The School of Mathematics

THE UNIVERSITY
of EDINBURGH

Surrogate Modelling of Chaotic Dynamical Systems

by

Alex Kneale

Dissertation Presented for the Degree of
MSc in Computational Applied Mathematics

August 2024

Supervised by
Dr Martin T. Brolly

Abstract

In this thesis, deterministic and probabilistic surrogate machine learning models for chaotic dynamical systems are constructed and evaluated. The chaotic systems examined are the Lorenz 63 and Lorenz 96 toy model systems. We will primarily construct Markovian or 'memoryless' deterministic and probabilistic surrogate models. The first systems we attempt to model are 'fully observed', in the sense that they rely on a full description of the system dynamics. Here, we find that both the deterministic and probabilistic surrogate models perform equally well. Then, Markovian deterministic and probabilistic surrogate models for 'partially observed' chaotic systems are constructed, which rely on some reduced-order description of the dynamics. The story here is quite different. When the scale separation between the observed and unobserved variables in the system is large, both deterministic and probabilistic Markovian models perform well. However, as the scale separation is decreased, the models tend to perform less well. That said, the deterioration in model performance with decreasing scale separation is much more severe for the deterministic models than it is for the probabilistic models. Thus, while deterministic models struggle to represent the statistics of a partially observed system where the effects of unobserved dynamics are significant, the probabilistic models perform adequately well. In other words, the stochastic nature of the probabilistic models can represent the unknown effects which the dynamics of the unobserved component of the system have on the observed component. That said, even though the probabilistic models perform substantially better than the deterministic models, they do not perform perfectly. In an attempt to improve upon these results, and motivated by ideas from Takens' embedding theorem, delay-embedded deterministic and probabilistic models are constructed which use the past and present states to predict the future state. While these delay-embedded models perform better than their Markovian counterparts in some respects, we find that constructing accurate delay-embedded models is non-trivial task. As these models work in a substantially larger input-space, consisting of past and present states of the system, finding the weights which minimize the loss function of the model becomes a more complex optimization problem. This added complexity in the optimization of the loss function leads to instability in the model results.

Acknowledgments

I would like to thank Martin, my supervisor, for helping me throughout this project.

Own Work Declaration

I hereby declare that this report is my own work.

Contents

1 Introduction **1**

2 Background Knowledge **2**
2.1 Lorenz 63 . . . 2
2.2 Lorenz 96 . . . 2
2.3 Takens' Embedding Theorem . . . 4
2.4 Markovianity Assumption . . . 6

3 Methodology **7**
3.1 Conditional Modeling . . . 7
3.2 Mixture Density Networks . . . 7
3.3 General Structure of Deterministic and Probabilistic Models . . . 9
3.4 Collecting Training Data . . . 11
3.5 Model Generated Trajectories . . . 12
3.6 Truth Dataset . . . 13
3.7 Evaluating Models . . . 13

4 Fully Observed System Setting **15**
4.1 Lorenz 63 . . . 16
4.2 Monoscale Lorenz 96 . . . 18
4.3 Comments . . . 20

5 Partially Observed System Setting **21**
5.1 Markovian Models . . . 21
5.1.1 Large Scale Separation . . . 21
5.1.2 Medium Scale Separation . . . 23
5.1.3 Small Scale Separation and Large Coupling Term . . . 26
5.1.4 Comments . . . 29
5.2 Delay-embedded Models . . . 29
5.2.1 Single Time Delay-embedded Model . . . 29
5.2.2 Double Time Delay-embedded Model . . . 31

6 Discussion **33**
6.1 Overview of Results . . . 33
6.2 Evaluation of Measures . . . 34
6.3 Strengths and Limitations . . . 34
6.4 Conclusions and Further Study . . . 34

Appendices **38**

A Lorenz 63 Data **38**
A.1 Deterministic Model . . . 38
A.2 ML Model . . . 41
A.3 MD Model . . . 44
A.4 SL Model . . . 47
A.5 SD Model . . . 50

B Monoscale Lorenz 96 **53**
B.1 Deterministic Model . . . 53
B.2 ML Model . . . 55
B.3 MD Model . . . 57
B.4 SL Model . . . 59
B.5 SD Model . . . 61

C Multiscale Lorenz 96, $h = 1$, $b = c = 10$, and $F = 20$ **63**
C.1 Deterministic Model 63
C.2 ML Model 65
C.3 MD Model 67
C.4 SL Model 69
C.5 SD Model 71

D Multiscale Lorenz 96, $h = 1$, $b = 10$, $c = 4$, and $F = 20$ **73**
D.1 Deterministic Model 73
D.2 ML Model 75
D.3 MD Model 77
D.4 SL Model 79
D.5 SD Model 81

E Multiscale Lorenz 96, $h = 5$, $b = 10$, $c = 2$, and $F = 20$ **83**
E.1 Deterministic Model 83
E.2 ML Model 85
E.3 MD Model 87
E.4 SL Model 89
E.5 SD Model 91

F Single time delay-embedded model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$, and $F = 20$ **93**
F.1 Deterministic Model 93
F.2 ML model 95
F.3 MD model 97
F.4 SL model 99
F.5 SD model 101

List of Tables

1 Kullback-Leibler divergence calculations for distributions of x, y and z in the Lorenz 63 system. . . . 17
2 Hellinger distance calculations for distributions of x, y and z in the Lorenz 63 system. . . . 17
3 Difference in the truth and model generated covariance matrix and temporal autocorrelation function, Lorenz 63 system. . . . 17
4 Log and energy scores, Lorenz 63 system. . . . 18
5 Mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during the generation of the model trajectories, Lorenz 63 system. . . . 19
6 Kullback-Leibler and Hellinger calculations, monoscale Lorenz 96 system. . . . 20
7 Temporal and spatial autocorrelation calculations, monoscale Lorenz 96 system. . . 20
8 Log and energy scores, monoscale Lorenz 96 system. . . . 20
9 Mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during the generating of the model trajectories, monoscale Lorenz 96 system. . . . 21
10 Kullback-Leibler divergence and Hellinger distance, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$ and $F = 20$. . . . 22
11 Temporal and spatial autocorrelation function calculations, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$ and $F = 20$. . . . 23
12 Log and energy scores, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$ and $F = 20$. . . . 23
13 Mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during generation of the model trajectories, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$ and $F = 20$. . . . 23
14 Kullback-Leibler divergence and Hellinger distance, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. . . . 24
15 Temporal and spatial autocorrelation calculations, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. . . . 25
16 Log and energy scores, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. . . . 25
17 Mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during the generation of the model trajectories, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. . . . 25
18 Kullback-Leibler divergence and Hellinger distance calculations, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. . . . 28
19 Temporal and spatial autocorrelation function calculations, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. . . . 28
20 Log and energy scores, multiscale Lorenz 96 with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 28
21 Mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during the generation of the model trajectories, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. . . . 28
22 Kullback-Leibler divergence and Hellinger distance calculations, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$, single time delay-embedded models. . . . 30
23 Temporal and spatial autocorrelation calculations, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$, single time delay-embedded models. . . . 31
24 Log and energy scores, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$, single time delay-embedded models. . . . 31
25 Mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during the generation of the model trajectories, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$, single time delay-embedded models. . . . 32

List of Figures

1 Attractor of the Lorenz 63 system. 2
2 Diagram of the physical interpretation of monoscale Lorenz 96 system. Figure obtained from [29]. 3
3 Diagram of multiscale Lorenz 96 system. Each of the x_j variables are coupled to $y_{j,k}$ variables, $k \in \{1, ..., 32\}$. Figure obtained from [2]. 4
4 MDN in one dimension, consisting of 3 Gaussian components. $N(1, 2)$ denotes a Gaussian distribution with mean 1 and variance 2. 8
5 Typical training and test error curves during training of a model. The test error decreases steadily, before it begins to increase due to overfitting 8
6 Diagram of neural network structure of Deterministic models. 10
7 Diagram of neural network structure of probabilistic MDNs. 10
8 Diagram of process of generating trajectories in models. 12
9 An illustration of how KDE is constructed in 1-dimension. Here there are six observations, indicated by the black lines. Each observation is then smoothed into small bumps (red curves above), which are summed together to obtain a density estimator (blue curve). Figure obtained from [6]. 14
10 x_i trajectory, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 26
A.1.1 Loss function when training Deterministic model, Lorenz 63 system. 38
A.1.2 Distribution function of x inferred from Deterministic model, Lorenz 63 system. 38
A.1.3 Distribution function of y inferred from Deterministic model, Lorenz 63 system. 39
A.1.4 Distribution function of z inferred from Deterministic model, Lorenz 63 system. 39
A.1.5 Temporal autocorrelation function of x inferred from Deterministic model, Lorenz 63 system. 39
A.1.6 Temporal autocorrelation function of y inferred from Deterministic model, Lorenz 63 system. 40
A.1.7 Temporal autocorrelation function of z inferred from Deterministic model, Lorenz 63 system. 40
A.2.1 Loss function when training ML model, Lorenz 63 system. 41
A.2.2 Distribution function of x inferred from ML model, Lorenz 63 system. 41
A.2.3 Distribution function of y inferred from ML model, Lorenz 63 system. 42
A.2.4 Distribution function of z inferred from ML model, Lorenz 63 system. 42
A.2.5 Temporal autocorrelation function of x inferred from ML model, Lorenz 63 system. 42
A.2.6 Temporal autocorrelation function of y inferred from ML model, Lorenz 63 system. 43
A.2.7 Temporal autocorrelation function of z inferred from ML model, Lorenz 63 system. 43
A.3.1 Loss function when training MD model, Lorenz 63 system. 44
A.3.2 Distribution function of x inferred from MD model, Lorenz 63 system. 44
A.3.3 Distribution function of y inferred from MD model, Lorenz 63 system. 45
A.3.4 Distribution function of z inferred from MD model, Lorenz 63 system. 45
A.3.5 Temporal autocorrelation function of x inferred from MD model, Lorenz 63 system. 45
A.3.6 Temporal autocorrelation function of y inferred from MD model, Lorenz 63 system. 46
A.3.7 Temporal autocorrelation function of z inferred from MD model, Lorenz 63 system. 46
A.4.1 Loss function when training SL model, Lorenz 63 system. 47
A.4.2 Distribution function of x inferred from SL model, Lorenz 63 system. 47
A.4.3 Distribution function of y inferred from SL model, Lorenz 63 system. 48
A.4.4 Distribution function of z inferred from SL model, Lorenz 63 system. 48
A.4.5 Temporal autocorrelation function of x inferred from SL model, Lorenz 63 system. 48
A.4.6 Temporal autocorrelation function of y inferred from SL model, Lorenz 63 system. 49
A.4.7 Temporal autocorrelation function of z inferred from SL model, Lorenz 63 system. 49
A.5.1 Loss function when training SD model, Lorenz 63 system. 50
A.5.2 Distribution function of x inferred from SD model, Lorenz 63 system. 50

A.5.3 Distribution function of y inferred from SD model, Lorenz 63 system. 51
A.5.4 Distribution function of z inferred from SD model, Lorenz 63 system. 51
A.5.5 Temporal autocorrelation function of x inferred from SD model, Lorenz 63 system. 51
A.5.6 Temporal autocorrelation function of y inferred from SD model, Lorenz 63 system. 52
A.5.7 Temporal autocorrelation function of z inferred from SD model, Lorenz 63 system. 52
B.1.1 Loss function when training Deterministic model, monoscale Lorenz 96 system. 53
B.1.2 Distribution function of x_i inferred from Deterministic model, monoscale Lorenz 96 system. 53
B.1.3 Spatial autocorrelation function inferred from Deterministic model, monoscale Lorenz 96 system. 54
B.1.4 Temporal autocorrelation function inferred from Deterministic model, monoscale Lorenz 96 system. 54
B.2.1 Loss function when training ML model, monoscale Lorenz 96 system. 55
B.2.2 Distribution function of x_i inferred from ML model, monoscale Lorenz 96 system. 55
B.2.3 Spatial autocorrelation function inferred from ML model, monoscale Lorenz 96 system. 56
B.2.4 Temporal autocorrelation function inferred from ML model, monoscale Lorenz 96 system. 56
B.3.1 Loss function when training MD model, monoscale Lorenz 96 system. 57
B.3.2 Distribution function of x_i inferred from MD model, monoscale Lorenz 96 system. 57
B.3.3 Spatial autocorrelation function inferred from MD model, monoscale Lorenz 96 system. 58
B.3.4 Temporal autocorrelation function inferred from MD model, monoscale Lorenz 96 system. 58
B.4.1 Loss function when training SL model, monoscale Lorenz 96 system. 59
B.4.2 Distribution function of x_i inferred from SL model, monoscale Lorenz 96 system. 59
B.4.3 Spatial autocorrelation function inferred from SL model, monoscale Lorenz 96 system. 60
B.4.4 Temporal autocorrelation function inferred from SL model, monoscale Lorenz 96 system. 60
B.5.1 Loss function when training SD model, monoscale Lorenz 96 system. 61
B.5.2 Distribution function of x_i inferred from SD model, monoscale Lorenz 96 system. 61
B.5.3 Spatial autocorrelation function inferred from SD model, monoscale Lorenz 96 system. 62
B.5.4 Temporal autocorrelation function inferred from SD model, monoscale Lorenz 96 system. 62
C.1.1 Loss function for Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$, $F = 20$. 63
C.1.2 Distribution function of x_i inferred from Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$, $F = 20$. 63
C.1.3 Spatial autocorrelation function inferred from Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$, $F = 20$. 64
C.1.4 Temporal autocorrelation function inferred from Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$, $F = 20$. 64
C.2.1 Loss function for ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 65
C.2.2 Distribution function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 65
C.2.3 Spatial autocorrelation function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 66
C.2.4 Temporal autocorrelation function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 66

C.3.1 Loss function for MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 67
C.3.2 Distribution function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 67
C.3.3 Spatial autocorrelation function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 68
C.3.4 Temporal autocorrelation function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 68
C.4.1 Loss function for SL model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 69
C.4.2 Distribution function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 69
C.4.3 Spatial autocorrelation function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 70
C.4.4 Temporal autocorrelation function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 70
C.5.1 Loss function for SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 71
C.5.2 Distribution function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 71
C.5.3 Spatial autocorrelation function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 72
C.5.4 Temporal autocorrelation function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 72
D.1.1 Loss function for Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 73
D.1.2 Distribution function of x_i inferred from Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$. 73
D.1.3 Spatial autocorrelation function inferred from Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = 10$ $c = 4$ and $F = 20$. 74
D.1.4 Temporal autocorrelation function inferred from Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 74
D.2.1 Loss function for ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 75
D.2.2 Distribution function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 75
D.2.3 Spatial autocorrelation function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 76
D.2.4 Temporal autocorrelation function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 76
D.3.1 Loss function for MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 77
D.3.2 Distribution function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 77
D.3.3 Spatial autocorrelation function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 78
D.3.4 Temporal autocorrelation function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 78
D.4.1 Loss function for SL model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 79
D.4.2 Distribution function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 79

D.4.3 Spatial autocorrelation function of x_i inferred from SL Model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 80
D.4.4 Temporal autocorrelation function of x_i inferred from SL Model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 80
D.5.1 Loss function for SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 81
D.5.2 Distribution function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 81
D.5.3 Spatial autocorrelation function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 82
D.5.4 Temporal autocorrelation function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$. 82
E.1.1 Loss function for Deterministic model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 83
E.1.2 Distribution function of x_i inferred from Deterministic model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 83
E.1.3 Spatial autocorrelation function inferred from Deterministic model, multiscale Lorenz 96 system with $h = 5$, $b = 10$ $c = 2$, $F = 20$. 84
E.1.4 Temporal autocorrelation function inferred from Deterministic model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$, $F = 20$. 84
E.2.1 Loss function for ML model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 85
E.2.2 Distribution function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 85
E.2.3 Spatial autocorrelation function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 86
E.2.4 Temporal autocorrelation function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 86
E.3.1 Loss function for MD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 87
E.3.2 Distribution function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 87
E.3.3 Spatial autocorrelation function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 88
E.3.4 Temporal autocorrelation function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 88
E.4.1 Loss function for SL model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 89
E.4.2 Distribution function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 89
E.4.3 Spatial autocorrelation function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 90
E.4.4 Temporal autocorrelation function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 90
E.5.1 Loss function for SD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 91
E.5.2 Distribution function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 91
E.5.3 Spatial autocorrelation function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 92
E.5.4 Temporal autocorrelation function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 92

F.1.1 Loss function for single time delay-embedded deterministic model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 93
F.1.2 Distribution function of x_i inferred from single time delay-embedded deterministic model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 93
F.1.3 Spatial autocorrelation function inferred from single time delay-embedded deterministic model for multiscale Lorenz 96 system with $h = 5$, $b = 10$ $c = 2$, $F = 20$. 94
F.1.4 Temporal autocorrelation function inferred from single time delay-embedded deterministic model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$, $F = 20$. 94
F.2.1 Loss function for single time delay-embedded ML model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 95
F.2.2 Distribution function of x_i inferred from single time delay-embedded ML model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 95
F.2.3 Spatial autocorrelation function of x_i inferred from single time delay-embedded ML model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 96
F.2.4 Temporal autocorrelation function of x_i inferred from single time delay-embedded ML model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 96
F.3.1 Loss function for single time delay-embedded MD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 97
F.3.2 Distribution function of x_i inferred from single time delay-embedded MD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 97
F.3.3 Spatial autocorrelation function of x_i inferred from single time delay-embedded MD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 98
F.3.4 Temporal autocorrelation function of x_i inferred from single time delay-embedded MD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 98
F.4.1 Loss function for single time delay-embedded SL model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 99
F.4.2 Distribution function of x_i inferred from single time delay-embedded SL model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 99
F.4.3 Spatial autocorrelation function of x_i inferred from single time delay-embedded SL model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 100
F.4.4 Temporal autocorrelation function of x_i inferred from single time delay-embedded SL model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 100
F.5.1 Loss function for single time delay-embedded SD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 101
F.5.2 Distribution function of x_i inferred from single time delay-embedded SD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 101
F.5.3 Spatial autocorrelation function of x_i inferred from single time delay-embedded SD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 102
F.5.4 Temporal autocorrelation function of x_i inferred from from single time delay-embedded SD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$. 102

1 Introduction

In recent years, there has been an interest in using surrogate machine learning models to study chaotic dynamical systems [7, 1]. The deterministic nature of chaotic systems prompts the use of deterministic surrogate models. However, for high-dimensional systems, these models can often be prohibitively expensive to implement in practice. Thus, surrogate models which work with simplified and often lower-order approximations of the full dynamics of the chaotic system are preferred [19, 7, 1, 30]. Takens' embedding theorem [9] has motivated a range of literature investigating how to construct accurate lower-dimensional deterministic surrogate models [20, 15]. However, the fact that a reduced order space is a space which is lacking information motivates the use of a probabilistic approach to the surrogate model. Indeed, one would think that the stochastic nature of the probabilistic surrogate model could, in effect, encode the information that is lost when moving to a reduced order description of the dynamics.

In this thesis, we examine and compare the performance of deterministic and probabilistic surrogate machine learning models to describe the dynamics of chaotic systems. We examine three systems: the Lorenz 63, monoscale Lorenz 96 and multiscale Lorenz 96 systems. We attempt to model each of these systems using both deterministic and probabilistic surrogate models. As the state spaces of the first two systems are not too large, we attempt to model them using full descriptions of the dynamics. More specifically, the performance of Markovian or 'memoryless' surrogate models, which predict the future state of the system given just the present state, are examined. The purpose of this experiment is to show that both deterministic and probabilistic Markovian surrogate models can reproduce the dynamics of a fully observed chaotic system fairly well. From now on, this regime will be referred to as the 'fully observed system setting', as indicated in Section 4.

As the state space of the multiscale Lorenz 96 system is prohibitively large, here we attempt to construct both deterministic and probabilistic Markovian surrogate models which rely on a lower-order representation of the dynamics. In this example, the lower-order representation of the dynamics takes the form of observing only some of the variables in the system. For this reason, this regime will be referred to as the 'partially observed system setting', as indicated in Section 5. The goal of this exercise is twofold. First, we want to see how the deterministic Markovian models hold up in a partially observed system setting. Second, we are investigating how the stochastic nature of the probabilistic Markovian model can represent the effects that the unobserved variables of the system have on the observed ones. The idea in mind is that the probabilistic models can, in effect, encode the information that is lost when moving from a fully observed system setting to a partially observed system setting.

Finally, Takens' embedding theorem [9] has motivated the development of machine learning models which rely on descriptions of the 'delay-embedded' partially observed state space [20]. As opposed to Markovian models, delay-embedded models rely on descriptions of the current and past states to make predictions about the future state. In the final bit of our exploration, we examine the performance of delay-embedded probabilistic and deterministic surrogate models for the partially observed multiscale Lorenz 96 system.

In this thesis we restrict ourselves to developing models for toy systems which are nonphysical, i.e. they are not real-life systems found in nature. However, it is important to note that these toy systems have been used in literature as useful benchmarks to assess models [19, 16, 30, 2, 7]. Therefore, the models we develop here may have applications in modelling chaotic systems which are found in nature.

2 Background Knowledge

2.1 Lorenz 63

The Lorenz 63 system is a three-dimensional system which simulates two parallel two-dimensional fluid plates. We assume that the plates are infinite in length, and that the lower plate is heated while the upper plate is cooled. The Lorenz 63 system was derived from the Boussinesq approximation of the Rayleigh-Bénard model of convection [16, 29]. The equations for the system are given by

$$\dot{x} = -\sigma x + \sigma y \tag{2.1}$$
$$\dot{y} = -xz + rx - y \tag{2.2}$$
$$\dot{z} = xy - bz. \tag{2.3}$$

While the physical interpretation of the variables and parameters in the equations are not relevant to the exploration in this thesis, we include them here out of general interest. $x \in \mathbb{R}$ is proportional to the intensity of the convective motion, $y \in \mathbb{R}$ to the difference between the ascending and descending components of the fluid and $z \in \mathbb{R}$ to the distortion of the temperature from linearity. The physical interpretations of the parameters are the following: $r \in \mathbb{R}$ is proportional to the scaled Rayleigh number of the the fluid, $\sigma \in \mathbb{R}$ to the Prandl number and $b \in \mathbb{R}$ to a physical dimension of the fluid layer. A common choice of parameter settings which ensures the system is chaotic is the following: $r = 28$, $b = \frac{8}{3}$ and $\sigma = 10$. In this domain, the attractor of the chaotic system is the famous 'butterfly wings', displayed in Fig. 1. In this attractor, trajectories spiral around the fixed points $(x, y, z) = (\pm\sqrt{b(r-1)}, \pm\sqrt{b(r-1)}, r-1)$, creating the wing-like shapes.

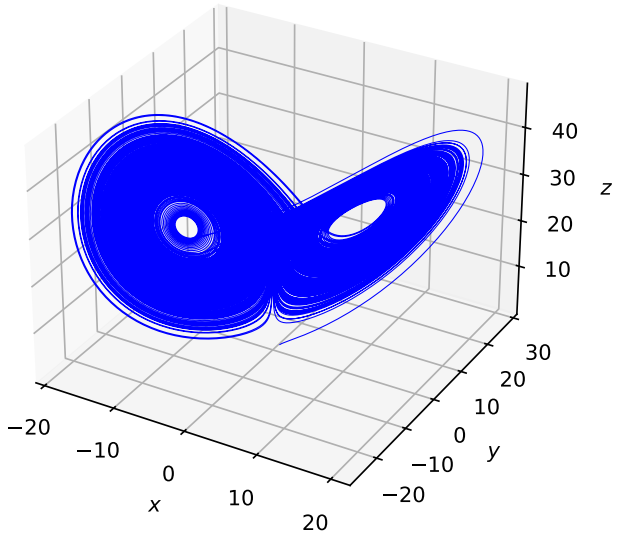

Figure 1: Attractor of the Lorenz 63 system.

2.2 Lorenz 96

The Lorenz 96 system was designed to study 'error' growth in the atmosphere. Here, 'error' refers to either the difference between the system's assumed initial state and its actual state, or the

difference between the future state predicted by the model and the actual future state. Like many other models designed by Lorenz, it was created by simplifying models for the atmosphere. More specifically, an underlying 'physical' model was simplified to create a model which still captures qualitative features of the atmosphere. Thus, although the Lorenz 96 system is nonphysical, it is still useful for gaining insights into the nature of atmospheric error growth [29].

There are two versions of the Lorenz 96 system: a monoscale version with only one timescale, and a multiscale version with two different timescales which is obtained by coupling two scaled versions of the monoscale version. The monoscale version consists of a system of dimension $n \in \mathrm{N}$, given by the set of equations

$$\dot{x}_j = x_{j-1}(x_{j+1} - x_{j-2}) - x_j + F, \qquad j = 1, ..., n \tag{2.4}$$

with 'boundary' conditions $x_{j-n} = x_{j+n} = x_j$ which results in circulant symmetry. Here, the variables $x_j \in \mathbb{R}$ of the system can be interpreted as values of some scalar atmospheric quantity, such as temperature or pressure, measured along a circle of constant latitude of the earth. The latitude circle is divided into n equal sectors, each with a unique x_j variable, so that $j = 1, ..., n$ indicates the longitude on the circle, as shown in Fig. 2. Therefore, if the variables x_j describe temperature in the atmosphere, x_2 would refer to the temperature at longitude unit 2 while x_5 would refer to the temperature at longitude unit 5, and so forth.

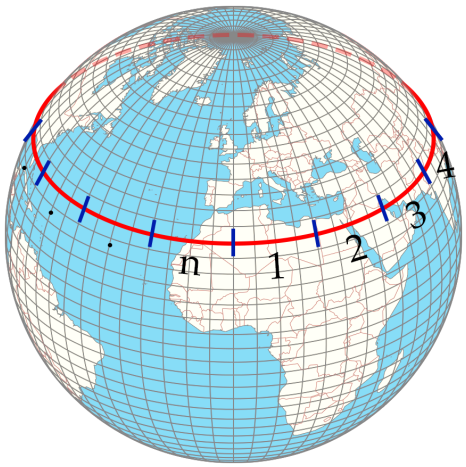

Figure 2: Diagram of the physical interpretation of monoscale Lorenz 96 system. Figure obtained from [29].

The set of Equations 2.4 above contain three characteristic processes of atmospheric systems: advection ($x_{j-1}(x_{j+1}-x_{j-2})$), dissipation ($-x_j$), and external forcing (F). Therefore, Equations 2.4 can be seen as modelling waves in the atmosphere [10]. Furthermore, the values of the system's two free parameters, n and F, drastically change its dynamics. Thus, the monoscale Lorenz 96 system is used in literature as a model problem for a variety of purposes. For example, the parameter settings $n = 40, 80, 120$ and $F = 8$ are commonly used for data assimilation [19], while $n = 8$ and $F = 18, 20$ are commonly used for stochastic parameterization [30].

The multiscale Lorenz 96 system is given by

$$\dot{x}_j = x_{j-1}(x_{j+1} - x_{j-2}) - x_j + F - \frac{hc}{b}\sum_{k=1}^{m} y_{j,k} \tag{2.5}$$

$$\dot{y}_{j,k} = cby_{j,k}(y_{j,k-1} - y_{j,k+1}) - cy_{j,k} + \frac{hc}{b}x_j \tag{2.6}$$

where $j = 1, .., n$ and $k = 1, ..., m$ and the variables satisfy 'boundary' conditions $x_{j-n} = x_{j+n} = x_j$, $y_{j-n,k} = y_{j+n,k} = y_{j,k}$, $y_{j,k-m} = y_{j-1,k}$ and $y_{j,k+m} = y_{j+1,k}$. The variables $x_j \in \mathbb{R}$ can be

interpreted in the same way as the monoscale version: values of some scalar atmospheric quantity measured along a circle of constant latitude of the earth. However, now for each x_j variable, we assume that there are m associated $y_{j,k} \in \mathbb{R}$ variables ($k = 1, ..., m$). Thus, looking at Equations 2.5 and 2.6, we can interpret the system as being composed of two coupled and scaled versions of the monoscale Lorenz 96 system. A diagram displaying the multiscale system for $n = 8$ and $m = 32$ is shown in Fig 3. The values of the parameters h, b and c determine the nature of the relationship between the two coupled systems. $h \in \mathbb{R}$ is the coupling constant, $b \in \mathbb{R}$ is the spatial-scale ratio and $c \in \mathbb{R}$ is the temporal-scale ratio. Thus, for $h = 1$, $b = c = 10$, the $y_{j,k}$ variables will tend to fluctuate 10 times more rapidly, but with 10 times smaller magnitude than the x_j variables, and the strength of coupling between the two systems is equal to 1 [30]. Thus, we can imagine our x_j variables as describing some large spatial scale phenomenon happening at a slow timescale. For each x_j, the $y_{j,k}$ ($k = 1, ..., m$) variables describe small spatial scale phenomena happening at a fast timescale, which are coupled to x_j. Thus, if x_j variables describe pressure, the $y_{j,k}$ variables may describe some small scale fluctuations in the atmosphere that affect pressure [7, 17].

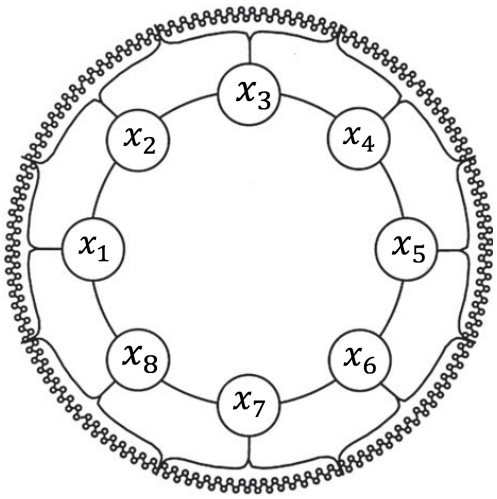

Figure 3: Diagram of multiscale Lorenz 96 system. Each of the x_j variables are coupled to $y_{j,k}$ variables, $k \in \{1, ..., 32\}$. Figure obtained from [2].

2.3 Takens' Embedding Theorem

Takens' embedding theorem provides a theoretical framework for reconstructing features of the phase space of a chaotic system, given a time series of measurements of a single observable. The proof and full description of this theorem are quite technical, and require a strong foundation in differential topology, which is outside of the scope of this thesis. Here, we will provide a brief description of parts of the theorem which are relevant to our investigation [9]. We will describe the theorem in the discrete time case, as even though the Lorenz 63 and 96 systems are continuous in time, to construct trajectories, integration schemes are used which, in effect, transform these systems from the continuous to the discrete time regime.

Suppose we have a discrete time system, defined by the mapping

$$\boldsymbol{z}^{t_{i+1}} = \phi(\boldsymbol{z}^{t_i}) \tag{2.7}$$

where $\phi : \mathbb{R}^d \to \mathbb{R}^d$ is a non-linear mapping, and its variables are defined in a d-dimensional state space $\boldsymbol{z}^{t_i} = (z_1^{t_i}, z_2^{t_i}, ..., z_d^{t_i})$ at some time $t = t_i$. We can assume that we have some initial state $\boldsymbol{z}^{t_0}$. Assume that, after a sufficiently large time all the states converge to some compact manifold $\nu \subseteq \mathbb{R}^d$. A manifold ν in $\mathbb{R}^d$ is defined as a topological space which is locally like $\mathbb{R}^d$. More formally, every point on ν has an open neighbourhood which is homeomorphic to an open

subset of $\mathbb{R}^d$ [9]. Two objects are homeomorphic if they can be deformed into each other by a continuous invertible mapping [12].

Now assume all the states converge to some attractor contained in manifold ν. Finally, assume that we can only observe one variable of the d-dimensional state space $\boldsymbol{z}^{t_i}$ at any given time t_i. Let us refer to this variable as $z_k^{t_i} \in \mathbb{R}$, where $z_k^{t_i}$ is one of the d variables $\{z_1^{t_i}, z_2^{t_i}, ..., z_d^{t_i}\}$. Put simply, Takens' embedding theorem states that the long-term data of one of these single variables, $z_k^{t_i}$, is enough to topologically reconstruct the attractor of the full high-dimensional system, $\nu \subseteq \mathbb{R}^d$, if certain conditions are satisfied. Thus, if we can only observe the value of $z_k^{t_i}$, say for a total number of times S, so that we record the timeseries $(z_k^{t_i}, z_k^{t_{i+1}}, z_k^{t_{i+2}}, ..., z_k^{t_{i+S-1}})$, then, if S is large enough, and other conditions are satisfied, we should be able to reconstruct the attractor ν.

To state Takens' embedding theorem in a more mathematically formal manner, we have to provide a few more definitions. First, let us define the box-counting dimension, b, of an attractor [18]. Assume we have a set which lies in m-dimensional Cartesian space. Now imagine covering the space with a grid of m-dimensional cubes of edge length ε. Thus, if $m = 1$, the cubes are intervals of length ε, whereas if $m = 2$, the cubes are squares of side length ε. Now we count the number of cubes, $N(\varepsilon)$, needed to cover the set. The box-counting dimension is then given by

$$b = \lim_{\varepsilon \to 0} \frac{\ln N(\varepsilon)}{\ln(1/\varepsilon)}.$$

Let us denote the box-counting dimension of our attractor ν as b. Now, let us assume we make an observation in $\mathbb{R}$ of the states at successive times $t_i, t_{i+1}, ...$ etc. This observation could, for example, be $z_k^{t_i}$. Say obtaining the observation from the full state of the system at time t_i, $\boldsymbol{z}^{t_i}$, is given by a differentiable function $y : \nu \to \mathbb{R}$. Therefore, the value of the observation at time t_i is given by $y(\boldsymbol{z}^{t_i})$. If our observation is $z_k^{t_i}$, then y is given by a left-multiplication of a matrix with zeros everywhere apart from a single one in the kth diagonal element, so that $y(\boldsymbol{z}^{t_i}) = z_k^{t_i}$. Finally, we must define an embedding. An embedding of a topological space X into a topological space Y, is a homeomorphism $f : X \to Y$ of X onto a subset of Y [23]. A homeomorphism is a continuous bijective (one-to-one) map, whose inverse is also continuous [23]. Now we can state a form of Takens' embedding theorem [9, 20].

Theorem 2.1 *Let ν be a compact manifold of an attractor with a box-counting dimension b. For a smooth diffeomorphism $\phi : \nu \to \nu$ and a smooth function $y : \nu \to \mathbb{R}$, there is a generic property which states that the mapping $\mathcal{F}_{\phi,y} : \nu \to \mathbb{R}^S$ is an embedding if $S > 2b$. Thus, $\mathcal{F}_{\phi,y}(\boldsymbol{z}) = [y(\boldsymbol{z}), y \circ \phi(\boldsymbol{z}), ..., y \circ \phi(\boldsymbol{z})^{S-1}]^T$ is an embedding if $S > 2b$. Returning to our original situation, letting $\boldsymbol{z} = \boldsymbol{z}^{t_i}$ and $y \circ \phi^j(\boldsymbol{z}^{t_i}) = z_k^{t_{i+j}}$, where $z_k^{t_{i+j}} \in \mathbb{R}$ is one of the d variables $\{z_1^{t_{i+j}}, z_2^{t_{i+j}}, ..., z_d^{t_{i+j}}\}$ and $j = 0, 1, ..., S-1$, the mapping above has the form $\mathcal{F}_{\phi,y}(\boldsymbol{z}^{t_i}) = [z_k^{t_i}, z_k^{t_{i+1}}, ..., z_k^{t_{i+S-1}}]^T$, and is an embedding of ν.*

We must note that Theorem 2.1 is a specific case of the theorem which is relevant to the topics discussed in this thesis. The implications of the theorem are the following: if one is examining a partially observed chaotic system (as in Section 5), then, assuming all conditions in Theorem 2.1 are satisfied, there exists a homeomorphism from the fully observed attractor to the partially observed setting. Thus, as homeomorphisms are invertible, it is in theory possible to go from the partially observed setting to the fully observed setting. From an information perspective, Theorem 2.1 states that, assuming a sufficiently long delay, the delay-embedded data of a single variable contains all the necessary information to reconstruct the full attractor of the system. In this thesis, we will not work directly with the theorem. We have included it here to provide some motivation for our discussion of the 'Markovianity Assumption' in Section 2.4. The discussion which follows from this section will be more relevant to our exploration than Takens' embedding theorem itself.

2.4 Markovianity Assumption

As we have seen, Takens' embedding theorem provides a theoretical framework for reconstructing the attractor of a system given the timeseries of a single variable of the state. Now we will talk about how the mapping of a discrete time system, as given in Equation 2.7, can be reconstructed from a partially observed timeseries [15]. Suppose we have a discrete time system as defined in Equation 2.7. Following notation in [15], we can say that this system is 'Markovian', in the sense that the next state, $\boldsymbol{z}^{t_{i+1}}$ depends only on the value of the current state $\boldsymbol{z}^{t_i}$. In other words, the system is memoryless. It is important to note, the term 'Markovian' is normally associated with memoryless stochastic processes. Here, we are simply extending the use of the term to refer to both deterministic and stochastic memoryless processes.

