# PARAMETER ESTIMATION FOR NONLINEAR STATE SPACE MODELS

by

Jessica Wong

Submitted in partial fulfillment of the requirements
for the degree of Master of Science

at

Dalhousie University
Halifax, Nova Scotia
April 2012

DALHOUSIE UNIVERSITY

DEPARTMENT OF MATHEMATICS AND STATISTICS

The undersigned hereby certify that they have read and recommend to the Faculty of Graduate Studies for acceptance a thesis entitled "PARAMETER ESTIMATION FOR NONLINEAR STATE SPACE MODELS" by Jessica Wong in partial fulfillment of the requirements for the degree of Master of Science.

Dated: April 23, 2012

Supervisor: ______________________________

Readers: ______________________________

______________________________

DALHOUSIE UNIVERSITY

DATE: April 23, 2012

AUTHOR: Jessica Wong

TITLE: PARAMETER ESTIMATION FOR NONLINEAR STATE SPACE MODELS

DEPARTMENT OR SCHOOL: Department of Mathematics and Statistics

DEGREE: M.Sc. CONVOCATION: October YEAR: 2012

Signature of Author

# TABLE OF CONTENTS

# List of Tables

# LIST OF FIGURES

# ABSTRACT

This thesis explores the methodology of state, and in particular, parameter estimation for time series datasets. Various approaches are investigated that are suitable for nonlinear models and non-Gaussian observations using state space models. The methodologies are applied to a dataset consisting of the historical lynx and hare populations, typically modeled by the Lotka-Volterra equations. With this model and the observed dataset, particle filtering and parameter estimation methods are implemented as a way to better predict the state of the system. Methods for parameter estimation considered include: maximum likelihood estimation, state augmented particle filtering, multiple iterative filtering and particle Markov chain Monte Carlo (PMCMC) methods. The specific advantages and disadvantages for each technique are discussed. However, in most cases, PMCMC is the preferred parameter estimation solution. It has the advantage over other approaches in that it can well approximate any posterior distribution from which inference can be made.

# List of Abbreviations and Symbols Used

| | |
|---|---|
| $\mathbf{A}'$ | transpose of matrix $\mathbf{A}$ |
| $\mathbf{A}^{-1}$ | inverse of matrix $\mathbf{A}$ |
| $p(.)$ | probability distribution or density function |
| $L(.)$ | likelihood function |
| $logL(.)$ | log-likelihood function |
| $N(\mu, \Sigma)$ | Gaussian distribution with mean $\mu$ and covariance $\Sigma$ |
| $U(a, b)$ | Uniform distribution in the region $(a, b)$ |
| t | time index |
| $\tau$ | lag index |
| $\sim$ | distributed according to |
| ACF | autocorrelation function |
| AR | auto regressive |
| EKF | extended Kalman filter |
| EnKF | ensemble Kalman filter |
| LR | likelihood ratio |
| MCMC | Markov chain Monte Carlo |
| MH | Metropolis Hastings |
| MIF | multiple iterative filtering |
| ML | maximum likelihood |
| MLE | maximum likelihood estimate |
| MSE | mean squared error |
| PF | particle filter |
| PMCMC | particle Markov chain Monte Carlo |
| RMSE | root mean squared error |
| SIR | sequential importance resampling |
| SIS | sequential importance sampling |

# ACKNOWLEDGEMENTS

I would like to acknowledge first, Dr. Michael Dowd, my thesis supervisor for his patience, support and guidance. His positive attitude and insight during our weekly meetings was a major factor and essential in the completion of this thesis. Also, thank you to Dr. Joanna Mills Flemming and Dr. Bruce Smith for their agreement to be readers in this examination process. My gratitude goes to the faculty and administrative staff who provided a wonderful environment in which to complete this Masters degree. My appreciation also goes to the Department of Mathematics and Statistics and NSERC Discovery for financial support. And last but not least, to my amazing family and friends for their endless support and faith in me.

# CHAPTER 1

# INTRODUCTION

When a dataset is characterized by a temporal dependence, it is often appropriate to treat it as a time series and describe it by a state space model. State space modeling relies on a dynamical model that is used to interpret the observations. The state space framework usually consists of two equations: the system equation and the observation equation. The system equation, or the dynamical model, describes the true state of the system, which is modeled as a Markov process, while the observation equation relates the unobservable true states to the observed data. This allows for inference on problems relating to the true states and parameters of the model.

The focus of this thesis explores the methodology of state, and in particular, parameter estimation for time series datasets using state space models. Various methodologies are explored for parameter estimation suitable for nonlinear dynamic models and non-Gaussian observations. Estimation of the states and parameters can be realized with the use of the observed dataset and a dynamic model using filtering techniques. Filtering techniques such as the Kalman filter for linear systems and sequential importance resampling (SIR) for (linear and) nonlinear systems are used for state estimation. When applied to a dataset modeled by a state space model, SIR estimates the true value of the system at observed time points. Methods for parameter estimation also exist and are built upon the foundation of state estimation. They include, maximum likelihood estimation (MLE), state augmented particle filtering, multiple iterative filtering (MIF) and particle Markov chain Monte Carlo (PMCMC) methods. In this thesis, parameter estimation is used to estimate the parameters that describe the dynamics of the model. These techniques are discussed further in Chapter 2.

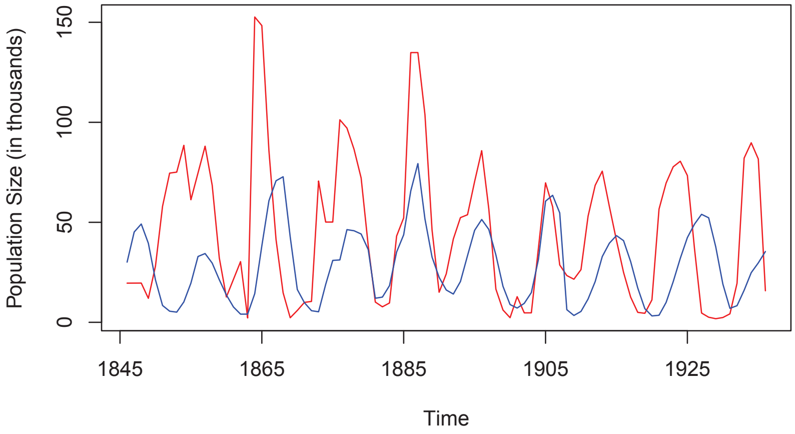


Figure 1.1: Time series plot of Lynx-Hare population (in thousands). Shown are the populations of the lynx (blue line) and the hare (red line).

The methodologies considered are then applied to a dataset consisting of Canadian lynx and snowshoe hare populations over a 90 year period, which is described below. The motivation is to estimate the state of the populations at each observed time point and to recover optimal parameter estimates that describe the lynx and hare populations. This can be accomplished by applying a discrete time, state space model to the dataset. Figure 1.1 is a time series plot of the dataset with a yearly time interval. The figure shows two oscillating series of observations where the blue line represents the lynx population and the red line represents the hare population. These oscillations are frequently cited in textbooks and analyzed by biologist and ecologists as it shows the characteristics of a predator-prey relationship (*Hewitt*, 1921; *May*, 1973; *Murray*, 1989; *Krebs et al.*, 2001). This classic time series is based on the historical fur trading records from the Hudson's Bay Company and has the assumption that the recorded number of trapped animals is proportional to the animals' population.

The lynx-hare time series dataset is often modeled by the Lotka-Volterra predator-prey equations. They are coupled nonlinear differential equations that describe the growth rates of the predator and prey populations. The parameters in the Lotka-Volterra equations consider the population growth of the prey in the absence of the predator and the decay of the predator in the absence of the prey. In addition, the Lotka-Volterra equations also consider the rates of growth and decay when interactions between the predator and prey occur. General state space models are introduced and applied to the lynx-hare dataset below to estimate these dynamic parameters, as well as the population levels.

## 1.1 General State Space Models

State space modeling estimates the states and the parameters of interest by considering the system and observations simultaneously, and by separating the errors for the process and the observations. Below is the system equation and the observation equation of the general state space model:

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}, \boldsymbol{\theta}) + \mathbf{e}_t^m \tag{1.1}$$

$$\mathbf{y}_t = g(\mathbf{x}_t, \boldsymbol{\phi}) + \mathbf{e}_t^o \tag{1.2}$$

where

$\mathbf{x}_t$: system state vector at time t

$\mathbf{e}_t^m$: system error vector at time t

$f(.)$: function describing the dynamics

$\boldsymbol{\theta}$: dynamical parameter

$\mathbf{y}_t$: observation vector at time t

$\mathbf{e}_t^o$: observation error vector at time t

$g(.)$: observation operator that relates $\mathbf{x}_t$ to $\mathbf{y}_t$

$\boldsymbol{\phi}$: parameter of observation operator.

The system equation describes the evolution of the unobserved system state, $\mathbf{x}_t$, whereas the observation equation relates the observations, $\mathbf{y}_t$, to the system state, $\mathbf{x}_t$. When a suitable state space model is defined for a dataset, filtering can be used to construct the filter probability density, $p(\mathbf{x}_t|\mathbf{Y}_t)$. The filter density is a probability function of the current state given all observations, $\mathbf{Y}_t$, which represents the set of observation vectors up to and including time t, such that, $\mathbf{Y}_t = \{\mathbf{y}_i, i = 1, ..., t\}$. As such, the states and parameters of interest can be estimated from this distribution. For example, equations (1.1) and (1.2) can be applied to the lynx-hare dataset, in which the Lotka-Volterra equations (or the predator-prey model) are used as the system equation, $f(.)$. (Note: the Lotka-Volterra equations are presented in further detail in Chapter 3.) In this example, the states and parameters of interest are: $\mathbf{x}_t$, the vector that contains the true (unobserved) lynx and hare populations at time t; and $\boldsymbol{\theta}$, the vector containing the parameters that describe the growth rates and death rates of the populations. The vector, $\mathbf{y}_t$, contains the observed lynx and hare populations and are used within the filtering solution as a means of estimating the states and parameters.

## 1.2 Filtering

When a dataset follows the framework of a nonlinear and non-Gaussian stochastic process, it is appropriate to use Bayesian methods to provide a probabilistic solution for state (and parameter) estimation problems. Here, Bayesian filtering is used to construct the filter probability density, $p(\mathbf{x}_t|\mathbf{Y}_t)$; or in words, to find the best estimate of the true system state given (noisy) observations of the system. In the lynx-hare example, given the observations of the lynx and hare populations over 90 years, the dataset is assumed to follow the predator-prey model with additive noise (i.e. a dynamic-stochastic nonlinear model) and filtering is an appropriate solution method to the dataset. Below, is the probabilistic solution to the filtering problem and follows the material in *Ristic et al.* (2004).

Assuming that all information up to time t-1 is available, (i.e. $p(\mathbf{x}_{t-1}|\mathbf{Y}_{t-1})$ is available) and using the system model, it is possible to obtain the predictive density of the state at time t. The predictive density is:

$$p(\mathbf{x}_t|\mathbf{Y}_{t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{Y}_{t-1})d\mathbf{x}_{t-1} \tag{1.3}$$

where the probabilistic model, $p(\mathbf{x}_t|\mathbf{x}_{t-1})$, is defined by the system equation, equation (1.1), and the known distribution of the system error, $\mathbf{e}_t^m$. At the next discrete time step, when new observations become available, the predictive density can be updated to include this new information. Bayes' rule is used to obtain the filter density:

$$p(\mathbf{x}_t|\mathbf{Y}_t) = \frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{Y}_{t-1})}{p(\mathbf{y}_t|\mathbf{Y}_{t-1})}. \tag{1.4}$$

The denominator, $p(\mathbf{y}_t|\mathbf{Y}_{t-1})$, can be evaluated by:

$$p(\mathbf{y}_t|\mathbf{Y}_{t-1}) = \int p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{Y}_{t-1})d\mathbf{x}_t. \tag{1.5}$$

Also, notice that $p(\mathbf{y}_t|\mathbf{x}_t)$ is known from the observation model as seen in equation (1.2) and is based on the distribution of the observation error, $\mathbf{e}_t^o$. Take note that in order to apply a filtering algorithm, it is required that the initial probability function, $p(\mathbf{x}_0)$, is specified.

The integrals used to evaluate the predictive and filter densities can be estimated using various filtering algorithms (described and examined in Chapter 2). Approaches for parameter estimation are built on this foundation and are the major focus of this thesis.

## 1.3 Thesis Overview

This thesis is structured as follows. Chapter 2 begins with the Kalman filter, a state estimation filtering technique for linear and Gaussian systems. Likelihood-based parameter estimation using the Kalman filter is then explained. Next, sequential importance resampling, a filtering method that can handle nonlinear systems is presented along with methods for parameter estimation. These methods include maximum likelihood estimation, state augmented particle filtering, and multiple iterative filtering. In addition, Markov chain Monte Carlo (MCMC) is used to improve the estimation of the states and parameters and to make its implementation more robust. This technique, called particle MCMC is used to estimate the dynamical parameters in state space models and is the ultimate focus of this thesis. Further, for each technique introduced in Chapter 2, a simple example is implemented using a synthetic dataset derived from a linear/Gaussian system. A comparison of each method's performance is also carried-out. Chapter 3 provides a description of the predator-prey model; a model that is assumed to define the lynx-hare dataset. The lynx-hare dataset is then analyzed using a classic population modeling technique. As well, the particle filtering techniques for state and parameter estimation from Chapter 2 are reexamined in the nonlinear/non-Gaussian setting. Examples using a synthetic predator-prey dataset and the real lynx-hare dataset are performed. A comparison of the methods are then undertaken. Finally, Chapter 4 concludes the thesis and includes some discussion on the advantages and disadvantages of each method.

# Chapter 2

# Background

The purpose of this chapter is to provide a background for filtering for state and parameter estimation and to illustrate its applications using simple examples. The examples will use a synthetic dataset modeled by an autoregressive process of order 1 (AR(1)). The chapter begins with the Kalman filter; a foundational method for state and parameter estimation in the linear and Gaussian setting. It will then progress to the more generalized case of the particle filter which can be applied to nonlinear models. The Kalman filter and particle filter sections will include a variety of methods for parameter estimation along with examples for state and parameter estimation. Further, Markov chain Monte Carlo (MCMC) update methods will be blended with particle filtering in two different ways as an attempt to reduce the loss of diversity in the ensemble of particles and for a more robust method for state and parameter estimation, especially static parameters.

## 2.1 The Kalman Filter

The Kalman filter is an algorithm used to estimate the state and/or parameters of a linear system with Gaussian noise. It recursively predicts and updates using the system model and observations to produce an optimal estimate of the state (*Kalman*, 1960). This section provides details of the Kalman filter algorithm and an example using a synthetic dataset. The example illustrates the filtering problem for state and parameter estimation in a linear/Gaussian setting.

Consider the following linear and discrete time state space model. Shown are its system and observation equations (respectively):

$$\mathbf{x}_t = \mathbf{D}_t\mathbf{x}_{t-1} + \mathbf{G}_t\mathbf{e}_t^m \tag{2.1}$$

$$\mathbf{y}_t = \mathbf{H}_t\mathbf{x}_t + \mathbf{e}_t^o \tag{2.2}$$

where

$\mathbf{x}_t$: system state vector at time t (m x 1)

$\mathbf{D}_t$: dynamics matrix (m x m)

$\mathbf{e}_t^m$: system noise (m x 1)

$\mathbf{G}_t$: matrix projecting the noise to the system (m x m)

$\mathbf{y}_t$: vector of observations at time t (n x 1)

$\mathbf{H}_t$: observation operator matrix (n x m)

$\mathbf{e}_t^o$: observation error (n x 1).

Here, it is assumed that the system noise, $\mathbf{e}_t^m$, and observation noise, $\mathbf{e}_t^o$, are normally distributed with a mean of zero. In addition, they are assumed to be sequences of independent random variables and independent of each other (i.e. Cov($\mathbf{e}_t^m$ , $\mathbf{e}_{t+\tau}^m$) = 0, Cov($\mathbf{e}_t^o$ , $\mathbf{e}_{t+\tau}^o$) = 0 for $\tau \neq 0$ and Cov($\mathbf{e}_t^m$ , $\mathbf{e}_{t+\tau}^o$) = 0 $\forall\tau$). The second moment of the system noise and observation noise also need to be specified. They are denoted by $\mathbf{Q}_t$ for $\mathbf{e}_t^m$ and $\mathbf{R}_t$ for $\mathbf{e}_t^o$ (i.e. $\mathbf{e}_t^m \sim$ N(0, $\mathbf{Q}_t$) and $\mathbf{e}_t^o \sim$ N(0, $\mathbf{R}_t$)). Also note, since $\mathbf{e}_t^m$ and $\mathbf{e}_t^o$ are normal random vectors, the system state, $\mathbf{x}_t$, is also a normal vector.

The filtering solution for state space models requires two essential steps: the prediction step and the measurement step. In the Kalman filter algorithm, the mean and covariance of the state, $\mathbf{x}_t$, are computed at both steps. First, some notation is established. Denote the predictive estimate of $\mathbf{x}_t$ by $\hat{\mathbf{x}}_{t|t-1}$; meaning $\mathbf{x}_t$ is estimated given observations only up to time step t-1. Likewise, denote the filter estimate of $\mathbf{x}_t$ by $\hat{\mathbf{x}}_{t|t}$, which estimates $\mathbf{x}_t$ given observations up to and including time step t. Finally, denote the error covariance matrix of $\hat{\mathbf{x}}_{t|t-1}$ and $\hat{\mathbf{x}}_{t|t}$ by $\mathbf{M}_t$ and $\mathbf{P}_t$, respectively.

In words, (given the filter estimates of the mean and covariance at time t-1) a single-stage transition of the Kalman filter algorithm begins with the filter density at time t-1,

$N(\hat{\mathbf{x}}_{t-1|t-1}, \mathbf{P}_{t-1})$. The Kalman filter uses the information at time t-1 and the system model to determine the predictive density, $N(\hat{\mathbf{x}}_{t|t-1}, \mathbf{M}_t)$. When new observations, $\mathbf{y}_t$, become available, the prediction is refined to produce a new filter density at time t, $N(\hat{\mathbf{x}}_{t|t}, \mathbf{P}_t)$. The algorithm below describes how the mean and covariance of $\mathbf{x}_t$ are calculated at each step.

**The Kalman Filter Algorithm**

Setup initial estimates: $\mathbf{x}_0 \sim N(\boldsymbol{\mu}_{x_0}, \mathbf{P}_0)$

For t = 1, 2, ... , N

Prediction Step:

- compute predictive mean: $\hat{\mathbf{x}}_{t|t-1} = \mathbf{D}_t\hat{\mathbf{x}}_{t-1|t-1}$
- compute prediction covariance: $\mathbf{M}_t = \mathbf{D}_t\mathbf{P}_{t-1}\mathbf{D}'_t + \mathbf{G}_t\mathbf{Q}_t\mathbf{G}'_t$

Measurement Step:

- compute filter covariance: $\mathbf{P}_t = (\mathbf{M}_t^{-1} + \mathbf{H}'_t\mathbf{R}_t^{-1}\mathbf{H}_t)^{-1}$
- compute filter mean: $\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - \mathbf{H}_t\hat{\mathbf{x}}_{t|t-1})$

where $\mathbf{K}_t$ is called the Kalman gain matrix and is calculated by: $\mathbf{K}_t = \mathbf{P}_t\mathbf{H}'_t\mathbf{R}_t{}^{-1}$

End For (t).

### 2.1.1 State Estimation

To demonstrate the application of the Kalman filter, an example using a dataset simulated from an AR(1) process is shown in Figure 2.1. Consider the simple state space model:

$$x_t = ax_{t-1} + e_t^m, \qquad e_t^m \sim N(0, \sigma_m) \tag{2.3}$$

$$y_t = x_t + e_t^o, \qquad e_t^o \sim N(0, \sigma_o) \tag{2.4}$$

where $x_t$, $y_t$ and $a$ are all scalars and $e_t^m$ and $e_t^o$ are purely random processes and independent in time and of one another.

For the purpose of this thesis, all examples in this chapter will use the same synthetic dataset. The dataset's 500 observations are generated from an AR(1) process that defines the true state, $x_t$, to which normally distributed observation error is added. Referring to equations (2.3) and (2.4), $a$ = 0.95, $\sigma_m$ = 1 and $\sigma_o$ = 1. The initial estimates specific to this example are $\mathbf{x}_0 = 0$ and $\mathbf{P}_0 = 0$. Figure 2.1 shows the results. The results show that the Kalman filter can estimate the system state very closely to its true value and see through the noisy observations.

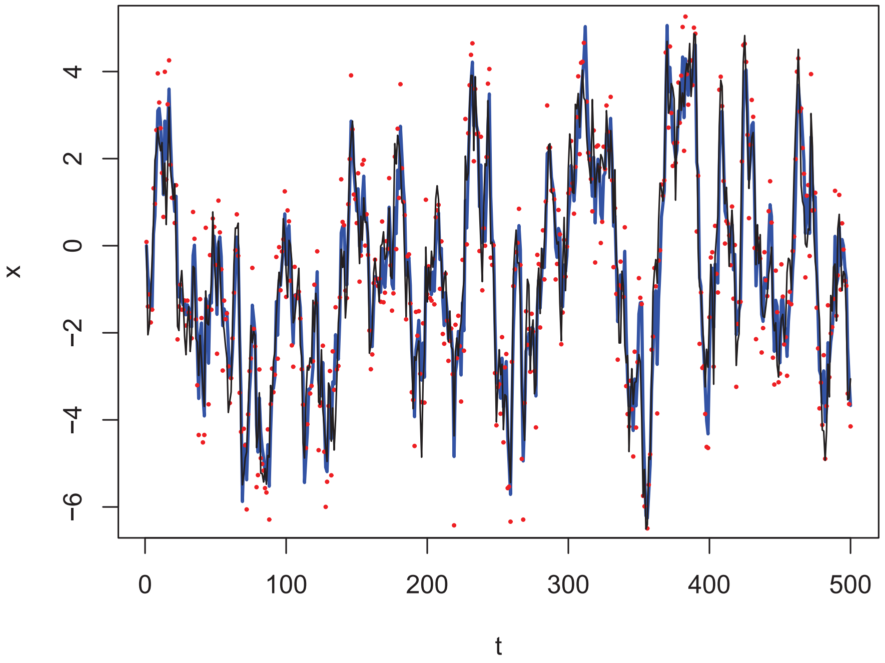


Figure 2.1: Estimated state of the AR(1) process. Shown are the observations (red dots), the true state (thin black line) and the Kalman filter estimates (thick blue line).

### 2.1.2 *Likelihood-Based Parameter Estimation*

Parameter estimation in state space models is used to estimate either the dynamical parameters or statistical parameters. For the purpose of this thesis, the statistical parameters will be specified, while the dynamical parameters will be estimated using a variety of methods. For example, referring back to equations (2.3) and (2.4), $a$ is the static parameter that will be estimated, while $\sigma_m$ and $\sigma_o$ are the statistical parameters that will be specified.

In order to estimate the dynamical parameters using the likelihood function, the innovations (the one step ahead prediction errors), $\mathbf{v_t}$, and its covariance error, $\mathbf{F}_t$, need to be defined. The innovations are given by:

$$\mathbf{v}_t = \mathbf{y}_t - \hat{\mathbf{y}}_{t|t-1} \tag{2.5}$$

where $\hat{\mathbf{y}}_{t|t-1} = \mathbf{H}_t\hat{\mathbf{x}}_{t|t-1}$. The innovations' error covariance matrix from *Harvey* (1991) is given by:

$$\mathbf{F}_t = \mathbf{H}_t\mathbf{M}_t\mathbf{H}_t' + \mathbf{R}_t. \tag{2.6}$$

Maximum likelihood estimation determines which values of the parameters are most consistent given the available data and finds the parameters which maximize the likelihood function. First, suppose the vector of unknown parameters is denoted by $\boldsymbol{\theta}$. Then the corresponding likelihood function of $\boldsymbol{\theta}$ is defined by:

$$\begin{aligned} L(\boldsymbol{\theta}|\mathbf{Y}_N) &= p(\mathbf{Y}_N|\boldsymbol{\theta}) \\ &= \prod_{t=1}^{N} p(\mathbf{y}_t|\mathbf{Y}_{t-1}, \boldsymbol{\theta}) \\ &= \prod_{t=1}^{N} p(\mathbf{v}_t|\boldsymbol{\theta}). \end{aligned}$$

Note that, $\mathbf{v}_t \sim N(0, \mathbf{F}_t)$. As such, the log-likelihood function can be written as:

$$logL(\boldsymbol{\theta}|\mathbf{Y}_N) = constant - \frac{1}{2}\sum_{t=1}^{N} log|\mathbf{F}_t| - \frac{1}{2}\sum_{t=1}^{N} \mathbf{v}_t'\mathbf{F}_t^{-1}\mathbf{v}_t. \tag{2.7}$$

Typically, the log-likelihood function is analyzed instead of the likelihood function since it is more convenient. As long as the data being analyzed follows a linear system with Gaussian noise, equation (2.7) can be evaluated using the Kalman filter. The value that maximizes equation (2.7) is the maximum likelihood estimate (MLE). To illustrate, consider the example dataset from the AR(1) process. As a reminder, the true value of the parameter is $a$ = 0.95. The log-likelihood for various values of $a$ are shown in Table 2.1. The parameter corresponding to the largest log-likelihood value is the MLE. In this example, $a$ = 0.91 is the MLE. However, take note that the likelihood is relatively flat for values of $a$ from 0.90 to 0.93.

Table 2.1: Results of likelihood-based parameter estimation using the Kalman filter.

| Value of $a$ | Log-likelihood |
|---|---|
| 0.85 | -943.9872 |
| 0.86 | -942.3947 |
| 0.87 | -941.0695 |
| 0.88 | -940.0148 |
| 0.89 | -939.2336 |
| 0.90 | -938.7286 |
| 0.91 | -938.5022 |
| 0.92 | -938.5565 |
| 0.93 | -938.8934 |
| 0.94 | -939.5143 |
| 0.95 | -940.4206 |
| 0.96 | -941.6130 |
| 0.97 | -943.0923 |
| 0.98 | -944.8586 |
| 0.99 | -946.9120 |
| 1.00 | -949.2523 |

## 2.2 The Particle Filter

When a dataset does not follow a linear system with Gaussian noise, the Kalman filter, is no longer an appropriate solution. However, there exists modifications to the Kalman filter that can approximately handle these nonlinear and non-Gaussian cases, such as the extended Kalman filter (EKF) and the ensemble Kalman filter (EnKF) (*Evensen*, 1992, 1994). In addition, particle filtering is also a useful solution technique that can provide exact solutions for nonlinear and non-Gaussian systems. It is a foundational method for state and parameter estimation procedures considered later on in this thesis. Particle filters rely on a set of samples, or ’particles’ that approximately represent the distribution of the system state at both the prediction and measurement steps. With the estimated distributions, approximations of relevant statistics can be computed.

This section provides the details behind particle filtering and an introduction to various available methods for parameter estimation. For simplicity, examples of these methods will also be illustrated for state and parameter estimation using the same synthetic dataset generated from the AR(1) process. Chapter 3 will provide examples of these methods in a nonlinear and non-Gaussian setting which extends the basic methodology.

### *2.2.1 Sequential Importance Resampling (SIR)*

The goal of particle filtering is to construct and sample from an appropriate probability function of the current state given all observations (i.e. $p(\mathbf{x}_t|\mathbf{Y}_t)$). The most common particle filter is sequential importance resampling (SIR) introduced by *Gordon et al.* (1993). Consider the general state space model from equations (1.1) and (1.2). Suppose at time t-1, a weighted sample from the following probability distribution is known:

$$\{\mathbf{x}_{t-1|t-1}^{(i)}, w_{t-1}^{(i)}\} \sim p(\mathbf{x}_{t-1}|\mathbf{Y}_{t-1}), \tag{2.8}$$

where

$\{.\}$ represents the ensemble of particles or the sample

$\mathbf{x}_{t-1|t-1}^{(i)}$ are weighted particles distributed as $p(\mathbf{x}_{t-1}|\mathbf{Y}_{t-1})$

(i) indexes each particle with $w_{t-1}^{(i)}$ as its corresponding normalized weight (i = 1, ... , M).

The SIR algorithm is run from t = 1, ..., N, where N, represents the discrete time point corresponding to the most recent available observation. A single iteration of the SIR algorithm is essentially a transition of the particle set from $\{\mathbf{x}_{t-1|t-1}^{(i)}\} \sim p(\mathbf{x}_{t-1}|\mathbf{Y}_{t-1})$ to $\{\mathbf{x}_{t|t}^{(i)}\} \sim p(\mathbf{x}_t|\mathbf{Y}_t)$ where the ensemble of particles in $\{\mathbf{x}_{t-1|t-1}^{(i)}\}$ and $\{\mathbf{x}_{t|t}^{(i)}\}$ are unweighted sample sets. Note that the single iteration of the SIR algorithm begins and ends with an unweighted ensemble set, however, in other particle filters, a weighted ensemble set is updated at every iteration (such as sequential importance sampling where the particle set remains weighted throughout as no resampling takes place). The algorithm for a single iteration of SIR is provided below:

Prediction Step:

- Each particle in the sample is shifted using the system equation, such that: $\mathbf{x}_{t|t-1}^{(i)} = f(\mathbf{x}_{t-1|t-1}^{(i)}, \mathbf{e}_t^{m,(i)})$ which gives the predictive density: $\{\mathbf{x}_{t|t-1}^{(i)}, w_{t-1}^{(i)}\} \sim p(\mathbf{x}_t|\mathbf{Y}_{t-1})$.

