Bayesian Inference for Stochastic Population Models

MMathStat Project

School of Mathematics & Statistics
Newcastle University

Nina Wilkinson

April 28, 2010

Mustard aphids are the most serious pest on the Brassica oilseeds in India which makes the study of mustard aphid population dynamics a popular area to research. We first use a classic deterministic approach to show that a simple birth-death model is not appropriate for the first four weeks of some mustard aphid data sets. Next we introduce two stochastic models which have immigration only during the first four weeks. In other papers the parameters in these models are estimated using a frequentist framework, where as we will use a Bayesian framework. As the parameter posterior is analytically intractable we need to use computationally intensive techniques in order to sample from it. In this thesis we use Importance Sampling and to avoid small weights in the resampling step we will implement the algorithm sequentially. Initially, we will consider simulated data to estimate the parameters, before outlining how this method can be applied to real data sets.

Contents

1. Introduction

1.1. Background

Aphids are a group of small plant-eating insects which are one of the most destructive insect pests of agricultural crops around the world. There is a huge worldwide economic loss on food and feed grains every year due to aphids of around $5 billion. This makes modelling aphid dynamics an important area to study. In this project we will focus on the mustard aphid which is the most serious pest on the Brassica oilseeds in India as they cause a 30-50% loss in seed yields. [Matis et al., 2007]

1.2. Previous Modelling Attempts

Fitting models to aphid data sets is a topic that has been researched extensively. Two key papers are Matis et al. [2007] and Matis et al. [2008]. In Matis et al. [2007] they discuss models that depend on both the current and the cumulative size of a population. They argue that local aphid populations become extinct due to earlier generations of aphids using up some dwindling resources including food and habitat resources. Furthermore, the larger the population of aphids the more predators are attracted, plant and aphid disease are more likely and there is an increase in the chemical reaction that causes the plant to lose sap. Therefore there is a cumulative size dependency and a model in which the death rate term depends on the cumulative size of the population is appropriate.

This model is shown in Matis et al. [2007] to fit aphid abundance curves which vary in shape and size. However for the mustard aphid data, there is a linear pattern occurring in the first few weeks, which the model fails to capture.

Matis et al. [2008] considers adding an immigration term into the previous model. They refer to immigration being the increase in a population which occurs independently of the current population size. Immigration is introduced at a time independent rate; say α which means its cumulative effects would be to increase the population size by a simple linear function αt. This is different to the birth rate and death rate terms which depend on the population size.

Matis et al. [2008] discuss their case for adding immigration to the model and the main points are as follows:

- Local populations of aphids are generally initiated when aphids migrate from other areas. The migration occurs due to the depletion of local resources at their original environment.
- Aphids remove sap from plants and secrete honeydew. This leads to a spread of fungi which means the plant is damaged and the aphids can no longer eat the plants. This is why we incorporate cumulative population size explicitly in our models.
- Aphid reproduction is dependent on temperature and aphid reproduction does not occur or is very low below certain temperatures, which fits in with the mustard aphid data due to the first four weeks in the study temperatures were not high enough for full reproduction to occur.

Matis et al. [2008] then discusses two immigration models which we will introduce in Chapter 2. These two models both have immigration only for around the first four weeks to cater for the linear pattern present during this time. In one model, immigration then stops and you have birth and death only occurring after this time and in the other model the immigration continues along with birth and death.

From the analysis of Matis et al. [2008], the authors show that these models both fit the data equally well, both models fit a very similar shaped curve to the data and that consideration into the likely length of immigration is important to decide between models.

Matis et al. [2008] use a frequentist framework to estimate their parameters for the birth, death and immigration rate as well as the time that the immigration only period stops by using non-linear least squares in conjunction with the ordinary differential equations used to represent the models. However, in this project we will use a Bayesian approach which allows incorporation of prior beliefs into the analysis. Moreover, for the model parameters considered here, it may be possible to elicit expert opinion.

The overall aim of the project is to develop and fit a stochastic model to some aphid data sets which depends on the cumulative size of the population. We will use a stochastic approach due to the fact that error can occur in the collection of the data and a stochastic model caters for that when there is a small sample size the population could become extinct. We will be using data for populations of mustard aphids on the Indian mustard flower.

1.3. The Data

The data sets we will be using are J8 and J9 from Matis et al. [2007], where J stands for the variety of aphid which is Brassica Juncea (the mustard aphid)

Week Number	Aphid Density in 1988 (J8)	Aphid Density in 1989 (J9)
1	13	0.4
2	22.5	8.2
3	29	18.5
4	33.5	30.9
5	55.5	111.5
6	89	540.7
7	100.5	755
8	57	346.7
9	24	1.9
10	1	1.9

Table 1.1.: Mustard aphid density for 10 weeks in 1988 and 1989.

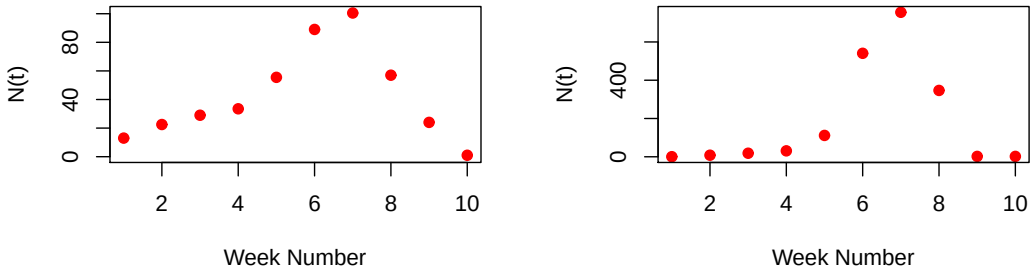

Figure 1.1.: Left panel: Plot of J8 data. Right panel: Plot of J9 data.

and the number stands for the year. The data was collected in 1988 and 1989 at Haryana Agricultural University where they counted weekly the number of aphids on the terminal 15cm of the central mainstem on 30 plants per data set. This was done to determine the aphid density. The data we will be using is shown in Table 1.1 and it is plotted in Figure 1.1.

2. The Model

2.1. Introduction

The first model we are going to consider for the mustard aphid data is a modified birth-death model, where the death rate depends on the cumulative size of the population. Let λ denote the birth rate, $N(t)$ the number of aphids at time t and $C(t)$ the number of aphids that have ever been born until time t. Representing the process as a pseudo biochemical reaction and dropping the arguments of $N(t)$ and $C(t)$ we have,

$$N \xrightarrow{\lambda} 2N + C. \tag{2.1}$$

For each birth the population increases by one, therefore N and C both increase by one. Let μ be the death rate. In terms of pseudo biochemical reactions we have,

$$N + C \xrightarrow{\mu} C. \tag{2.2}$$

For each death, N decreases by one and C remains unchanged.

2.2. Deterministic Approach

By assuming that $N(t)$ and $C(t)$ vary continuously with time, and assuming mass action kinetic rate laws associated with Equations (2.1) and (2.2) we can formulate a deterministic model. A deterministic model is a model that produces a solution that occurs in a non-random manner. This means that given a set of initial conditions the outcome is fixed. If we take a deterministic approach we can describe $N(t)$ and $C(t)$ over time by the set of differential equations,

$$\frac{dN(t)}{dt} = \lambda N(t) - \mu C(t)N(t) \quad \text{and} \quad \frac{dC(t)}{dt} = \lambda N(t) \tag{2.3}$$

where,

- λ is the birth rate;
- μ is the death rate;

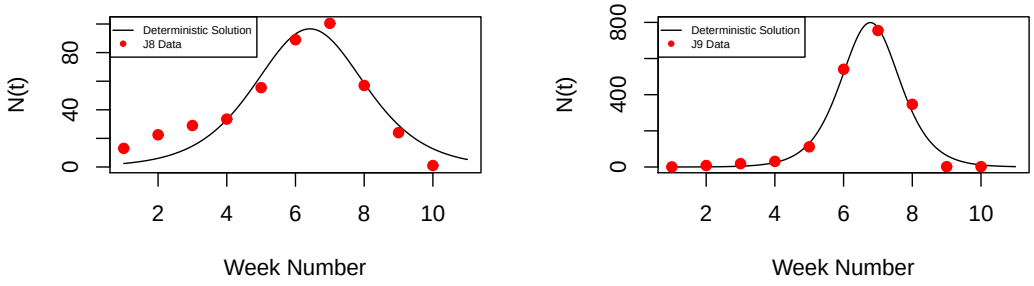

Figure 2.1.: Plot of the deterministic solution against the data for the a) J8 data and b) J9 data.

- t is the time in weeks;
- $N(t)$ is the number of aphids at time t;
- $C(t)$ is the number of aphids that have ever been born at time t therefore $C(t) \geq N(t) \forall t$.

Let N_{max} be the maximum of $N(t)$ and t_{max} be the time when N_{max} occurs. We can now solve Equations (2.3) to get

$$N(t) = 4N_{max}e^{-b(t-t_{max})}(1 + e^{-b(t-t_{max})})^{-2}. \tag{2.4}$$

In equation (2.4), $b \approx \mu$ and t_{max} and N_{max} are measurable quantities which are functions of λ and μ and can be obtained from Matis et al. [2007]. If we fit the deterministic solution (2.4) to the J8 and J9 data sets using the values given in Matis et al. [2007] we obtain Figure 2.1.