Now suppose that we can only observe some of the variables of the state. Let us refer to these observable variables as $\boldsymbol{x}^{t_{i+1}} \in \mathbb{R}^{d_x}$, and the variables we cannot observe as $\boldsymbol{y}^{t_{i+1}} \in \mathbb{R}^{d_y}$. The mapping in Equation 2.7 can be written equivalently as

$$\boldsymbol{x}^{t_{i+1}} = \phi_1(\boldsymbol{x}^{t_i}, \boldsymbol{x}^{t_i}) \tag{2.8}$$

$$\boldsymbol{y}^{t_{i+1}} = \phi_2(\boldsymbol{x}^{t_i}, \boldsymbol{y}^{t_i}). \tag{2.9}$$

Now assume ϕ_1 and ϕ_2 are also unknown. Suppose now that there is a scale separation between the two systems $\boldsymbol{x}^{t_i}$ and $\boldsymbol{y}^{t_i}$. Say the amount of scale separation is given by $\varepsilon > 0$, so that the scale separation is large when ε is small. Here we are purposefully keeping the meaning of the scale separation vague. When we look at the multiscale Lorenz 96 system in Section 5, its meaning will be more concrete. For now, we just imagine the scale separation as a measure of how much effect the unobserved variables have on the dynamics of the observed dynamics. When $\varepsilon \ll 1$ and the scale separation is large, the unobservable variables affect the dynamics of the observable dynamics less, whereas when the $\varepsilon \approx 1$ and the scale separation is small, the unobservable variables affect the dynamics of the observable dynamics more. Now we observe that, by considering the solution of Equation 2.9 as a function of the history of the variable $\boldsymbol{x}^t$, the influence of $\boldsymbol{y}^t$ on the solution $\boldsymbol{x}^t$ can be captured by a family of operators $m_t(\cdot)$, which takes as parameters the historical trajectory of $\boldsymbol{x}^t$, $\{\boldsymbol{x}^{t_s}\}_{s=0}^{i}$, and the unobserved initial condition $\boldsymbol{y}^{t_0}$, so Equation 2.8 can be written as

$$\boldsymbol{x}^{t_{i+1}} = \phi_{m_t}(\{\boldsymbol{x}^{t_s}\}_{s=0}^{i}, \boldsymbol{y}^{t_0}). \tag{2.10}$$

This is an example of a non-Markovian mapping, as $\boldsymbol{x}^{t_{i+1}}$ depends not only on the present observable state, $\boldsymbol{x}^{t_i}$, but on the past observable states as well. However, if the scale separation between the observable and unobservable states is small enough, then we can simplify the relation in Equation 2.10 to

$$\boldsymbol{x}^{t_{i+1}} = \phi_m(\boldsymbol{x}^{t_i}). \tag{2.11}$$

Here, we have said that if the effects of the unobservable state $\boldsymbol{y}^{t_i}$ on the dynamics of $\boldsymbol{x}^{t_i}$ are negligible, then we can simplify the non-Markovian model in Equation 2.10 with a Markovian model. A Markovian model is appealing, as it requires fewer parameters than the non-Markovian model. The relevance of this discussion of Markovianity to our exploration is the following. For most of the thesis, we work with Markovian deterministic and probabilistic surrogate models which rely on the state at time t_i to find the state at time t_{i+1}, like in Equation 2.11. However, in Section 5, we will be working in a partially-observed system setting, and we will look at one parameter setting in particular for which the Markovian models fail, due to the scale separation between the unobserved and observed variables being too small. Thus, here we resort to implementing models which rely on memory of the observable state, as in Equation 2.10, in the hope that they will perform better. We will refer to these models as 'delay-embedded' models. It is important to note, that from Equation 2.10, we do not expect these delay-embedded models to work perfectly, as we do not have access to the initial state of

the unobservable variable, $\boldsymbol{y}^{t_0}$.

3 Methodology

As stated in the introduction, the central goal of this thesis is to compare the performance of surrogate deterministic and probabilistic models for chaotic dynamical systems. For most of the exploration, the models we examine are Markovian, meaning that the current state at time t_i is used to find the state at time t_{i+1}. However, in Section 5.2 onwards, we will work with delay-embedded models, in which the present and (some) past states at times t_i, t_{i-1}, t_{i-2}, etc., are used to find the state at time t_{i+1}. The deterministic models assume the update rule of the system is deterministic, while the probabilistic models assume it is governed by a probability distribution function (PDF). The next few sections will describe in more detail how these two types of models are constructed. First, we will outline the theory which is necessary for the explanation of the probabilistic models.

3.1 Conditional Modeling

The goal of conditional modeling or conditional density estimation (CDE) is to infer the conditional probability density $p(\boldsymbol{Y}|\boldsymbol{X})$, where $\boldsymbol{X}$ and $\boldsymbol{Y}$ are two random variables. In this thesis, we restrict ourselves to the case that $\boldsymbol{X}$ and $\boldsymbol{Y}$ are continuous random variables. Unlike regression, which aims to model just the conditional means $\mathbb{E}[\boldsymbol{Y}|\boldsymbol{X}]$, CDE offers the advantage of attempting to gain knowledge about the variability of $\boldsymbol{Y}|\boldsymbol{X}$ [5].

In parametric conditional models, we assume that the distribution of $\boldsymbol{Y}|\boldsymbol{X}$ belongs to a certain family of parametric distributions, so that

$$p(\boldsymbol{Y}|\boldsymbol{X}) = \rho(\boldsymbol{Y}; \boldsymbol{\theta}(\boldsymbol{X})) \tag{3.1}$$

where $\rho(\cdot; \boldsymbol{\theta})$ is the probability density of a family of distributions parameterized by $\boldsymbol{\theta}$. When constructing a parametric conditional model, one must choose the form of ρ and $\boldsymbol{\theta}(\boldsymbol{X})$ [5].

3.2 Mixture Density Networks

A mixture density network (MDN) is a conditional model where $\boldsymbol{\theta}(\boldsymbol{X})$ in Equation 3.1 is represented by an artificial neural network and $\rho(\cdot\,; \boldsymbol{\theta})$ is given by a mixture distribution [4, 3]. The density function of general mixture distribution consisting of N_c components is given by

$$\rho(\cdot\,; \boldsymbol{\theta}) = \sum_{j=1}^{N_c} \alpha_j \rho_j(\cdot\,; \boldsymbol{\theta}_j)$$

where $\rho_j(\cdot\,; \boldsymbol{\theta}_j)$ is the density of the jth component, with parameters $\boldsymbol{\theta}_j$. Furthermore, $\boldsymbol{\theta} = [(\alpha_1, \boldsymbol{\theta}_1), \ldots, (\alpha_{N_c}, \boldsymbol{\theta}_{N_c})]$ where α_j are component weights which must satisfy

$$\sum_{j=1}^{N_c} \alpha_j = 1.$$

The above relation ensures that the normalization property of the density function $\rho(\cdot\,; \boldsymbol{\theta})$ is preserved. The rationale behind having a mixture density network with multiple components ($N_c > 1$) is to allow $\rho(\cdot; \boldsymbol{\theta})$ to have greater flexibility. In this thesis, ρ_j are Gaussian distributions, so that the $\boldsymbol{\theta}_j$ are the conditional means and covariances of the ith component of the distribution function, $\boldsymbol{\theta}_j = (\boldsymbol{\mu}_j(\boldsymbol{Y}|\boldsymbol{X}), \mathbf{C}_j(\boldsymbol{Y}|\boldsymbol{X}))$. Thus, an MDN with multiple components will be more flexible, and will be able to represent non-Gaussian distributions, unlike an MDN with just a single component. This flexibility can be seen in Fig. 4, where we have an example of a one-dimensional MDN with three Gaussian components.

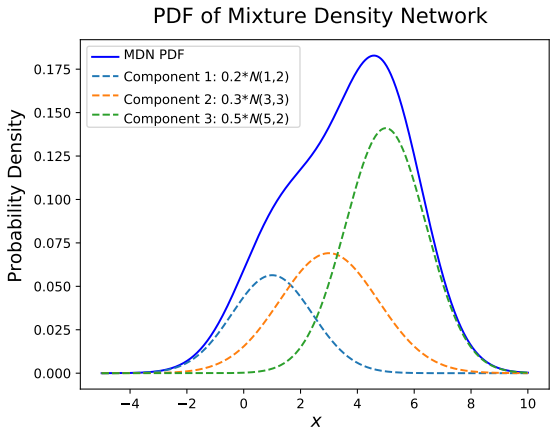

Figure 4: MDN in one dimension, consisting of 3 Gaussian components. $N(1, 2)$ denotes a Gaussian distribution with mean 1 and variance 2.

Furthermore, as the neural network is itself parametric with parameter $\boldsymbol{w}$, MDNs effectively model $p(\boldsymbol{Y}|\boldsymbol{X})$ with $\rho(\boldsymbol{Y}; \boldsymbol{\theta}(\boldsymbol{X}; \boldsymbol{w}))$. In this paper, a multilayer perceptron [22] is chosen as the architecture for all neural networks with $\tanh(x)$ activation functions. Therefore, $\boldsymbol{w}$ consists of weights and biases.

As done in [5], a negative log likelihood function of observations $\mathcal{D} = \{\boldsymbol{X}_i, \boldsymbol{Y}_i\}$ is chosen as the loss function for MDN models. This is given by

$$\mathcal{L}(\boldsymbol{w}; \mathcal{D}) = \sum_i -\ln \rho(\boldsymbol{Y}_i; \boldsymbol{\theta}(\boldsymbol{X}_i; \boldsymbol{w})).$$

Therefore, training an MDN amounts to finding the values of weights and biases ($\boldsymbol{w}^*$) which minimise the loss $\mathcal{L}(\boldsymbol{w}; \mathcal{D})$. It is worth mentioning that the quality of a parametric conditional model lies in how well $\theta(\boldsymbol{X}; \boldsymbol{w}^*)$ represents the true distribution of $\boldsymbol{Y}|\boldsymbol{X}$. As MDNs use an artificial neural network to model $\theta(\boldsymbol{X})$, and these networks are highly flexible, MDNs tend to generalize poorly unless regularization techniques are used. For this reason, all MDNs in this thesis employed the 'early stopping' regularization technique [21]. Early stopping consists in setting aside a small portion of the training data. This dataset is referred to as the test data, and it is not used in the steps of the optimization scheme. The function of the test data is to provide an estimate of the model's generalization error throughout the evolution of the optimization scheme. The test error tends to plateau before training error does, and then eventually tends to increase due to the model overfitting to the training dataset. This phenomenon is displayed in Fig. 5. In early stopping, we monitor the test error and stop the optimization scheme when the test error has plateaued and before it begins to increase, thus avoiding our neural network model overfitting to the training dataset and generalizing poorly.

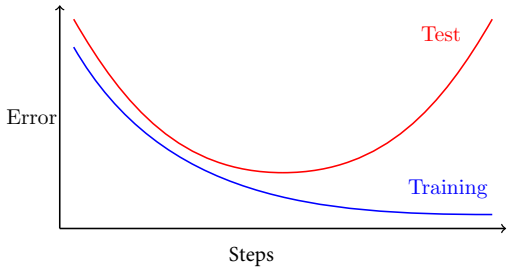

Figure 5: Typical training and test error curves during training of a model. The test error decreases steadily, before it begins to increase due to overfitting

3.3 General Structure of Deterministic and Probabilistic Models

While the approaches of the deterministic and probabilistic MDN models vary, much of their underlying structure is the same. Let us call the state of the system that we are trying to model $\boldsymbol{x}^{t_i} \in \mathbb{R}^{d_x}$, at time t_i. For the Lorenz 63 system, for example, $\boldsymbol{x}^{t_i} = [x^{t_i}, y^{t_i}, z^{t_i}]^T$. The goal of every model in this thesis is to estimate the displacement at time t_i, $\boldsymbol{dx}^{t_i} = \boldsymbol{x}^{t_{i+1}} - \boldsymbol{x}^{t_i} \in \mathbb{R}^{d_x}$. The Markovian models estimate $\boldsymbol{dx}^{t_i}$ given the current state, $\boldsymbol{x}^{t_i}$, while the non-Markovian models estimate $\boldsymbol{dx}^{t_i}$ given the current and (some) past states, e.g. $[\boldsymbol{x}^{t_i}, \boldsymbol{x}^{t_{i-1}}]^T$, $[\boldsymbol{x}^{t_i}, \boldsymbol{x}^{t_{i-1}}, \boldsymbol{x}^{t_{i-2}}]^T$, etc. From now on, we will refer to the model-input as some 'information vector' $\boldsymbol{y}^{t_i} \in \mathbb{R}^{d_y}$ at time t_i. Thus, for Markovian models, $\boldsymbol{y}^{t_i} = \boldsymbol{x}^{t_i}$, while for non-Markovian models the information vector may be given by $\boldsymbol{y}^{t_i} = [\boldsymbol{x}^{t_i}, \boldsymbol{x}^{t_{i-1}}]^T$, $\boldsymbol{y}^{t_i} = [\boldsymbol{x}^{t_i}, \boldsymbol{x}^{t_{i-1}}, \boldsymbol{x}^{t_{i-2}}]^T$ etc. All models are developed in Python using Tensorflow [26] and Tensorflow Probability [27].

In both the deterministic and probabilistic models, a feed-forward neural network is used. The underlying hidden layer structure of the neural network is the same for all models, and is that of a standard multilayer perceptron with 6 hidden layers [22]. The first four hidden layers have 256 neurons and the remaining two have 512. The activation function $\tanh(x)$ is applied to each of the hidden layers. Thus, the activity of hidden layer $j \geq 1$ is

$$\boldsymbol{h}_j = \tanh(W_j \boldsymbol{h}_{j-1} + \boldsymbol{b}_j)$$

and for $j = 1$, the activity is

$$\boldsymbol{h}_1 = \tanh(W_1 \boldsymbol{y}^{t_i} + \boldsymbol{b}_1)$$

.

where $W_j \in \mathbb{R}^{d_j \times d_{j-1}}$ and $\boldsymbol{b}_j$ are the weight and bias parameters of the jth layer, with d_j neurons. $\boldsymbol{y}^{t_i}$ is the input of the model. Writing all the weights and biases as a single vector: $\boldsymbol{w} = \{W_j, \boldsymbol{b}_j\}$.

In the deterministic models, one assumes that there is a deterministic relationship between the displacement at time t_i, $\boldsymbol{dx}^{t_i}$, and the information vector $\boldsymbol{y}^{t_i}$. The model's input and output are thus $\boldsymbol{y}^{t_i}$ and $\boldsymbol{dx}^{t_i}$, respectively. A diagram of the deterministic model structure is given in Fig. 6. The model is trained on a dataset consisting of information vectors and displacements: $D = \{\boldsymbol{y}^{t_i}, \boldsymbol{dx}^{t_i}\}$. The model is then trained to find the weights $\boldsymbol{w}$ which minimize the mean-squared-error (MSE) of training dataset using the Adam algorithm [11]. The output of the network for a given input ($\boldsymbol{y}^{t_i}$) will be noted as $\tilde{\boldsymbol{dx}}_{t_i} \in \mathbb{R}^{d_x}$, where the tilde denotes approximation of the displacement. A description of how the training data is constructed for the models will follow in Section 3.4.

In the probabilistic MDN models, one assumes there is a stochastic relationship between the displacement at time t_i, $\boldsymbol{dx}^{t_i}$, and the current information vector, $\boldsymbol{y}^{t_i}$. More specifically, one assumes the state trajectories are produced by a discrete time process where $p(\boldsymbol{dx}^{t_i}|\boldsymbol{x}^{t_i}, \boldsymbol{x}^{t_{i-1}}, \boldsymbol{x}^{t_{i-2}}, ...) = p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$. Therefore, for $\boldsymbol{y}^{t_i} = \boldsymbol{x}^{t_i}$, one assumes the system is produced by a discrete time Markov process, where $p(\boldsymbol{dx}^{t_i}|\boldsymbol{x}^{t_i}, \boldsymbol{x}^{t_{i-1}}, \boldsymbol{x}^{t_{i-2}}, ...) = p(\boldsymbol{dx}^{t_i}|\boldsymbol{x}^{t_i})$. Thus, the neural network in the models encodes

$$\boldsymbol{\theta}(\cdot) = \{\alpha_j(\cdot), \boldsymbol{\mu}_j(\cdot), \mathbf{C}_j(\cdot)\}_{j=1}^{N_c}$$

such that the distribution of conditional displacements ($p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$) is given by

$$p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i}) = \sum_{j=1}^{N_c} \alpha_j(\boldsymbol{y}^{t_i}) \det(2\pi \mathbf{C}_j(\boldsymbol{y}^{t_i}))^{-\frac{1}{2}} \exp[-\frac{1}{2}(\boldsymbol{dx}^{t_i} - \boldsymbol{\mu}_j(\boldsymbol{y}^{t_i}))^T \mathbf{C}_j^{-1}(\boldsymbol{y}^{t_i})(\boldsymbol{dx}^{t_i} - \boldsymbol{\mu}_j(\boldsymbol{y}^{t_i}))].$$

Here $\boldsymbol{\mu}_j(\cdot)$ and $\mathbf{C}_j(\cdot)$ are the mean vector and covariance matrix of component j. A diagram of the structure of the probabilistic model is given in Fig. 7. Furthermore, in this exploration we examine how the number of components (N_c) and the form of the covariance matrix ($\mathbf{C}_j(\cdot)$) affects the performance of the probabilistic MDN models. More specifically, two N_c values are

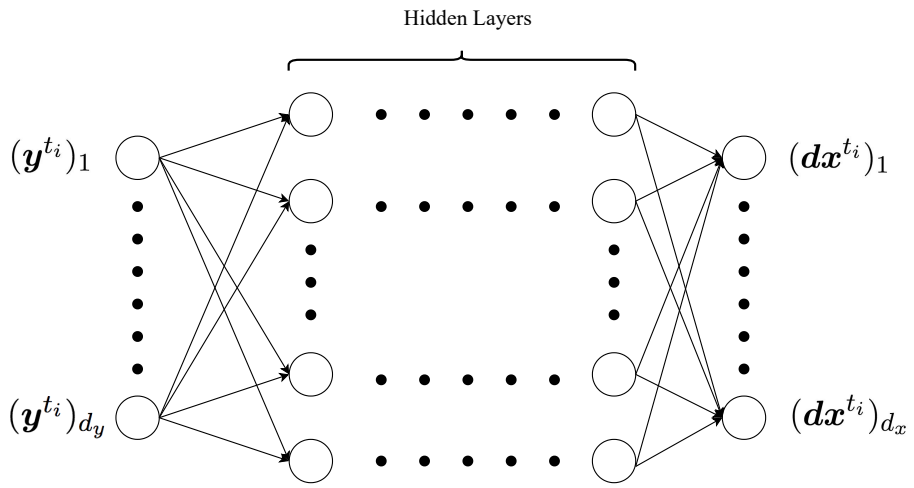

Figure 6: Diagram of neural network structure of Deterministic models.

tested: $N_c = 1$ and $N_c = 32$. For each of these N_c values, two different forms of the covariance matrix are tested: a strictly diagonal covariance matrix, and a matrix without any restrictions on its form. The point of this exercise is to test the performance of different probability distributions in describing fully-observed and partially-observed chaotic systems. The MDN with $N_c = 1$ and a diagonal covariance matrix, assumes that the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ is given by a Gaussian distribution where the components $\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i}$ are independent. From now on, this model will be referred to as the single-diagonal (SD) model. Likewise, the MDN with $N_c = 1$ and an unrestricted covariance matrix assumes that the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ is given by a Gaussian distribution where the components $\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i}$ may be dependent on each other. Thus, this model offers more flexibility compared to the SD model. This model will be referred to from now as the single-lower (SL) model. Here, 'lower' refers to the fact that the covariance matrices are stored as a lower-triangular Cholesky factor in Tensorflow. The model with $N_c = 32$ and a diagonal covariance matrix offers much more flexibility than either the SD or SL models, as this model has 32 Gaussian distribution components. However, here we work under the assumption that components of $\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i}$ are independent. This model shall be called the mixed-diagonal (MD) model. Finally, the model with $N_c = 32$ and an unrestricted covariance matrix offers the most flexibility. This model will be referred to as the mixed-lower (ML) model.

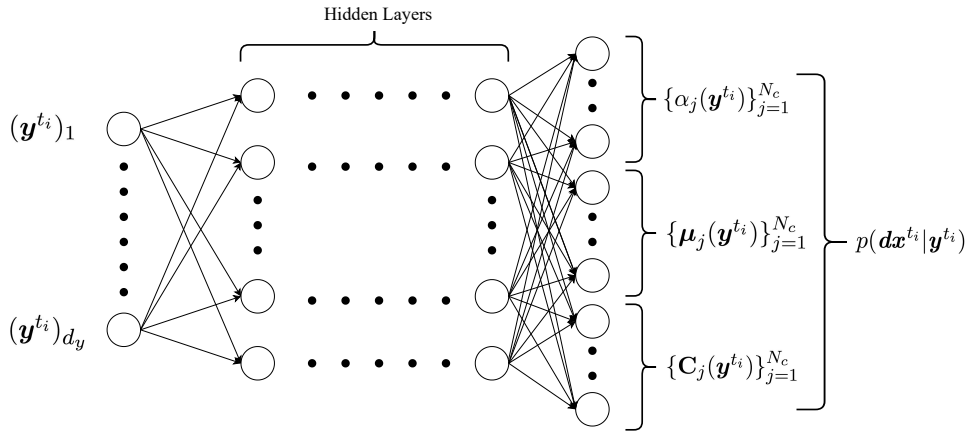

Figure 7: Diagram of neural network structure of probabilistic MDNs.

Therefore, this exploration will, in effect, examine five models, four of which are probabilistic

MDNs. The deterministic model will be referred to from now on as the 'Deterministic model'. Here the uppercase letter is used to refer specifically to the model, and to avoid confusion with the adjective 'deterministic'. For each chaotic dynamical system studied, each model is trained using the same dataset, $D = \{\boldsymbol{y}^{t_i}, \boldsymbol{dx}^{t_i}\}$, which will be described in Section 3.4. However, for the probabilistic models, a negative log likelihood of observations D is chosen as a loss function as described in Section 3.2. The loss function is minimized using the Adam algorithm [11]. Furthermore, the early stopping regularization strategy is used in all models. Here, 80% of the training data is used to train the model, while 20% is kept as a validation set. Furthermore, during the training of the models with the Adam optimizer [11], check-pointing or saving weight values every 10 epochs is performed. Here, an epoch corresponds to all data being used once in the Adam algorithm [11]. Check-pointing weights is done to ensure that if the training of the model finishes on a point which corresponds to overfitting to the training data (a point on the right side of u-shape in the test error as displayed in Fig. 5), then we can effectively go back along the test error curve and access the check-pointed weights corresponding to the minimum in the u-shaped test error.

3.4 Collecting Training Data

To collect training data for the three chaotic systems studied in this thesis, a conventional Runge-Kutta 4th order integration scheme (RK4) [25] is used to integrate Equations 2.1, 2.2, 2.3, 2.4, 2.5 and 2.6 to collect trajectory values. The integration scheme is implemented using a timestep (dt) value of 0.001. As the local truncation error of the RK4 scheme is $\mathcal{O}(dt^5)$ [25], this choice of dt value ensures local truncation error is negligible. The conventional parameter values $r = 28$, $b = \frac{8}{3}$ and $\sigma = 10$ are used in the simulations of the Lorenz 63 trajectories. These settings ensure the system is chaotic. For the monoscale Lorenz 96 scheme, parameter values commonly used for testing stochastic parameterization methods [30] are chosen: $n = 8$ and $F = 20$. These values ensure the system is chaotic. Finally, for the multiscale Lorenz 96 scheme, a variety of parameter values are chosen throughout the exploration in Section 5. However, the free parameter values $n = 8$ and $m = 32$ are used throughout as done in [2].

For each system and choice of parameter values, training data trajectories are generated using the following method. First, a single trajectory of length $10^4 * 100$ timesteps is generated from a randomly generated initial state $\boldsymbol{x}^{t_0} \in \mathbb{R}^{d_x}$ using the RK4 integration scheme. Each component of the initial state $\boldsymbol{x}^{t_0}$ is sampled from a standard uniform distribution between 0 and 1. The trajectory generated from this random initial state is called the 'seed' trajectory. To generate training datasets, the state of the seed trajectory is saved every 10^4 timesteps. The states are saved every 10^4 timesteps as the temporal autocorrelation functions of the systems studied in this thesis are, for the most part, close to zero for $t = 10^4 * dt = 10$ Model Time Units (MTU). We must note that the systems we work with vary greatly. Thus, how close the temporal autocorrelation function of a system is to zero varies from system to system.

Thus, 100 states are saved from the seed trajectory: $\{\boldsymbol{x}^{t_{10^4}}, \boldsymbol{x}^{t_{2*10^4}}, ..., \boldsymbol{x}^{t_{10^6}}\}$. These 100 states are then used as initial states for 100 trajectories, each of length 10^5 timesteps, which are, again, generated using the RK4 integration scheme. These trajectories are then joined together into one file and used as the training data for the neural networks. Thus, the training data consisted of $100 * 10^5 = 10^7$ data points.

The rationale for creating a training dataset consisting of 100 independent trajectories, is to lessen the training dataset's dependence on the randomly generated initial state $\boldsymbol{x}^{t_0}$, and to ensure the data set explores the attractors of both systems more fully. From these trajectories, the corresponding information vectors and displacement vectors, $\{\boldsymbol{y}^{t_i}\}$ and $\{\boldsymbol{dx}^{t_i}\}$ are found and used to construct a training dataset $D = \{\boldsymbol{y}^{t_i}, \boldsymbol{dx}^{t_i}\}$. The data is standardized before training [14], to improve model training stability. Here, we transform both the input data $\{\boldsymbol{y}^{t_i}\}$ and output data $\{\boldsymbol{dx}^{t_i}\}$, separately, by subtracting the mean and dividing each component by its standard deviation. This way, each component of the transformed training data has zero mean and unit variance.

3.5 Model Generated Trajectories

Simulated trajectories are collected from the models to calibrate model performance. Here model performance refers to a model's ability to reproduce statistics of the trajectory of a chaotic system. For each model, 100 trajectories each of length 10^5 timesteps are generated. These trajectories are generated in a similar way to that of the training data. An initial 'seed' trajectory of length $10^4 * 100$ timesteps is generated using the true RK4 update rule for the system. Each component of the initial state of the system $\boldsymbol{x}^{t_0} \in \mathbb{R}^{d_x}$ is sampled from a standard uniform distribution between 0 and 1. To generate initial conditions for data collection, information vectors are constructed from the seed trajectory every 10^4 timesteps. Thus, 100 information vectors are saved from the seed trajectory: $\{\boldsymbol{y}^{t_{10^4}}, \boldsymbol{y}^{t_{2*10^4}}, ..., \boldsymbol{y}^{t_{10^6}}\}$. For Markovian models, $\{\boldsymbol{y}^{t_{10^4}}, \boldsymbol{y}^{t_{2*10^4}}, ..., \boldsymbol{y}^{t_{10^6}}\} = \{\boldsymbol{x}^{t_{10^4}}, \boldsymbol{x}^{t_{2*10^4}}, ..., \boldsymbol{x}^{t_{10^6}}\}$, while for non-Markovian models $\{\boldsymbol{y}^{t_{10^4}}, \boldsymbol{y}^{t_{2*10^4}}, ..., \boldsymbol{y}^{t_{10^6}}\} = \{[\boldsymbol{x}^{t_{10^4}}, \boldsymbol{x}^{t_{10^4+1}}]^T, [\boldsymbol{x}^{t_{2*10^4}}, \boldsymbol{x}^{t_{2*10^4+1}}]^T, ..., [\boldsymbol{x}^{t_{10^6}}, \boldsymbol{x}^{t_{10^6+1}}]^T\}$, etc. The rationale for constructing initial conditions in this way, is to ensure that all states that make up information vectors $\{\boldsymbol{y}^{t_{10^4}}, \boldsymbol{y}^{t_{2*10^4}}, ..., \boldsymbol{y}^{t_{10^6}}\}$ are states on the attractor of the system. This feature provides increased stability to our model generated data. These 100 initial information vectors are then used as initial input states for 100 trajectories, each of length 10^5 timesteps, generated by the models. Each trajectory is generated by continuously inputting $\boldsymbol{y}^{t_i}$ into the models. The corresponding model output, $\tilde{\boldsymbol{dx}}^{t_i}$, is found and added onto the current state, $\boldsymbol{x}^{t_i}$, which is contained within $\boldsymbol{y}^{t_i}$ (by definition $\boldsymbol{y}^{t_i}$ contains the current state and may contain past states as well). A diagram of this process is shown in Fig. 8. For example, if $\boldsymbol{y}^{t_i} = [\boldsymbol{x}^{t_i}, \boldsymbol{x}^{t_{i-1}}]^T$, then, after finding $\tilde{\boldsymbol{dx}}^{t_i}$, we compute $\boldsymbol{x}^{t_{i+1}} = \boldsymbol{x}^{t_i} + \tilde{\boldsymbol{dx}}^{t_i}$ and therefore find $\boldsymbol{y}^{t_{i+1}} = [\boldsymbol{x}^{t_{i+1}}, \boldsymbol{x}^{t_i}]$. Therefore, both the current state and information vector are updated at every timestep. In the case of the Deterministic model, $\tilde{\boldsymbol{dx}}^{t_i}$ are just taken as the model output. In the case of the probabilistic MDNs, the outputs $\tilde{\boldsymbol{dx}}^{t_i}$ are sampled from the outputted distribution, $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$.

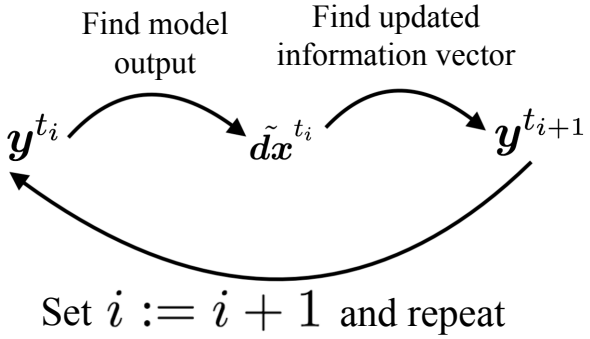

Figure 8: Diagram of process of generating trajectories in models.

Finally, for the probabilistic MDNs, at each iteration of the generation of the output $\tilde{\boldsymbol{dx}}^{t_i}$, the determinant of the covariance matrix of the distribution function $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ is calculated and saved. This is done to quantify to what extent the shape of the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ approaches that of a Dirac delta function for a given input $\boldsymbol{x}^{t_i}$, as the determinant of the covariance matrix of the distribution is a measure of the variance of the distribution. As the systems we are looking at are deterministic, the probabilistic models may operate in a deterministic limit, displaying close to zero variance. Therefore, we are interested in seeing when the probabilistic models are truly probabilistic (displaying non-zero variance) and when they are, instead, operating in a deterministic limit. After collecting determinant values, $|\mathbf{C}(\cdot)|$, the average, $\overline{|\mathbf{C}(\cdot)|}$, is calculated and used as a measure for seeing whether a model is operating in a deterministic limit. This measure should be treated with caution for two reasons. Firstly, $\overline{|\mathbf{C}(\cdot)|}$ just gives us the average of the determinant of the covariance matrix of the distribution function, without giving us any information about its variance. Secondly, and more importantly, this measure has no threshold values associated with deterministic versus probabilistic limits. In other words,

there is no threshold value for $\overline{|\mathbf{C}(\cdot)|}$ where the deterministic limit ends. We will simply use this measure to see which models appear to be operating closer to a deterministic limit.

3.6 Truth Dataset

To evaluate the performance of the models, a single 'truth' trajectory of length 10^7 timesteps or 10^4 MTU is generated using the true RK4 update rule for each system. Each component of the initial state of the system, $\boldsymbol{x}^{t_0} \in \mathbb{R}^{d_x}$ is sampled from a standard uniform distribution between 0 and 1. The statistics of this truth trajectory is then extracted and used as a benchmark to compare those of the model generated trajectories. It is important to note that the truth trajectory consists of a single trajectory of length 10^7 timesteps, while the model generated data consists of 100 trajectories each of length 10^5. Therefore, while the total number of data points is the same in both the truth and model datasets, their forms are different. However, as the model generated states all had initial conditions that are points on the attractor of the system, as explained in Section 3.5, this difference in form should have minimal effect on the results.

3.7 Evaluating Models

After collecting truth and model generated trajectories, different measures are used to quantify their performance. While the measures that will be compared and examined vary from system to system, some common ones are used throughout, which we will define here. These measures can be divided into two classes: stationary statistics measures and scoring rules. Stationary statistics measures examine the long-term behaviour of the model generated trajectories, while the scoring rules examine how well the probabilistic models reproduce short trajectories. We will first discuss the stationary statistics measures.

To compare PDFs, two measures will be used. These are the Kullback-Leibler divergence [13] and Hellinger distance [2]. Both measures are used to compare a given PDF $p(x)$ to a reference PDF $q(x)$, where $x \in \mathbb{R}$ is a continuous random variable. The Kullback-Leibler divergence, $KL(p, q)$, is defined as

$$KL(p,q) = \int_{-\infty}^{\infty} p(x) \log\left(\frac{p(x)}{q(x)}\right) dx.$$

This quantity can be interpreted as the amount of information lost when using $p(x)$ instead of $q(x)$. The Hellinger distance, $H^2(p, q)$, for a continuous random variable $x \in \mathbb{R}$ is defined as

$$H^2(p,q) = \frac{1}{2}\int_{-\infty}^{\infty} \left(\sqrt{p(x)} - \sqrt{q(x)}\right)^2 dx.$$

A lower value of both $KL(p, q)$ and $H^2(p, q)$ indicate that $p(x)$ better approximates $q(x)$. These two measures have different properties which are desirable. The Kullback-Leibler divergence is invariant under parameter transformations while the Hellinger distance is symmetric with respect to p and q [13, 2]. For this reason, both measures are used. In our exploration, we will not have access to the actual PDFs of either the truth data or that generated by the models. We will instead attempt to estimate the PDFs using Kernel Density Estimation (KDE) [6]. KDE is a non-parametric density estimator, meaning we do not assume the underlying PDF that we are trying to estimate belongs to any particular parametric family. This makes KDE a flexible method. To define KDE, let $\boldsymbol{X_1}, \boldsymbol{X_2}, ..., \boldsymbol{X_n} \in \mathbb{R}^d$ be an independent, identically distributed random sample from an unknown distribution P with a density function p. The KDE of the underlying distribution can then be expressed as

$$\hat{p}_n(\boldsymbol{x}) = \frac{1}{nh^d}\sum_{i=1}^{n} K\left(\frac{\boldsymbol{x} - \boldsymbol{X_i}}{h}\right).$$

Here, $K : \mathbb{R}^d \to \mathbb{R}$ is a smooth function called the Kernel function and $h > 0$ is the smoothing bandwidth which controls the amount of smoothing. While many types of Kernel functions exist,

a common one is the Gaussian Kernel, where

$$K(\boldsymbol{x}) = \frac{\exp(-\frac{\|x\|^2}{2})}{\nu_{1,d}}.$$

Here, $\nu_{1,d}$ is the normalizing constant $\nu_{1,d} = \int \exp(-\frac{\|x\|^2}{2})dx$. All KDEs used in this thesis will have Gaussian Kernel functions. Note that the same amount of smoothing, h, is applied in every direction. Intuitively, KDE can be thought of as smoothing each data point into a Gaussian-shaped bump. KDE then sums over all these bumps to obtain an estimate of the PDF of the underlying distribution. This process is shown for a 1-dimensional example in Fig. 9. Therefore, in regions with many observations, KDE will be larger, as there will be more bumps around. On the other hand, in regions with few observations, the density value from summing over bumps is low, but never exactly zero. This last quality of KDEs makes calculating values such as the Kullback-Leibler divergence and the Hellinger distance straightforward.

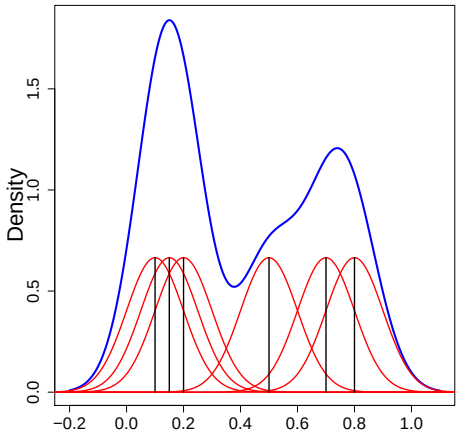

Figure 9: An illustration of how KDE is constructed in 1-dimension. Here there are six observations, indicated by the black lines. Each observation is then smoothed into small bumps (red curves above), which are summed together to obtain a density estimator (blue curve). Figure obtained from [6].

Furthermore, temporal and spatial autocorrelation functions of truth and model generated trajectories are inferred. The temporal autocorrelation function of variable $x^{t_i} \in \mathbb{R}$ which consists of a timeseries of length n with temporal mean value $\bar{x} = \sum_{i=1}^{n} x^{t_i}$ is given by

$$\rho_x(\tau) = \frac{\sum_{i=\tau+1}^{n}(x^{t_i} - \bar{x})(x^{t_{i-\tau}} - \bar{x})}{\sum_{i=1}^{n}(x^{t_i} - \bar{x})^2}.$$

Here $\tau \in \{0 \cup \mathbb{Z}^+\}$ is the temporal lag. Thus, when comparing the truth and model generated trajectories, care must be taken as the length of the timeseries are different. The spatial autocorrelation function of variable timeseries $\boldsymbol{x}^{t_i} \in \mathbb{R}^{d_x}$ which has d_x components, and spatial mean at time t_i, $\bar{x}^{t_i} = \sum_{j=1}^{d_x}(\boldsymbol{x}^{t_i})_j$, is given by

$$\rho_{\boldsymbol{x}}(\delta j) = \frac{\sum_{j=\delta j+1}^{d_x}((\boldsymbol{x}^{t_i})_j - \bar{x}^{t_i})((\boldsymbol{x}^{t_i})_{j-\delta j} - \bar{x}^{t_i})}{\sum_{j=1}^{d_x}((\boldsymbol{x}^{t_i})_j - \bar{x}^{t_i})^2}.$$

Here, $\delta j \in \{0 \cup \mathbb{Z}^+\}$ is the spatial lag. To find a more accurate value for the spatial autocorrelation function of the timeseries, the average over all t_i values is taken.

Finally, the probabilistic models will be evaluated using two probabilistic scoring rules [8]. Unlike the measures we have looked at so far, which examine the long-term or stationary statis-

tics of the model trajectories, scoring rules attempt to quantify model performance in a more 'active' setting. More specifically scoring rules attempt to quantify how well models can reproduce certain trajectories. As scoring rules originate from meteorology, we can think of these certain trajectories as 'weathers'. Thus, scoring rules examine how well a model can reproduce an average instance of weather given some initial weather condition. To calculate scoring rules, a single truth trajectory of length $10^4 * 10^3$ timesteps is run, using the RK4 update rule. Shorter trajectories of length 10^3 timesteps are examined from this long trajectory every 10^4 timesteps. Thus, if the state of the system at time t_i is given by $\boldsymbol{x}^{t_i} \in \mathbb{R}^{d_x}$, then the state of the system at times $\{t_0, t_1, ..., t_{10^3}\}$, $\{t_{10^4}, t_{10^4+1}, ..., t_{10^4+10^3}\}$, etc are examined. Thus 10^3 shorter trajectories, each of length 10^3 timesteps, are examined. These shorter trajectories can be seen as individual instances of weather. 10^3 timesteps is chosen as a length, as this corresponds to 1 MTU, and thus seems a natural choice for a standard length of weather. Furthermore, 1 MTU corresponds to 5 days for the multiscale Lorenz 96 system, and has been used in literature as a standard weather length for this reason [2]. Only 10^3 weather trajectories are examined, as this is a computationally feasible amount of trajectories which should yield fairly accurate results. From these weather trajectories, the initial and final states of the system are saved. If the system we are observing is $\boldsymbol{x}^{t_i} \in \mathbb{R}^{d_x}$, then $\boldsymbol{x}^{t_b}_{truth}$ and $\boldsymbol{x}^{t_f}_{truth}$ denote the initial and final states of each weather. To examine model performance, 10^3 model generated trajectories are run for each $\boldsymbol{x}^{t_b}_{truth}$ value. The final states of the model generated trajectory, $\boldsymbol{x}^{t_f}_{model}$, are saved and then used to calculate the scoring rules. 10^3 model generated trajectories are generated for each 'weather', as this is a computationally feasible amount. Ideally, more would be run if time permitted.

