

ABSTRACT

Title of dissertation: Gaussian Process Regression for Model Estimation

Degree Candidate: Balaji Vasan Srinivasan

Degree & Year: Master of Science, 2008

Thesis directed by: Professor Ramani Duraiswami
Department of Computer Science

State estimation techniques using Kalman filter and Particle filters are used in a number of applications like tracking, econometrics, weather data assimilation, etc. These techniques aim at estimating the state of the system using the system characteristics. System characteristics include the definition of system's dynamical model and the observation model. While the Kalman filter uses these models explicitly, particle filter based estimation techniques use these models as part of sampling and assigning weights to the particles. If the state transition and observation models are not available, an approximate model is used based on the knowledge of the system. However, if the system is a total black box, it is possible that the approximate models are not the correct representation of the system and hence will lead to poor estimation.

This thesis proposes a method to deal with such situations by estimating the models and the states simultaneously. The thesis concentrates on estimating the system's dynamical model and the states, given the observation model and the noisy observations. A Gaussian process regression based method is developed for estimating the model. The regression method is sped up from $O(N^2)$ to $O(N)$ using an data-dependent online approach for fast Gaussian summations. A relevance vector machine based data selection scheme is used to propagate the model over iterations. The proposed method is tested on a Local Ensemble Kalman Filter based estimation for the highly non-linear Lorenz-96 model.

Gaussian process regression for model estimation

by

Balaji Vasan Srinivasan

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2008

Advisory Committee:

Professor Ramani Duraiswami, Chair / Advisor
Professor Raghu Murtugudde
Professor Rama Chellappa

©Copyright by

Balaji Vasan Srinivasan

2008

DEDICATION

I dedicate this thesis to my parents who are supporting me in all my endeavors and are instrumental in what I am in this life of mine. I also place all my work as an offering to the Lotus feet of Bhagawan Sri Sathya Sai Baba.

ACKNOWLEDGEMENTS

I am grateful to those people who have made this research and thesis possible and those who have made my experience in graduate school one that I will always remember fondly.

First and foremost I'd like to thank my advisor, Professor Ramani Duraiswami for giving me an invaluable opportunity to work on challenging and extremely interesting projects. I should say I have been amazingly fortunate to have an advisor who gave me the freedom to explore on my own, and at the same time the guidance to recover when my steps faltered. His passion and unparalleled enthusiasm has inspired me to think better and have taught me different ways to look at a problem.

I also thank Professor Rama Chellappa and Professor Raghu Murtugudde for serving on my Masters thesis committee. The course on Statistical Pattern Recognition that I took under Professor Chellappa was the launchpad for my research and the knowledge I gained from the course has formed the basis for all my research. The discussions I have had with Professor Raghu Murtugudde, has been very insightful and has been helpful in understanding the problems better. Both of them have been inspirational figures for me in terms of excellence a person can

achieve in a field.

I don't have enough words to thank Dr. Ashwin Swaminathan and Kiran Kumar for all the fundaes and advices. They are two persons I have looked up to at times of distress. I would also like to thank my current roommates Shivsubramani, Karthick and Rangarajan for supporting me when I was working on my thesis. I would also like to thanks my friends Bargava, Ranjit, Sidharth, Karthik, Rajesh, Raghuraman and Suhasini for the bright moments that have made my graduate life memorable.

I am forever indebted to my parents, sister and brother-in-law because I am what I am due to their support, effort and sacrifices. They have been by my side at all situations giving me the much needed emotional support always.

Finally, I thank Almighty for giving me the physical and mental strength to work on this thesis.

TABLE OF CONTENTS

List of Tables	vii
List of Figures	viii
1 Introduction	1
1.1 Motivation	1
1.1.1 Approaches to Dual estimation	3
1.2 State Space Representation	5
1.3 State Estimation	6
1.3.1 Models f and g	7
1.4 Organization	8
1.5 Novel Contributions	9
2 Gaussian Processes in Regression	11
2.1 Introduction	11
2.1.1 Covariance function	14
2.2 Complexity Issues in GP	15
2.3 Fast Method Selection	15
3 Local Ensemble Kalman Filter	17
3.1 Kalman Filter	17
3.2 Local Ensemble Kalman filter (LEKF)	18
3.2.1 Lorenz-96 Model	20
3.3 Performance of LEKF	22
3.4 Estimation with an incorrect model	22
3.5 Lyapunov Exponents	22
4 Proposed Algorithm	29
4.1 Generic Framework	29
4.2 Model propagation	30
4.2.1 Informative Vector Machine	31
4.3 Complete Algorithm	32

5	Performance of the algorithm	34
5.1	IVM Data Size selection	34
5.2	State and Model Estimation	36
5.3	Performance over iterations	38
6	Conclusions and Discussions	45
6.1	Main contributions	45
6.2	Summary	46
6.3	Further Possible Work	47
6.3.1	Irregular spatial and temporal data sampling	47
6.3.2	IVM Complexity	48
6.3.3	“Non-parametric” model for the parameters	48
A	More about GPR	50
A.1	Posterior density	50
A.2	Maximum Likelihood	53
B	Various steps in LEKF	54
B.1	Localization	55
B.2	Transformation	56
C	Informative Vector Machine	58
C.1	Assumed Density Filtering (ADF)	58
C.1.1	Active Set selection	59
C.2	Data Point Selection	59
	Bibliography	61

LIST OF TABLES

5.1	Mean relative error over 10s of iterations	37
6.1	Estimation errors using different models	46
6.2	Usage for the different methods	47

LIST OF FIGURES

1.1	State Space Formulation	6
3.1	Chaotic nature of the Lorenz-96 model is shown. Small changes in initial conditions lead to large variation in predicted states	20
3.2	State locations on a latitude circle in Lorenz model	21
3.3	Estimated states at 40 locations of the Lorenz model with the knowledge of the Lorenz model at 5th and 10th time instants	23
3.4	Estimated states at 40 locations of the Lorenz model with the knowledge of the Lorenz model at 20th and 40th time instants	24
3.5	Estimated states at 40 locations of the Lorenz model estimated with a wrong linear model as system model at 5th and 10th time instants	25
3.6	Estimated states at 40 locations of the Lorenz model estimated with a wrong linear model as system model at 20th and 40th time instants	26
3.7	RMSE between true state and state estimated using a Wrong model - illustrates the divergence of the estimation technique	28
5.1	Relative Error vs Size of active set in IVM; The active set is selected based on the entropy measure in Equation 4.3 and this data is used for model prediction in each step of model estimation	35
5.2	The CPU time taken for data selection using IVM for different size of the active set	36
5.3	State tracking with the proposed method: True and estimated at 2 (out of 40) locations of the Lorenz model	39
5.4	State tracking with the proposed method: True and estimated at 2 (out of 40) locations of the Lorenz model	40
5.5	Estimated states at 40 locations of the Lorenz model estimated with the proposed algorithm at 5th and 10th time instants of estimation	42
5.6	Estimated states at 40 locations of the Lorenz model estimated with the proposed algorithm at 20th and 30th time instants of estimation	43
5.7	Estimated states at 40 locations of the Lorenz model estimated with the proposed algorithm at 40th and 50th time instants of estimation	44
B.1	States at grid locations in 2-Dimensions	55

Chapter 1

Introduction

1.1 Motivation

A system with latent states observed through noisy observations have been studied for several decades. State estimation techniques like Kalman filter and particle filters are used to estimate the latent states using the noisy observations. Kalman filters and particle filters have several state tracking applications like visual tracking [1], acoustic tracking [3], impedance tracking [44], deformation tracking [36] and trajectory tracking [52]. However, both particle filter and Kalman filter are limited by the requirement that the system model needs to be specified *a priori*.

In order to solve state estimation for imprecisely known systems, simultaneous state and model estimation have been studied using different approaches. Kalman filter [17] was developed in the early 1960s for solving state estimation [15] prob-

lems. Ever since then, the possibility of using the Kalman filter for simultaneously estimating the states and model parameters have been discussed. The earliest work [39] on dual estimation was to estimate the parameters of a linear regression model along with the states of the system, and the regression model parameters were used for subsequent predictions. The Kalman filter used in this scenario was the linear Kalman filter. However because of the limitations of the linear Kalman filter, simultaneous model-state estimation were modified using the *Extended Kalman Filter* for non-linear problems [4, 6, 8, 32]. It was observed that the simultaneous estimations with Extended Kalman Filter did not have good convergence properties for all scenarios. [25, 26] discuss about this convergence issue and provide conditions in which good convergence will be obtained in Extended Kalman Filter.

With the improvement in modern learning methods and development of Kalman filter for more complex scenarios, the dual estimation problems were solved using newer approaches in the late 1990s. With the emergence of Unscented Kalman Filter [16], the ability of Kalman filter to solve highly non-linear systems for state estimation has increased. Furthermore, with new learning methods, the non-linear system learning have also become more and more sophisticated. [46] uses the Unscented Kalman filter with an Artificial Neural Network to model a non-linear system. [9] learned a non-linear dynamical system using Artificial Network in an Expectation Maximization framework.

These simultaneous estimation methods have had several application, like for example, [18] applied the ideas to Image Restoration where an Extended Kalman filter is used to estimate the blurring matrix along with the states in 2D images. With a dual-estimation approach, although the overall complexity of the system increases, it makes state estimation applicable to imprecisely specified systems. [12, 43, 51] apply a simultaneous model estimation approach to visual tracking and servoing. The dual estimation ideas have also been extended to acoustic tracking [7], hydrological modeling [29] and biochemical tracking [41]. The different approaches for dual estimation is discussed in the next sub-section.

1.1.1 Approaches to Dual estimation

Based on the formulation of the simultaneous estimation problem and the solution, the simultaneous estimation methods can be classified into three types:

Dual Estimation: [45, 48] used an artificial neural network in conjunction with an Extended Kalman filter to estimate the model and states. Here the parameters of the neural model were estimated using an additional Kalman filter. Because these approaches use two Kalman filters, one for model parameter estimation and another for state estimation, they are referred as Dual KF methods. [46, 47] indicate the shortcomings of the usual extended Kalman filter and propose an unscented transformation based Kalman filter and use it for dual estimation in a similar framework. [30] extended the Dual

KF idea to hydrological models using an Ensemble Kalman Filter.

Joint Estimation: Instead of using two Kalman filters to estimate the model parameters and states respectively, an alternate approach would be to replace the two Kalman filters in the dual Kalman with a single Kalman filter ([5] have some earliest work) that estimates jointly the states and the model parameters (the state vector now comprises both the states and the model parameters) in the same step. A major shortcoming in the dual KF and the joint KF approaches is that it is not easy to solve for highly complex system. When the complexity and hence the size of the parameters of the model increase, it becomes increasingly infeasible to define the dynamics for the parameters to be utilized in the Kalman filter framework.

Expectation Maximization based approach: Expectation Maximization (EM) based dual estimation methods are very robust and can be used to estimate complex models. [9] propose an EM-based dual estimation method, where an Artificial Neural network is trained at each time instant to update a model in the M -step of the EM algorithm. Unlike the dual Kalman filter, this method does not use a Kalman filter to estimate the parameters of the neural network, it instead tries to find a maximum-likelihood estimation of the parameters of the model. Of the previously proposed methods for dual estimation, this appears to be the best performing.

Although neural networks are very powerful, one problem with them is that they are a parametric formulation, which might be a limiting factor in modeling highly complex systems. Motivated by this problem, a non-parametric Gaussian process regression based model estimation is proposed in this thesis in an Expectation-Maximization framework. The advantage of Gaussian process regression is that it can be used to model highly non-linear complex systems without assuming any parametric form. A detailed explanation of Gaussian Process Regression is given in Chapter 2. The proposed method is tested with the Lorenz-96 data model [27] in a Local Ensemble Kalman Filter [13, 33].

In order to establish the various notations that will be used in this thesis, the state space representation of a system and state estimation using standard filtering are discussed in subsequent sections of this chapter.

1.2 State Space Representation

State space representation [15] is used to perform inference and learning in dynamical systems. In the standard model, the hidden state $x(t)$ evolves in time according to a dynamical model driven by past states with additive noise,

$$x_{t+1} = f(x_t) + \delta, \tag{1.1}$$

while it is observed via the “outputs” $y(t)$, related to the state $x(t)$ as

$$y_t = g(x_t) + \epsilon. \tag{1.2}$$

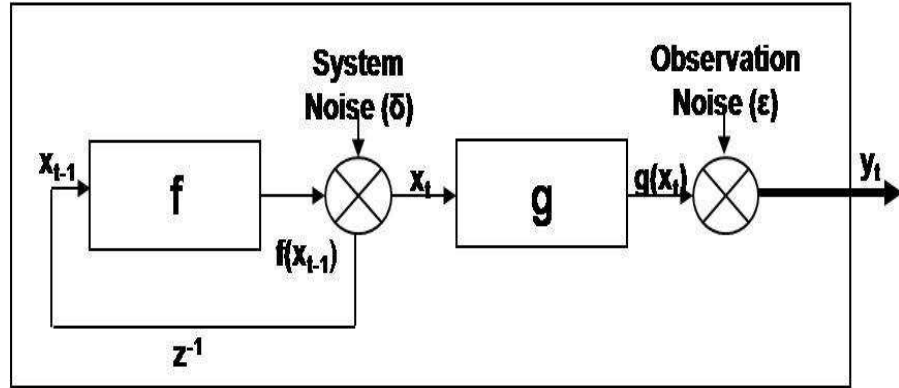


Figure 1.1: State Space Formulation

Here, “ f ” is the dynamical model for the state transition and can be linear or non-linear; δ describes the system noise; “ g ” is called the observation model, and may also be linear or non-linear; and ϵ is the observation noise. The “filtering” problem for dynamical systems is to solve for the states x_t , given system dynamics f , observation model g and noisy observations y_t . The subscripts here denote the time instants under consideration. Such a system is shown in Figure 1.1.