Measurement Step:

- The new weights are determined by calculating the likelihood of observations given the predicted values. They are updated recursively by: $w_t^{(i)} = p(\mathbf{y}_t|\mathbf{x}_{t|t-1}^{(i)})$. As a result, higher weights are assigned to the particles that are closer to the observations.

The weights are normalized such that they sum to one. The particles can then be represented as: $\{\mathbf{x}_{t|t-1}^{(i)}, w_t^{(i)}\} \sim p(\mathbf{x}_t|\mathbf{Y}_t)$.

- $\{\mathbf{x}_{t|t-1}^{(i)}, w_t^{(i)}\}$ are resampled with replacement. The resampling results in an unweighted set of particles since $w_t^{(i)}$ = 1/M $\forall$ i = 1, ..., M and is then represented by: $\{\mathbf{x}_{t|t}^{(i)}\} \sim p(\mathbf{x}_t|\mathbf{Y}_t)$.

Note that, although there are a number of particle filtering methods, sequential importance resampling (SIR) will be the method of focus. It is less affected by *particle degeneracy* (than sequential importance sampling (SIS)) where the particles' weights collapse and could potentially lead to the entire sample set degenerating to a single particle. This is avoided with SIR since the particles are resampled with replacement with probabilities proportional to $\{w_t^{(i)}\}$. However, SIR still faces *sample impoverishment*, where the number of unique particles in the sample diminish over time. In extreme cases, this could results in a single repeated value for the entire set of particles. Sequential Markov chain Monte Carlo methods are used as an attempt to lessen the loss of diversity in the set of particles. This will be further discussed in Section 2.3.

To apply the SIR algorithm, the functions from the state space model, $f(.)$ and $h(.)$ need to be defined, as well as the statistics of the system and observation errors (i.e. the variances of $\mathbf{e}_t^m$ and $\mathbf{e}_t^o$). In addition, the prior distribution which the initial particles are sampled from is required. To demonstrate its application, an example using the AR(1) dataset is shown in Figure 2.2. The example is implemented using a particle sample size of 200. Moreover, the initial particles are sample from $\{\mathbf{x}_0^{(i)}\} \sim N(0,1)$ and the variances of $\mathbf{e}_t^m$ and $\mathbf{e}_t^o$ are set to $\sigma_m = 1$ and $\sigma_o = 1$. Note that these initial conditions of the particle's prior distribution and the values of $\sigma_m$ and $\sigma_o$ will be used for all SIR based examples in Section 2.2 and 2.3.

Comparing the results from the Kalman filter in Figure 2.1 and SIR in Figure 2.2, the variance of the residuals between the state estimates and the true states can be computed. The variance of the residuals from the Kalman filter is 0.6225961 while the variance of the residuals from SIR is 0.632611. In addition, a comparison of the filter variances is shown in Figure 2.3. The results from the Kalman filter (Figure 2.3 (a)) show that, $\mathbf{P}_t$ converges to a value of 0.6075891 while the filter variance from the particle filter (Figure 2.3 (b)) is noisy and centered around a mean of 0.5967233. Also, take note that in this example, the

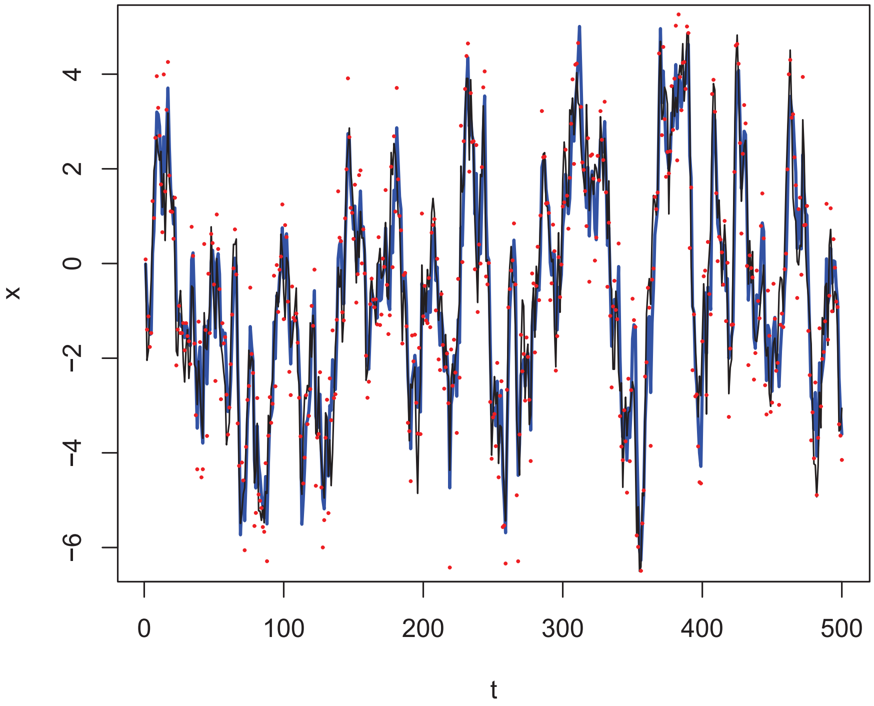


Figure 2.2: Estimated state of the AR(1) process. Shown are the observations (red dots), the true state (thin black line) and the particle filter estimate (thick blue line).

Kalman filter was initialized with a filter variance of $\mathbf{P}_0 = 0$, while the initial particles in the particle filter were drawn from the distribution, $\{\mathbf{x}_0^{(i)}\} \sim N(0,1)$. The Kalman filter is the ”exact” solution for state estimation in the linear and Gaussian case and the particle filter uses a sample based approximation. This comparison concludes that the particle filter is a useful tool in estimating the ”exact” state as evaluated from the Kalman filter. Although the particle filter is not ”exact”, it can handle nonlinear and non-Gaussian models which will be demonstrated in Chapter 3.

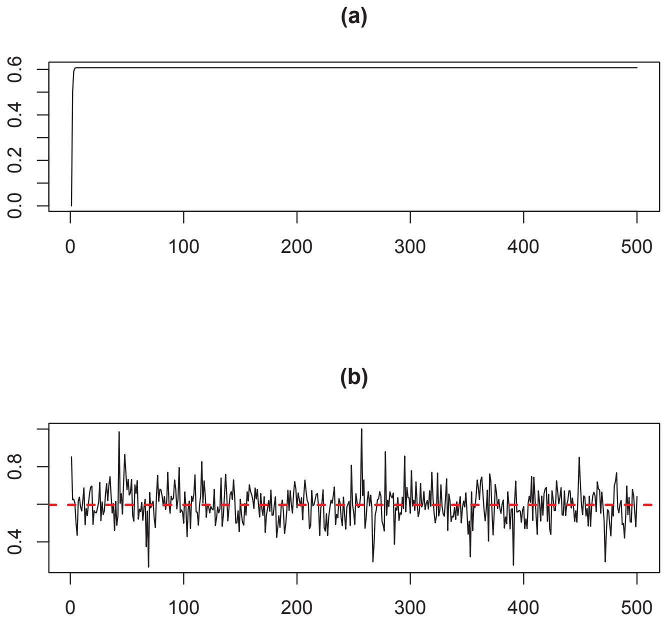


Figure 2.3: Plots of the filter variances computed by: (a) the Kalman filter and (b) the particle filter (black lines). Plot (b) also shows the mean of the filter variance (dashed red line).

### 2.2.2 *Maximum Likelihood Estimation*

Similar to the Kalman filter, maximum likelihood methods can also be used to estimate unknown static parameters. The maximum likelihood method estimates the parameters using the observed data. The likelihood function of the parameter vector, $\boldsymbol{\theta}$, is defined by:

$$L(\boldsymbol{\theta}|\mathbf{Y}_N) = p(\mathbf{Y}_N|\boldsymbol{\theta}) = \prod_{t=1}^{N} p(\mathbf{y}_t|\mathbf{Y}_{t-1}, \boldsymbol{\theta}), \tag{2.9}$$

where $p(\mathbf{y}_t|\mathbf{Y}_{t-1},\boldsymbol{\theta})$ is the conditional density of the observations, $\mathbf{y}_t$ given $\mathbf{Y}_{t-1}$. It can be calculated using equation (2.10)

$$p(\mathbf{y}_t|\mathbf{Y}_{t-1},\boldsymbol{\theta}) = \int p(\mathbf{y}_t|\mathbf{x}_t,\boldsymbol{\theta},\mathbf{Y}_{t-1})p(\mathbf{x}_t|\mathbf{Y}_{t-1},\boldsymbol{\theta})d\mathbf{x}_t. \tag{2.10}$$

Observe that $p(\mathbf{y}_t|\mathbf{x}_t,\boldsymbol{\theta},\mathbf{Y}_{t-1})$ = $p(\mathbf{y}_t|\mathbf{x}_t,\boldsymbol{\theta})$ can be evaluated since the observation density is known and is based on the observation noise, $\mathbf{e}^o_t$. Also, note that $p(\mathbf{x}_t|\mathbf{Y}_{t-1},\boldsymbol{\theta})$ is the predictive density and samples can be drawn from it (i.e. $\mathbf{x}^{(i)}_{t|t-1} \sim p(\mathbf{x}_t|\mathbf{Y}_{t-1},\boldsymbol{\theta})$). As a result, it can be approximated by:

$$\int p(\mathbf{y}_t|\mathbf{x}_t,\boldsymbol{\theta})p(\mathbf{x}_t|\mathbf{Y}_{t-1},\boldsymbol{\theta})d\mathbf{x}_t \approx \frac{1}{M}\sum_{i=1}^{M} p(\mathbf{y}_t|\mathbf{x}^{(i)}_{t|t-1},\boldsymbol{\theta}). \tag{2.11}$$

Therefore, substituting equation (2.11) and (2.10) into (2.9), gives the likelihood function,

$$L(\boldsymbol{\theta}|\mathbf{Y}_N) \approx \prod_{t=1}^{N}\frac{1}{M}\sum_{i=1}^{M} p(\mathbf{y}_t|\mathbf{x}^{(i)}_{t|t-1},\boldsymbol{\theta}), \tag{2.12}$$

and the log likelihood function,

$$logL(\boldsymbol{\theta}|\mathbf{Y}_N) \approx \sum_{t=1}^{N} log(\sum_{i=1}^{M} p(\mathbf{y}_t|\mathbf{x}^{(i)}_{t|t-1},\boldsymbol{\theta})) - NlogM. \tag{2.13}$$

Equation (2.13) is evaluated using the particle filter. As an example, consider the simple state space model from equations (2.3) and (2.4) and the dataset from the AR(1) process. The results for profiling the likelihood with respect to the parameter, $a$, using a particle sample size of 200, are shown in Table 2.2. As a reminder, the true value of the parameter is $a$ = 0.95. In this example, the value corresponding to the maximum log-likelihood is $a$ = 0.92. Recall Table 2.1 and note that this estimate of $a$ = 0.92 is close to the MLE evaluated by the Kalman filter. As an additional note, when comparing the results from Tables 2.1 and 2.2, the log-likelihood values are not identical since the results using SIR are not ”exact”. They are approximate and depend on the number of particles used. As the sample size, M $\rightarrow \infty$, the set of particles will provide an ”exact” representation of the densities of interest, and as a result, the ”exact” estimates.

Table 2.2: The results of likelihood-based parameter estimation using SIR.

| Value of $a$ | Log-likelihood |
|---|---|
| 0.85 | -950.2724 |
| 0.86 | -950.8097 |
| 0.87 | -948.3962 |
| 0.88 | -946.8039 |
| 0.89 | -947.3001 |
| 0.90 | -947.4689 |
| 0.91 | -947.1498 |
| 0.92 | -943.8871 |
| 0.93 | -946.5560 |
| 0.94 | -947.6808 |
| 0.95 | -945.1897 |
| 0.96 | -944.6773 |
| 0.97 | -949.4245 |
| 0.98 | -951.1951 |
| 0.99 | -952.9748 |
| 1.00 | -957.5125 |

### *2.2.3 State Augmentation*

Another method of parameter estimation for state space models utilizes state augmentation. This method approximates the joint filter density between the state and unknown parameters. Parameter estimation via state augmentation allows the parameter vector, $\boldsymbol{\theta}_t$, to evolve with time, where $\boldsymbol{\theta}_t$ represents the parameter estimate specific to time t. However, this does not imply that the parameters are truly time dependent, as there are ways to make the parameter values converge to static values (*Kitagawa*, 1998; *Ionides et al.*, 2006). In this approach, the state vector is augmented to include the vector of unknown parameters, such that,

$$\tilde{\mathbf{x}}_t = \begin{bmatrix} \mathbf{x}_t \\ \boldsymbol{\theta}_t \end{bmatrix}.$$

For example, if state augmentation is applied to the state space model from equations (2.3) and (2.4) the system and observation equations (respectively) become,

$$\begin{bmatrix} x_t \\ a_t \end{bmatrix} = \begin{bmatrix} a_{t-1}x_{t-1} \\ a_{t-1} \end{bmatrix} + \begin{bmatrix} e_t^m \\ e_t^\theta \end{bmatrix}$$

$$y_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_t \\ a_t \end{bmatrix} + e_t^o.$$

Here, it can be noted that state augmentation has turned this former linear problem into a nonlinear one. In this example, the dynamical parameter, $a_t$, evolves with time such that, $a_t = a_{t-1} + e_t^\theta$, where $e_t^\theta$ is some additional white noise. Note, the statistics of $e_t^\theta$ need to be specified (e.g. if $e_t^\theta \sim N(0, \sigma_\theta)$ then $\sigma_\theta$ needs to be specified). The augmented states can also be rewritten in state space form:

$$\tilde{\mathbf{x}}_t = f(\tilde{\mathbf{x}}_{t-1}) + \tilde{\mathbf{e}}_t \tag{2.14}$$

$$y_t = g(\tilde{\mathbf{x}}_t) + e_t^o \tag{2.15}$$

where,

$$\tilde{\mathbf{x}}_t = \begin{bmatrix} x_t \\ a_t \end{bmatrix} \qquad \text{and} \qquad \tilde{\mathbf{e}}_t = \begin{bmatrix} e_t^m \\ e_t^\theta \end{bmatrix}.$$

The SIR algorithm can be used to obtain the augmented state vector, $\tilde{\mathbf{x}}_t$, which yields the joint filter density that includes the parameters: $\{\tilde{\mathbf{x}}_{t|t}^{(i)}\} = \{\mathbf{x}_{t|t}^{(i)}, \boldsymbol{\theta}_t^{(i)}\} \sim p(\mathbf{x}_t, \boldsymbol{\theta}_t | \mathbf{Y}_t)$.

Figure 2.4 shows the results from an example of parameter estimation via state augmentation. The dataset from the AR(1) process is used. This example also uses a particle sample of size 200 and a parameter error term of $e_t^\theta \sim N(0, \sigma_\theta = 0.05)$. Figure 2.4 shows the trace of the parameter, $a_t$, as it evolves over time. As a result, the trace of $a$ appears to move about a mean of $a = 0.822065$. Comparing these results with the results from maximum likelihood estimation, state augmentation does not appear to produce a result as accurate as the ML method. However, using state augmentation is advantageous because it has a much faster computation time than maximum likelihood estimation and since it is in standard state space form, it is readily solvable by standard filtering algorithms.

As seen in the previous example, effectively estimating static parameters via state augmentation may be difficult. Aside from ML methods, there are other alternative techniques that are better suited to estimate time-constant parameters. For example, multiple iterative filtering is a method mainly used to estimate static parameters. This method uses state augmentation with an additional concept of decreasing the variance of the parameter's error term over multiple iterations of the state augmented particle filtering

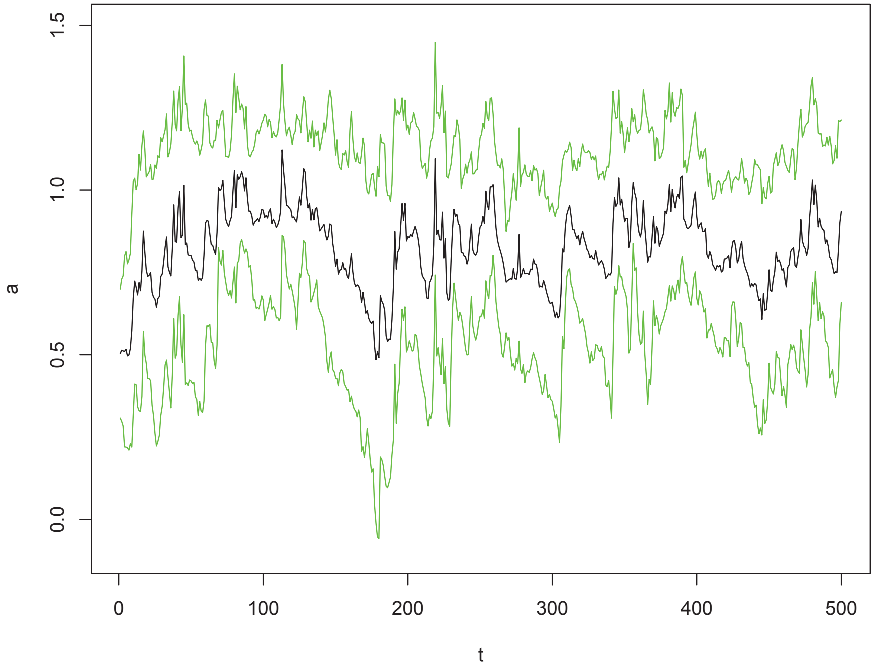


Figure 2.4: Trace of the single parameter $a$ in an AR(1) process; estimated via state augmentation. Shown are the 95% confidence bounds (green lines) and the parameter $a$ (black line). The true value of a = 0.95.

algorithm (*Ionides et al.*, 2006). It is investigated below.

### *2.2.4 Multiple Iterative Filtering*

The previous section provides details of parameter estimation using augmented states where the set of particles advance through the filter only once. Multiple iterative filtering is a method that uses the time-varying parameter technique to produce optimal estimates for time-constant parameters. With additional iterations of the state augmented particle filter, the maximum likelihood of the parameters can be estimated. This can be achieved by reducing the variance of the error term in the parameter process, $\mathrm{e}_t^\theta$, at each iteration. As the variance of $\mathrm{e}_t^\theta$ approaches zero, the parameters will eventually become fixed at a constant value that approximately represents their corresponding maximum likelihood estimate (*Ionides et al.*, 2006).

To estimate the parameters of an augmented state space model, a multiple iterated filtering algorithm is provided below.

**Multiple Iterative Filtering (MIF) Algorithm**

```
Suppose, $\mathbf{e}_t^\theta \sim N(0, \sigma_\theta^2)$

  • Select starting values for $\hat{\theta}_{k=0}$, $\sigma_{\theta,k=0}$, the number of iterations, K, and the discount
    factor, $\alpha$ $(0 < \alpha < 1)$

(Remark: K and $\alpha$ are chosen such that $\sigma_\theta^2$ will be small enough that the parameter, $\hat{\theta}_k$,
will be fixed at a fairly constant value by the end of the iterative algorithm.)
For iterations, k = 1, 2, ... , K:

  • Let $\sigma_{\theta,k} = \alpha\sigma_{\theta,k-1}$

    For observations, t = 1, 2, ... , N:

      • Run the state augmented particle filter to obtain:

          $\hat{\theta}_{t,k} = \hat{\theta}_t(\hat{\theta}_{k-1}, \sigma_{\theta,k}) = E[\theta_t|\mathbf{Y}_t]$
          $V_{t,k} = V_t(\hat{\theta}_{k-1}, \sigma_{\theta,k}) = Var(\theta_t|\mathbf{Y}_{t-1})$

    End For (t)

  • Set $\hat{\theta}_k = \hat{\theta}_{k-1} + V_{1,k} \sum_{t=1}^{N} V_{t,k}^{-1}(\hat{\theta}_{t,k} - \hat{\theta}_{t-1,k})$
```

Table 2.3: The results of parameter estimation using multiple iterated filtering with an augmented state space model.

| Iteration | Estimate of a |
|---|---|
| 1 | 0.8111517 |
| 2 | 0.8651654 |
| 3 | 0.8726499 |
| 4 | 0.8952580 |
| 5 | 0.9014250 |
| 6 | 0.8918252 |
| 7 | 0.9100771 |
| 8 | 0.8743331 |
| 9 | 0.8732276 |

```
        where $\hat{\theta}_{t=0,k} = \hat{\theta}_{k-1}$
End For (k)
Use $\hat{\theta}_{k=K}$ as the final estimate.
```

Figure 2.5 shows the results of parameter estimation using multiple iterated filtering with an augmented state space model. The dataset generated from the AR(1) process and a particle sample size of 500 is used. In addition, the starting values of $\hat{\theta}_{k=0}$ = 0, $\sigma_{\theta,k=0}$ = 0.1, K = 9 and $\alpha$ = 0.6 were used. Table 2.3 shows the parameter estimates to each corresponding iteration from Figure 2.5. The results show that MIF yields a final parameter estimate of $a$ = 0.8732276. This is a better estimate than the estimate evaluated from only a single run of the state augmented particle filter. However, MIF requires greater computational effort, as it runs the augmented state particle filter K iterations before a maximized likelihood estimate is reached. In addition, MIF does not quite reach the same estimated parameter value as the MLE from Section 2.2.2. Taking the average of multiple runs of the MIF algorithm could improve results. An increase in the number of particles or an increase in the number of iterations could also improve results but would be more computationally costly.

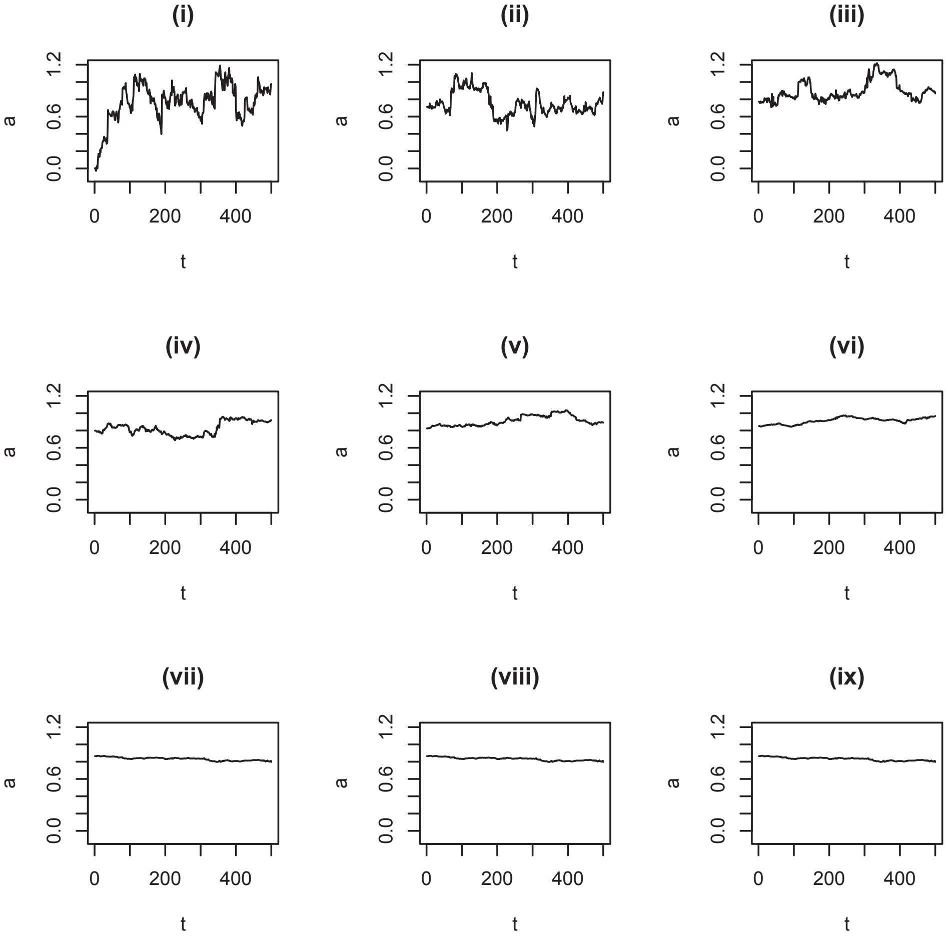


Figure 2.5: Models of the single parameter $a$ in an AR(1) process; estimated via multiple iterative state augmented particle filtering. Shown are the traces of the parameter $a$ generated in each of the 9 iterations. The true value of $a = 0.95$.

## 2.3 Markov Chain Monte Carlo (MCMC)

MCMC methods generate samples from a target distribution which is known up to a proportionality constant. It begins with a randomly selected initial value. As the iterations of the MCMC step progress, new values are accepted/rejected and generally, are only dependent on the previously sampled value. As a result, the subsequent samples are independent of the initial value. After a large number of iterations, the results become a realization of a Markov chain with a stationary distribution. With regard to particle filtering, the MCMC technique that will be focused on is the Metropolis-Hastings (MH) algorithm.

### *2.3.1 Sequential MCMC*

A common issue when using SIR alone, is the loss of diversity in the set of particles. This occurs when after resampling, only a few unique values in the set, $\{\mathbf{x}_{t|t}^{(i)}\}$, remain. To assess this, it is useful to measure the effective number of particles, $P_{\text{effective}}$, which can be calculated by dividing the number of distinct particles by the total number of particles in $\{\mathbf{x}_{t|t}^{(i)}\}$. To alleviate a low $P_{\text{effective}}$, a popular solution is to add an MCMC update step after resampling in the SIR algorithm (*Carpenter et al.*, 1999; *MacEachern et al.*, 1999; *Gilks and Berzuini*, 2001). Below is a particle filter Metropolis-Hastings (PF-MH) algorithm. It is essentially the SIR algorithm with an embedded MH step. At time t, after resampling (and if $P_{\text{effective}}$ is too low, usually below $30\%$, which is the value used below), each resampled particle will be compared to a newly generated (predictive) particle from the same system model. The likelihood ratio of the newly generated particle over the resampled particle will be calculated. At that point, the new particle will be accepted over the resampled particle with an acceptance probability equal to the likelihood ratio (or equal to 1 if the likelihood ratio is greater than 1). This technique is used for state estimation and reduces the loss of diversity (in the set of particles) compared to using SIR alone. However, it can also be used for parameter estimation with state augmentation and multiple iterative filtering. As an additional note, the same notation and implementation of the SIR algorithm from Section 2.2.1 will be used for the remainder of this thesis.

**Particle Filter with Metropolis-Hastings (PF-MH) Update Algorithm**

Note: before beginning the PF-MH algorithm, the number of particles, M, the state space model (system and observations equations) and observations, $\mathbf{Y}_t$ (where t = 1, ..., N), are required.

```
For observation, t = 0:
    • Sample M particles, $\{\mathbf{x}_t^{(i)}\}$ from a prior distribution
For observation, t = 1, ... , N:
    • Using the state space model and observations $\mathbf{y}_t$, run the SIR algorithm (from Section 2.2.1) to obtain $\{\mathbf{x}_{t|t}^{(i)}\}$
    • Initialize i = 1; randomly sample a single particle from the resampled ensemble (i.e. $\{\mathbf{x}_{t|t}^{(i)}\}$)
    • Compute $P_{\text{effective}}$
    If $P_{\text{effective}} \leq 0.3$:
        The MH update step:
        For particle, i = 2, ... , M:
            • Sample a random (trial) particle from $\{\mathbf{x}_{t-1|t-1}^{(i)}\}$
            • Pass the trial particle through the system model to get a sample from $\{\mathbf{x}_{t|t-1}^{(i)}\}$
              and denote the trial particle by $\mathbf{x}_{t|t-1}^{(i)*}$
            • Compute the acceptance probability:
                  $LR = \frac{p(\mathbf{y}_t|\mathbf{x}_{t|t-1}^{(i)*})}{p(\mathbf{y}_t|\mathbf{x}_{t|t}^{(i-1)})}$
            • Draw $u \sim U(0,1)$
              If $min(1, LR) \geq u$:
                  • $\mathbf{x}_{t|t}^{(i)} = \mathbf{x}_{t|t-1}^{(i)*}$
              Else:
                • $\mathbf{x}_{t|t}^{(i)} = \mathbf{x}_{t|t}^{(i-1)}$
        End For (i)
    Else, skip MH update step
End For (t).
```

Figure 2.6 shows the estimated states of the AR(1) process using the PF-MH algorithm and a particle sample size of 200. In addition, for this example, the MH update step was implemented at every iteration and time step (regardless of whether $P_{\text{effective}}$ was below 0.3). It is evident from Figure 2.6 that the PF-MH procedure is able to see through the noisy observations and closely recover the true system states. Further, Figure 2.7 shows the plots of $P_{\text{effective}}$ before and after the MH update step and their respective means. The mean of $P_{\text{effective}}$ before the MH step is 0.50032 while the mean of $P_{\text{effective}}$ after the MH step is 0.54626. These results demonstrate that the PF-MH algorithm is successful at improving the statistical diversity. However, there is a trade-off; a main criticism of such MCMC algorithms is that they are computationally expensive.