The two scoring rules we use are the negative log score, and the energy score. Both of these scoring rules are strictly proper, meaning that their expectation, with respect to data, is minimized uniquely by the correct model [8]. Thus, a better performing model will have, on average, a lower score than a worse performing model. The negative log score is defined as

$$L(p, \boldsymbol{x}^{t_f}_{truth}) = -\ln\left(p(\boldsymbol{x}^{t_f}_{truth})\right).$$

Here $p(\cdot)$ is the PDF of $\boldsymbol{x}^{t_f}_{model}$. To construct an estimate for $p(\cdot)$, a KDE is constructed using the set of $\boldsymbol{x}^{t_f}_{model}$ values. Therefore, all log score values should be taken with a grain of salt, as the PDF of $\boldsymbol{x}^{t_f}_{model}$ are constructed using just 10^3 $\boldsymbol{x}^{t_f}_{model}$ values for each 'weather'. Thus, the KDEs will likely give a rough estimation of the true PDF of $\boldsymbol{x}^{t_f}_{model}$, especially when the dimension of $\boldsymbol{x}^{t_f}_{model}$ is large. To find an estimate for the average log score, the average of $L(p, \boldsymbol{x}^{t_f}_{truth})$ over all 'weathers' is taken. The energy score is given by

$$ES(P, \boldsymbol{x}^{t_f}_{truth}) = E_P(\|\boldsymbol{x'}^{t_f}_{model} - \boldsymbol{x}^{t_f}_{truth}\|) - \frac{1}{2}E_P(\|\boldsymbol{x'}^{t_f}_{model} - \boldsymbol{x''}^{t_f}_{model}\|)$$

where $\boldsymbol{x'}^{t_f}_{model}$ and $\boldsymbol{x''}^{t_f}_{model}$ are independent copies of a random vector with distribution P of $\boldsymbol{x}^{t_f}_{model}$. To calculate this score in practice, half of the 10^3 $\boldsymbol{x}^{t_f}_{model}$ values are assigned to $\boldsymbol{x'}^{t_f}_{model}$ and the other half are assigned to $\boldsymbol{x''}^{t_f}_{model}$. Thus, unlike the log score where we have to estimate the distribution of $\boldsymbol{x}^{t_f}_{model}$ with KDE, we avoid doing this in the energy score. Consequently, the energy score, in the context of this thesis, will likely be a more reliable scoring rule than the log score. As with the log score, to find an estimate for the average energy score, the average of $L(p, \boldsymbol{x}^{t_f}_{truth})$ over all 'weathers' is taken.

4 Fully Observed System Setting

We will first look at two systems whose state spaces are small enough that they can be modelled using a full description of their dynamics. These are the Lorenz 63 and monoscale Lorenz 96 systems. As we are working with a full description of the dynamics, and both systems are Markovian, attempting to construct a deterministic Markovian surrogate model seems natural.

However, we will also attempt to construct probabilistic Markovian surrogate models, with the goal in mind of showing that they, too, can capture the statistics of a chaotic dynamical system.

4.1 Lorenz 63

To model the Lorenz 63 system, training data is generated as described in Section 3.4. As the models are Markovian, $\boldsymbol{y^{t_i}} = \boldsymbol{x^{t_i}}$, where $\boldsymbol{x^{t_i}} = (x^{t_i}, y^{t_i}, z^{t_i}) \in \mathbb{R}^3$ denotes the state of the system at time t_i. Deterministic and probabilistic models are constructed as described in Section 3.3 and trained. The loss functions for each of these models are given in Appendix A in Figs. A.1.1, A.2.1, A.3.1, A.4.1, A.5.1. The training and test error are shown against epoch value for each graph. Interestingly, there is a considerable amount of noise in the loss functions of all the probabilistic models. The SL model in particular displays a worrying amount of noise. This is an important factor to consider when evaluating the performance of the models, as noise can interfere with finding the true minimum of the loss function. After training the models, trajectories are then generated from these models as described in Section 3.5.

To analyse the performance of the models, different quantities are compared. First, the PDFs of the x, y and z variables for the model and truth trajectories are compared. Images of these model and truth PDFs are given in Appendix A, in Figs. A.1.2, A.1.3, A.1.4, A.2.2, A.2.3, A.2.4, A.3.2, A.3.3, A.3.4, A.4.2, A.4.3, A.4.4, A.5.2, A.5.3 and A.5.4. These plots are created by binning the model and truth trajectory x, y and z values in a histogram. Qualitatively, we can infer from these plots that the Deterministic, ML and MD models reproduce the distributions better than the SD and SL models. Furthermore each model appears to reproduce the PDFs of the x, y and z variables equally well, i.e. the SL model appears to reproduce the PDF of x just as well as it does the PDF of y and so forth.

To quantitatively compare the distributions, the Kullback-Leibler divergence and Hellinger distance are calculated. KDE is used to estimate a form of the truth and model distributions, using a sub-sample of the truth and model datasets. The size of the sub-sample is 10^5 timesteps, or 1% of the total data. A sub-sample is used to construct the KDEs, as using the full dataset would be computationally infeasible. Therefore, the Kullback-Leibler divergence and Hellinger distance results should be taken with a grain of salt, as we are effectively calculating these values using an estimation for the distribution that relies on a reduced version of the dataset.

$p_r(\cdot)$ will refer to the distribution of the truth trajectories, while $p_m(\cdot)$ will refer to that of the given model. The results for the Kullback-Leibler Divergence and Hellinger Distance calculations are given in Tables 1 and 2. We can infer that both measures are consistent with each other in the way models are ranked, i.e. the Deterministic model ranks first for reproducing the x and y distributions but second for reproducing the z distribution according to both measures, and so forth. We can also infer that the Deterministic, ML and MD models all score an order of magnitude lower than those of the SD and SL models: $\mathcal{O}(10^{-4})$ compared to $\mathcal{O}(10^{-3})$ for Kullback-Leibler divergence, and $\mathcal{O}(10^-3)$ compared to $\mathcal{O}(10^-2)$ for Hellinger distance. Therefore, the Deterministic, ML and MD models reproduce the distributions significantly better than the SD and SL models. Therefore, the MDNs with more components, and which have more flexibility, reproduce the PDFs significantly better than the MDNs with a single component, which have less flexibility. However, surprisingly, the SD model consistently reproduces the distributions better than the SL model. Intuitively, one would expect the opposite to be true, as the SL model offers more flexibility. However, as noted previously, the loss function of the SL model displayed a considerable amount of noise. Thus, the optimized weights used to generate the data for the SL model may not correspond to a local minimum in its loss function.

To gain insights into the nature of the variance and covariance of the variables in the system, the covariance matrices of the truth and model generated trajectories are calculated. We will denote these covariance matrices by Σ_r and Σ_m, respectively. To compare Σ_r and Σ_m, the Frobenius-norm of the difference between the matrices divided by the Frobenius-norm of Σ_r is used as a measure. These calculations are shown in the first column in Table 3. As can be seen, the Deterministic, MD and ML models all have a score which is an order of magnitude lower than the SL and SD models. Therefore, it seems that these three models reproduce the variance

Model	$KL(p_m(x), p_r(x))$	$KL(p_m(y), p_r(y))$	$KL(p_m(z), p_r(z))$
Deterministic	2.21E-4	2.46E-4	2.97E-4
ML	2.91E-4	3.19E-4	2.65E-4
MD	2.35E-4	3.12E-4	3.04E-4
SL	1.82E-3	1.47E-3	1.50E-3
SD	1.40E-3	1.16E-3	4.59E-4

Table 1: Kullback-Leibler divergence calculations for distributions of x, y and z in the Lorenz 63 system.

Model	$H^2(p_m(x), p_r(x))$	$H^2(p_m(y), p_r(y))$	$H^2(p_m(z), p_r(z))$
Deterministic	7.18E-3	7.94E-3	8.75E-3
ML	8.25E-3	8.95E-3	8.29E-3
MD	7.39E-3	8.91E-3	8.89E-3
SL	2.11E-2	1.92E-2	1.90E-2
SD	1.83E-2	1.71E-2	1.08E-2

Table 2: Hellinger distance calculations for distributions of x, y and z in the Lorenz 63 system.

and covariance in the variables of the system considerably more accurately than the SL and SD models. Thus, the same models which reproduce the PDFs of x, y and z more accurately, also reproduce a more accurate covariance matrix.

Model	$\frac{\|\Sigma_m - \Sigma_r\|_F}{\|\Sigma_r\|_F}$	$\frac{\|\rho_{x,m}(\tau) - \rho_{x,r}(\tau)\|_2}{\|\rho_{x,r}(\tau)\|_2}$	$\frac{\|\rho_{y,m}(\tau) - \rho_{y,r}(\tau)\|_2}{\|\rho_{y,r}(\tau)\|_2}$	$\frac{\|\rho_{z,m}(\tau) - \rho_{z,r}(\tau)\|_2}{\|\rho_{z,r}(\tau)\|_2}$
Deterministic	4.93E-3	6.39E-2	6.46E-2	3.60E-2
ML	4.53E-3	4.97E-2	5.26E-2	3.62E-2
MD	5.61E-3	5.33E-2	5.47E-2	3.60E-2
SL	2.09E-2	1.26E-1	1.20E-1	9.09E-2
SD	1.26E-2	7.44E-2	7.16E-2	3.87E-2

Table 3: Difference in the truth and model generated covariance matrix and temporal autocorrelation function, Lorenz 63 system.

Furthermore, to gain insights into how the variables changed over time, the temporal autocorrelation functions of the truth and model generated trajectories are found. The temporal autocorrelation functions of x, y and z are found for lag values between 0 and 4 MTU. The truth and model temporal autocorrelation functions are displayed in Appendix A in Figs. A.1.5, A.1.6, A.1.7, A.2.5, A.2.6, A.2.7, A.3.5, A.3.6, A.3.7, A.4.5, A.4.6, A.4.7, A.5.5, A.5.6 and A.5.7. Qualitatively, we can see that the SL model reproduces the temporal autocorrelation functions of the x and y variables considerably less well than the other models. Apart from this detail, there is little separating the models. We'll denote the truth and model generated temporal autocorrelation function of variable x', where x' is one of three variables $\{x, y, z\}$, as $\rho_{x',r}(\tau)$ and $\rho_{x',m}(\tau)$, respectively. Here r denotes the autocorrelation function of the truth trajectories and m that of the models. As the temporal autocorrelation functions tend to be smooth, it sufficed to find values for these functions every 0.01 MTU or $10dt$. To compare the truth and model

generated function values, the 2-norm of the difference between the two functions (evaluated at the given lag values), divided by the 2-norm of the truth autocorrelation function is calculated. The results are given in the second, third and fourth columns in Table 3. As can be seen, the SL model scores an order of magnitude higher than the other models for reproducing $\rho_{x,m}(\tau)$ and $\rho_{y,m}(\tau)$. Aside from this, all the other models perform fairly similarly. Interestingly, the SD model , which performed considerably worse than the MD, ML and Deterministic models in previous scores, performs just as well as them here. Therefore, it appears an MDN model may still be able to reproduce the relationship the variables have with time, while reproducing the distributions of the variables less well.

For the probabilistic models, the mean log scores and energy scores are calculated as described in Section 3.7. The scores are given in Table 4. As can be seen, when it comes to ranking the performance of the models, the two scoring rules do not agree with each other. According to the log score, the SL and SD models perform considerably better than the ML and MD models, whereas the opposite is true according to the energy score. As discussed in Section 3.7, the energy score should be seen as the more reliable scoring rule, as KDEs are not used to estimate a distribution. Therefore, one ought to trust the ranking of the energy score over that of the log score. Furthermore, the fact that the log scores in Table 4 seem large ($\mathcal{O}(10^2)$ and $\mathcal{O}(10^3)$) and quite unpredictable is not encouraging. Finally, comparing the scores of the ML and MD models, and SL and SD models, we can see how the number of components (N_c) appears to be the main factor that separates the models. The ML and MD models score similarly, just like the SL and SD models do.

Model	$\bar{L}$	$\overline{ES}$
ML	1.75E3	1.99E-1
MD	1.71E3	1.99E-1
SL	1.80E2	8.32E-1
SD	1.87E2	8.31E-1

Table 4: Log and energy scores, Lorenz 63 system.

Finally, for the probabilistic models, we can analyse the mean value of the determinant of the covariance matrix in the distribution of $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during the generating of the model-trajectories ($\overline{|\mathbf{C}(\cdot)|}$). These values are given in Table 5. As can be seen, the average determinant values for all models are low ($\mathcal{O}(10^-11)$ and $\mathcal{O}(10^-13)$). Therefore, we can hypothesise that the probabilistic models are, on average, working close to some deterministic limit. This result would not be surprising given the deterministic nature of the Lorenz 63 system, and the fact that the Deterministic model reproduces the statistics of the system well. If this is the case, then we can infer that the SL and SD models reproduce a less accurate deterministic dynamics of the system. In other words, even though the SL and SD models correctly discerned that the underlying dynamics are deterministic, the deterministic relationship they inferred is less correct than that of the MD and ML models.

4.2 Monoscale Lorenz 96

To model the monoscale Lorenz 96 system, training data is generated as described in Section 3.4. Here, like in Section 4.1, our models assume Markovianity and the information vector is given by the current state. Thus $\boldsymbol{y}^{\boldsymbol{t_i}} = \boldsymbol{x}^{\boldsymbol{t_i}}$, where $\boldsymbol{x}^{\boldsymbol{t_i}} = [x_1^{t_i}, x_2^{t_i}, ..., x_8^{t_i}]^T \in \mathbb{R}^8$ denotes the state of the system at time t_i. Deterministic and probabilistic models are constructed as described in Section 3.3 and trained. The loss functions of the models are given in Appendix B in Fig. B.1.1, B.2.1, B.3.1, B.4.1, B.5.1. As can be seen, the loss functions of the MD and SL models display some level of noise. The amount of noise in the SL function is particularly concerning. After training, model trajectories are generated as described in Section 3.5.

Model	$\overline{\lvert\mathbf{C}(\cdot)\rvert}$
ML	2.35E-13
MD	2.05E-11
SL	2.15E-11
SD	2.82E-13

Table 5: Mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during the generation of the model trajectories, Lorenz 63 system.

To analyse the performance of the models, different quantities are compared. First the distribution functions of the x_i variables for the model and truth trajectories are compared. As the attractor of the x_i variable is the same for all $i \in \{1, 2, ..., 8\}$ [29], the distributions functions are also the same. Thus, it sufficed to find the distribution for x_i, $\forall i \in \{1, 2, ..., 8\}$. Images of these model distributions as compared to the truth distribution are given in Appendix B in Fig. B.1.2, B.2.2, B.3.2, B.4.2, B.5.2. These plots are created by binning the model and truth x_i values in a histogram. Qualitatively, we can infer from these plots that all the models reproduce the distributions fairly accurately. The SL model has some issues reproducing the peak of the distribution. To quantitatively compare the model and truth distributions, the Kullback-Leibler divergence and Hellinger distances are calculated. Again, univariate KDEs are used to estimate the form of these distributions using a sub-sample of dataset which consists of $8 * 10^5$ timesteps or 1% of the total data. We will refer to the truth and model distributions as $p_r(x_i)$ and $p_m(x_i)$, respectively. Results for $KL(p_m(x_i), p_r(x_i))$ and $H^2(p_m(x_i), p_r(x_i))$ are given in Table 6. Again, we do not attempt to propagate the error on these calculations. Immediately, we can note how the model scores are lower than the scores we saw in the Lorenz 63 system. The Kullback-Leibler values are $\mathcal{O}(10^{-5})$ or $\mathcal{O}(10^{-3})$, and the Hellinger Distance values are $\mathcal{O}(10^{-2})$ or $\mathcal{O}(10^{-3})$. Comparing these values to those for the Lorenz 63 system, we can say that all the models appear to reproduce the distributions of x_i variables to a fairly high degree of accuracy. That said, the SL model scores are orders of magnitude larger than the other models. This may be due to the loss function of the model exhibiting a significant amount of noise.

The spatial and temporal autocorrelation functions of the truth and model generated trajectories are also inferred. We will refer to the truth spatial and temporal autocorrelation of variable x_i as $\rho_{x_i,r}(\delta i)$ and $\rho_{x_i,r}(\tau)$, respectively. Those of the model generated trajectories will be called $\rho_{x_i,m}(\delta i)$ and $\rho_{x_i,m}(\tau)$, respectively. The spatial autocorrelation functions are calculated for $\delta_i = \{0, 1, 2, 3, 4, 5\}$. The functions are displayed in Appendix B in Figs. B.1.3, B.2.3, B.3.3, B.4.3 and B.5.3. Qualitatively, there is little separating these functions. The temporal autocorrelation functions are found for lag values between 0 and 4 MTU every 0.01 MTU or $10dt$. The functions are displayed in Appendix B in Figs. B.1.4, B.2.4, B.3.4, B.4.4 and B.5.4. Again, qualitatively, there is little separating these functions. To quantitatively compare the truth and model generated spatial and temporal autocorrelation function values, the 2-norm of the difference between the two functions (evaluated at the given lag values), divided by the 2-norm of the truth autocorrelation function is calculated. The results are given in Table 7. As can be seen, the temporal autocorrelation scores are all $\mathcal{O}(10^{-2})$ or $\mathcal{O}(10^{-3})$. Furthermore, all the spatial autocorrelation scores are $\mathcal{O}(10^{-3})$. Therefore, both the probabilistic and deterministic models reproduce the temporal and spatial autocorrelation functions of the Lorenz 96 system fairly accurately. The differences between model scores here are fairly marginal.

The log and energy scores are calculated for the probabilistic models. The results are given in Table 8. As can be seen, the scoring rules are not in agreement with each other regarding the ranking of the models. We saw the same phenomenon in Section 4.1. Again, one would instinctively trust the energy score values over those of the log score, as the KDEs used to calculate the log scores are likely to be even more inaccurate here than they were in Section 4.1, as we are now working in an 8-dimensional state space. Furthermore, the large and unpredictable

Model	$KL(p_m(x_i), p_r(x_i))$	$H^2(p_m(x_i), p_r(x_i))$
Deterministic	4.19E-5	3.18E-3
ML	9.33E-5	4.41E-3
MD	7.60E-5	3.95E-3
SL	7.31E-3	3.93E-2
SD	9.47E-5	4.63E-3

Table 6: Kullback-Leibler and Hellinger calculations, monoscale Lorenz 96 system.

Model	$\frac{\|\rho_{x_i,m}(\tau)-\rho_{x_i,r}(\tau)\|_2}{\|\rho_{x_i,r}(\tau)\|_2}$	$\frac{\|\rho_{x_i,m}(\delta i)-\rho_{x_i,r}(\delta i)\|_2}{\|\rho_{x_i,r}(\delta i)\|_2}$
Deterministic	1.27E-2	3.48E-3
ML	1.25E-2	6.75E-3
MD	1.94E-2	6.56E-3
SL	2.74E-2	7.94E-3
SD	8.87E-3	3.70E-3

Table 7: Temporal and spatial autocorrelation calculations, monoscale Lorenz 96 system.

values of log scores in Table 8 does not aid the scoring rule's apparent reliability. Looking at the energy scores we can see that they are all fairly similar, but that the SL model scores an order magnitude than the other models. This in agreement with our analysis of the stationary statistics of the models, where we saw that the SL model appeared to perform marginally less well than the other models.

Model	$\bar{L}$	$\overline{ES}$
ML	6.06E4	8.67
MD	5.49E4	8.56
SL	3.15E3	1.73E1
SD	3.40E5	6.28

Table 8: Log and energy scores, monoscale Lorenz 96 system.

Finally, for the probabilistic models, we can analyse the mean value of the determinant of the covariance matrix in the distribution of $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during the generation of the model trajectories ($\overline{|\mathbf{C}(\cdot)|}$). The results are given in Table 9. As can be seen, the $\overline{|\mathbf{C}(\cdot)|}$ values in all the models are small. This quantity is particularly small in the SL and SD models. Again, it would not be too surprising for these probabilistic models to be operating in a deterministic limit, considering the underlying deterministic nature of the Lorenz 96 system and how well the Deterministic model reproduces the statistics of the system.

4.3 Comments

In the last two sections, we have seen how both the deterministic and probabilistic Markovian models can reproduce the statistics of a fully observed chaotic systems quite well. We have seen how the probabilistic models often operate in a deterministic limit, i.e. displaying close to zero variance. Furthermore, we have seen that the ML and MD probabilistic models, which

Model	$\overline{\lvert \mathbf{C}(\cdot) \rvert}$
ML	8.13E-9
MD	2.14E-12
SL	7.19E-17
SD	7.05E-21

Table 9: Mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during the generating of the model trajectories, monoscale Lorenz 96 system.

offer more flexibility perform marginally better than the probabilistic SL and SD models, for the Lorenz 63 system. Finally, we have seen that the loss functions of the SL model display considerably more noise during training than the loss functions of the other models. In the next section, we will see how our models perform in a partially observed system setting.

5 Partially Observed System Setting

Now we will look at the multiscale Lorenz 96 system. As described in Section 2.2, this system consists of two coupled systems. The first system ($[x_1, x_2, ..., x_8]^T$) is 8-dimensional. For each x_j in the first system, there are $m = 32$ coupled $y_{j,k}$ variables ($j = 1, 2..., 32$). Thus, the state space of system has dimension $8 + 8 * 32 = 264$, and so constructing a network to describe the full dynamics of the system would be computationally infeasible. We therefore turn to surrogate models which rely on a reduced description of the dynamics. Throughout this section, we will assume we can only observe the x_j variables. Thus, we are working in a partially observed system setting. We will attempt to construct models which reproduce the statistics of the x_j variables, without having any access to the $y_{j,k}$ variables. These models will have the same structure and design as described in Section 3.3. As we are in a partially observed system setting, the Markovianity assumption we have made till now may not always apply here, as discussed in Section 2.4. In Section 5.1, we will first work with Markovian models under different parameter settings. Here, the information vector is given by $\boldsymbol{y^{t_i}} = [x_1^{t_i}, x_2^{t_i}, ..., x_8^{t_i}]^T$. We will eventually find a setting where the Markovian assumption no longer holds. After that, in Section 5.2, we will examine constructing non-Markovian delay-embedded models, with the goal of improving on the failures of the Markovian models.

5.1 Markovian Models

Here, we will examine both deterministic and probabilistic Markovian models with $\boldsymbol{y}^{t_i} = [x_1^{t_i}, x_2^{t_i}, ..., x_8^{t_i}]^T$. We examine these models under different parameter settings.

5.1.1 Large Scale Separation

Here we examine the system with the 'conventional' parameter settings $h = 1, b = 10, c = 10, F = 20$. For these settings, the temporal and spatial scale separation between the x_j and $y_{j,k}$ variables is large, and so it should be relatively 'easy' to model the x_j variables decoupled from the $y_{j,k}$ variables [2]. Here, we can imagine our x_j variables as describing some large spatial scale phenomenon happening at slow timescale. For each x_j, the $y_{j,k}$ ($k = 1, ..., 32$) variables describe small spatial scale phenomena happening at fast timescale, which are coupled to x_j. More specifically, our x_j variables describe a phenomenon which is happening at 10 times the spatial scale, but 10 times less rapidly than the $y_{j,k}$ variables. Furthermore, the strength of coupling between two systems is equal to 1. Finally, the large forcing term, $F = 20$, ensures the system is chaotic [2].

The loss functions for the models are given in Appendix C in Fig. C.1.1, C.2.1, C.3.1, C.4.1 and C.5.1. As can be seen, the loss functions for all the models are smooth and plateau nicely. To calibrate the performance of the models, the same measures are compared as in Section 4.2. We first examine the distribution functions of x_i, as shown in Appendix C in Fig. C.1.2, C.2.2, C.3.2, C.4.2 and C.5.2. Qualitatively, we can say all the models appear to reproduce the distribution equally well. Furthermore, the shape of the truth distribution matches that found in [2], validating our methodology. To quantitatively evaluate how well the models reproduce the distribution, the Kullback-Leibler divergence and Hellinger distance are calculated using a KDE that is constructed with $8 * 10^5$ data points (1% of the total data). The calculations are given in Table 10. As can be seen, the scores for all the models are relatively low: $\mathcal{O}(10^{-4})$ and $\mathcal{O}(10^{-5})$ for Kullback-Leibler divergence, and $\mathcal{O}(10^{-3})$ and $\mathcal{O}(10^{-2})$ for Hellinger distance. We can therefore say that all the models reproduce the distribution of x_i to a high degree of accuracy.

Model	$KL(p_m(x_i), p_r(x_i))$	$H^2(p_m(x_i), p_r(x_i))$
Deterministic	6.37E-5	3.96E-3
ML	7.22E-4	1.35E-2
MD	3.70E-4	9.12E-3
SL	2.35E-4	7.51E-3
SD	1.16E-4	4.92E-3

Table 10: Kullback-Leibler divergence and Hellinger distance, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$ and $F = 20$.

The temporal and spatial autocorrelation functions of the truth and model trajectories are calculated in the same way as in Section 4.2. Graphs of the temporal autocorrelation functions are given in Appendix C in Figs. C.1.4, C.2.4, C.3.4, C.4.4 and C.5.4. Qualitatively, we can say that the ML and SL models reproduce the function slightly less accurately than the Deterministic, MD and SD model. More specifically, the ML and SL models have trouble reproducing the first trough in the function. Graphs of the spatial autocorrelation functions are given in Appendix C in Figs. C.1.3, C.2.3, C.3.3, C.4.3 and C.5.3. Qualitatively, there is little separating how well these models reproduce the function. To quantitatively compare the truth and model temporal and spatial autocorrelation functions, again, the two norm of their difference divided by the two norm of the truth function is used. The results are shown in Table 11. As can be seen, the temporal autocorrelation scores are all of the same order ($\mathcal{O}(10^{-2})$), and quite low. However, the ML and SL models score marginally higher than the other models. We can guess that this has to do with the models failing to reproduce the first trough in the function. The spatial autocorrelation scores for the ML and SL models are an order of magnitude larger than those of the other three models. Thus, the models with full covariance matrix seem to reproduce the spatial autocorrelation function less well than the other models. However, it must also be noted that the scores of the ML and SL models are not worryingly high. The main takeaway from these scores is that all models can reproduce the temporal and spatial autocorrelation function of the system fairly well.

The log and energy scores for the probabilistic models are displayed in Table 12. Once again, the scoring rules do not rank the models in the same way. According to the log score, the ML and MD models perform considerably better than the SL and SD models. However, according to the energy score, there is little separating the model performances. Again, one would instinctively trust the energy score more, due to concerns surrounding our method for calculating the log score, as described previously.

Finally, for the probabilistic models, we can examine $\overline{|\mathbf{C}(\cdot)|}$, as given in Table 13. As can be seen, the average determinant values for all models are small. Comparing these values to

Model	$\frac{\|\rho_{x_i,m}(\tau)-\rho_{x_i,r}(\tau)\|_2}{\|\rho_{x_i,r}(\tau)\|_2}$	$\frac{\|\rho_{x_i,m}(\delta i)-\rho_{x_i,r}(\delta i)\|_2}{\|\rho_{x_i,r}(\delta i)\|_2}$
Deterministic	2.50E-2	6.30E-3
ML	5.80E-2	3.85E-2
MD	2.38E-2	6.57E-3
SL	3.86E-2	2.75E-2
SD	2.51E-2	4.13E-3

Table 11: Temporal and spatial autocorrelation function calculations, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$ and $F = 20$.

Model	$\bar{L}$	$\overline{ES}$
ML	6.22E3	3.63
MD	6.39E3	3.79
SL	1.04E4	3.39
SD	1.13E4	3.15

Table 12: Log and energy scores, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$ and $F = 20$.

the corresponding values that we have already seen in previous systems as given in Tables 5 and 9, we can see that they are just as small or smaller. It would not be surprising for these probabilistic models to be working in the deterministic limit, given how well the Deterministic model performs in this system setting.

Model	$\overline{\|\mathbf{C}(\cdot)\|}$
ML	6.35E-24
MD	4.00E-12
SL	2.42E-24
SD	3.21E-25

Table 13: Mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during generation of the model trajectories, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$ and $F = 20$.

5.1.2 Medium Scale Separation

Here, we examine a system that is similar to the one in Section 5.1.1, but with a different temporal scale separation, $c = 4$. Again, the large forcing term, $F = 20$, ensures the system is chaotic [2]. Here, the $y_{j,k}$ $(k = 1, ..., 32)$ variables are moving at a temporal scale that is only four times faster than the x_j variables. Therefore, as the temporal scale separation between the systems is less pronounced, it may be said that the dynamics of the $y_{j,k}$ variables have more influence on the x_j variables than in Section 5.1.1. Thus, when we construct our surrogate models which rely on descriptions of $[x_1^{t_i}, x_2^{t_i}, ..., x_8^{t_i}]^T$, these models are working with data that has less information about the system than the models in Section 5.1.1. Therefore, it is more difficult to model the x_j variables decoupled from the $y_{j,k}$ variables here than it was in Section 5.1.1.

The loss functions for the models are given in Appendix D in Figs. D.1.1, D.2.1, D.3.1,

D.4.1, D.5.1. As can be seen, all the loss functions are smooth apart from that of the SL model. Here the loss function displays some noise, but not a worrying amount. Furthermore, all the test error loss functions appear to plateau nicely. To calibrate the performance of the models, the same measures are compared as in Sections 4.2 and 5.1.1. The truth and model distribution functions of the x_i variables are given in Appendix D in Figs. D.1.2, D.2.2, D.3.2, D.4.2 and D.5.2. We can say that all the models qualitatively reproduce the truth distribution to a high-degree of accuracy. Furthermore, the shape of the truth distribution agrees with that found in [2], validating our methodology. To quantitatively compare the accuracy of the model distributions, the Kullback-Leibler divergence and Hellinger distance are calculated, and shown in Table 14. As can be seen, all the model scores are of the same order of magnitude for both the Kullback-Leibler divergence ($\mathcal{O}(10^{-3})$) and the Hellinger distance ($\mathcal{O}(10^{-2})$). The Deterministic model scores marginally better than the other models, but not significantly so. Comparing these scores to those in Section 5.1.1 shown in Table 10, we can note that the scores here tend to be orders of magnitude larger than those in the previous section. Therefore, now that we are working with a system where our information vector $\boldsymbol{y}^{t_i} = [x_1^{t_i}, x_2^{t_i}, ..., x_8^{t_i}]^T$ contains less information than in Section 5.1.1, we see that all the models are performing less well. This result is to be expected, as the scale separation between the x_j and $y_{j,k}$ variables is smaller here, meaning that the Markovianity assumption is less valid. That said, none of the models are working particularly badly. Therefore, we cannot say that the Markovianity assumption has broken down. Furthermore, we can see that the Deterministic model is still working just as well as the probabilistic models.

Model	$KL(p_m(x_i), p_r(x_i))$	$H^2(p_m(x_i), p_r(x_i))$
Deterministic	3.34E-3	2.50E-2
ML	6.06E-3	4.02E-2
MD	6.29E-3	3.31E-2
SL	7.02E-3	4.21E-2
SD	8.50E-3	3.39E-2

Table 14: Kullback-Leibler divergence and Hellinger distance, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

The truth and model temporal autocorrelation functions are shown in Appendix D.1 in Figs. D.1.4, D.2.4, D.3.4, D.4.4 and D.5.4. Qualitatively, all models appear to reproduce the function fairly well. Not much separates the models. The same things can be said about the spatial autocorrelation functions, which are shown in D.1 in Figs. D.1.3, D.2.3, D.3.3, D.4.3 and D.5.3. The scores for how well the models reproduced these functions are given in Table 15. We can see that the model scores are all of the same order of magnitude: $\mathcal{O}(10^{-2})$ for temporal autocorrelation and $\mathcal{O}(10^{-3})$ for spatial autocorrelation. Therefore, we can say that all the models reproduce the spatial and temporal autocorrelation functions fairly well. Furthermore, comparing these values to the corresponding ones in Section 5.1.1 shown in Table 11, we can see that they are around the same order of magnitude. Thus, it appears the Markovian surrogate models can still reproduce the spatial and temporal autocorrelation functions of the system.

The log and energy scores for the probabilistic models are displayed in Table 16. The story here is fairly similar to that which we saw in Section 5.1.1. As can be seen, log scores vary much more dramatically than the energy scores. Furthermore, comparing the energy score values to those in Section 5.1.1 in Table 12, we can note that the scores are generally larger. This is to be expected, as we are working in a regime where the information vector contains less information than in Section 5.1.1. Therefore, again, the models appear to be performing marginally less well here than in Section 5.1.1.

Finally, the calculated $\overline{|\mathbf{C}(\cdot)|}$ values are given in Table 17. As can be seen, the $\overline{|\mathbf{C}(\cdot)|}$ of

the SL and SD models are far lower than those of the ML and MD models. Therefore, the SL and SD models appear to be largely working in a deterministic regime, while the ML and MD models appear to be working in a more probabilistic regime. The fact that some models are working in a deterministic regime is not surprising, as we have seen how the Deterministic model reproduces the statistics of the system just as well as the probabilistic models. However, it is interesting to see that the ML and MD models, which again reproduced the statistics of the system just as well as the other models, are working in a more probabilistic regime. Furthermore, the two probabilistic models that are working in a probabilistic limit are those with $N_c = 32$ components. Thus, the multiscale Lorenz 96 system with parameter settings $h = 1$, $b = 10$, $c = 4$ and $F = 20$ appears to be verging on the limit of the Markovianity assumption. In other words, the information vector $\boldsymbol{y}^{t_i} = [x_1^{t_i}, x_2^{t_i}, ..., x_8^{t_i}]^T$ contains enough information that a deterministic Markovian model can perform fairly well. However, a flexible probabilistic Markovian model with $N_c = 32$ components can also perform just as well. In the next section, we will see how the deterministic and probabilistic models work under parameter settings where the temporal scale separation between the x_j and $y_{j,k}$ variables is even less pronounced, and the coupling constant is larger.

Model	$\frac{\|\rho_{x_i,m}(\tau)-\rho_{x_i,r}(\tau)\|_2}{\|\rho_{x_i,r}(\tau)\|_2}$	$\frac{\|\rho_{x_i,m}(\delta i)-\rho_{x_i,r}(\delta i)\|_2}{\|\rho_{x_i,r}(\delta i)\|_2}$
Deterministic	1.52E-2	2.29E-3
ML	3.01E-2	7.15E-3
MD	2.88E-2	5.35E-3
SL	2.52E-2	5.75E-3
SD	1.69E-2	3.27E-3

Table 15: Temporal and spatial autocorrelation calculations, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

Model	$\bar{L}$	$\overline{ES}$
ML	1.31E4	1.22E1
MD	1.33E4	1.22E1
SL	1.64E4	1.15E1
SD	2.69E5	1.18E1

Table 16: Log and energy scores, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

Model	$\overline{\|\mathbf{C}(\cdot)\|}$
ML	2.11E-4
MD	2.10E-2
SL	2.86173E-22
SD	2.18664E-30

Table 17: Mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during the generation of the model trajectories, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

5.1.3 Small Scale Separation and Large Coupling Term

Now we examine a system that is similar to those in Sections 5.1.1 and 5.1.2, but now the coupling strength between the x_j and $y_{j,k}$ variables is increased to $h = 5$, and the temporal scale separation is decreased to $c = 2$. A plot of an x_i trajectory, as shown in Fig. 10, suffices to show the chaotic nature of the system. Thus, here the dynamics of the $y_{j,k}$ variables have much more influence on the x_j variables than they did in Section 5.1.2, as the strength of coupling is five times larger and the temporal scale separation is half. Therefore, the information vector $\boldsymbol{y}^{t_i} = [x_1^{t_i}, x_2^{t_i}, ..., x_8^{t_i}]^T$ which the Markovian models we construct here rely on contains significantly less information than in Sections 5.1.1 and 5.1.2.

These parameter settings are 'invented' in the sense that there is no relevant literature about the multiscale Lorenz 96 system with these settings. The values are chosen to see how the models coped in a partially observed system setting where the information vector is missing significant amounts of information. The idea here is to see how the Deterministic model performs, and to see if the stochastic nature of the probabilistic model can encode the information that is lost when moving from a fully observed system setting to a partially observed system setting.

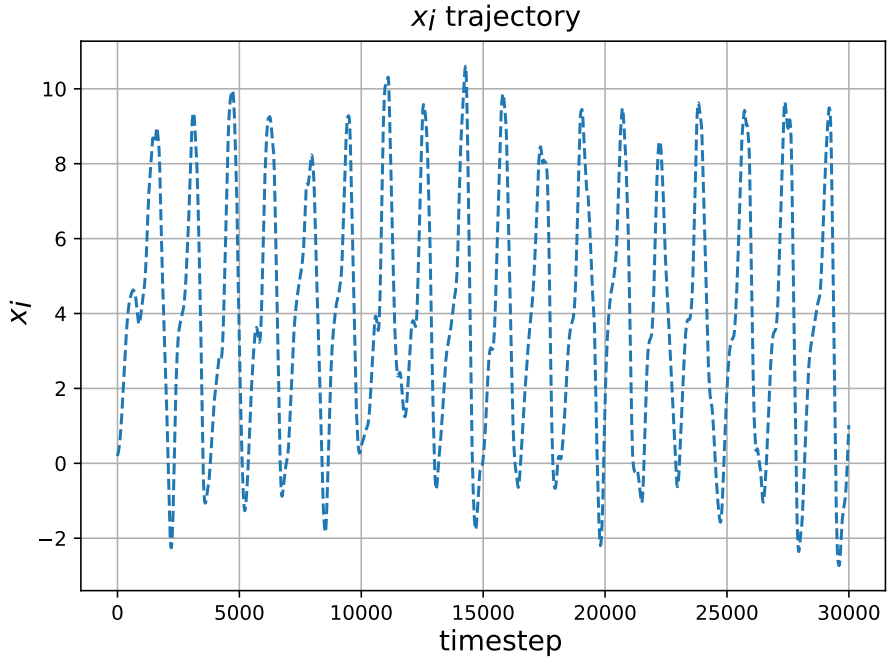

Figure 10: x_i trajectory, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

Plots of the loss functions of the models are given in Appendix E in Figs. E.1.1, E.2.1, E.3.1, E.4.1, E.5.1. The loss functions are all smooth and plateau nicely. However, for the first time in our exploration so far, the loss function of the Deterministic model does not converge to zero. For the ML model loss function, displayed in Fig. E.2.1, we take advantage of check-pointing the weights to use the weights corresponding to the bottom of the u-shaped test error curve in Fig. E.1.1.