1.3 State Estimation

For a system as defined in the previous section, state estimation problem is to solve for the states x_t , given system dynamics f , observation model g and noisy observations y_t . There are different ways of state estimation depending upon the

nature of f and g and a particular choice of estimation technique depends on the size and nature of the problem. Kalman filter and particle filter are popular state estimation techniques used in various applications. In a Kalman filter, the posterior density of the state at every step is assumed to be Gaussian and is parameterized by its mean and covariance. This requires the system and observation noise to be Gaussian as well ($\delta \sim N(0, Q)$, $\epsilon \sim N(0, R)$). Particle filters do not need the assumption of Gaussianity. They are Monte Carlo methods where estimation is started with a set of particles, each assigned a weight. The weights of each particle and the particles themselves are propagated over time to perform state estimation. Each of these methods have their own advantages and disadvantages.

The absence of Gaussianity assumption makes the particle filter very generic and usable in multi-modal applications like in visual tracking [2, 14]. However, one drawback of particle filter is that the number of particles and hence the computational complexity of the filter increases with dimensions of the problem. Kalman filter and its variants have the same complexity with increased dimensions of the observation and state space. This makes it more suited for high-dimensional problems like weather data assimilation [13, 33].

1.3.1 Models f and g

The main assumption in both these methods is that the state transition model f and the observation model g are available. The Kalman filter uses these models

explicitly to reduce the cost between the estimated state and the state that would have resulted in the observation. However in the particle filter, the transition model f is a part of the *importance density function* (used to sample the particles) and the observation model g is used for assigning weights to each particle.

The state transition dynamics (f) and observation model (g) can be defined in different ways. It may be specified as a differential equation (e.g., [33]) or as an approximate model that defines the dynamics (e.g., [11]). If the state-transition model is inexact, the resulting state estimation problem is solved poorly. Also in some cases, the system under consideration will be a complete black box with little knowledge about its model. In these cases, it would be necessary to estimate the model and state simultaneously. This is called dual estimation. In this thesis, one such approach to simultaneous model and state estimation is proposed and discussed.

1.4 Organization

The thesis is organized as follows. Chapter 2 introduces Gaussian process regression in a Bayesian framework. It also shows the computations complexity in Gaussian process regression and a fast method for solving it. Further the idea is extended with an online data-dependent method selection, which selects the fastest method (from a group of methods) for the data. Chapter 3 introduces the Local Ensemble Kalman filter (LEKF) along with the Lorenz-96 data model. It

also shows the performance of the LEKF with and without the knowledge of the model. In Chapter 4, the proposed algorithm is introduced with a model propagation strategy. Finally the results are presented and discussed in Chapter 5 before providing concluding discussions in Chapter 6

1.5 Novel Contributions

The main contributions in this thesis are as follows:

1. A non-parametric approach is proposed to estimate the model on-the-fly in the state estimation problem in an Expectation Maximization framework. Existing methods use a parametric formulation which can be limiting in complex scenarios.
2. Fast summation, iterative methods and a Relevant Vector Machine ([42]) like compression are used to propagate the model using a concise and relatively efficient data representation.
3. The proposed method to the Lorenz-96 data model to test its performance with a highly non-linear system

Publication

The work presented in this thesis has resulted in two Conference publication (one accepted and one submitted).

- Vlad I. Morariu, Balaji Vasan Srinivasan, Vikas C. Raykar, Ramani Duraiswami, and Larry S. Davis. *Automatic online tuning for fast Gaussian summation*, Advances in Neural Information Processing Systems (NIPS), 2008. [accepted]
- Balaji Vasan Srinivasan and Ramani Duraiswami, *Non-parametric dual estimation of dynamical systems using Gaussian Process Regression*, Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS), 2009 [submitted]

Chapter 2

Gaussian Processes in Regression

2.1 Introduction

Traditional regression uses a given set of data to fit an assumed function. $y = f(x) + \epsilon, \epsilon \sim N(0, \sigma^2)$ Looking at this from a Bayesian perspective [50], suppose there are various possible models $\{f_1, f_2, f_3, \dots\}$ that can be possibly the desired solution, a prior probability can be placed on this and let that be called $P(f)$. Now, when the data is available, the posterior probability of each model can be modified based on the observation. Suppose we have data $D : \{x_i, y_i\}_{i=1}^N$. Now the likelihood of target corresponding to each of the data points can be evaluated as $P(t_i|x_i, f)$ assuming that the data points are independent of each other. Combining the prior and likelihood, the posterior can be obtained as

$$P(f|D) \propto P(f) \times \prod_{i=1}^N P(t_i|x_i, f) \quad (2.1)$$

The prior placed here is on the model. Different prior leads to different model realization. The problem in this framework is that the prior model should be relevant to the model under consideration, if not can lead to poor representation of the underlying model. It is not practically possible to choose a model framework for each problem. So it would be better to have a non-parametric framework for regression problems. Gaussian process regression [35] aims at providing the solution in a *non-linear non-parametric* framework by placing the prior on function values than the model.

Gaussian Process regression defines the distribution over functions and inference takes place directly in the function-space. It can be seen as a collection of random variables which are jointly Gaussian. A Gaussian process is completely specified by its mean function $m(x)$ and covariance function $k(x, x')$ given by

$$m(x) = E[f(x)] \tag{2.2}$$

$$k(x, x') = E[(f(x) - m(x))(f(x') - m(x')))] \tag{2.3}$$

This is generally written as $f(x) \sim GP(m(x), k(x, x'))$. Given the data D, the joint distribution of the function outputs "f" is assumed to be a Gaussian ($f \sim N(0, K)$). Once this is assumed, if a new data x_* arrives, the corresponding output f_* is jointly Gaussian with f (by the Gaussian process assumption), and is given by,

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim N \left(0, \begin{bmatrix} K(X, X) & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix} \right) \tag{2.4}$$

If there are n training points and n_* testing points, $K(X, X_*)$ denotes the $(n \times n_*)$ matrix of covariances evaluated at all pairs of training and testing points. To get the posterior distribution over the functions, this joint distribution must be restricted to those functions which agree with the observations. In other words, we have to remove the functions which are inconsistent with the observations. In probabilistic terms, we have to evaluate $P(f_*|X_*, X, f)$, which because of the Gaussianities turns out to be another Gaussian [35] given by,

$$P(f_*|X_*, X, f) \sim N \left(K(X_*, X)K(X, X)^{-1}f, K(X_*, X_*) - K(X_*, X)K(X, X)^{-1}K(X, X_*) \right) \quad (2.5)$$

Details of this posterior's Gaussianity is given in Appendix A. However, the f 's here are all noiseless observations which are rare in practical scenarios. So extending the joint distribution for a noisy observation with noise variance $\sigma^2 I$,

$$\begin{bmatrix} y \\ f_* \end{bmatrix} \sim N \left(0, \begin{bmatrix} K(X, X) + \sigma^2 I & K(X, X_*) \\ K(X_*, X) & K(X_*, X_*) \end{bmatrix} \right) \quad (2.6)$$

and the posterior modifies as

$$P(f_*|X_*, X, f) \sim N \left(K(X_*, X)\tilde{K}(X, X)^{-1}f, K(X_*, X_*) - K(X_*, X)\tilde{K}(X, X)^{-1}K(X, X_*) \right) \quad (2.7)$$

where $\tilde{K} = (K + \sigma^2 I)^{-1}$. Equation 2.7 is the posterior in the Gaussian Process Regression. To summarize, given data $D : \{x_i, y_i\}_{i=1}^N$, and a new test point x_* , the posterior of the function output, f_* is $p(f_*|x_*, D) \sim N(m, V)$ where, $m = k(x_*)^T(K + \sigma^2 I)^{-1}y$, and $V = K(x_*, x_*) - k(x_*)^T(K + \sigma^2 I)^{-1}k(x_*)$. "m" gives the predicted value of f_* at x_* and V is the variance of the prediction.

2.1.1 Covariance function

The covariance function K can be chosen to reflect the prior information about the problem. For large scale problems, in the absence of the prior knowledge, the negative squared exponential (Gaussian) is the most widely used covariance function. This function is the one that we use in this paper.

$$K(x, x') = \sigma_f^2 \exp\left(-\sum_{k=1}^d \frac{(x_k - x'_k)^2}{h_k^2}\right)$$

The $d + 2$ parameters ($[h_1, h_2, \dots, h_d, \sigma_f, \sigma]$) are the hyper-parameters of the Gaussian process. Intuitively, the hyperparameters ‘ h ’ of the Covariance function can be thought of as a weighting factor for each of the input dimension. A very high h along one dimension would result in the corresponding exponential term zeroing out along that dimension. In other words the radius of influence (for the covariance function) is lesser along that dimension. For a 1-dimensional case, very high h would result in a covariance function almost equal to identity, and a small h would result in a dense Covariance function.

The gaussian process can model highly non-linear problems and the non-parametric nature of the formulation makes it even more flexible and viable option. Because of its non-parametric nature and versatile capabilities, it has been used in several applications [19, 21, 49]. However, one disadvantage is that the evaluation of the mean and variance involves the inversion of a N-by-N matrix and a matrix-vector multiplication making it $O(N^3)$. In the next section, we discuss this in detail and provide a solution in literature for speeding up Gaussian processes.

2.2 Complexity Issues in GP

The main contributor to the complexity is the calculation of mean m and variance V . Consider the evaluation of the mean; denoting $(K + \sigma^2 I)^{-1}y$ as η , the mean can be written as $m = k(x^*)^T \eta$. η involves the inversion of the covariance function and hence has $O(N^3)$ complexity. Once η is available, m can be evaluated in $O(N)$. [37] provides an $O(N)$ algorithm for solving for m . [37] proposes the use of conjugate gradient method to solve the linear system $(K + \sigma^2 I) \times \eta = y$ to find η . This reduces the complexity to $O(kN^2)$ where k is a constant. Also, it proposes the use of an Improved Fast Gauss Transform (IFGT) [38] to improve this to $O(N)$. It is also shown in [37] that using an inexact krylov subspace [40] the conjugate gradient iteration can be performed in an increasingly inexact manner as the iteration proceeds. Thus by using a combination of Conjugate Gradient and Improved Fast Gauss Transform, [37] achieves $O(N)$ complexity for the evaluation of m . On a similar note, the variance is shown to be evaluated in $O(N^2)$.

2.3 Fast Method Selection

With the use of Conjugate Gradient based method to solve the linear system, the core problem is reduced to a summation of gaussian kernel and there are also other methods (other than IFGT) available in literature to speed-up such a summation. Dual-tree methods [10, 24] approach the problem by building trees for the data

points. It is also possible to use the IFGT approach with the dual trees to achieve speed up. Also, for some data, the scaling in these fast methods are so high that a direct method is better than these approaches. To deal with this problem of plenty, we have proposed an automatic method selection strategy [31] for choosing the fastest method between direct summation, dual trees, IFGT and IFGT + trees. This method performs as well as the fastest approach and in some cases switches dynamically between methods to outperform all the methods.

This method¹. was used for the Gaussian process regression in this thesis. To evaluate the $d + 2$ hyper-parameters, the maximum likelihood method (explained in Appendix A) presented in [35] was used ².

¹From <http://www.umiacs.umd.edu/~morariu/figtree/>

²From <http://www.gaussianprocess.org/gpml/code>

Chapter 3

Local Ensemble Kalman Filter

3.1 Kalman Filter

For a system with Gaussian noise given by,

$$x_{t+1} = f(x_t) + \delta, \delta \sim N(0, Q)$$

$$y_t = g(x_t) + \epsilon, \epsilon \sim N(0, R)$$

Kalman filter aims to estimate the state x_t by minimizing the cost function J ,

$$J = \sum_{j=1}^n [y_t - g(x_t^-)]^T R_t^{-1} [y_t - g(x_t^-)] \quad (3.1)$$

where, $x_t^- = f(x_{t-1})$. Assuming that the posterior density is Gaussian, the Kalman filter based state estimation defines a factor K called “Kalman Gain” using the observation model g , observation noise variance R and the covariance of the predicted

states x_t^- . The final state estimate is then defined by,

$$x_t = x_t^- + K(y_t - g(x_t^-)) \quad (3.2)$$

Such a formulation divides Kalman filter into two steps;

1. Prediction step: Predict x_t^- using x_{t-1} : $x_t^- = f(x_{t-1})$
2. Update step: Alter x_t using y_t ; $x_t = x_t^- + K(y_t - g(x_t^-))$

The classical Kalman filter was developed for linear models (observation and system dynamics) only. However, it was extended for non-linear cases by linearizing the non-linear models at the points of interest and using the Jacobian based evaluations. The Kalman filter for non-linear models is called ‘‘Extended Kalman Filter’’.