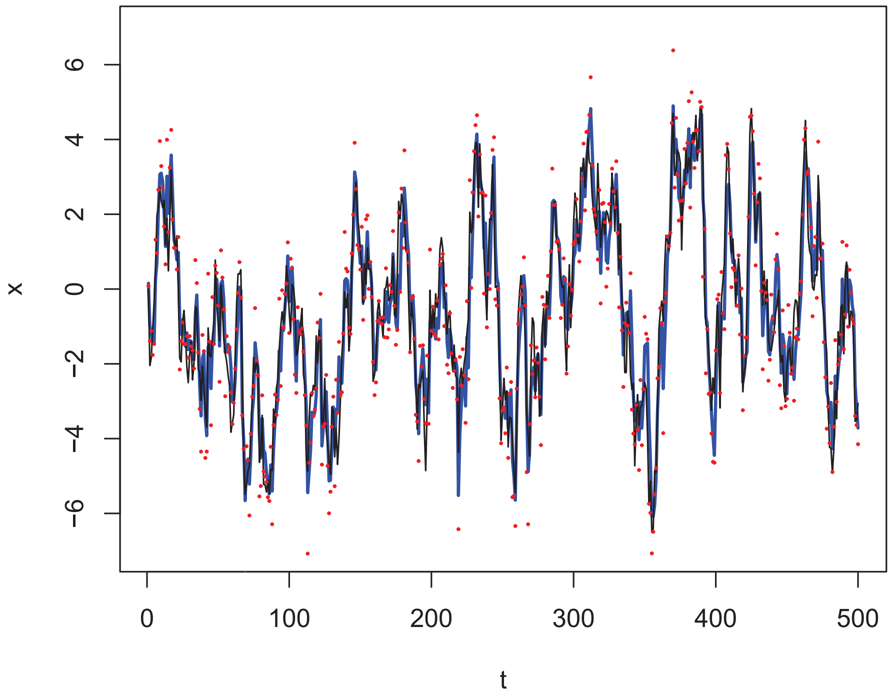


Figure 2.6: Estimated state of the AR(1) process. Shown are the observations (red dots), the true state (thin black line) and the PF-MH estimated state (thick blue line).

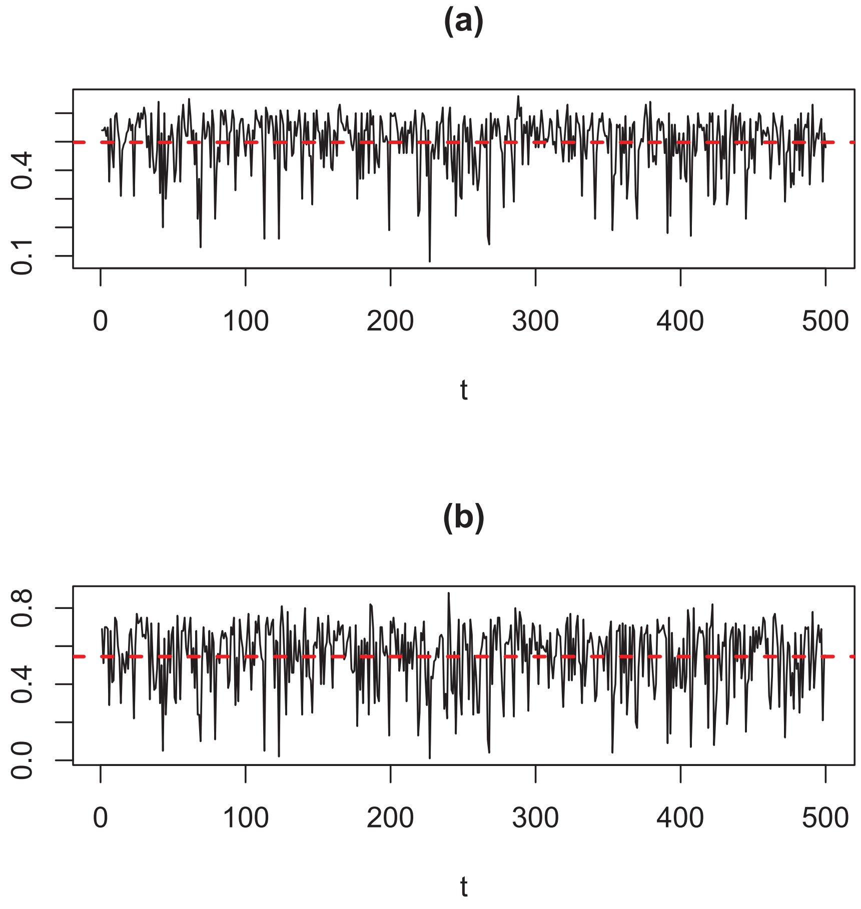


Figure 2.7: The plots of $P_{\text{effective}}$ (black line) and the mean of $P_{\text{effective}}$ (red dashed line): (a) before the MH update step (and after resampling) with a mean of 0.50032 and (b) after the MH update step with a mean of 0.54626.

### *2.3.2 Particle MCMC*

Particle Markov chain Monte Carlo algorithms (PMCMC) are typically used to solve advanced problems in parameter estimation where standard approaches tend to fall short (*Andrieu et al.*, 2010). For example, in *Jones et al.* (2010), MCMC algorithms were assessed in a complex scenario using a nonlinear stochastic-dynamic phytoplankton-zooplankton model. In particular, of interest, is the MH-PF-MH method that was used. It entails a PF-MH subroutine that is embedded within a MH MCMC algorithm. The technique allows for joint state and parameter estimation.

The objective of PMCMC algorithms is to return a sample set that approximately follows the target distribution of the states and parameters of interest. This is done by using the output of a SIR algorithm that targets $\{\mathbf{x}_{t|t}^{(i)}\} \sim p(\mathbf{x}_t|\mathbf{Y}_t)$. The set of particles distributed by $p(\mathbf{x}_t|\mathbf{Y}_t)$ will be used as the proposal distribution in the PMCMC update. The ultimate focus of this thesis is to use PMCMC algorithms to estimate the dynamical parameters in state space models.

To reiterate, sequential MCMC is a particle filter with an embedded MCMC update step. In contrast, the particle MCMC algorithm is an MCMC algorithm with an embedded particle filter at each iteration of the chain. Specifically, the PMCMC algorithm is a series of K iterations, where each iteration uses a particle filter subroutine to evaluate a proposal distribution that is based on newly generated parameter values, denoted by, $\boldsymbol{\theta}^*$. This proposal distribution is then used to calculate the likelihood of $\boldsymbol{\theta}^*$ given all observations, $\mathbf{Y}_N$. Lastly, $\boldsymbol{\theta}^*$ will be accepted by the MCMC step with a probability equal to the likelihood ratio of the new versus the old (or previously accepted) parameter estimates. Below is an implementation of particle MCMC using the Metropolis-Hastings algorithm with an embedded PF-MH subroutine. For further details, see *Jones et al.* (2010).

**MH-PF-MH Algorithm**

Note: the number of iterations, K and the proposal distribution, $\mathbf{Q}$ used to generate the parameters $\boldsymbol{\theta}_k$ need to be determined.

For iteration k = 1:

- Arbitrarily set $\boldsymbol{\theta}_{k=1}$
- Run the PF-MH algorithm to obtain $\{\mathbf{x}_{t|t}^{(i)}\}$ for t = 1, ... , N

```
    • Evaluate $L(\boldsymbol{\theta}_k|\mathbf{Y}_N) \approx \prod_{t=1}^{N} \frac{1}{M} \sum_{i=1}^{M} p(\mathbf{y}_t|\mathbf{x}_{t|t}^{(i)})$

For iteration, k = 2, ... , K:

    • Trial parameters are generated from: $\boldsymbol{\theta}_k^* = \boldsymbol{\theta}_{k-1} + \mathbf{Q}$

    • Run the PF-MH algorithm to evaluate: $L(\boldsymbol{\theta}_k^*|\mathbf{Y}_N)$

The MH update step:

    • Compute the acceptance probability:

        $LR = \frac{L(\boldsymbol{\theta}_k^*|\mathbf{Y}_N)}{L(\boldsymbol{\theta}_{k-1}|\mathbf{Y}_N)}$

    • Draw $u \sim U(0,1)$

    If $min(1, LR) \geq u$:

        • $\boldsymbol{\theta}_k = \boldsymbol{\theta}_k^*$

    Else:

        • $\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1}$

End For (k).
```

Figure 2.8 shows the estimated parameter, $a$, of the AR(1) process using the MH-PF-MH algorithm with a particle sample size of 200, an initial parameter value of $a$ = 0.6, and K = 10,000 iterations. In addition, the new values of $\boldsymbol{\theta}$ are generated from $\boldsymbol{\theta}_k^* = \boldsymbol{\theta}_{k-1} + \mathbf{Q}$ where $\mathbf{Q} \sim N(0, 0.02)$. Also note, in this example, to save some computational time, the MH step embedded in the SIR algorithm is only implemented when $P_{\text{effective}}$ falls below 0.3. In Figure 2.8, the black line represents the trace of $a$ and the red dashed line represents the true value of $a = 0.95$.

The acceptance rate for this example is computed and yields a value of 4.46$\%$. The acceptance rate indicates that the trial parameters generated from, $\boldsymbol{\theta}_k^* = \boldsymbol{\theta}_{k-1} + \mathbf{Q}$, are accepted 4.46$\%$ of the time out of a total of K = 10,000 iterations. This is a relatively low acceptance rate and means that the chain is not moving much. To alleviate a low acceptance rate, a simple solution is to inflate the observation error variance in equation (2.4).

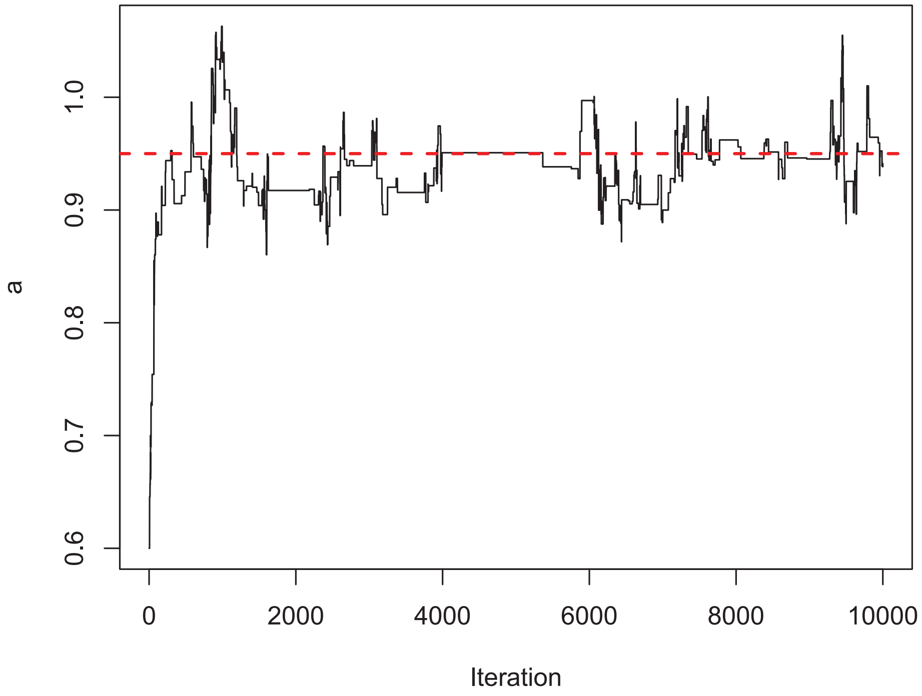


Figure 2.8: Plot of the parameter $a$ in an AR(1) process; estimated using the MH-PF-MH update algorithm. Shown are the estimated values of the parameter $a$ (black line) and the true value of $a$ = 0.95 (red dashed line).

To remove the bias that results from the particles chosen at the beginning of the MH iterations, the first k iterations are discarded. The first k iterations are called the *burn-in stage*. The length of the burn-in period can be estimated by plotting the autocorrelation function (ACF) of the estimated parameter vector, $\boldsymbol{\theta}$ over the k iterations.

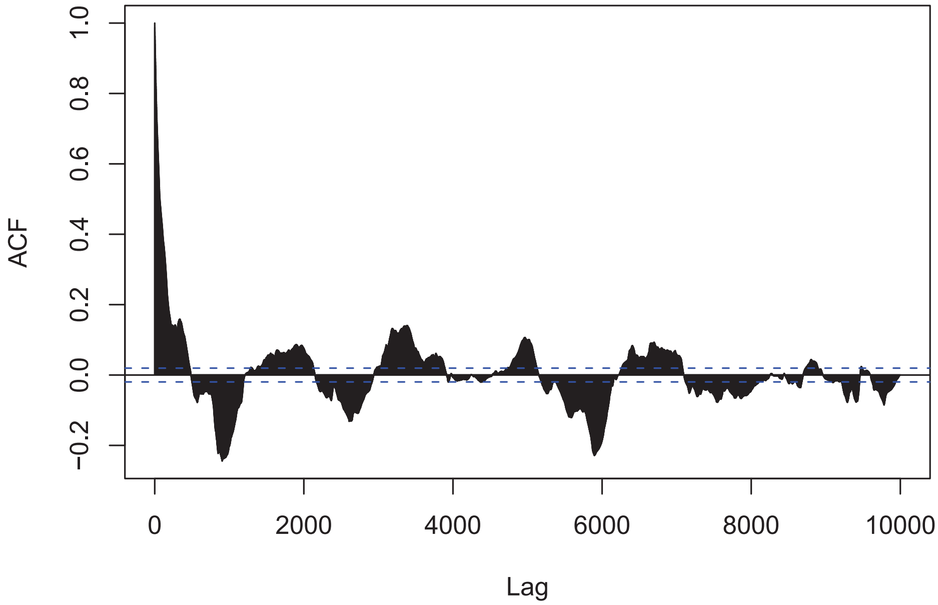


Figure 2.9: Plot of the ACF for the MH-PF-MH estimated parameter $a$ in an AR$(1)$ process. The length of the burn-in period is approximately k = 485 iterations long.

Figure 2.9 is the ACF plot corresponding to the results in Figure 2.8. The results show that the decorrelation time of the chain, and hence the burn-in period is approximately k = 485 iterations long. After discarding the burn-in period, the results from Figure 2.8 show that $a \sim N(0.9406947, 0.0264019)$. As a reminder, the true value of the static parameter is $a = 0.95$. With respect to the previously estimated values of $a$, the MLE from Table 2.2 is $a = 0.92$, the estimate via state augmented particle filtering from Figure 2.4 is $a = 0.822065$ and the estimate using MIF from Figure 2.5 is $a = 0.8732276$. Comparing the results from the ML method, state augmentation and MIF, the MH-PF-MH algorithm is more robust and produces a more accurate estimate of the parameter, $a$. However, as mentioned previously, it is computationally expensive. For every iteration, the MH-PF-MH algorithm requires the SIR to generate M number of particles for each time step (from t = 1, ... , N). This example shows that the PMCMC algorithm is useful but time consuming in estimating the static parameter, $a$, from the AR(1) process. Yet, the PMCMC algorithm is able to make suitable estimations for static parameters in a wide range of models. In Chapter 3, the PMCMC algorithm along with all the state and parameter estimation techniques discussed in Section 2.2 and 2.3 will be applied using a nonlinear state space model and with no assumptions of Gaussian error.

# CHAPTER 3

# MODELING POPULATIONS

This chapter discusses the lynx-hare dataset and the predator-prey model introduced by *Lotka* (1920) and *Volterra* (1926). In this thesis, the predator-prey model is used to define the evolution of the animals' populations and only considers the observations of the lynx-hare populations with no additional environmental or external factors. Note that the predator-prey model is a nonlinear population model with no assumption of Gaussian noise; therefore, Chapter 3 will only consider the estimation techniques that are suitable for such systems as discussed in Section 2.2 and 2.3 (i.e. particle filtering methods as opposed to Kalman filtering methods).

## 3.1 The Predator-Prey Model

The Lotka-Volterra predator-prey model consists of two equations, first, the growth equation for the prey (hare) population:

$$\frac{dN_1}{dt} = r_1 N_1 - b_1 N_1 N_2 \tag{3.1}$$

and second, the growth equation for the predator (lynx) population:

$$\frac{dN_2}{dt} = -r_2 N_2 + b_2 N_2 N_1 \tag{3.2}$$

where

$N_1$: population of the prey

$N_2$: population of the predator

$r_1$: prey growth rate in absence of the predators

$r_2$: decay rate of the predator in absence of the prey

$b_1$: predation rate by the predators on the prey

$b_2$: growth rate of the predators (dependent on the prey population).

To explain the Lotka-Volterra equations, there are some assumptions that should be noted. Firstly, in the absence of predators, the prey population is assumed to grow exponentially according to $\frac{dN_1}{dt} = r_1 N_1, r_1 \geq 0$. Likewise, in the absence of the prey, the predator population decreases at a rate given by $\frac{dN_2}{dt} = -r_2 N_2, r_2 \geq 0$. However, when both the predator and prey are present, it is assumed that there is a decay rate in the prey population and a growth rate in the predator population due to the consumption of the prey by the predators, at a rate proportional to the frequency of predator-prey interactions. The frequency of interactions is assumed to be proportional to the product of the prey population and the predator population (i.e. $N_1 N_2$), rendering the process a nonlinear one. With appropriately chosen coefficients the solutions allow for periodic predator and prey co-oscillations.

Typically, the Lotka-Volterra equations fail to fully capture the evolution of the predator and prey populations. Noise is often added to the model as a simple improvement to make the output of the dynamics more realistic. However, prior to adding the noise, the continuous model is simplified by discretizing the Lotka-Volterra equations. The discretized equations are:
Predator (lynx):

$$N_{2,t} = N_{2,t-1} + \Delta(-r_2 N_{2,t-1} + b_2 N_{2,t-1} N_{1,t-1}) \tag{3.3}$$

Prey (hare):

$$N_{1,t} = N_{1,t-1} + \Delta(r_1 N_{1,t-1} - b_1 N_{1,t-1} N_{2,t-1}) \tag{3.4}$$

where $\Delta$ is a time increment (typically small). The examples in this thesis assume that $\Delta = 0.1$. These discretized equations are used as the system equations in the general state space model. Referring back to equation (1.1) from Chapter 1, the discretized equations (3.3) and (3.4) can be rewritten as $\mathbf{x}_t = f(\mathbf{x}_{t-1}, \boldsymbol{\theta}) + \mathbf{e}_t^m$, where $\mathbf{x}_t$, is a column vector consisting of the states, $N_{1,t}$ and $N_{2,t}$ at time t and $\boldsymbol{\theta}$, consists of the static parameters $r_1, b_1, r_2$ and $b_2$.

For a better understanding of the static parameters and their effects on the oscillations of the predator-prey model, a simple sensitivity analysis of the parameters is implemented. The approach involves varying the values of one parameter at a time, while keeping the remaining parameters fixed. Table 3.1 shows the sensitivity analysis results by presenting the percentage change in the populations of the predator and prey from the base case (with no noise) where each parameter is varied as a percentage of the base case. In this example, the base case consists of the parameter values: $r_1 = 0.65, b_1 = 0.023, r_2 = 0.65$ and $b_2 = 0.014$. The results from Table 3.1 and Figures A.1, A.2, A.3, and A.4 in Appendix A, indicate that $r_1$ is the most sensitive to small changes. However, $b_1$ and $b_2$ are the most sensitive to larger negative changes. The parameters, $b_1$ and $b_2$ correspond to the interactions between the prey and predator populations. It is also evident that the percentage change in the populations is not proportional to the percentage change in the parameters. This is due to the nonlinearity of the model. When the populations of the predator and prey are plotted, the changes are seen in the amplitude and phasing of the cycles. In general, the parameters in the predator equation ($r_2$ and $b_2$) effect the prey population to a higher degree than the predator population. Likewise, the parameters in the prey equation ($r_1$ and $b_1$) effect the predator population more than in the prey population.

## 3.2 Synthetic Data for a Predator-Prey Model

All examples in this section will use the same synthetic dataset. The dataset's 200 observations are generated from the predator-prey model plus additional noise. Referring to equations (3.3) and (3.4), the synthetic observations are based on the parameter values, $r_1 = 0.65, b_1 = 0.023, r_2 = 0.65, b_2 = 0.014$ and $\Delta = 0.1$. Furthermore, the additive noise is a bi-variate normal vector, where each noise variable is distributed by, $e_t^{N_1} \sim N(0, 1)$ (for the prey) and $\mathbf{e}_t^{N_2} \sim N(0, 1)$ (for the predator) and are independent of one another. Figure 3.1 shows the time series plot of the idealized synthetic dataset.

Table 3.1: The percentage change in the population of the predator and prey given the percentage change in parameter value. Each column represents a parameter that is changing and shows the corresponding percentage change in the state of the predator/prey.

| | Mean % Change in States | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Predator | | | | Prey | | | |
| % Change in (base case) Parameter | $r_1$ | $b_1$ | $r_2$ | $b_2$ | $r_1$ | $b_1$ | $r_2$ | $b_2$ |
| -90 | -70.61 | 1140.04 | 41.50 | 89.72 | 70.23 | 24.92 | -83.091 | 1394.43 |
| -80 | -78.35 | 578.41 | 44.23 | 22.54 | 69.67 | 125.66 | -69.154 | 452.79 |
| -70 | -63.97 | 390.11 | 78.14 | 40.06 | 33.71 | 44.88 | -45.211 | 726.36 |
| -60 | -40.42 | 362.53 | 39.51 | 64.25 | 51.33 | 67.89 | -32.829 | 288.50 |
| -50 | -22.74 | 153.88 | 52.27 | 106.53 | 60.32 | 50.62 | -24.931 | 260.82 |
| -40 | -20.51 | 62.90 | 68.54 | 57.30 | 49.90 | 52.06 | -0.299 | 194.88 |
| -30 | 25.39 | 32.24 | 53.88 | 17.18 | 62.99 | 57.14 | 22.967 | 146.26 |
| -20 | 31.42 | 17.36 | 31.15 | 2.18 | 86.65 | 8.46 | 33.994 | 104.65 |
| -10 | 7.08 | 6.54 | 12.54 | -1.77 | 53.92 | -1.52 | 11.158 | 20.65 |
| 0 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.000 | 0.00 |
| 10 | 27.23 | -5.16 | -3.36 | 10.09 | 3.46 | 3.40 | 4.917 | -3.28 |
| 20 | 86.12 | -10.27 | -1.47 | 24.58 | 19.95 | 6.66 | 14.686 | -3.03 |
| 30 | 146.86 | -15.33 | -1.56 | 38.81 | 36.75 | 9.65 | 23.618 | -2.29 |
| 40 | 194.79 | -19.94 | -4.77 | 50.61 | 48.84 | 12.94 | 29.938 | -1.99 |
| 50 | 221.63 | -23.73 | -4.39 | 59.85 | 53.21 | 17.57 | 47.886 | -2.46 |
| 60 | 210.78 | -26.49 | -11.78 | 67.16 | 49.37 | 25.07 | 221.928 | -3.79 |
| 70 | 191.33 | -28.23 | -26.62 | 73.24 | 42.66 | 37.48 | 91.121 | -5.98 |
| 80 | 220.69 | -29.16 | -28.80 | 78.58 | 129.24 | 56.85 | 31.665 | -8.96 |
| 90 | 206.89 | -29.59 | -22.98 | 83.42 | 42.09 | 82.73 | 56.120 | -12.66 |
| 100 | 203.29 | -29.52 | 1.57 | 87.75 | 17.67 | 105.98 | 185.178 | -16.91 |

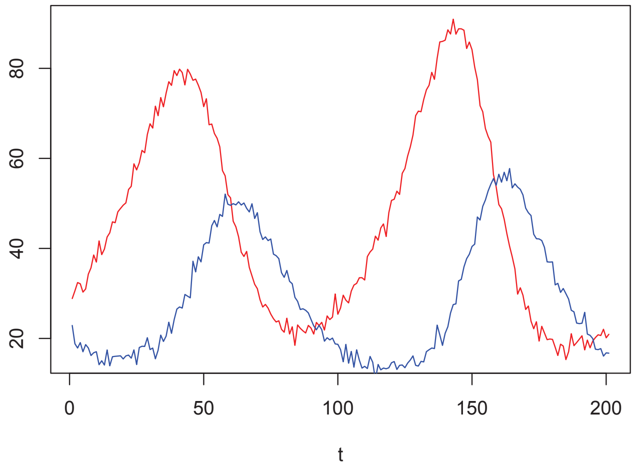


Figure 3.1: Time series plot of the simulated dataset from the predator-prey model with additive noise and parameter values: $r_1$ = 0.65, $b_1$ = 0.023, $r_2$= 0.65 and $b_2$ = 0.014.

### *3.2.1 State Estimation*

The SIR and PF-MH algorithm from Chapter 2 are applied to the synthetic predator-prey dataset. Figure 3.2 shows the results for state estimation. In this example, both the SIR and PF-MH algorithm use a particle sample size of 200 and assumes a system error of $\mathbf{e}_t^m \sim N(\mu_m, \Sigma_m)$, where $\mu_m$ is a (2x1) vector of zeros and $\Sigma_m = I$, a (2x2) identity matrix and an observation error of $\mathbf{e}_t^o \sim N(\mu_o, \Sigma_o)$, where $\mu_o$ is a (2x1) vector of zeros and $\Sigma_o = I$, the (2x2) identity matrix. In addition, the initial 200 particles (for i = 1, ... , 200) are sampled from the prior distribution:

$$\{\mathbf{x}_{t=0}^{(i)}\} = \begin{pmatrix} N_{1,t=0} \sim N(15,1) \\ N_{2,t=0} \sim N(15,1) \end{pmatrix}.$$

Note that this prior distribution for the initial particle sample set will be used for the remainder of this section. The results of the SIR algorithm and the PF-MH algorithm from Figure 3.2 are compared by computing the mean of the squared errors (MSE) between the estimated state and the true state. When examining the synthetic predator dataset, the MSE is 4.7704 for the SIR algorithm and 3.0765 for the PF-MH algorithm. In regards to the synthetic prey dataset, the MSE is 5.9295 for the SIR algorithm and 2.7851 for the PF-MH algorithm. These figures indicate that the PF-MH algorithm is more accurate in terms of reducing the difference between the estimates and the true states since the incorporation of the MCMC step has improved the sample diversity. In addition, scatter plots of the true values versus the estimated states of the predator and prey populations using both algorithms are shown in Figure 3.3 and a regression analysis is executed which yields correlation coefficients of 0.993369 and 0.995891 for the SIR and PF-MH algorithms (respectively). This produces the same conclusion; the PF-MH algorithm recovers state estimates that are slightly closer to the true values of the populations than the SIR algorithm.

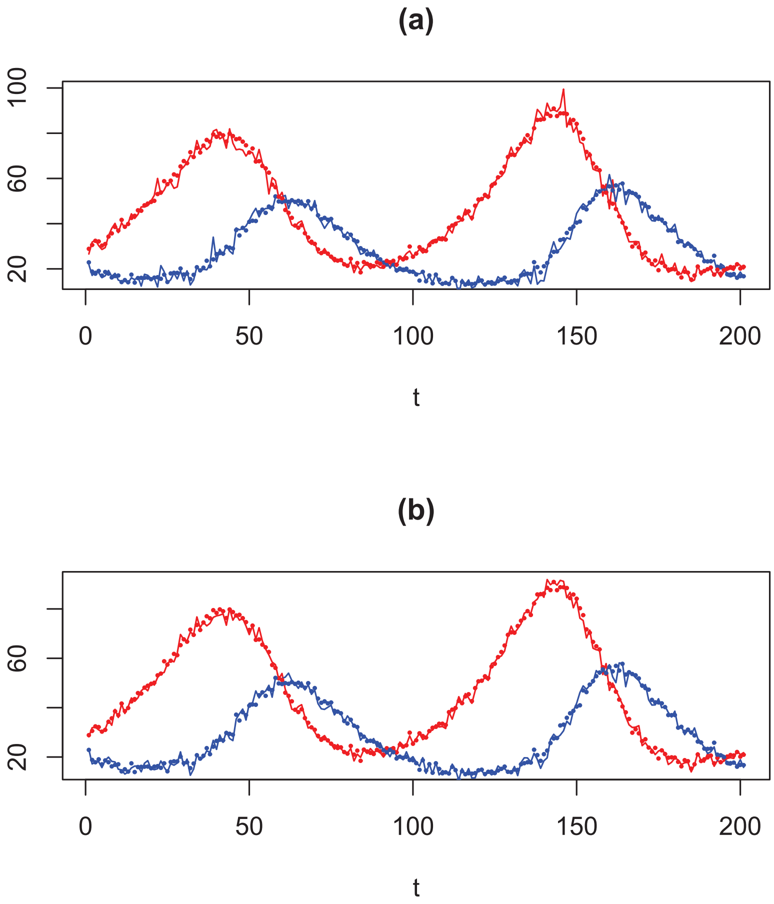


Figure 3.2: Estimated states of the synthetic predator-prey dataset using: (a) the SIR algorithm and (b) the PF-MH algorithm. Shown are the observations of the predator and prey populations (blue and red dots respectively) and the estimated states (blue and red lines respectively).

