Bayesian Analysis of 2003-2004 Measles Outbreak in Niamey, Niger

Matthew Ferrari

May 2009

Contents

1 The Basic Removal Model

Recall the example of fitting the basic, closed epidemic using the chain binomial. We can write the incidence at each time step as a function of the indicence in the prior time step,

$$I_{t+1} \sim \text{binomial}(S_t, 1 - e^{-\beta\, I_t}) \qquad \text{where} \qquad S_t = S_0 - \sum_{u=1}^{t} I_u$$

We can easily write the likelihood for β and S_0. Let I represent the entire sequence of cases $\{I_t\}_{t=0}^{n}$. Then the likelihood is just the probability we observed the data I given the parameters β, S_0:

$$f(I|\beta, S_0) = \prod_{t=0}^{n-1} \binom{S_t}{I_{t+1}} (1 - e^{-\beta\, I_t})^{I_t}\, (e^{-\beta\, I_t})^{S_t - I_{t+1}}$$

Now we are going to apply this model to estimate the initial susceptible population size and the transmission rate for measles for a specific example; an outbreak of measles in Niamey, Niger (Grais et al (2006) Proc. Roy. Soc. Trop. Med. Hyg.:100(9),867-873).

2 Background

Niamey, the capital of Niger, has approximately 750,000 inhabitants and has seasonal measles outbreaks during the dry season (October-May). The magnitude of outbreaks varies greatly from year to year Figure 1, due both to strong non-linear dynamics and the stochastic process of local extinction and re-introduction following large outbreaks. There was a large, well-documented outbreak starting in November 2003 and lasting until June 2004. In that time there were approx. 11,000 cases reported Figure 2. For the purposes of this example, we will assume that under-reporting is negligible. In the 24th week of the outbreak, Medecins Sans Frontieres conducted a reactive vaccination campaign with a goal of vaccinating 50% of all children 6-59 months.

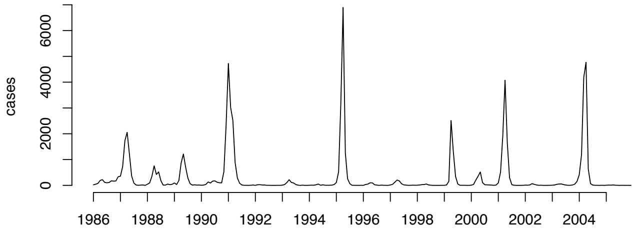

Figure 1: Weekly measles cases in Niamey 1986-2004

There are 3 communes within Niamey. The outbreak began nearly simultaneously in Commune 1 and 2 and then spread to Commune 3 (south of the Niger River) after 4 weeks.

3 The basic model

Using Bayes rule, the posterior can be written as:

$$P(\theta|data) = \frac{P(data|\theta)P(\theta)}{P(data)} \tag{1}$$

or, recalling that $P(data)$ is a constant,

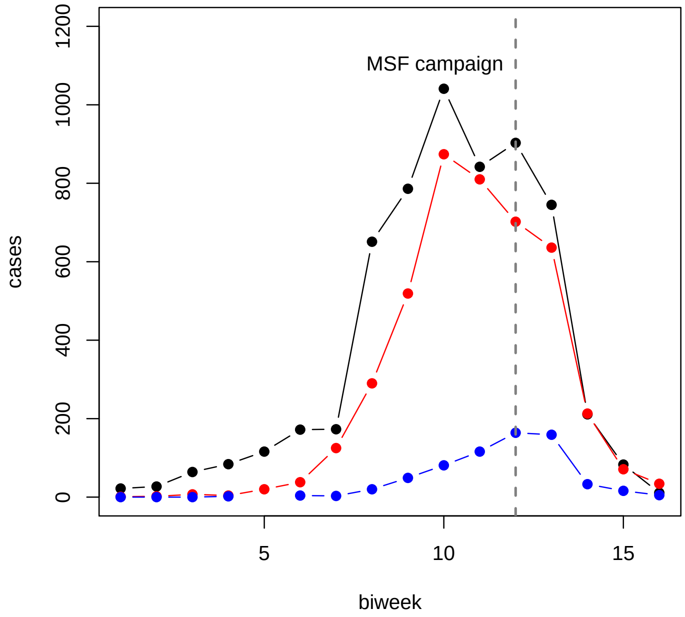

Figure 2: Reported cases in 3 communes of Niamey: Commune 1, 2, and 3, are black, red, and blue respectively.

$$P(\theta|data) \propto P(data|\theta)P(\theta). \tag{2}$$

Here we have:

$$P(\beta, S_0|I) \propto L(\beta, S_0|I)P(\beta)P(S_0). \tag{3}$$

Note that this is the relationship for the joint posterior, and depends on the joint likelihood for β and S_0. It is often simpler to deal with univariate distributions, so we wil take advantage of Block Sampling to approximate the joint posterior. Recall that for Block Sampling, we will iteratively draw from each parameter (β and S_0) conditional on the other. Note that, conditional on S_0, the likelihood for *beta* is binomial, and visa versa. Thus, we can write a generic likelihood function, that takes the parameters in the argument ”par” and the data in the argument ”vecs”:

```
likelihood<-function(par,vecs){
        S0<-par[1]
        beta<-par[2]
        I<-vecs$I
        S<-floor(S0-cumsum(I[1:(length(I)-1)]))
        p<-1-exp(-beta*(I[1:(length(I)-1)]))
        L<-dbinom(I[2:length(I)],S,p,log=TRUE)
        Lik<-sum(L,na.rm=TRUE)
        # note that this returns the log-likelihood
}
```