Extended Kalman filter is not usable in large problems because of the cost of evaluation of covariance of the states at each instant. So instead of evolving a single state over time, an ensemble of states are evolved over time and this ensemble carries information about the covariance. This is the ‘‘Ensemble Kalman Filter’’.

3.2 Local Ensemble Kalman filter (LEKF)

The ensemble Kalman filter is not efficient for large scale problems like in weather data assimilation. In order to modify the ensemble Kalman filter for such large problems, [13, 33] introduced a spatial localization strategy coupled with a PCA-

based subspace analysis for the ensemble Kalman filter. This is the Local Ensemble Kalman Filter. The basic steps in Local Ensemble Kalman Filter are,

Step 1: Initialize an ensemble of states for the initial time

Step 2: Predict the states to the next time instant using the state transition model

Step 3: Associate a local region spatially to each state and consider only the state vectors in this region

Step 4: Use a PCA transformation to move the states from the current space to a low dimensional subspace

Step 5: Perform the Kalman filter based update to the predicted states (in the low dimensional subspace)

Step 6: Move back to the local space and use the locally calculated vectors as the actual state of the system

Step 7: Go to step 2

A detailed explanation of the localization and the transformation in LEKF is given in Appendix B

3.2.1 Lorenz-96 Model

Ott et. al who propose LEKF [33] used a Lorenz-96 [27,28] 40-variable toy-model to test their performance. We also use the same model with LEKF. According to this model, the states in a spatially distributed system was defined by the differential equation,

$$\dot{x}(j, t) = [x(j + 1, t) - x(j - 2, t)]x(j - 1, t) - x(j, t) + F \quad (3.3)$$

Here, $j=1,2,\dots,J$ are the spatial locations at which the states are measured. The F here is called "forcings" and together with J determines the chaotic nature of the system.

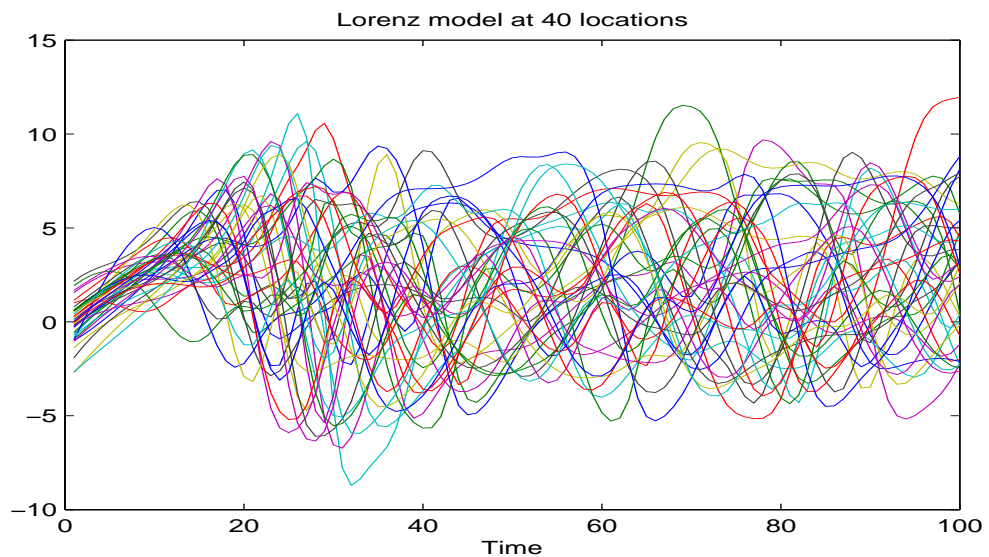


Figure 3.1: Chaotic nature of the Lorenz-96 model is shown. Small changes in initial conditions lead to large variation in predicted states

In a weather assimilation scenario, these spatial locations can be thought of as

points along a latitude circle at which the states need to be evaluated (or at which the observations are available). Also, $x(-1) = x(J - 1)$, $J = 40$ and $F = 8$ like in [33]. This can be thought to something like in Figure 3.2.



Figure 3.2: State locations on a latitude circle in Lorenz model

This model is widely used in many data assimilation literature, because of its chaotic and non-linear nature. It has also been shown in [27] that the evolution of states in this model are similar to the evolution of a meteorological quantity. Figure 3.1 shows how the states evolve from close initial state, this also reveals the non-linear and chaotic nature of the data. For all the analysis in this thesis, it was assumed that there are no observations missing spatially and the data are sampled at regular intervals without any missing observations at any instant.

3.3 Performance of LEKF

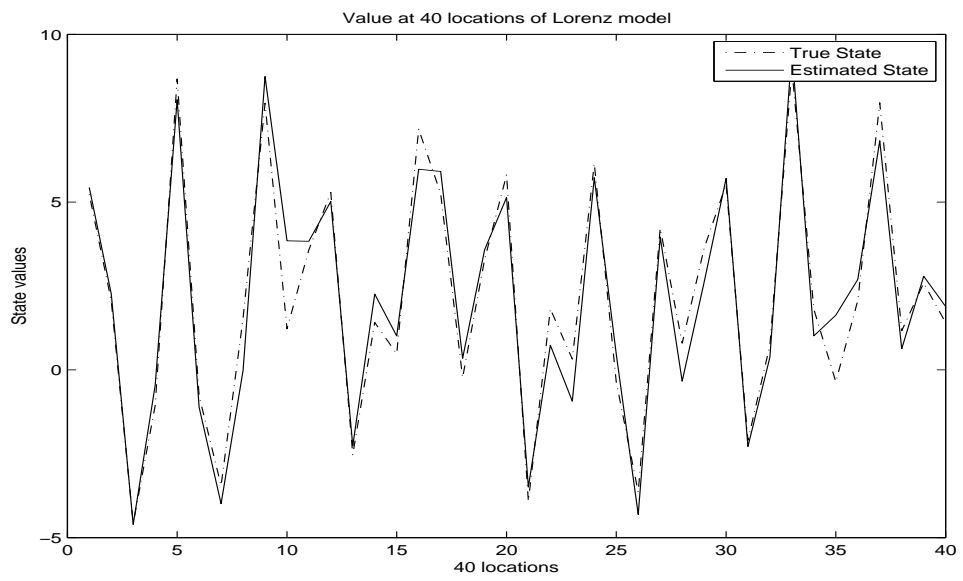
The LEKF was used with the Lorenz-96 data model and the estimation was performed with the knowledge of the data model (equation 3.3). The results of the estimation at the 40 sites of the Lorenz model is shown in figures 3.3 and 3.4. It can be seen that the iteration converges very fast with the knowledge of the system dynamical model.

3.4 Estimation with an incorrect model

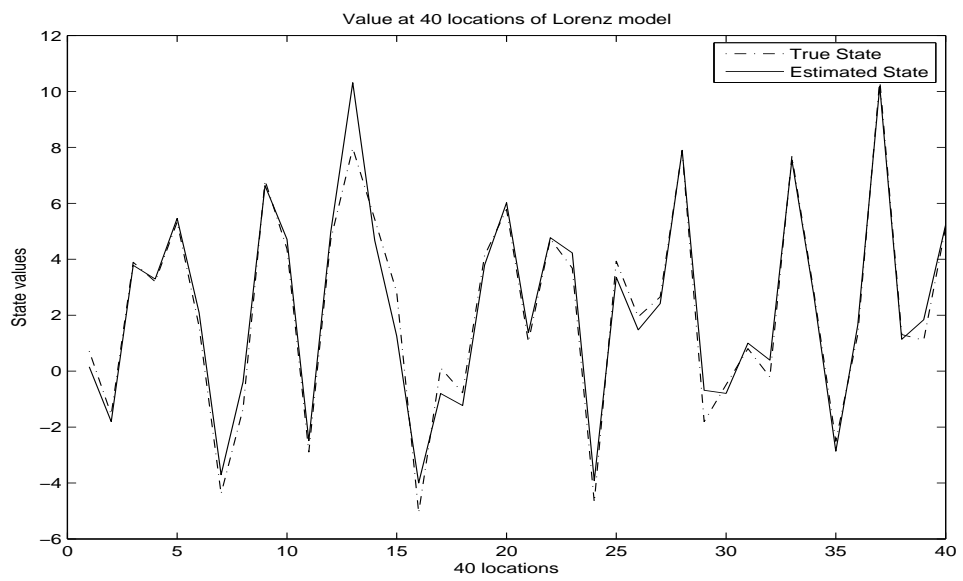
In order to motivate the problem of model estimation, the LEKF was used with the Lorenz-data model, however an incorrect model was used in place of the actual model (a linear model was used in the prediction step instead of solving equation 3.3). This is to motivate the scenario where the state estimation needs to be performed without the knowledge of the system dynamical model. As expected, such a model diverges from the true state value and this is shown in figures 3.5 and 3.6.

3.5 Lyapunov Exponents

The F in the equation 3.3 is called “forcing” and together with J , it determines the Lyapunov exponent of the system. Lyapunov exponent denotes the long-term average growth rate of a very small error. In other words, it denotes the rate at

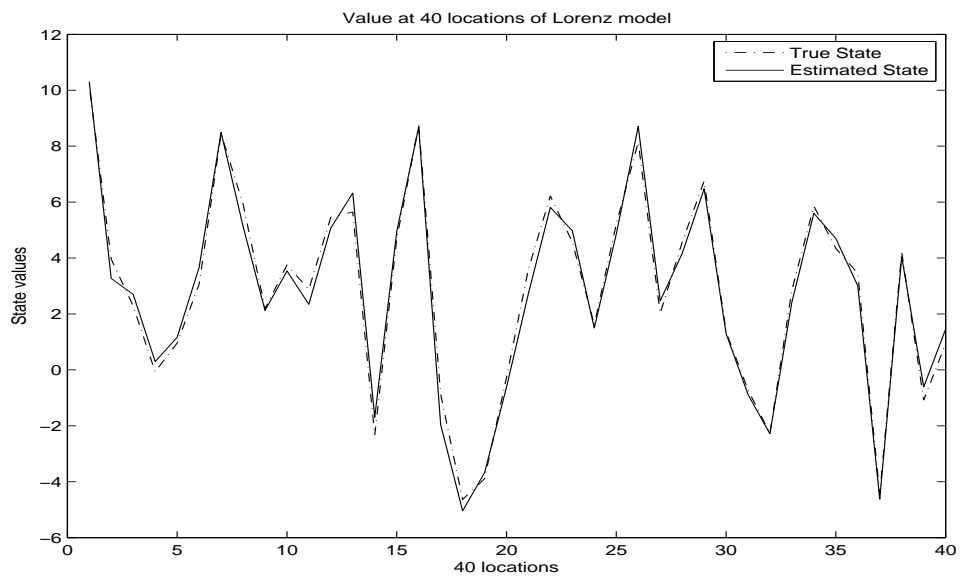


(a) States at 40 locations @ t=5

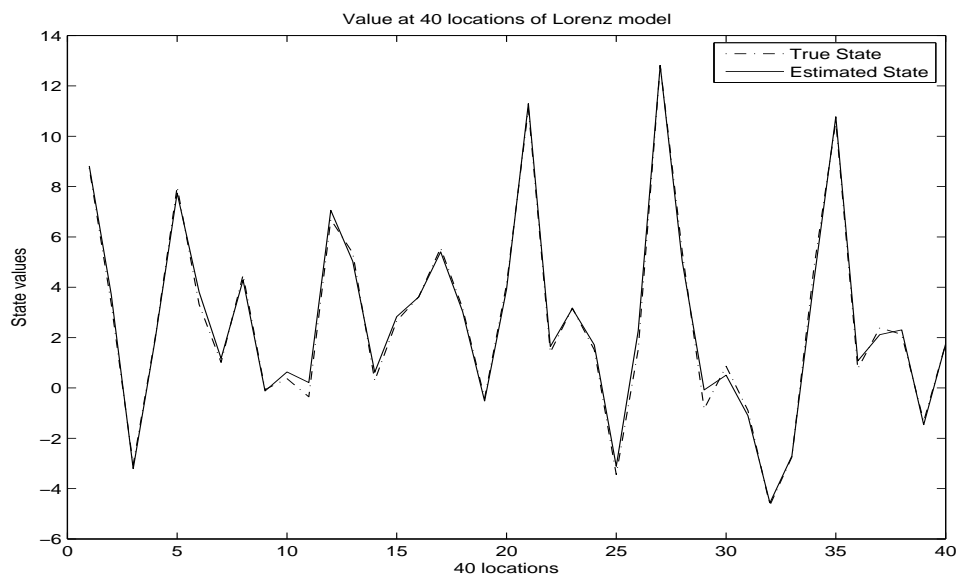


(b) States at 40 locations @ t=10

Figure 3.3: Estimated states at 40 locations of the Lorenz model with the knowledge of the Lorenz model at 5th and 10th time instants