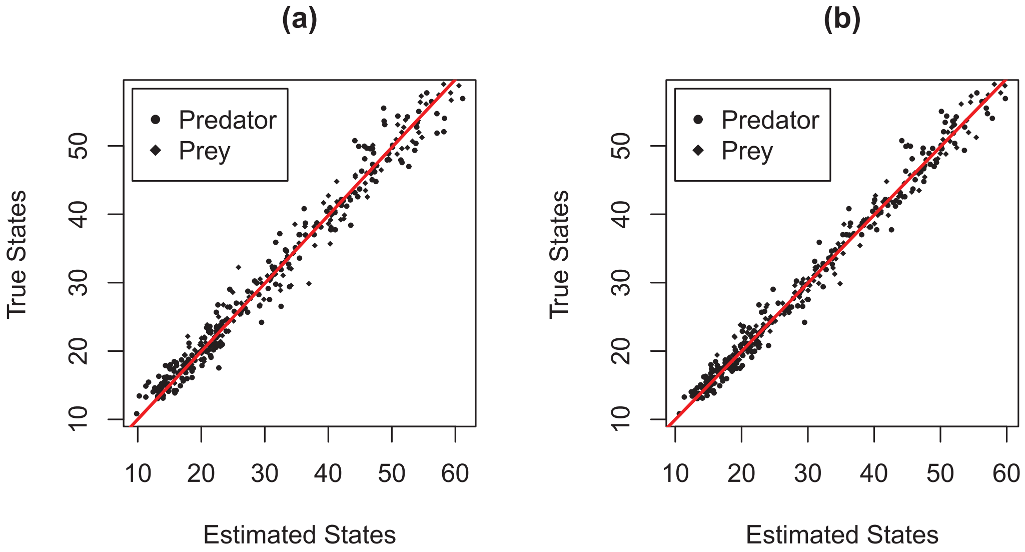


Figure 3.3: Scatter plots of the true and the estimated values of the predator and prey populations using: (a) the SIR algorithm and (b) the PF-MH algorithm where the red line represents the regression line.

### *3.2.2 Parameter Estimation*

This section attempts to estimate the four dynamical parameters in the predator-prey model using the various methods examined in Chapter 2. In these examples, the same synthetic predator-prey dataset is used.

#### *3.2.2.1 Maximum Likelihood Estimation*

The first approach considered is the maximum likelihood method. The implementation of the ML method uses a particle sample size of 300 and assumes a system error of $\mathbf{e}_t^m \sim N(\mathbf{0}, \Sigma_m)$ and an observation error of $\mathrm{e}_t^o \sim N(\mathbf{0}, \Sigma_o)$, where both $\Sigma_m$ and $\Sigma_o$ are (2x2) diagonal matrices with the diagonal entries equal to 2.

Figure 3.4 shows four profile log-likelihoods where each profile represents an individual parameter varying while the remaining three parameters are fixed at their base case values. The log-likelihoods are calculated for each parameter with the range of values, $r_1$ = [0.30, 1.50], $b_1$ = [0, 0.019], $r_2$ = [0.21, 1.00] and $b_2$ = [-0.015, 0.031] while fixing the values of the other parameters at their (known) true values. As a result, the MLE for each dynamical parameter is $r_1 = 1.18, b_1 = 0.011, r_2 = 0.50$ and $b_2 = 0.013$. Recall that the true dynamical parameter values are $r_1 = 0.65, b_1 = 0.023, r_2 = 0.65$ and $b_2 = 0.014$.

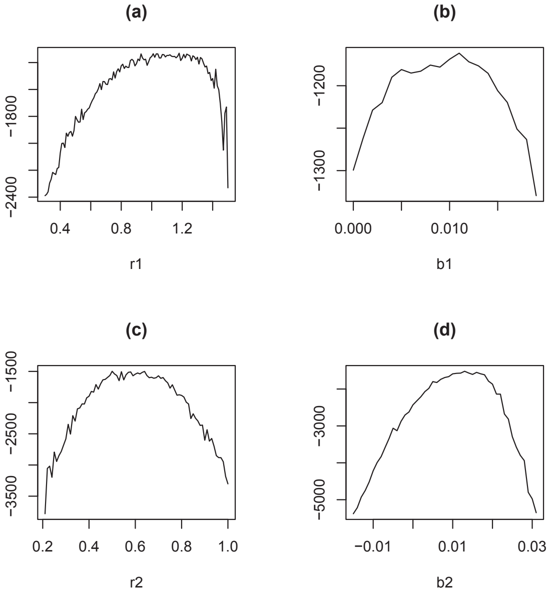


Figure 3.4: Plots of the log-likelihood function for 1 varying parameter (with the remaining 3 parameters fixed) using the synthetic predator-prey dataset. The figures show: (a) $r_1$ varying (b) $b_1$ varying (c) $r_2$ varying and (d) $b_2$ varying.

These results indicate that this method does not recover the true parameter values accurately. However, notice from Figure 3.4 (for plots (a) $r_1$ and (b) $b_1$ in particular) that the peaks of the profiles are relatively flat. It is also evident that there is a visible roughness to each of the log-likelihoods. This is due to the Monte Carlo variation in the sample based estimation procedure, a feature that is problematic when used with optimization algorithms that are normally used to maximize the likelihood.

The ML method is repeated, but in this example, estimates two dynamical parameters simultaneously. The same particle sample size and errors from the previous ML example are used. Figure 3.5 shows the six contour plots of the log-likelihood functions. Each contour plot represents a set of two varying parameters with the remaining two parameters fixed at their base case values. According to Figure 3.5, the corresponding MLEs are shown in Table 3.2. Notice in Figure 3.5 (c) and (d) that there appears to be a set of MLEs along the diagonal of the plots. This relationship suggests that there is a dependence among the parameters $(r_2, b_2)$ and $(r_1, b_1)$. Therefore, by increasing the value of $b_2$ (for example), a MLE can be artificially found by increasing $r_2$ accordingly (and vice versa). The same outcome is implied for the relationship between $r_1$ and $b_1$. This makes sense in terms of the predator-prey population model which has multiple ways of increasing or decreasing the populations. In general, the contour plots show regions of high valued log-likelihoods. This demonstrates that it is difficult to effectively estimate the individual dynamical parameters in a nonlinear system using maximum likelihood estimation due to the dependence structure imposed by the predator-prey dynamics.

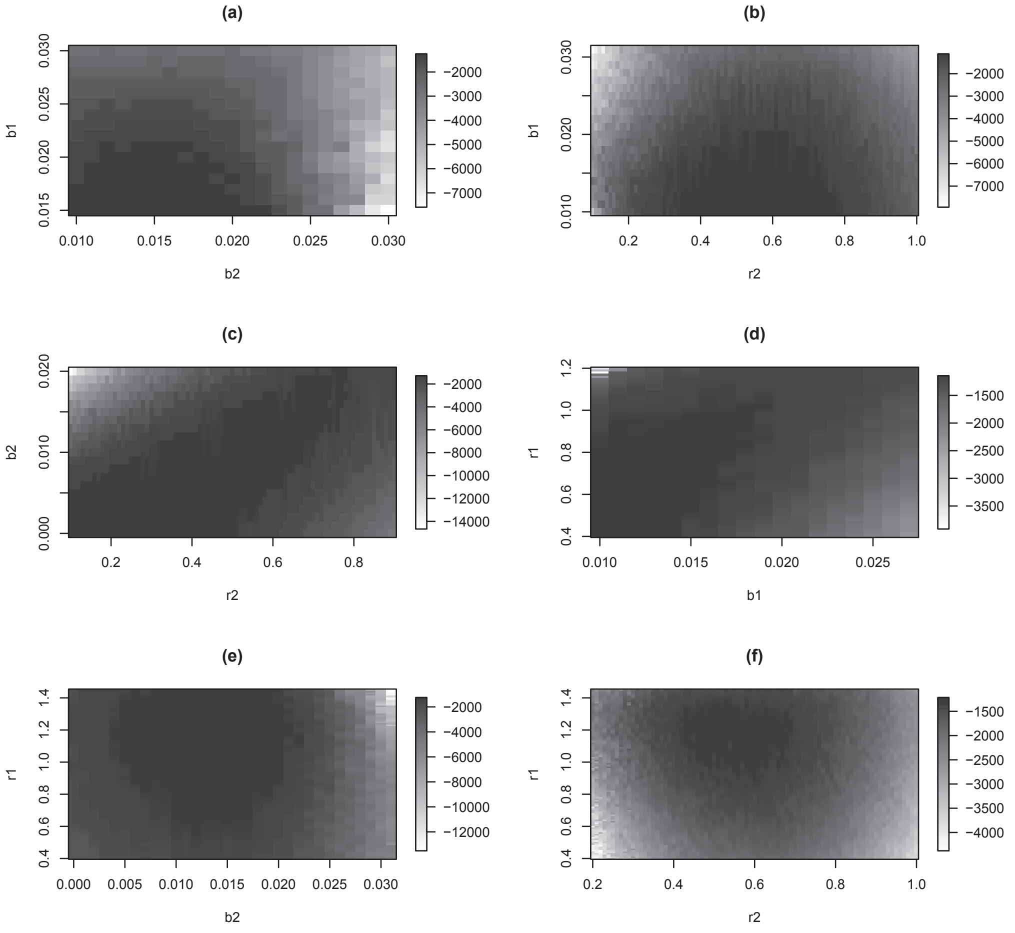


Figure 3.5: Plots of the log-likelihood function for 2 varying parameter (with the remaining 2 parameters fixed) using the synthetic predator-prey dataset. The figures show: (a) $b_1$ and $b_2$ varying (b) $b_1$ and $r_2$ varying (c) $b_2$ and $r_2$ varying (d) $r_1$ and $b_1$ varying (e) $r_1$ and $b_2$ varying and (f) $r_1$ and $r_2$ varying.

Table 3.2: The results of likelihood-based parameter estimation for the synthetic predator-prey model and dataset. The values of the log-likelihoods correspond to the set of two parameters and their maximum likelihood estimates.

| Parameters | MLEs | Value of the Maximum Log-likelihood |
|---|---|---|
| $(b_1, b_2)$ | (0.015, 0.017) | -1252.231 |
| $(b_1, r_2)$ | (0.010, 0.540) | -1120.362 |
| $(b_2, r_2)$ | (0.000, 0.170) | -1279.951 |
| $(r_1, b_1)$ | (0.430, 0.010) | -1144.034 |
| $(r_1, b_2)$ | (1.130, 0.015) | -1252.329 |
| $(r_1, r_2)$ | (1.150, 0.520) | -1215.770 |

#### *3.2.2.2 State Augmentation*

The next parameter estimation technique utilizes state augmented particle filtering. Recall that this method uses the SIR algorithm to approximate the joint filter density between the state and unknown parameters. It augments the state vector, $\mathbf{x}_t$, to include the vector of unknown parameters, $\boldsymbol{\theta}_t$. The example will initially estimate the parameters individually with one parameter varying while the remaining three parameters are fixed. Figure 3.6 shows the trace of each parameter. In this example, the particle sample size is 400 and assumes the same system and observation errors, $\mathbf{e}_t^m \sim N(\mathbf{0}, \Sigma_m)$ and $\mathbf{e}_t^o \sim N(\mathbf{0}, \Sigma_o)$, where both $\Sigma_m$ and $\Sigma_o$ are (2x2) diagonal matrices with the diagonal entries equal to 2. Each of the dynamical parameters in Figure 3.6 evolve with time such that, for (a) $r_{1,t} = r_{1,t-1} + e_t^r$, (b) $b_{1,t} = b_{1,t-1} + e_t^b$, (c) $r_{2,t} = r_{2,t-1} + e_t^r$ and (d) $b_{2,t} = b_{2,t-1} + e_t^b$ where $e_t^r \sim N(0, 0.05)$ and $e_t^b \sim N(0, 0.001)$. In addition, the initial parameter positions for $r_1$ and $r_2$ are sampled from the prior distribution, $N(0.5, 0.1)$, while the initial parameter positions for $b_1$ and $b_2$ are sampled from $N(0.01, 0.005)$. Referring to Figure 3.6, the traces of the dynamical parameters tend to travel about the means of, (a) $r_1 = 0.6170$, (b) $b_1 = 0.0196$, (c) $r_2 = 0.6398$ and (d) $b_2 = 0.0116$. These estimates are closer to the true parameter values than the estimates recovered in the likelihood based method.

To test the estimation of multiple parameters simultaneously, the state augmented particle filter is executed when all four dynamical parameters are varying. The parameters evolve with time such that, $\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \mathbf{e}_t^\theta$, where $\mathbf{e}_t^\theta$ is the vector,

$$\mathbf{e}_t^{\theta} = \begin{bmatrix} e_t^{r_1} \sim N(0, 0.05) \\ e_t^{b_1} \sim N(0, 0.001) \\ e_t^{r_2} \sim N(0, 0.05) \\ e_t^{b_2} \sim N(0, 0.001) \end{bmatrix}.$$

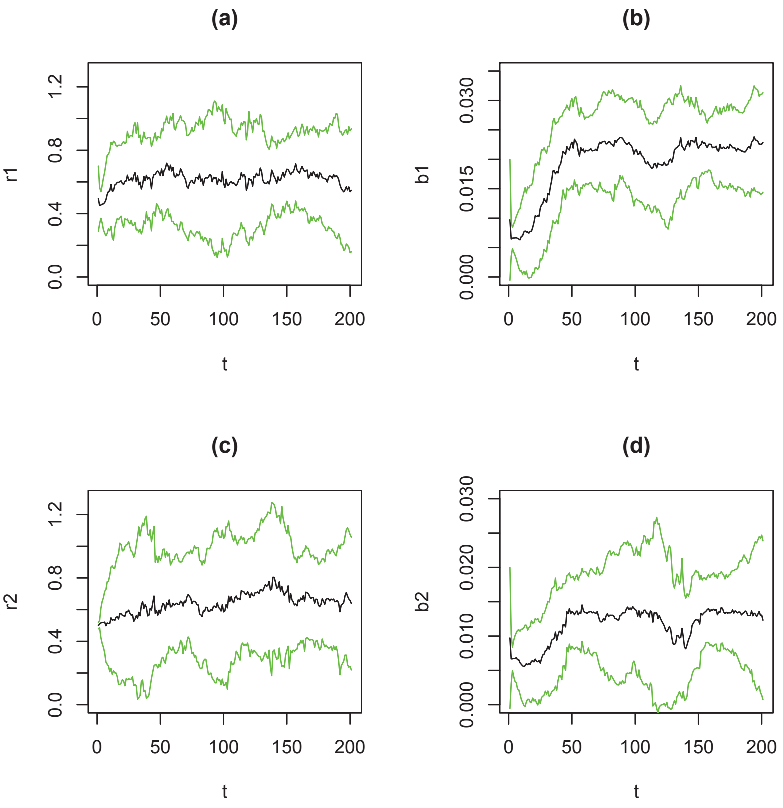


Figure 3.6: Trace of the static, dynamical parameters that model the synthetic predator-prey dataset. The parameters are estimated individually with one parameter varying (and the remaining three parameters fixed) via state augmentation. Shown are the 95% confidence bounds (green lines) and the parameter estimates (black line) where the figures show: (a) the trace of $r_1$ (b) the trace of $b_1$ (c) the trace of $r_2$ and (d) the trace of $b_2$.

Unchanged from the previous example, the initial parameter positions for $r_1$ and $r_2$ are sampled from the prior distribution, $N(0.5, 0.1)$, and the initial parameter positions for $b_1$ and $b_2$ are sampled from $N(0.01, 0.005)$. Figure 3.7 displays the trace of the estimated dynamical parameters when estimated simultaneously via state augmentation. Furthermore, the results from Figure 3.7 illustrate that the traces move about the means of $r_1 = 0.4733$,

$b_1 = 0.0154$, $r_2 = 0.5228$ and $b_2 = 0.0108$. These estimates are not as accurate as the values of estimating one parameter at a time via state augmentation. Evidently, it is difficult to simultaneously and accurately estimate multiple parameters (in a nonlinear system with a dependence structure) using state augmented particle filtering. As an additional note, these estimates do not settle down to constant values. However, recall that this synthetic predator-prey example uses constant parameter values of $r_1$ = 0.65, $b_1$ = 0.023, $r_2$= 0.65 and $b_2$ = 0.014 and therefore are not time-varying. As a result, the traces shown in Figure 3.7 are indicative of a dependence structure.

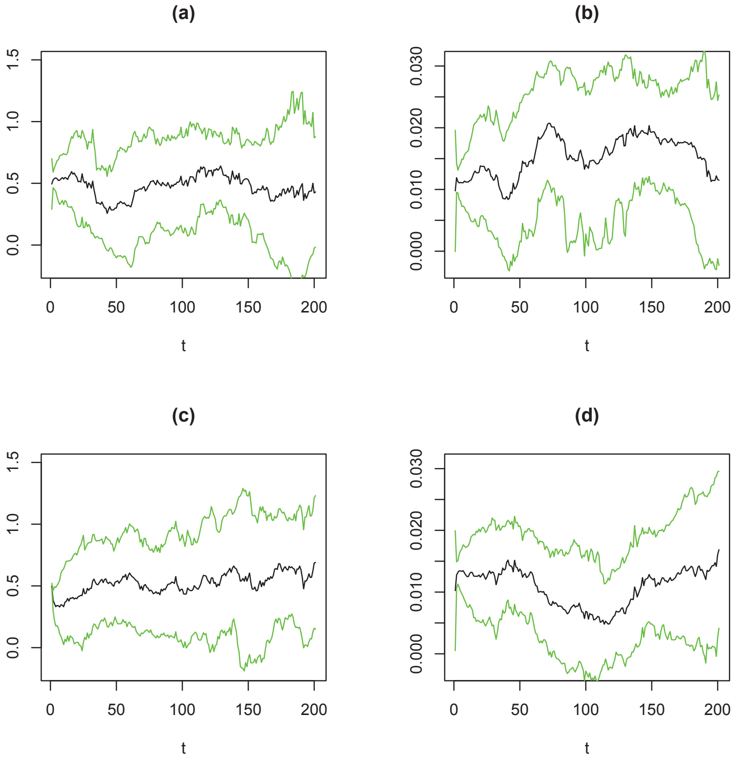


Figure 3.7: Traces of the dynamical parameters that model the synthetic predator-prey dataset. The parameters are estimated simultaneously via state augmentation. Shown are the 95% confidence bounds (green lines) and the parameter estimates (black line) where the figures show: (a) the trace of $r_1$ (b) the trace of $b_1$ (c) the trace of $r_2$ and (d) the trace of $b_2$.

### *3.2.2.3 Multiple Iterative Filtering*

The next example features the multiple iterative filtering algorithm. This example uses a particle sample size of 200, a discount factor of $\alpha = 0.7$ and assumes the same system

Table 3.3: The results of parameter estimation (of the synthetic predator-prey dataset) after twelve iterations of the MIF algorithm. Note the parameters of the predator-prey model are estimated individually. Each column of the estimations represent the parameter that varies while the remaining three parameters are fixed.

| Iteration | Estimate of $r_1$ | Estimate of $b_1$ | Estimate of $r_2$ | Estimate of $b_2$ |
|---|---|---|---|---|
| 1 | 0.1717854 | 0.004205185 | 0.08050758 | 0.007266131 |
| 2 | 0.3428177 | 0.011057986 | 0.17331980 | 0.013016326 |
| 3 | 0.4366784 | 0.014805843 | 0.21111472 | 0.010865334 |
| 4 | 0.4951432 | 0.017192456 | 0.25083057 | 0.012645292 |
| 5 | 0.5378438 | 0.018383749 | 0.28424824 | 0.014000958 |
| 6 | 0.5566364 | 0.019543945 | 0.28867741 | 0.014672408 |
| 7 | 0.5739561 | 0.020810057 | 0.29483931 | 0.015959497 |
| 8 | 0.5855186 | 0.021416252 | 0.30528206 | 0.016207431 |
| 9 | 0.5907506 | 0.021893999 | 0.31055583 | 0.016042155 |
| 10 | 0.5966658 | 0.022046881 | 0.31199082 | 0.016187363 |
| 11 | 0.6016919 | 0.022346022 | 0.31336732 | 0.016188371 |
| 12 | 0.6040568 | 0.022425236 | 0.31398200 | 0.016101349 |

and observation errors, $\mathbf{e}_t^m \sim N(\mathbf{0}, \Sigma_m)$ and $\mathbf{e}_t^o \sim N(\mathbf{0}, \Sigma_o)$, where both $\Sigma_m$ and $\Sigma_o$ are (2x2) diagonal matrices with the diagonal entries equal to 2. Table 3.3 shows the results for estimating the dynamical parameters individually using the MIF algorithm (i.e. allowing one parameter to vary while holding the remaining three parameters fixed). For each parameter, the initial starting values, $\hat{\theta}_{k=0}$ are, $r_1 = 0$, $b_1 = 0.005$, $r_2 = 0$ and $b_2 = 0.005$ and the initial standard deviations, $\sigma_{\theta,k=0}$ are, $\sigma_{r_1} = 0.05$, $\sigma_{b_1} = 0.005$, $\sigma_{r_2} = 0.05$ and $\sigma_{b_2} = 0.005$. In addition, this example uses K = 12 iterations. As a result, the MIF algorithm yields final parameter estimates of, $r_1 = 0.6040568$, $b_1 = 0.022425236$, $r_2 = 0.31398200$ and $b_2 = 0.016101349$. These estimates are in a close proximity to the true parameter values except for parameter, $r_2$. This may perhaps imply that the data is not informative about $r_2$. However, it is also worth noting that overall, $r_2$ is less sensitive compared to the other dynamical parameters.

Estimating the four dynamical parameters simultaneously is attempted using the MIF algorithm. Again, the MIF algorithm is setup with a particle sample size of 200, a discount factor of $\alpha = 0.7$, K = 12 iterations and assumes the same system and observation errors, $\mathbf{e}_t^m \sim N(\mathbf{0}, \Sigma_m)$ and $\mathbf{e}_t^o \sim N(\mathbf{0}, \Sigma_o)$, where both $\Sigma_m$ and $\Sigma_o$ are (2x2) diagonal matrices with the diagonal entries equal to 2. The initial starting values of the parameters in $\hat{\boldsymbol{\theta}}_{k=0}$

Table 3.4: The results of parameter estimation (of the synthetic predator-prey dataset) after twelve iterations of the MIF algorithm. Note the parameters of the predator-prey model are estimated simultaneously.

| Iteration | Estimate of $r_1$ | Estimate of $b_1$ | Estimate of $r_2$ | Estimate of $b_2$ |
|---|---|---|---|---|
| 1 | 0.3324379 | 0.015104058 | 0.2700656 | 0.03832258 |
| 2 | 0.4061069 | 0.009058946 | 0.3613434 | 0.02731976 |
| 3 | 0.3721362 | 0.014597475 | 0.3696583 | 0.01716036 |
| 4 | 0.3125684 | 0.015603633 | 0.3823655 | 0.01561051 |
| 5 | 0.3145455 | 0.017006550 | 0.3754020 | 0.01575950 |
| 6 | 0.3064652 | 0.016976685 | 0.3896831 | 0.01590219 |
| 7 | 0.3094856 | 0.017439638 | 0.3761400 | 0.01691502 |
| 8 | 0.3105321 | 0.017891057 | 0.3805513 | 0.01584019 |
| 9 | 0.3032411 | 0.017369636 | 0.3814908 | 0.01452703 |
| 10 | 0.3015772 | 0.017262597 | 0.3786975 | 0.01413970 |
| 11 | 0.3014240 | 0.016986490 | 0.3778125 | 0.01370592 |
| 12 | 0.3013843 | 0.016979189 | 0.3790630 | 0.01361188 |

are, $r_1 = 0.30$, $b_1 = 0.050$, $r_2 = 0.30$ and $b_2 = 0.050$ and the initial standard deviations in $\boldsymbol{\sigma}_{\theta,k=0}$ are, $\sigma_{r_1} = 0.05$, $\sigma_{b_1} = 0.005$, $\sigma_{r_2} = 0.05$ and $\sigma_{b_2} = 0.005$. Table 3.4 shows the results of this example and generates final parameter estimates of $r_1 = 0.3013843$, $b_1 = 0.016979189$, $r_2 = 0.3790630$ and $b_2 = 0.01361188$. It appears that the parameter, $b_2$ is the only recovered parameter that is close to its true value. In this MIF example, estimating the four dynamical parameters simultaneously does not yield results as accurate as estimating the parameters one at a time.

#### *3.2.2.4 Particle MCMC*

The final examined parameter estimation technique is the MH-PF-MH algorithm. Figure 3.8 shows the traces of the estimated dynamical parameters, using the MH-PF-MH algorithm. The parameters are estimated individually with one parameter varying and the remaining three parameters fixed. The example also uses a particle sample size of 200, initial parameter values of $r_1 = 0.50$, $b_1 = 0.010$, $r_2 = 0.30$ and $b_2 = 0.005$, K = 500 iterations and assumes the system and observation errors, $\mathbf{e}_t^m \sim N(\mathbf{0}, \Sigma_m)$ and $\mathbf{e}_t^o \sim N(\mathbf{0}, \Sigma_o)$, where $\Sigma_m = I$, the (2x2) identity matrix and $\Sigma_o$ is a (2x2) diagonal matrix with the diagonal entries equal to 3. In addition, the trial parameter values are generated from (a) $r_{1,t}^* = r_{1,t-1} + e_t^r$, (b) $b_{1,t}^* = b_{1,t-1} + e_t^b$, (c) $r_{2,t}^* = r_{2,t-1} + e_t^r$ and (d) $b_{2,t}^* = b_{2,t-1} + e_t^b$, where $e_t^r \sim N(0, 0.05)$ and $e_t^b \sim N(0, 0.001)$. Using the results from Figure 3.8, the burn-in period is computed for each trace of the dynamical parameters. After removing the burn-in period, the estimates of the parameters are, $r_1 \sim N(0.65916, 0.01027)$, $b_1 \sim N(0.02277, 0.00031)$, $r_2 \sim N(0.65098, 0.01024)$ and $b_2 \sim N(0.01380, 0.00030)$. Recall that the true dynamical parameters have values of, $r_1 = 0.65$, $b_1 = 0.023$, $r_2 = 0.65$ and $b_2 = 0.014$. This shows that when estimating the dynamical parameters one at a time, the MH-PF-MH algorithm closely recovers the true parameter values, more so than any of the other parameter estimation techniques discussed in this thesis.

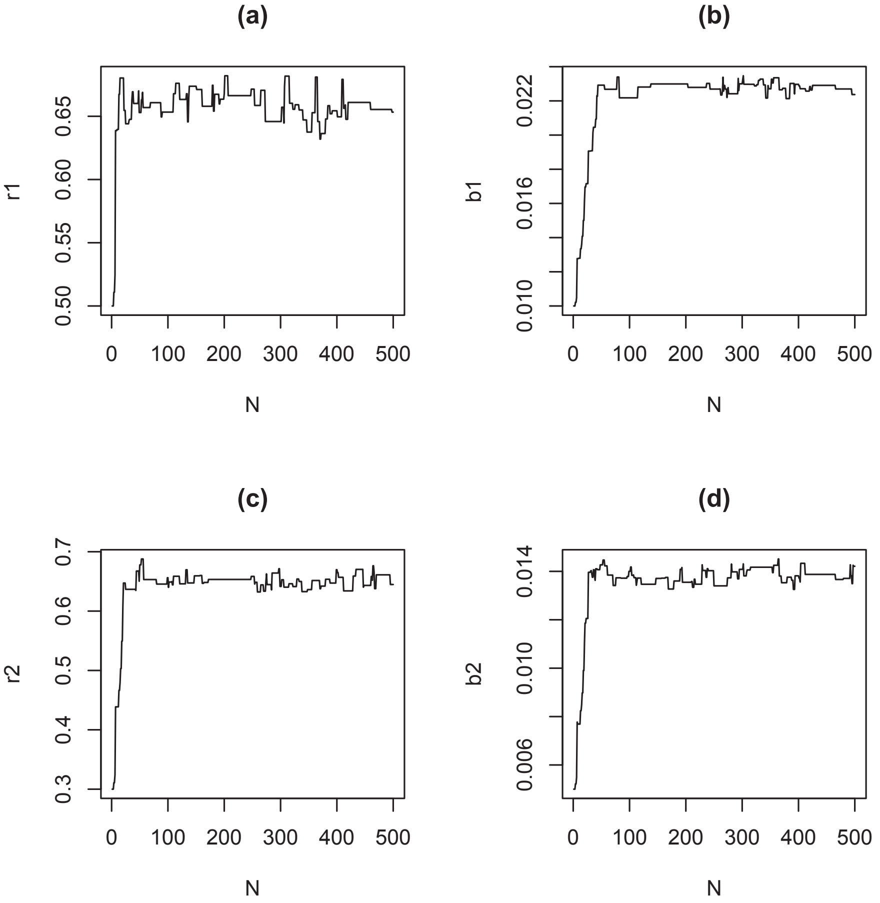


Figure 3.8: Plots of the static parameters that model the synthetic predator-prey dataset. The parameters are estimated individually with one parameter varying (and the remaining three parameters fixed) using the MH-PF-MH update algorithm. Shown are the estimated values of the parameters: (a) $r_1$ (b) $b_1$ (c) $r_2$ and (d) $b_2$.