The conditional posterior is the likelihood times the prior, which we'll get to below. We can then simulate the joint posterior distribution for β and S_0 as follows :

1. Define storage vectors for β and S_0
2. Choose initial values for β^1 and S_0^1
3. Propose a candidate value for S_0^2.
4. Evaluate conditional posterior of the candidate S_0^2 conditional on β^1
5. Evaluate conditional posterior of S_0^1 conditional on β^1
6. Use Metropolis-Hastings to choose $S_0^2 =$ candidate S_0^2 or S_0^1
7. Propose a candidate value for β^2.
8. Evaluate conditional posterior of the candidate β^2 conditional on S_0^2
9. Evaluate conditional posterior of β^1 conditional on S_0^2
10. Use Metropolis-Hastings to choose $\beta^2 =$ candidate β^2 or β^1
11. Store β^2 and S_0^2
12. Return to (3), repeat.

In **R**-Code this looks like:

```
#first read in the data
dat<-read.csv("commune.measles.csv",header=F)
#choose the time series for the Commune 1
I<-dat[,1]
#make the data list to pass to the likelihood function (above)
vecs<-list(I=I)
#set the number of MCMC iterations
iter<-100000
#set storage vectors
S0<-numeric(iter)
beta<-numeric(iter)
#choose initial values
```

```
S0[1]<-floor(1.9*sum(I))
beta[1]<-1e-4
#set step sizes for proposing new candidate values of parameters
S0.step<-10
beta.step<-5e-6
for(i in 2:iter){

        #propose a new value for S0 (note that I've restricted
        #proposals so that S0 must be greater than the sum of reported cases)
        S0.tmp<-0
        while(S0.tmp<sum(I)){
          S0.tmp<-S0[i-1]+sample(c(-1,1),1)*rpois(1,S0.step)}

        # evaluate the marginal posterior at the candidate value of S0
        # note: these are log likelihoods, so we take the sum, not the product
        numer<- likelihood(c(S0.tmp,beta[i-1]),vecs) +
           dunif(S0.tmp,sum(I),10*sum(I),log=T)

        #evaluate the marginal posterior at the of the old value of S0
        denom<- likelihood(c(S0[i-1],beta[i-1]),vecs) +
           dunif(S0[i-1],sum(I),10*sum(I),log=T)

        #Accept or Reject using Metropolis-Hastings
        ifelse(runif(1)<exp(numer-denom),S0[i]<-S0.tmp,
            S0[i]<-S0[i-1])

        #Do the same for beta
        beta.tmp<-0
        while(beta.tmp<1e-9){
          beta.tmp<-beta[i-1]+sample(c(-1,1),1)*beta.step}

        numer<- likelihood(c(S0[i],beta.tmp),vecs) +
           dbeta(beta.tmp,1,1,log=T)
        denom<- likelihood(c(S0[i],beta[i-1]),vecs) +
           dbeta(beta[i-1],1,1,log=T)
        ifelse(runif(1)<exp(numer-denom),beta[i]<-beta.tmp,
           beta[i]<-beta[i-1])

        #cat(i,".\n")        #this is a handy counter that will stream
                                       #the iteration number on the screen,
                                       #so you know that your computer is working
}
```

To implement this algorithm I've had to make two important additions.

1. Choose prior distributions for β and S_0.

2. Choose proposal functions for the candidate values of β and S_0

3.1 Prior distributions

Choosing priors can be a bit of an art. Sometimes one chooses priors because they reflect some prior belief or knowledge of the parameter values. Sometimes one chooses priors because they simplify the computation. Here I have chosen relatively arbitrary priors. I have chosen a uniform prior for S_0, on the interval sum(I) to 10*I, because I know that there have to be more susceptibles than there were cases, but I don't have a good feel for the upper bound. I have chosen a uniform prior for β on the interval 0 to 1.

