}\right|}\right)^2 &\geq \left|\mathbf{v}^H\mathbf{M}^{-1}\mathbf{w}+\mathbf{w}^H\mathbf{M}^{-1}\mathbf{v}+\mathbf{w}^H\mathbf{M}^{-1}\mathbf{w}+\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v}\right| \\ \left(\sqrt{\mathbf{w}^H\mathbf{M}^{-1}\mathbf{w}}+\sqrt{\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v}}\right)^2 &\geq \left|(\mathbf{w}+\mathbf{v})^H\mathbf{M}^{-1}(\mathbf{w}+\mathbf{v})\right| \\ \sqrt{\mathbf{w}^H\mathbf{M}^{-1}\mathbf{w}}+\sqrt{\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v}} &\geq \sqrt{\left|(\mathbf{w}+\mathbf{v})^H\mathbf{M}^{-1}(\mathbf{w}+\mathbf{v})\right|}. \end{aligned} \tag{C.7}$$

(C.7) is equivalent to (C.4). Hence, the Mahalanobis norm is a valid metric QED.

Theorem C.1 Cauchy-Schwartz Inequality

For any 2 nonzero vectors $\mathbf{x} \in \mathbb{C}^D$, $\mathbf{y} \in \mathbb{C}^D$, and $D \in \mathbb{Z}^+$,

$$(\mathbf{x}^H \mathbf{y})^2 \leq (\mathbf{x}^H \mathbf{x})(\mathbf{y}^H \mathbf{y}) \tag{C.8}$$

with equality if $\mathbf{y} = c\mathbf{x}$ for some $c \in \mathbb{C}$, since $(\mathbf{x}^H c\mathbf{x})^2 = (\mathbf{x}^H \mathbf{x})(\mathbf{x}^H c^H c\mathbf{x})$. However, a more general form of the Cauchy-Schwartz Inequality is required:

Theorem C.2 Extended Cauchy-Schwartz Inequality

For any 2 nonzero vectors $\mathbf{u} \in \mathbb{C}^D$ and $\mathbf{v} \in \mathbb{C}^D$, any symmetric positive definite matrix $\mathbf{M} \in \mathbb{R}^{D \times D}$, and $D \in \mathbb{Z}^+$,

$$(\mathbf{u}^H \mathbf{v})^2 \leq (\mathbf{u}^H \mathbf{M}\mathbf{u})(\mathbf{v}^H \mathbf{M}^{-1}\mathbf{v}), \tag{C.9}$$

with equality if $\mathbf{v} = c\mathbf{M}\mathbf{u}$ for some $c \in \mathbb{C}$ (or, equivalently, $\mathbf{u} = c_2 \mathbf{M}^{-1}\mathbf{v}$ for some $c_2 \in \mathbb{C}$).

Proof

Since $\mathbf{M}$ is symmetric positive definite, there exists a non-singular matrix $\mathbf{M}^{1/2}$ such that $\left(\mathbf{M}^{1/2}\right)^2 = \mathbf{M}$, with inverse $\mathbf{M}^{-1/2} = \left(\mathbf{M}^{1/2}\right)^{-1}$. Let $\mathbf{x} = \mathbf{M}^{1/2}\mathbf{u}$ and $\mathbf{y} = \mathbf{M}^{-1/2}\mathbf{v}$. Then

$$\begin{aligned} (\mathbf{u}^H \mathbf{v})^2 &= (\mathbf{x}^H \mathbf{M}^{-1/2}\mathbf{M}^{1/2}\mathbf{y})^2 \\ &= (\mathbf{x}^H \mathbf{y})^2 \\ &\leq (\mathbf{x}^H \mathbf{x})(\mathbf{y}^H \mathbf{y}) = (\mathbf{u}^H \mathbf{M}\mathbf{u})(\mathbf{v}^H \mathbf{M}^{-1}\mathbf{v}). \end{aligned} \tag{C.10}$$

Equality in (C.10) holds exactly if $\mathbf{y} = c\mathbf{x}$ for some $c \in \mathbb{C}$, *i.e.*, if $\mathbf{M}^{-1/2}\mathbf{v} = c\mathbf{M}^{1/2}\mathbf{u}$ or $\mathbf{v} = c\mathbf{M}\mathbf{u}$ for some $c \in \mathbb{C}$.