To evaluate the performance of the models, the same measures are compared as in Sections 4.2, 5.1.1 and 5.1.2. Histogram plots of the distributions of the x_i variables are shown in Appendix E in Figs. E.1.2, E.2.2, E.3.2, E.4.2, E.5.2. Note that some of the model generated plots had a much larger range than the truth distribution. For the sake of easier legibility, the model generated plots are displayed just in the range of the truth distribution. Looking at these plots, we can immediately see that the probabilistic models reproduce the distribution better than the Deterministic model. Furthermore, we can see that the distribution functions of the ML and MD models are smoother than those of the SL and SD models. To quantitatively compare model performance, the Kullback-Leibler divergence and Hellinger distance values are

displayed in Table 18. As can be seen, the Kullback-Leibler divergence of the Deterministic model is an order of magnitude larger than that of the SL and SD models, and two orders of magnitude larger than that of the ML and MD models. Furthermore, the Hellinger distance of the Deterministic model is by far the largest, and a whole order of magnitude larger than those of the ML and MD models. Therefore, we can say that the Deterministic model reproduces the distribution function considerably less accurately than the probabilistic models. Looking at the scores of the probabilistic models, there is a clear ranking: the SD model performs the worst, the SL model performs slightly better than the SD model, the MD model performs considerably better than either the SL and SD models, and the ML model performs even better than the MD model. Thus, the more flexible the probabilistic model, the better it reproduces the distribution function of x_i. This is exactly as we would expect, as a more flexible probabilistic model has a greater ability to represent the effects which the unobservable dynamics have on the observable dynamics. Furthermore, it is important to note that, while the most flexible ML probabilistic model reproduces the distribution function better than the other models, it still does not reproduce it perfectly. The Kullback-Leibler divergence and Hellinger distance values are still significant: $\mathcal{O}(10^{-2})$.

The temporal autocorrelation functions are given in Appendix E in Figs. E.1.4, E.2.4, E.3.4, E.4.4 and E.5.4. As can be seen, none of the models reproduce the temporal autocorrelation function well. However, the ML and MD models reproduce the function considerably better than the SL, SD and Deterministic models, in the sense that the function has the correct periodicity. The functions of the SL, SD and Deterministic models do not even have a sinusoidal shape. The corresponding performance scores for the temporal autocorrelation function are given in the first column of Table 19. As can be seen, the ML and MD models score an order of magnitude lower than the SL, SD and Deterministic models, which all score around the same value. Thus, the ML and MD models reproduce the temporal autocorrelation function significantly better than the other models, but still not well.

The spatial autocorrelation functions are given in Appendix E in Figs. E.1.3, E.2.3, E.3.3, E.4.3 and E.5.3. As can be seen, the ML and SL models reproduce the functions almost perfectly, while the MD and SD models reproduce it slightly less well. The spatial autocorrelation function of the Deterministic model looks nothing like that of the truth. This qualitative ranking agrees with the corresponding spatial autocorrelation function scores shown in the second column of Table 19. As we can see, the ML and SL models score an order of magnitude lower than the MD and SD models, while the Deterministic model scores by far the highest. Therefore, it appears that the form of the covariance matrix has a greater affect on a probabilistic model's ability to reproduce the spatial autocorrelation function than the number of components in the model.

The log and energy scores for the probabilistic models are displayed in Table 20. Looking at the log scores, we can immediately note that they are all of the same order of magnitude, and surprisingly little separates them. This log score result is quite different from the equivalent results we saw in Sections 5.1.1 and 5.1.2. There we saw that the log score values were fairly large and unpredictable. However, one should instinctively trust the energy score values more than the log score values. Here, we find that the ML and MD models perform consistently better than the SL and SD models. Nevertheless, the difference in scores between the models is less than an order of magnitude. Furthermore, comparing the energy scores to the equivalent results in Sections 5.1.1 and 5.1.2, we can note that the scores here are generally higher. This is again to be expected given that here we are working with an information vector which contains considerably less information than in Sections 5.1.1 and 5.1.2.

Finally, the calculated $\overline{|\mathbf{C}(\cdot)|}$ values are given in Table 21. As can be seen, none of the $\overline{|\mathbf{C}(\cdot)|}$ values are that close to zero. Thus, none of the probabilistic models appear to be working in a deterministic limit. This last finding agrees with our argument from before that it is the stochastic quality of the probabilistic models that allow them to encode the lost information missing in the information vector, and perform better than the Deterministic model.

Model	$KL(p_m(x_i), p_r(x_i))$	$H^2(p_m(x_i), p_r(x_i))$
Deterministic	1.82	4.36E-1
ML	2.24E-2	4.79E-2
MD	5.24E-2	8.92E-2
SL	1.69E-1	1.99E-1
SD	6.13E-1	2.74E-1

Table 18: Kullback-Leibler divergence and Hellinger distance calculations, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

Model	$\frac{\|\rho_{x_i,m}(\tau) - \rho_{x_i,r}(\tau)\|_2}{\|\rho_{x_i,r}(\tau)\|_2}$	$\frac{\|\rho_{x_i,m}(\delta i) - \rho_{x_i,r}(\delta i)\|_2}{\|\rho_{x_i,r}(\delta i)\|_2}$
Deterministic	1.89	7.77E-1
ML	4.80E-1	2.40E-2
MD	4.69E-1	1.96E-1
SL	1.90	1.50E-2
SD	1.89	1.86E-1

Table 19: Temporal and spatial autocorrelation function calculations, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

Model	$\bar{L}$	$\overline{ES}$
ML	7.37E2	2.02E1
MD	7.60E2	2.19E1
SL	7.81E2	3.00E1
SD	7.60E2	3.06E1

Table 20: Log and energy scores, multiscale Lorenz 96 with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

Model	$\overline{\|\mathbf{C}(\cdot)\|}$
ML	1.27E-2
MD	6.17E-2
SL	2.50E-3
SD	3.63E-3

Table 21: Mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during the generation of the model trajectories, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

5.1.4 Comments

In the last three sections, we have examined modelling a partially observed chaotic system with Markovian deterministic and probabilistic surrogate models. We have examined how these models perform under different parameter settings. We have seen how, decreasing the temporal scale separation and increasing the coupling strength between the unobserved and observed variables in the system causes the Markovian deterministic and probabilistic surrogate models to perform less well. This result was largely to be expected from our discussion in Section 2.4. That said, we have seen how the more flexible probabilistic models still manage to perform tolerably well. This is due to the stochastic nature of the models being able to encode the lost information in the partially observed system setting. In the next section, we will examine constructing non-Markovian delay-embedded models with the aim of improving upon our results in Section 5.1.3.

5.2 Delay-embedded Models

Here we will examine constructing models which rely on delay-embedded descriptions of the state space. The motivation for doing this comes from our discussion in Sections 2.3 and 2.4. Here, we will not work directly with Takens' theorem: as the observable state space we are working with is larger than $\mathbb{R}$, it does not apply here. Furthermore, constructing a model which relies on the full memory of the observable state, as in Equation 2.10, is computationally infeasible. Instead, we will use our discussion in Sections 2.3 and 2.4 to justify creating models which rely on delay-embedded information. We will only construct two types of delay-embedded models. The first one is a single time delay-embedded model, which takes as input the current and previous state of the system, so that $\boldsymbol{y^{t_i}} = [x_1^{t_i}, x_2^{t_i}, ..., x_8^{t_i}, x_1^{t_{i-1}}, x_2^{t_{i-1}}, ..., x_8^{t_{i-1}}]$. The second model is a double time delay-embedded model which takes as input the current, and two previous states, so that $\boldsymbol{y^{t_i}} = [x_1^{t_i}, x_2^{t_i}, ..., x_n^{t_i}, x_1^{t_{i-1}}, x_2^{t_{i-1}}, ..., x_n^{t_{i-1}}, x_1^{t_{i-2}}, x_2^{t_{i-2}}, ..., x_n^{t_{i-2}}]$. It is worth mentioning that running these models is much more computationally expensive than the Markovian models in Section 5.1, as we are working with a larger input-space.

5.2.1 Single Time Delay-embedded Model

The single time delay-embedded model takes as input $\boldsymbol{y^{t_i}} = [x_1^{t_i}, x_2^{t_i}, ..., x_8^{t_i}, x_1^{t_{i-1}}, x_2^{t_{i-1}}, ..., x_8^{t_{i-1}}]^T$. While the information vector here contains more information than the vector $[x_1^{t_i}, x_2^{t_i}, ..., x_8^{t_i}]^T$, the quantity of extra information it contains is unlikely to be significant. Thus, here we are examining how the models respond to working with an information vector that contains marginally more information than the one in Section 5.1.3.

The loss functions for the models are given in Appendix F in Figs. F.1.1, F.2.1, F.3.1, F.4.1 and F.5.1. As can be seen, the loss functions are smooth and plateau nicely for all the models apart from the Deterministic and SD models. The Deterministic model test loss function exhibits some noise, but not a worrying amount. Furthermore, it plateaus to a value which is larger than zero. For the SD model, the test loss function exhibits significant amounts of noise. However, the general trajectory of the loss function appears to plateau. Check-pointed weights corresponding to the minimum in the test error are used as the model weights for this model.

To determine the performance of the models, the same measures are compared as in Section 5.1. The probability distributions of the variables, x_i, are given in Appendix F in Figs. F.1.2, F.2.2, F.3.2, F.4.2 and F.5.2. Comparing these figures to the equivalent figures for the Markovian models we saw in Section 5.1.3 (as given in Appendix E in Figs. E.1.2, E.2.2, E.3.2, E.4.2 and E.5.2) we can say that the Deterministic model performs considerably worse in the range of the truth distribution. The model distribution is flat and close to zero. The model distribution has not been displayed for its full range here, for visual convenience. However, the full-range distribution showed that the trajectories moved out to plus and minus infinity. Thus, the Deterministic model appears produce unstable trajectories. The distribution of the ML model has the correct shape of the truth distribution, but appears to be more horizontally stretched out here than in the Markovian model. The MD model distribution looks similar to the equivalent

Markovian model distribution. The SL and SD models capture the distribution significantly better than their equivalent Markovian models. To quantitatively compare these models to each other and to their Markovian counterparts, the Kullback-Leibler divergences and Hellinger distances are calculated and are given in Table 22. Again, the Deterministic model scores the highest, as was the case in the equivalent Markovian model, but, the score is more than three times higher here. The ML and MD models perform considerably worse than their Markovian counterparts, scoring a whole order of magnitude higher. Finally, the SL and SD models perform considerably better than their Markovian counterparts, scoring an order of magnitude lower. Thus, the SL and SD models outperform the ML and MD models.

The results suggest that the marginally more informative information vectors allowed the less flexible probabilistic models to reproduce the distribution of x_i considerably better than in the Markovian case. However, for the more flexible models, which have more parameters, this extra information hindered their ability to reproduce the distributions. Furthermore, the extra information hindered the Deterministic model's performance as well. One can hypothesise that the models performing less well than their Markovian counterparts, as occurred for the more flexible probabilistic models and the Deterministic model, has to do with the positive effects of the marginal increase in information being outweighed by the increase in the dimension of the information vector. Working with a higher-dimension information vector increases the difficulty of the optimization problem. Thus, for the probabilistic models with $N_c = 32$ components, increasing the dimension of the information vector makes the optimization problem considerably harder, outweighing the positive effects of the marginally more informative information vector. However, for the models with only $N_c = 1$ component, increasing the dimension of the information vector makes the optimization problem harder, but not hard enough to outweigh the positive effects of the marginally more informative information vector. For the Deterministic model, one can hypothesise that the extra information in the information vector is still not enough to construct an accurate deterministic surrogate model, and that the extra dimension caused the model to perform more poorly than its Markovian counterpart.

Model	$KL(p_m(x_i), p_r(x_i))$	$H^2(p_m(x_i), p_r(x_i))$
Deterministic	5.60	6.63E-1
ML	4.01E-1	1.94E-1
MD	1.18E-1	1.05E-1
SL	1.78E-2	4.82E-2
SD	5.43E-2	1.01E-1

Table 22: Kullback-Leibler divergence and Hellinger distance calculations, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$, single time delay-embedded models.

The temporal autocorrelation functions are given in Appendix F in Figs. F.1.4, F.2.4, F.3.4, F.4.4 and F.5.4. Qualitatively, we can say that none of the models reproduce the truth function well, but one can argue that the SL model reproduces the general shape of the function the best. The scores for these functions are shown in the first column in Table 23. As can be seen, all the model scores are high and of the same order. Comparing these scores to those of the equivalent models in Section 5.1.3 as given in Table 19, one can see that the ML and MD models perform significantly worse, while the SL and SD models only perform marginally better. Thus, even though the SL and SD models reproduced the distribution function of x_i well, this achievement does not translate into reproducing the temporal autocorrelation function to a similar degree of accuracy.

The spatial autocorrelation functions are given in Appendix F in Figs. F.1.3, F.2.3, F.3.3, F.4.3 and F.5.3. Qualitatively, we can say that the probabilistic models, in particular the SL and SD models, reproduce the function much better than the Deterministic model. The scores

for these functions are shown in the second column in Table 23. As can be seen, the SL and SD models score an order of magnitude lower than the other models. Thus, although the SL and SD models struggled to reproduce the temporal autocorrelation function, they seemed to reproduce the spatial one much more easily. We saw a similar phenomenon in our analysis of the temporal and spatial autocorrelation functions in Section 5.1.3.

Model	$\frac{\|\rho_{x_i,m}(\tau)-\rho_{x_i,r}(\tau)\|_2}{\|\rho_{x_i,r}(\tau)\|_2}$	$\frac{\|\rho_{x_i,m}(\delta i)-\rho_{x_i,r}(\delta i)\|_2}{\|\rho_{x_i,r}(\delta i)\|_2}$
Deterministic	1.96	6.45E-1
ML	1.47	3.34E-1
MD	1.47	2.04E-1
SL	1.36	3.86E-2
SD	1.59	4.47E-2

Table 23: Temporal and spatial autocorrelation calculations, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$, single time delay-embedded models.

.

The log and energy scores for the probabilistic models are displayed in Table 24. As can be seen, the log scores are all of the same order of magnitude. However, these score still appear to be quite large and unpredictable. The energy scores are almost identical for all models. Furthermore, comparing the model energy scores here to their Markovian model counterparts in Table 20, we can see that they are all higher. Therefore, it appears that the single time delay-embedded probabilistic models reproduce instances of weather less well than their Markovian counterparts.

Model	$\bar{L}$	$\overline{ES}$
ML	2.96E4	3.21E1
MD	4.11E4	3.22E1
SL	6.22E4	3.22E1
SD	9.01E4	3.22E1

Table 24: Log and energy scores, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$, single time delay-embedded models.

Finally, the calculated $\overline{|\mathbf{C}(\cdot)|}$ values are given in Table 25. Again, the $\overline{|\mathbf{C}(\cdot)|}$ values for the models are not close to zero ($\mathcal{O}(10^{-2})$). Thus, the models appear not to be working in a deterministic limit. Again, this result shows how it is the stochastic nature of the probabilistic models that give them an edge over the Deterministic model.

5.2.2 Double Time Delay-embedded Model

We will now examine surrogate models which rely on a double time delay-embedded description of the dynamics. These models are trained and their performance examined using the same measures in Section 5.2.1. We find that the results for all the models show significant instability. That is, the model generated trajectories tend to move to positive and negative infinity. Thus, the trajectories do not appear to get stuck on the attractor of the system. In short, the models fail to capture any resemblance of the truth dynamics of the system. For this reason, the quantitative and qualitative results have not been included here, as little information can be discerned from them. Following our argument in Section 5.2.1, we can hypothesise that the

Model	$\overline{\lvert\mathbf{C}(\cdot)\rvert}$
ML	2.52E-2
MD	5.49E-2
SL	6.08E-2
SD	1.14E-2

Table 25: Mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$ during the generation of the model trajectories, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$, single time delay-embedded models.

marginal increase in information in the information vector is outweighed by the considerable increase in input dimension.

6 Discussion

6.1 Overview of Results

In this thesis, we have examined the performance of deterministic and probabilistic surrogate models to describe chaotic dynamical systems. To begin with, we looked at using Markovian surrogate models to describe the dynamics of fully observed chaotic systems. Using a deterministic Markovian surrogate model was a natural choice, as the underlying dynamics of the fully observed systems were deterministic. However, we saw that the probabilistic models could perform just as well as the deterministic models. Furthermore, by examining the mean determinant of the covariance in the distribution $p(\boldsymbol{dx}^{t_i}|\boldsymbol{y}^{t_i})$, $\overline{|\mathbf{C}(\cdot)|}$, we saw how the probabilistic models here operated in a deterministic limit. This is not surprising given the underlying deterministic nature of the dynamical systems.

We then examined the performance of probabilistic and deterministic Markovian surrogate models in a partially observed system setting. More specifically, we examined models which modelled the dynamics of the x_j variables in the multiscale Lorenz 96 system, and ignored the $y_{j,k}$ variables. By tuning the parameters in the system, we could control the effect the dynamics of the unobserved $y_{j,k}$ variables had on the observed x_j variables. By gradually changing parameter values so that the unobserved $y_{j,k}$ variables had more of an effect on the observed x_j variables, we saw how the Markovian models gradually performed less well. This is due to the Markovianity assumption of the models breaking down, as we moved to a regime where the models were working with data that lacked significant amounts of information. However, the degree to which the model performance deteriorated varied from model to model. We saw how the probabilistic models performed considerably better than the Deterministic model, in the most 'extreme' parameter setting where the $y_{j,k}$ variables had the most effect on the observed x_j variables ($h = 5$, $b = 10$, $c = 2$ and $F = 20$). Furthermore, we saw how the more flexible probabilistic models performed better than the less flexible ones. Therefore, it appears that the stochastic nature of the probabilistic models could encode the lost information missing from the data fed to the models. Thus, the more flexible a probabilistic model was, the better it was able to encode the lost information. However, even the most flexible probabilistic models were not perfect. The temporal autocorrelation function of the system, in particular, proved difficult to reproduce.

In an attempt to improve upon the Markovian deterministic and probabilistic model results for the multiscale Lorenz 96 system with parameter settings $h = 5$, $b = 10$, $c = 2$ and $F = 20$, delay-embedded probabilistic and deterministic surrogate models were constructed. The motivation for delay-embedded models was discussed in Section 2.3 and 2.4. We found that constructing delay-embedded models was a non-trivial task. Delay-embedded models operate in a larger input space. This makes the optimization problem of finding the weights which minimize the loss function considerably more complex. First we looked at a single time delay-embedded model, which took as input the current and past state of the system. We found that the performance of the SL and SD models was considerably improved as compared to their Markovian counterparts, according to some measures. However, even these models still struggled to reproduce the temporal autocorrelation function of the system. The ML, MD and Deterministic models, on the other hand, performed considerably worse than their Markovian counterparts, according to all measures. The following is a possible explanation for the probabilistic model results. The single time delay-embedded models are working with inputs which have only marginally more information than their Markovian counterparts. Thus it appears that for the ML and MD models, the positive effects from the marginal increase in information is outweighed by negative effects of the increase in input dimension, while the opposite is true for the SL and SD models. This result is to be expected considering that the ML and MD models have 32 components while the SD and SL models only have 1. Thus, the optimization problem is much more difficult for the ML and MD models than it is for the SL and SD models. For the Deterministic Model, we can discern that the marginal increase in information is outweighed by the increase in dimension. Finally, delay-embedded models which took as input the current and past two states of the system were

constructed. Here, we found that all the models, both deterministic and probabilistic, broke down, and produced unstable results. One can hypothesise that these results are largely due to the increased input dimension of the models.

6.2 Evaluation of Measures

The measures we used in our exploration can be divided up into two classes: those which compared 'stationary' statistics, and the scoring rules, which compared the performance of the models in reproducing specific trajectories, or instances of 'weather'. We saw that, for the most part, the stationary statistics measures were in agreement with each other in terms of ranking the models. Thus, models which performed better according to the Kullback-Leibler divergence, tended to perform better according to the measure used to evaluate the temporal autocorrelation function, and so forth. However, the scoring rules were hardly ever in agreement with each other. More specifically, the log score appeared to be particularly prone to fluctuations. While the energy score ranking of the models tended to agree with the ranking according to the stationary statistics measures, the log score tended to be much more unpredictable. The root of this unpredictability lies in the way the scores were calculated. This was a limitation in the methodology, and shall be discussed in the next section.

6.3 Strengths and Limitations

Evaluating our exploration, we can say that the main strength was the number of probabilistic models we examined. Working with four different probabilistic models allowed us to witness how increased flexibility in a probabilistic model allows it to better encode missing information. This ability was most visibly exemplified when we looked at the multiscale Lorenz 96 system with parameters $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

Arguably the main limitation in the methodology was the way the Kullback-Leibler divergence, Hellinger distance and scoring rules were calculated. Both Kullback-Leibler divergence and Hellinger distance were calculated using KDEs which were constructed using just 1% of the available dataset. Thus, there was room for error to have an effect on the results. Only 1% of the available dataset was used due to computational and time constraints. However, when it came to ranking the performance of different models, the values of the Kullback-Leibler divergences and Hellinger distances were, for the most part, in agreement both with each other and with the corresponding qualitative analysis of the PDFs given in the appendix. Thus, we can speculate that the error on the Kullback-Leibler divergences and Hellinger distances may not have too significant. The same cannot be said for the calculation of the scoring rules. The two scoring rules, the log and energy score, were often in disagreement with each other in terms of ranking the performance of the models. Furthermore, the values of some of the log scores, in particular, were surprising and did not agree with any of the other results. The energy score rankings, on the other hand, were generally in agreement with the stationary statistics measures. The shortcomings in the scoring rules were likely due to the fact that only 10^3 model generated trajectories were run for each instance of 'weather'. Thus, only 10^3 data points were used to construct the multivariate KDEs to calculate the log score for each 'weather'. Constructing accurate multivariate KDEs is a non-trivial task, especially working an 8-dimensional state space, as we did in the monoscale and multiscale Lorenz 96 systems. Thus, generating 10^5 or more model trajectories likely would have led to more accurate results. However, computational and time constraints limited us to generating just 10^3 model trajectories. This constraint also will have affected the accuracy of the energy score. Even though this quantity was not calculated using KDEs, which made it more reliable as a measure, generating 10^5 or more model trajectories likely would have led to more accurate results.

6.4 Conclusions and Further Study

We have seen how deterministic and probabilistic Markovian surrogate models for fully observed chaotic dynamical systems perform equally well. Furthermore, we have seen that, for

partially observed chaotic systems, a Markovianity assumption, as given in Equation 2.11, is valid if the scale separation between the unobserved and observed variables in the system is large. As the scale separation decreases, the Markovianity assumption gradually becomes less valid, and the deterministic and probabilistic Markovian surrogate models perform less well. However, the most flexible probabilistic models perform considerably better than their less flexible counterparts and the deterministic models. Thus, the stochastic nature of the probabilistic models is clearly a useful tool for describing partially observed chaotic dynamical systems. While Takens' embedding theorem has motivated a range of literature for developing accurate deterministic delay-embedded models for partially observed chaotic systems, we saw in Section 5.2 how implementing such models in practice is non-trivial and computationally expensive. Thus, probabilistic Markovian models have shown themselves to be a simpler and computationally less expensive alternative.

Possible extensions to this exploration would revolve around implementing better performing deterministic and probabilistic delay-embedded models. As we saw in Section 5.2, implementing delay-embedded models is a non-trivial task, which would require more time to experiment with different methods and to conduct relevant research. With more available time, one idea which would be easy to attempt is the implementation of a single time delay-embedded model, but with a longer delay. Here, the input of the model would be the current and past state k time steps back, at times t_i and t_{i-k}. By doing this, we could investigate whether information vectors containing the state at time t_i and t_{i-k} contain marginally more information than an information vector containing the state at time t_i and t_{i-1}, and whether they could thus improve delay-embedded model performance. Analysis of the system's temporal autocorrelation function could motivate the values of k to experiment with.

More long-term extensions to this project would be to implement similar models to real systems, and not to just toy models. In fluid dynamics, in particular, there has been a considerable amount of interest recently in implementing probabilistic surrogate models [5, 1, 24]. Fluid turbulence, in particular, would be a natural area to explore. Numerical simulation of fluid turbulence is notoriously expensive [28], and there is thus a variety of literature regarding construction of surrogate reduced-order models of turbulent flows [24]. Therefore, one could naturally extend our investigation of Markovian and delay-embedded deterministic and probabilistic surrogate models to the study of turbulent flows.

References

[1] N. Agarwal, D. Kondrashov, P. Dueben, E. Ryzhov, and P. Berloff. A comparison of data-driven approaches to build low-dimensional ocean models. *Journal of Advances in Modeling Earth Systems*, 13, 2021.

[2] H. Arnold, I. M. Moroz, and T. N. Palmer. Stochastic parameterizations and model uncertainty in the lorenz '96 system. *Phil Trans R Soc A*, 371, 2013.

[3] C. M. Bishop. Mixture density networks. Technical report, Aston University, 1994.

[4] C. M. Bishop. *Pattern recognition and machine learning*. 2006.

[5] M. T. Brolly. Inferring ocean transport statistics with probabilistic neural networks. *Journal of Advances in Modeling Earth Systems*, 15, 2023.

[6] Y. C. Chen. A tutorial on kernel density estimation and recent advances. *Biostatistics Epidemiology*, 1:161–187, 2017.

[7] D. J. Gagne, H. M. Christensen, A. C. Subramanian, and A. H. Monahan. Machine learning for stochastic parameterization:generative adversarial networks in thelorenz '96 model. *JAMES*, 12(3), 2020.

[8] T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102:359–378, 2007.

[9] J. P. Huke. Embedding nonlinear dynamical systems: A guide to takens' theorem. 2006.

[10] J. Kerin and H. Engler. On the lorenz '96 model and some generalizations. *Discrete Continuous Dynamical Systems Series B*, 27(2), 2022.

[11] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[12] S. G. Krantz. *Handbook of Complex Variables*, chapter The Concept of Homeomorphism, page 86. Birkhäuser Boston, MA, 1 edition, 1999.

[13] S. Kullback and R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22:79–86, 1951.

[14] Y. A. LeCun, L. Bottou, G. Orr, and K. R. Müller. *Efficient backprop. In Neural networks: Tricks of the trade*, pages 9–48. Springer, 2 edition, 2012.

[15] M. E. Levine and A. M. Stuart. A framework for machine learning of model error in dynamical systems. *Communications of the American Mathematical Society*, 2:283–344, 2022.

[16] E. Lorenz. Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20:130–141, 2024.

[17] D. Orrell. Model error and predictability over different timescales in the lorenz '96 systems. *AMS*, 60(17), 2003.

[18] E. Ott. *Chaos in Dynamical Systems*, chapter Strange attractors and fractal dimension, pages 71–72. Cambridge University Press, 2 edition, 2002.

[19] E. Ott, B. R. Hunt, I. Szunyogh, A. V. Zimin, E. J. Kostelich, M. Corazza, E. Kalnay, D. Patil, and J. A. Yorke. A local ensemble kalman filter for atmospheric data assimilation. *Tellus A*, 56(5), 2004.

[20] H. Peng, W. Wang, P. Chen, and R. Liu. Defm: Delay-embedding-based forecast machine for time series forecasting by spatiotemporal information transformation. *Chaos*, 34, 2024.

[21] L. Prechelt. *Neural networks: Tricks of the trade*, chapter Early stopping—But when?, pages 53–67. 2nd edition, 2012.

[22] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. Technical report, UCSD Institute for Cognitive Science, 1985.

[23] T. B. Rushing. *Topological embeddings*, chapter 1, pages 1–2. Elsevier, 1973.

[24] I. Shokar, R. Kerswell, and P. Haynes. Stochastic latent transformer: Efficient modelling of stochastically forced zonal jets. *Journal of Advances in Modeling Earth Systems*, 6, 2024.

[25] J. M. Stewart. *Python for Scientists*. Cambridge University Press, 2014.

[26] TensorFlow Developers. Tensorflow.

[27] TensorFlow Probability Developers. Tensorflow probability.

[28] G. K. Vallis. *Essentials of Atmospheric and Oceanic Dynamics*, chapter 10, page 188–212. Cambridge University Press, 2019.

[29] D. L. van Kekem. *Dynamics of the Lorenz 96 Model*. PhD thesis, University of Groningen, 2018.

[30] D. Wilks. Effects of stochastic parametrizations in the lorenz '96 system. *Quarterly Journal of the Royal Meteorological Society*, 131(606):389–407, 2005.

Appendices

A Lorenz 63 Data

A.1 Deterministic Model

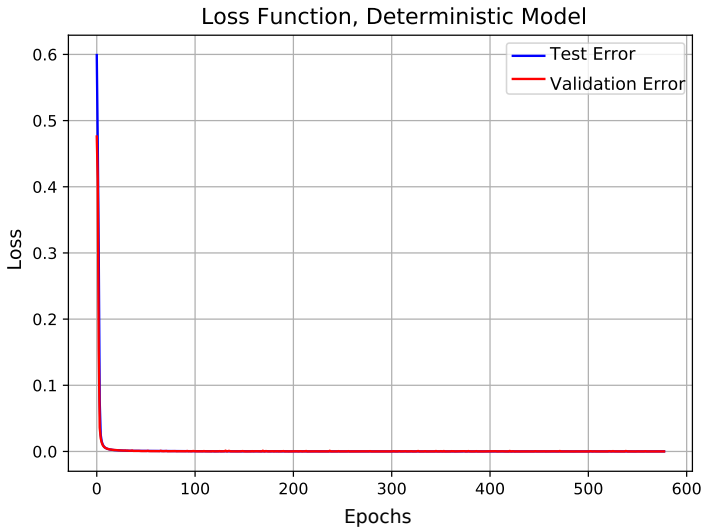

Figure A.1.1: Loss function when training Deterministic model, Lorenz 63 system.

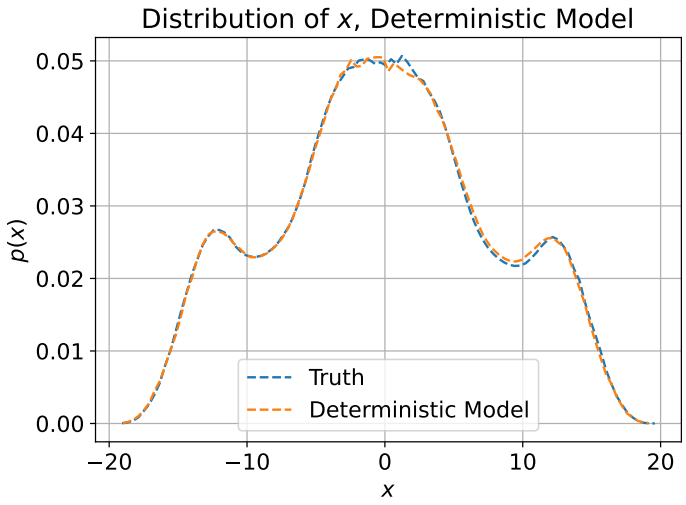

Figure A.1.2: Distribution function of x inferred from Deterministic model, Lorenz 63 system.

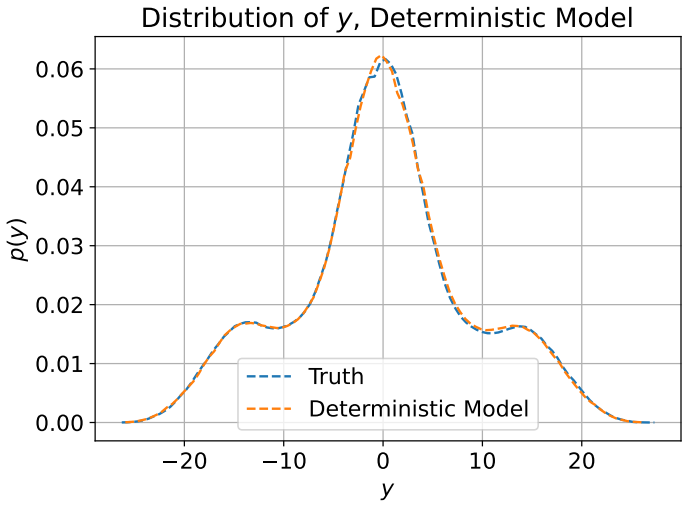

Figure A.1.3: Distribution function of y inferred from Deterministic model, Lorenz 63 system.

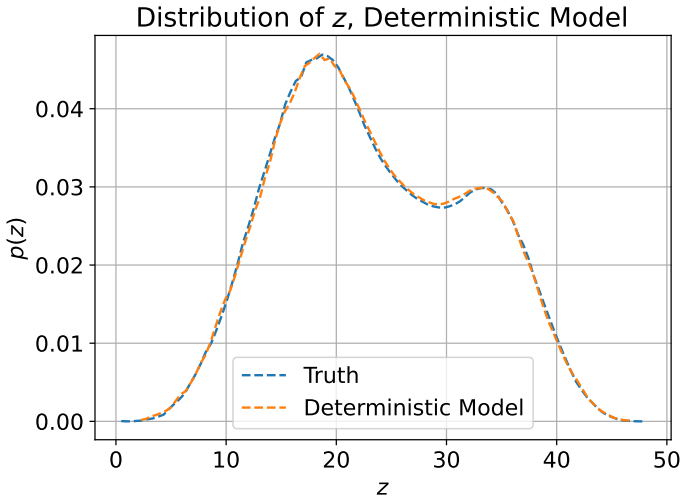

Figure A.1.4: Distribution function of z inferred from Deterministic model, Lorenz 63 system.

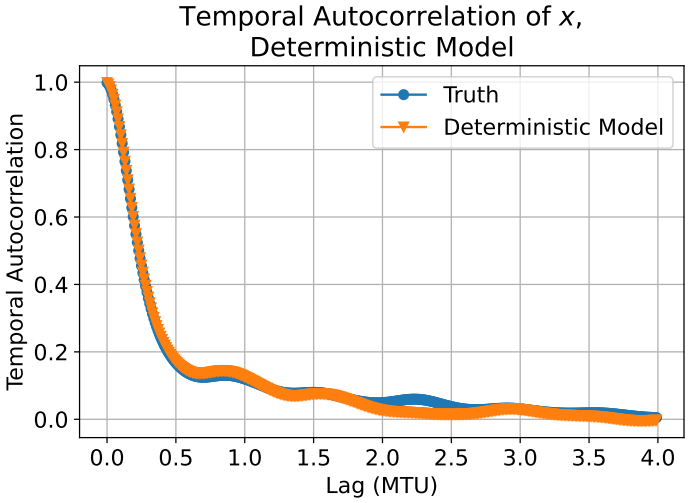

Figure A.1.5: Temporal autocorrelation function of x inferred from Deterministic model, Lorenz 63 system.

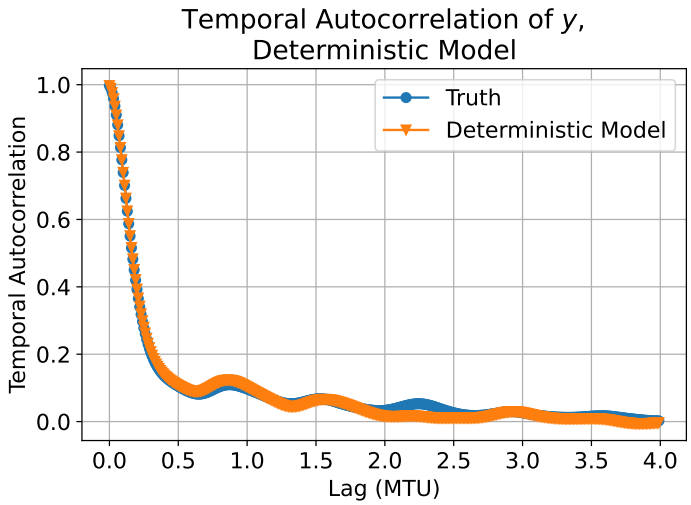

Figure A.1.6: Temporal autocorrelation function of y inferred from Deterministic model, Lorenz 63 system.

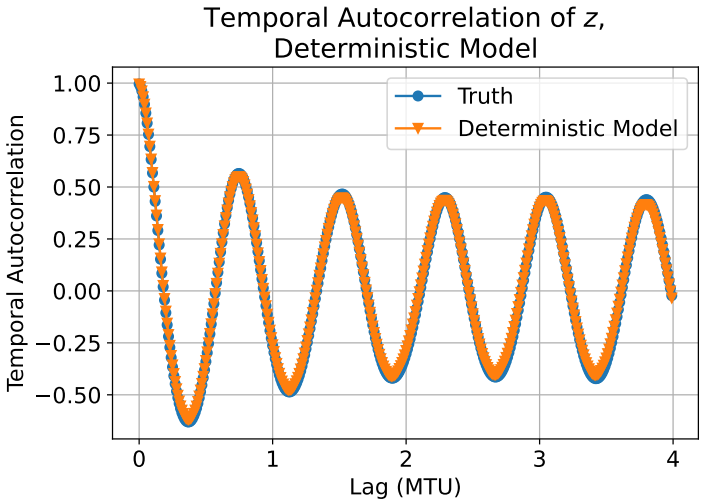

Figure A.1.7: Temporal autocorrelation function of z inferred from Deterministic model, Lorenz 63 system.

A.2 ML Model

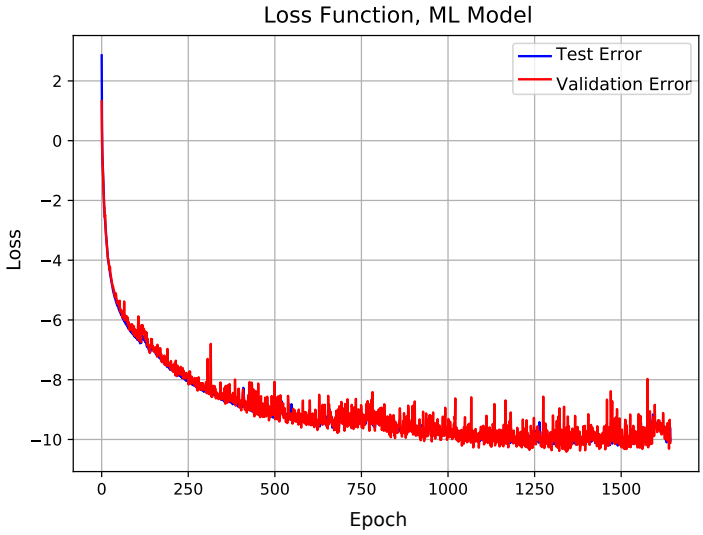

Figure A.2.1: Loss function when training ML model, Lorenz 63 system.