From Figure 2.1 we observed that the deterministic model is generally a good fit for both sets of data. We notice that birth and death rates depend on population size and they cause an increase or decrease by an exponential function. We shall consider adding an immigration rate term into the model which refers to the increase in a population that occurs independently of current population size. This immigration term will cause the population size to increase by the simple linear function αt, where α is the immigration rate. We can see that there is a linear pattern occurring in both data sets for the first 4 weeks which is shown in Figure 2.1. The deterministic model is not a good fit during this time. In order to

cater for the linear pattern during the first four weeks, we will consider models with immigration only during this time. This is backed by research as it is shown that for many aphid species, local populations are initiated by the migration of winged aphids from other areas and that aphid reproduction is temperature dependent and it is virtually or totally absent below certain threshold temperatures [Matis et al., 2008]. Therefore for the data we have that the temperatures were too cold during the first four weeks for aphid reproduction and it is sensible to consider adding an immigration component to our model.

Let α denote the immigration. Representing this as a biochemical reaction we get,

$$\emptyset \xrightarrow{\alpha} N + C. \tag{2.5}$$

Therefore immigration results in an increase by N and C by one irrespective of the current population size.

The deterministic model including immigration is,

$$\frac{dN(t)}{dt} = \lambda N(t) - \mu C(t)N(t) + \alpha \quad \text{and} \quad \frac{dC(t)}{dt} = \lambda N(t) + \alpha \tag{2.6}$$

where,

- λ is the birth rate;
- μ is the death rate;
- α is the immigration rate;
- t is the time in weeks;
- $N(t)$ is the number of aphids at time t.
- $C(t)$ is the number of aphids that have ever been born at time t therefore $C(t) \geq N(t) \forall t$.

However the model shown in Equation (2.6) will also not be a very good fit up until around week four due to the linear pattern occurring during this time. In order to cater for the linear pattern which occurs in the first four weeks of the data we can introduce the parameter τ which represents the time when the immigration only period ends. We know that τ will be about 4 from inspection of our data. This then means we have the models shown in Equations (2.7) and (2.8) to consider.

Our first model has immigration only until time τ and then from $t = \tau$ onwards there is no immigration present, but only births and deaths.

$$\frac{dN(t)}{dt} = \begin{cases} \alpha & \text{if } t < \tau, \\ (\lambda - \mu C(t))N(t) & \text{if } t \geq \tau, \end{cases} \quad \text{and} \quad \frac{dC(t)}{dt} = \begin{cases} \alpha & \text{if } t < \tau, \\ \lambda N(t) & \text{if } t \geq \tau. \end{cases} \tag{2.7}$$

Another model is where we again have immigration only until time τ, however from this time onwards immigration continues, as well as births and deaths.

$$\frac{dN(t)}{dt} = \begin{cases} \alpha & \text{if } t < \tau, \\ \alpha + (\lambda - \mu C(t))N(t) & \text{if } t \geq \tau, \end{cases} \quad \text{and} \quad \frac{dC(t)}{dt} = \begin{cases} \alpha & \text{if } t < \tau, \\ \alpha + \lambda N(t) & \text{if } t \geq \tau. \end{cases} \tag{2.8}$$

We now consider a stochastic treatment of the problem. This is the focus of the next section.

2.3. Stochastic Approach

A stochastic model is one where the next state of the model is determined randomly by a probability distribution which means that given the initial conditions there are many possible realisations.

We are going to use a stochastic model since:

- A stochastic model caters for the random variation that occurs in nature. If we did an experiment in nature where we are counting the number of aphids on a leaf, every time we did the experiment we could obtain different counts.
- A stochastic model gives us an idea of the variability that can occur because if you use a stochastic model you can obtain many realisations from a model with certain initial conditions and parameters.
- A stochastic model is more appropriate for small populations. A deterministic model does not cater for all possibilities, for example if we have $\alpha = 0$ and $N(0) = 1$ we are starting the simulations with only one aphid so the population can die out and this possibility would be catered for by a stochastic model but not by a deterministic model.

However there are disadvantages in using stochastic models:

- Computation can be more complex.
- The mathematics is more complex.

- The simulation takes longer.

For transparency, we will formulate the model for the case when we have immigration only until time τ and from time $t = \tau$ immigration continues, along with births and deaths. If we let $p_{N,C}$ be the probability the population of aphids is of size N and the cumulative size of the population is of size C at time t and Δt be the time step. In the time interval $(t, t + \Delta t)$,

- the probability of a birth is $\lambda N \Delta t$;
- the probability of a death is $\mu N C \Delta t$;
- the probability of immigration is $\alpha \Delta t$;
- the probability of no event occurring in the time interval is $1 - \lambda N \Delta t - \mu N C \Delta t - \alpha \Delta t$.

We must make Δt small enough so that at the most one event occurs in the time interval $(t, t + \Delta t)$. [Renshaw, 1991]

Therefore we have,

$$p_{N,C}(t + \Delta t) = p_{N,C}(t)\left[1 - \Delta t\alpha\right] + \alpha \Delta t p_{N-1,C-1}(t), \quad \text{for} \quad t < \tau,$$

$$\begin{aligned} p_{N,C}(t + \Delta t) = {} & p_{N,C}(t)\left[1 - \Delta t(\lambda N + \mu N C + \alpha)\right] + p_{N-1,C-1}(t)\left[\Delta t(\lambda(N-1) + \alpha)\right] \\ & + p_{N+1,C}(t)\left[\Delta t \mu (N+1) C\right], \quad \text{for} \quad t \geq \tau. \end{aligned} \tag{2.9}$$

If we divide equation (2.9) by Δt and let $\Delta t \to 0$ we obtain

$$\frac{dp_{N,C}(t)}{dt} = \alpha p_{N-1,C-1}(t) - \alpha p_{N,C}(t), \quad \text{for} \quad t < \tau,$$

$$\begin{aligned} \frac{dp_{N,C}(t)}{dt} = {} & p_{N-1,C-1}(t)\left[\lambda(N-1) + \alpha\right] - p_{N,C}(t)\left[\lambda N + \mu N C + \alpha\right] \\ & + p_{N+1,C}(t)\left[\mu(N+1)C\right], \quad \text{for} \quad t \geq \tau, \end{aligned} \tag{2.10}$$

where $N, C = 0, 1, 2, \ldots$

Similarly, we can formulate the model for the case when we have immigration only until $t = \tau$ and after this time we have no immigration, only birth and death.

$$\frac{dp_{N,C}(t)}{dt} = \alpha p_{N-1,C-1}(t) - \alpha p_{N,C}(t), \quad \text{for} \quad t < \tau,$$

$$\frac{dp_{N,C}(t)}{dt} = p_{N-1,C-1}(t)\left[\lambda(N-1)\right] - p_{N,C}(t)\left[\lambda N + \mu NC\right] + p_{N+1,C}(t)\left[\mu(N+1)C\right], \quad \text{for} \quad t \geq \tau, \tag{2.11}$$

where $N, C = 0, 1, 2, \ldots$

If Equations (2.10) and (2.11) can be solved for $p_{N,C}(t)$, then a complete description of the process for each model is available.

2.4. Stochastic Simulation Techniques

The models we are considering describe a Markov jump process with continuous time and discrete state space. Despite the difficulty in solving Equations (2.10) and (2.11) there exists methods to simulate such a process. Two of these are the Gillespie algorithm and the Tau Leap algorithm. We must first introduce some notation.

Let $Y(t) = (N(t), C(t))^T$ be the state of the system at time t and $\theta = (\lambda, \mu, \alpha)^T$ and define the respective immigration, birth and death reaction rates to be $h_1(Y(t), \theta)$, $h_2(Y(t), \theta)$ and $h_3(Y(t), \theta)$ and $h_0(Y(t), \theta)$ to be the sum of all the individual reactions. So, for example if we choose model (2.10) for $t < \tau$ we would obtain,

$$h_1(Y(t), \theta) = \alpha, \quad h_2(Y(t), \theta) = 0, \quad h_3(Y(t), \theta) = 0 \quad \text{and} \quad h_0(Y(t), \theta) = \alpha.$$

For $t \geq \tau$ we would get,

$$h_1(Y(t), \theta) = \alpha, \quad h_2(Y(t), \theta) = \lambda N(t), \quad h_3(Y(t), \theta) = \mu N(t) C(t)$$
$$\text{and} \quad h_0(Y(t), \theta) = \alpha + \lambda N(t) + \mu N(t) C(t).$$

The Gillespie Algorithm can be used to find the time, w until the next reaction, where $Y(t)$ is the current state of the system and $w \sim \exp(h_0(Y(t), \theta))$. The type of reaction occurring, for example whether birth, death or immigration occurs which will be calculated with probability proportional to the individual rate. Let $Y_{(i-1,i]} = \{Y(t) : t \in (i-1, i]\}$ denote the value of the process in $(i-1, i]$ and the connected density $f(Y_{(i-1,i]}|Y(i-1), \theta)$. If we use the Gillespie algorithm or the Tau Leap algorithm on an interval $(i-1, i]$ conditional on $Y(i-1)$ and θ we will obtain a sample from $f(Y_{(i-1,i]}|Y(i-1), \theta)$. This is important for the inference scheme of Chapters 3 and 4.