As a consequence of the Extended Cauchy-Schwartz Inequality, for a given vector $\mathbf{v} \in \mathbb{C}^D$, and a given symmetric positive definite matrix $\mathbf{M}$,

$$\max_{\mathbf{u}\in\mathbb{C}^D} \frac{(\mathbf{u}^H\mathbf{v})^2}{\mathbf{u}^H\mathbf{M}\mathbf{u}} = \mathbf{v}^H\mathbf{M}^{-1}\mathbf{v}\,, \tag{C.11}$$

and the maximum is attained for any vector $\mathbf{u}$ proportional to $\mathbf{M}^{-1}\mathbf{v}$:

$$\begin{aligned}(\mathbf{u}^H\mathbf{v})^2 &\le (\mathbf{u}^H\mathbf{M}\mathbf{u})(\mathbf{v}^H\mathbf{M}^{-1}\mathbf{v})\\ (\mathbf{u}^H\mathbf{M}\mathbf{u})^2 &\le (\mathbf{u}^H\mathbf{M}\mathbf{u})(\mathbf{u}^H\mathbf{M}\mathbf{M}^{-1}\mathbf{M}\mathbf{u}).\end{aligned} \tag{C.12}$$

It is easy to build on these results to show also that the Mahalanobis Norm is a valid norm. The property required is:

5. $d(k\mathbf{x},k\mathbf{y}) = |k|\,d(\mathbf{x},\mathbf{y})$ for any $k\in\mathbb{C}$.

$$d(k\mathbf{x},k\mathbf{y}) = \sqrt{\left((\mathbf{x}-\mathbf{y})^H k^H\right)\mathbf{M}^{-1}\left(k(\mathbf{x}-\mathbf{y})\right)} = |k|\,d(\mathbf{x},\mathbf{y})\,. \tag{C.13}$$

Note

$d^2(\mathbf{x},\mathbf{y})$ is not a valid metric:

Suppose $d^2(\mathbf{x},\mathbf{y})$ is a valid metric. Then, the triangle inequality should hold:

$$\begin{aligned}d^2(\mathbf{x},\mathbf{y}) + d^2(\mathbf{y},\mathbf{z}) &\ge d^2(\mathbf{x},\mathbf{z})\\ (\mathbf{x}-\mathbf{y})^H\mathbf{M}^{-1}(\mathbf{x}-\mathbf{y}) + (\mathbf{y}-\mathbf{z})^H\mathbf{M}^{-1}(\mathbf{y}-\mathbf{z}) &\ge (\mathbf{x}-\mathbf{z})^H\mathbf{M}^{-1}(\mathbf{x}-\mathbf{z}),\end{aligned} \tag{C.14}$$

$\forall\mathbf{x},\mathbf{y},\mathbf{z}\in\mathbb{C}^D$. Substituting $\mathbf{w}=\mathbf{x}-\mathbf{y}$ and $\mathbf{v}=\mathbf{y}-\mathbf{z}$ into (C.14),

$$\begin{aligned}\mathbf{w}^H\mathbf{M}^{-1}\mathbf{w} + \mathbf{v}^H\mathbf{M}^{-1}\mathbf{v} &\ge (\mathbf{w}+\mathbf{v})^H\mathbf{M}^{-1}(\mathbf{w}+\mathbf{v})\\ 0 &\ge \mathbf{v}^H\mathbf{M}^{-1}\mathbf{w} + \mathbf{w}^H\mathbf{M}^{-1}\mathbf{v}.\end{aligned} \tag{C.15}$$

Substituting $\mathbf{w} = \mathbf{M}\mathbf{u}$ into (C.15),

$$\begin{aligned}0 &\ge \mathbf{v}^H\mathbf{u} + \mathbf{u}^H\mathbf{v}\\ 2\,\mathrm{Re}\left[\mathbf{v}^H\mathbf{u}\right] &\le 0\\ \mathrm{Re}\left[\mathbf{v}^H\mathbf{M}^{-1}\mathbf{w}\right] &\le 0\\ \mathrm{Re}\left[(\mathbf{y}-\mathbf{z})^H\mathbf{M}^{-1}(\mathbf{x}-\mathbf{y})\right] &\le 0.\end{aligned} \tag{C.16}$$

(C.16) is not necessarily true; if $\mathbf{w} = \mathbf{v}$, the positive definiteness of $\mathbf{M}^{-1}$ ensures that $\mathbf{w}^H\mathbf{M}^{-1}\mathbf{w} > 0$. This contradicts (C.16). Thus, $\mathbf{x} - \mathbf{y} = \mathbf{y} - \mathbf{z}$ is a counterexample for (C.14) and hence $d^2(\mathbf{x},\mathbf{y})$ is not a valid metric.