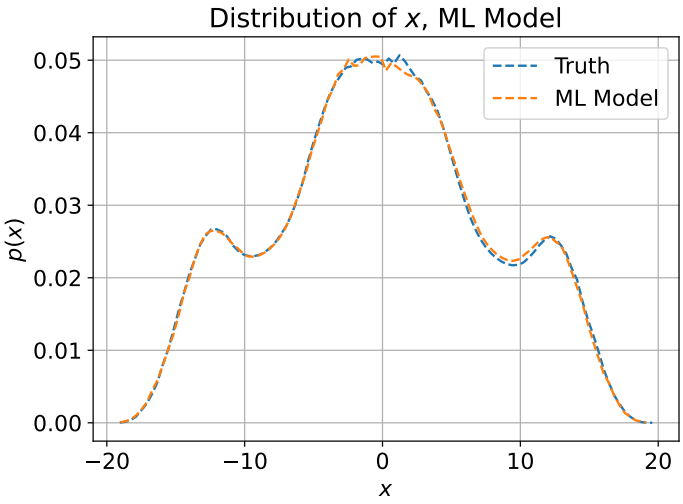

Figure A.2.2: Distribution function of x inferred from ML model, Lorenz 63 system.

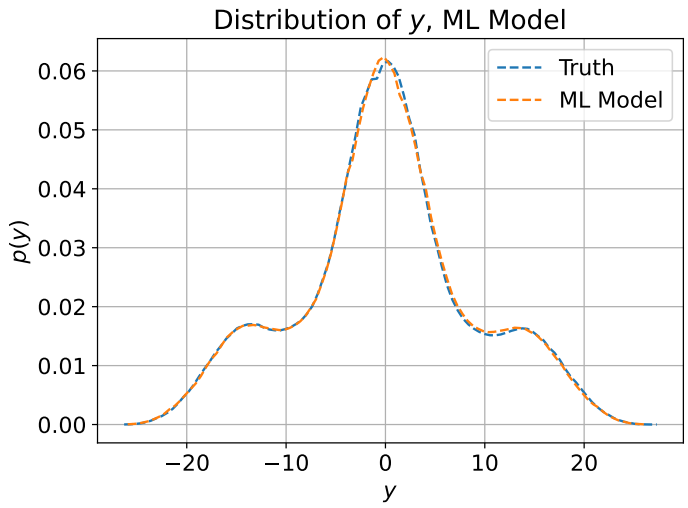

Figure A.2.3: Distribution function of y inferred from ML model, Lorenz 63 system.

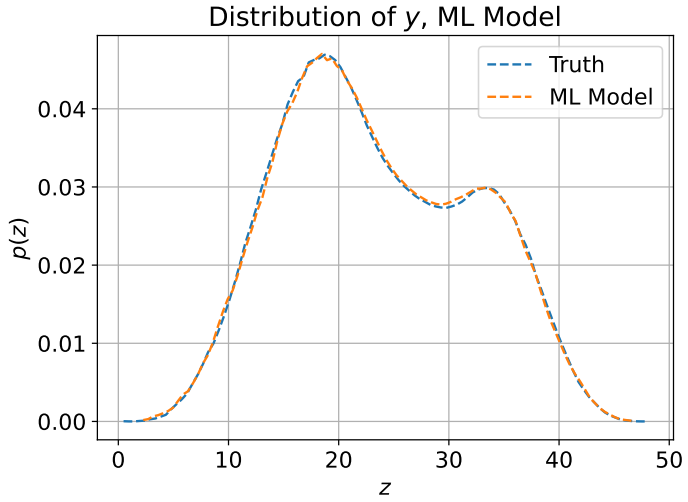

Figure A.2.4: Distribution function of z inferred from ML model, Lorenz 63 system.

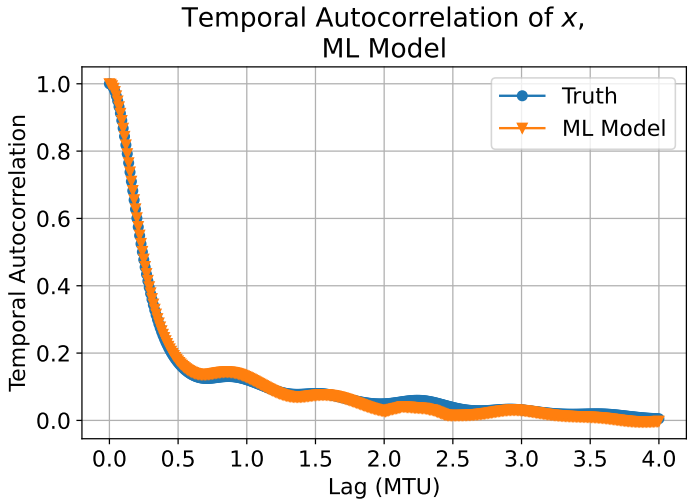

Figure A.2.5: Temporal autocorrelation function of x inferred from ML model, Lorenz 63 system.

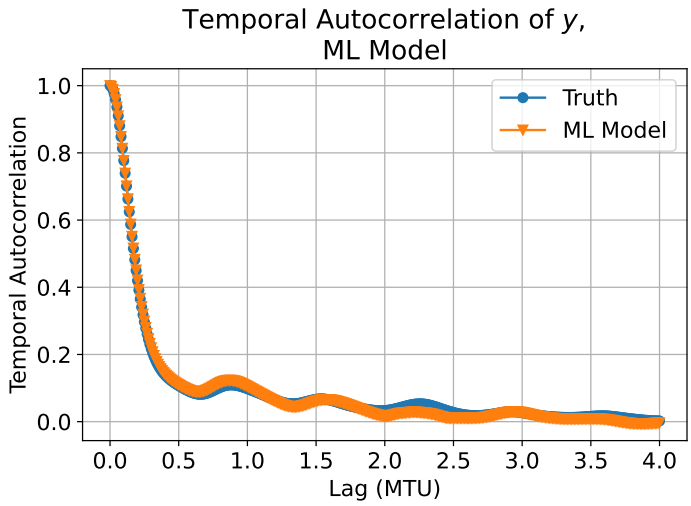

Figure A.2.6: Temporal autocorrelation function of y inferred from ML model, Lorenz 63 system.

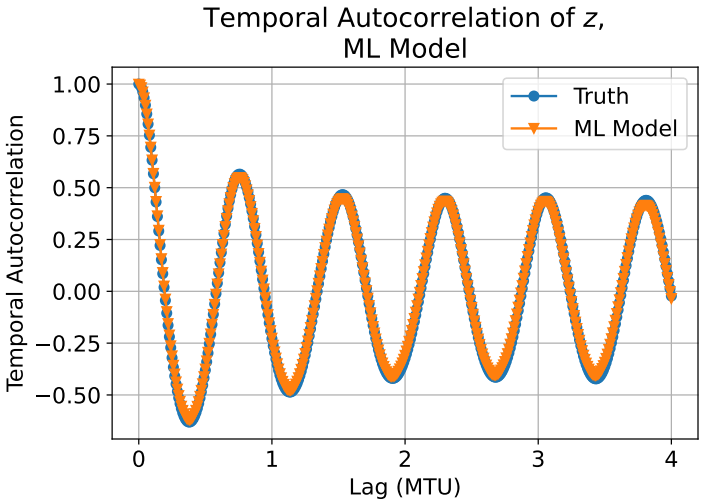

Figure A.2.7: Temporal autocorrelation function of z inferred from ML model, Lorenz 63 system.

A.3 MD Model

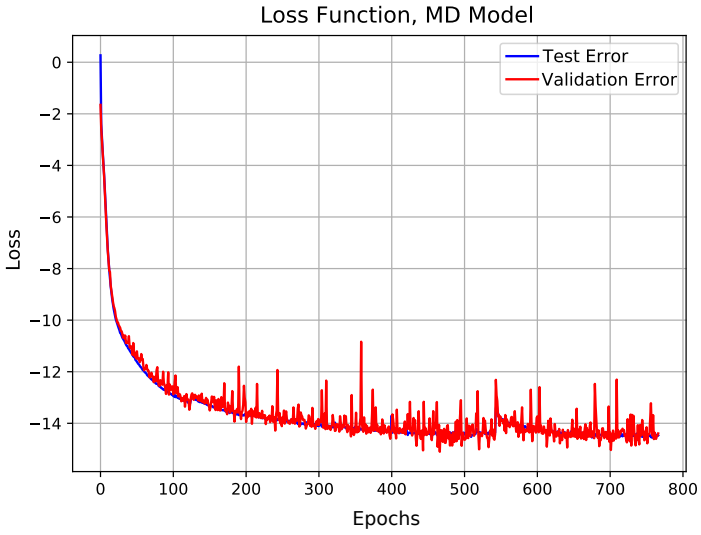

Figure A.3.1: Loss function when training MD model, Lorenz 63 system.

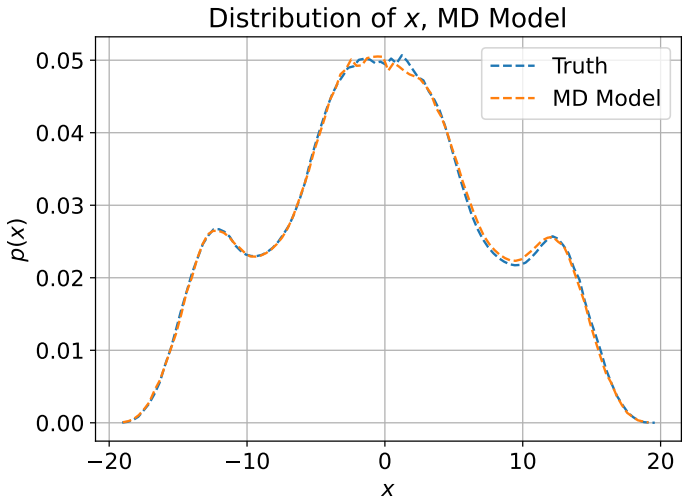

Figure A.3.2: Distribution function of x inferred from MD model, Lorenz 63 system.

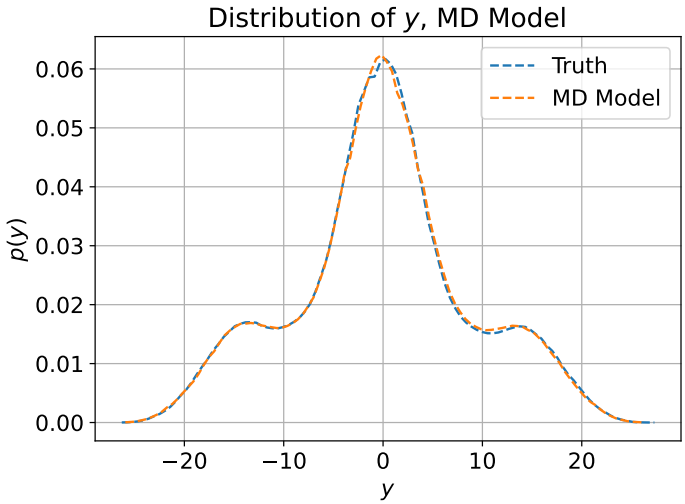

Figure A.3.3: Distribution function of y inferred from MD model, Lorenz 63 system.

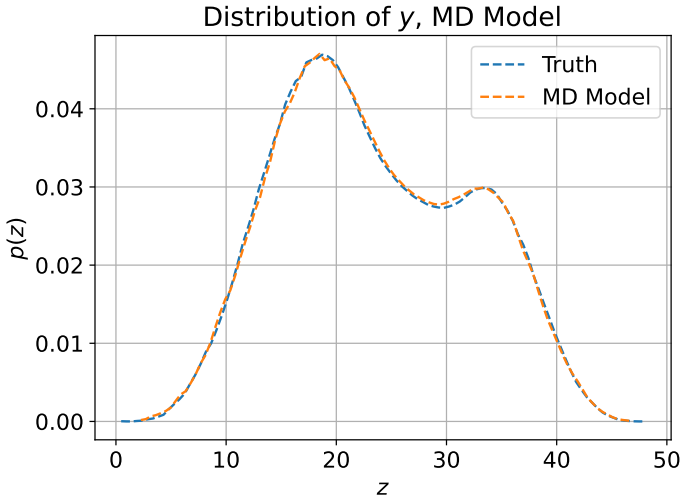

Figure A.3.4: Distribution function of z inferred from MD model, Lorenz 63 system.

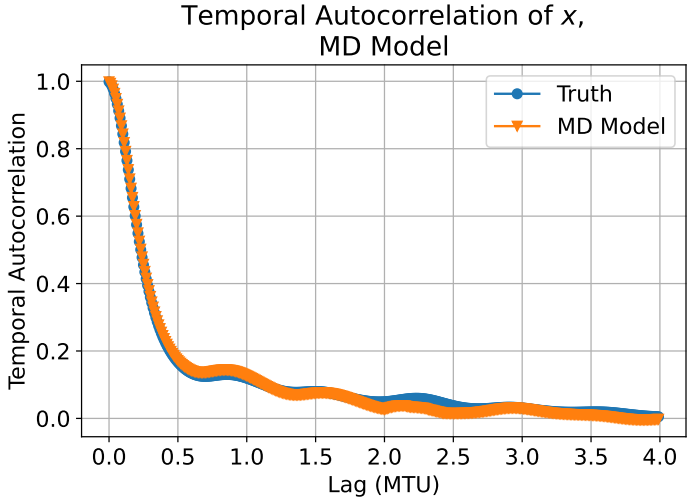

Figure A.3.5: Temporal autocorrelation function of x inferred from MD model, Lorenz 63 system.

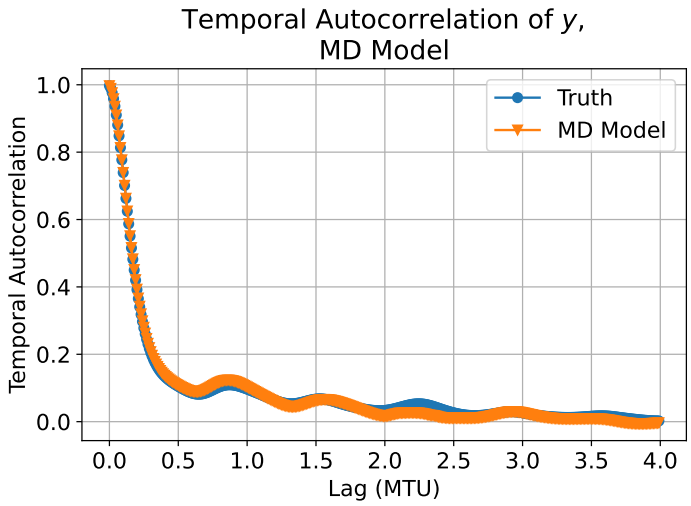

Figure A.3.6: Temporal autocorrelation function of y inferred from MD model, Lorenz 63 system.

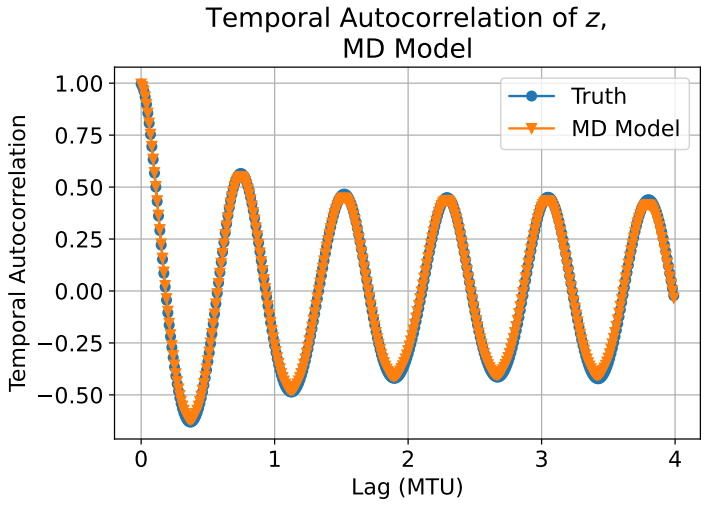

Figure A.3.7: Temporal autocorrelation function of z inferred from MD model, Lorenz 63 system.

A.4 SL Model

Figure A.4.1: Loss function when training SL model, Lorenz 63 system.

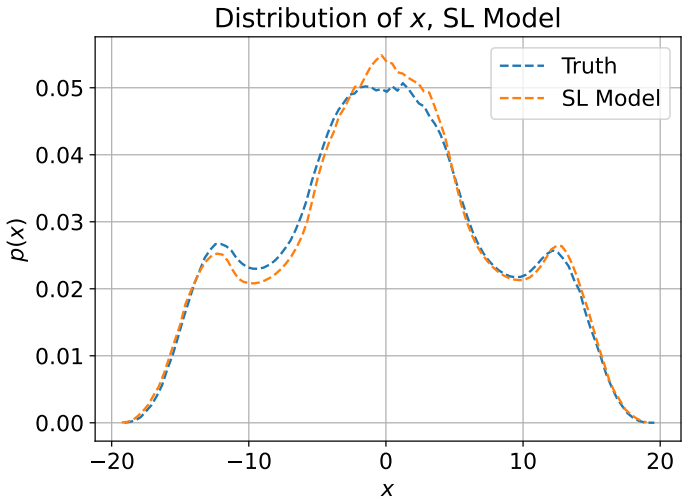

Figure A.4.2: Distribution function of x inferred from SL model, Lorenz 63 system.

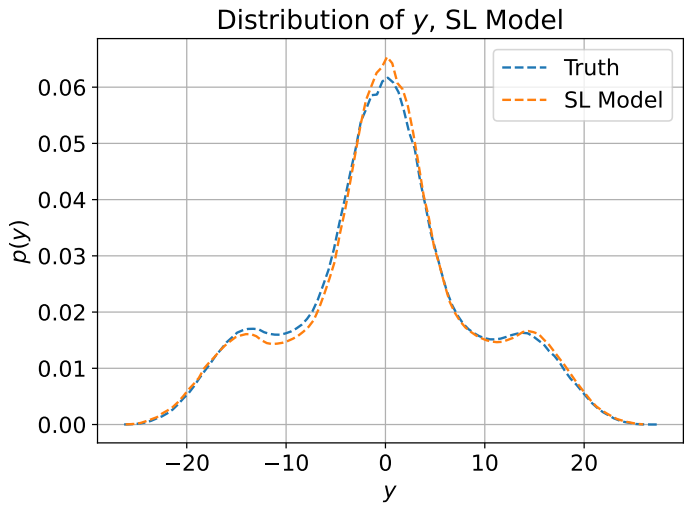

Figure A.4.3: Distribution function of y inferred from SL model, Lorenz 63 system.

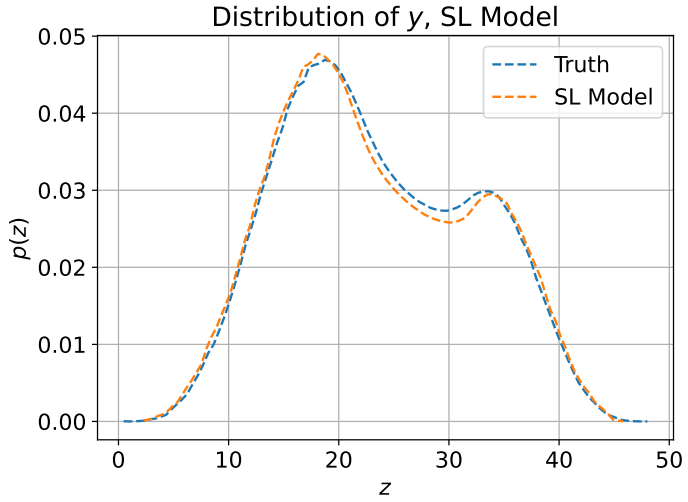

Figure A.4.4: Distribution function of z inferred from SL model, Lorenz 63 system.

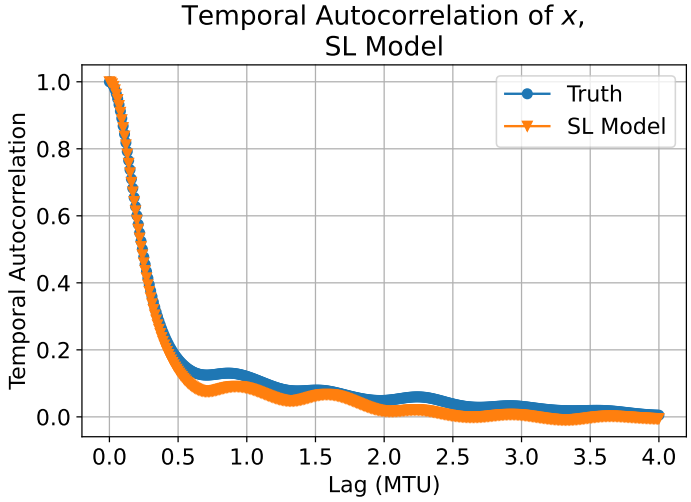

Figure A.4.5: Temporal autocorrelation function of x inferred from SL model, Lorenz 63 system.

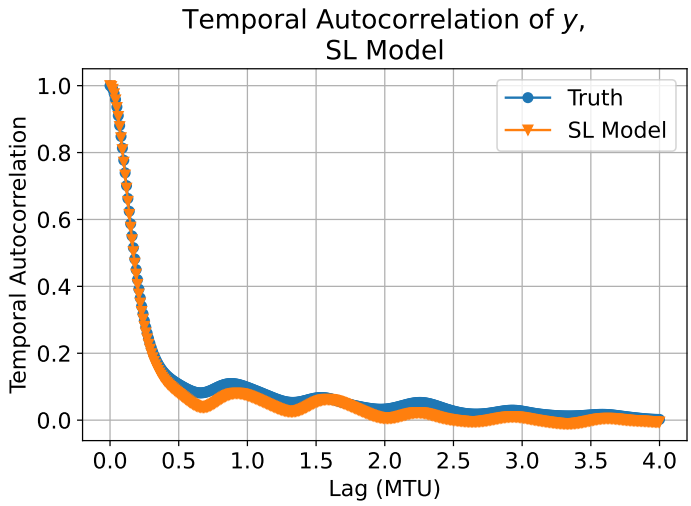

Figure A.4.6: Temporal autocorrelation function of y inferred from SL model, Lorenz 63 system.

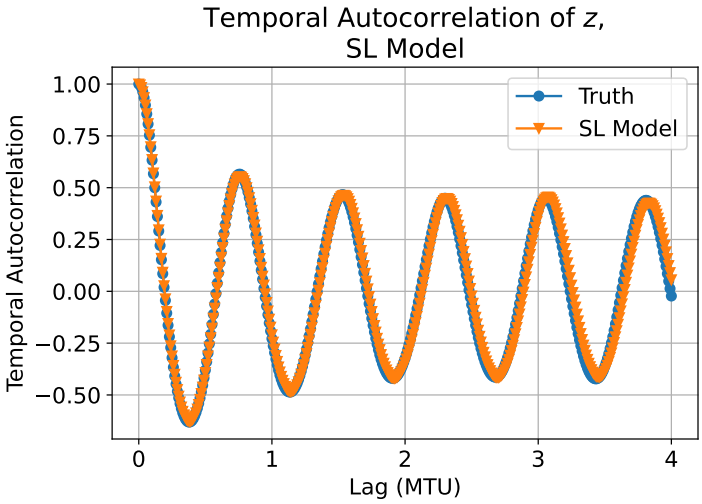

Figure A.4.7: Temporal autocorrelation function of z inferred from SL model, Lorenz 63 system.

A.5 SD Model

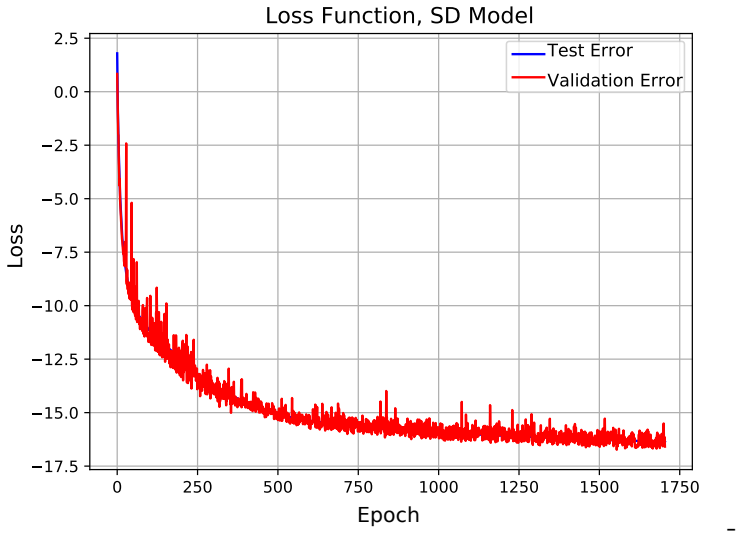

Figure A.5.1: Loss function when training SD model, Lorenz 63 system.

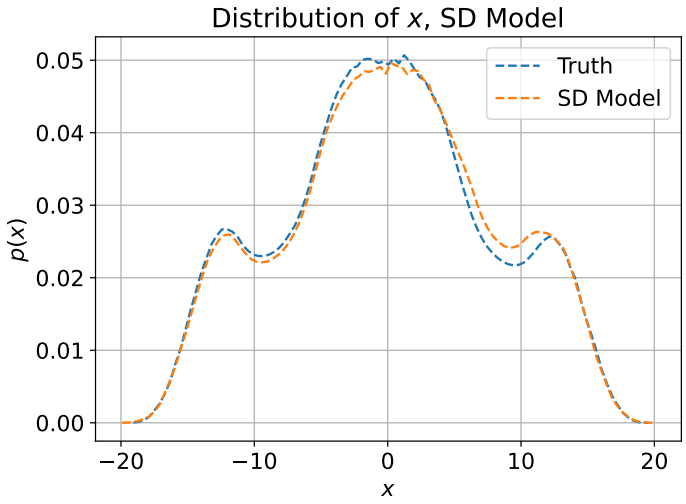

Figure A.5.2: Distribution function of x inferred from SD model, Lorenz 63 system.

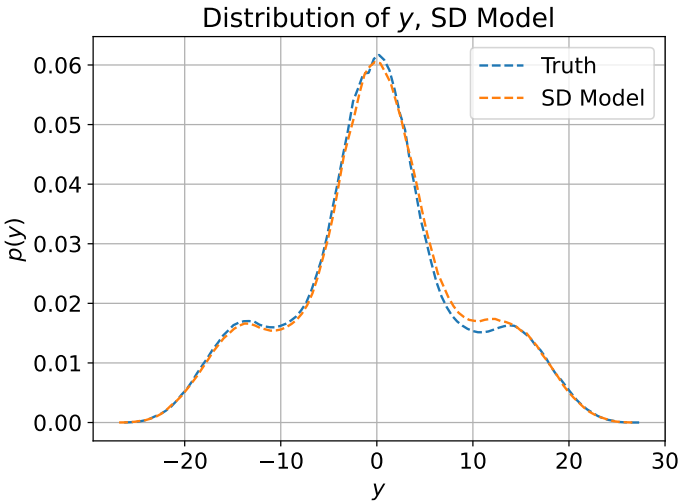

Figure A.5.3: Distribution function of y inferred from SD model, Lorenz 63 system.

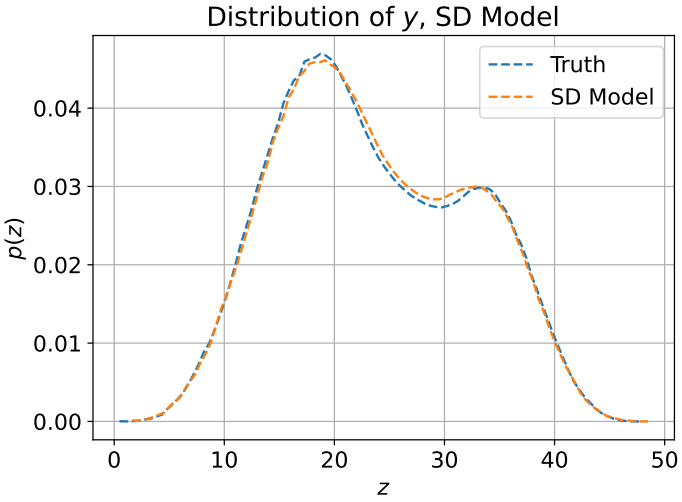

Figure A.5.4: Distribution function of z inferred from SD model, Lorenz 63 system.

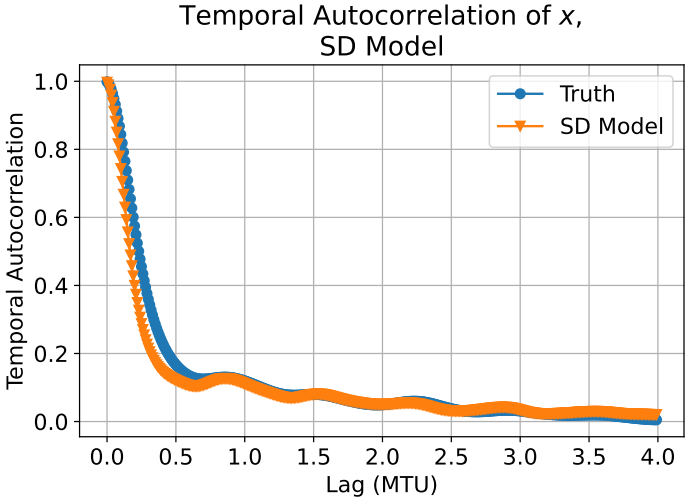

Figure A.5.5: Temporal autocorrelation function of x inferred from SD model, Lorenz 63 system.

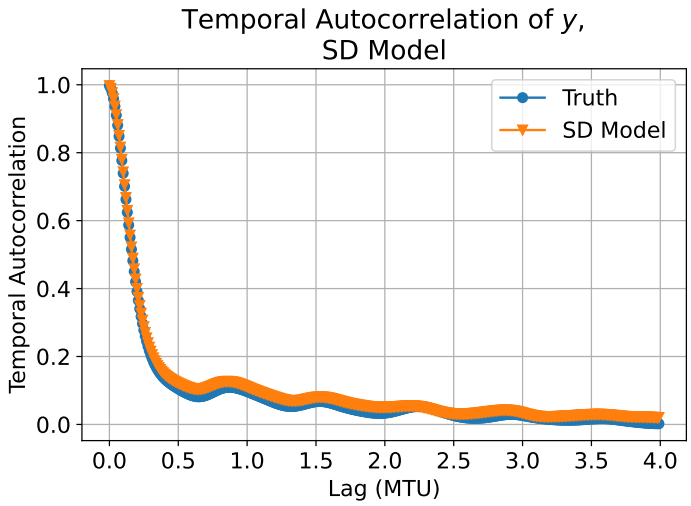

Figure A.5.6: Temporal autocorrelation function of y inferred from SD model, Lorenz 63 system.

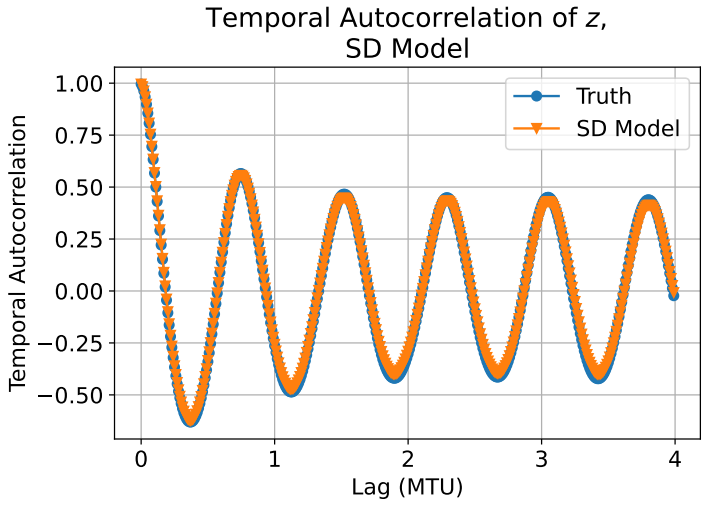

Figure A.5.7: Temporal autocorrelation function of z inferred from SD model, Lorenz 63 system.

B Monoscale Lorenz 96

B.1 Deterministic Model

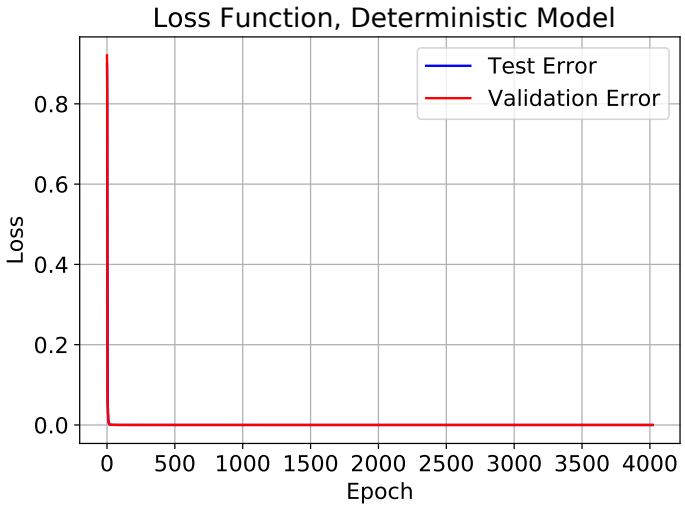

Figure B.1.1: Loss function when training Deterministic model, monoscale Lorenz 96 system.

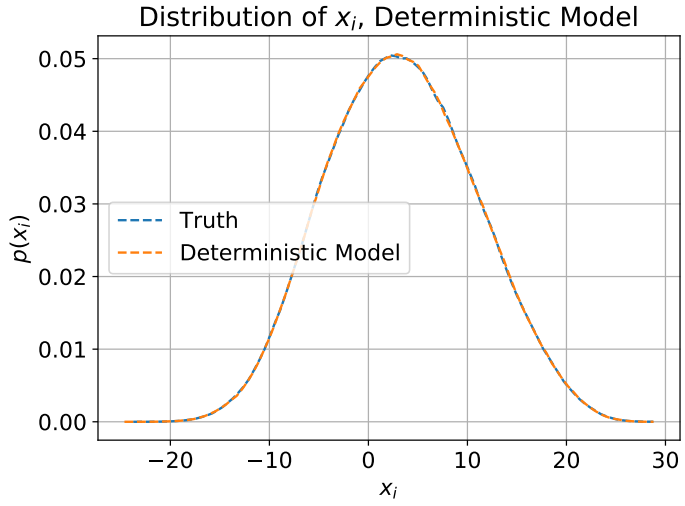

Figure B.1.2: Distribution function of x_i inferred from Deterministic model, monoscale Lorenz 96 system.

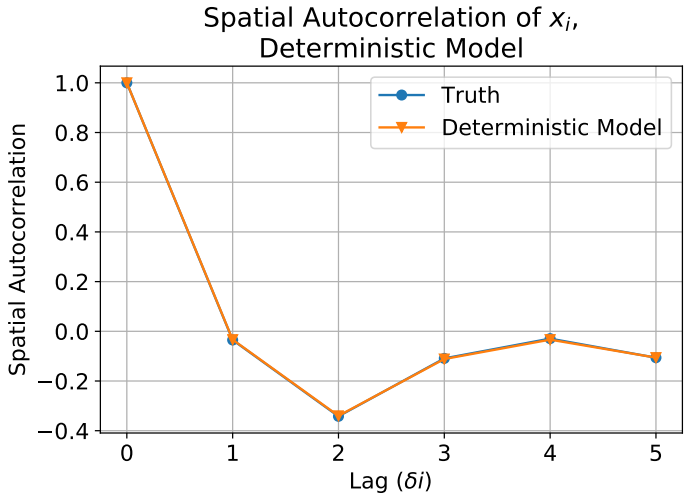

Figure B.1.3: Spatial autocorrelation function inferred from Deterministic model, monoscale Lorenz 96 system.

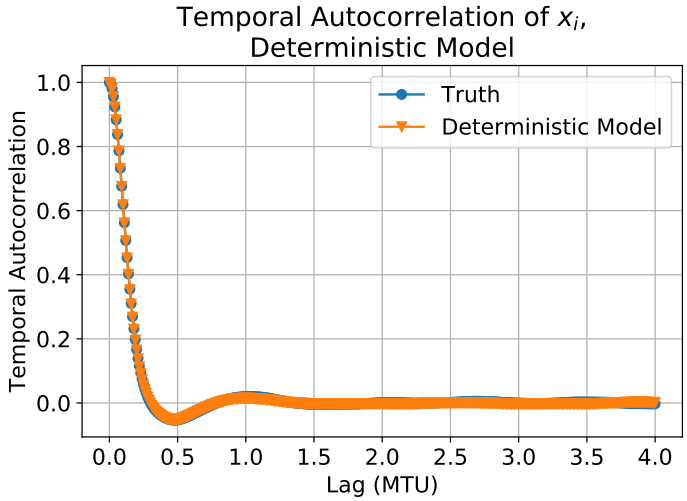

Figure B.1.4: Temporal autocorrelation function inferred from Deterministic model, monoscale Lorenz 96 system.

B.2 ML Model

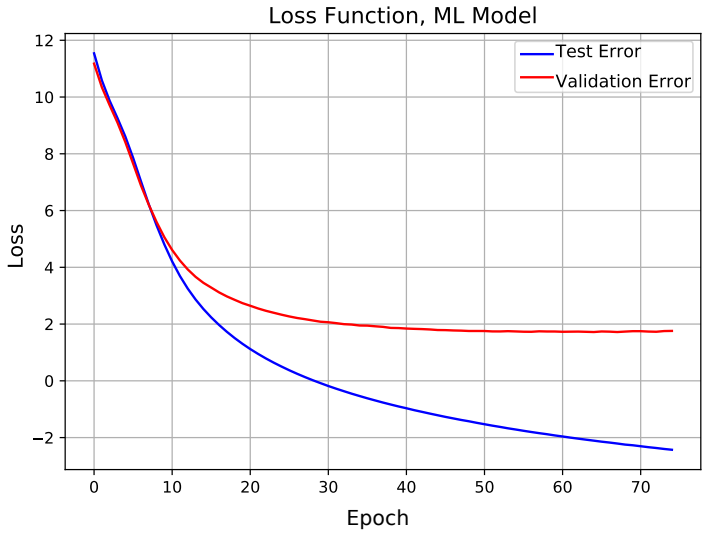

Figure B.2.1: Loss function when training ML model, monoscale Lorenz 96 system.