2.4.1. The Gillespie Algorithm

The Gillespie algorithm creates a possible solution of a stochastic equation. In 1945 the algorithm was produced by Joseph L. Doob and modified and made better known by Dan Gillespie in 1977. Over time this algorithm has been used to simulate more complicated systems as computers have become faster. The Gillespie algorithm is an exact method for the numerical simulation of a Markov jump process. [Gillespie, 1977]

The algorithm for model (2.10) is as follows:

1. Initialise the system at $t = 0$ and with an initial number of aphids, $N(0)$ and an initial cumulative size of the population $C(0)$, for example $N(0) = 1$ and $C(0) = 1$.

2. Calculate the individual rates for immigration, a birth or a death occurring respectively.
   - For $t < \tau$ we have,
   $$h_1 = \alpha, \quad h_2 = 0 \quad \text{and} \quad h_3 = 0.$$
   - For $t \geq \tau$ we have,
   $$h_1 = \alpha, \quad h_2 = \lambda N(t) \quad \text{and} \quad h_3 = \mu N(t)C(t).$$

3. Calculate the overall rate, h_0.
   - For $t < \tau$ we have $h_0 = \alpha$.
   - For $t \geq \tau$ we have $h_0 = \alpha + \lambda N(t) + \mu N(t)C(t)$.

   If $N(t)$ is big then the overall rate will also be large.

4. Simulate a proposed time to the next reaction where $w \sim exp(h_0)$. Then let $t = t + w$.

5. Pick what reaction occurs at the proposed time.
   - If $t < \tau$ we have immigration so we increase N and C by one.
   - If $t \geq \tau$ we pick a reaction with probability proportional to its individual rate. Select $u \sim U(0, 1)$ to choose which reaction happens.

   $$u < \frac{h_1 + h_2}{h_0} \Rightarrow \text{ birth or immigration } \Rightarrow N = N + 1 \text{ and } C = C + 1$$
   $$\text{otherwise } \Rightarrow \text{ death } \Rightarrow N = N - 1$$

6. Repeat from step 2 if $t < t_{max}$. Otherwise we stop.

2.4.2. The Tau Leap Algorithm

The Tau Leap algorithm was introduced in 2001 and adapted in 2005 to improve speed and efficiency. This method is an approximate alternative to the Gillespie algorithm and uses a time discretisation. [Wilkinson, 2006]

The algorithm for model (2.10) is as follows:

1. Initialise system at $t = 0$, $N = 1$, $C = 1$ and for i in $1 : m$, where $m = t_{max}/\Delta t$ and Δt is the time step.
2. Calculate the individual rates for immigration, birth and death respectively.
   - For $t < \tau$ we have,
   $$h_1 = \alpha, \quad h_2 = 0 \quad \text{and} \quad h_3 = 0.$$
   - For $t \geq \tau$ we have,
   $$h_1 = \alpha, \quad h_2 = \lambda N \quad \text{and} \quad h_3 = \mu NC.$$
3. Simulate number of reactions of each type following a Poisson distribution.
   $$r_1 \sim Pois(h_1\Delta t) \quad r_2 \sim Pois(h_2\Delta t) \quad \text{and} \quad r_3 \sim Pois(h_3\Delta t)$$
4. Let $N = N + r_1 + r_2 - r_3$ and $C = C + r_1 + r_2$
5. If $N < 0$, let $N = 0$ and stop. If $i < m$ we put $i = i + 1$ and repeat from 2. Otherwise we stop.

2.5. Simulations for the Mustard Aphid Data

We have used R in order to write R functions to complete both the Gillespie and Tau Leap algorithms. If we run these codes with the parameters given in Matis et al. [2008] we get simulated mustard aphid data and this can be shown for the J8 data in Figure 2.2. These plots shows that both algorithms seem to produce reaslisations which are a good fit to our data especially for a time step of 0.01, where the realisations for both algorithms look very similar.

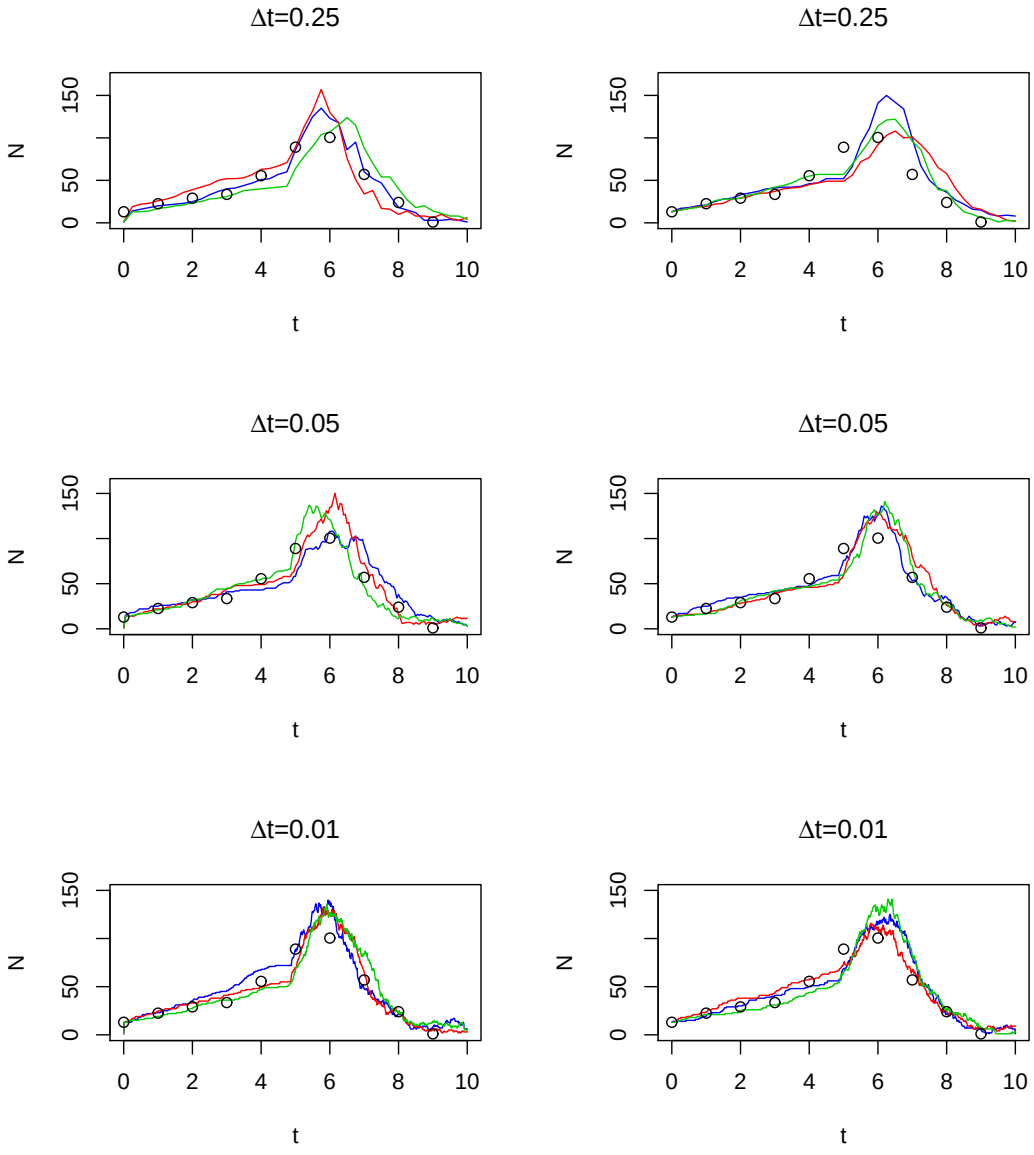

Figure 2.2.: Plot of the simulated data against the J8 data for different time steps. We have used the Gillespie algorithm outputted at each time step for the plots on the left and the Tau Leap algorithm for the plots on the right.

2.6. Comparing Algorithms

The Tau Leap algorithm has many advantages including:

- Simulations are much faster than the Gillespie method. From timing the code 10 times each and taking an average we find that the Tau Leap algorithm runs in R about 5.42 times faster that the Gillespie algorithm code. (To run the Tau Leap algorithm once it took 2.476 seconds and the Gillespie algorithm took 13.428 seconds.)
- Simpler R code.
- Produces results similar to Gillespie method for small time step. It gives exact samples as the time step approaches zero.

However we must consider the disadvantages which are:

- We must assume that the time step is small enough to assume the rate is constant over the time step.
- The Tau Leap algorithm is not as accurate as the Gillespie method as the Gillespie method is exact as it works out the time till the next reaction where as the Tau Leap method only finds out how many reactions of each type occur at each time point.

Therefore we should use the Tau Leap algorithm with a small enough time step so that method is fast and still reasonably accurate.

We have produced many histograms to assess the accuracy of the Tau-Leap algorithm at different time steps against the exact Gillespie algorithm. This was done by running each algorithm 100 times and producing histograms of the simulated states for given time points. We can see from the histograms in Figure 2.3 that there is a peak at zero and this is because we are starting with a small number of aphids in my simulations and the death rate increases due to the cumulative term therefore it is very likely that the population will die out. The histograms were produced using a stochastic model where we have birth, death and immigration at all times and the death rate depends on $C(t)$. It is also shown in Figure 2.3 that as you reduce the time step the histograms become more and more similar. With a time step of 0.01 both algorithms produce very similar histograms; therefore from now on we will use the Tau Leap algorithm with a time step of 0.01 as it is faster than the Gillespie algorithm and also as accurate.