The MH-PF-MH algorithm is implemented again, estimating the four dynamical parameters simultaneously. The results are shown in Figure 3.9. The example uses a particle sample size of 200, initial parameter values of $r_1 = 0.30$, $b_1 = 0.010$, $r_2 = 0.30$ and $b_2 = 0.005$, K = 10,000 iterations and assumes the system and observation errors, $\mathbf{e}_t^m \sim N(\mathbf{0}, \Sigma_m)$ and $\mathbf{e}_t^o \sim N(\mathbf{0}, \Sigma_o)$, where $\Sigma_m = I$, the (2x2) identity matrix and $\Sigma_o$ is a (2x2) diagonal matrix with the diagonal entries equal to 4.5. In addition, the new values of $\boldsymbol{\theta}$ are generated from $\boldsymbol{\theta}_k^* = \boldsymbol{\theta}_{k-1} + \mathbf{Q}$ where,

$$\mathbf{Q} = \begin{bmatrix} Q_{r_1} \sim N(0, 0.05) \\ Q_{b_1} \sim N(0, 0.001) \\ Q_{r_2} \sim N(0, 0.05) \\ Q_{b_2} \sim N(0, 0.001) \end{bmatrix}.$$

The burn-in period for all four parameters in Figure 3.9 is approximately 225 iterations long. After discarding the burn-in period, the results show that the parameters appear to have an approximate distribution of, $r_1 \sim N(0.6907, 0.0433)$, $b_1 \sim N(0.0240, 0.0013)$, $r_2 \sim N(0.6219, 0.0586)$ and $b_2 \sim N(0.0133, 0.0012)$. Figure 3.10 also shows the histograms corresponding to the plots in Figure 3.9. The histograms represent the approximate distribution of each of the dynamical parameters. As seen in all previous examples in Section 3.2, effectively estimating multiple parameters is a more difficult task than estimating the parameters one at a time. Despite this fact, the MH-PF-MH algorithm is still able to closely recover the four parameter's true values even when they are estimated simultaneously.

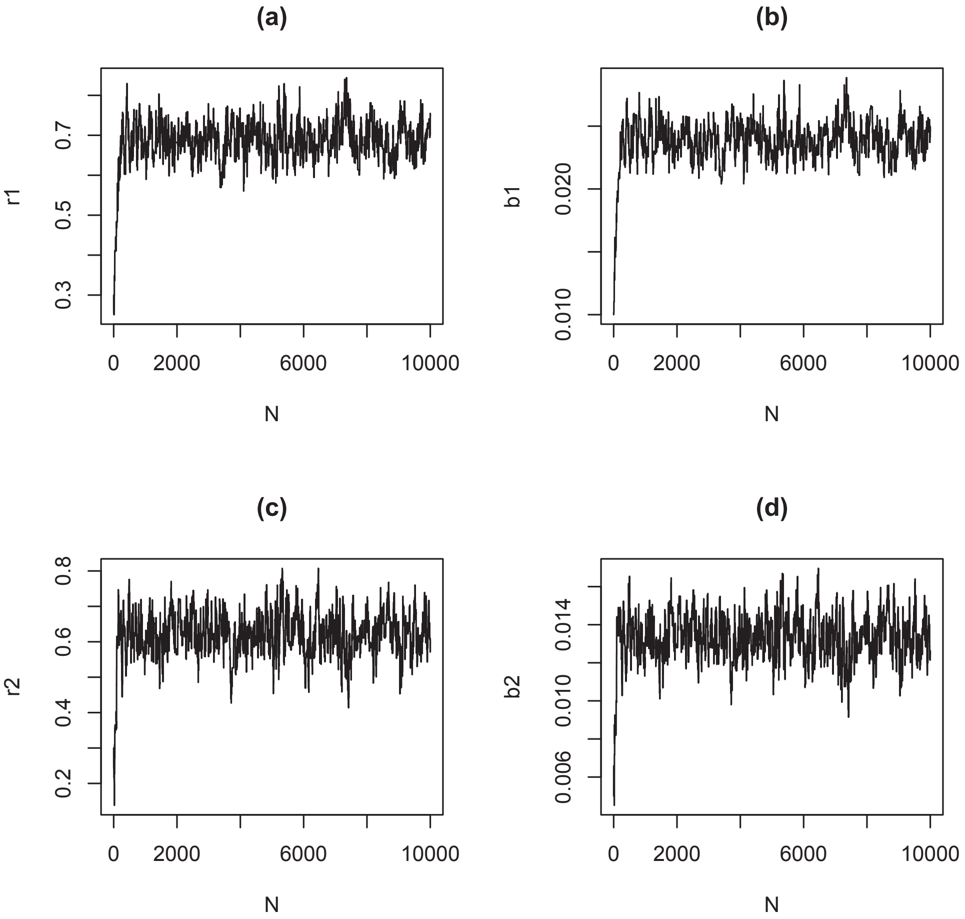


Figure 3.9: Plots of the static parameters that model the synthetic predator-prey dataset. The parameters are estimated simultaneously using the MH-PF-MH update algorithm. Shown are the estimated values of the parameters: (a) $r_1$ (b) $b_1$ (c) $r_2$ and (d) $b_2$.

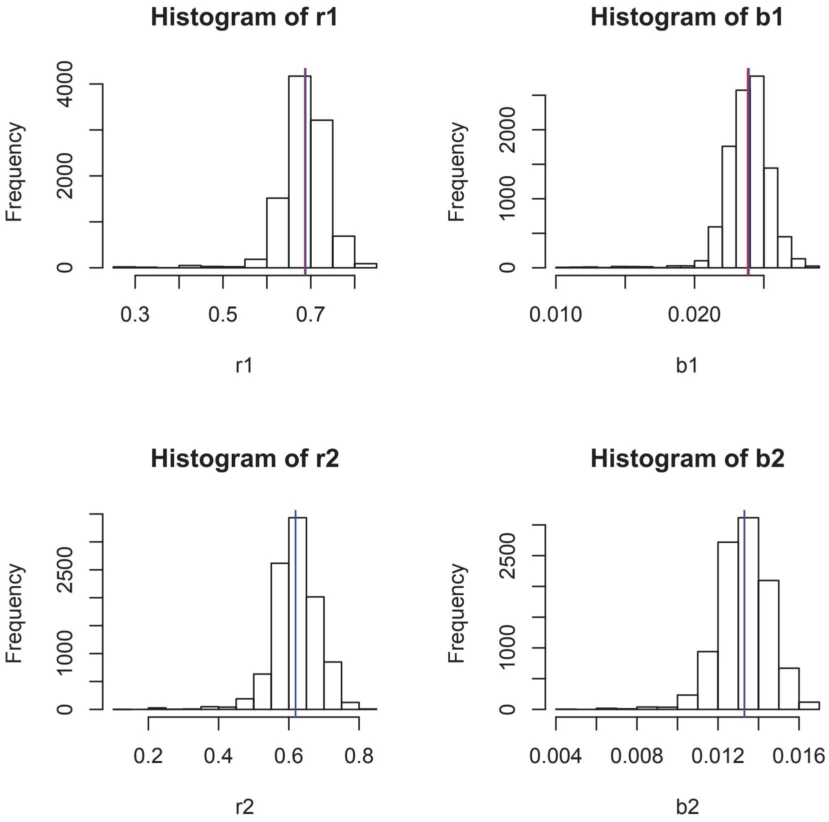


Figure 3.10: Histograms of the dynamical parameters from the predator prey model, corresponding to the results from Figure 3.9. Note, the red vertical lines represent the mean of the distributions and the blue vertical lines represent the median of the distributions.

## 3.3 Lynx-Hare Dataset

This section applies the techniques used in Section 3.2 to the lynx-hare dataset. To begin the state and parameter estimation of this dataset, an appropriate initial model for the lynx-hare populations and therefore, the four dynamical parameters ($r_1$, $b_1$, $r_2$, $b_2$) need to be specified. Equations for the initial parameter estimates are given by

$$b_1 = \frac{r_1}{N_2}, \qquad b_2 = \frac{r_2}{N_1}, \qquad \sqrt{r_2} = \frac{2\pi}{T\sqrt{r_1}}, \qquad r_1 = A\frac{N_2}{N_1}\frac{2\pi}{T}. \tag{3.5}$$

These equations are from *Renshaw* (1993) and provide approximate estimates of the parameters using dynamical considerations from the features of the observed series. Here, $T$, represents the period of the population cycle and $A$ is the amplitude ratio of the prey to the predator. It is assumed that the lynx-hare dataset has a period of $T = 12$ and an amplitude ratio of $A = 1.9807$. As a result, the initial parameter estimates are, $r_1 = 0.6472, b_1 = 0.02284, r_2 = 0.6508$ and $b_2 = 0.01433$. Figure A.9 in Appendix A shows the estimated lynx and hare populations based on these initial estimates.

### *3.3.1 State Estimation*

The states, $N_{1,t}$ and $N_{2,t}$, are estimated using the SIR algorithm and the PF-MH algorithm. Figure 3.11 shows the results of both the algorithms using the previously calculated initial parameter estimates and a particle sample size of 200. In addition, it is assumed that the system and observation errors are, $\mathbf{e}_t^m \sim N(\mathbf{0}, \Sigma_m)$ and $\mathbf{e}_t^o \sim N(\mathbf{0}, \Sigma_o)$, where both $\Sigma_m$ and $\Sigma_o$ are (2x2) diagonal matrices with the diagonal entries equal to 2. The MSE between the estimated states and the observed states are computed for the SIR example and the PF-MH example. The MSEs for the lynx and hare populations (respectively) are 11.9243 and 26.3576 for the SIR algorithm and 8.0436 and 24.0311 for the PF-MH algorithm. Correlation coefficients are also computed and yield values of 0.9919226 for the SIR algorithm and 0.9947966 for the PF-MH algorithm. Figure 3.12 also shows scatter plots of the observed values versus the estimated states of the lynx and hare populations for both algorithms. The combination of these results indicate that the PF-MH algorithm slightly outperforms the SIR algorithm in terms of recovering closer state estimates to the observed lynx-hare populations.

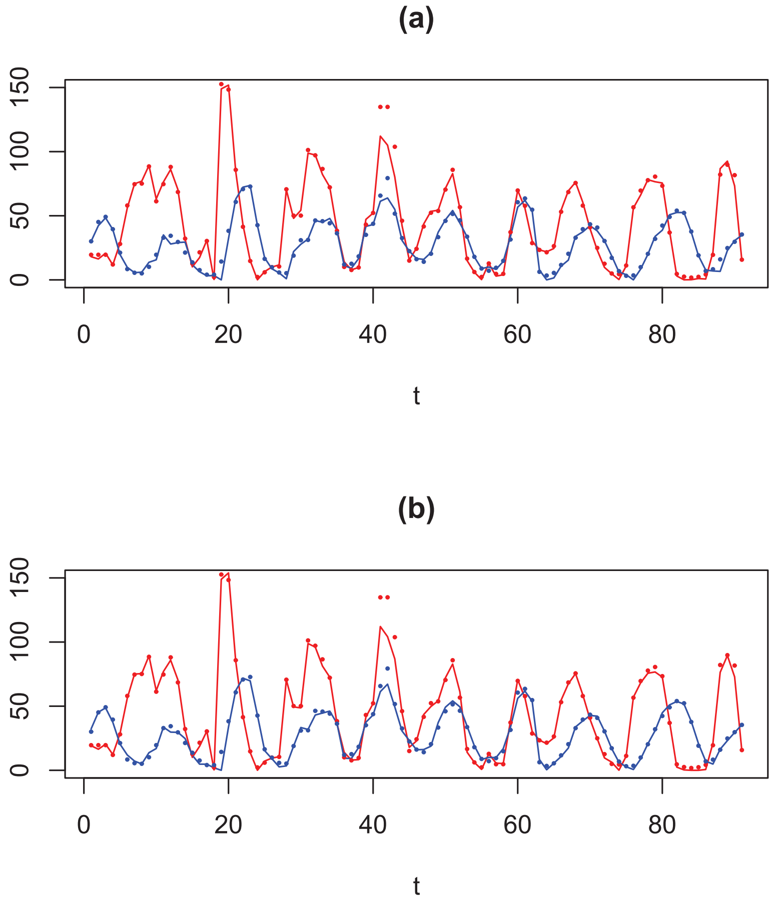


Figure 3.11: Estimated states of the lynx-hare populations (in thousands) using the predator-prey model and (a) the SIR algorithm and (b) the PF-MH algorithm. Shown are the observations of the lynx and hare populations (blue and red dots respectively) and the estimated states (blue and red lines respectively).

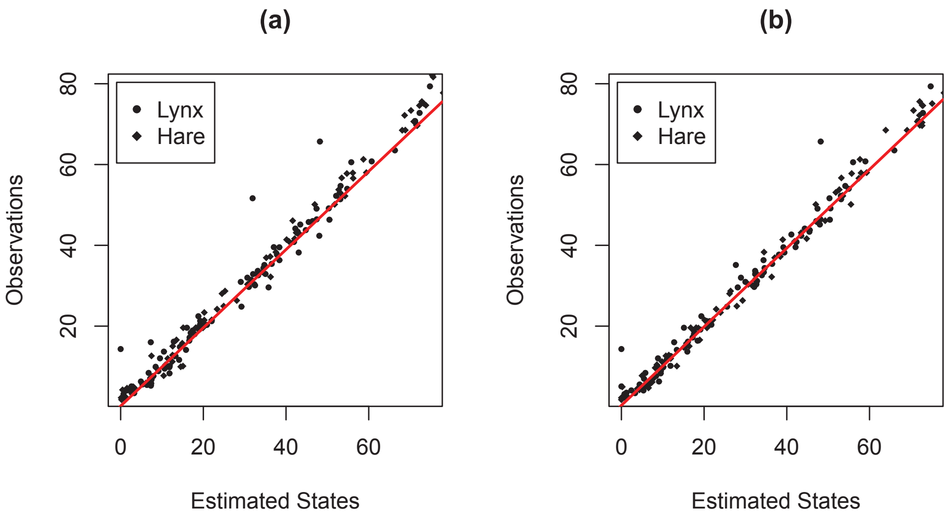


Figure 3.12: Scatter plots of the observations and the estimated values of the lynx and hare populations using: (a) the SIR algorithm and (b) the PF-MH algorithm where the red line represents the line of best fit.

### *3.3.2 Static Parameter Estimation*

Applying the parameter estimation techniques to the entire lynx-hare dataset is inappropriate. This is due to the changing periods and amplitudes of the dataset throughout the 90 year period, which implies that the parameters themselves are non-stationary. In addition, the estimation techniques described in this thesis are primarily used to recover static parameters (with the exception of state augmentation). This section attempts static parameter estimation for one segment (or window) of the dataset and then attempts parameter estimation for the entire dataset by dividing it into multiple overlapping windows. In doing so, it is assumed that the lynx-hare dataset is described by slowly evolving dynamical parameters.

Section 3.2 applies the estimation techniques to the synthetic predator-prey dataset and shows that two oscillations worth of information is enough to successfully recover the dynamical parameters. The segment of the lynx-hare dataset shown in Figure 3.13 is from 1909 to 1931 and also shows a window of approximately two oscillations. This segment is chosen for its consistency (in terms of the period, phase shift and amplitude) and the length of the segment is chosen to provide enough information for effective parameter estimation. Thus, for the purpose of this section, only the data from 1909 to 1931 (which consist of 23 observations) will be used.

The equations from (3.5) are used to obtain an initial model for the window corresponding to the lynx-hare dataset from 1909 to 1931. In addition, the parameters were slightly scaled to provide a closer match to the observed data. As a result, the initial estimates of the dynamical parameters are $r_1 = 0.577159, b_1 = 0.02939023, r_2 = 0.6419528$ and $b_2 = 0.0218594$ and uses a $\Delta = 0.1$. The estimated lynx-hare populations using these initial estimates are shown along with the observed lynx-hare data in Figure 3.13.

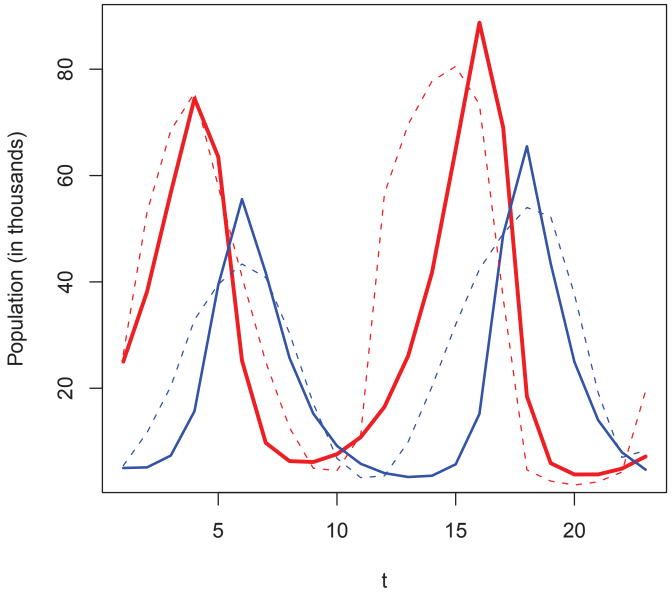


Figure 3.13: Time series plot from 1909-1931 of the observed lynx-hare populations and the estimated lynx-hare populations using the initial estimates of the dynamical parameters. Shown are the observed populations of the lynx (blue dashed line) and the hare (red dashed line) and the estimated populations of the lynx (thick blue line) and the hare (thick red line).

### *3.3.2.1 Maximum Likelihood Estimation*

This section begins with the likelihood based parameter estimation technique. This example of the ML method uses a particle sample size of 300 and assumes a system error of $\mathbf{e}_t^m \sim N(\mathbf{0}, \Sigma_m)$ and an observation error of $\mathbf{e}_t^o \sim N(\mathbf{0}, \Sigma_o)$, where both $\Sigma_m$ and $\Sigma_o$ are (2x2) diagonal matrices with the diagonal entries equal to 2. Further, the ML method is implemented by estimating a pair of the dynamical parameters simultaneously. For each pair, two parameters are varied while keeping the remaining two parameters fixed at their initial estimated values. The results for each pair of the parameters are illustrated in Figure 3.14, which shows the contour plots of their log-likelihood functions. In addition, when examining Figure 3.14, it is not as evident as it is in Figure 3.5 that a region of high valued log-likelihoods exist; however, significant Monte Carlo variability in the estimated log-likelihoods is evident.

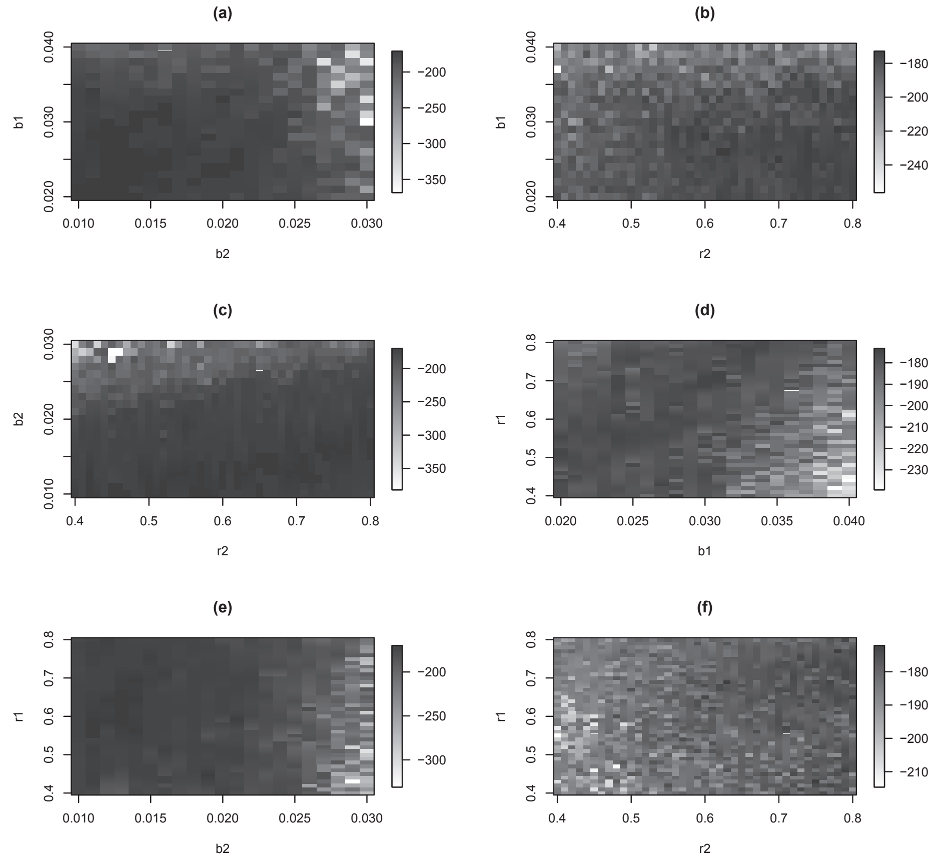


Figure 3.14: Plots of the log-likelihood function for 2 varying parameters (with the remaining 2 parameters fixed) using the lynx-hare dataset. The figures show: (a) $b_1$ and $b_2$ varying (b) $b_1$ and $r_2$ varying (c) $b_2$ and $r_2$ varying (d) $r_1$ and $b_1$ varying (e) $r_1$ and $b_2$ varying and (f) $r_1$ and $r_2$ varying.

Table 3.5 displays the MLEs corresponding to the contour plots in Figure 3.14. These six sets of estimated dynamical parameters are used to reconstruct the predator-prey model that describes the window of lynx-hare data. The ML estimated model plus the observations of the lynx-hare populations are plotted and illustrated in Figure A.10. Using

Table 3.5: The results of likelihood-based parameter estimation for the predator-prey model and lynx-hare dataset. The values of the log-likelihoods correspond to the set of two parameters and their maximum likelihood estimates.

| Parameters | MLEs | Value of the Maximum Log-likelihood |
|---|---|---|
| $(b_1, b_2)$ | (0.026, 0.012) | -170.8696 |
| $(b_1, r_2)$ | (0.025, 0.700) | -172.7982 |
| $(b_2, r_2)$ | (0.012, 0.550) | -169.6896 |
| $(r_1, b_1)$ | (0.550, 0.022) | -173.2389 |
| $(r_1, b_2)$ | (0.650, 0.010) | -169.9622 |
| $(r_1, r_2)$ | (0.690, 0.630) | -172.1461 |

Table 3.6: The RMSEs from likelihood-based parameter estimation corresponding to Table 3.5.

| Parameters | MLEs | RMSEs for the Lynx and Hare |
|---|---|---|
| $(b_1, b_2)$ | (0.026, 0.012) | 28.02451 and 68.57258 |
| $(b_1, r_2)$ | (0.025, 0.700) | 14.13784 and 18.26743 |
| $(b_2, r_2)$ | (0.012, 0.550) | 26.12112 and 58.30325 |
| $(r_1, b_1)$ | (0.550, 0.022) | 18.67945 and 26.88493 |
| $(r_1, b_2)$ | (0.650, 0.010) | 26.38672 and 86.83374 |
| $(r_1, r_2)$ | (0.690, 0.630) | 14.80166 and 16.81630 |

Figure A.10, the root mean squared errors (RMSEs) between the observed values and the estimated values of the lynx and hare populations are calculated for each set of the parameter estimates. They are shown in Table 3.6. Reviewing the results from Table 3.5 and 3.6, likelihood based parameter estimation suggests best estimates (with a RMSE of 14.80166 and 16.81630 for the lynx and hare) of $r_1 = 0.690$, $b_1 = 0.02939023$, $r_2 = 0.630$ and $b_2 = 0.0218594$.

#### *3.3.2.2 State Augmentation*

State augmented particle filtering is the parameter estimation technique examined next. In this case, state augmentation is used to estimate the four dynamical parameters simultaneously. It is implemented with a particle sample size of 500 and assumes that the system error is $\mathbf{e}_t^m \sim N(\mathbf{0}, \Sigma_m)$ and the observation error is $\mathbf{e}_t^o \sim N(\mathbf{0}, \Sigma_o)$, where both $\Sigma_m$ and $\Sigma_o$ are (2x2) diagonal matrices with the diagonal entries equal to 2. Figure 3.15 shows the traces of the dynamical parameters where they evolve with time such that, $\boldsymbol{\theta}_t = \boldsymbol{\theta}_{t-1} + \mathbf{e}_t^\theta$ and $\mathbf{e}_t^\theta$ is the vector,

$$\mathbf{e}_t^{\theta} = \begin{bmatrix} e_t^{r_1} \sim N(0, 0.05) \\ e_t^{b_1} \sim N(0, 0.001) \\ e_t^{r_2} \sim N(0, 0.05) \\ e_t^{b_2} \sim N(0, 0.001) \end{bmatrix}.$$

Recall, that $\boldsymbol{\theta}_t$ is the vector of (unobserved) dynamical parameters. Similar to the synthetic predator prey state augmentation example, the initial parameter positions for $r_1$ and $r_2$ are sampled from the prior distribution, $N(0.5, 0.1)$, and the initial parameter positions for $b_1$ and $b_2$ are sampled from $N(0.01, 0.005)$. Referring to Figure 3.15, the parameters move about the means of, $r_1 = 0.4012, b_1 = 0.0202, r_2 = 1.0143$ and $b_2 = 0.0178$. Again, the parameter estimates are used to reconstruct the lynx-hare predator-prey model (as seen in Figure A.11) and the RMSEs are computed. The RMSEs are, 33.01519 for the lynx and 63.66073 for the hare. These values are higher than the RMSEs calculated for the best estimates from the ML method. This implies that the dynamical parameters estimated via state augmentation do not describe the lynx-hare populations as accurately as the estimates produced from the profile log-likelihoods and their MLEs.

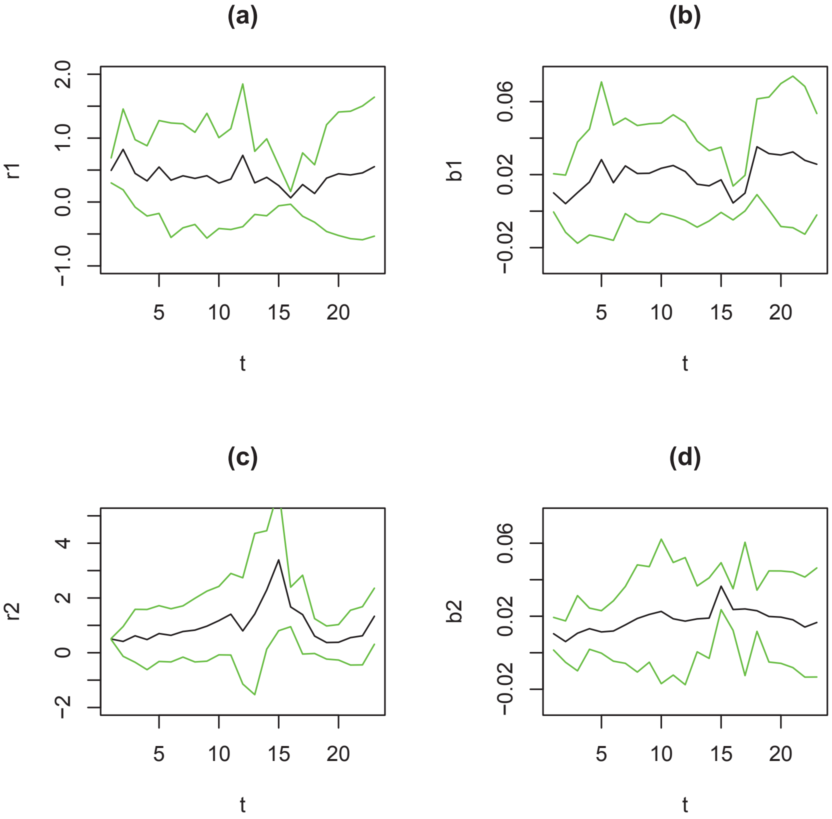


Figure 3.15: Traces of the dynamical parameters from state augmentation that model the chosen segment of the lynx-hare dataset (from 1909 to 1931). The parameters are estimated simultaneously via state augmentation. Shown are the 95% confidence bounds (green lines) and the parameter estimates (black line) where the figures show: (a) the trace of $r_1$ (b) the trace of $b_1$ (c) the trace of $r_2$ and (d) the trace of $b_2$.