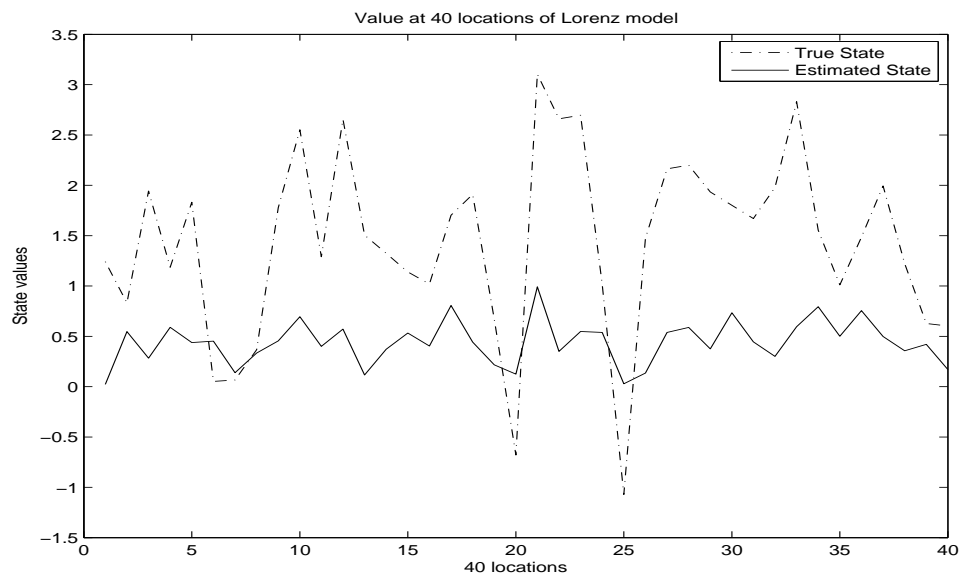


(a) States at 40 locations @ t=20

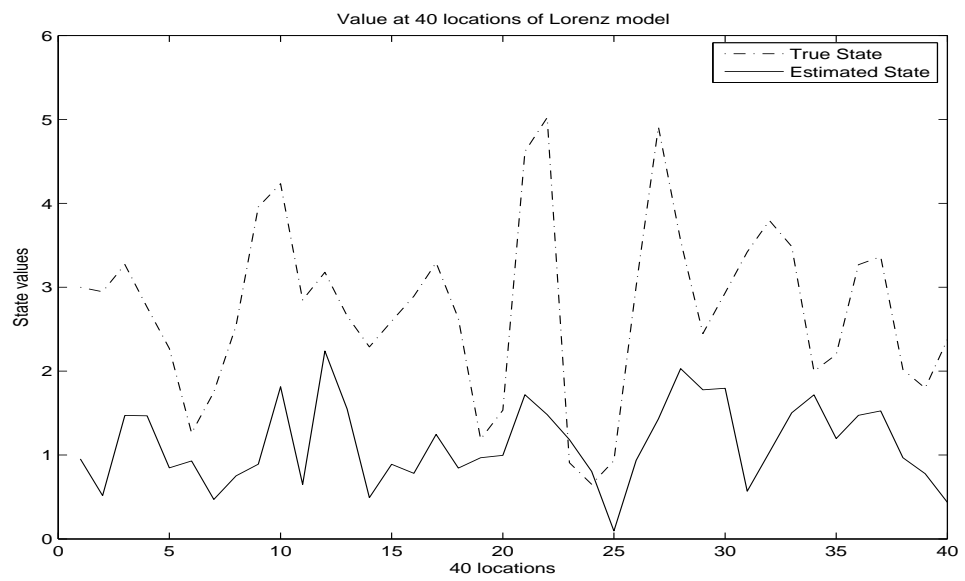


(b) States at 40 locations @ t=40

Figure 3.4: Estimated states at 40 locations of the Lorenz model with the knowledge of the Lorenz model at 20th and 40th time instants

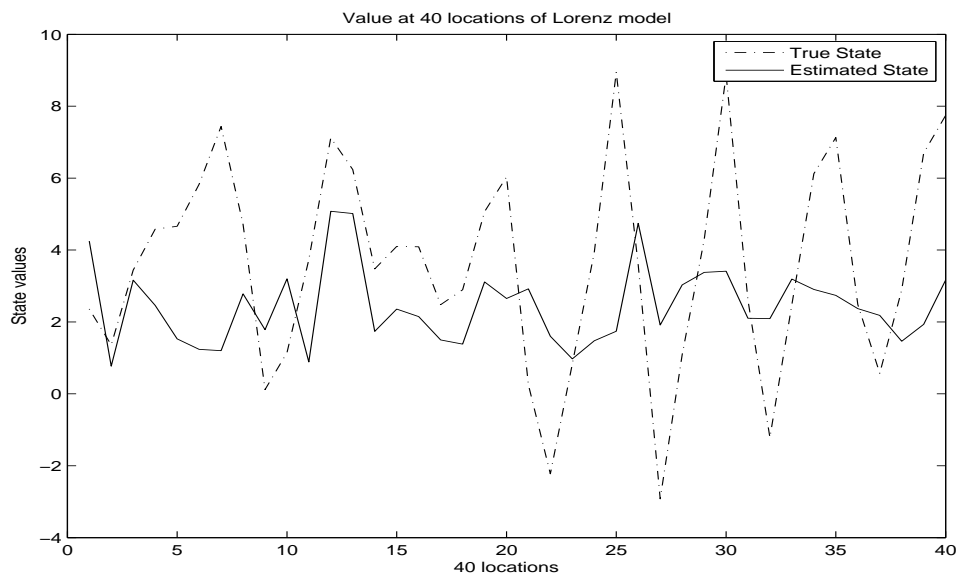


(a) States at 40 locations @ t=5

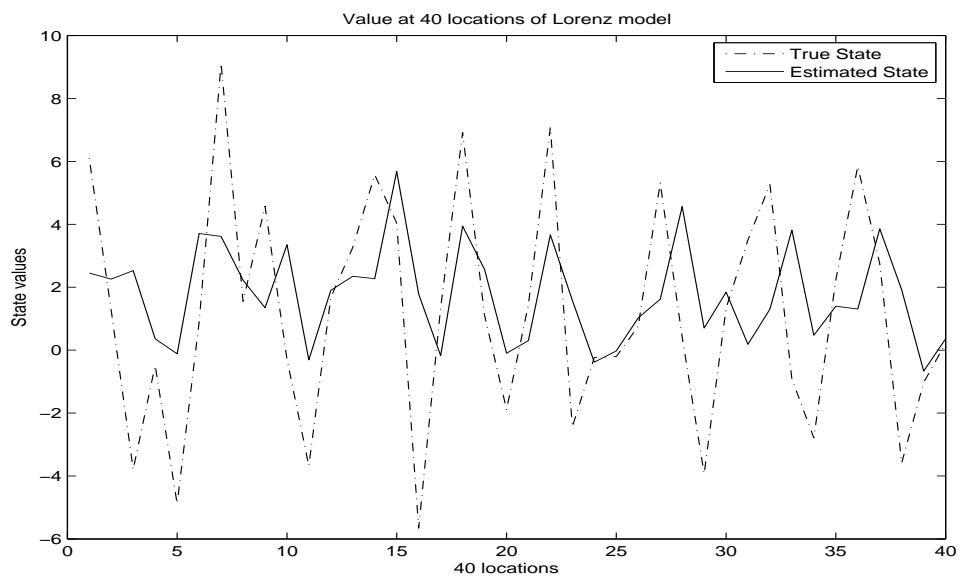


(b) States at 40 locations @ t=10

Figure 3.5: Estimated states at 40 locations of the Lorenz model estimated with a wrong linear model as system model at 5th and 10th time instants



(a) States at 40 locations @ t=20



(b) States at 40 locations @ t=40

Figure 3.6: Estimated states at 40 locations of the Lorenz model estimated with a wrong linear model as system model at 20th and 40th time instants

which small errors will amplify. A low Lyapunov exponent means that the system can be predicted even if the error is allowed to propagate over some iterations. However, for a moderate or high Lyapunov exponent, error propagation would mean that the error in the system is to grow at a faster rate. For a forcing $F = 8$ and $J = 40$, the system above has a high Lyapunov exponent.

When performing state estimation for a system with high Lyapunov exponents, it is necessary that the state estimation does not propagate the error, or the error reduce as iterations proceed. In a wrong linear model, since the error in the initial iterations is not corrected and the same faulty model is used for further estimation, the error diverges to a large extent as the iterations proceed. This can be seen from 3.7.

The proposed approach (which will be introduced in the next chapter) aids in such a situation where the model is not only unknown, but also requires more complex modeling technique.

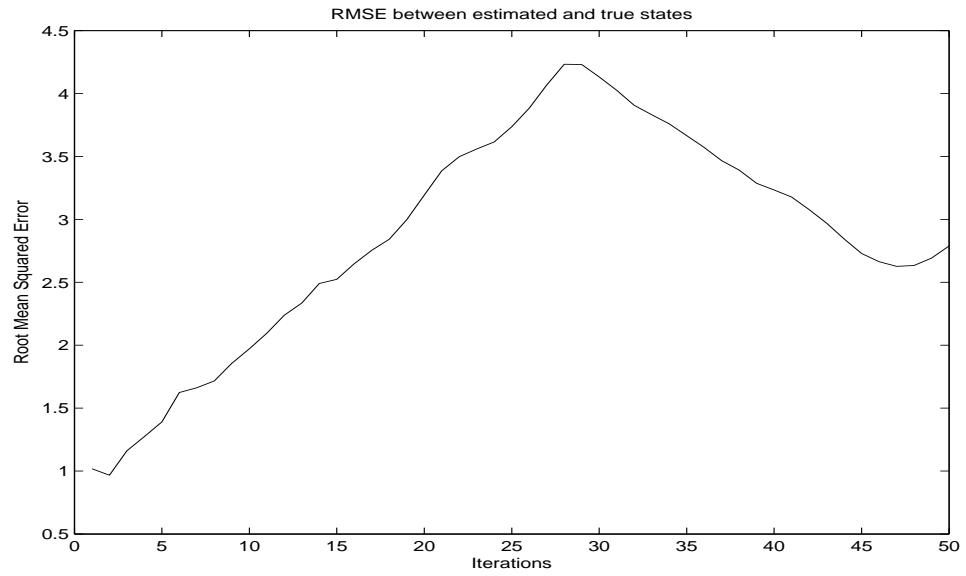


Figure 3.7: RMSE between true state and state estimated using a Wrong model - illustrates the divergence of the estimation technique

Chapter 4

Proposed Algorithm

4.1 Generic Framework

Step 1 Start with an approximate model, f_0 and initial state x_0

Step 2 At time t , using model f_{t-1} , propagate [prediction step] state $x_{t-1|t-1}$ to $x_{t|t-1}$

Step 3 Using the observation y_t , update the state to $x_{t|t}$ [update step]

Step 4 Using states $x_{t-1|t-1}$ and $x_{t|t}$, update/evaluate model f_t using Gaussian Process Regression.

Step 5 Update $t = t + 1$, Go to step 2

In this generic framework, the initial model f_0 can be an approximation of the

actual model (if available), or if x_{-1} and x_0 are available, the model f_0 can be obtained by using step 4. A good initial guess of states would help in a faster convergence. The prediction and update steps (step 2 and step 3) are similar to those in a Kalman filter or a particle filter. The choice of a particular method depends on the nature of the problem and the algorithm assumes that the state estimation has a good performance for the given scenario. The step 4 of model estimation, uses the Gaussian process regression specified in section 2 and 3 to model f_t trained with inputs as x_{t-1} and output as x_t .

4.2 Model propagation

Gaussian Process Regression is a non-parametric *kernel* regression technique, and like all kernel methods, the training data is required for predictions as well. In the current problem, the training data are the estimated states and the prediction is done for future states. As the model propagates, it is necessary to have all the training data accumulated over past iterations for future predictions. This might result in two problems. First, the initial iterations can be noisy, and retaining this data in latter iterations can cause poor predictions. Second, the data size might increase exponentially as the iterations proceed. A solution to both problems is to retain some relevant data and throw other data after each iteration. One way to do this would be to use just data from $t - 1$ and t to predict the value at $t + 1$. Alternatively, a moving window can be used to throw the oldest data as new data

are accumulated. These methods, however, may result in any previously learnt characteristic of the system being lost.