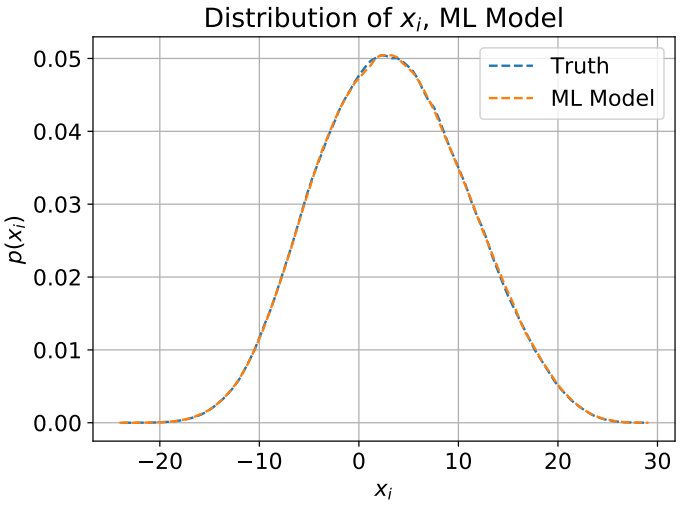

Figure B.2.2: Distribution function of x_i inferred from ML model, monoscale Lorenz 96 system.

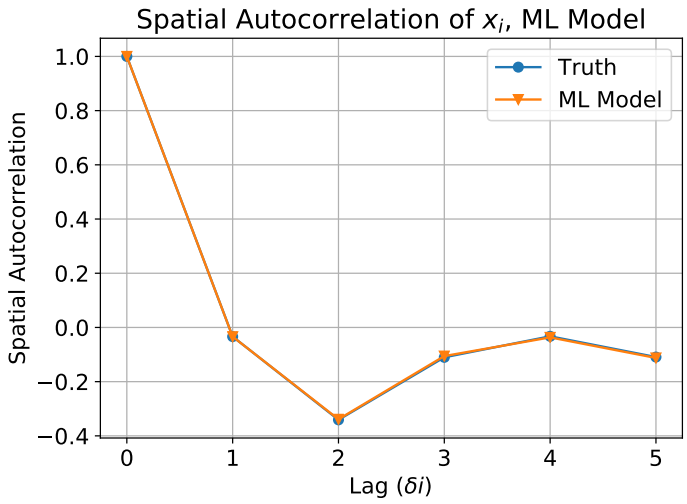

Figure B.2.3: Spatial autocorrelation function inferred from ML model, monoscale Lorenz 96 system.

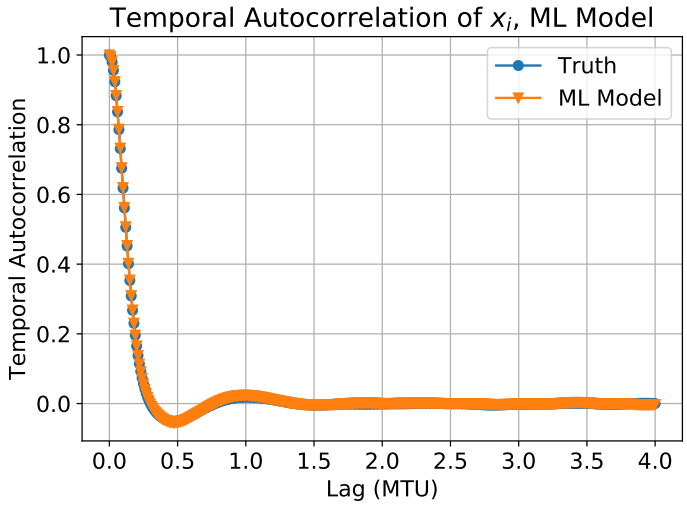

Figure B.2.4: Temporal autocorrelation function inferred from ML model, monoscale Lorenz 96 system.

B.3 MD Model

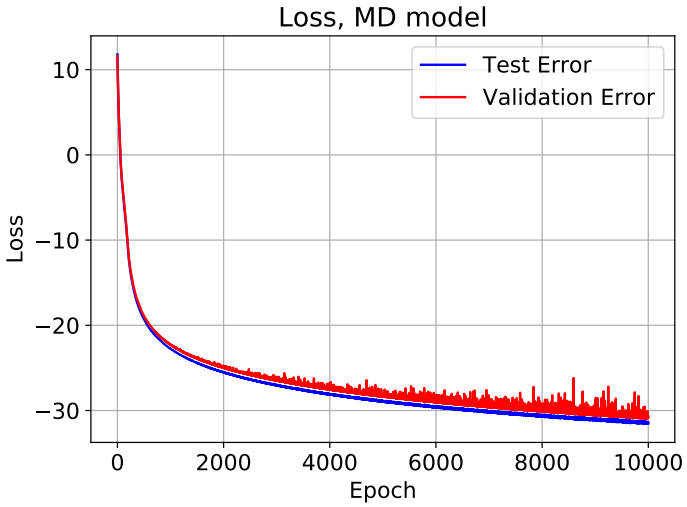

Figure B.3.1: Loss function when training MD model, monoscale Lorenz 96 system.

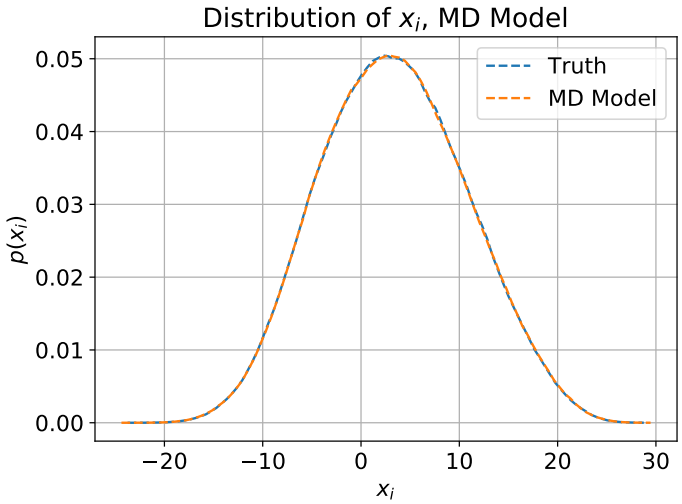

Figure B.3.2: Distribution function of x_i inferred from MD model, monoscale Lorenz 96 system.

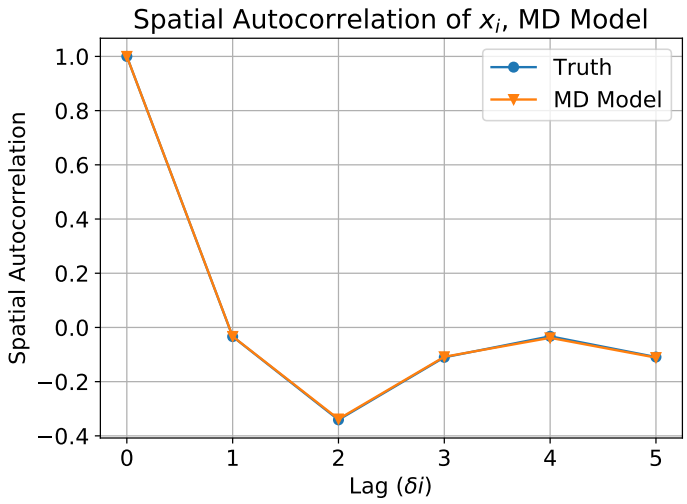

Figure B.3.3: Spatial autocorrelation function inferred from MD model, monoscale Lorenz 96 system.

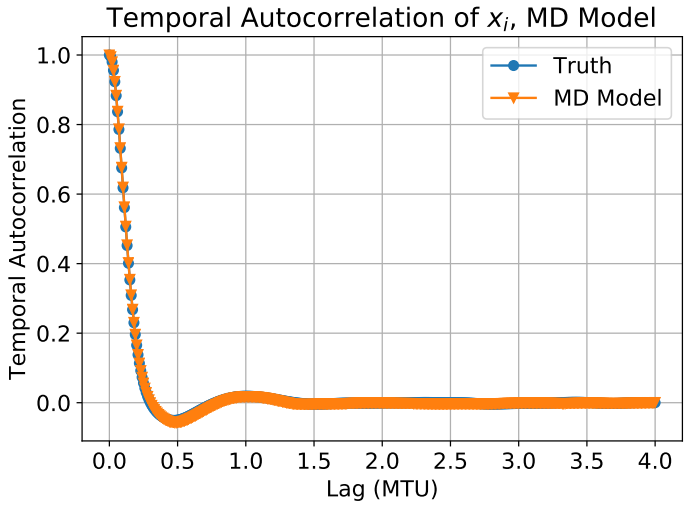

Figure B.3.4: Temporal autocorrelation function inferred from MD model, monoscale Lorenz 96 system.

B.4 SL Model

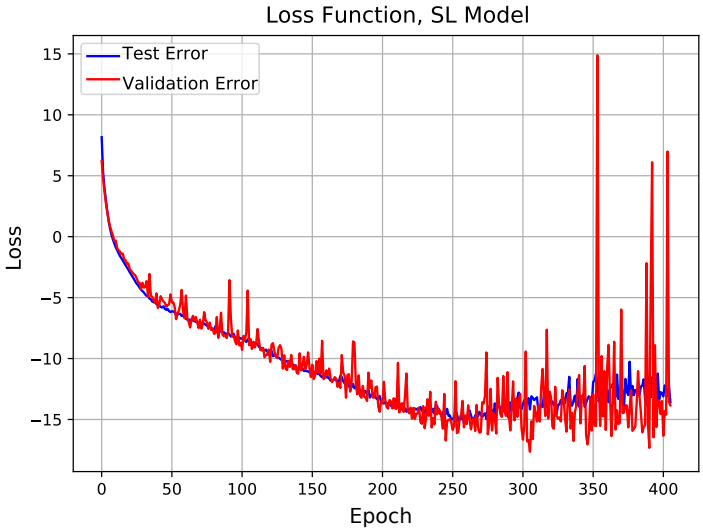

Figure B.4.1: Loss function when training SL model, monoscale Lorenz 96 system.

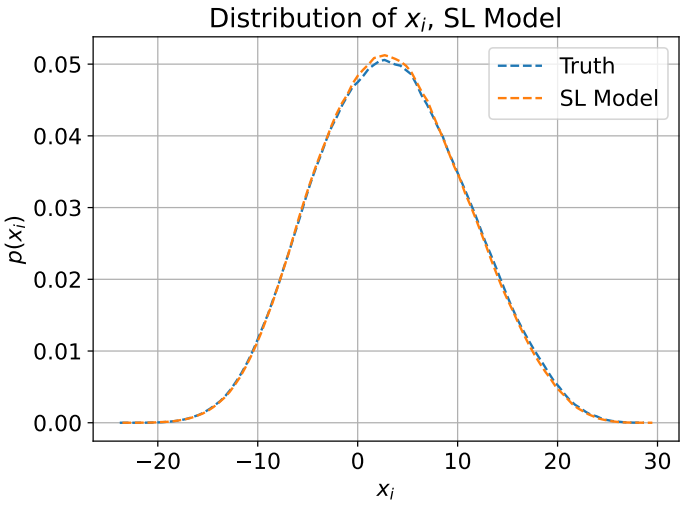

Figure B.4.2: Distribution function of x_i inferred from SL model, monoscale Lorenz 96 system.

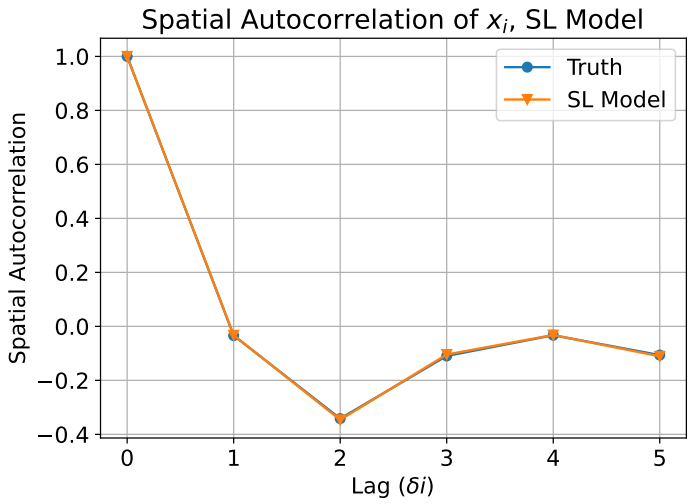

Figure B.4.3: Spatial autocorrelation function inferred from SL model, monoscale Lorenz 96 system.

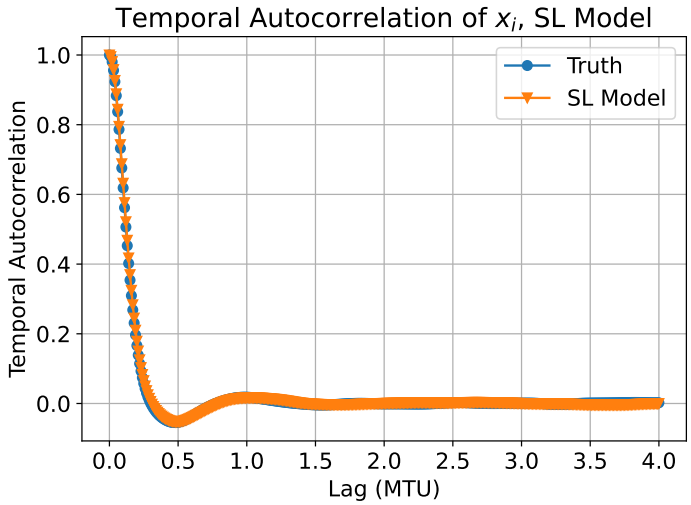

Figure B.4.4: Temporal autocorrelation function inferred from SL model, monoscale Lorenz 96 system.

B.5 SD Model

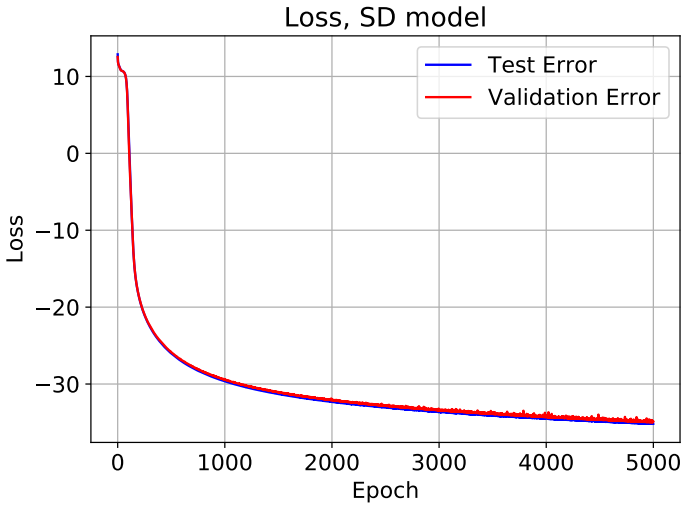

Figure B.5.1: Loss function when training SD model, monoscale Lorenz 96 system.

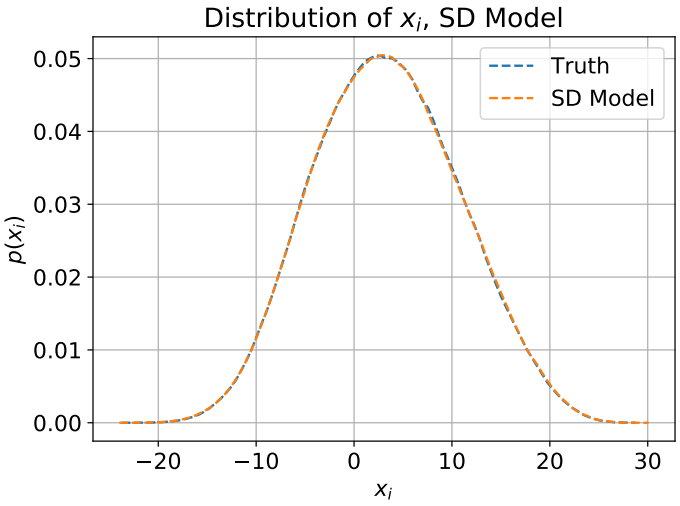

Figure B.5.2: Distribution function of x_i inferred from SD model, monoscale Lorenz 96 system.

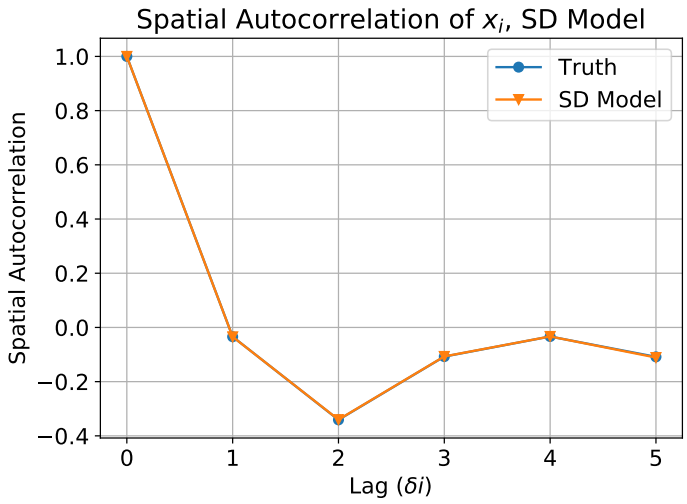

Figure B.5.3: Spatial autocorrelation function inferred from SD model, monoscale Lorenz 96 system.

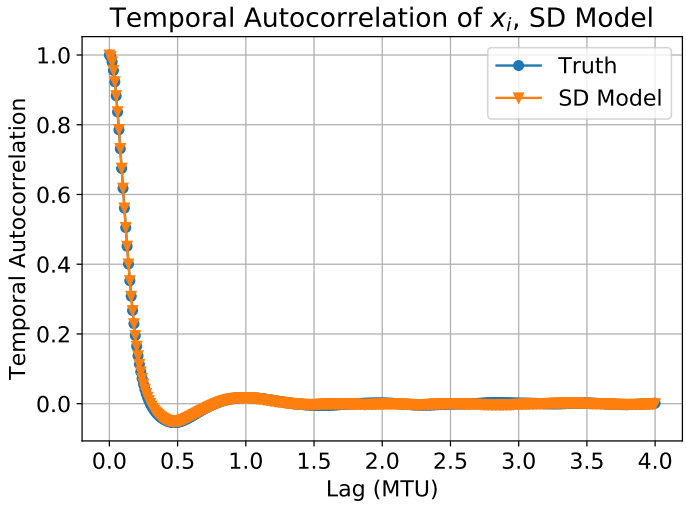

Figure B.5.4: Temporal autocorrelation function inferred from SD model, monoscale Lorenz 96 system.

C Multiscale Lorenz 96, $h = 1$, $b = c = 10$, and $F = 20$

C.1 Deterministic Model

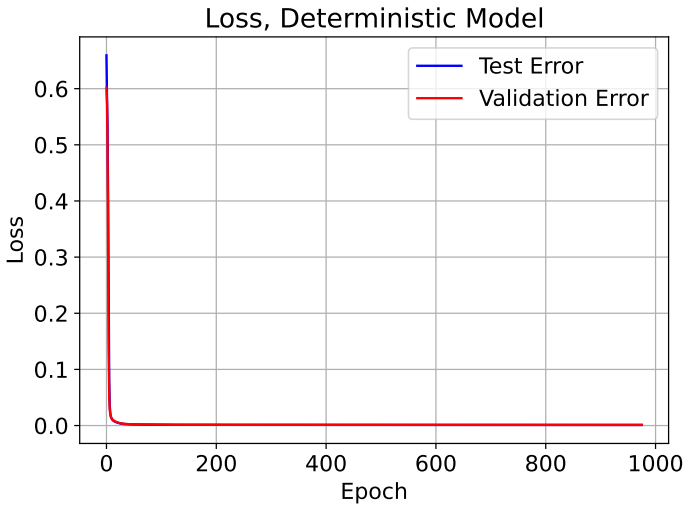

Figure C.1.1: Loss function for Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$, $F = 20$.

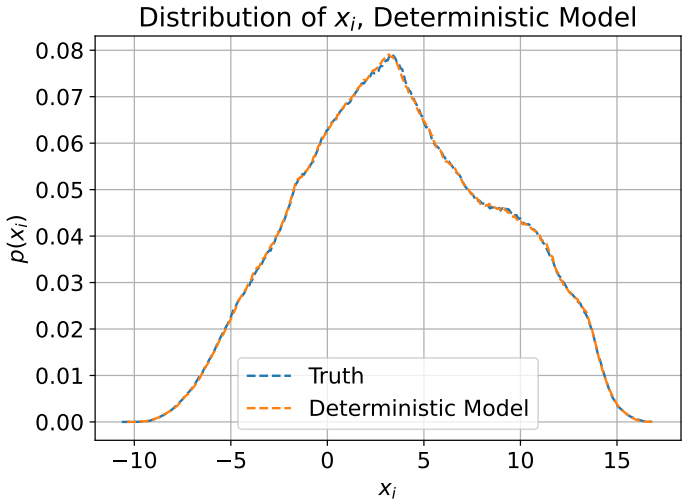

Figure C.1.2: Distribution function of x_i inferred from Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$, $F = 20$.

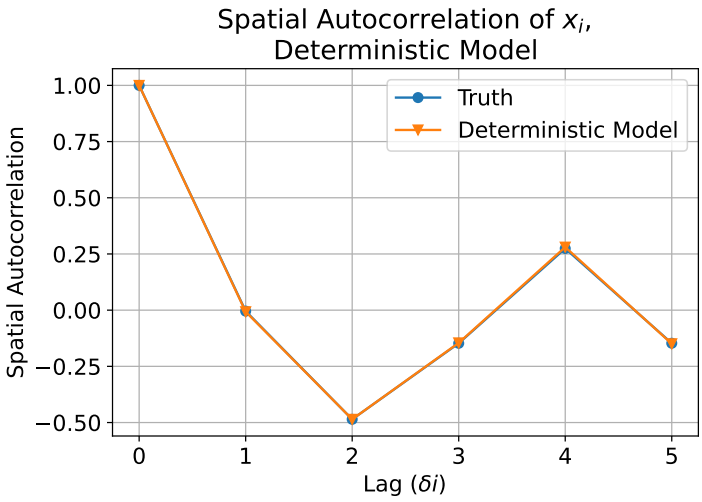

Figure C.1.3: Spatial autocorrelation function inferred from Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$, $F = 20$.

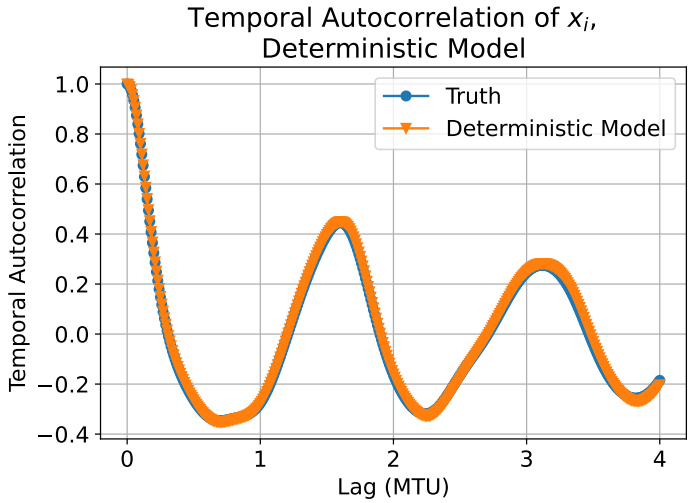

Figure C.1.4: Temporal autocorrelation function inferred from Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = c = 10$, $F = 20$.

C.2 ML Model

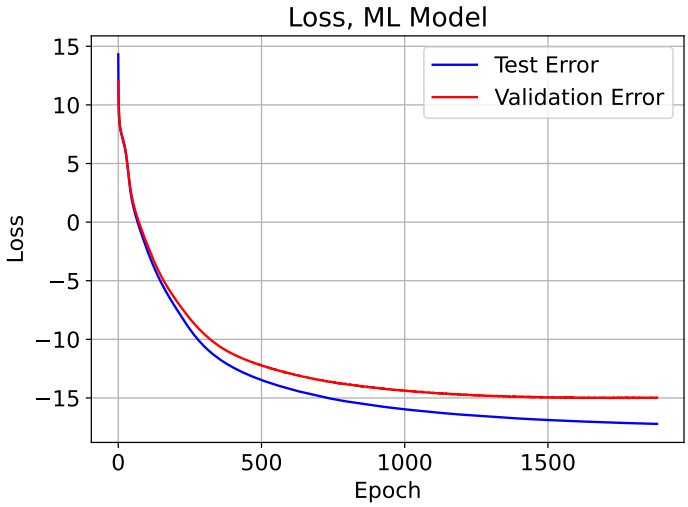

Figure C.2.1: Loss function for ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

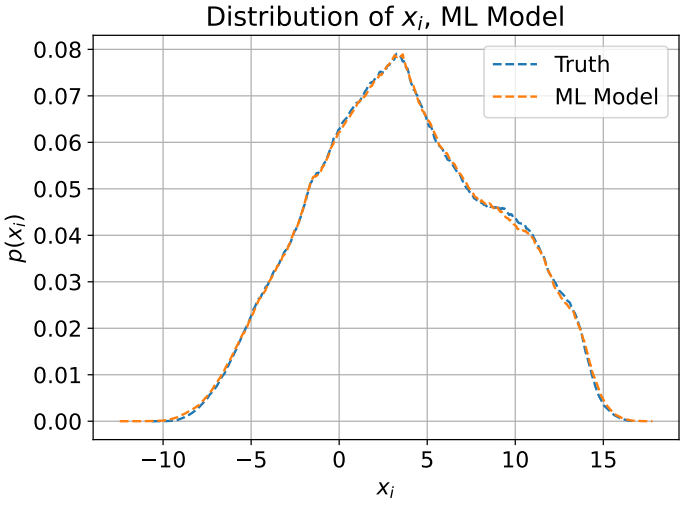

Figure C.2.2: Distribution function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

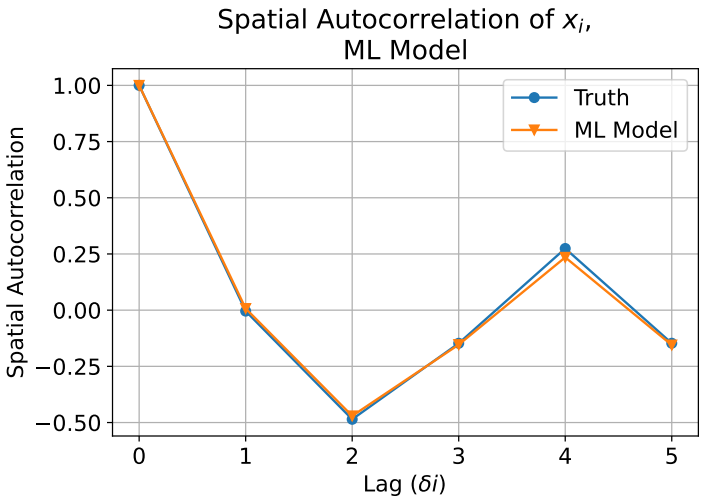

Figure C.2.3: Spatial autocorrelation function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

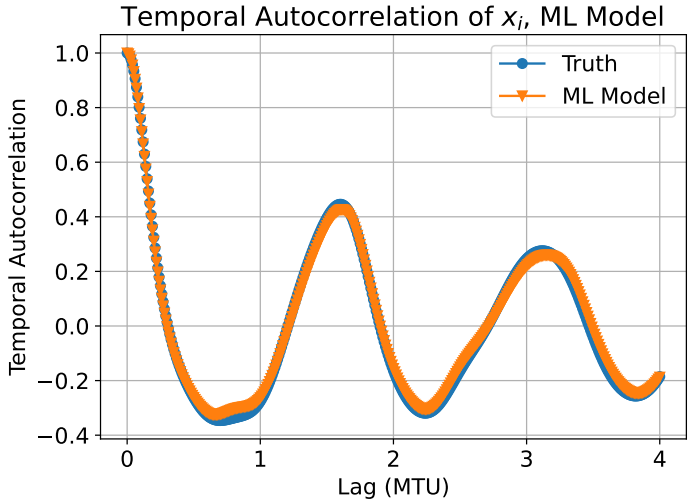

Figure C.2.4: Temporal autocorrelation function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

C.3 MD Model

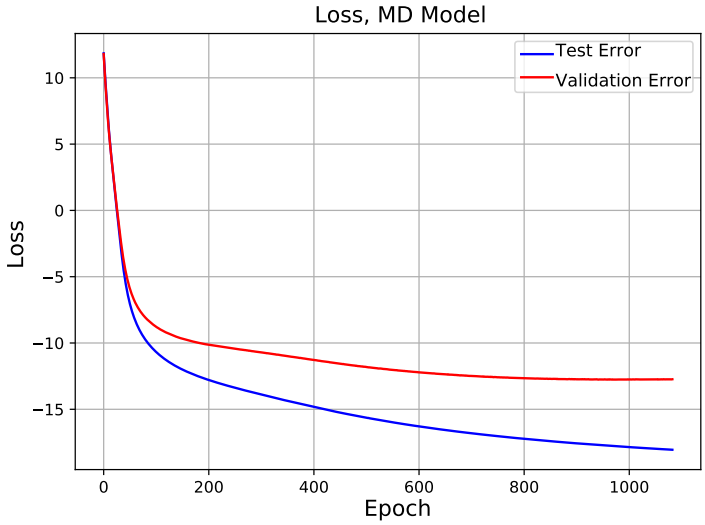

Figure C.3.1: Loss function for MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

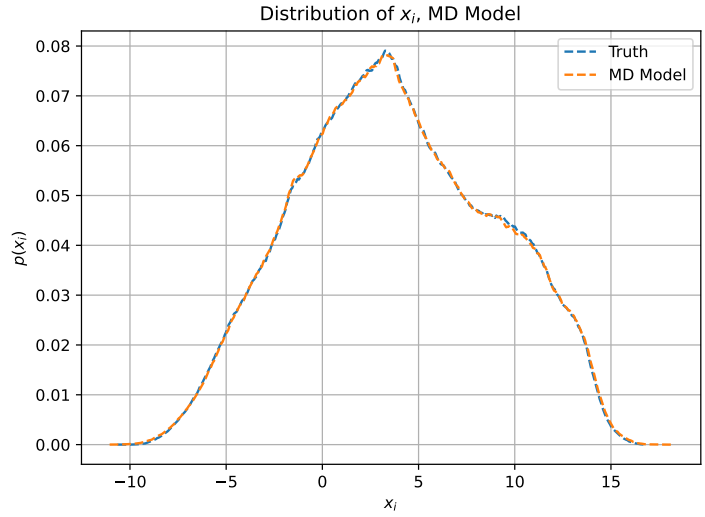

Figure C.3.2: Distribution function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

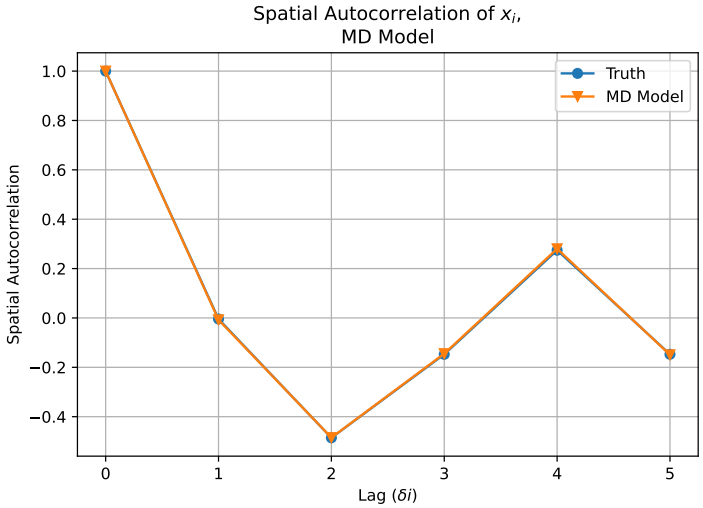

Figure C.3.3: Spatial autocorrelation function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

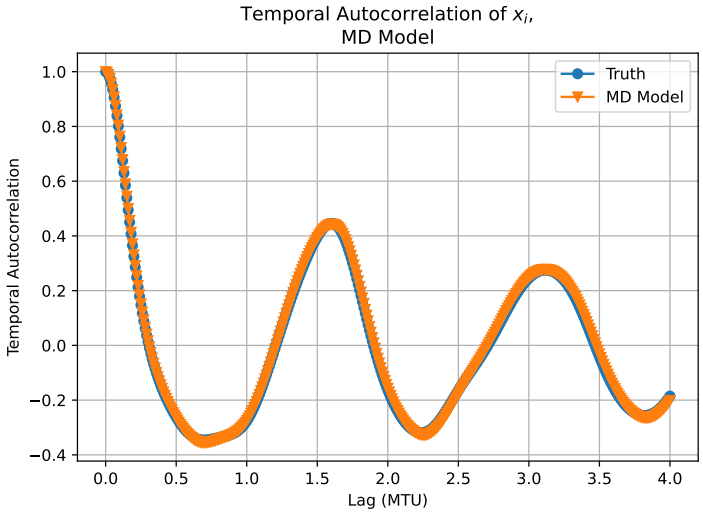

Figure C.3.4: Temporal autocorrelation function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

C.4 SL Model

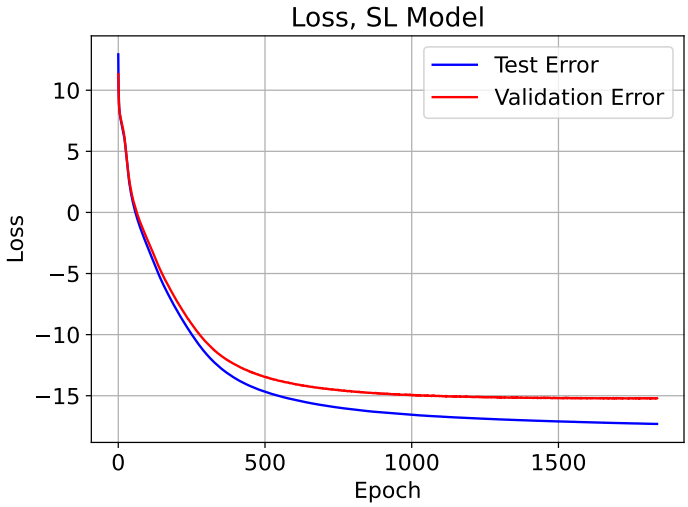

Figure C.4.1: Loss function for SL model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

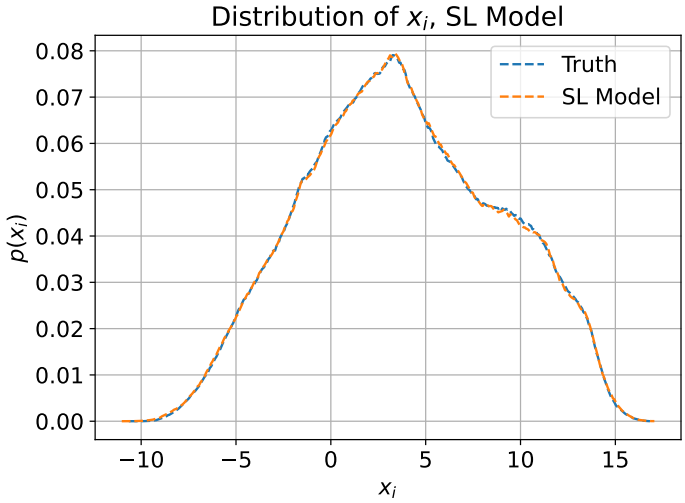

Figure C.4.2: Distribution function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

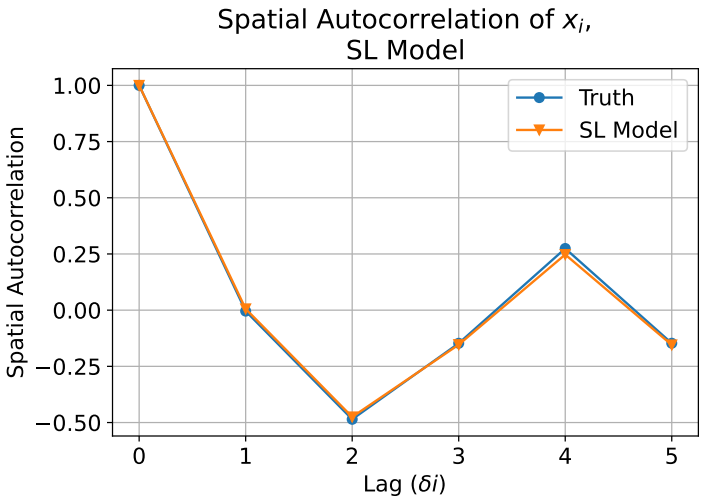

Figure C.4.3: Spatial autocorrelation function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

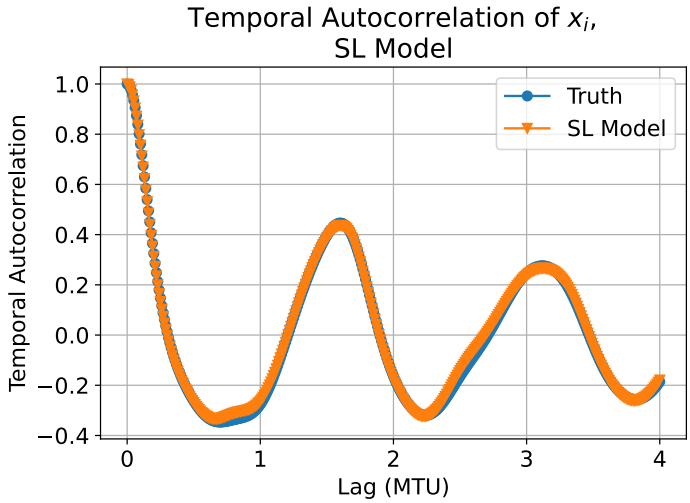

Figure C.4.4: Temporal autocorrelation function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

C.5 SD Model

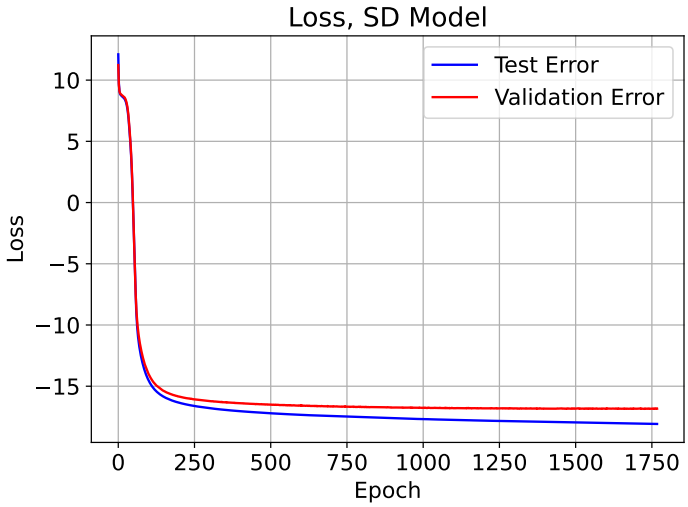

Figure C.5.1: Loss function for SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