Table 3.7: The results of parameter estimation (of the window of the lynx-hare dataset) after sixteen iterations of the MIF algorithm. Note the parameters of the predator-prey model are estimated simultaneously.

| Iteration | Estimate of $r_1$ | Estimate of $b_1$ | Estimate of $r_2$ | Estimate of $b_2$ |
|---|---|---|---|---|
| 1 | 0.6181365 | 0.01183534 | 0.5692287 | 0.020922434 |
| 2 | 0.5668200 | 0.01246191 | 0.4946093 | 0.013783483 |
| 3 | 0.5794797 | 0.01692100 | 0.4569367 | 0.011393320 |
| 4 | 0.6244345 | 0.01784326 | 0.4482243 | 0.008344242 |
| 5 | 0.6194103 | 0.02031142 | 0.4527350 | 0.008313549 |
| 6 | 0.6216470 | 0.02072017 | 0.4548048 | 0.009573483 |
| 7 | 0.6138090 | 0.02070080 | 0.4658748 | 0.009340477 |
| 8 | 0.6175948 | 0.02005266 | 0.4675257 | 0.008976902 |
| 9 | 0.6178928 | 0.02044609 | 0.4683908 | 0.009109928 |
| 10 | 0.6199285 | 0.02048547 | 0.4675356 | 0.009100534 |
| 11 | 0.6202310 | 0.02052716 | 0.4677789 | 0.009284136 |
| 12 | 0.6199073 | 0.02053258 | 0.4679018 | 0.009326388 |
| 13 | 0.6204618 | 0.02065104 | 0.4681984 | 0.009255612 |
| 14 | 0.6204911 | 0.02068566 | 0.4689182 | 0.009276763 |
| 15 | 0.6204869 | 0.02074893 | 0.4689820 | 0.009339121 |
| 16 | 0.6205990 | 0.02076818 | 0.4689508 | 0.009315681 |

### *3.3.2.3 Multiple Iterative Filtering*

The four dynamical parameters are estimated simultaneously using the segment of the lynx-hare dataset and the multiple iterative filtering algorithm. The MIF algorithm is setup with a particle sample size of 400, a discount factor of $\alpha = 0.7$, K = 16 iterations and assumes a system error of $\mathbf{e}_t^m \sim N(\mathbf{0}, \Sigma_m)$ and an observation error of $\mathbf{e}_t^o \sim N(\mathbf{0}, \Sigma_o)$, where $\Sigma_m$ and $\Sigma_o$ are (2x2) diagonal matrices with the diagonal entries equal to 2 for $\Sigma_m$ and 4 for $\Sigma_o$. The initial estimates of, $r_1 = 0.577159, b_1 = 0.02939023, r_2 = 0.6419528$ and $b_2 = 0.0218594$ are used. In addition, the initial standard deviations in $\boldsymbol{\sigma}_{\theta,k=0}$ are, $\sigma_{r_1} = 0.05$, $\sigma_{b_1} = 0.005$, $\sigma_{r_2} = 0.05$ and $\sigma_{b_2} = 0.005$. The results of the MIF algorithm are shown in Table 3.7 and presents final parameter estimates of $r_1 = 0.3013843$, $b_1 = 0.016979189$, $r_2 = 0.3790630$ and $b_2 = 0.01361188$. The reconstructed lynx-hare predator-prey model is presented in Figure A.12 with RMSEs of 45.81733 for the lynx and 88.88903 for the hare.

#### *3.3.2.4 Particle MCMC*

Lastly, the MH-PF-MH algorithm is utilized to attempt the accurate estimation of the dynamical parameters. This example estimates the parameters simultaneously and uses a particle sample size of 200. As well, the implementation of the MH-PF-MH algorithm is setup with K = 10,000 iterations, initial parameter values of $r_1 = 0.577159, b_1 = 0.02939023, r_2 = 0.6419528$ and $b_2 = 0.0218594$ and assumes a system error of $\mathbf{e}_t^m \sim N(\mathbf{0}, \Sigma_m)$ and an observation error of $\mathbf{e}_t^o \sim N(\mathbf{0}, \Sigma_o)$, where $\Sigma_m$ and $\Sigma_o$ are (2x2) diagonal matrices with the diagonal entries equal to 3 for $\Sigma_m$ and 10 for $\Sigma_o$. Furthermore, the new values of $\boldsymbol{\theta}$ are generated from $\boldsymbol{\theta}_k^* = \boldsymbol{\theta}_{k-1} + \mathbf{Q}$ where,

$$\mathbf{Q} = \begin{bmatrix} Q_{r_1} \sim N(0, 0.05) \\ Q_{b_1} \sim N(0, 0.001) \\ Q_{r_2} \sim N(0, 0.05) \\ Q_{b_2} \sim N(0, 0.001) \end{bmatrix}.$$

In this example, a clear burn-in period is not evident. Figure 3.17 shows the histograms of each of the dynamical parameters corresponding to the results in Figure 3.16. The values of the medians in Figure 3.17 are used as the final dynamical parameter estimates. As a result, the final parameter estimates are, $r_1 = 0.929775$, $b_1 = 0.034667$, $r_2 = 0.686116$ and $b_2 = 0.011302$. Figure A.13 shows the reconstructed lynx-hare predator prey model using these MH-PF-MH estimated parameters and produces RMSEs of 24.81985 for the lynx and 68.11814 for the hare. The coefficient of variation (CV) for each of the parameter distributions is computed and yields values of 0.3009441 for $r_1$, 0.1969873 for $b_1$, 0.8015852 for $r_2$ and 0.711433 for $b_2$. As a comparison, the CVs for the synthetic predator-prey MH-PH-MH example are also computed. The values are, 0.07940717 for $r_1$, 0.06992277 for $b_1$, 0.1059487 for $r_2$ and 0.09561 for $b_2$. The CVs from the lynx-hare example are much higher than the CVs from the synthetic predator-prey example. This could indicate that the lynx-hare population data does not evolve according to this particular predator-prey model or that the lynx-hare data alone does not provide enough information to recover the dynamical parameters. In addition, there appears to be a trade off between fitting the phase and amplitude of the segment of lynx-hare data. The values of the CVs for the dynamical parameters also suggest very large error bars. This indicates that the chain has difficulty converging to a set (or a more narrow range) of dynamical

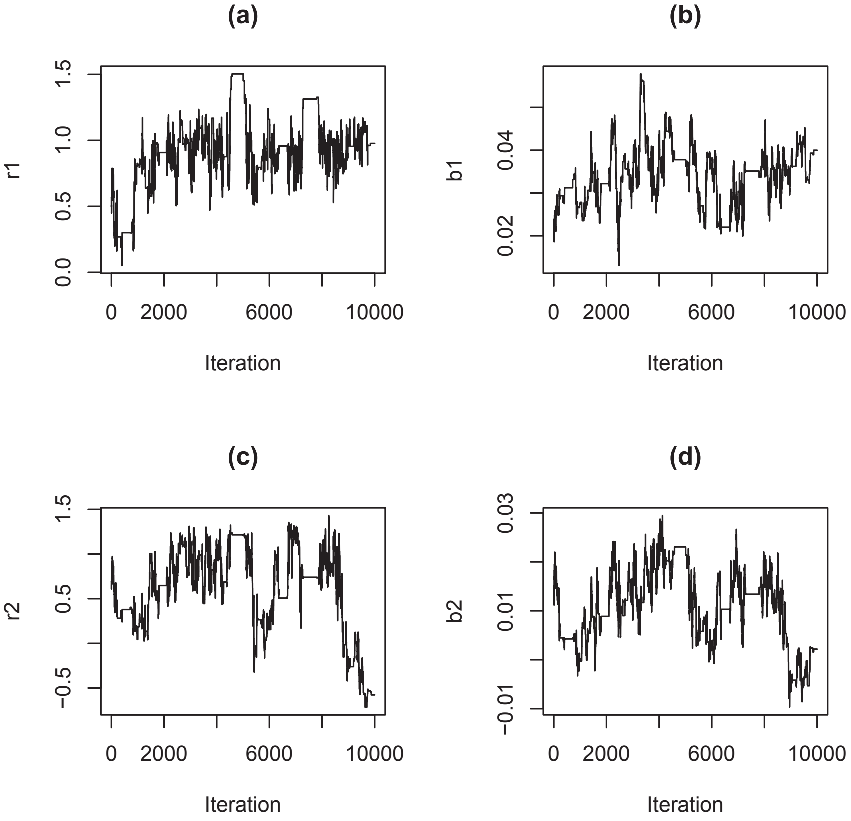


Figure 3.16: Plots of the dynamical parameters from the predator prey model, using the window of the lynx-hare dataset and estimated using the MH-PF-MH update algorithm. Shown are the estimated values of the parameters: (a) $r_1$, (b) $b_1$, (c) $r_2$ and (d) $b_2$.

parameter values. As a result, PMCMC recognizes that a set of values for the dynamical parameters may not exist for this particular model because the lynx-hare data segment appears to have inconsistent dynamics relative to the Lotka-Volterra predator prey model.

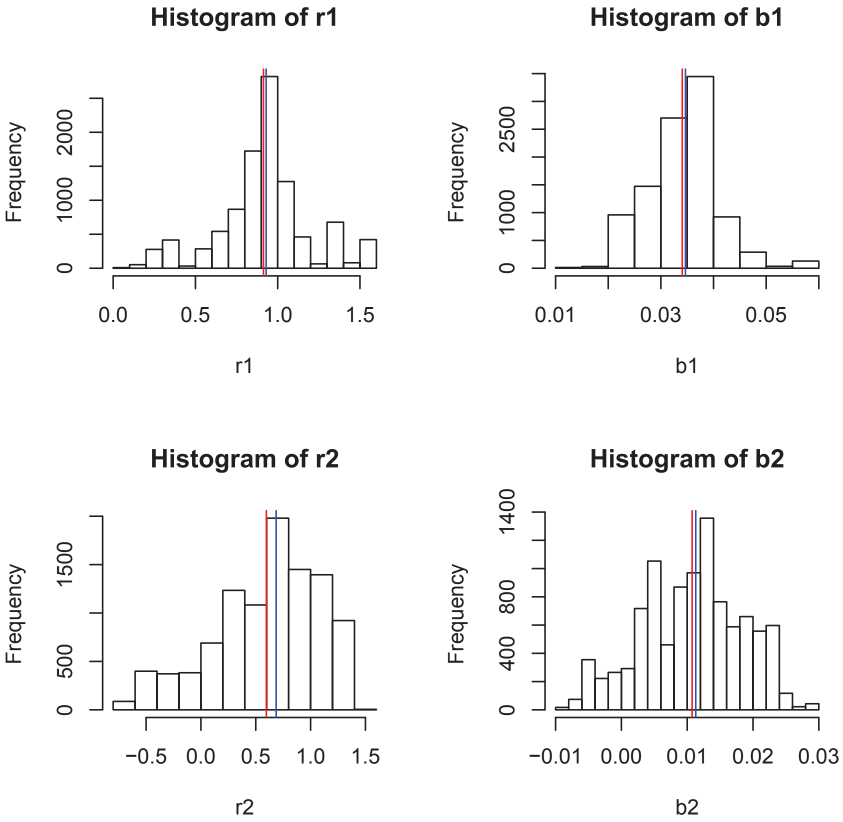


Figure 3.17: Histograms of the dynamical parameters from the predator prey model, corresponding to the results from Figure 3.16. Note, the red vertical lines represent the mean of the distributions and the blue vertical lines represent the median of the distributions.

Table 3.8 shows a summary of the parameter estimation procedures discussed in this chapter. It states the RMSEs along with the sets of final parameter values estimated by each of the parameter estimation techniques. The initial parameter estimates appear to fit the lynx-hare data segment the best (with respect to the RMSEs), mainly because the parameter values were manually scaled to provide a better fit for the data segment. This was intended to improve the efficiency of the parameter estimation algorithms.

Table 3.8: A performance summary of the parameter estimation techniques for fitting the segment of the lynx-hare dataset. The values of the RMSEs corresponding to the parameter estimation techniques along with their final dynamical parameter estimates.

| Technique | Final Parameter Estimates | RMSE |
|---|---|---|
| | $(r_1, b_1, r_2, b_2)$ | (lynx, hare) |
| Initial Estimates | (0.5772, 0.0294, 0.6420, 0.0219) | (11.15701, 18.17424) |
| MLE | (0.6900, 0.0294, 0.6300, 0.0219) | (14.80166, 16.81630) |
| State Augmentation | (0.4012, 0.0202, 1.0143, 0.0178) | (33.01519, 63.66073) |
| MIF | (0.6206, 0.0208, 0.4690, 0.0093) | (45.81733, 88.88903) |
| PMCMC | (0.9300, 0.0347, 0.6861, 0.0113) | (24.81985, 68.11814) |

### *3.3.3 Slowly Evolving Parameter Estimation*

In order to apply the MH-PF-MH algorithm to the entire dataset, an approach is to assume that the lynx-hare dataset has dynamical parameters that evolve slowly with time. As a result, the dataset is sectioned into 8 different segments (or windows). Each window consists of approximately two oscillations (20 observations) with overlapping windows of one oscillation (10 observations). The MH-PF-MH algorithm is applied to each window and run for K = 10,000 iterations. The equations from (3.5) are used to calculate initial parameter estimates for each window. In addition, the same setup from the previous MH-PF-MH example are used here. Figure 3.18 shows the slow evolution of each parameter and Figure 3.19 shows the model of the lynx-hare populations based on the parameter estimates from Figure 3.18. The estimated states within the overlapping windows are averaged in order to produce the smooth curves in Figure 3.19.

Unfortunately, the slowly evolving parameter estimates do not produce oscillations that fit the real lynx-hare dataset well and produce RMSEs of 92.68522 for the hare and 33.52253 for the lynx. However, the oscillations appear to fit at times and diverge at others. This issue is similar to the example of the single segment of observations from 1909 to 1931, in that the dynamics are inconsistent relative to the observations. Histograms of the parameters' distributions are shown in Appendix A for each of the windows. The histograms show that at times, the distribution is widely dispersed, implying that the set of parameter values are difficult to identify. As a result, large CVs and error bars are generated.

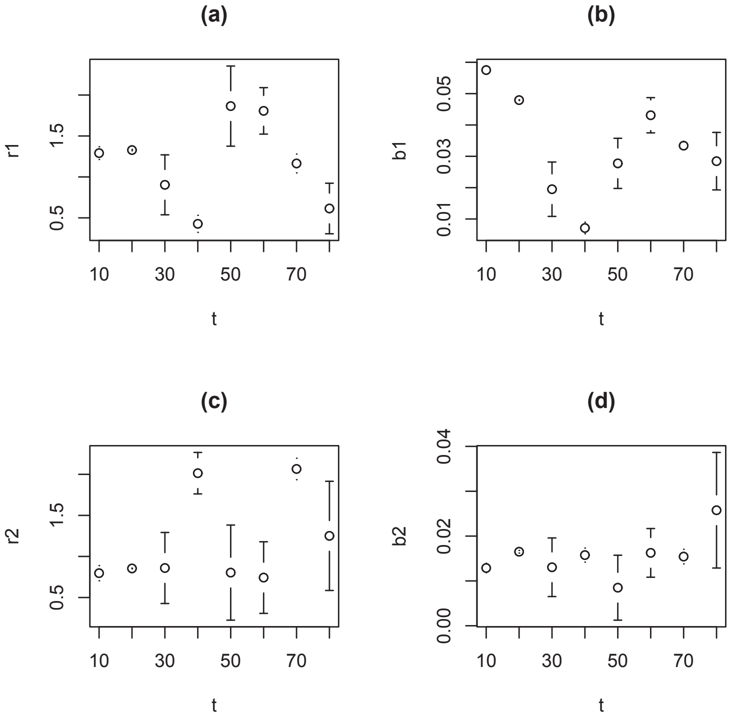


Figure 3.18: The slow evolution of the dynamical parameters: (a) $r_1$, (b) $b_1$, (c) $r_2$ and (d) $b_2$ with error bars representing one standard deviation of uncertainty from the median of each parameter's approximate distributions. The entire lynx-hare dataset is used and the dynamical parameters are estimated using the MH-PF-MH algorithm.

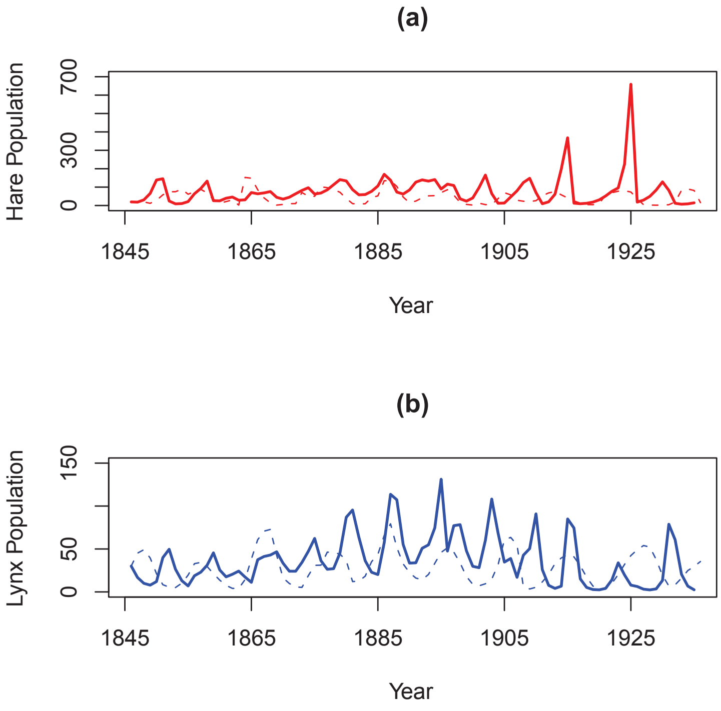


Figure 3.19: Shown are the observed and estimated states of (a) the hare and (b) the lynx corresponding to the estimated slowly evolving dynamical parameters from Figure 3.18. The dashed blue and red lines represent the observed lynx and hare populations and the solid blue and red lines represent the estimated states of the lynx and hare populations.

# CHAPTER 4

# DISCUSSION AND CONCLUSION

This thesis examines various filtering techniques for state and parameter estimation. It begins with a linear and Gaussian example and utilizes the Kalman filter for the estimation of the true values of the state and parameters of the linear system. The Kalman filter demonstrates that it is a useful tool that is able to dismiss the noise in the observations to accurately recover the true values of the states and parameters. The particle filter and MCMC also prove to be useful for state and parameter estimation for a variety of underlying systems with no requirements of linearity or normality.

The linear and Gaussian example is followed by a nonlinear and non-Gaussian example using synthetic data generated from the predator-prey model. In these synthetic dataset examples, the PF-MH algorithm shows that it is more accurate than SIR in terms of reducing the difference between the estimates and the true states. This is due to the incorporation of the MCMC step which has improved the sample diversity.

The parameter estimation techniques examined are maximum likelihood estimation, state augmentation, multiple iterative filtering and particle Markov chain Monte Carlo. The main issue with maximum likelihood estimation is that Monte Carlo methods tend to result in rough likelihood profiles that make it difficult to determine the maximum likelihood estimates. However, state augmentation and multiple iterative filtering attempt to provide alternative solutions as they do not require the optimization of likelihoods and are also efficient in terms of their computation time. In addition, state augmentation is straightforward to implement since it is in standard state space form which makes it readily solvable by standard filtering algorithms. If time is a concern, state augmentation is the most efficient parameter estimation technique discussed here and is an appropriate parameter estimation alternative that provides a rough approximation for parameter estimates.

Particle MCMC is the last parameter estimation technique examined and is the main focus of this thesis. In the synthetic dataset examples, it recovers the parameter estimates closest to their true values (relative to the other techniques examined). The main issue and biggest disadvantage to PMCMC is its computational time. Recall that for every iteration of the PMCMC procedure, M number of particles are generated for each time point from t = 1, ..., N. Generally, the PMCMC procedure is implemented for a large number of iterations in order to observe and discard the burn-in period and to accurately simulate an approximation of the posterior distribution. In the lynx-hare example, the PMCMC algorithm is implemented for a total of K = 10,000 iterations and has a run time of approximately 2 days using Dalhousie University's Department of Mathematics and Statistics NX Server powered by a Dual core Xeon Woodcrest 3 Ghz processor.

In the lynx-hare population example using the real data, no parameter estimation technique distinctly outperforms any of the other techniques. Referring back to Figure 3.17, PMCMC suggests that the lynx-hare population data does not evolve according to the dynamics of the predator-prey model and possibly that the lynx-hare data alone does not provide enough information to recover the dynamical parameters. Although the classic lynx-hare time series dataset is usually modeled by the Lotka-Volterra equations, some believe that the lynx-hare population actually follows the dynamics of other ecological models. For instance, *Zhang et al.* (2007) believe that the population's underlying factor for the oscillations are not caused by the interactions of the lynx and hare but rather the change in climate. *Zhang et al.* (2007) also list other potential models which have been used in previous research to describe the lynx-hare population. These models incorporate the abundance of the hares' plant consumption. This suggests that a more complex dynamical model may be needed to describe this system. Therefore, further knowledge on biology and ecology is required to yield the true underlying model for the lynx and hare population.

Overall, this thesis examines a number of solutions for parameter estimation in nonlinear state space models. The analysis presents specific advantages and disadvantages to each technique. However, if time is not a constraint, PMCMC is the preferred parameter estimation method. It has the advantage over other approaches in that it can approximate any posterior distribution for which inference can be made and is a promising technique for static parameter estimation.

# Bibliography

Andrieu, C., A. Doucet, and R. Holenstein, Particle markov chain monte carlo methods, *Journal of the Royal Statistical Society*, *73*, 269–342, 2010.

Carpenter, J., P. Clifford, and P. Fearnhead, Building robust simulation-based filters for evolving data sets, *Tech. rep.*, University of Oxford, Department of Statistics, 1999.

Evensen, G., Using the extended kalman filter with a multilayer quasi-geostrophic ocean model, *Journal of Geophysical Research*, *97*, 17,905–17,924, 1992.

Evensen, G., Sequential data assimilation with a nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics, *Journal of Geophysical Research*, *99*, 10,143–10,162, 1994.

Gilks, W. R., and C. Berzuini, Following a moving target-monte carlo inference for dynamic bayesian models, *Journal of the Royal Statistical Society*, *63*, 127–146, 2001.

Gordon, N. J., D. J. Salmond, and A. F. M. Smith, Novel approach to nonlinear/non-gaussian bayesian state estimation, *Radar and Signal Processing, IEE*, *140*, 107–113, 1993.

Harvey, A. C., *Forecasting, Structural Time Series Models and the Kalman Filter*, Cambridge University Press, 1991.

Hewitt, C. G., *The conservation of the wild life of Canada*, C. Scribner's Sons, 1921.

Ionides, E. L., C. Breto, and A. A. King, Inference for nonlinear dynamical systems, *Proceedings of the National Academy of Science*, *103*, 18,438–18,443, 2006.

Jones, E., J. Parslow, and L. Murray, A bayesian approach to state and parameter estimation in a phytoplankton-zooplankton model, *Australian Meteorological and Oceanographic Journal*, *59*, 7–16, 2010.

Kalman, R. E., A new approach to linear filtering and prediction problems, *Transactions of the ASME - Journal of Basic Engineering*, *82*, 35–45, 1960.

Kitagawa, G., A self-organizing state-space model, *Journal of the American Statistical Association*, *93*, 1203–1215, 1998.

Krebs, C. J., R. Boonstra, S. Boutin, and A. Sinclair, What drives the 10-year cycles of snowshoe hares?, *American Institute of Biological Sciences*, *51*, 25–35, 2001.

Lotka, A. J., Undamped oscillations derived from the law of mass action, *Journal of the American Chemical Society*, *42*, 1595–1599, 1920.

MacEachern, S. N., M. Clyde, and J. S. Liu, Sequential importance sampling for nonparametric bayes models: The next generation, *The Canadian Journal of Statistics*, *27*, 251–267, 1999.

May, R. M., *Stability and Complexity in Model Ecosystems*, Princeton University Press, 1973.

Murray, J. D., *Mathematical Biology*, Springer-Verlag, 1989.

Renshaw, E., *Modelling Biological Populations in Space and Time*, Cambridge University Press, 1993.

Ristic, B., S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*, Artech House Publishers, 2004.

Volterra, V., Fluctuations in the abundance of a species considered mathematically, *Nature*, *118*, 558–560, 1926.

Zhang, Z., Y. Tao, and Z. Li, Factors affecting hare-lynx dynamics in the classic time series of the hudson bay company, canada, *Climate Research*, *34*, 83–89, 2007.

# Appendix A

# Additional Figures

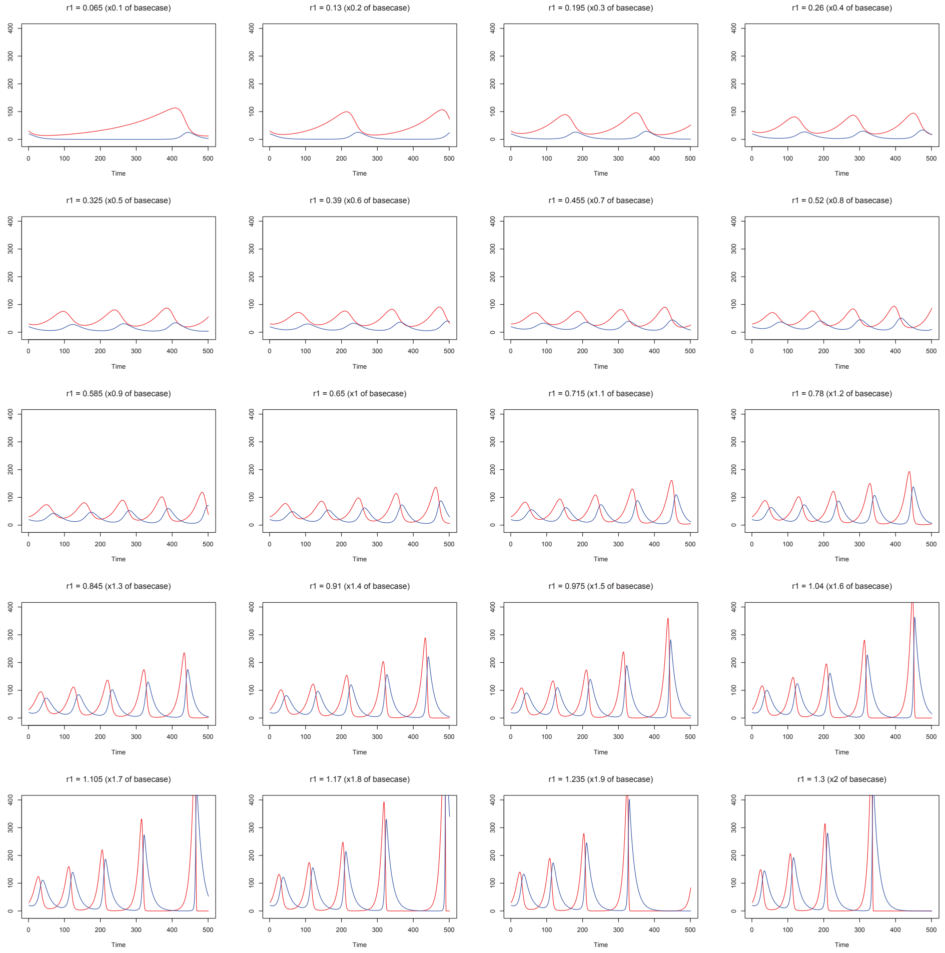


Figure A.1: Plots of the predator-prey model using parameter values $b_1 = 0.023$, $r_2 = 0.65$ and $b_2 = 0.014$ with the value of $r_1$ changing as a percentage of the base case value. Note that the base case value of $r_1 = 0.65$.