The problem of selecting the most relevant data from a large set of data has been discussed previously in the literature and a number of sparse machine learning methods proposed. The relevant vector machine (RVM; [42]) deals with selecting the most relevant from a given set of training data using an incremental additions from the training set to the relevant set (also called active set). A similar approach in the Gaussian Process framework is available as the Informative Vector Machine (IVM) [22,23] where at each iteration the most informative (relevant) of the current data is retained based on an entropy criteria.

4.2.1 Informative Vector Machine

IVM is used in sparse Gaussian process methods [20] to get a sparse representation of data to speed up Gaussian process regression. The core idea here is to add data points one by one, but determining the point that best approximates the posterior distribution. Consider the posterior distribution $P(f|D)$, and taking its log, we obtain

$$P(f|D) \propto P(f) \times \prod_{i=1}^N P(t_i|x_i, f), \quad (4.1)$$

$$\log P(f|D) \propto \log P(f) + \sum_{i=1}^N \log P(t_i|x_i, f). \quad (4.2)$$

IVM utilizes this structure and adds the data points from the training set to a reduced active set using an entropy based measure. Suppose Σ_{i-1} is the posterior covariance with $i - 1$ datapoints. The entropy change when a point n is added as the i^{th} inclusion is given by [23],

$$\Delta H = -\frac{1}{2} \log |\Sigma_{i,n}| + \frac{1}{2} \log |\Sigma_{i-1}| \quad (4.3)$$

A rigorous analysis of the entropy and data selection can be seen in [22, 23]. Some important relations and derivations are shown in Appendix C. The data thus selected is then used for future predictions.

4.3 Complete Algorithm

We use IVM to provide a concise representation of the data from past iteration and current iterations. Suppose the IVM algorithm selects an active set of size ‘ M ’ from data of size ‘ N ’, it takes $O(N)$ time to select each of the M points [22]. So it has an overall complexity of $O(MN)$. Thus, the overall model estimation algorithm involves using IVM to select an active set of data, and use this data to train a Gaussian process model and use this model to predict the state at the next iteration. The overall complexity of the $O(MN)$ for IVM, $O(M)$ for training the Gaussian process and $O(N)$ for prediction. The complete algorithm is:

Step 1 Start with an approximate model M_0 and initial state x_0

Step 2 At time t , using model f_{t-1} , propagate [prediction step] state $x_{t-1|t-1}$ to $x_{t|t-1}$

Step 3 Using the observation y_t , update the state to $x_{t|t}$ [update step]

Step 4 Append states $x_{t-1|t-1}$ and $x_{t|t}$ to the data from past iterations and use IVM to extract the active set, I . Using the states in the active set I , update/evaluate model f_t using Gaussian Process Regression

Step 5 Update $t = t + 1$, Go to step 2

Chapter 5

Performance of the algorithm

In this chapter, the performance of the proposed Gaussian Process Regression based model estimation will be discussed.

5.1 IVM Data Size selection

In order to select the size of the active set in the Informative Vector Machine, the dual estimation was performed for different sizes of the active set. The relative error between the true and the estimated state after 50 iterations is shown in Figure 5.1. It can be seen that as we increase the size of the active set, the relative error drops. The reason for the increasingly better performance with the increase in the size of the active set could be because if the size of the active set is small IVM might result in the selection of the noisy data because of the poor representation of the posterior. However, it should also be noted that increasing the size of the

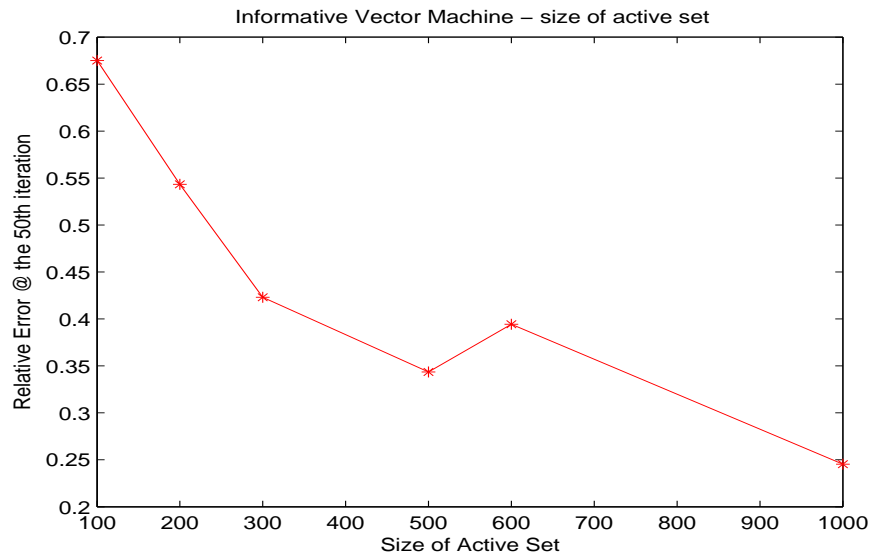


Figure 5.1: Relative Error vs Size of active set in IVM; The active set is selected based on the entropy measure in Equation 4.3 and this data is used for model prediction in each step of model estimation

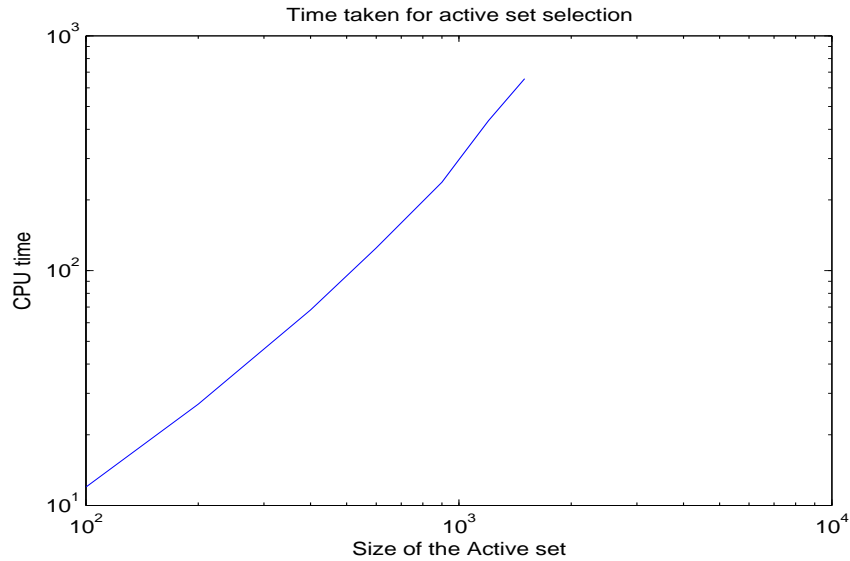


Figure 5.2: The CPU time taken for data selection using IVM for different size of the active set

active set also increases the complexity of the active set selection step. This is shown in figure 5.2 where the CPU time taken for data selection using Informative Vector Machine is shown for different active set size. In all our experiments, an active set of size 1000 was used.

5.2 State and Model Estimation

In order to illustrate the performance of the proposed method, the method was tested on the LEKF with data generated from the Lorenz model. Figures 5.3 and 5.4 show the state tracking at 4 (out of 40) locations of the Lorenz model. It can

Table 5.1: Mean relative error over 10s of iterations

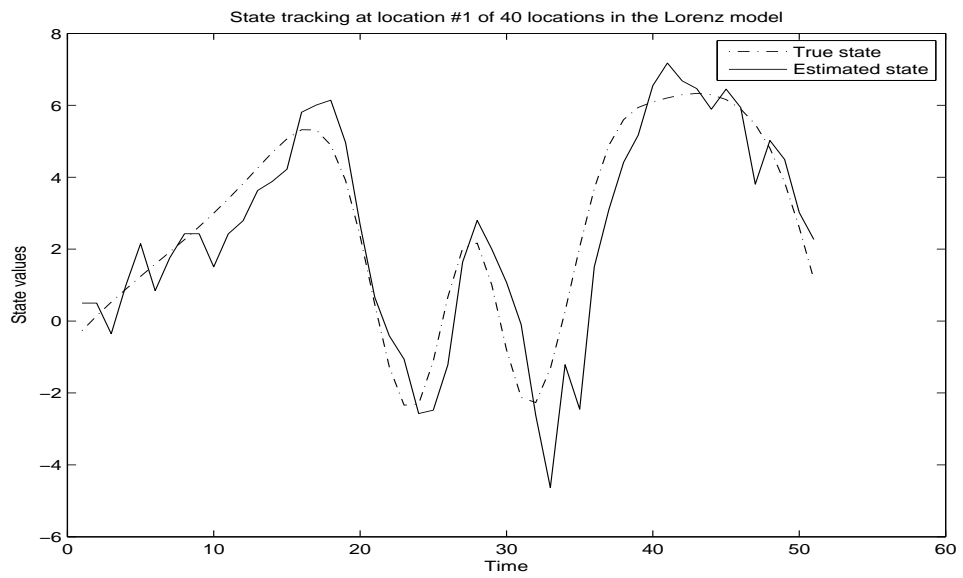
Models	1-10	11-20	21-30	31-40	41-50
Exact model	0.2213	0.1084	0.0851	0.0674	0.0681
Incorrect model	0.7773	0.6282	0.8020	0.9164	0.7974
GP based model	0.6456	0.4205	0.4545	0.4159	0.3111

be seen that with the proposed algorithm, although, the model is noisy, the states are tracked correctly.

The relative errors over iteration for the estimation with the correct model, with the wrong model and with the proposed Gaussian Process based model are shown in Figure 5.2. It can be seen that the error decreases rapidly for the LEKF with the known model. The proposed algorithm, although it has a high error initially, reduces gradually. With a wrong model, the error diverges. The mean relative error over successive ten iterations is shown in Table 5.1. It can be seen that although the wrong model and the Gaussian Process (GP) based model have high errors initially, the wrong model diverges and the proposed model reduces in error.

5.3 Performance over iterations

In order to illustrate how the model is initially error prone and how it learns as the iteration proceeds, estimated and actual states as the iterations proceed is shown for all 40 locations of the Lorenz model at various iterations in figures 5.5, 5.6 and 5.7. Each figure shows the true states at that time instant and the estimated states at that time instant.

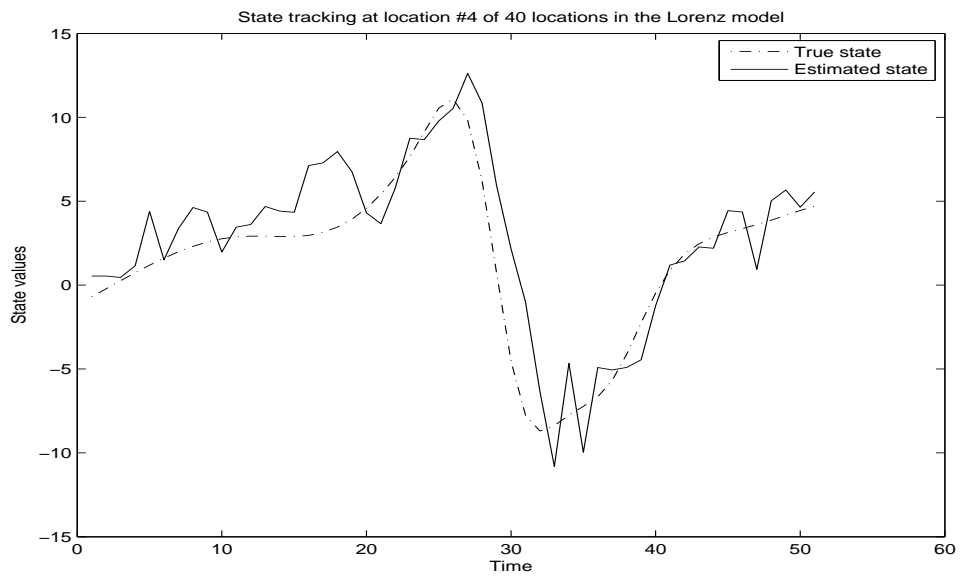


(a) States at location 1

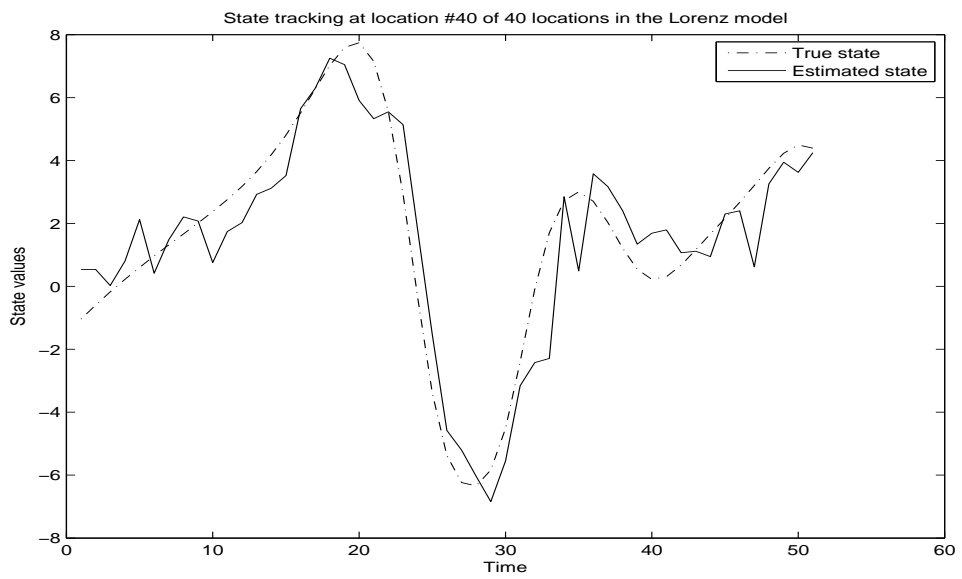


(b) States at location 13

Figure 5.3: State tracking with the proposed method: True and estimated at 2 (out of 40) locations of the Lorenz model