If we use the Tau Leap algorithm with parameter values from Matis et al. [2008] with a time step of 0.01 and run the simulation 100 times and the take the

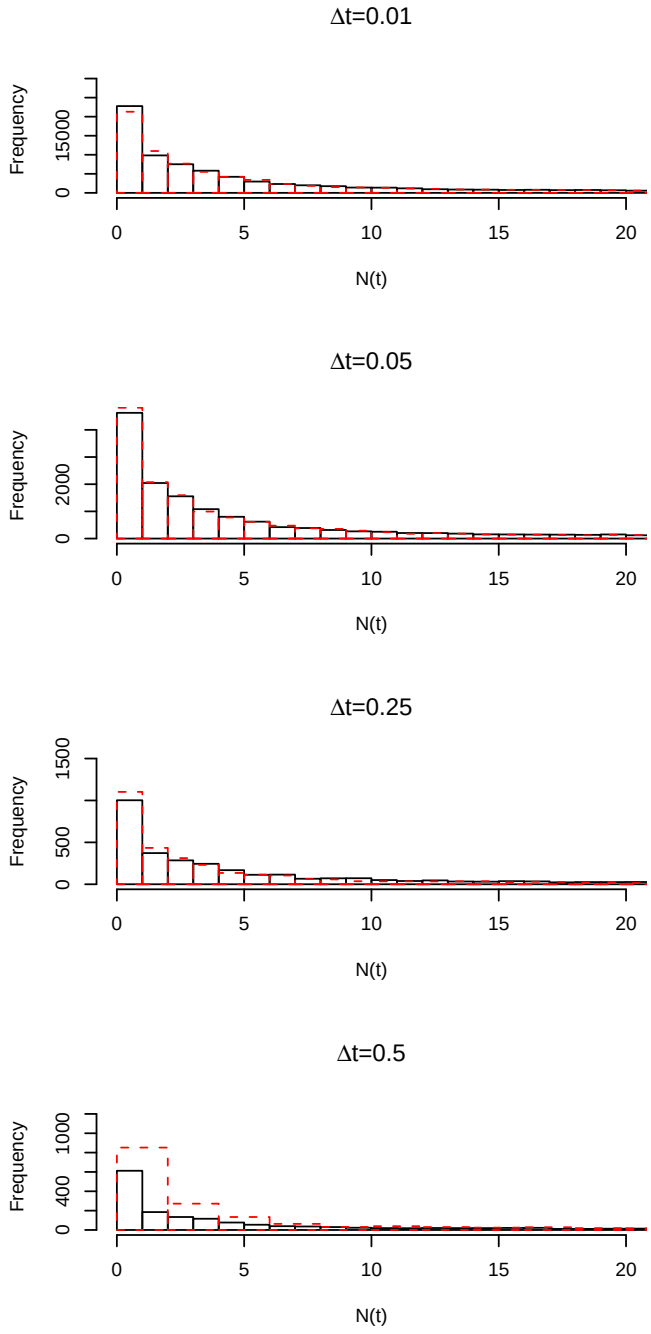

Figure 2.3.: Histograms comparing the Gillespie and Tau Leap algorithms when they are each ran 100 times with different time steps. The black line represents the Gillespie algorithm and the red dashed line represents the Tau Leap algorithm.

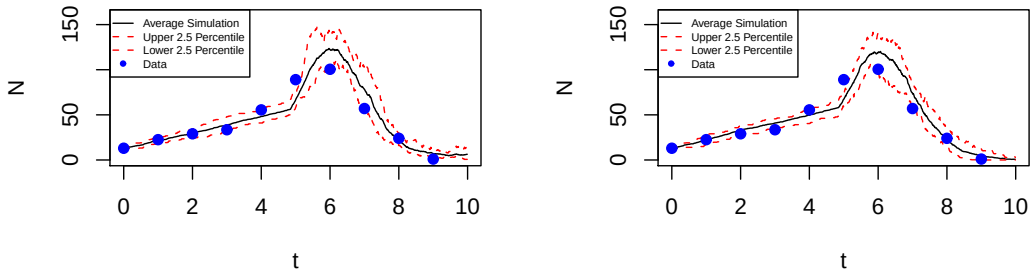

Figure 2.4.: Plot to show results of multiple stochastic simulations for J8 data. The plot to the left represents model (2.11) and the plot to the right represents model (2.10).

average and the upper and lower quartiles over each time step we can plot these curves against the data set for J8 which is shown in Figure 2.4. We can then see from these plots that both models shown in Equations (2.10) and (2.11) produce simulations which fit the data very well. The graph to the right of Figure 2.4 shows that the model shown in equation (2.10) is a better fit around the peak of the data at week 6.

2.7. Reducing Computational Cost

So far the code for the Tau Leap algorithm contains everything inside loops which slow down the running of the code. We can make the code a lot quicker by removing the immigration only part outside the loop as immigration is constant over the time so you do not need to break the immigration only period up into time steps. The code is shown in listing A.2. The reason we can do this is because of the summation rule for Poisson random variables as shown in equation (2.12).

$$\sum_{i=1}^{m} Pois(\alpha \Delta t) = Pois(m\alpha \Delta t) \tag{2.12}$$

3. Importance Sampling

3.1. Introduction

So far, we have been using values from papers in the simulations for the parameters λ, μ, α and τ. However, it would be better if we could estimate these values ourselves using empirical data. In the aphid data $C(t)$ is not observed anywhere. We will assume that $N(t)$ is not observed precisely and that $X_{0:T} = \{X(t) : t = 0, 1, \ldots, T\}$ is observed and that $X(t)$ is connected to $N(t)$ via the measurement error density $f(X(t)|N(t))$. It could be that $X(t) \sim N(N(t), \sigma^2)$ and therefore the measurement error density would be a normal density. We need a method that will let us find out values for θ using the observations $X_{0:T}$. We will use Bayesian methods where $f(Y(0))$ and $f(\theta)$ are the prior distributions of $Y(0)$ and θ respectively. As the posterior distribution is proportional to prior multiplied by likelihood, we have that the joint posterior distribution of θ and the underlying process $Y_{[0,T]} = \{Y(t) : t \in [0, T]\}$ is

$$f(\theta, Y_{[0,T]}|X_{0:T}) \propto f(Y(0))f(\theta)f(Y_{(0,T]}|Y(0), \theta)\prod_{i=0}^{T} f(X(i)|N(i)) \tag{3.1}$$

This density (3.1) is analytically intractable, therefore we need a method that will sample from it which does not need the constant of proportionality. If we sample from it and then keep only the parameter draws we will have a sample from the marginal parameter posterior. There are two methods for sampling that we are going to consider which are Sampling Importance Resampling (SIR) and Sequential Importance Sampling. The latter method is a sequential implementation of the former and belongs to an area known as Sequential Monte Carlo, or particle filtering.

3.2. Sampling Importance Resampling (SIR)

This is a method where we have a target density $f(\cdot)$ which is only required up to proportionality and we sample from an approximating density $g(\cdot)$. This method

has the disadvantage that the sample produced is only approximately distributed like $f(\cdot)$.

The algorithm goes as follows:

1. Sample R points $\left\{\theta^1, \ldots, \theta^R\right\}$ from $g(\cdot)$.
2. Calculate a normalised weight w_r for every θ^r, where
$$w_r = \frac{f(\theta^r)/g(\theta^r)}{\sum_{i=1}^{R} f(\theta^i)/g(\theta^i)}, r = 1, \ldots, R.$$
3. Draw a sample of size S from the discrete distribution $\left\{\theta^1, \ldots, \theta^S\right\}$ with probabilities $w_1, \ldots, w_R$.
4. The sample $\left\{\theta^1, \ldots, \theta^S\right\}$ is approximately distributed $f(\cdot)$.

$f(\cdot)$ is only required up to proportionality because if you multiply a density by a constant the weights would remain unchanged. This method can be shown to work by looking at the distribution function of θ, where θ is univariate.

$$\widetilde{F}_\theta(a) = \sum_{\{r:\theta^r \leq a\}} w_r = \frac{\sum_{r=1}^{R} f(\theta^r)/g(\theta^r) I(\theta^r \leq a)}{\sum_{i=1}^{R} f(\theta^i)/g(\theta^i)}$$

where,

$$I(\theta^r \leq a) = \begin{cases} 1 & \text{if } \theta^r \leq a, \\ 0 & \text{otherwise.} \end{cases}$$

If we let $R \to \infty$ we get

$$\widetilde{F}_\theta(a) \to \frac{\int_\theta [f(\theta)/g(\theta)] I(\theta \leq a) g(\theta) d\theta}{\int_\theta [f(\theta)/g(\theta)] g(\theta) d\theta}.$$

Cancelling $g(\theta)$ in each integral gives

$$\widetilde{F}_\theta(a) = \frac{\int_\theta f(\theta) I(\theta \leq a) d\theta}{\int_\theta f(\theta) d\theta} = P(\theta \leq a).$$

As we need R to tend to infinity for this to work we would need R to be very large for the algorithm to work well enough, especially if f and g are not similar so that only a few points will have weights which are not zero or very close to zero. Therefore R needs to be big enough to cover the support of f sufficiently. We will take $S = R$ in future.