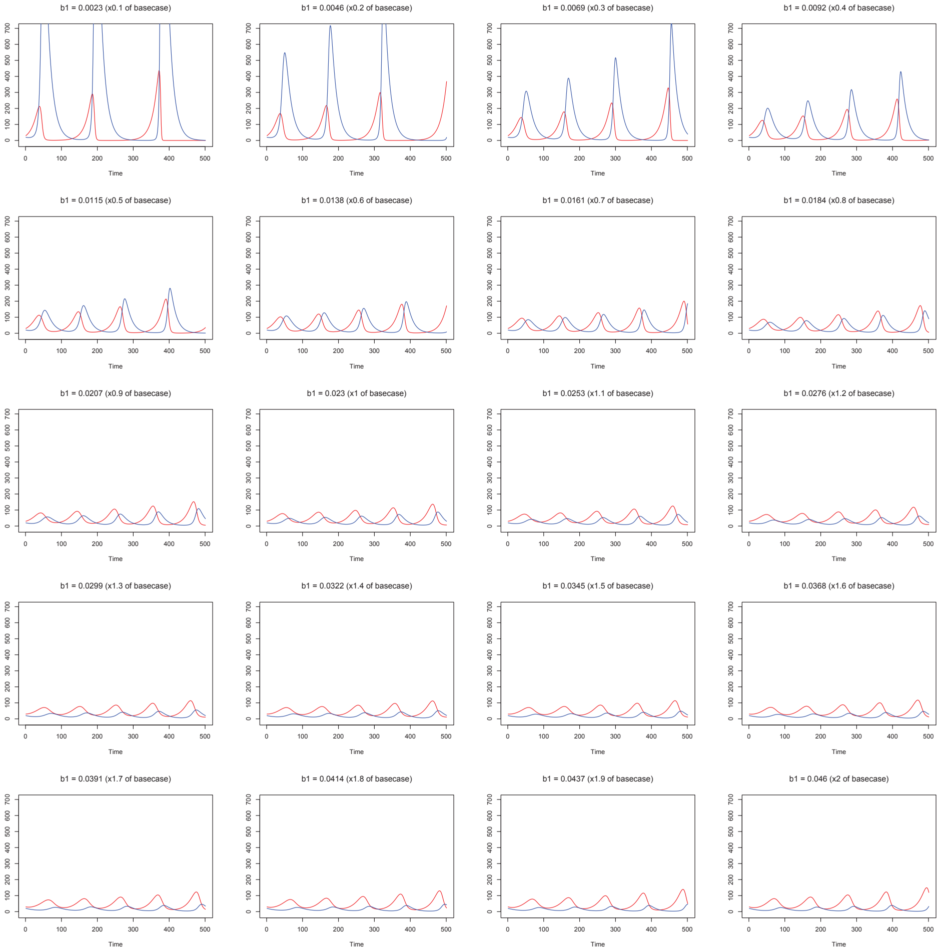


Figure A.2: Plots of the predator-prey model using parameter values $r_1$ = 0.65, $r_2$ = 0.65 and $b_2$ = 0.014 with the value of $b_1$ changing as a percentage of the base case value. Note that the base case value of $b_1$ = 0.023.

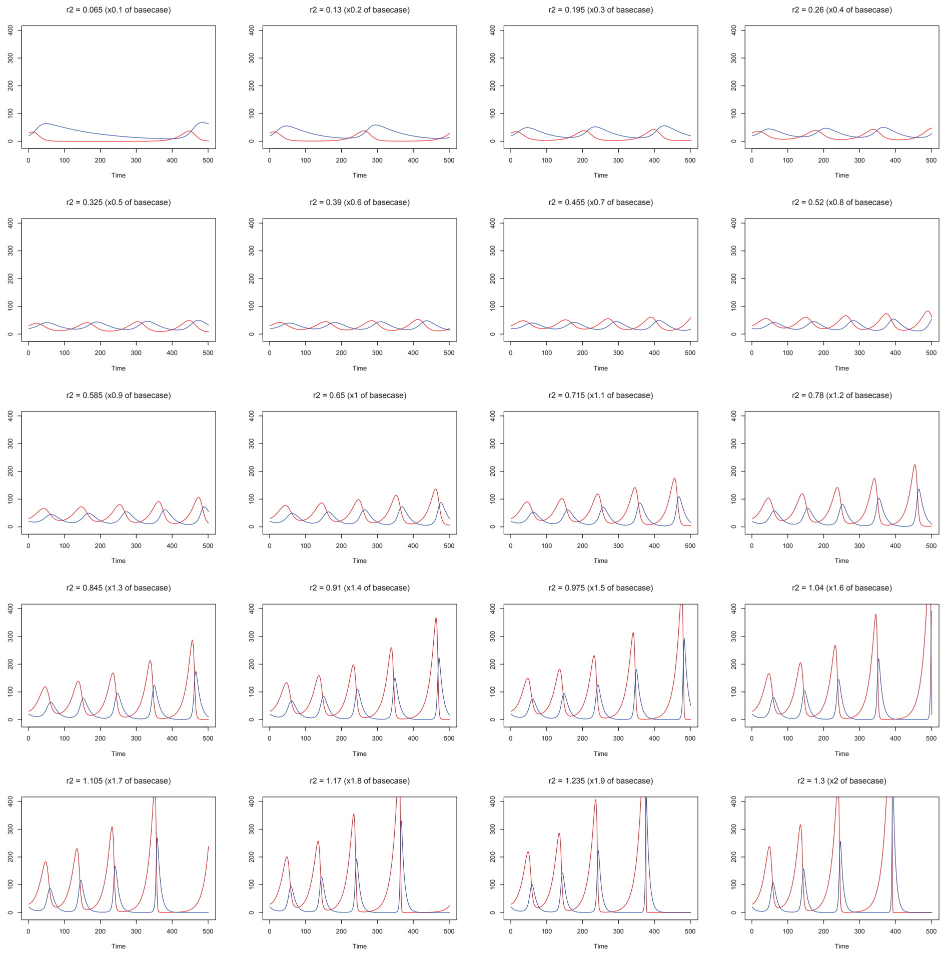
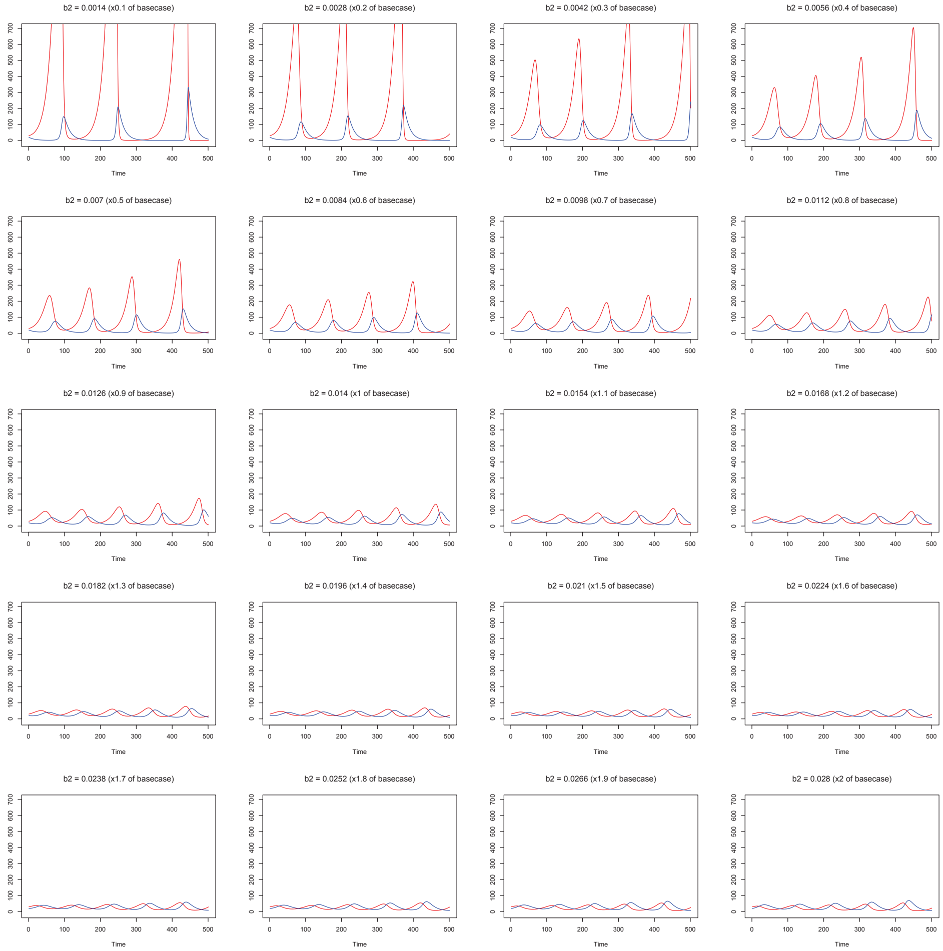


Figure A.3: Plots of the predator-prey model using parameter values $r_1 = 0.65$, $b_1 = 0.023$ and $b_2 = 0.014$ with the value of $r_2$ changing as a percentage of the base case value. Note that the base case value of $r_2 = 0.65$.



b2 = 0.0014 (x0.1 of basecase)

b2 = 0.0028 (x0.2 of basecase)

b2 = 0.0042 (x0.3 of basecase)

b2 = 0.0056 (x0.4 of basecase)

b2 = 0.007 (x0.5 of basecase)

b2 = 0.0084 (x0.6 of basecase)

b2 = 0.0098 (x0.7 of basecase)

b2 = 0.0112 (x0.8 of basecase)

b2 = 0.0126 (x0.9 of basecase)

b2 = 0.014 (x1 of basecase)

b2 = 0.0154 (x1.1 of basecase)

b2 = 0.0168 (x1.2 of basecase)

b2 = 0.0182 (x1.3 of basecase)

b2 = 0.0196 (x1.4 of basecase)

b2 = 0.021 (x1.5 of basecase)

b2 = 0.0224 (x1.6 of basecase)

b2 = 0.0238 (x1.7 of basecase)

b2 = 0.0252 (x1.8 of basecase)

b2 = 0.0266 (x1.9 of basecase)

b2 = 0.028 (x2 of basecase)

Figure A.4: Plots of the predator-prey model using parameter values $r_1$ = 0.65, $b_1$ = 0.023 and $r_2$ = 0.65 with the value of $b_2$ changing as a percentage of the base case value. Note that the base case value of $b_2$ = 0.014.

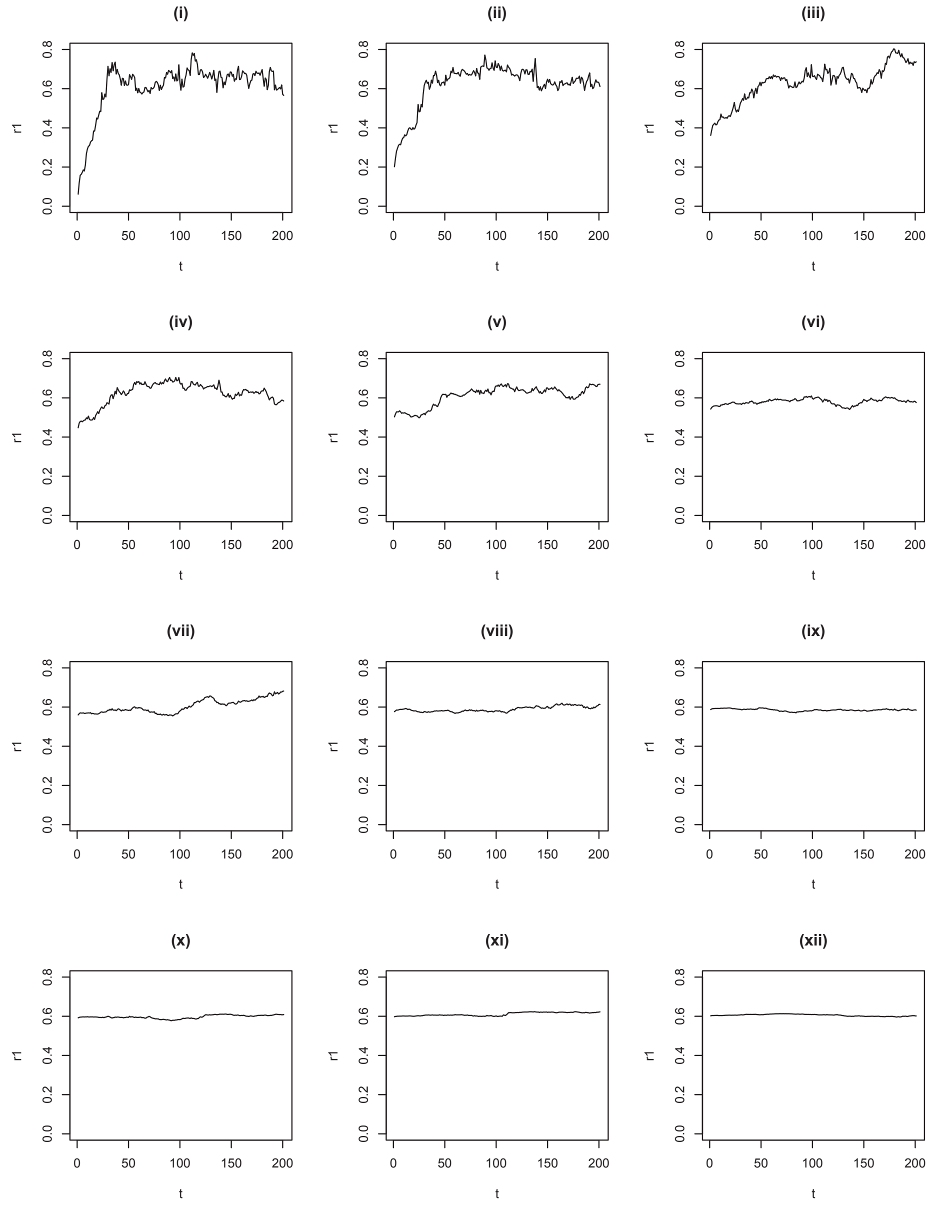


Figure A.5: Plots of the parameter $r_1$ in the predator-prey model using the synthetic dataset; estimated via multiple iterative state augmented particle filtering. Shown are the traces of the parameter $r_1$ generated in 12 iterations. The true value of $r_1 = 0.65$. Note: $r_1$ is the only parameter varying and the remaining three parameters are fixed.

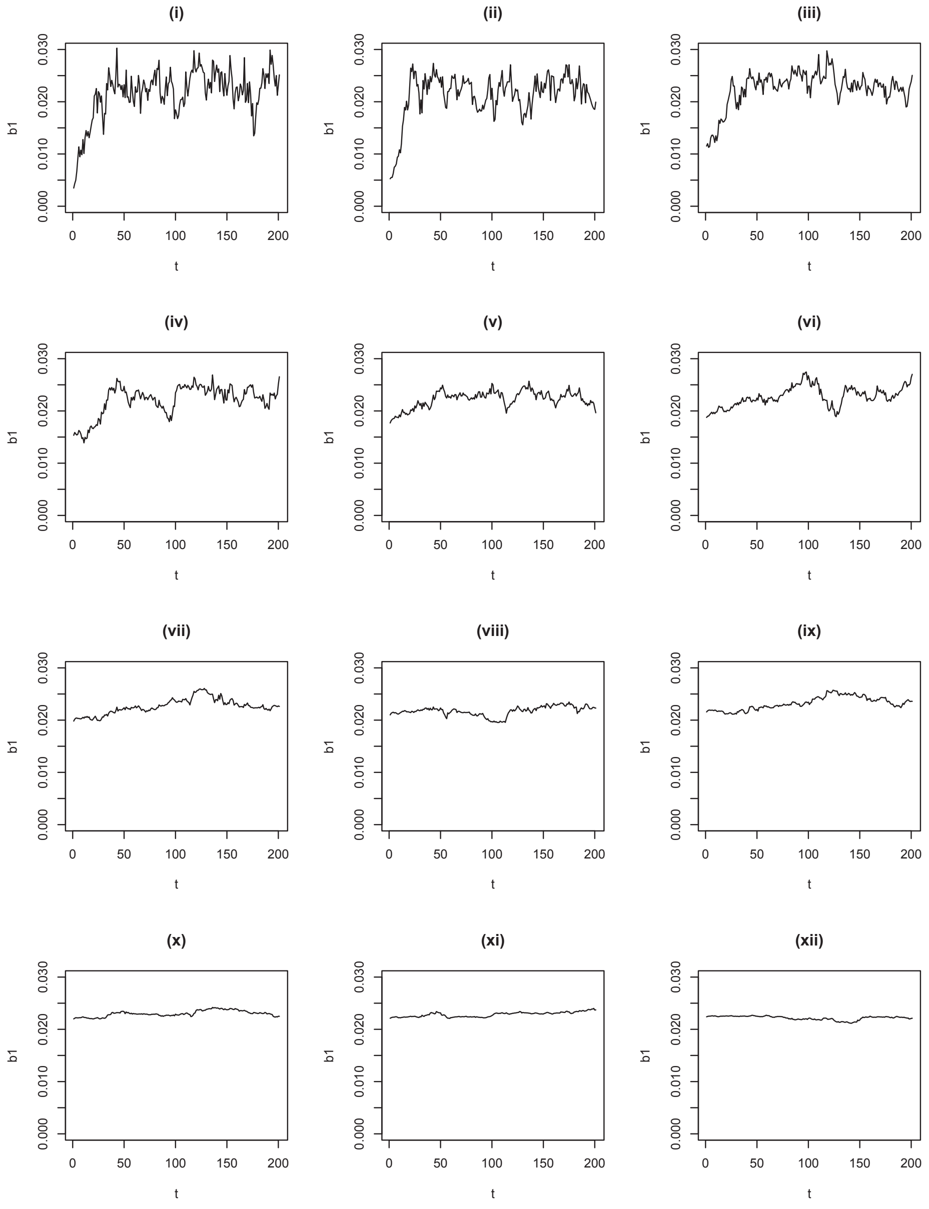


Figure A.6: Plots of the parameter $b_1$ in the predator-prey model using the synthetic dataset; estimated via multiple iterative state augmented particle filtering. Shown are the traces of the parameter $b_1$ generated in 12 iterations. The true value of $b_1 = 0.023$. Note: $b_1$ is the only parameter varying and the remaining three parameters are fixed.

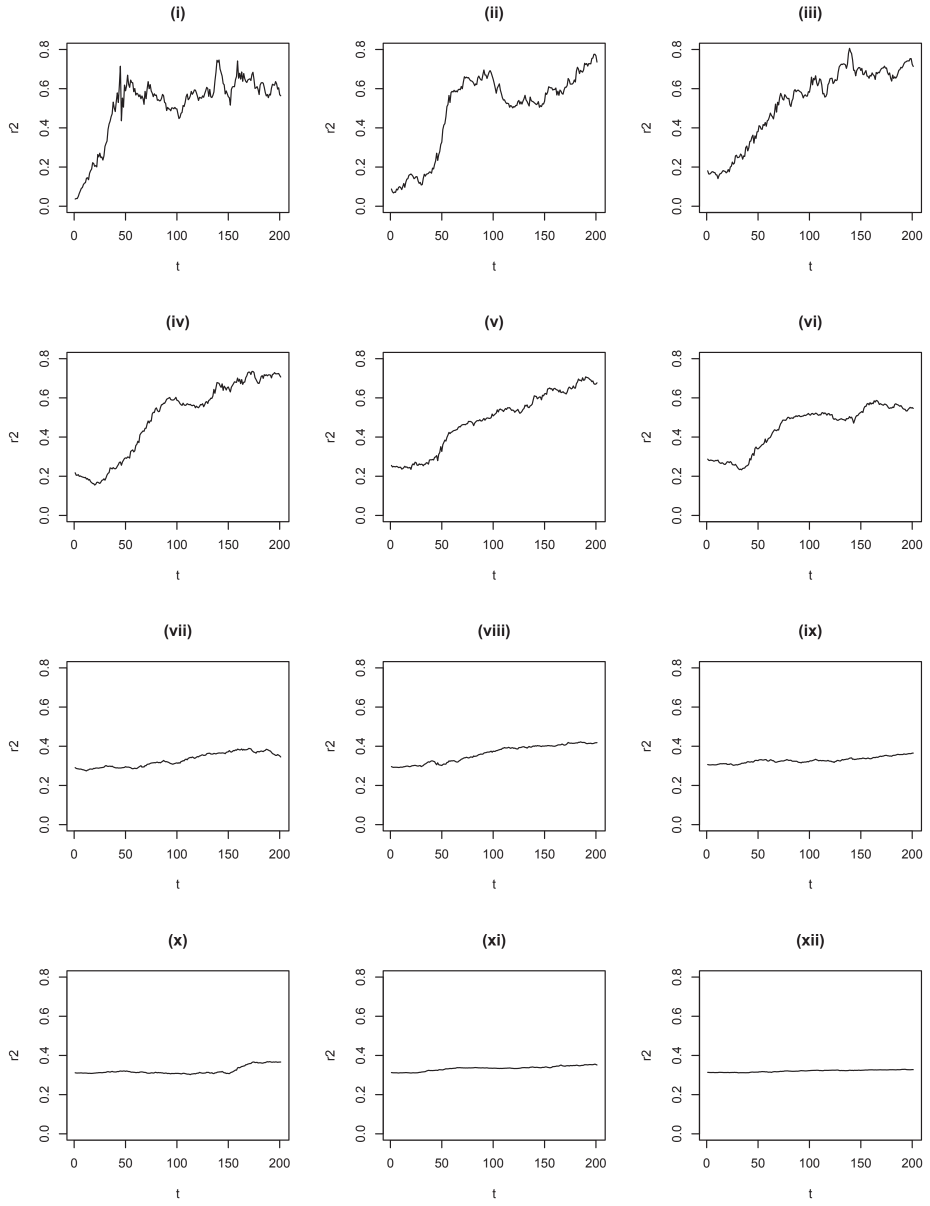


Figure A.7: Plots of the parameter $r_2$ in the predator-prey model using the synthetic dataset; estimated via multiple iterative state augmented particle filtering. Shown are the traces of the parameter $r_2$ generated in 12 iterations. The true value of $r_2 = 0.65$. Note: $r_2$ is the only parameter varying and the remaining three parameters are fixed.

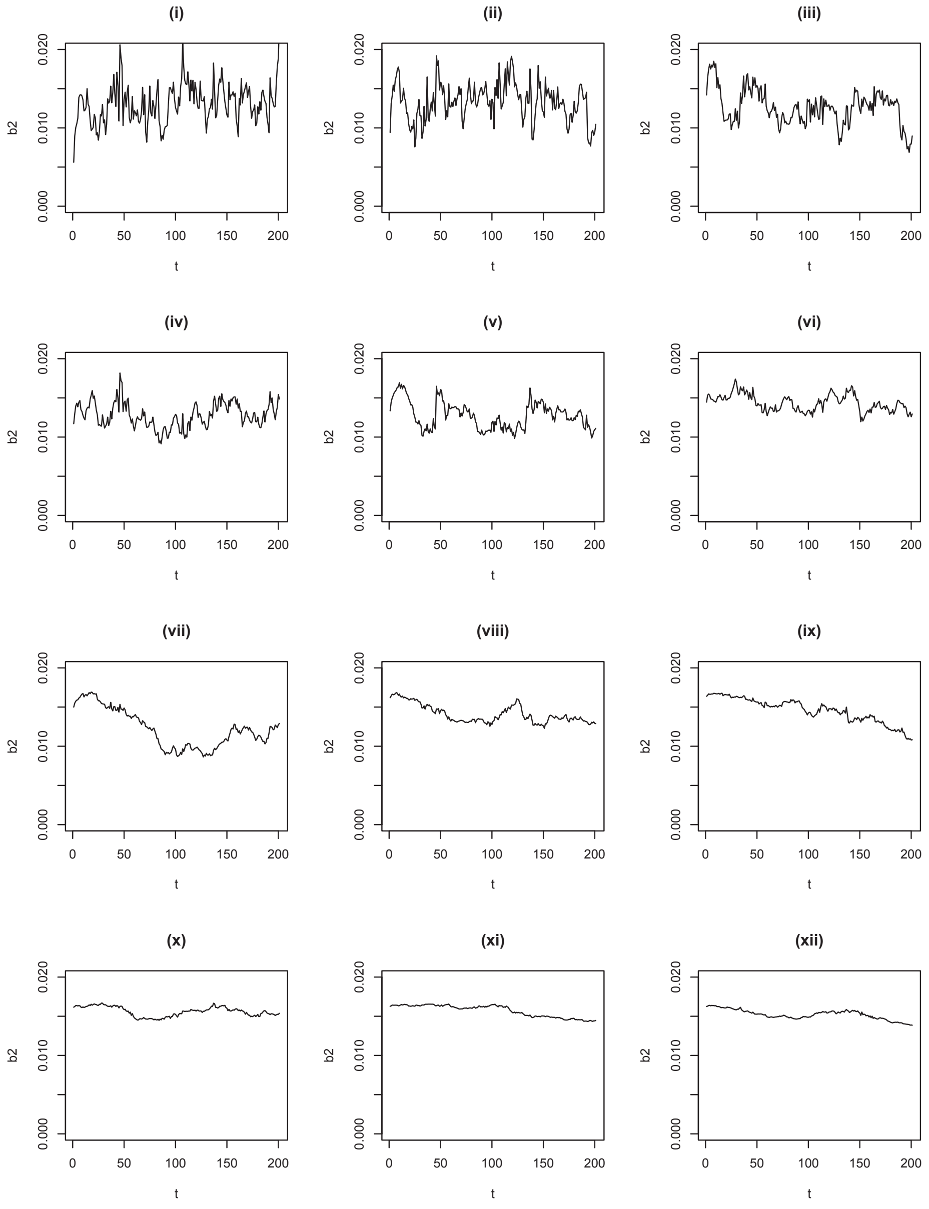


Figure A.8: Plots of the parameter $b_2$ in the predator-prey model using the synthetic dataset; estimated via multiple iterative state augmented particle filtering. Shown are the traces of the parameter $b_2$ generated in 12 iterations. The true value of $b_2 = 0.014$. Note: $b_2$ is the only parameter varying and the remaining three parameters are fixed.

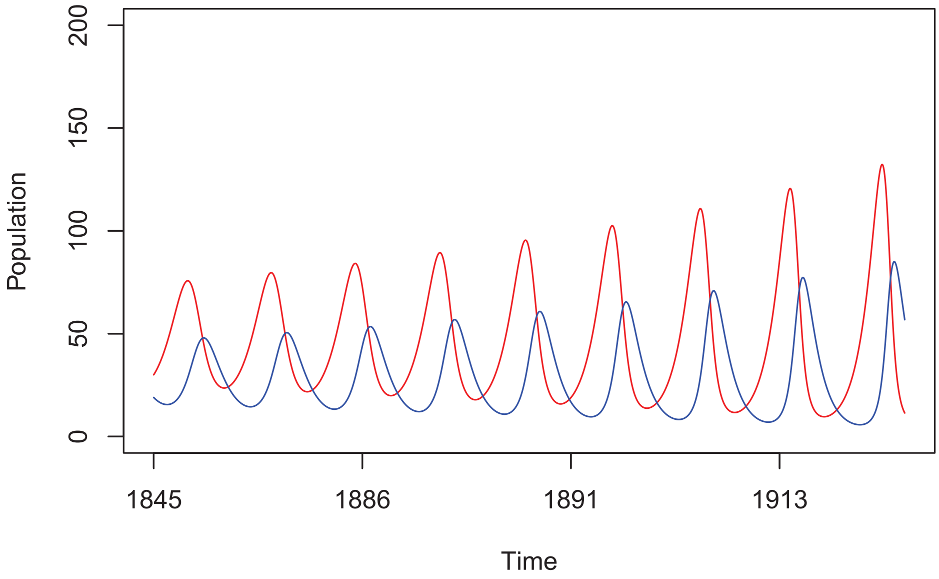


Figure A.9: Time series plot of lynx-hare population (in thousands).