(a) States at location 4

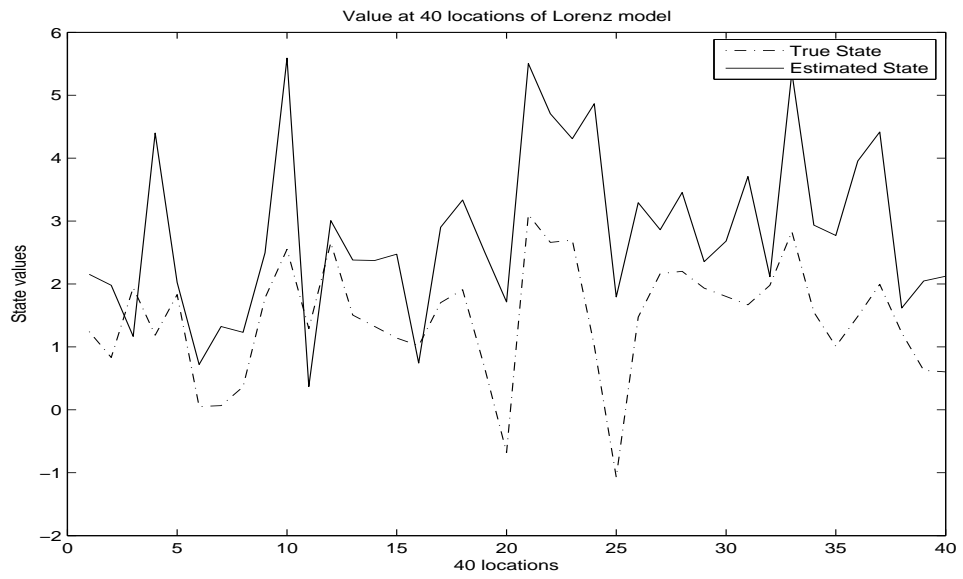


(b) States at location 40

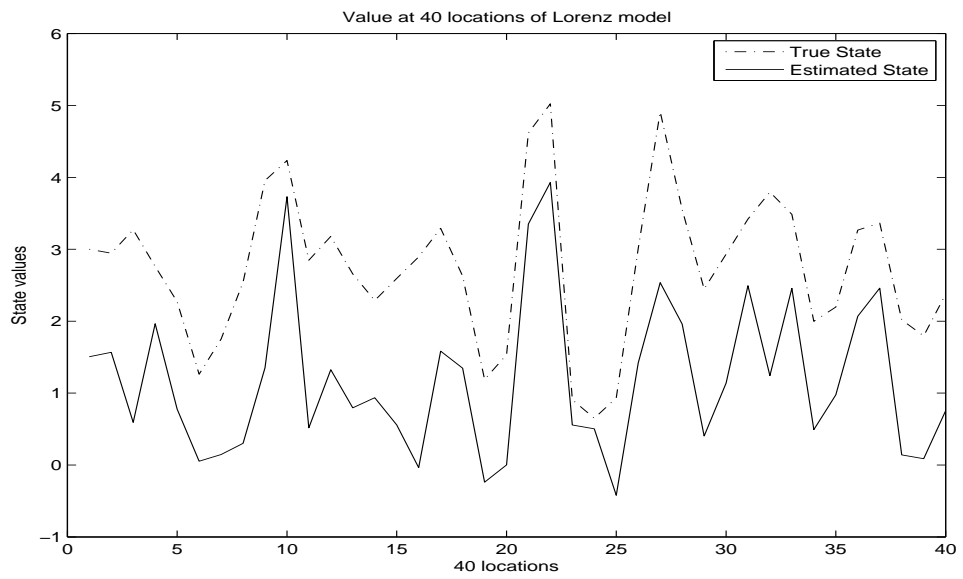
Figure 5.4: State tracking with the proposed method: True and estimated at 2 (out of 40) locations of the Lorenz model



Relative error for the three different methods - estimation with known model, with a wrong model, with the proposed algorithm

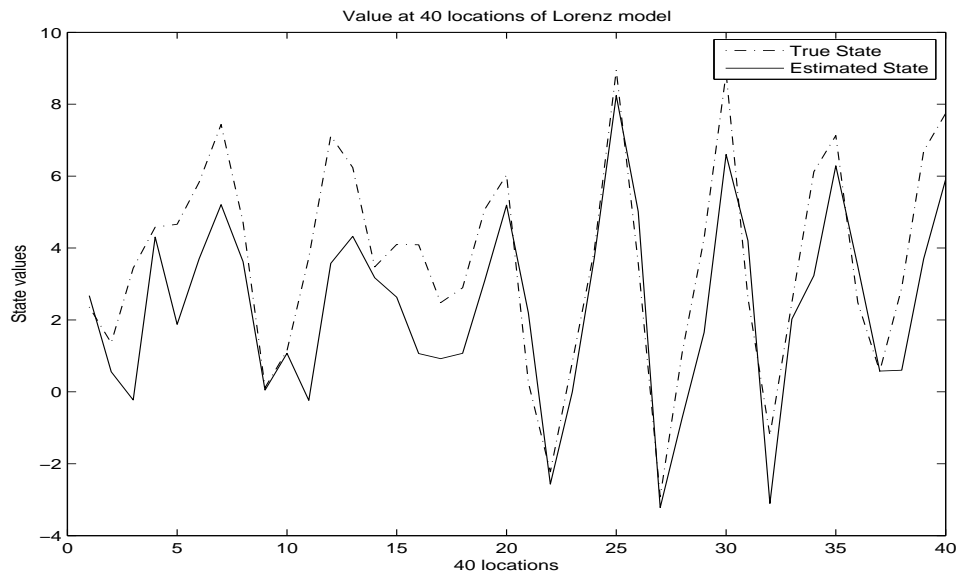


(a) States at 40 locations @ t=5

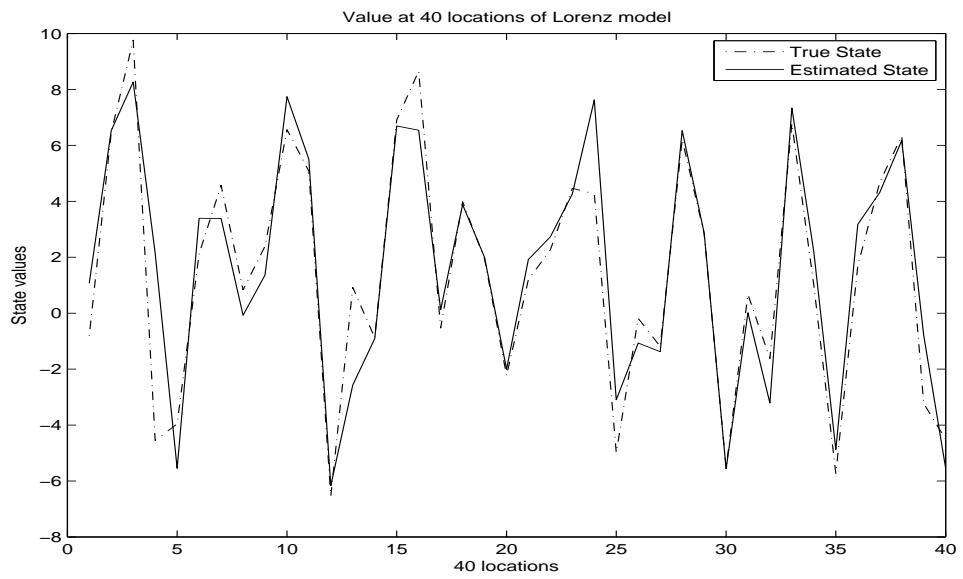


(b) States at 40 locations @ t=10

Figure 5.5: Estimated states at 40 locations of the Lorenz model estimated with the proposed algorithm at 5th and 10th time instants of estimation

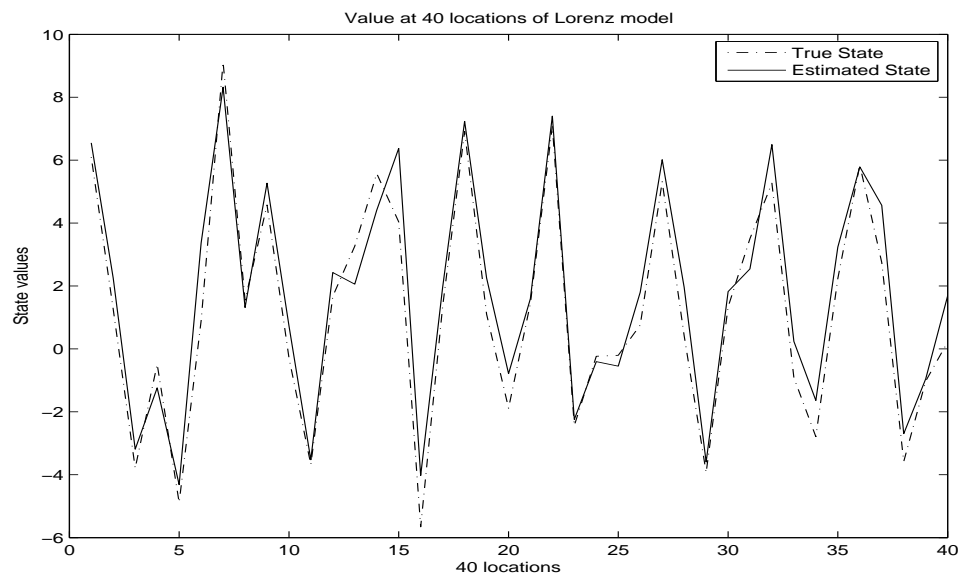


(a) States at 40 locations @ t=20

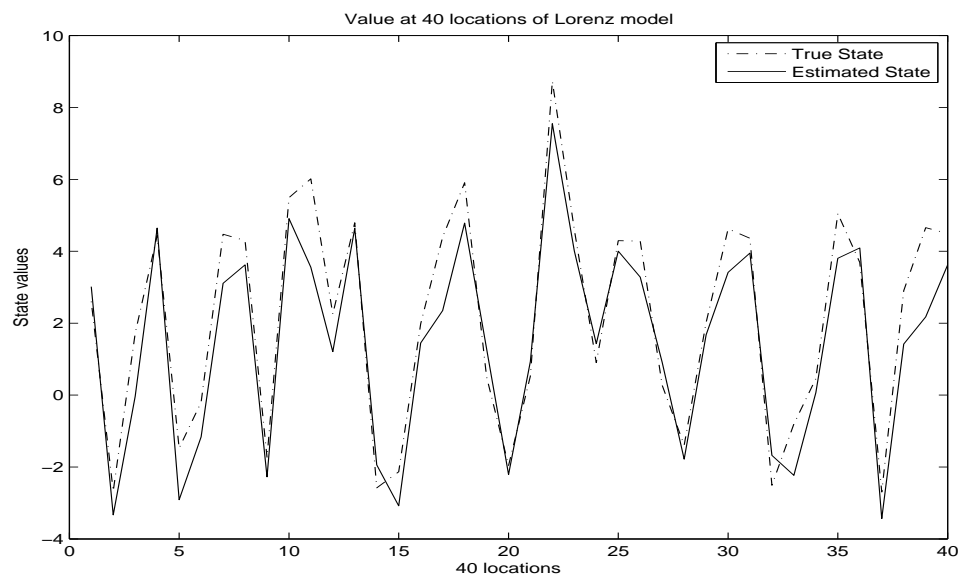


(b) States at 40 locations @ t=30

Figure 5.6: Estimated states at 40 locations of the Lorenz model estimated with the proposed algorithm at 20th and 30th time instants of estimation



(a) States at 40 locations @ t=40



(b) States at 40 locations @ t=50

Figure 5.7: Estimated states at 40 locations of the Lorenz model estimated with the proposed algorithm at 40th and 50th time instants of estimation

Chapter 6

Conclusions and Discussions

6.1 Main contributions

The main contributions in this thesis are as follows:

1. A non-parametric approach is used to estimate the model on-the-fly in the state estimation problem.
2. Fast summation, iterative methods and IVM based compression are used to propagate the model using a concise and relatively efficient data representation.
3. The proposed method was used with the Lorenz-96 toy model to test its performance

6.2 Summary

In an estimation problem, there can be three possible scenarios - (1) the state dynamical model can be known exactly, (2) the state dynamical model has an approximation available and (3) the model is not available at all.