APPENDIX D
Expectation Maximization

Expectation Maximization (EM) is an iterative method which can be used to estimate mixture distribution parameters. Define the prior distribution (prior to incorporating information about the location of the data points) of the *j*-th cluster to be:

$$P(\omega_j) \triangleq \frac{N_j}{N}, \tag{D.1}$$

where $j = 1, ..., M_c$, N_j is the number of points in the *j*-th cluster, and N is the total number of points in the dataset [147]. The mean of the cluster is then given as:

$$\boldsymbol{\mu}_j = \frac{1}{N_j} \sum_{i \in I_j} \boldsymbol{\psi}_i , \tag{D.2}$$

where I_j is the set of indices of points belonging to the *j*-th cluster and $\boldsymbol{\psi}_i$ is the *i*-th data point. The variance of each cluster is given as:

$$\sigma_j^2 = \frac{1}{pN_j} \sum_{i \in I_j} \left\| \boldsymbol{\psi}_i - \boldsymbol{\mu}_j \right\|^2 , \tag{D.3}$$

where p is the dimension of the data.

Not knowing which component generated each data point, we consider a hypothetical complete data set in which each data point is labelled with the component that generated it. So, for each *i*-th data point, $\boldsymbol{\psi}_i$, there is a corresponding class label, z_i, which is an integer in the range 1, ..., M_c. The complete data point is $\boldsymbol{\xi}_i \triangleq \begin{pmatrix} \boldsymbol{\psi}_i \\ z_i \end{pmatrix}$. The likelihood of a complete data point if $z_i = j$ is

$$\begin{aligned} p((\boldsymbol{\psi}_i \quad z_i = j)^T \mid \boldsymbol{\theta}) &= p(\boldsymbol{\psi}_i \mid z_i = j, \boldsymbol{\theta}) P(z_i = j \mid \boldsymbol{\theta}) \\ &= p(\boldsymbol{\psi}_i \mid \boldsymbol{\theta}_j) P(z_i = j \mid \boldsymbol{\theta}), \end{aligned} \tag{D.4}$$

where $\boldsymbol{\theta} = \begin{pmatrix} \boldsymbol{\theta}_1 \\ \vdots \\ \boldsymbol{\theta}_{M_c} \end{pmatrix}$ and $\boldsymbol{\theta}_j$ contains the parameters for each component, *i.e.* mean and variance. As with Eq. (3.15),

$$p(\boldsymbol{\psi}_i \mid \boldsymbol{\theta}) = \sum_{j=1}^{M_c} p(\boldsymbol{\psi}_i \mid \boldsymbol{\theta}_j) P(z_i = j \mid \boldsymbol{\theta}) . \tag{D.5}$$

Form the function

$$\begin{aligned} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(i_t)}) &\triangleq E\left[\sum_{i=1}^{N} \ln\big(p(\boldsymbol{\xi}_i \mid \boldsymbol{\theta}) \big) \right] \\ &= \sum_{j=1}^{M_c} \sum_{i=1}^{N} \Big[\ln\big(p(\boldsymbol{\psi}_i, z_i \mid \boldsymbol{\theta}) \big) \Big] P(z_i = j \mid \boldsymbol{\psi}_i, \boldsymbol{\theta}^{(i_t)}), \end{aligned} \tag{D.6}$$

where $\boldsymbol{\theta}^{(i_t)}$ contains the parameters for each i_t-th iteration (do not confuse i_t with i), and $\sum_{i=1}^{N} \ln\big(p(\boldsymbol{\xi}_i \mid \boldsymbol{\theta}) \big)$ is the log-likelihood, similar to Eq. (3.17). Define

$$P^{(i_t)}(\omega_j \mid \boldsymbol{\psi}_i) \triangleq P(z_i = j \mid \boldsymbol{\psi}_i, \boldsymbol{\theta}^{(i_t)}), \tag{D.7}$$

where $P^{(i_t)}(\omega_j \mid \boldsymbol{\psi}_i)$ is the expected posterior distribution of the class labels given the observed data at the i_t-th iteration. Substituting Eq. (D.4) and Eq. (D.7) into Eq. (D.6),

$$Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(i_t)}) = \sum_{j=1}^{M_c} \sum_{i=1}^{N} \Big[\ln\big(p(\boldsymbol{\psi}_i \mid \boldsymbol{\theta}_j) \big) + \ln\big(P(\omega_j) \big) \Big] P^{(i_t)}(\omega_j \mid \boldsymbol{\psi}_i) . \tag{D.8}$$