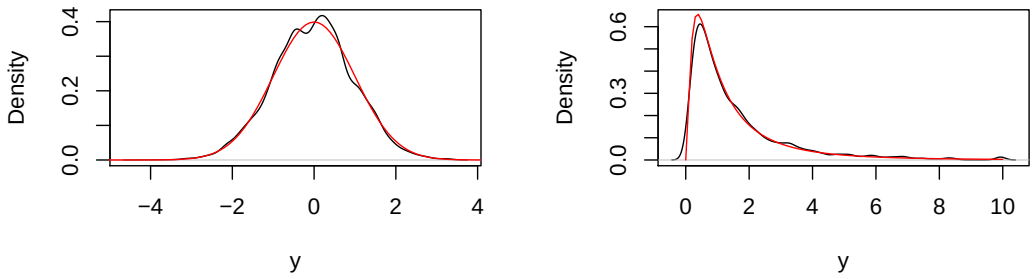

Figure 3.1.: Example of the SIR method. The black line is a kernel density approximation of f and the red line is the actual density. We have on the left a $N(0,1)$ target density and on the right a $LN(0,1)$ target density.

3.2.1. An example of SIR

To illustrate SIR, consider the target density $f(\cdot)$ to be a Standard Normal distribution, $N(0,1)$, and the proposal distribution $g(\theta)$ to be a Uniform distribution, $U(-6,6)$. The code for this is shown in Listing B.1. Therefore we have

$$f(\theta) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\theta^2\right), \quad -\infty < \theta < \infty,$$

$$g(\theta) = \frac{1}{12}, \quad -6 < \theta < 6.$$

We will use $R = 10,000$ in the following examples. There is a plot of the results on the left of Figure 3.1 and you can see that the results produced are very close to the Standard Normal distribution. We can also change this function to work for other target densities and proposal distributions, for example if we take

$$f(\theta) = \frac{1}{\theta\sqrt{2\pi}} \exp\left(-\frac{1}{2}(\log(\theta))^2\right), \quad \theta > 0,$$

$$g(\theta) = \exp(-\theta), \quad \theta > 0.$$

The code is shown in Listing B.2. As shown on the right side of Figure 3.1 again the method works very well and produces a good approximation for the target density.

3.3. Sequential Importance Sampling

Sequential Importance Sampling is a method where we use each observation in turn to find the posterior distribution and then this distribution becomes your new prior. As an example lets take $Y|X \sim N(X, \sigma^2)$ where $X \sim Exp(\lambda)$ and lets suppose that σ^2 is known and we want to estimate λ.

Let Y_1 be our first observation. The algorithm is as follows:

1. Take a sample of size R from $\lambda_j \sim f(\cdot)$ and $X_1^j \sim Exp(\lambda_j)$ where $j = 1, \ldots, R$.

2. Calculate the unnormalised weights for $j = 1, \ldots, R$

$$\begin{aligned} w_j^u &= \frac{f(\lambda_j, X_1^j|Y_1)}{f(\lambda_j)f(X_1^j|\lambda_j)} \\ &\propto \frac{f(\lambda_j)f(X_1^j|\lambda_j)f(Y_1|\lambda_j, X_1^j)}{f(\lambda_j)f(X_1^j|\lambda_j)} \\ &= f(Y_1|\lambda_j, X_1^j) \\ &= \phi(Y_1; X_1^j, \sigma^2), \end{aligned}$$

   where $\phi(\cdot; \mu, \sigma^2)$ represents the density of a Normal distribution with mean μ and variance σ^2.

3. Normalise the weights:

$$w_j = \frac{w_j^u}{\sum_{j=1}^R w_j}.$$

4. Draw a sample from $\{\lambda_j\}$ with probabilities $w_j, j = 1, \ldots, R$ which gives us a sample from the posterior $\lambda|y_1$. We now set this posterior to be our new prior and we repeat from step 2 with our new observation.

3.3.1. An example of Sequential Importance Sampling

To illustrate Sequential Importance Sampling, let $Y|X \sim N(X, \sigma^2)$ where $X \sim Exp(\lambda)$ and suppose that $\sigma^2 = 1$ and we want to estimate λ. Let our prior for λ be a $U(0, 6)$ and let the true value for λ be 3. See Listing B.3 to find the code which implements the algorithm. The code works by using the posterior from the results of observation Y_1 as the prior for the next observation Y_2. When we run the code with $R = 100,000$ we can obtain plots after each observation as shown in Figure 3.2. These plots show that this method produces a distribution which

has a mean of 3.164 even after just five observations which is not far off the true value for λ which is 3. If we used more observations we would get a value which is much closer to 3. We can also see that the variance reduces and the plots become more normally distributed as the number of observations increases.

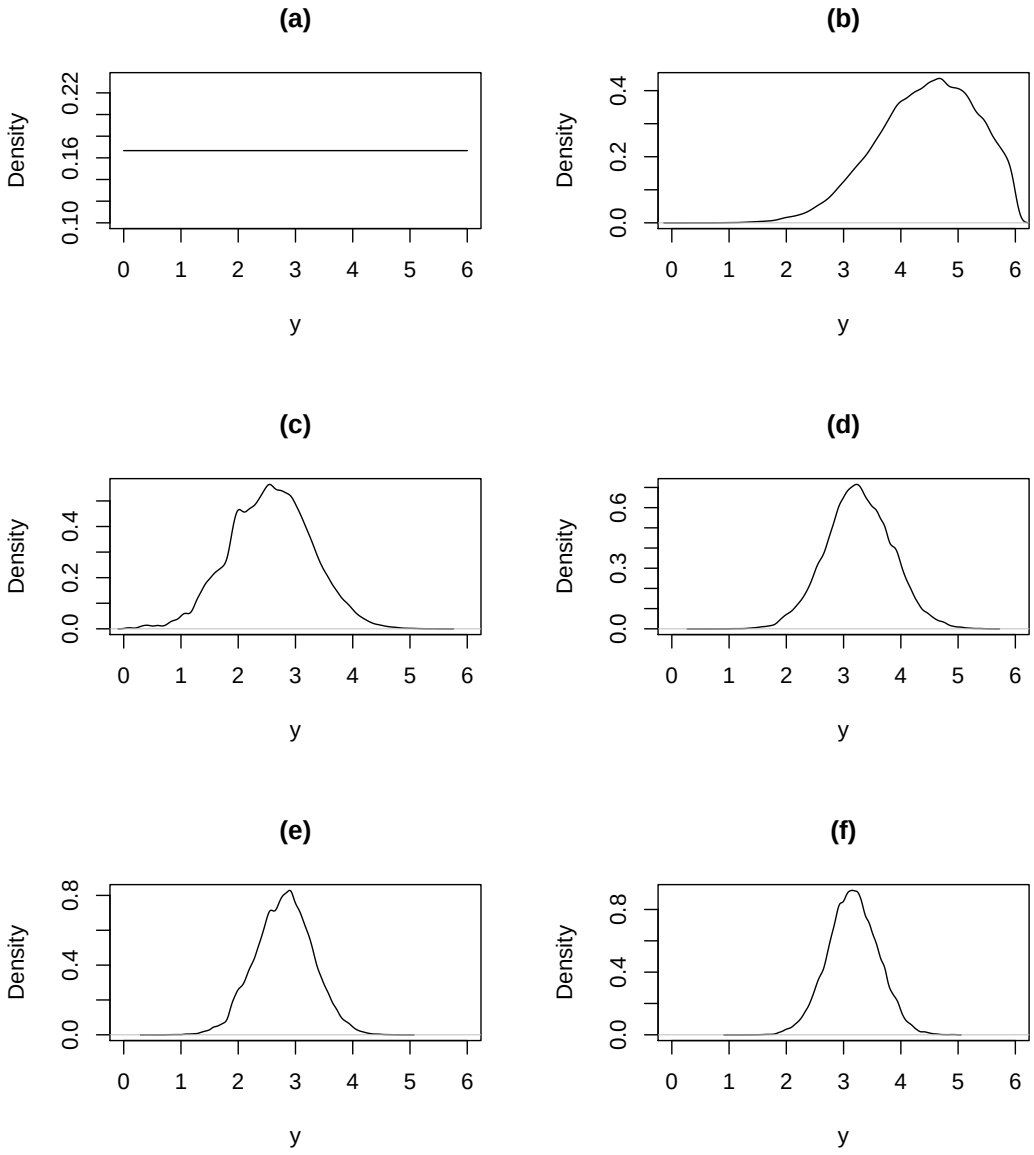

Figure 3.2.: Example of the SIS method. We have the plot for the prior on λ (a) before observation Y_1, (b) after observation Y_1, (c) after observation Y_2, (d) after observation Y_3, (e) after observation Y_4 and (f) after observation Y_5.

4. Application of methods to the Aphid Model

4.1. Introduction

We have two immigration models and we now want to estimate the parameters for each model using a computationally intensive technique. The reason for doing this is that we could find out the parameters for any aphid data set if we have a method that estimated the parameters accurately. We will first look at how Sampling Importance Resampling could be applied to our models.