If the model is known exactly and not difficult to compute, it is preferable to use the model directly. However, if the model evaluations are computationally infeasible, it is better to use an approximation (if available) of the model which is easier to compute. If the approximation is inaccurate or unavailable, dual estimation approaches should be followed. The estimation errors when different models are used is tabulated in Table 6.1.

Exact model:	With a good initialization, the estimation errors reduce as the iterations proceed.
Approximate model:	If the approximation of the actual model is not erroneous this method will yield good results
Estimated model:	If the model estimation is robust and learns the underlying model, the errors in dual estimation methods will reduce as iterations proceed, although initially the errors will be high.

Table 6.1: Estimation errors using different models

The usage of the different methods is shown in Table 6.2

Table 6.2: Usage for the different methods

Exact model:	If model is known and is not expensive to evaluate.
Approximate model:	If model is unknown or if the model is difficult to compute.
Estimated model:	If an approximation of the model is not available, or if the model is complex and the approximation available is not close to the actual model.

6.3 Further Possible Work

6.3.1 Irregular spatial and temporal data sampling

In the proposed approach, the sampling of the data is assumed to be complete. That is, there are no missing observations spatially and the observations are sampled at all possible intervals. This is too ideal an assumption to hold true practically. The observations might not be available at all spatial locations at all time instants. In other words there can be missing observations. With the proposed method, if the observations are available at every second time instant (ie. $y_t, y_{t+2}, y_{t+4}, \dots$), the model estimated will predict the current state using state two instants back (ie. $x_t = f(x_{t-2})$). Missing spatial observations will simply be ignored for

model estimation.

However, it would be interesting to look at different approaches to solving this problem without ignoring the missing observations and model the state predictions using a first-order Auto-Regressive model by accounting for the missing observations. One approach to explore would be method similar to [34], to move into a subspace to estimate the missing observations and use it for estimation.

6.3.2 IVM Complexity

The IVM used in the proposed algorithm is the bottle-neck in the overall complexity. It would be interesting to see if this step can be accelerated or replaced with a faster compression scheme.

The complexity introducing step in IVM is the entropy based data selection. IVM adds a data point, and checks the next suitable point that can be added from all points. While this works very well, it would be interesting to see if a simultaneous similarity measurement scheme can be devised which enables a selection of a representative subset of the data simultaneously instead of an incremental addition.

6.3.3 “Non-parametric” model for the parameters

While the approach proposed is very robust, there might cases where there is a model available but the parameters of the model are unknown. For example, in

the weather data assimilation, the system dynamics are generally defined by a set of differential equations with poorly known coefficients. In case of a particle filter based tracking, there are some standard motion models like Brownian motion to define the transition dynamics. In these case, more than developing a generic model for the dynamics itself, we might need a model to estimate the parameters of the system dynamics. In these scenarios, it is possible to use a dual Kalman filter or joint Kalman filter approaches. It would be interesting to explore a non-parametric approach to predict the system parameter in a framework similar to the one proposed in this thesis

Appendix A

More about GPR

In this appendix, the posterior and maximum likelihood estimation of the Gaussian process regression [35] is explained.

A.1 Posterior density

Consider a zero mean Gaussian process with associated training data $D = \{X, f\}$ and test data $D_* = \{X_*, f_*\}$. By the assumptions of a Gaussian Process, $\{f_*, f\}$ are jointly Gaussian, i.e,

$$\begin{bmatrix} f_* \\ f \end{bmatrix} \sim N(0, \hat{K}) \quad (\text{A.1})$$

$$\hat{K} = \begin{bmatrix} K_* & \tilde{K} \\ \tilde{K}^T & K \end{bmatrix} \quad (\text{A.2})$$

where, $K = K(X, X)$, $\tilde{K} = K(X, X_*)$ and $K_* = K(X_*, X_*)$.

Defining, $\Lambda = \begin{bmatrix} \Lambda_{11} & \Lambda_{12} \\ \Lambda_{21} & \Lambda_{22} \end{bmatrix} = \hat{K}^{-1}$, and writing the joint distribution of f and f_* ,

$$P \left(\begin{bmatrix} f_* \\ f \end{bmatrix} \right) = \frac{1}{(2\pi)^{N/2} \|\hat{K}\|^{1/2}} \exp \left(-\frac{1}{2} F^T \hat{K}^{-1} F \right) \quad (\text{A.3})$$

where N is the size of data X and $F = \begin{bmatrix} f \\ f_* \end{bmatrix}$. Considering the exponential part only in the above equation,

$$\text{exp part} = -\frac{1}{2} F^T \hat{K}^{-1} F \quad (\text{A.4})$$

$$= -\frac{1}{2} F^T \Lambda F \quad (\text{A.5})$$

$$= -\frac{1}{2} [f_*^T \Lambda_{11} f_* + 2f_*^T \Lambda_{12} f + f^T \Lambda_{22} f] \quad (\text{A.6})$$

If two variables are jointly Gaussian, the conditional probability of one give the other is also Gaussian, i.e., $P(f_*|f) \sim N(\mu, \Sigma)$. For such a Gaussian, the exponential part will be,

$$\text{exp part} = -\frac{1}{2} \left((f_* - \mu)^T \Sigma^{-1} (f_* - \mu) \right) \quad (\text{A.7})$$

$$= -\frac{1}{2} (f_*^T \Sigma^{-1} f_* - 2y_* \Sigma^{-1} \mu + \text{constant}) \quad (\text{A.8})$$

Comparing equations A.6 and A.8 and equating the linear and quadratic terms,

$$\Sigma = \Lambda_{11}^{-1} \quad (\text{A.9})$$

$$\mu = \Lambda_{11}^{-1} \Lambda_{12} y \quad (\text{A.10})$$

In order to get Λ_{11} and Λ_{12} , recall,

$$\begin{pmatrix} K(X_*, X_*) & K(X_*, X) \\ K(X, X_*) & K(X, X) \end{pmatrix}^{-1} = \begin{pmatrix} \Lambda_{11} & \Lambda_{12} \\ \Lambda_{21} & \Lambda_{22} \end{pmatrix} \quad (\text{A.11})$$

Using an identity in Matrix inversion,

$$\begin{pmatrix} A & B \\ B^T & D \end{pmatrix}^{-1} = \begin{pmatrix} M & -MBD^{-1} \\ -D^{-1}B^TM & D^{-1} + D^{-1}B^TMBD^{-1} \end{pmatrix} \quad (\text{A.12})$$

where $M = (A - BD^{-1}B^T)^{-1}$.

Using equations A.12 and A.11 in A.10 and A.9,

$$\Sigma = K(X_*, X_*) - K(X_*, X)K(X, X)^{-1}K(X, X_*) \quad (\text{A.13})$$

$$\mu = K(X_*, X)K(X, X)^{-1}f \quad (\text{A.14})$$

For noisy observations, y with variance σ^2I , replace K with K_y , where $K_y = K + \sigma^2I$,

$$\Sigma = K(X_*, X_*) - K(X_*, X)(K(X, X) + \sigma^2I)^{-1}K(X, X_*) \quad (\text{A.15})$$

$$\mu = K(X_*, X)(K(X, X) + \sigma^2I)^{-1}y \quad (\text{A.16})$$

This is what is given in the expression 2.7.

A.2 Maximum Likelihood

To evaluate the maximum likelihood estimate of the parameters of the likelihood given by

$$P(y|X, \Theta) = \frac{1}{(2\pi)^{N/2}|K_y|^{1/2}} \exp\left(-\frac{1}{2}yK_y^{-1}y\right) \quad (\text{A.17})$$

Taking log on both sides, the log-likelihood is,

$$\log(y|X, \Theta) = -\frac{1}{2}y^T K_y^{-1}y + \frac{1}{2} \log |K_y| - \frac{N}{2} \log(2\pi) \quad (\text{A.18})$$

Differentiating with respect to theta,

$$\frac{\partial}{\partial \Theta}(\log P(y|X, \Theta)) = -\frac{1}{2}y^T K_y^{-1T} \left(\frac{\partial K_y}{\partial \Theta}\right) K_y^{-1}y + \frac{1}{2} \text{tr} \left(K_y^{-1} \frac{\partial K_y}{\partial \Theta} \right) \quad (\text{A.19})$$

$$= -\frac{1}{2} \text{trace} \left((\alpha \alpha^T - K_y^{-1}) \frac{\partial K_y}{\partial \Theta} \right) \quad (\text{A.20})$$

where, $\alpha = K_y^{-1}y$. Using equation A.18 and A.20 with minimization approaches like Conjugate Gradient, it is possible to evaluate the hyperparameters Θ of the covariance function K .

Appendix B

Various steps in LEKF

Let the states be denoted by $x(\hat{r}, t)$, \hat{r} indicating the location (latitude and longitude) of the states in space and t indicating the time instant under consideration. The first step in LEKF is prediction of the current state from past state using equation 3.1.

$$x^b(\hat{r}, t) = f(x^a(\hat{r}, t - 1)) \quad (\text{B.1})$$

where f is the state transition dynamics of the system (similar to equation 1.1, superscript b indicated the background (predicted) state and superscript a indicates the analysis (updated) states. Before the normal update step of the Kalman filter there are two important steps in LEKF - Localization and Transformation.

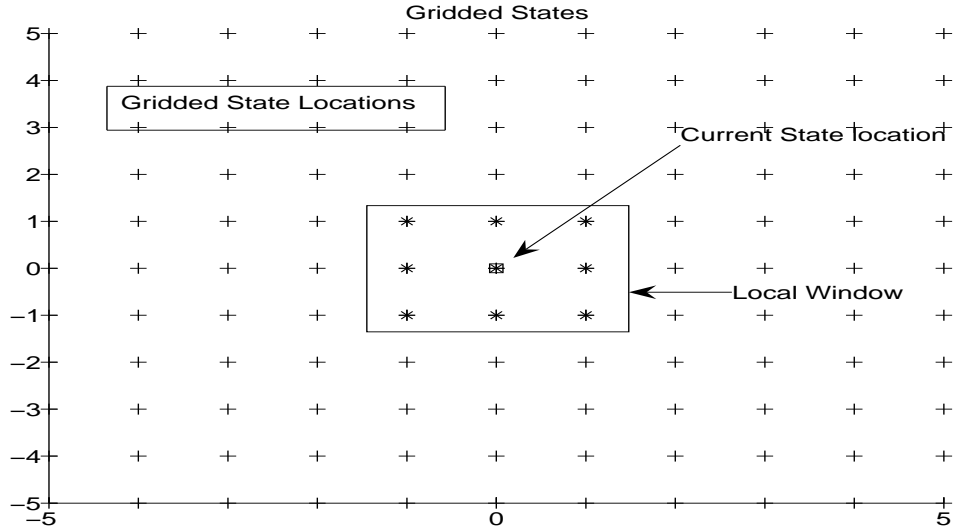


Figure B.1: States at grid locations in 2-Dimensions

B.1 Localization

Given $X^b(\hat{r}, t)$, a local window of size $l \times l$ is defined centered at the current location. The local state vector (denoted as $X_{mn}^b(\hat{r}, t)$) encompasses the state information in this local region only. For example, consider the grid in figure B.1. For the overall states appear as here, the local 3×3 window is defined around the state of consideration (denoted by the squared-asterisk). The localization step in LEKF is also similar to this.

B.2 Transformation

Once localized, it is possible to define the local mean and covariances from the localized ensembles,

$$\bar{x}_{mn}^b(\hat{r}, t) = \frac{1}{k} \sum_{i=1}^k x_{mn}^{b(i)}(\hat{r}, t) \quad (\text{B.2})$$

$$P_{mn}^b = X_{mn}^b X_{mn}^{bT} \quad (\text{B.3})$$

$$X_{mn}^b = (k)^{(1/2)} [x_{mn}^{b(1)} - \bar{x}_{mn}^b | x_{mn}^{b(2)} - \bar{x}_{mn}^b | \dots | x_{mn}^{b(k)} - \bar{x}_{mn}^b] \quad (\text{B.4})$$

where k is the size of the ensemble. If $\lambda_{mn}^{(i)}$ and $u_{mn}^{(i)}$ denote the i^{th} eigen-value and eigen-vector of the covariance matrix, P_{mn}^b , then

$$P_{mn}^b = \sum_{i=1}^k \lambda_{mn}^{(i)} u_{mn}^{(i)} u_{mn}^{(i)T} \quad (\text{B.5})$$

$$\approx \sum_{i=1}^{k'} \lambda_{mn}^{(i)} u_{mn}^{(i)} u_{mn}^{(i)T} \quad (\text{B.6})$$

where $k' < k$ indicates the size of a subspace in which the localized states can be represented. The eigen values and eigen vectors are assumed to be sorted from the largest value to the smallest. Using the reduced eigen space for defining a transformation matrix,

$$Q_{mn} = [u_{mn}^{(1)} | u_{mn}^{(2)} | \dots | u_{mn}^{(k')}] \quad (\text{B.7})$$

Now the transformation a vector w from the localized space to the lower subspace is given by

$$\hat{w} = Q_{mn}^T w \quad (\text{B.8})$$

and the transformation of a matrix from the localized space to the lower subspace is given by

$$\hat{M} = Q_{mn}^T M Q_{mn} \quad (\text{B.9})$$

Once transformed using Q , equation 3.2 can be used to update the background (predicted) \hat{x}_{mn}^b states to the analysis states, \hat{x}_{mn}^a . Once the analysis states at the subspace is available, it can be transformed to the local space by using Q^T instead of Q above.