Note that $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(i_t)})$ is a function of the parameters $P(\omega_j)$ and $\boldsymbol{\theta}_j$, while $P^{(i_t)}(\omega_j)$ and $\boldsymbol{\theta}_j^{(i_t)}$ are fixed values.

The calculation of Q is the "expectation" step of the algorithm. To compute the new set of parameter values, $\boldsymbol{\theta}^{(i_t+1)}$, we optimize $Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(i_t)})$, *i.e.*

$$\boldsymbol{\theta}^{(i_t+1)} = \arg\max_{\boldsymbol{\theta}} Q(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(i_t)}) . \tag{D.9}$$

This is the "maximization" step of the algorithm.

GLOSSARY

Definitions in this glossary follow those in Ref. [41-43] closely.

attractor

An invariant subset (see **invariant subset**) of the phase space which is reached asymptotically as time $t \to \infty$.

basin of attraction

In nonlinear **dissipative systems**, it is possible for more than one **attractor** to exist for a single parameter setting. Different initial conditions will evolve towards one or other of the co-existing attractors. The closure of the set of initial conditions which approaches a given attractor is called the basin of attraction of that attractor. The boundary between one basin of attraction and another is called the basin boundary.

beamwidth

Angle subtended by beam (see Figure 1.1).

box-counting dimension

The box-counting dimension D_0 of a set U is defined as

$$D_0 \triangleq \lim_{\varepsilon \to 0^+} \sup \frac{\log N(\varepsilon)}{\log(1/\varepsilon)},$$

where $N(\varepsilon)$ is the number of balls of diameter ε required to cover U. See Section 2.4.

capacity dimension

Synonym for **box-counting dimension.**

capillary waves

Capillary waves are small waves (less than about 1.73cm); their velocity is determined mainly by the surface tension of water [1].

chaotic attractor

See **attractor**.

chaotic invariant

In this thesis, chaotic invariants (Section 2.4) are quantities like **box-counting dimension** and **Lyapunov** exponents. Chaotic invariants are unchanged under nonlinear changes of coordinate system [43].

continuous dependence on initial conditions

Let **F** be a function defined on the open set $U \in \mathbb{R}^D$, $D \in \mathbb{Z}^+$. Assume that **F** has **Lipschitz constant** L in the variable **v** on U. Let $\mathbf{v}(t)$ and $\mathbf{w}(t)$ be solutions of the differential equation $\dot{\mathbf{v}} = \mathbf{F}(\mathbf{v})$, and let $[t_0, t_1]$ be a subset of the domains of both solutions. Then continuous dependence on initial conditions [70] means

$$\|\mathbf{v}(t) - \mathbf{w}(t)\| \leq \|\mathbf{v}(t_0) - \mathbf{w}(t_0)\| e^{L(t-t_0)},$$

$\forall t \in [t_0, t_1]$.

correlation dimension

The correlation dimension, D_2, is defined as

$$D_2 \triangleq \lim_{r \to 0} \frac{\log\big(C(r)\big)}{\log(r)},$$

where $C(r)$ is the correlation sum. The correlation sum is in turn defined as

$$C(r) \triangleq \frac{1}{N^{\boldsymbol{\Psi}}(N^{\boldsymbol{\Psi}}-1)} \sum_{n=1}^{N^{\boldsymbol{\Psi}}} \sum_{n_2=1, n \neq n_2}^{N^{\boldsymbol{\Psi}}} u\left(r - \left\|\boldsymbol{\Psi}(n) - \boldsymbol{\Psi}(n_2)\right\|\right),$$

where $r \in \mathbb{R}$, n and n_2 are dummy variables, $N^{\boldsymbol{\Psi}}$ is the number of embedding vectors and $u(\bullet)$ is the step function. See Section 2.4.2 for details.

cross validation

A method of evaluating parameters or classifiers by dividing the training set into several parts, and in turn using one part to test the function fitted to the remaining parts.