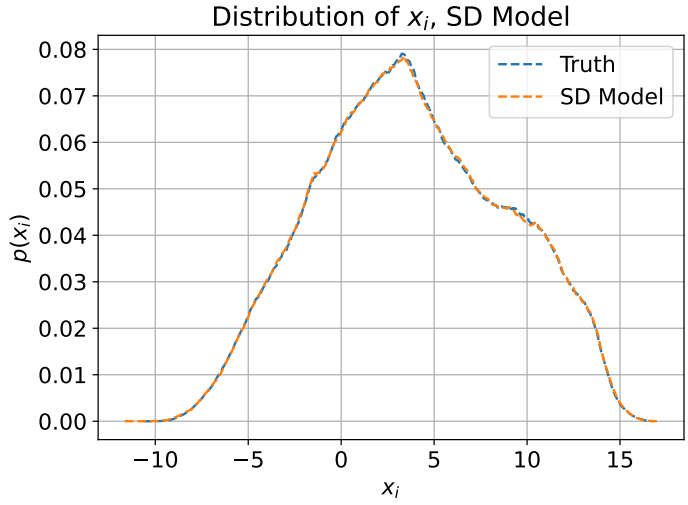

Figure C.5.2: Distribution function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

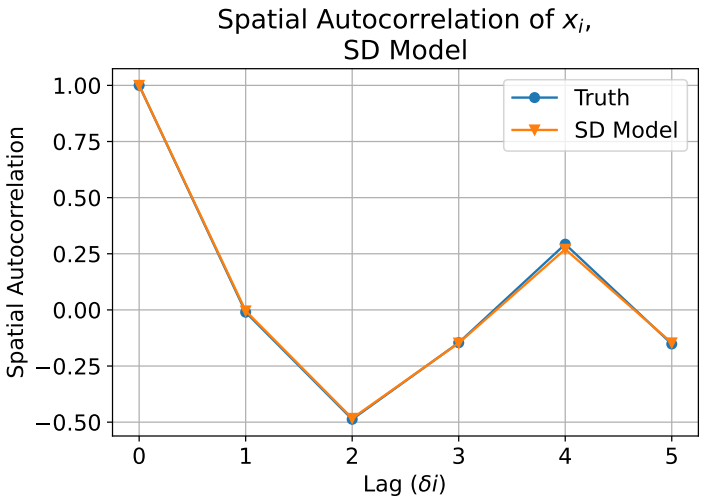

Figure C.5.3: Spatial autocorrelation function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

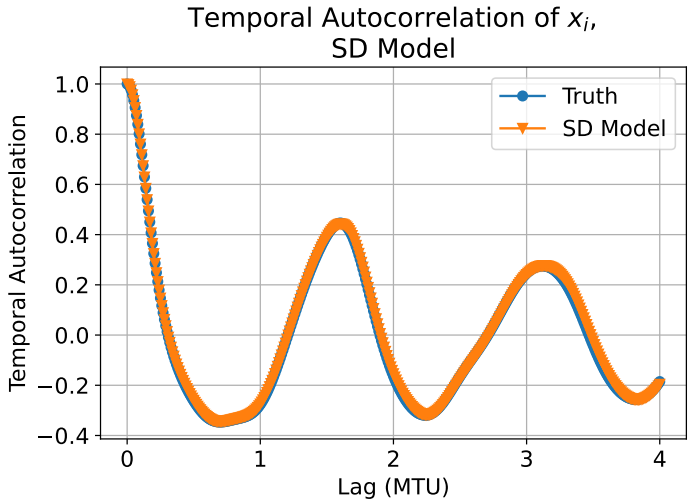

Figure C.5.4: Temporal autocorrelation function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

D Multiscale Lorenz 96, $h = 1$, $b = 10$, $c = 4$, and $F = 20$

D.1 Deterministic Model

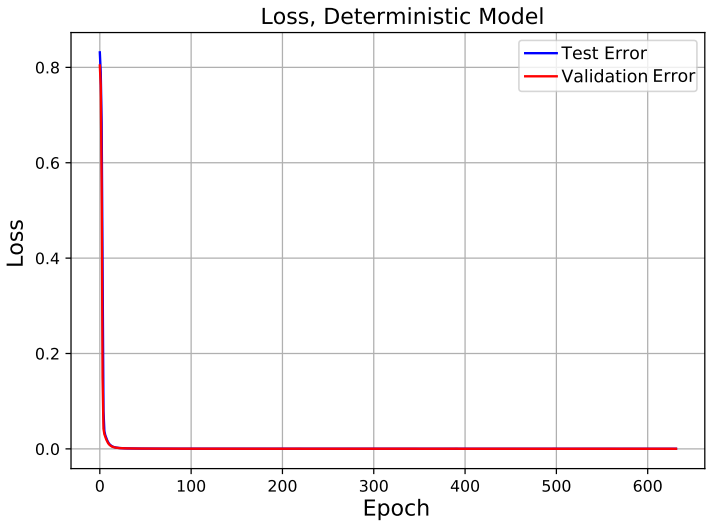

Figure D.1.1: Loss function for Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

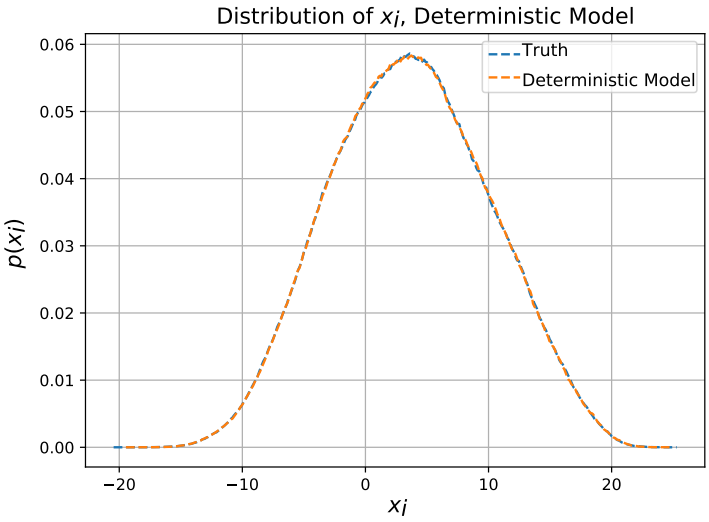

Figure D.1.2: Distribution function of x_i inferred from Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 10$ and $F = 20$.

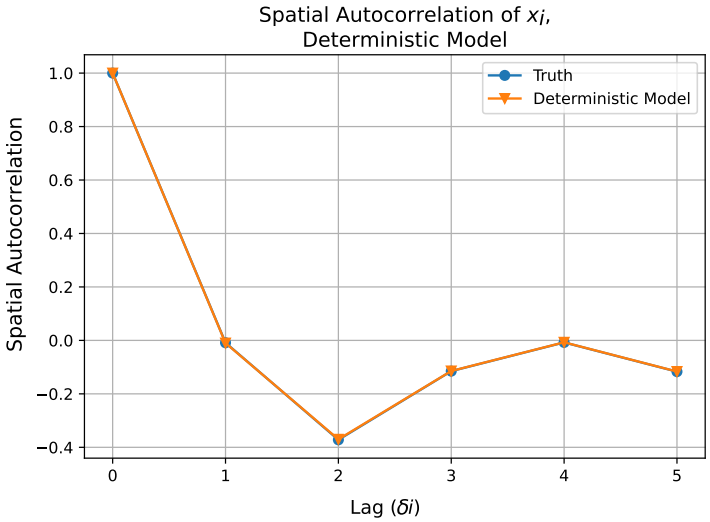

Figure D.1.3: Spatial autocorrelation function inferred from Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = 10$ $c = 4$ and $F = 20$.

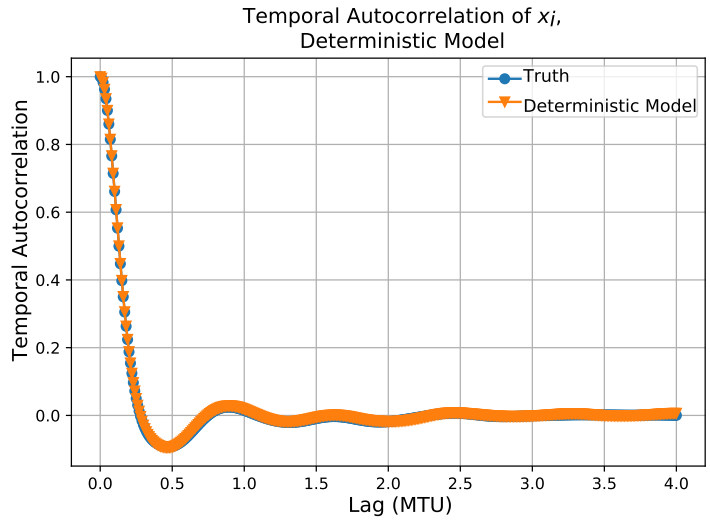

Figure D.1.4: Temporal autocorrelation function inferred from Deterministic model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

D.2 ML Model

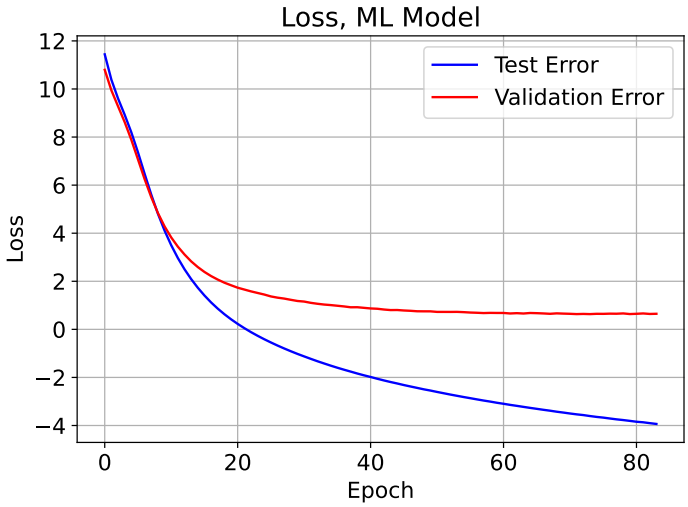

Figure D.2.1: Loss function for ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

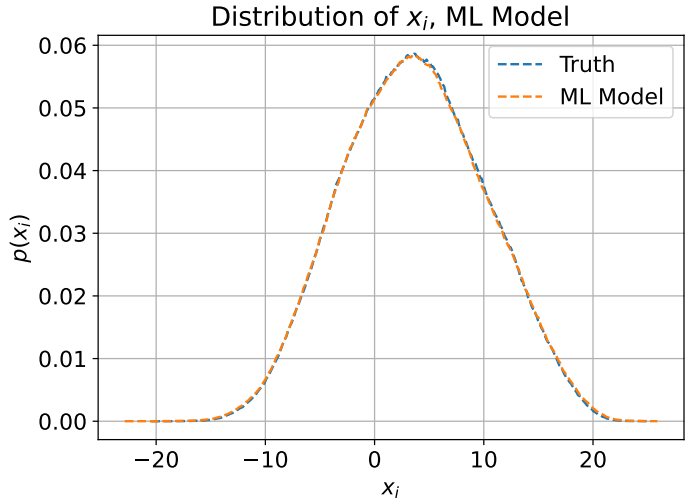

Figure D.2.2: Distribution function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

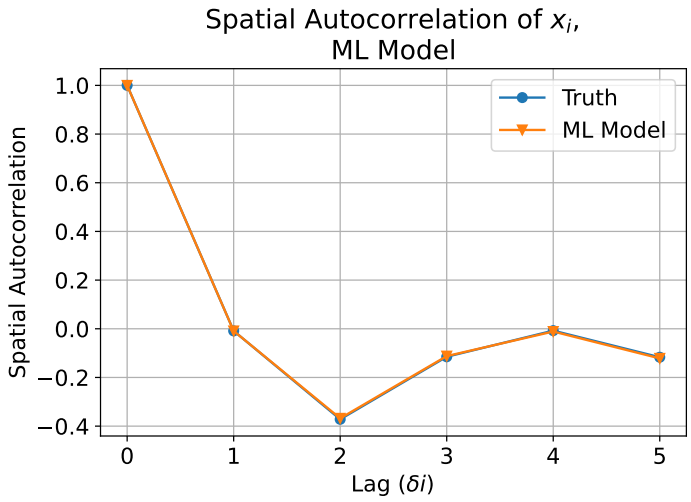

Figure D.2.3: Spatial autocorrelation function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

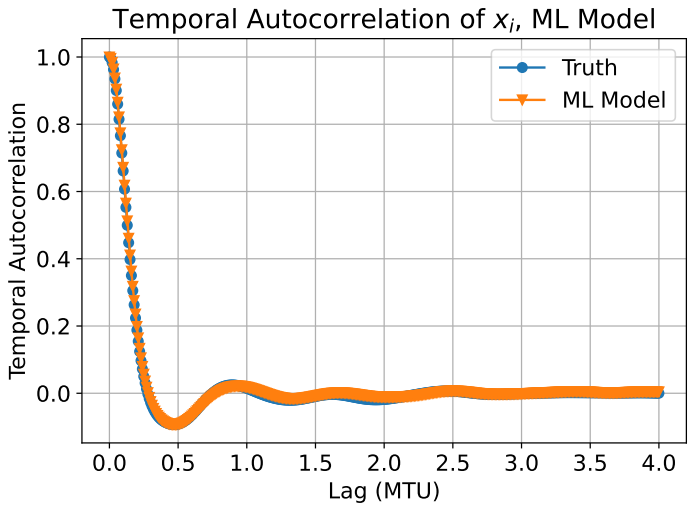

Figure D.2.4: Temporal autocorrelation function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

D.3 MD Model

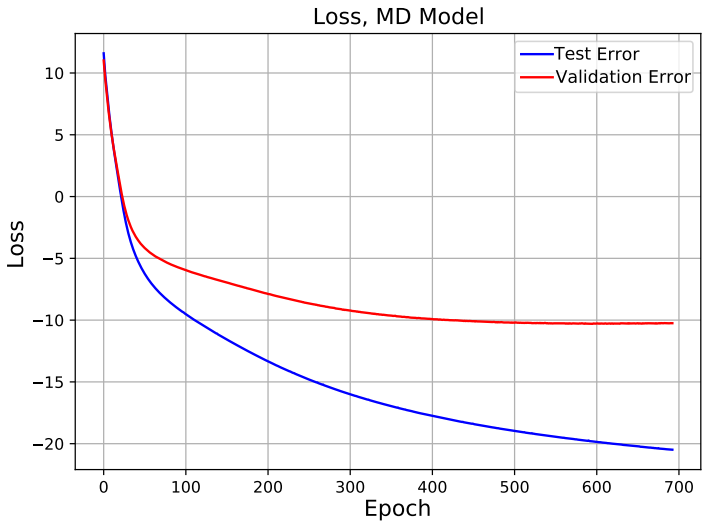

Figure D.3.1: Loss function for MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

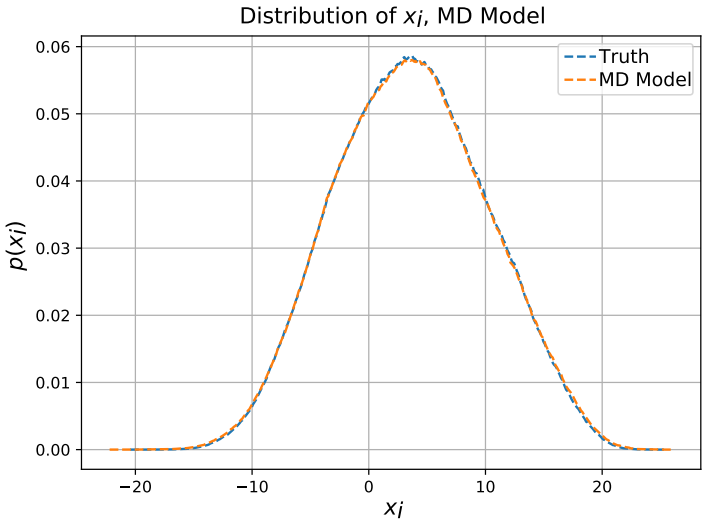

Figure D.3.2: Distribution function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

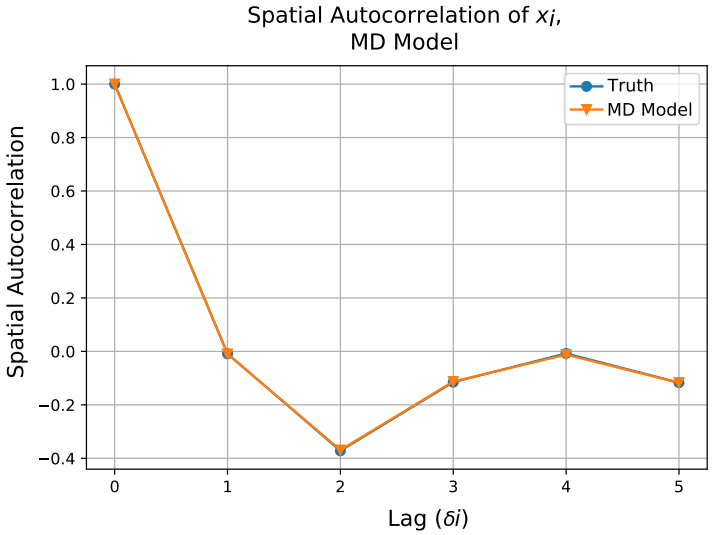

Figure D.3.3: Spatial autocorrelation function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

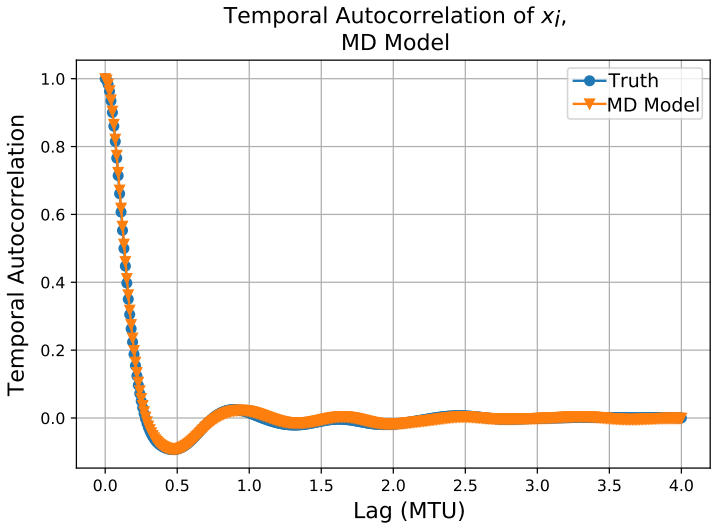

Figure D.3.4: Temporal autocorrelation function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

D.4 SL Model

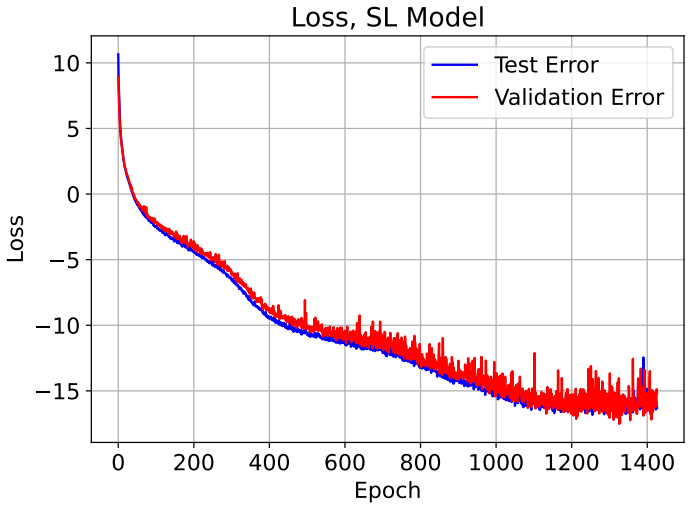

Figure D.4.1: Loss function for SL model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

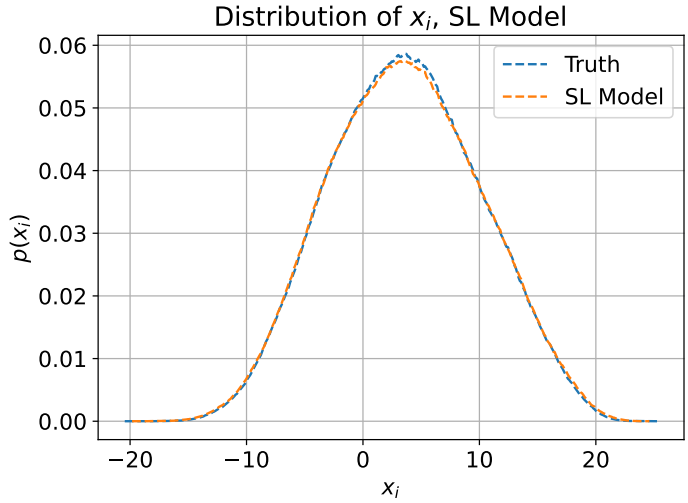

Figure D.4.2: Distribution function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

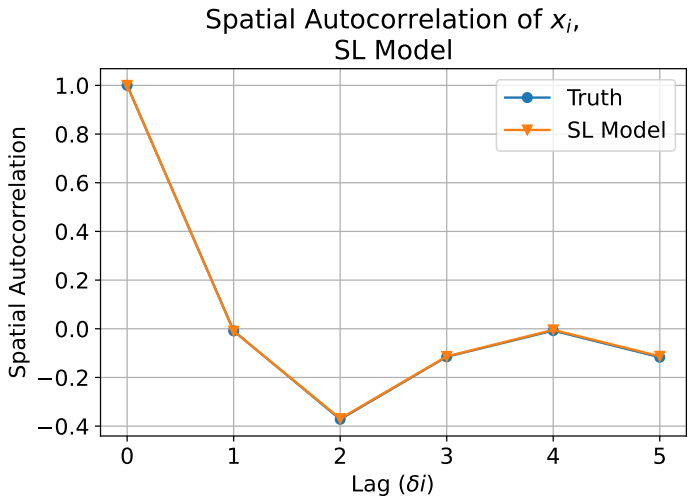

Figure D.4.3: Spatial autocorrelation function of x_i inferred from SL Model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

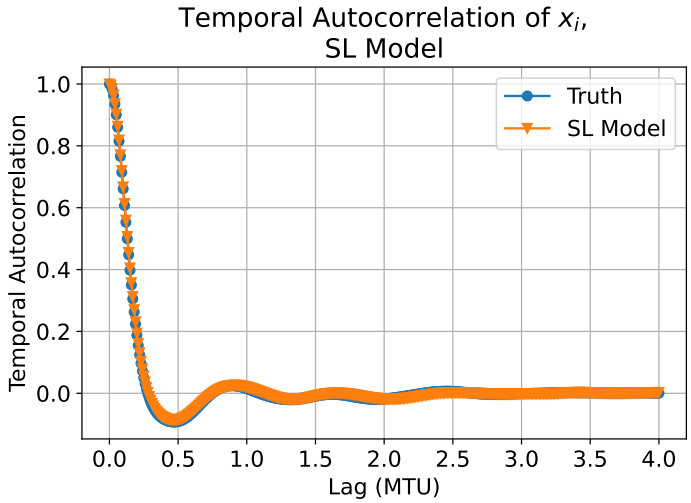

Figure D.4.4: Temporal autocorrelation function of x_i inferred from SL Model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

D.5 SD Model

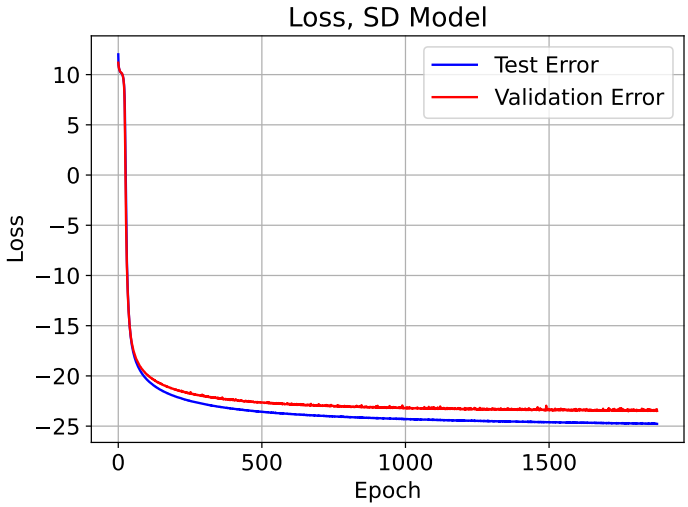

Figure D.5.1: Loss function for SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

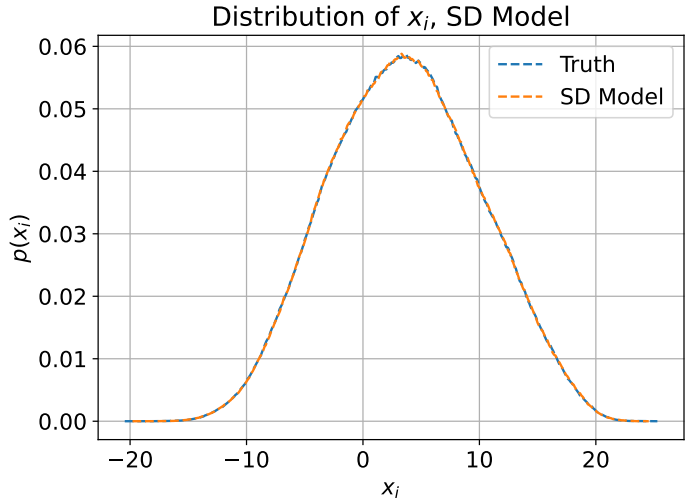

Figure D.5.2: Distribution function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

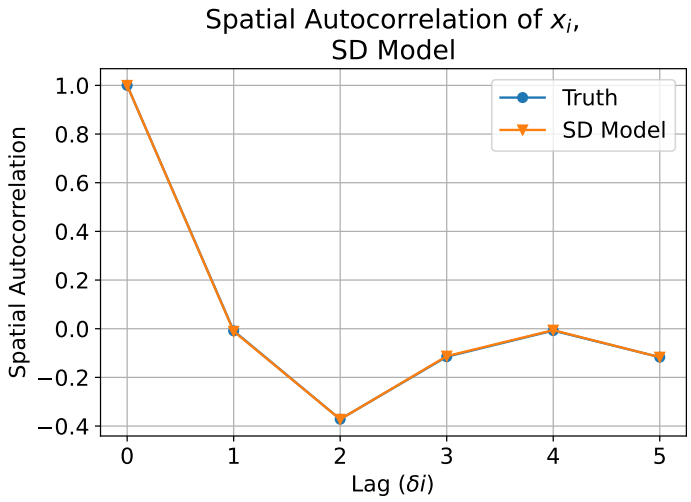

Figure D.5.3: Spatial autocorrelation function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

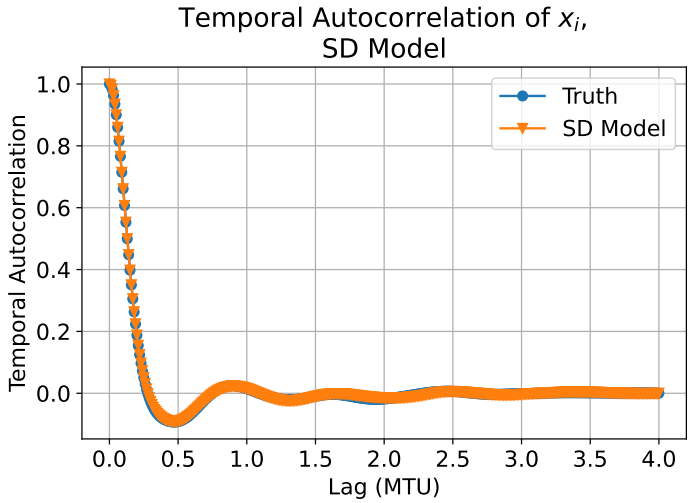

Figure D.5.4: Temporal autocorrelation function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 1$, $b = 10$, $c = 4$ and $F = 20$.

E Multiscale Lorenz 96, $h = 5$, $b = 10$, $c = 2$, and $F = 20$

E.1 Deterministic Model

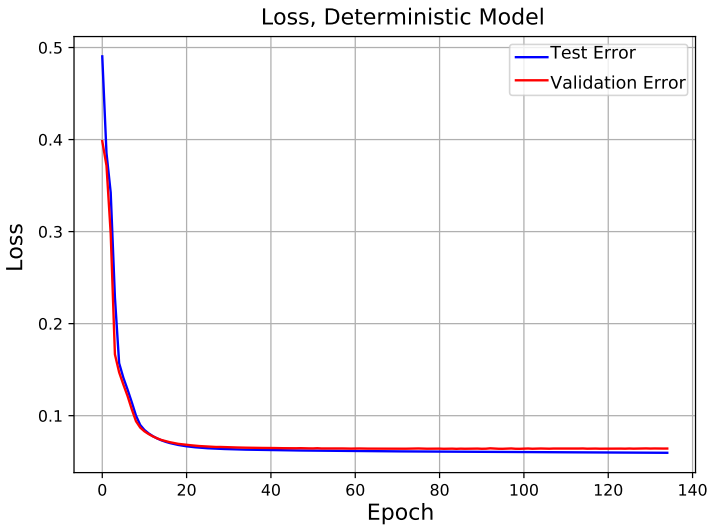

Figure E.1.1: Loss function for Deterministic model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

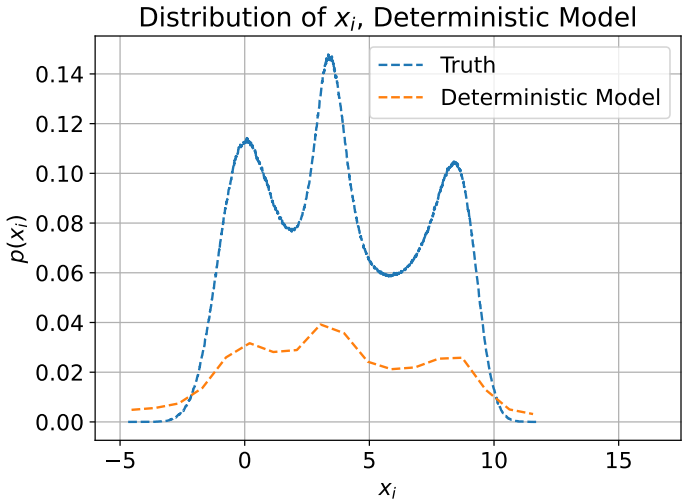

Figure E.1.2: Distribution function of x_i inferred from Deterministic model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

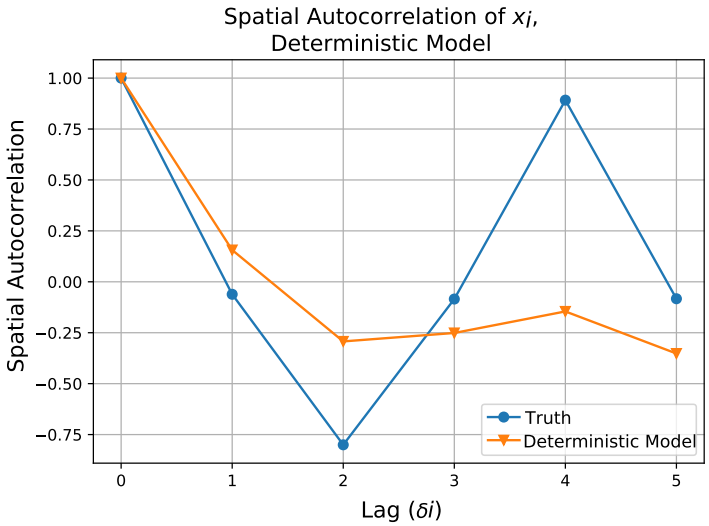

Figure E.1.3: Spatial autocorrelation function inferred from Deterministic model, multiscale Lorenz 96 system with $h = 5$, $b = 10$ $c = 2$, $F = 20$.

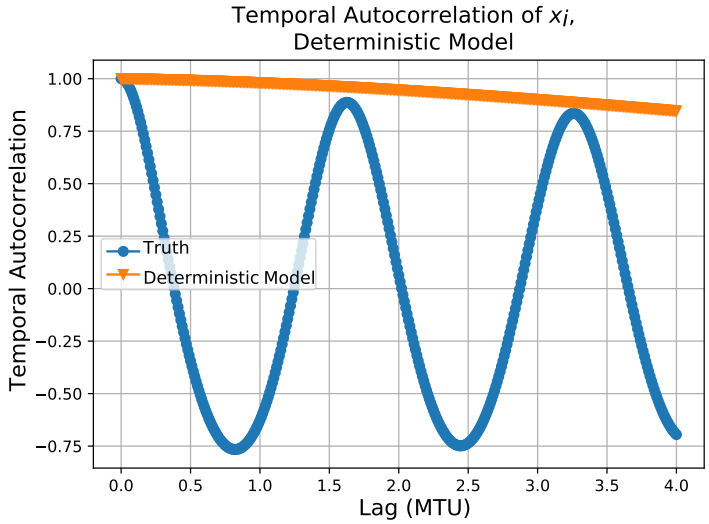

Figure E.1.4: Temporal autocorrelation function inferred from Deterministic model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$, $F = 20$.