The localization is performed for each state location and the final localized analysis state is taken as the global state. More detailed analysis of this can be found in [33] and [13].

Appendix C

Informative Vector Machine

C.1 Assumed Density Filtering (ADF)

Consider a regression model, $y = f(X) + \epsilon$, $\epsilon \sim N(0, \sigma^2 I)$. By the Gaussian process assumption, $P(f|X, \Theta) \sim N(0, K)$, where Θ are the set of hyperparameters.

$$P(y, f|X, \Theta) = P(f|X, \Theta) \prod_{n=1}^N P(y_n|f_n) \quad (\text{C.1})$$

From the earlier relations, $P(y_n|f_n) \sim N(f_n, \sigma^2)$.

Assumed density filtering (ADF) [22,23] is an online learning approach in which data points are absorbed one at a time. Here, equation C.1 is written as,

$$P(y, f) = \prod_{n=0}^N t_n(f) \quad (\text{C.2})$$

where $t_0 = P(f|X, \Theta)$ and $t_i = P(y_i|f_i)$. ADF takes advantage of this factorization to build an approximation $q(f)$ to the true posterior $P(f|y)$.

C.1.1 Active Set selection

Initially the active set (I) is empty. Using the factorization as in equation C.2, by adding a point n_1 to the active set, the new posterior is given by,

$$\hat{p}_1(f) \propto q_0(f)t_{n_1}(f) \quad (\text{C.3})$$

where $q_0(f) = t_0(f)$. The new approximation is given by,

$$KL(\hat{p}_1||q_1) = - \int \hat{p}_1(f) \log \frac{q_1(f)}{\hat{p}_1(f)} df \quad (\text{C.4})$$

More generally, for the inclusion of i^{th} point,

$$\hat{p}_i(f) = \frac{q_{i-1}(f)t_{n_i}(f)}{Z_i}; \quad (\text{C.5})$$

$$Z_i = \int q_{i-1}(f)t_{n_i}(f)df \quad (\text{C.6})$$

ADF minimizes $KL(\hat{p}_i|||q_i)$ to select the next point to be added to the active set I by using moment matching.

$$q_i(f) \sim N(\mu_i, \Sigma_i) \quad (\text{C.7})$$

C.2 Data Point Selection

In IVM, with the knowledge of $q_{i-1}(f)$, the next point n_i to be added is determined by the change in the entropy of the posterior process. The change in the entropy can be seen as a measure of reduction in the level of uncertainty. In other words, IVM selects a point that would add the most information to the posterior in the

ADF sense. The change in entropy is given by,

$$\Delta H = -\frac{1}{2} (\log |\Sigma_{i,n}| - \log |\Sigma_{i-1}|) \quad (\text{C.8})$$

Equation C.8 is the same as 4.3.

BIBLIOGRAPHY

- [1] A kalman filter based visual tracking algorithm for an object moving in 3d. In *IROS '95: Proceedings of the International Conference on Intelligent Robots and Systems-Volume 1*, page 342, Washington, DC, USA, 1995. IEEE Computer Society.
- [2] Sanjeev Arulampalam, Simon Maskell, and Neil Gordon. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50:174–188, 2002.
- [3] Hideki Asoh, Futoshi Asano, Takashi Yoshimura, and Kiyoshi Yamamoto. An application of a particle filter to bayesian multiple sound source tracking with audio and video information fusion. In *in Proc. Int. Conf. on Information Fusion (IF)*, pages 805–812, 2004.
- [4] K.J. Astrom and P Eykhoff. System identification - a survey. *Automatica*, pages 123–162, 1971.
- [5] J.T. Connor, R.D. Martin, and L.E. Atlas. Recurrent neural networks and robust time series prediction. *Neural Networks, IEEE Transactions on*, 5(2):240–254, Mar 1994.
- [6] H. Cox. On the estimation of state variables and parameters for noisy dynamic systems. *Automatic Control, IEEE Transactions on*, 9(1):5–12, Jan 1964.
- [7] Li Deng and Xuemin Shen. Maximum likelihood in statistical estimation of dynamic systems: decomposition algorithm and simulation results. *Signal Process.*, 57(1):65–79, 1997.
- [8] J. Farison, R. Graham, and Jr. Shelton, R. Identification and control of linear discrete systems. *Automatic Control, IEEE Transactions on*, 12(4):438–442, Aug 1967.
- [9] Zoubin Ghahramani and Sam T. Roweis. Learning nonlinear dynamical systems using an em algorithm. In *Advances in Neural Information Processing Systems 11*, pages 599–605. MIT Press, 1999.

- [10] Alexander G. Gray and Andrew W. Moore. N-body problems in statistical learning. In *Advances in Neural Information Processing Systems 13*, pages 521–527. MIT Press, 2000.
- [11] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund. Particle filters for positioning, navigation, and tracking. *Signal Processing, IEEE Transactions on*, 50(2):425–437, Feb 2002.
- [12] A. H. Abdul Hafez and C. V. Jawahar. Target model estimation using particle filters for visual servoing. In *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition*, pages 651–654, Washington, DC, USA, 2006. IEEE Computer Society.
- [13] B. R. Hunt, E. J. Kostelich, and I. Szunyogh. Efficient Data Assimilation for Spatiotemporal Chaos: a Local Ensemble Transform Kalman Filter. *ArXiv Physics e-prints*, November 2005.
- [14] Michael Isard and Andrew Blake. Condensation conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28, 1998.
- [15] Andrew H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, April 1970.
- [16] Simon J. Julier and Jeffrey K. Uhlmann. A new extension of the kalman filter to nonlinear systems. In *In Int. Symp. Aerospace/Defense Sensing, Simul. and Controls*, pages 182–193, 1997.
- [17] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME Journal of Basic Engineering*, (82 (Series D)):35–45, 1960.
- [18] H. Kaufman, J. Woods, S. Dravida, and A. Tekalp. Estimation and identification of two-dimensional images. *Automatic Control, IEEE Transactions on*, 28(7):745–756, Jul 1983.
- [19] Neil Lawrence. Probabilistic non-linear principal component analysis with gaussian process latent variable models. *J. Mach. Learn. Res.*, 6:1783–1816, 2005.
- [20] Neil Lawrence, Matthias Seeger, and Ralf Herbrich. Fast sparse gaussian process methods: The informative vector machine. In *Advances in Neural Information Processing Systems 15*, pages 609–616. MIT Press, 2003.
- [21] Neil D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *In NIPS*, page 2004, 2004.

- [22] Neil D. Lawrence and Ralf Herbrich. A sparse bayesian compression scheme - the informative vector machine. presented at nips 2001 workshop on kernel methods. In *Presented at NIPS 2001 Workshop on Kernel Methods*, 2001.
- [23] Neil D. Lawrence, John C. Platt, and Michael I. Jordan. Extensions of the informative vector machine. In *In*. Springer-Verlag, 2005.
- [24] Dongryeol Lee, Alexander Gray, and Andrew Moore. Dual-tree fast gauss transforms. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 747–754. MIT Press, Cambridge, MA, 2006.
- [25] L. Ljung. Convergence analysis of parametric identification methods. *Automatic Control, IEEE Transactions on*, 23(5):770–783, Oct 1978.
- [26] L. Ljung. Asymptotic behavior of the extended kalman filter as a parameter estimator for linear systems. *Automatic Control, IEEE Transactions on*, 24(1):36–50, Feb 1979.
- [27] E. Lorenz. Predictability - a problem partly solved. *Proceedings on predictability held at ECMWF*, 1996.
- [28] E. N. Lorenz and K. A. Emanuel. Optimal Sites for Supplementary Weather Observations: Simulation with a Small Model. *Journal of Atmospheric Sciences*, 55:399–414, February 1998.
- [29] H. Moradkhani, S. Sorooshian, H. V. Gupta, and P. R. Houser. Dual state parameter estimation of hydrological models using ensemble kalman filter. *Advances in Water Resources*, 28:135–147, February 2005.
- [30] H. Moradkhani, S. Sorooshian, H. V. Gupta, and P. R. Houser. Dual state parameter estimation of hydrological models using ensemble Kalman filter. *Advances in Water Resources*, 28:135–147, February 2005.
- [31] Vlad I. Morariu, Balaji Vasan Srinivasan, Vikas C. Raykar, Ramani Duraiswami, and Larry S. Davis. Automatic online tuning for fast gaussian summation. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.
- [32] L. Nelson and E. Stear. The simultaneous on-line estimation of parameters and states in linear systems. *Automatic Control, IEEE Transactions on*, 21(1):94–98, Feb 1976.
- [33] 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*, 56:415–428, 2004.

- [34] S.R. Rao, R. Tron, R. Vidal, and Yi Ma. Motion segmentation via robust subspace separation in the presence of outlying, incomplete, or corrupted trajectories. *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008.
- [35] Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, December 2005.
- [36] Yogesh Rathi, Namrata Vaswani, Allen Tannenbaum, and Anthony Yezzi. Particle filtering for geometric active contours with application to tracking moving and deforming objects. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 2*, pages 2–9, Washington, DC, USA, 2005. IEEE Computer Society.
- [37] Vikas C. Raykar and Ramani Duraiswami. Fast large scale gaussian process regression using approximate matrix-vector products. *Learning workshop*, March 2007.
- [38] Vikas C. Raykar and Ramani Duraiswami. The improved fast gauss transform with applications to machine learning. In Leon Bottou, Olivier Chapelle, Dennis Decoste, and Jason Weston, editors, *Large Scale Kernel Machines*, pages 175–201. MIT Press, 2007.
- [39] RJ Orford RE Kopp. Linear regression applied to system identification for adaptive control systems. *DTIC Research Report*, pages 2300–2306, October 1963.
- [40] Valeria Simoncini and Daniel B. Szyld. Theory of inexact krylov subspace methods and applications to scientific computing. *SIAM J. Sci. Comput.*, 25(2):454–477, 2003.
- [41] Xiaodian Sun, Li Jin, and Momiao Xiong. Extended kalman filter for estimation of parameters in nonlinear state-space models of biochemical networks. *PLoS ONE*, 3(11):e3758, Nov 2008.
- [42] M. Tipping. The relevance vector machine. In *Advances in Neural Information Processing Systems, San Mateo, CA*. Morgan Kaufmann, 2000.
- [43] A. Van Rhijn and J.D. Mulder. Optical tracking and automatic model estimation of composite interaction devices. *Virtual Reality Conference, 2006*, pages 135–142, March 2006.

- [44] M. Vauhkonen, P.A. Karjalainen, and J.P. Kaipio. A kalman filter approach to track fast impedance changes in electrical impedance tomography. *Biomedical Engineering, IEEE Transactions on*, 45(4):486–493, April 1998.
- [45] E. Wan and A. Nelson. Dual kalman filtering methods for nonlinear prediction, 1997.
- [46] Eric A. Wan and Rudolph Van Der Merwe. The unscented kalman filter for nonlinear estimation. In *Proceedings of Symposium 2000 on Adaptive Systems for Signal Processing, Communication and Control*, pages 153–158, 2000.
- [47] Eric A. Wan, Rudolph Van Der Merwe, and Alex T. Nelson. Dual estimation and the unscented transformation. In *Neural Information Processing Systems*, pages 666–672. MIT Press, 2000.
- [48] Eric A. Wan and Alex T. Nelson. Neural dual extended kalman filtering: applications in speech enhancement and monaural blind signal separation. In *Proc. of IEEE Workshop on Neural Networks and Signal Processing*, pages 466–475, 1997.
- [49] J.M. Wang, D.J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(2):283–298, Feb. 2008.
- [50] C. K. I. Williams. Prediction with gaussian processes: from linear regression to linear prediction and beyond. pages 599–621, 1999.
- [51] Y. C. Wong and M. K. Sundareshan. Equivalent velocity tracking model for estimation of target maneuvers and design of neural network-based tracking algorithms. In O. E. Drummond, editor, *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, volume 3373 of *Presented at the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, pages 166–180, September 1998.
- [52] Liu Xin, Pei Hailong, and Li Jianqiang. Trajectory prediction based on particle filter application in mobile robot system. *Control Conference, 2008. CCC 2008. 27th Chinese*, pages 389–393, July 2008.