curse of dimensionality

Essentially, the curse of dimensionality [148] refers to the exponential growth of hypervolume as a function of dimensionality. Consider function approximation of the following process $y = f(\boldsymbol{\psi})$, where $\boldsymbol{\psi} \in \mathbb{R}^p$. In order to approximate the function $f(\bullet)$ from the data, the whole p dimensional input space must be covered with data samples. Suppose for a unit interval, n samples are required to cover the interval, then in p dimension, n^p data points are needed [114]. Thus the number of data points required increase exponentially.

degree of freedom

The number of independent coordinates necessary to describe the position and momentum of a system in Euclidean space.

design set

During cross validation, this is part of the training set which is used as a training set. The purpose is to tune the hyperparameters of the neural network, e.g. number of centers, based on performance.

dissipative system

A dissipative system is one whereby the total energy is not conserved. In the long term, the system converges onto the attractor, and transients can be ignored. In a dissipative system, the sum of Lyapunov exponents is less than 0.

embedding delay

The lag (integer) between 2 consecutive elements of the embedding vector is called the embedding delay. See Section 2.2 for details.

embedding dimension

The (integer) dimension of phase space required to unfold the attractor of a nonlinear system from the observation of scalar signals from the source. See Section 2.3 for details.

Expectation-Maximization (EM) algorithm

The EM algorithm is an algorithm which uses maximum likelihood techniques to estimate missing features [89] (see Appendix D for details).

false nearest neighbours

If an attractor is projected onto a space which is too low dimensional to unfold it completely, some points will be projected near to each other, although they are not originally close together. These points are called false nearest neighbours.

fractal dimension

If the **box-counting dimension** of the set U is non-integer, it is said to be fractal.

generalization

A measure of the ability of a neural network to perform well on unseen or future examples. Alternatively, such a measure applied to a method to design neural networks. The term originates from psychology and refers to the ability to infer the correct structure from examples.

gravity waves

Long wavelength water waves. Gravity waves are so named because their velocity of propagation is determined primarily by gravity [1].

grazing angle

The angle between the land or sea surface and the radar signal is called the grazing angle (see Figure 1.1).

Gronwall inequality

Nearby solutions can diverge no faster than an exponential rate determined by the **Lipschitz constant** of the differential equation [70]. The Gronwall inequality is related to the **continuous dependence on initial conditions**.

Hamiltonian system

A Hamiltonian system is one whereby the total energy is conserved. It is so named because their time evolution can be described by Hamilton's equations. There are no attractors in a Hamiltonian system, and the sum of Lyapunov exponents is 0.

HF-band

Electromagnetic radiation of 3-30MHz.

hyperparameter

A parameter which is adjusted in **cross validation**, which is not one of the weights of the neural network. For example, the number of neurons in the hidden layer, M_c, or the regularization parameter γ.

intermittency

Occurrence of fluctuations that alternate 'randomly' between long periods of regular behaviour and relatively short irregular bursts.

invariant subset

Let $U \subset \mathbb{R}^D$, $D \in \mathbb{Z}^+$ and $f : \mathbb{R}^D \to \mathbb{R}^D$. The set U is an invariant set if $f(U) = U$.

K_u-band

Electromagnetic radiation of 8-12GHz.

Lipschitz

Let **F** be a function defined on the open set $U \in \mathbb{R}^D$, $D \in \mathbb{Z}^+$. **F** is said to be Lipschitz on U if there exists a constant $L < \infty$ such that $\|\mathbf{F}(\mathbf{v}) - \mathbf{F}(\mathbf{w})\| \leq L\|\mathbf{v} - \mathbf{w}\|$, $\forall \mathbf{v}, \mathbf{w} \in U$. The constant L is called a **Lipschitz constant** for **F** [70].

Lipschitz constant

See **Lipschitz**.