E.2 ML Model

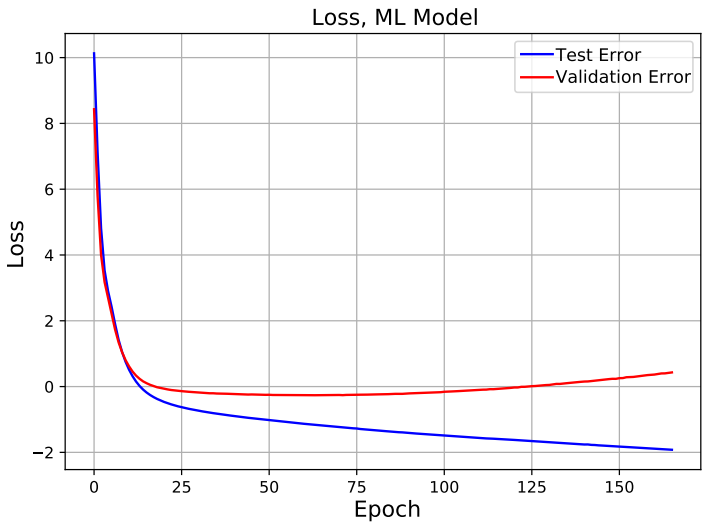

Figure E.2.1: Loss function for ML model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

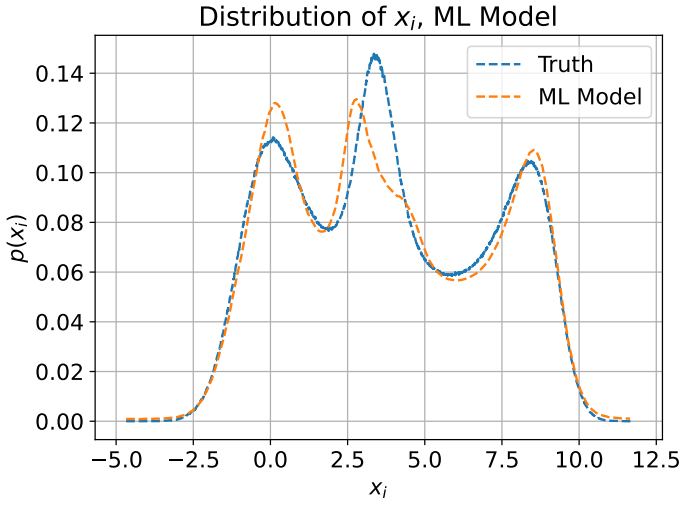

Figure E.2.2: Distribution function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

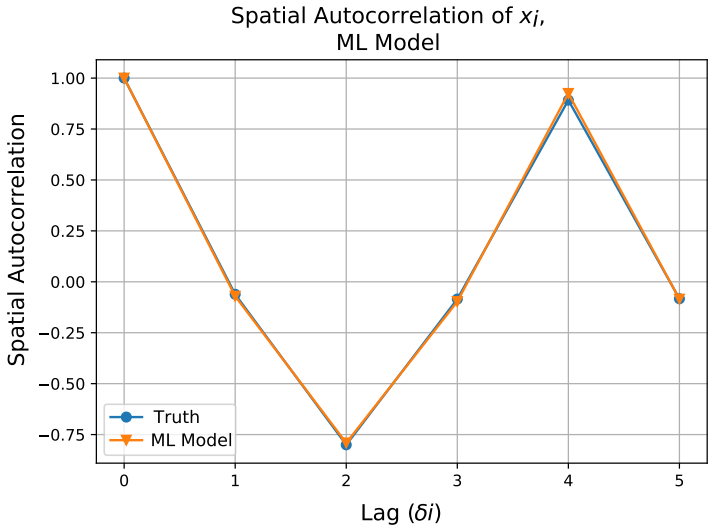

Figure E.2.3: Spatial autocorrelation function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

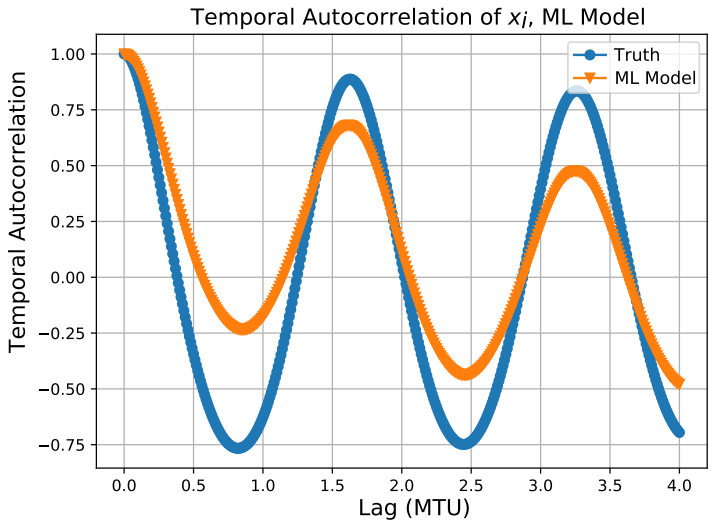

Figure E.2.4: Temporal autocorrelation function of x_i inferred from ML model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

E.3 MD Model

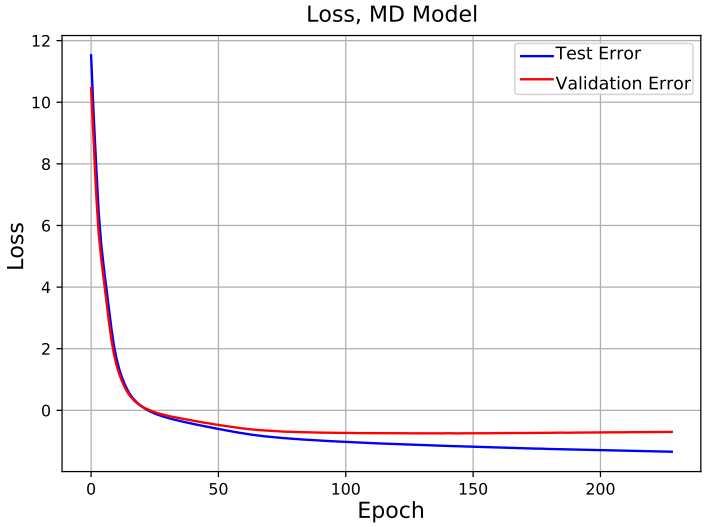

Figure E.3.1: Loss function for MD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

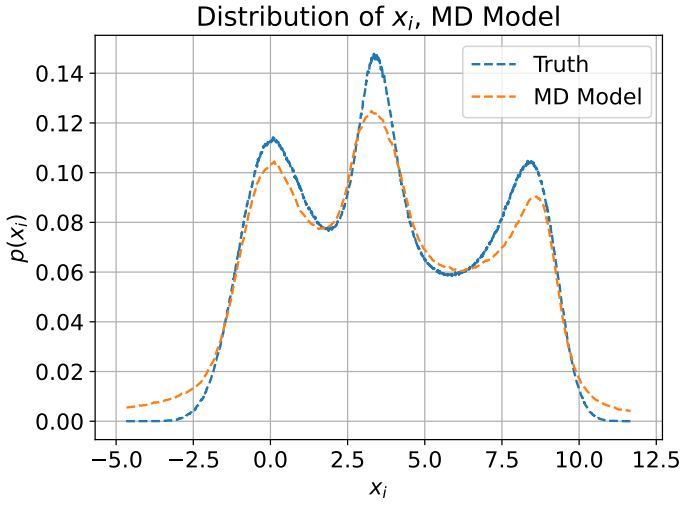

Figure E.3.2: Distribution function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

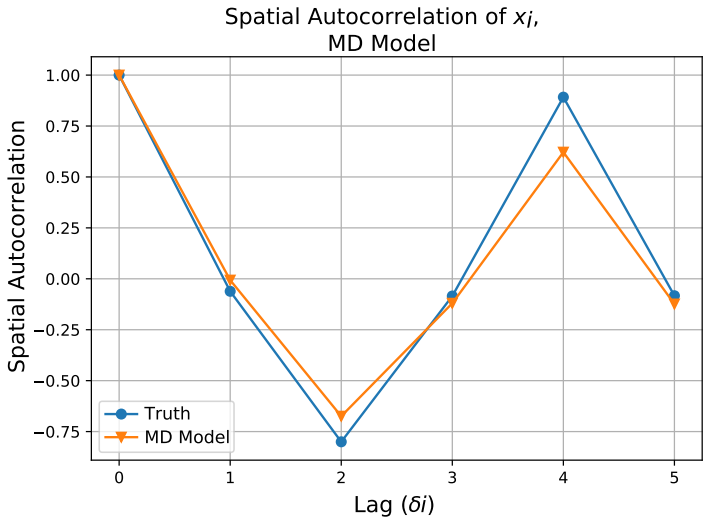

Figure E.3.3: Spatial autocorrelation function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

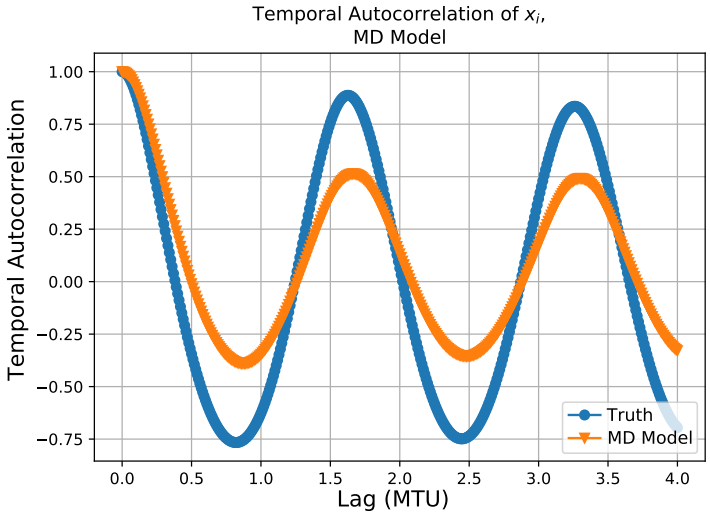

Figure E.3.4: Temporal autocorrelation function of x_i inferred from MD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

E.4 SL Model

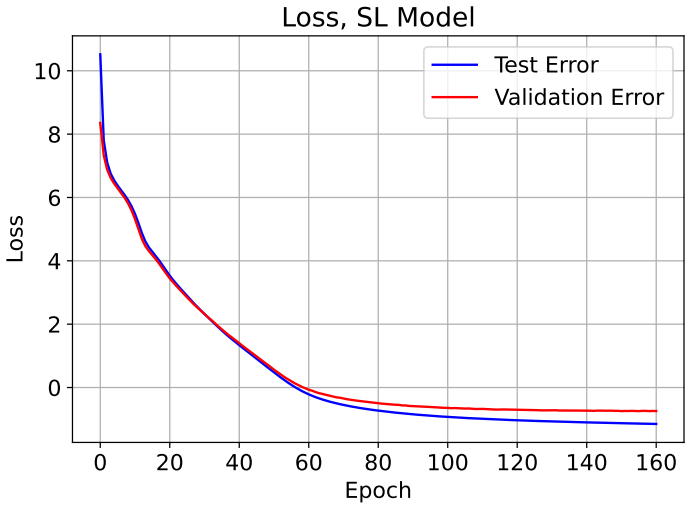

Figure E.4.1: Loss function for SL model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

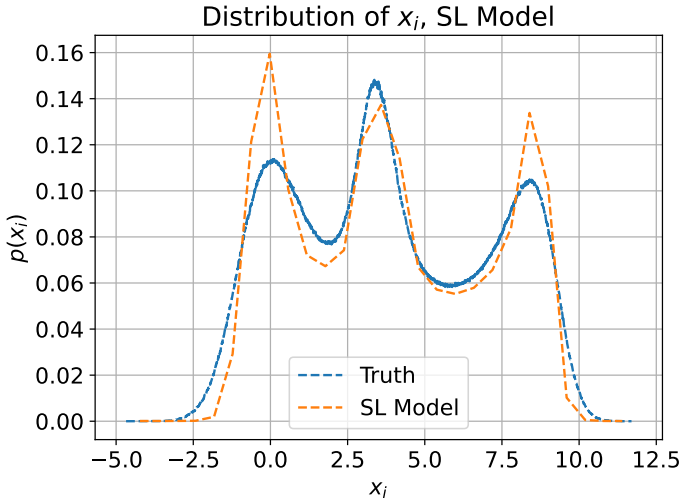

Figure E.4.2: Distribution function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

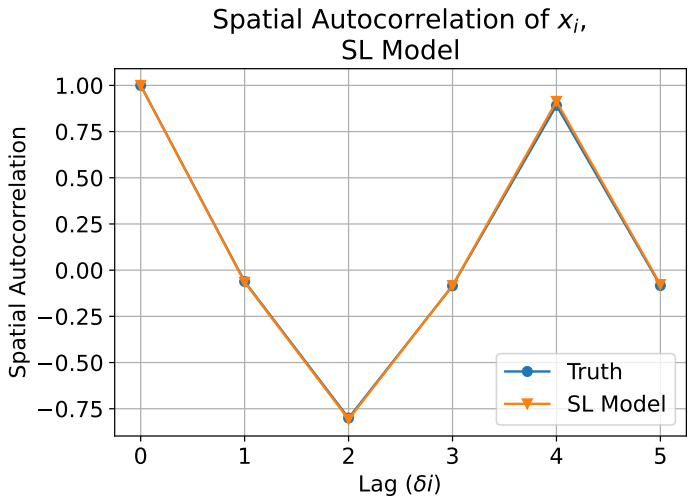

Figure E.4.3: Spatial autocorrelation function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

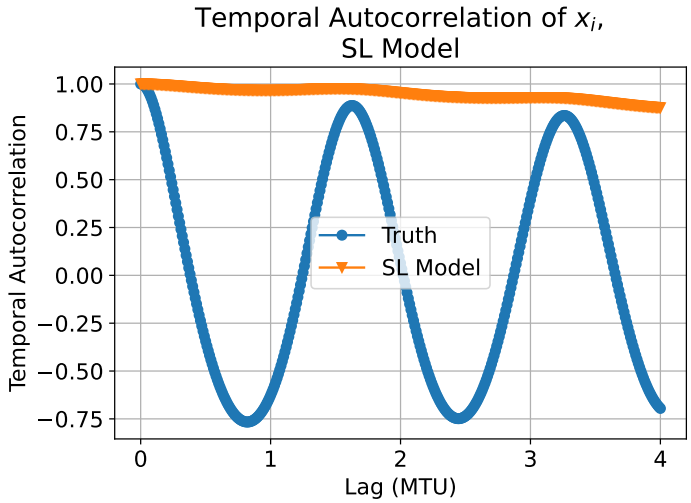

Figure E.4.4: Temporal autocorrelation function of x_i inferred from SL model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

E.5 SD Model

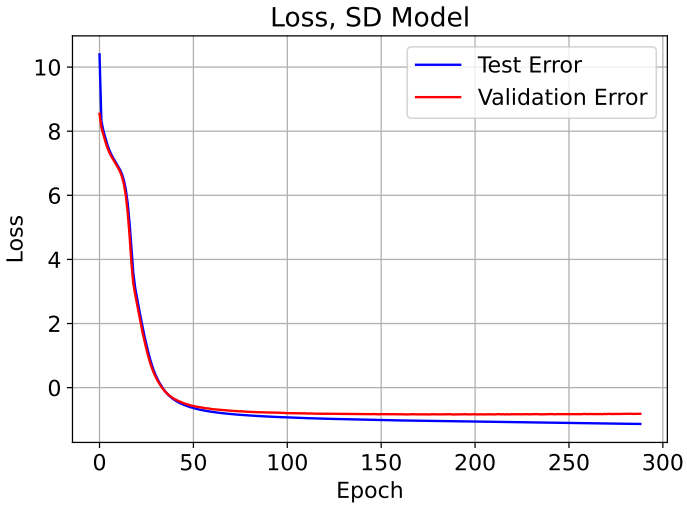

Figure E.5.1: Loss function for SD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

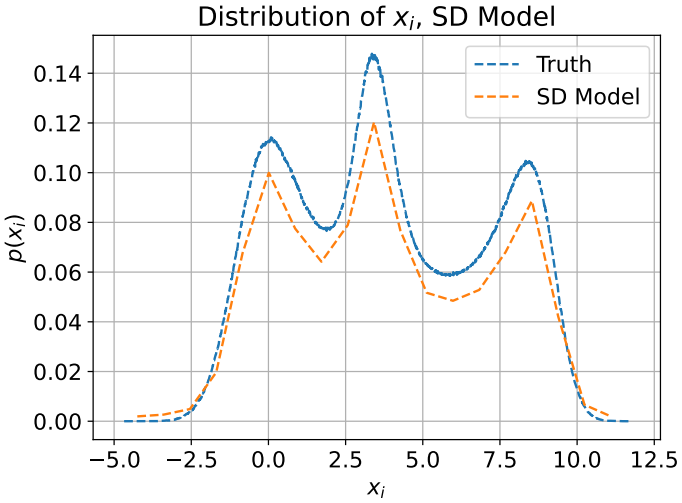

Figure E.5.2: Distribution function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

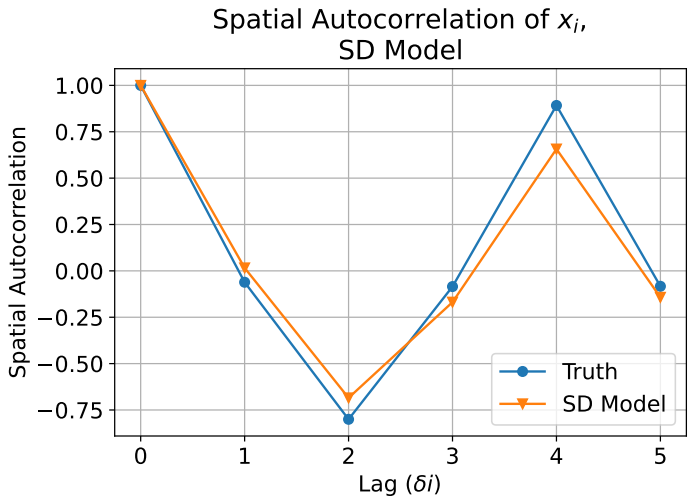

Figure E.5.3: Spatial autocorrelation function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

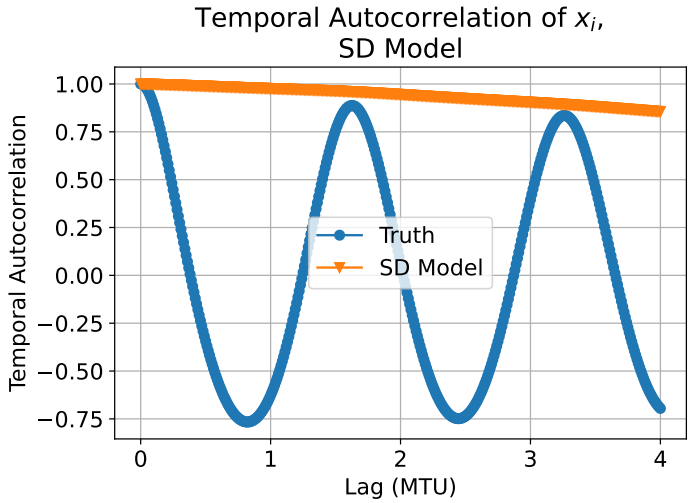

Figure E.5.4: Temporal autocorrelation function of x_i inferred from SD model, multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

F Single time delay-embedded model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$, and $F = 20$

F.1 Deterministic Model

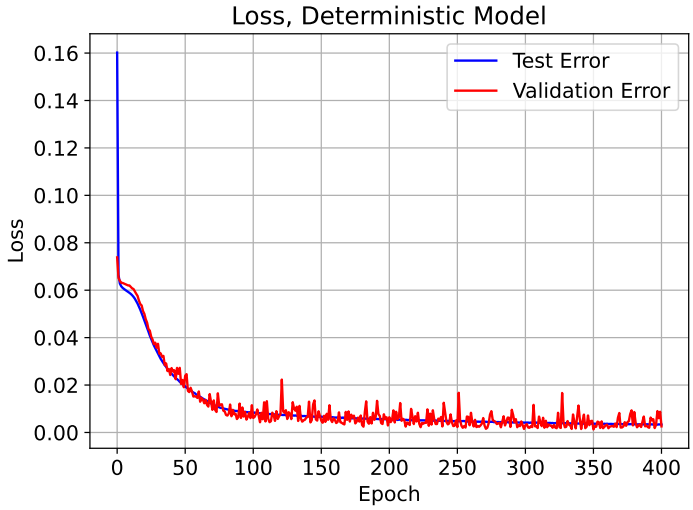

Figure F.1.1: Loss function for single time delay-embedded deterministic model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

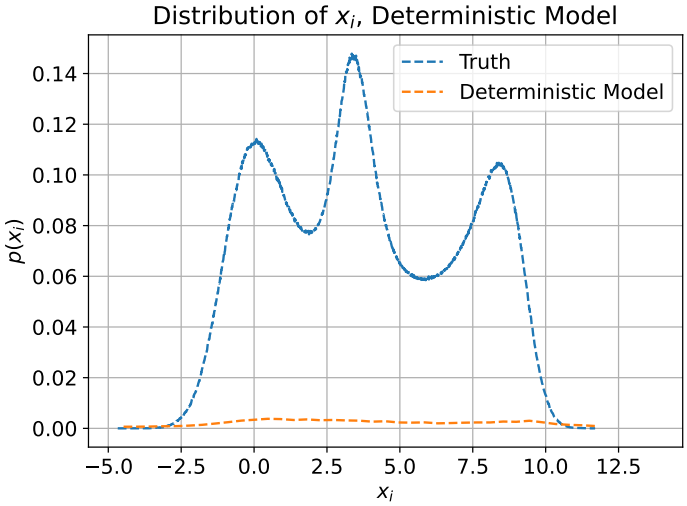

Figure F.1.2: Distribution function of x_i inferred from single time delay-embedded deterministic model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

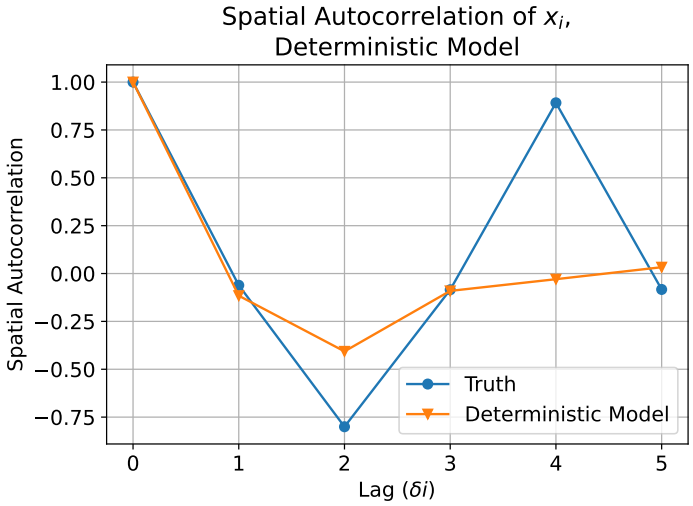

Figure F.1.3: Spatial autocorrelation function inferred from single time delay-embedded deterministic model for multiscale Lorenz 96 system with $h = 5$, $b = 10$ $c = 2$, $F = 20$.

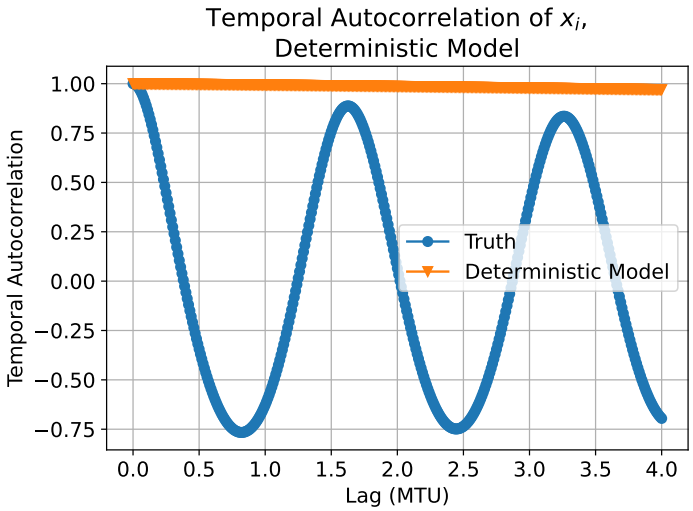

Figure F.1.4: Temporal autocorrelation function inferred from single time delay-embedded deterministic model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$, $F = 20$.

F.2 ML model

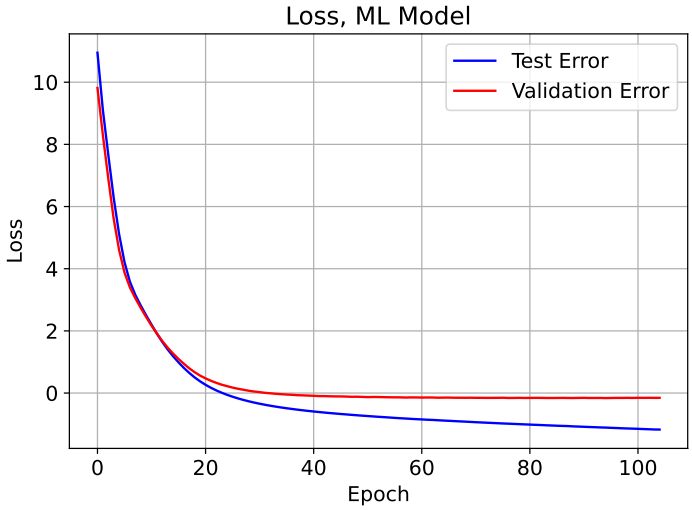

Figure F.2.1: Loss function for single time delay-embedded ML model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

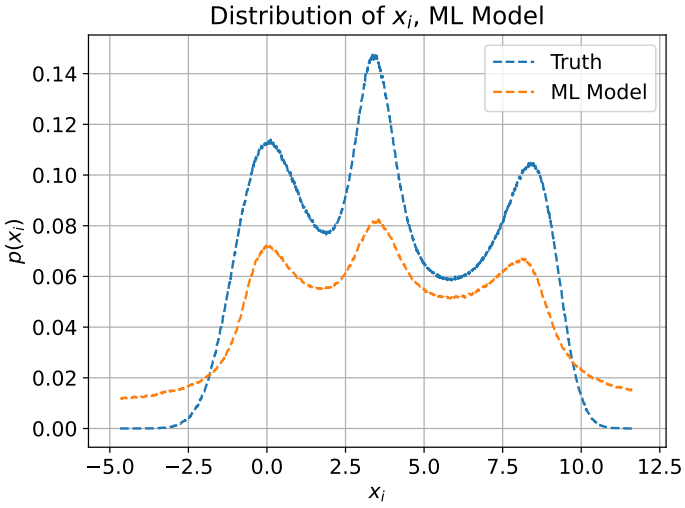

Figure F.2.2: Distribution function of x_i inferred from single time delay-embedded ML model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

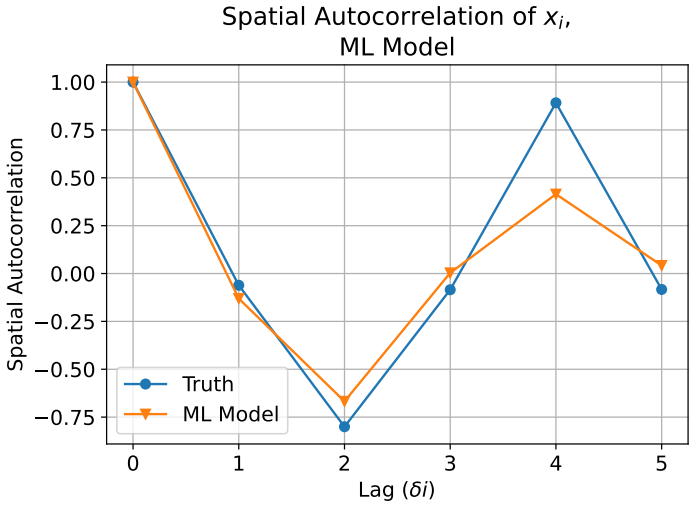

Figure F.2.3: Spatial autocorrelation function of x_i inferred from single time delay-embedded ML model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

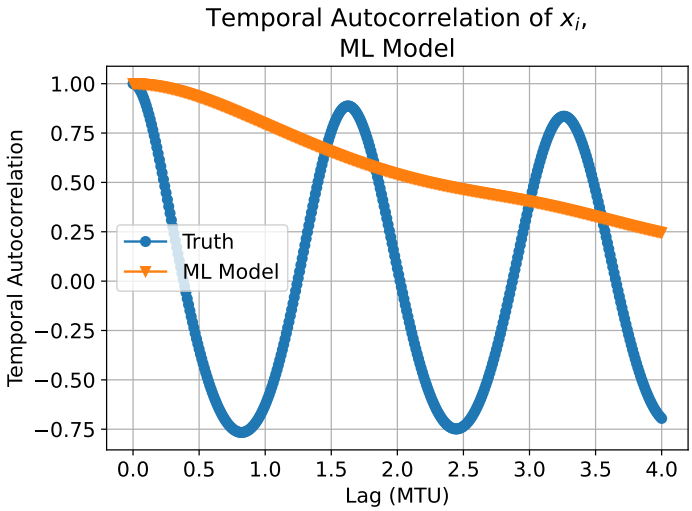

Figure F.2.4: Temporal autocorrelation function of x_i inferred from single time delay-embedded ML model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

F.3 MD model

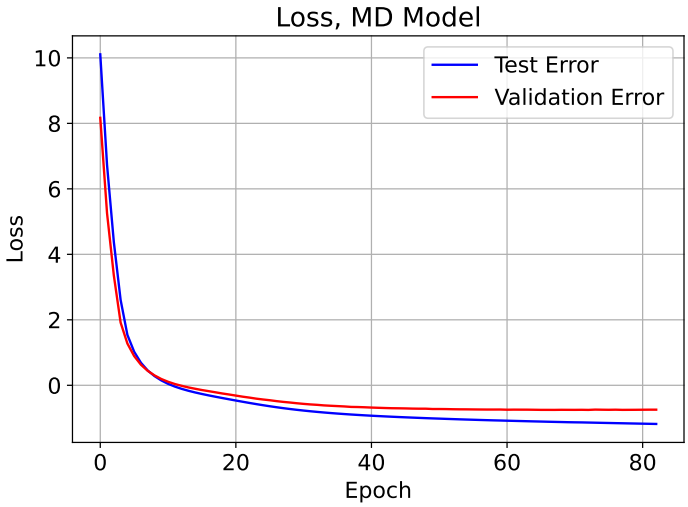

Figure F.3.1: Loss function for single time delay-embedded MD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

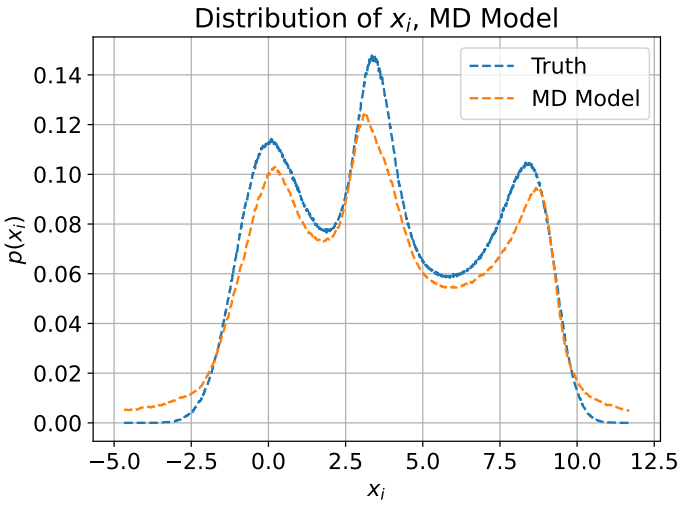

Figure F.3.2: Distribution function of x_i inferred from single time delay-embedded MD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

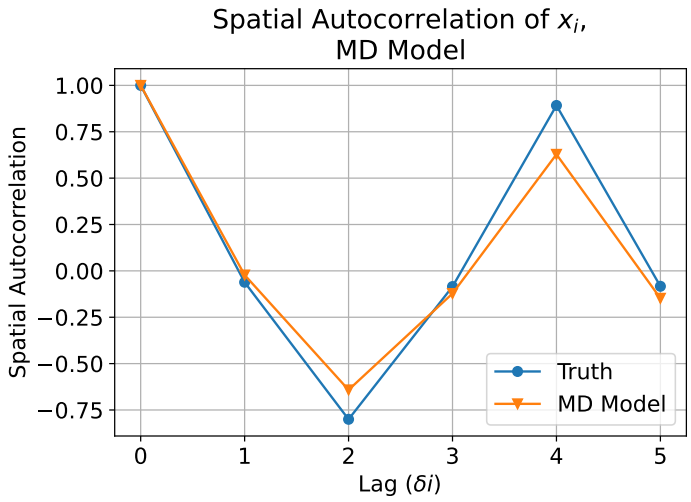

Figure F.3.3: Spatial autocorrelation function of x_i inferred from single time delay-embedded MD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

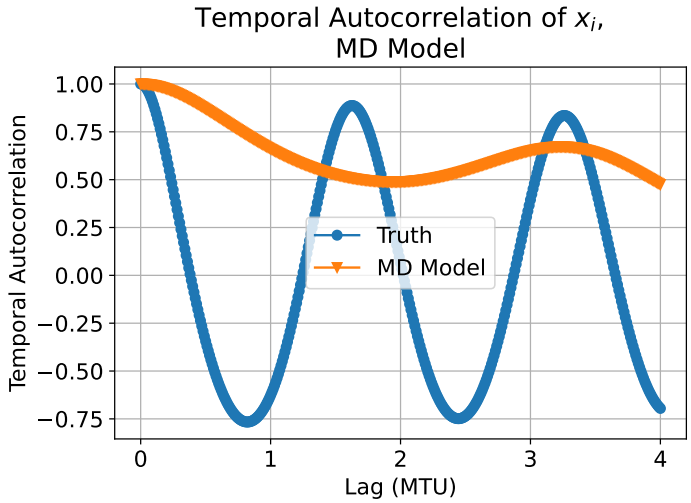

Figure F.3.4: Temporal autocorrelation function of x_i inferred from single time delay-embedded MD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

F.4 SL model

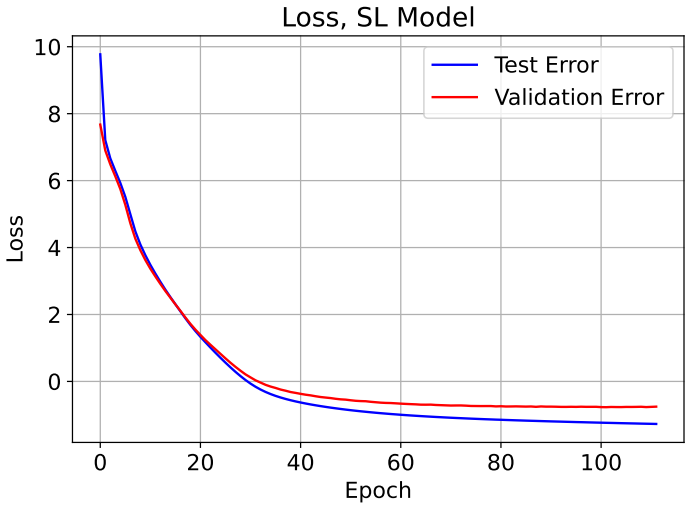

Figure F.4.1: Loss function for single time delay-embedded SL model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

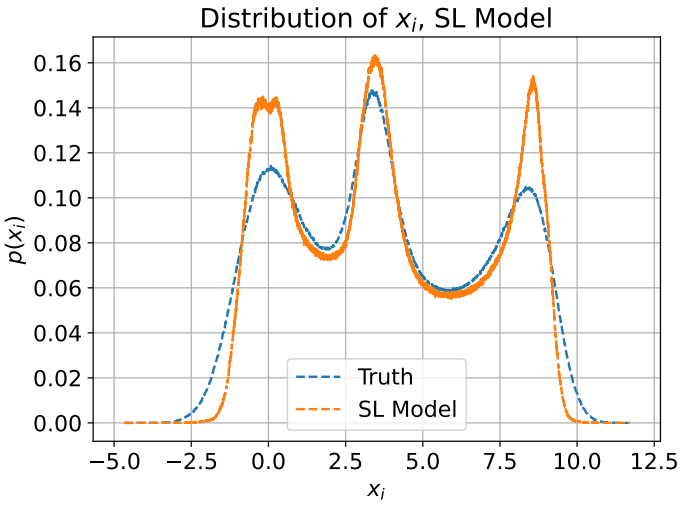

Figure F.4.2: Distribution function of x_i inferred from single time delay-embedded SL model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

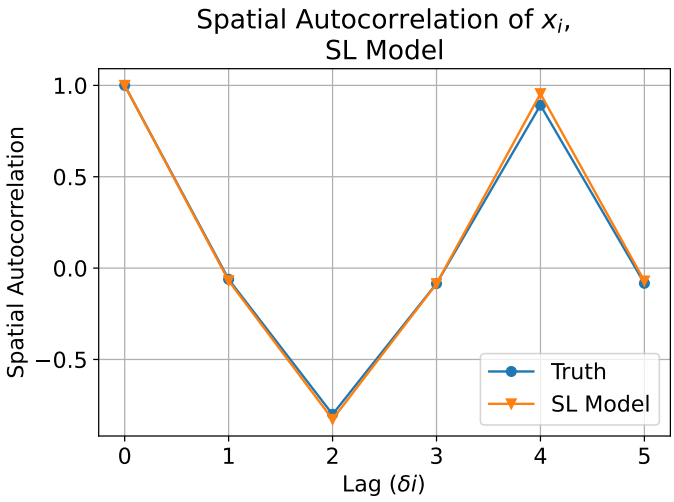

Figure F.4.3: Spatial autocorrelation function of x_i inferred from single time delay-embedded SL model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

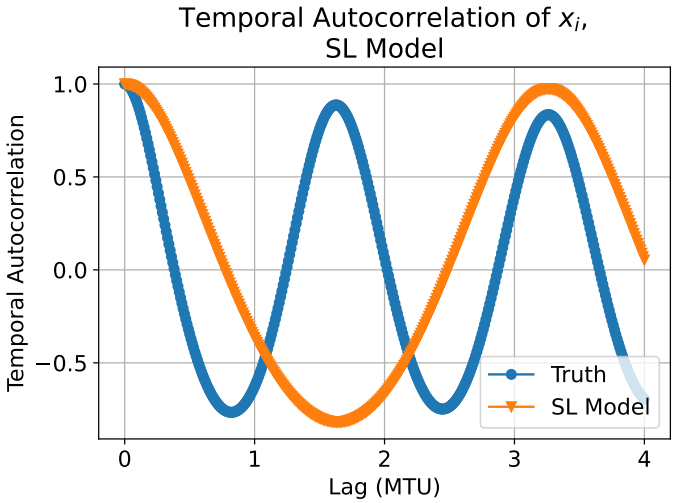

Figure F.4.4: Temporal autocorrelation function of x_i inferred from single time delay-embedded SL model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

F.5 SD model

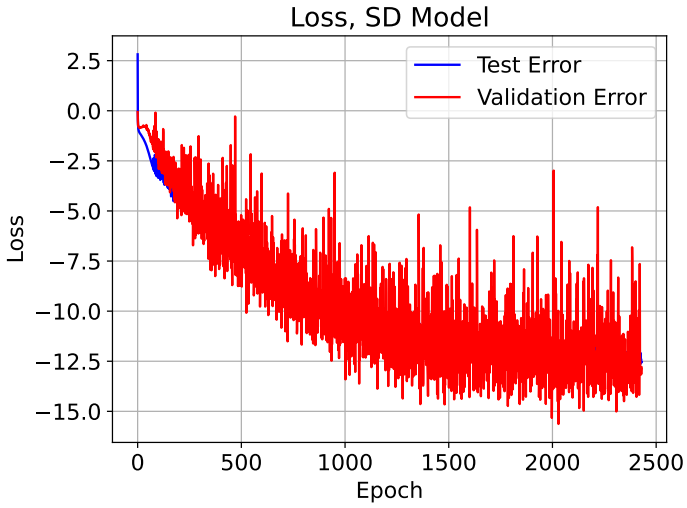

Figure F.5.1: Loss function for single time delay-embedded SD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

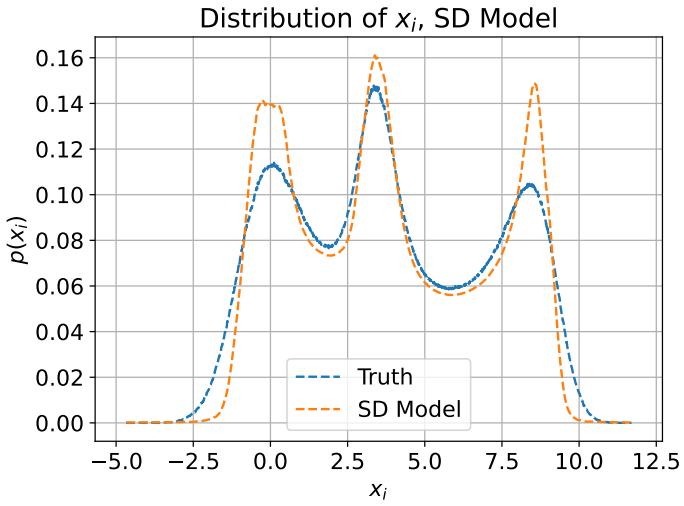

Figure F.5.2: Distribution function of x_i inferred from single time delay-embedded SD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

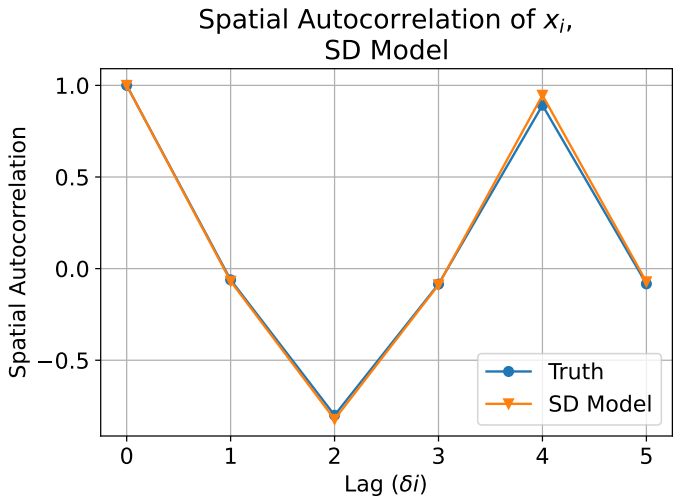

Figure F.5.3: Spatial autocorrelation function of x_i inferred from single time delay-embedded SD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.

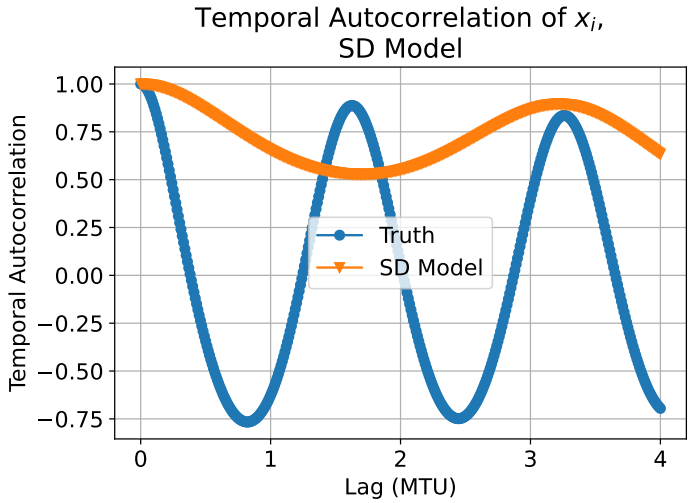

Figure F.5.4: Temporal autocorrelation function of x_i inferred from from single time delay-embedded SD model for multiscale Lorenz 96 system with $h = 5$, $b = 10$, $c = 2$ and $F = 20$.