4.2. Application of SIR to the Aphid Model

It is possible to apply the SIR method to the aphid model. The target density would be the joint posterior of θ and $Y_{[0,T]}$ (3.1). The algorithm is as follows:

1. For $j = 1, \ldots, R$, sample
   - $\theta_j \sim f(\cdot)$ from the prior distributions for λ, μ, α and τ.
   - $Y^j(0) \sim f(\cdot)$ from the prior distributions for $N(0)$ and $C(0)$.
   - $Y^j_{(0,T]} \sim f(\cdot|Y^j(0), \theta^j)$ using the tau leap algorithm.
2. Calculate the unnormalised weights. For $j = 1, \ldots, R$,

$$w_j^u = \frac{f(\theta^j, Y^j_{[0,T]}|X_{0:T})}{f(Y^j(0))f(\theta^j)f(Y^j_{(0,T]}|Y^j(0), \theta^j)}$$
$$\propto \frac{f(Y^j(0))f(\theta^j)f(Y^j_{(0,T]}|Y^j(0), \theta^j)\prod_{i=0}^{T} f(X(i)|N^j(i))}{f(Y^j(0))f(\theta^j)f(Y^j_{(0,T]}|Y^j(0), \theta^j)}$$
$$\propto \prod_{i=0}^{T} f(X(i)|N^j(i)).$$

3. Calculate the normalised weights. For $j = 1, \ldots, R$,

$$w_j = \frac{w_j^u}{\sum_{j=1}^{R} w_j^u}.$$

4. Sample from the discrete distribution $\left\{(\theta^j, Y^j_{[0,T]})^T, j = 1, \ldots, R\right\}$ with probabilities $w_j, j = 1, \ldots, R$.

If R is too small then degeneration will occur and most of the weights will be zero or very close to zero, this means that the final sample will only have one or two values in it. This means we will have to use a different method called Sequential Importance Sampling where our prior for θ is updated with each observation.

4.3. Sequential Importance Sampling and it's Application to the Aphid model

We want to use Sequential Importance Sampling as opposed to straight forward importance sampling for our aphid models in order to avoid degeneration. We have the joint prior for θ and $Y(0)$ as $f(\theta)f(Y(0))$ which we can sample from. Therefore we can assimilate the information in $X(0)$ by sampling from $f(\theta, Y(0)|X(0)) \propto f(Y(0))f(\theta)f(X(0)|N(0))$. By sampling from the prior in the first step of the algorithm, only the measurement error density need be evaluated to construct the weights.

1. For $j = 1, \ldots, R$, sample

$$\theta^j \sim f(\theta^j) \quad \text{and} \quad Y^j(0) \sim f(Y^j(0))$$

from their priors.

2. Find the unnormalised weights:

$$w_j^u \propto f(X(0)|N^j(0)).$$

3. Find the normalised weights:

$$w_j = \frac{w_j^u}{\sum_{j=1}^{R} w_j^u}.$$

4. Sample from the discrete distribution $\left\{(\theta^j, Y^j(0))^T, j = 1, \ldots, R\right\}$ with probabilities w_j, $j = 1, \ldots, R$.

Now we include the information in $X(1)$ by sampling from

$$f(\theta, Y_{[0,1]}|X_{0:1}) \propto f(Y(0))f(\theta)f(Y_{(0,1]}|Y(0),\theta)\prod_{i=0}^{1} f(X(i)|N(i))$$
$$\propto f(\theta, Y(0)|X(0))f(Y_{(0,1]}|Y(0),\theta)f(X(1)|N(1)).$$

Given the sample from $f(\theta, Y(0)|X(0))$, we obtain a sample from $f(\theta, Y_{[0,1]}|X_{0:1})$ as follows:

1. For $j = 1, \ldots, R$, sample
$$Y^j_{(0,1]} \sim f(\cdot|Y^j(0), \theta^j)$$
using the tau leap algorithm.

2. Find the unnormalised weights for $j = 1, \ldots, R$:
$$w^u_j \propto f(X(1)|N^j(1)).$$

3. Find the normalised weights for $j = 1, \ldots, R$:
$$w_j = \frac{w^u_j}{\sum_{j=1}^R w^u_j}.$$

4. Take a sample from the discrete distribution $\left\{(\theta^j, Y^j(1))^T, j = 1, \ldots, R\right\}$ with probabilities w_j where $j = 1, \ldots, R$.

As a general rule, we need to obtain a sample from $f(\theta, Y_{[0,i]}|X_{0:i})$ by using the sample $\left\{(\theta^j, Y^j(i-1))^T, j = 1, \ldots, R\right\}$ from $f(\theta, Y_{[0,i-1]}|X_{0:i-1})$ in the following way:

1. For $j = 1, \ldots, R$, sample
$$Y^j_{(i-1,i]} \sim f(\cdot|Y^j(i-1), \theta^j)$$
using the tau leap algorithm.

2. Find the unnormalised weights. For $j = 1, \ldots, R$:
$$w^u_j \propto f(X(i)|N^j(i)).$$

3. Normalise the weights. For $j = 1, \ldots, R$:
$$w_j = \frac{w^u_j}{\sum_{j=1}^R w_j}.$$

4. Sample from the discrete distribution $\left\{(\theta^j, Y^j(i))^T, j = 1, \ldots, R\right\}$ with probabilities w_j, $j = 1, \ldots, R$.

In general, we run the algorithm as each observation becomes available. The posterior sample obtained at time $i-1$ is used as a prior sample at time i.

5. Results

5.1. Introduction

Ultimately we would like to apply Sequential Importance Sampling to the two data sets J8 and J9 in order to make inferences about the parameters λ, μ, α and τ, where τ is the time when immigration only stops. A program was written which implements the SIS method and can be used with a few small changes to cater for both models (2.10) and (2.11). This code is in Listing B.4. First of all we will run this code with simulated data where we know the values for my parameters in order to validate our inference scheme.

We simulate data using the tau leap algorithm and add some noise to the simulated data. We use the following values for the parameters: $\lambda = 1.7$, $\mu = 0.01$, $\alpha = 4$ and $\tau = 4$. We choose our error density to be normally distributed with a mean of zero and a variance of σ^2. Therefore $X(t)|N(t) \sim N(N(t), \sigma^2)$.

5.2. Results for no immigration after τ

We have run the tau leap algorithm for model (2.11) and adding noise of the form $N(0, \sigma^2)$ where we will have $\sigma^2 = 10$. Running the tau leap algorithm we get the following values for my first 10 observations for model (2.11): (6, 11, 17, 19, 27, 63, 73, 42, 9, 3, 0). We have also chosen prior distributions for my parameters. For λ, μ and α we have chosen a $Ga(0.5, 0.1)$ prior because it has quite a large variance of 50 so you could say it is non-informative and it has a mean of 5 which is not equal to any of the parameter values. For the parameter τ we have chosen a tight prior as from previous studies we know that τ should be around 4 therefore this is reflected in our prior and we have chosen a normal distribution with mean 4 and variance 0.5. We have also chosen a prior for σ^2 to be a $U(5, 15)$ because we want quite a tight support for σ^2 but we do not want to favour any values within that support. The posterior averages for the parameters at each time point are shown in Table 5.1 for model (2.11).

We see that the mean values for the parameters by week 10 are very good estimates of the parameters we used to simulate the data, especially for λ, μ and σ^2 which are only out by -0.103, -0.001 and -0.294 respectively. However α is incorrect by -1.008 which may seem unsatisfactory for α but we do expect some

Table 5.1.: The average value for the parameters at each time step.

Time (Weeks)	Average λ	Average μ	Average α	Average τ	Average σ^2
1	5.250	5.253	2.379	4.008	9.478
2	5.322	5.348	1.742	4.093	9.241
3	5.253	4.995	2.992	4.478	9.247
4	4.658	5.005	2.992	4.996	8.797
5	3.274	0.016	2.992	4.888	9.121
6	1.882	0.010	2.992	4.922	9.982
7	1.548	0.009	2.992	4.833	9.734
8	1.623	0.010	2.992	4.868	9.612
9	1.602	0.009	2.992	4.862	9.801
10	1.597	0.009	2.992	4.859	9.706

error due to the noise we are adding to the simulated data and the jittering of the parameters that we do in the code. τ is also incorrect by 0.859 but again we do expect some error. There is a plot of the distributions of the parameters at week 10 in Figure 5.1 against the prior distributions. You can see that the posteriors for λ and μ look normally distributed, however for α the posterior looks very similar to the gamma prior. τ also looks normally distributed but it has two peaks and σ^2 does not have a noticeable distribution, which could be down to the fact that we are using $R = 50000$ particles in the simulations and we may need to increase this number.

5.3. Results for model with immigration after τ

We have also ran the same code with a few slight changes with new simulated data so that it caters for model (2.10). We used the same values for the parameters and the same prior distributions that we used for model (2.11) and the same value for R. The posterior means at each time point is shown in table 5.2.

We can see that for this model the estimates for λ, μ, α, and τ are better estimates as they were for model (2.11) as the differ by 0.075, 0, -0.206 and 0.897 respectively. However the value for σ^2 is not as accurate as it differs by 1.497. We expect this model to not be as accurate because there are more parameters being estimated after time τ. This could be improved by increasing R. Again in figure 5.2 we have plotted the posterior distributions and the posteriors are very similar shapes as for model (2.11).