Lyapunov exponent

The rate at which nearby orbits diverge from each other after small perturbations when the evolution of a nonlinear system is chaotic. See Section 2.4.3.

metric space

A metric space $\mathbb{M}$ is a set, together with a distance function $d : \mathbb{M} \times \mathbb{M} \to \mathbb{R}$ which satisfies the following conditions:

1. $d(x, y) \geq 0$,
2. $d(x, y) = 0$ if and only if $x = y$,
3. $d(x, y) = d(y, x)$,
4. $d(x, y) + d(y, z) \geq d(x, z)$,

where $\forall x, y, z \in \mathbb{M}$. An example of a metric space is a finite dimensional vector space with the Euclidean norm as the distance function.

multifractals

Multiscale, non-uniform fractals are called multifractals.

orbit

Consider a map $f:U\to U$. Then the orbit is the set $\{y, f(y), f^2(y), \cdots\}$, where $y\in U$ and denotes f^{ς} denotes the ς-th iteration of map *f*. A **trajectory** generated by differential equations is also referred to as an orbit.

phase portrait

The phase portrait is a plot in **phase space** of the **orbit** evolution.

phase space

Consider the system such that $\frac{d\boldsymbol{\Psi}}{dt}=f(\boldsymbol{\Psi})$, where $\boldsymbol{\Psi}\in U\subset\mathbb{R}^D$ and $D\in\mathbb{Z}^+$. The phase space is the set *U*.

radial basis function (RBF)

The RBF is a function approximation method which uses a weighted sum of nonlinear functions. See Chapter 3 for details.

Runge-Kutta method

A numerical technique which can be used to solve ODEs (See Chapter 16 of Ref. [103] for details).

sea spikes

Large amplitude radar echo.

Self Organizing Map (SOM)

The SOM is a biologically inspired neural network able to perform a nonlinear mapping from a high dimension to a lower dimension. Each neuron is initialized to a random weight vector **w**. Initialize the learning rate α to 1.

Each iteration i_t, the neuron whose weight vector matches the input vector **x** most closely is chosen as the winner. The winning unit and neighbouring units update their weights according to the formula $\mathbf{w}(i_t+1) = \mathbf{w}(i_t) + \alpha(i_t)h(i_t)\left[\mathbf{x} - \mathbf{w}(i_t)\right]$, where $h(\bullet)$, the neighbouring function is typically a Gaussian function which gradually becomes narrower. Initially, α decreases rapidly as the SOM organizes itself, but in the second phase, α decreases slowly for final convergence.

sensitive dependence on initial conditions

Let f be a map on a **metric space** $\mathbb{M}$; one criterion for f to be defined as chaotic is sensitive dependence on initial conditions. Mathematically [149], f possesses sensitive dependence on initial conditions, if there exists $\varepsilon \in \mathbb{R}^+$ such that for any $y_0 \in \mathbb{M}$, and any open set $U \subset \mathbb{M}$ containing y_0, there exists $y \in U$ and $\varsigma \in \mathbb{Z}^+$ such that $\left\| f^{\varsigma}(y) - f^{\varsigma}(y_0) \right\| > \varepsilon$. This means that no matter how precisely an initial condition is defined, there are nearby states which eventually diverge from it.

state space

Synonym for **phase space.**

Taken's embedding theorem

Taken's embedding theorem states that it is possible to reconstruct state space from a time series of measurements. See Section 2.1 for details.

test set

A set of examples used only to assess the performance (generalization) of a fully-specified neural network.

training set

A set of examples used for learning, *i.e.* to fit the weights of the neural network.

trajectory

Consider an ODE, $\frac{d\boldsymbol{\Psi}}{dt} = f(\boldsymbol{\Psi})$, where $\boldsymbol{\Psi} \in \mathbb{R}^D$, $D \in \mathbb{Z}^+$. The solution $\boldsymbol{\Psi}(t)$, from a given initial condition $\boldsymbol{\Psi}_0 = \boldsymbol{\Psi}(t_0)$, plotted in **phase space** is called a trajectory or **orbit**.

transitivity

One criterion for a system to be defined as chaotic, is transitivity [149]. Let f be a map on a **metric space** $\mathbb{M}$. Then f is topologically transitive if for any pair of nonempty open sets $U \subset \mathbb{M}$ and $V \subset \mathbb{M}$, there exists $\varsigma \in \mathbb{Z}^+$ such that $f^{\varsigma}(U) \cap V \neq \varnothing$.

Intuitively, under a transitive map, a point wanders all over the space $\mathbb{M}$ where its orbit gets arbitrarily close to every point in $\mathbb{M}$.

validation set

During cross validation, this is part of the training set which is used as a test set. The purpose is to tune the hyperparameters of the neural network, e.g. number of centers, based on performance.

VHF-band

Electromagnetic radiation of 30-300MHz.

X-band

Electromagnetic radiation of 4-8GHz.