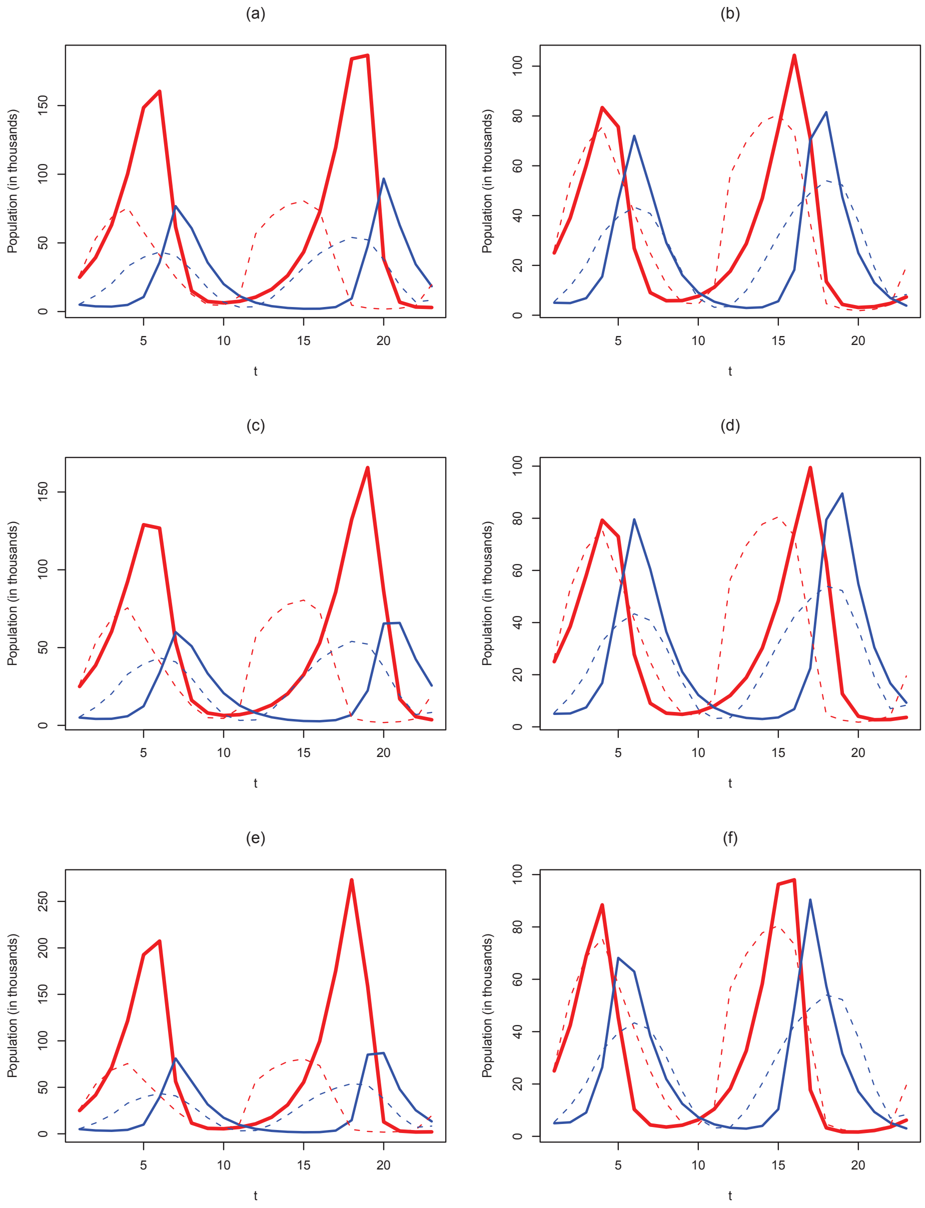


Figure A.10: Time series plots of the observed and estimated segment of the lynx-hare dataset (from 1909-1931) based on the MLEs. Shown are the observed populations of the lynx (dashed blue line) and the hare (dashed red line) and the estimated populations of the lynx (thick blue line) and the hare (thick red line).

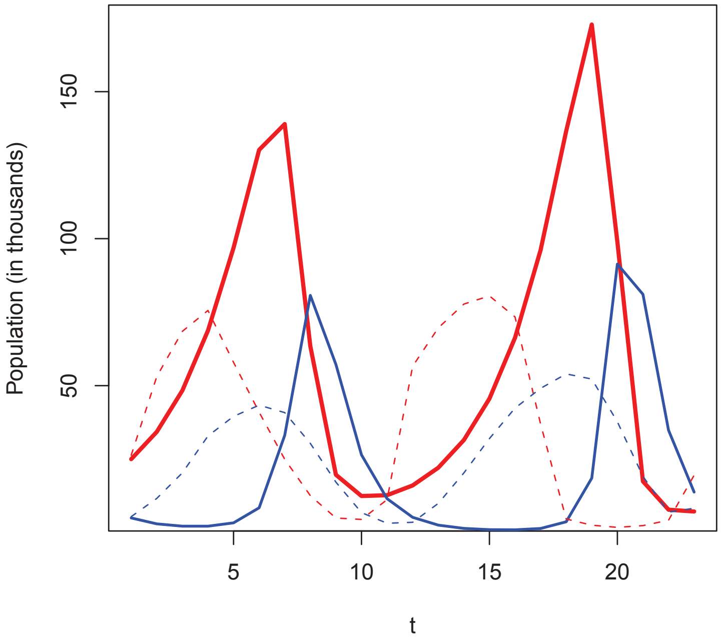


Figure A.11: Time series plot of the observed and estimated subsection of the lynx-hare dataset (from 1909-1931) based on the estimates via state augmentation. Shown are the observed populations of the lynx (dashed blue line) and the hare (dashed red line) and the estimated populations of the lynx (thick blue line) and the hare (thick red line).

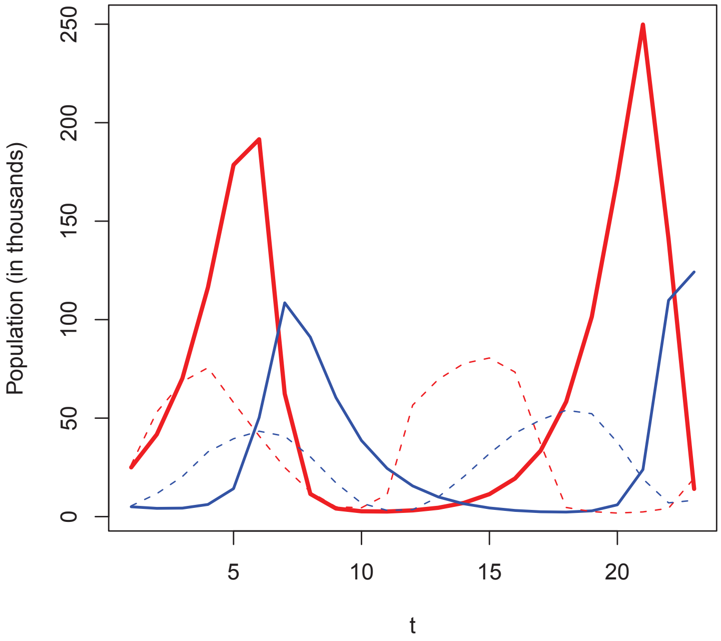


Figure A.12: Time series plot of the observed and estimated subsection of the lynx-hare dataset (from 1909-1931) based on the estimates using MIF. Shown are the observed populations of the lynx (dashed blue line) and the hare (dashed red line) and the estimated populations of the lynx (thick blue line) and the hare (thick red line).

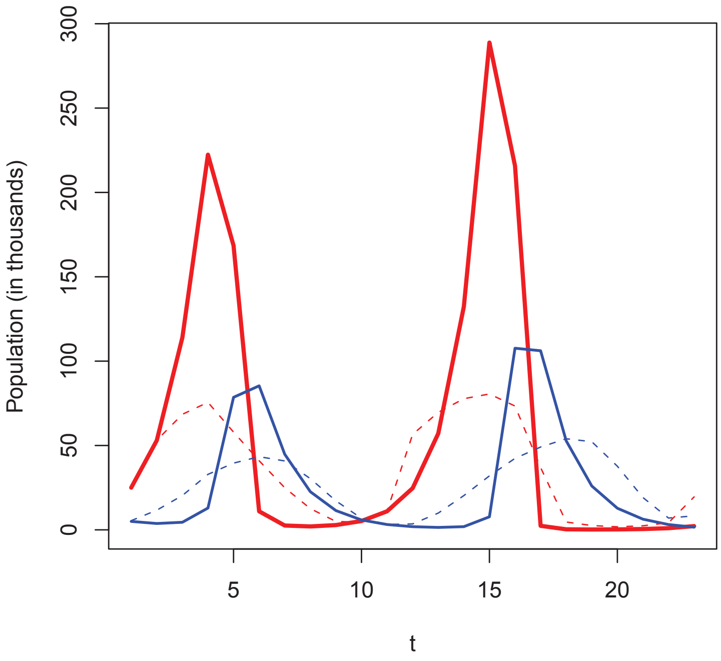


Figure A.13: Time series plot of the observed and estimated subsection of the lynx-hare dataset (from 1909-1931) based on the estimates using the MH-PF-MH algorithm. Shown are the observed populations of the lynx (dashed blue line) and the hare (dashed red line) and the estimated populations of the lynx (thick blue line) and the hare (thick red line).

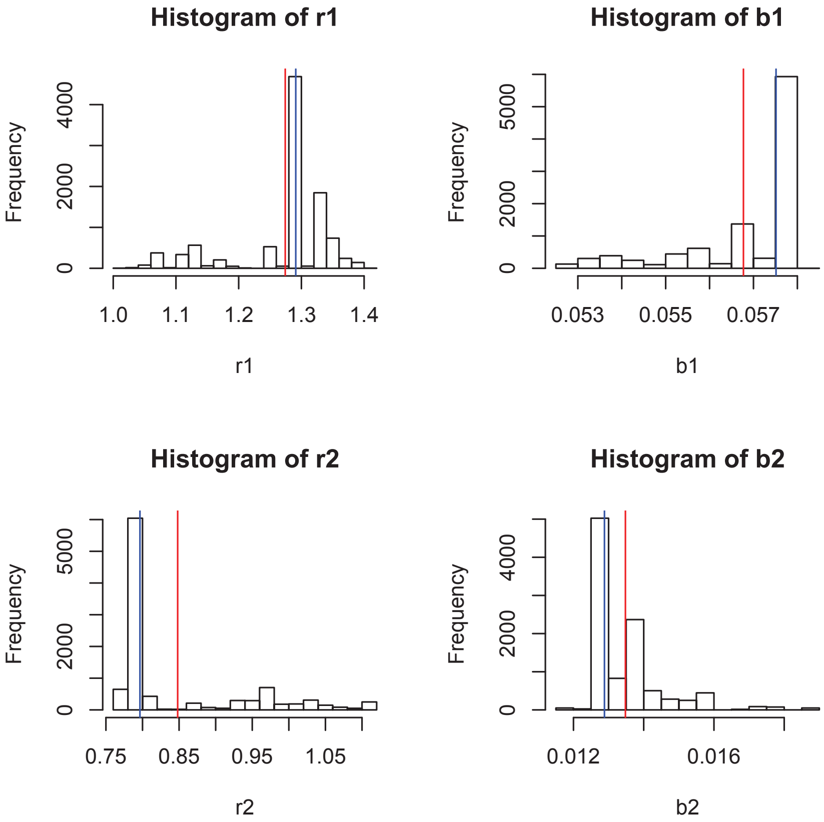


Figure A.14: Histograms of the dynamical parameters from the predator prey model, running the MHPFMH algorithm for 10,000 iterations. The segment of the lynx-hare dataset from 1845-1864 is used. Shown are the estimated values of the parameters: (a) $r_1$, (b) $b_1$, (c) $r_2$ and (d) $b_2$. Note, the red vertical lines represent the mean of the distributions and the blue vertical lines represent the median of the distributions.

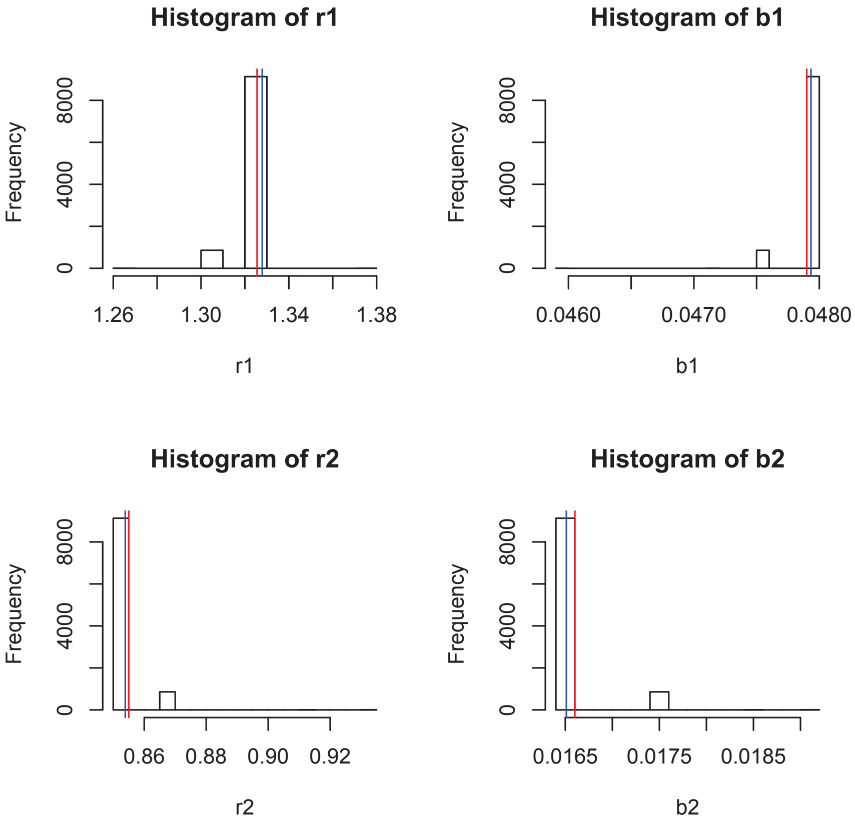


Figure A.15: Histograms of the dynamical parameters from the predator prey model, running the MHPFMH algorithm for 10,000 iterations. The segment of the lynx-hare dataset from 1855-1874 is used. Shown are the estimated values of the parameters: (a) $r_1$, (b) $b_1$, (c) $r_2$ and (d) $b_2$. Note, the red vertical lines represent the mean of the distributions and the blue vertical lines represent the median of the distributions.

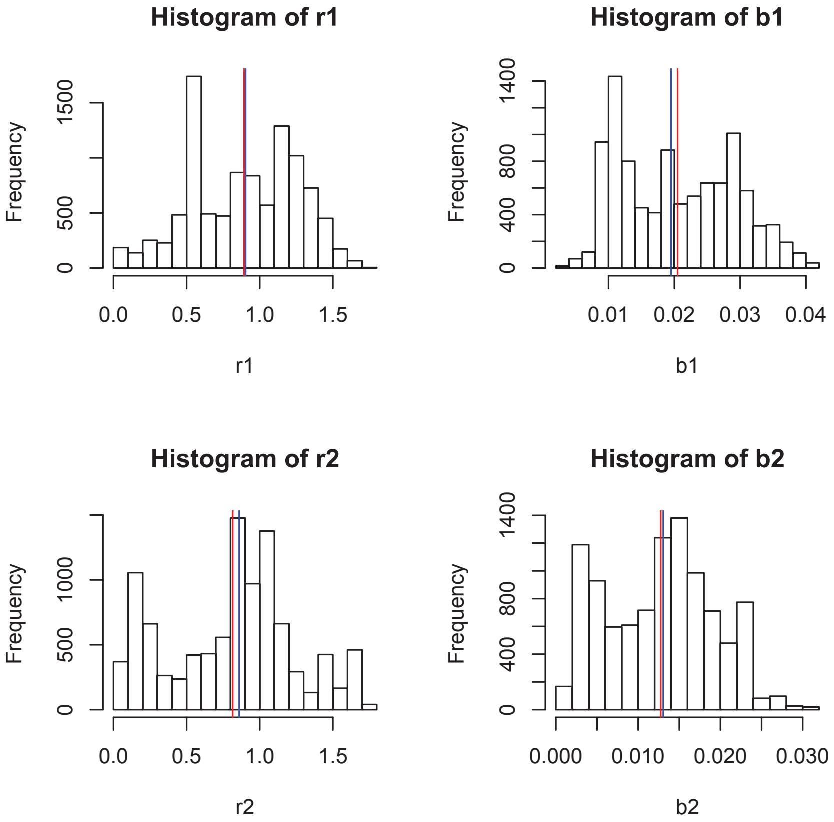


Figure A.16: Histograms of the dynamical parameters from the predator prey model, running the MHPFMH algorithm for 10,000 iterations. The segment of the lynx-hare dataset from 1865-1884 is used. Shown are the estimated values of the parameters: (a) $r_1$, (b) $b_1$, (c) $r_2$ and (d) $b_2$. Note, the red vertical lines represent the mean of the distributions and the blue vertical lines represent the median of the distributions.

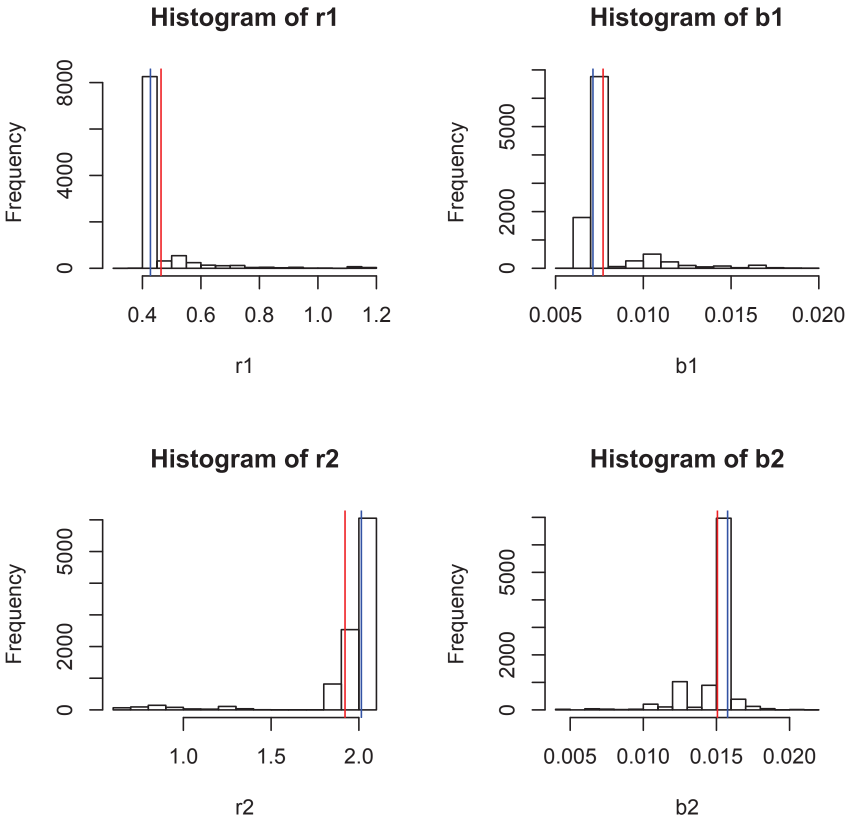


Figure A.17: Histograms of the dynamical parameters from the predator prey model, running the MHPFMH algorithm for 10,000 iterations. The segment of the lynx-hare dataset from 1875-1894 is used. Shown are the estimated values of the parameters: (a) $r_1$, (b) $b_1$, (c) $r_2$ and (d) $b_2$. Note, the red vertical lines represent the mean of the distributions and the blue vertical lines represent the median of the distributions.

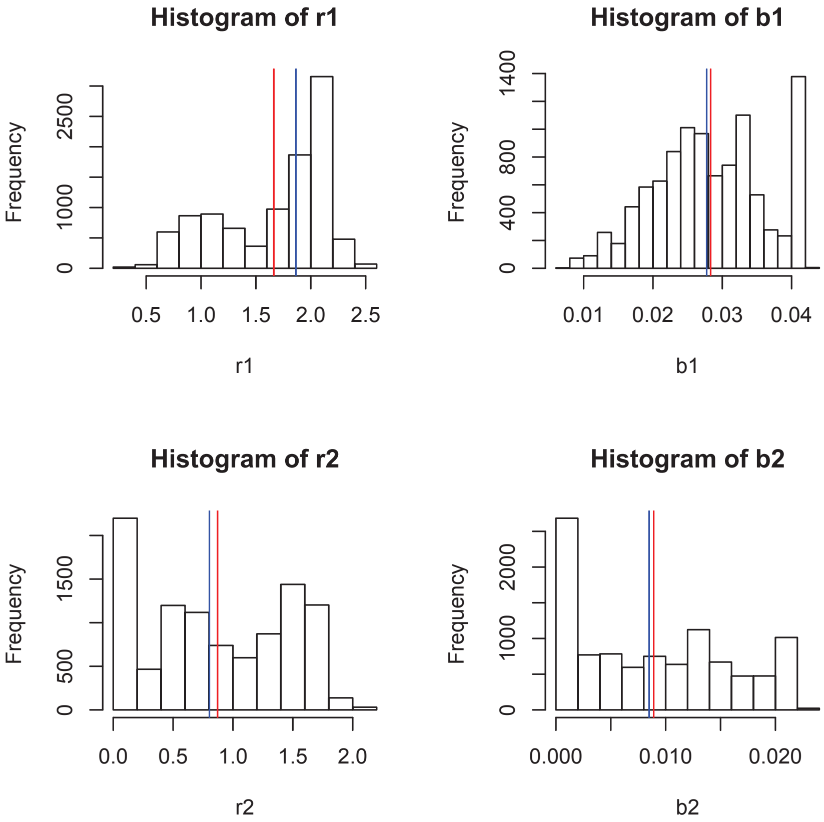


Figure A.18: Histograms of the dynamical parameters from the predator prey model, running the MHPFMH algorithm for 10,000 iterations. The segment of the lynx-hare dataset from 1885-1904 is used. Shown are the estimated values of the parameters: (a) $r_1$, (b) $b_1$, (c) $r_2$ and (d) $b_2$. Note, the red vertical lines represent the mean of the distributions and the blue vertical lines represent the median of the distributions.

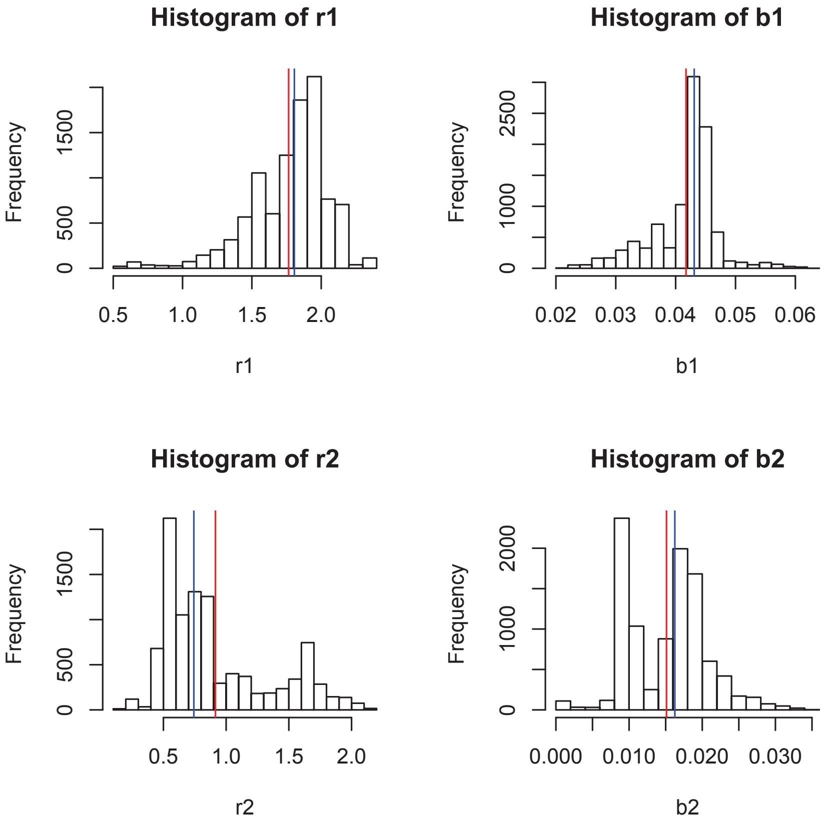


Figure A.19: Histograms of the dynamical parameters from the predator prey model, running the MHPFMH algorithm for 10,000 iterations. The segment of the lynx-hare dataset from 1895-1914 is used. Shown are the estimated values of the parameters: (a) $r_1$, (b) $b_1$, (c) $r_2$ and (d) $b_2$. Note, the red vertical lines represent the mean of the distributions and the blue vertical lines represent the median of the distributions.

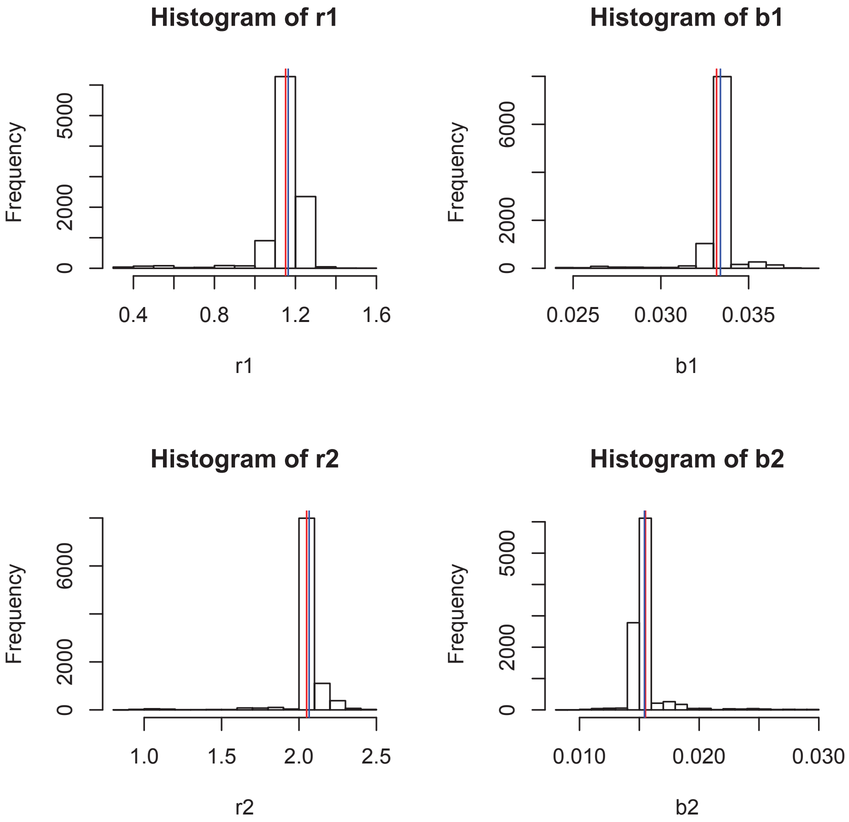


Figure A.20: Histograms of the dynamical parameters from the predator prey model, running the MHPFMH algorithm for 10,000 iterations. The segment of the lynx-hare dataset from 1905-1924 is used. Shown are the estimated values of the parameters: (a) $r_1$, (b) $b_1$, (c) $r_2$ and (d) $b_2$. Note, the red vertical lines represent the mean of the distributions and the blue vertical lines represent the median of the distributions.

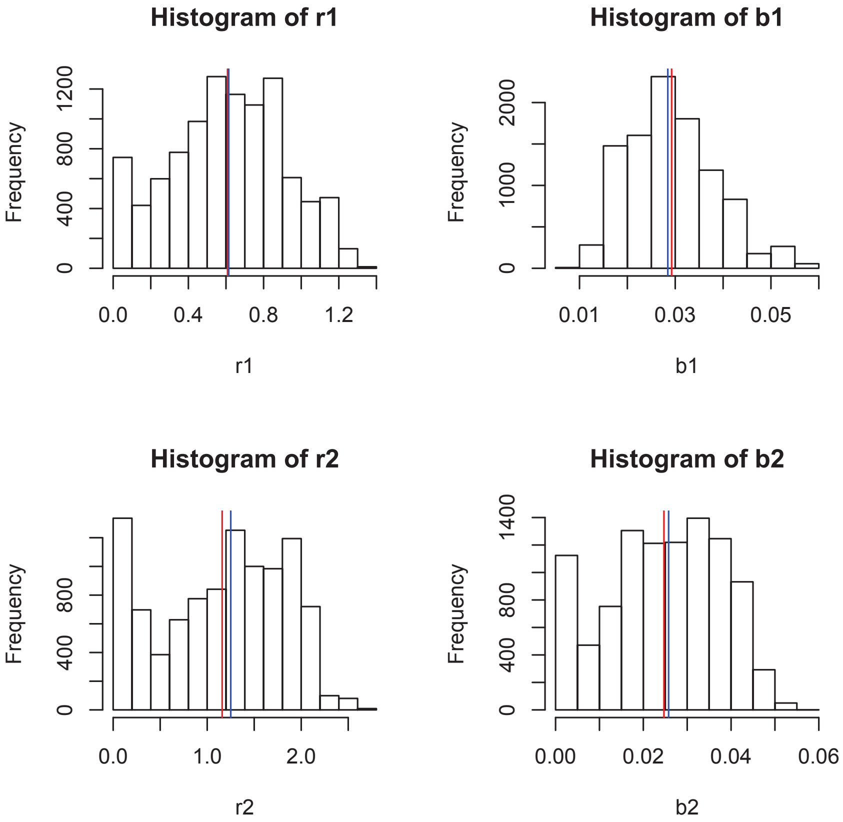


Figure A.21: Histograms of the dynamical parameters from the predator prey model, running the MHPFMH algorithm for 10,000 iterations. The segment of the lynx-hare dataset from 1915-1934 is used. Shown are the estimated values of the parameters: (a) $r_1$, (b) $b_1$, (c) $r_2$ and (d) $b_2$. Note, the red vertical lines represent the mean of the distributions and the blue vertical lines represent the median of the distributions.