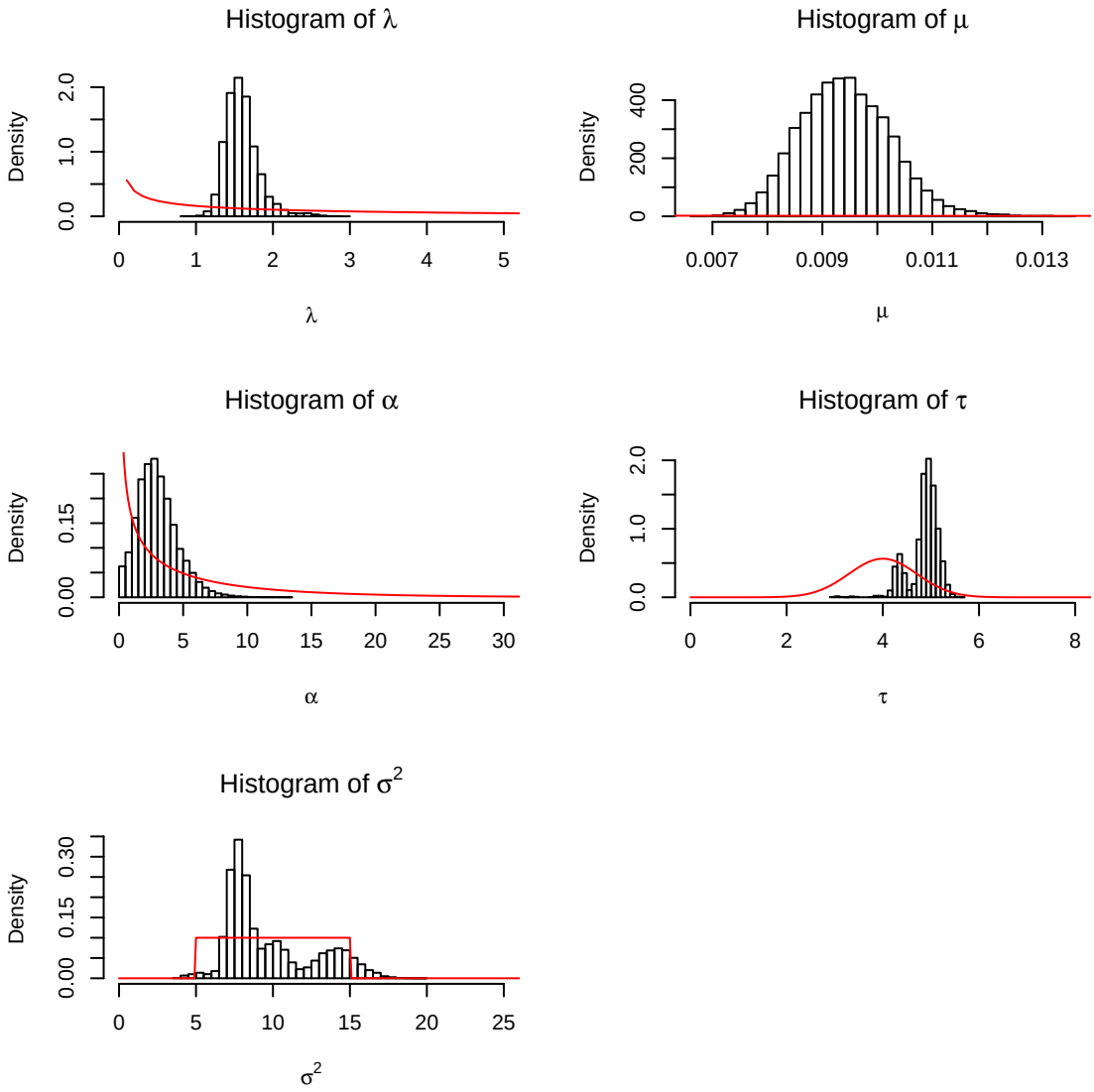

Figure 5.1.: The red lines represent the prior distributions for each parameter and the black lines are obtained from the posterior samples for the parameters in model (2.11) and the variance of the measurement error σ^2.

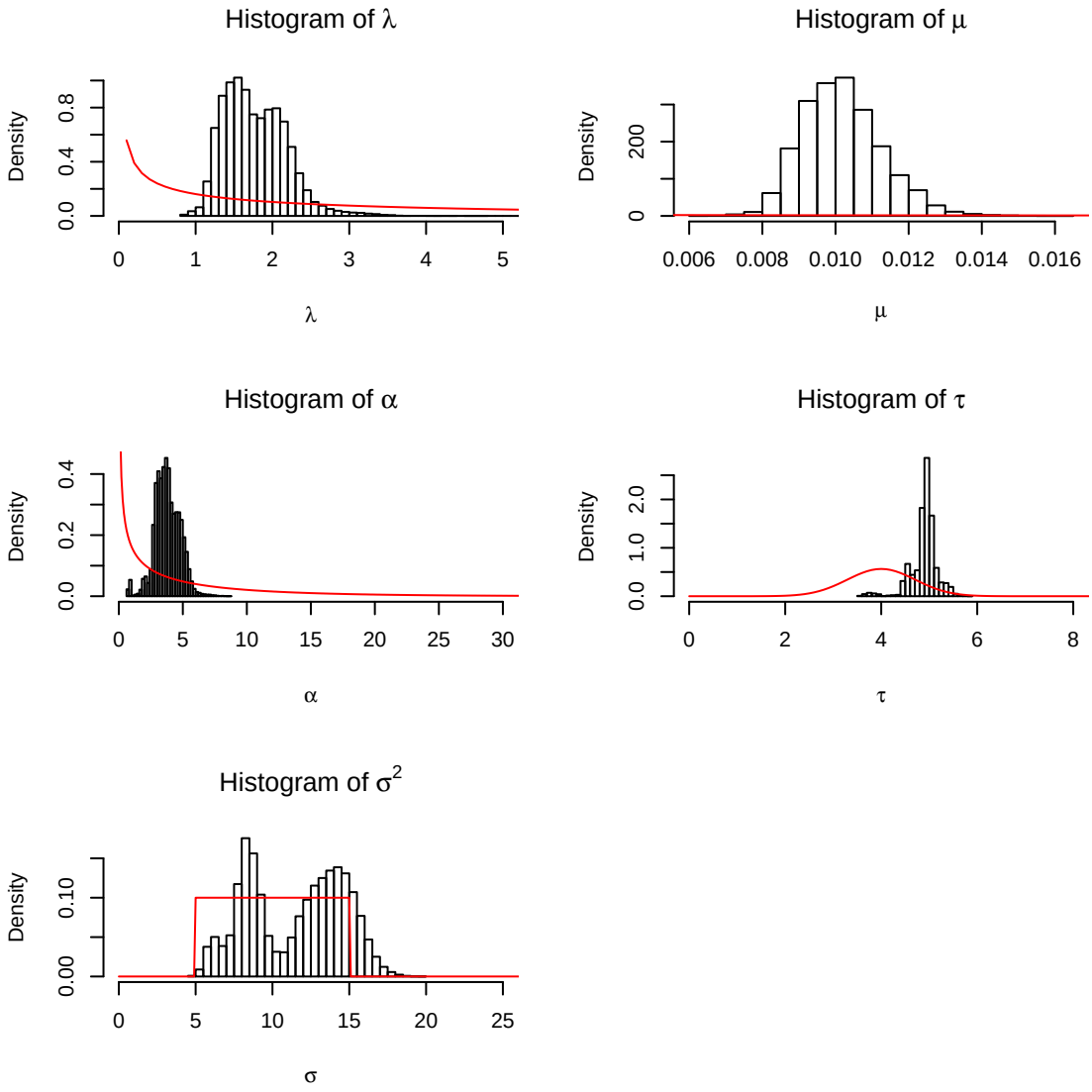

Figure 5.2.: The red lines represent the prior distributions for each parameter and the black lines are obtained from the posterior samples for the parameters in model (2.10) and the variance of the measurement error σ^2.

Table 5.2.: The average value for the parameters at each time step.

Time (Weeks)	Average λ	Average μ	Average α	Average τ	Average σ^2
1	5.140	5.245	2.933	4.005	9.511
2	5.148	5.241	3.521	4.097	9.326
3	5.026	4.934	4.292	4.505	8.990
4	3.712	3.978	4.508	5.078	8.514
5	3.920	0.015	4.258	5.286	8.389
6	2.872	0.011	3.804	5.229	9.653
7	1.911	0.011	3.668	4.943	11.377
8	1.762	0.010	3.724	4.890	11.951
9	1.746	0.010	3.788	4.891	11.826
10	1.775	0.010	3.794	4.897	11.497

6. Conclusions and Further Work

During this project we have analysed two immigration models which fit the mustard aphid data very well. Using Sequential Importance Sampling we have estimated the parameters for simulated data quite accurately and we could see that the model with immigration present at all times was a slightly better choice by looking at the plots of the data and the fitted stochastic models.

In theory Sequential Importance Sampling should work very well and for simulated data it does, however for the real data it is very difficult to avoid sample impoverishment without widening your error distribution which could be due to lack of fit. There are many opportunities for further work in this subject area. Firstly we could use Bayes Factors to compare models which is a Bayesian method used as an alternative to hypothesis testing. The definition of a Bayes Factor, B is the ratio of the marginal likelihoods of the two models of interest. The equation for the marginal likelihood for model, M_j is shown in equation (6.1). Here $\underline{y}$ is the data, θ is a vector of the parameters, and $L_{M_j}(\theta; \underline{y})$ is the likelihood for the model, M_j.

$$\int f^{(0)}_{M_j}(\theta) L_{M_j}(\theta; \underline{y}) d\theta \tag{6.1}$$

Therefore the definition of a Bayes Factor, B is shown in equation (6.2).

$$B = \frac{\int f^{(0)}_{M_1}(\theta) L_{M_1}(\theta; \underline{y}) d\theta}{\int f^{(0)}_{M_2}(\theta) L_{M_2}(\theta; \underline{y}) d\theta} \tag{6.2}$$

Using Bayesian methods for model comparison means that you average over the parameters so all parameter values are taken into account and therefore the comparison does not depend of the parameters of each model. The benefit of this method is that is prevents overfitting as it does not favour models with too much model structure.

We will need to use an approximate marginal likelihood value by taking an average of the weights calculated at each time step of importance sampling and then taking a product of all the averages. Let R be the number of particles, T be the number of observations and w_k be the weights of each particle. Therefore our marginal likelihood, $\hat{p}(y_{1:T})$, will be as shown in equation (6.3).

$$\hat{p}(y_{1:T}) = \prod_{i=0}^{T-1} \frac{1}{R} \left(\sum_{k=1}^{R} w_k^{(i+1)} \right) \tag{6.3}$$

Therefore our Bayes Factor is $B = \hat{p}_{M_1}/\hat{p}_{M_2}$ for our two models. We favour M_1 if the Bayes Factor is large as we favour the model with the larger likelihood.

Finally, implementing Sequential Importance Sampling for the real data and using another computationally intensive technique instead of SIS, for example Markov chain Monte Carlo, remain the subject of ongoing work.

A. Data Simulation

Listing A.1: Gillespie Function

```
gillespie2 = function(maxtime, d_t,theta.row,nc.vec)
{   time = 0; tstep = 0;
    i = 1
    len = (maxtime / d_t)+1
    NC.matrix=matrix(0,nrow=len,ncol=2)
    Time = vector(length = len)
    while(time < maxtime)
    {
        #Calculate the overall rate
        rate = theta.row[1]*nc.vec[1] +
        theta.row[2]*nc.vec[1]*nc.vec[2] + theta.row[3]
        if(rate == 0)#everything is dead.
            time = maxtime
        else
            time = time + rexp(1, rate)

        #Store values at discrete intervals
        while(time > tstep & time < maxtime)
        {
            NC.matrix[i,1]=nc.vec[1]
            NC.matrix[i,2]=nc.vec[2]
            Time[i] = tstep
            tstep = tstep + d_t
            i = i+1
        }

        #Don't go by maxtime
        if(time >= maxtime)
            break;

        #Choose which reaction happens
        u = runif(1)
```

```
        if(u<(theta.row[1]*nc.vec[1])/rate)#birth
        {
            nc.vec[1] = nc.vec[1]+1
            nc.vec[2] = nc.vec[2]+1
        }else if(u < (theta.row[1]*nc.vec[1] +
        theta.row[2]*nc.vec[1]*nc.vec[2])/rate){
            nc.vec[1] = nc.vec[1]-1
        }else if(u < 1) {
            nc.vec[1] = nc.vec[1]+1
            nc.vec[2] = nc.vec[2]+1
        }
    }
    return(NC.matrix)
}
```

Listing A.2: Tau Leap Function

```
tleap3 = function(maxtime, theta.row, nc.vec, d_t=0.01)
{
  tau = theta.row[4]
  if(tau > 0){
    r1 = rpois(1, theta.row[3]*min(tau, maxtime))
    nc.vec[1] = nc.vec[1] + r1
    nc.vec[2] = nc.vec[2] + r1
    maxtime = maxtime - tau
  }

  if(maxtime < 0){return(nc.vec)}

  n = maxtime/d_t;
  for(i in 1:n)
  {
    h1 = theta.row[1]*nc.vec[1]
    h2 = theta.row[2]*nc.vec[1]*nc.vec[2]
    h3 = theta.row[3]*theta.row[6]

    r1 = rpois(1, h1*d_t)
    r2 = rpois(1, h2*d_t)
    r3 = rpois(1, h3*d_t)

    nc.vec[1] = max(0, nc.vec[1] + r1 - r2 + r3)
```

```
    nc.vec[2] = nc.vec[2] + r1 + r3

  }

    return(nc.vec)
}
```

B. Importance Sampling

Listing B.1: SIR Function 1

```
SIR = function(R)
{
        x = runif(R, -6, 6)
        w = dnorm(x)/dunif(x, -6, 6)
        w_r = w/sum(w)
        y = sample(x, R, replace = TRUE, prob = w_r)
        return(y)
}
```

Listing B.2: SIR Function 2

```
SIR = function(R)
{
        x = rexp(R, 1)
        w = dlnorm(x)/dexp(x, 1)
        w_r = w/sum(w)
        y = sample(x, R, replace = TRUE, prob = w_r)
        return(y)
}
```

Listing B.3: SIS Function 2

```
SIS=function(R)
{
lambda=runif(R,0,6)
obs=rexp(1, rate = 1/3)
weights = dnorm(obs, lambda, 1)
nweights = weights/(sum(weights))
posterior=sample(lambda, R, prob=nweights, replace=TRUE)
return(posterior)
}
```

```
#####

SIS2=function(R,lambdaprior)
{
obs=rexp(1, rate = 1/3)
weights=dnorm(obs,lambdaprior,1)
nweights=weights/(sum(weights))
posterior=sample(lambdaprior,R,prob=nweights, replace=TRUE)
return(posterior)
}
```

Listing B.4: Sequential Importance Sampling

```
##### alpha less than tau and mu, lambda, alpha after tau


wresamp2 = function(obs, R = 10000, theta.matrix, NCp.matrix)
{
  V = length(obs)
  sp.matrix = matrix(0, ncol=2, nrow=V)
  weight = rep(0, R)
  # matrix of resampled N and C values
  Sp.matrix = matrix(0, ncol=2, nrow=R)
  for(i in 1:R)
  {
    sp.matrix[1,] = NCp.matrix[i,] #Get first state value
    for(j in 1:(V-1))
    { #run simulator forwards
      state = tleap3(maxtime = 1, theta.row = theta.matrix[i,],
      nc.vec = sp.matrix[j,], d_t = 0.01)
      #get state at each observation time
      sp.matrix[j+1,] = state
    }
    Sp.matrix[i,] = sp.matrix[2,]

  }
  weight = exp((dnorm(obs[2], Sp.matrix[,1],
  sqrt(theta.matrix[,5]), log = TRUE))) #evaluate weights
  s = sample(x = seq(1,R), size=R, prob = weight,
  replace = TRUE) #resample step
  #return resampled particles
```

```
  return(list(theta.matrix[s,], Sp.matrix[s,], weight))
}

#####

wresampseq = function(R)
{
    obs=c(5,  9, 13, 19, 24, 84, 56, 18, 10,  5,  1)
    theta.matrix = matrix(0, ncol = 6, nrow = R)
    #prior for lambda, mu, alpha, tau and sigma squared.
    theta.matrix = priors(theta.matrix, R)
    #prior for initial aphid count
    Np = sample(c(4,5,6), R, replace = TRUE)
    Cp = Np+1 #prior for cummulative aphid count
    NCp.matrix = cbind(Np, Cp)
    #Have to do deal with first obs
    weight = exp(dnorm(rep(obs[1],R), NCp.matrix[,1],
    sqrt(theta.matrix[,5]), log = TRUE))
    s = sample(x = seq(1,R),size = R,prob = weight,
    replace = TRUE)
    for (i in 1:5)
    {
      theta.matrix[,i] = jitter.parameter(theta.matrix[s,i], R)
    }
    NCp.matrix = NCp.matrix[s,]
    #Now do pairs
    distribution = list()
    for (i in 1:10)
    {
      theta.matrix[,4] = theta.matrix[,4]-i
      p = wresamp2(obs = obs[i:(i+1)], R, theta.matrix,
      NCp.matrix)
      theta.matrix = p[[1]]
      theta.matrix[,4] = theta.matrix[,4]+i
      for (k in 1:5)
        {
          theta.matrix[,k] = jitter.parameter(theta.matrix[,k],
          R)
        }
      distribution[[i]] = theta.matrix
      NCp.matrix = p[[2]]
```

```
    }
    return(list(theta.matrix,NCp.matrix,distribution))
}

#####

 jitter.parameter = function(vec, R)
 {
    x = exp(log(vec)+rnorm(R, 0, sqrt(0.01*var(log(vec)))))
    return(x)
 }

#####

 priors = function(theta.matrix, R)
 {
   #lambda, mu and alpha prior samples
   for (i in 1:3)
    {
      theta.matrix[,i] = rgamma(R, 0.5, 0.1)
    }
    # tau prior samples
    theta.matrix[,4] = rnorm(R, 4, sqrt(0.5))
    # sigma^2 prior samples
    theta.matrix[,5] = runif(R, 5, 15)
    theta.matrix[,6] = 1
    return(theta.matrix)
  }
```

Bibliography

C. S. Gillespie and A. Golightly. Bayesian inference for generalized stochastic population growth models with application to aphids. *Journal of the Royal Statistical Society*, 59:341–357, 2008.

D. T. Gillespie. Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81:2340–2361, 1977.

J. H. Matis, T. R. Kiffe, T. I. Matis, J. A. Jackman, and H. Singh. Population size models based on cumulative size, with application to aphids. *Ecological Modelling*, 205:81–92, 2007.

J. H. Matis, T. R. Kiffe, T. I. Matis, and C. Chattopadhyay. Generalized aphid population growth models with immigration and cumulative-size dependent dynamics. *Mathematical Biosciences*, 215:137–143, 2008.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2009. URL `http://www.R-project.org`. ISBN 3-900051-07-0.

E. Renshaw. *Modelling Biological Populations in Space and Time*. Cambridge University Press, 1991.

D. J. Wilkinson. *Stochastic Modelling for Systems Biology*. Chapman & Hall/CRC, 2006.