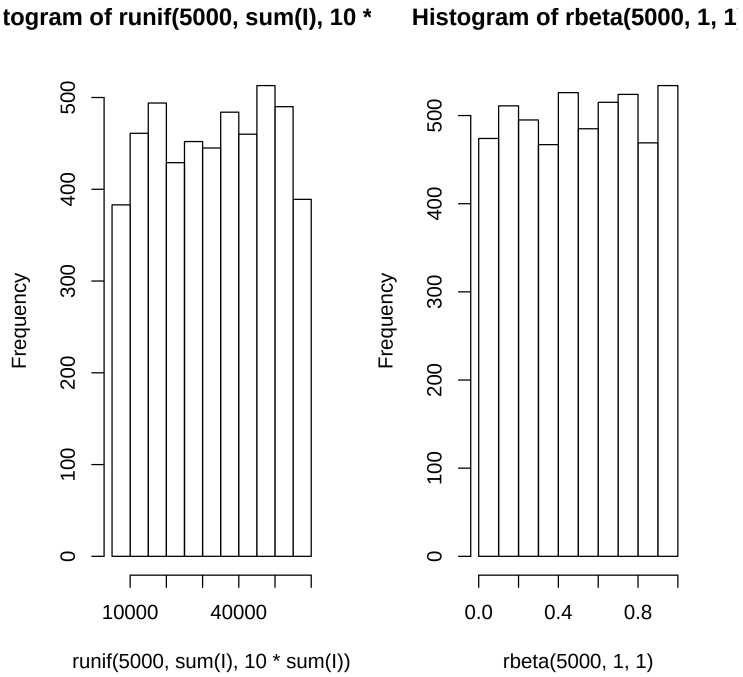

Figure 3: Prior distributions for β and S_0.

3.2 Proposal functions

Formally, the Metropolis-Hastings algorithm depends on the proposal function and should include a term for the probability of getting to the proposed value of the parameter from the previous value and visa versa. Here I have chosen to propose candidate values using a random walk, which means that those probabilities are the same and they cancel out in the Metropolis-Hastings algorithm. This is simply a matter of convenience, and for complicated problems it is often worth the effort to use very specific proposal distributions that are not symmetric (like a random walk).

The step size for the random walk doesn't formally affect the algorithm (i.e. given a long enough time, it should work for any step size), but practically it can be very important for the performance of the MCMC chains. Too small a step size and the chains will mix slowly. To large a step size and you will too often propose values with very low likelihood and the chains will tend to get stuck on values that can't be rejected for long periods.

4 Output of Markov Chains

1. The Markov chain of parameters are a sample from the posterior distribution. We can visualize the posterior by looking at a histogram of the values in the chains.

2. These are numerical simulations, and so we before we do inference we need to evaluate the chains to make sure that they have converged on the appropriate distributions and are no longer influenced by the initial values we gave it. For simple problems we can do this by plotting the chains.

3. When analyzing the chains (i.e. making histograms, calculating point and interval estimates) it is conventional to discard the first N draws to minimize the impact of the initial conditions. It should be apparent from an inspection of the chains when they have converged to the posterior. There are a variety of formal ways to verify if chains have converged. The best involve starting multiple chains from different initial conditions and verifying that they go to the same place: see Cowles and Carlin (1996) "Markov Chain Monte Carlo Convergence Diagnostics: a copmarative review" JASA.

4. Because these random draws were generated using a Markov chain, there is strong autocorrelation in the draws Figure 4. It is common to sub-sample the chains so to approximate independent draws from the posterior. Here I will use only every 100th draw from the chain.

5. The resulting vectors are random draws from the posterior distributions. Thus point and interval estimates are straightforward to obtain. Any measure of central tendency for the posterior draws (i.e. mean, median, mode) can be used as a point estimate; the mean of the posterior is a conventional point estimate . Interval estimates can be simply obtained as the quantiles of the posterior sample (i.e. 5th and 95th percentiles).

6. Under the formalism of Bayes, parameters are thought of as random variables. Thus, it is straightforward to do inference on functions of the draws from the posterior, by simply applying that function to the elements of

the Markov chain. For example, $R_0 = \beta * S_0$; so if we want to get the posterior for R_0 we need only to multiply the elements of the Markov chain for β by the elements of the Markov chain for S_0 to get a new chain that approximates the posterior distribution for R_0.

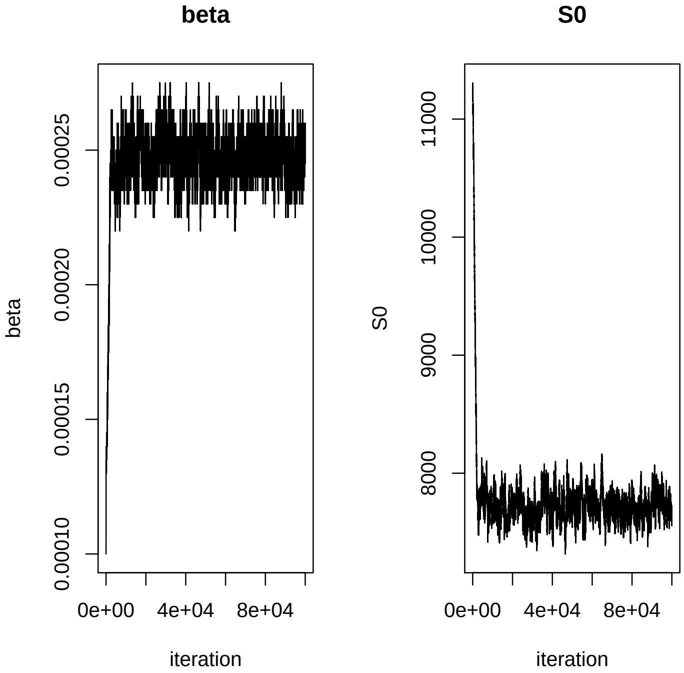

Figure 4: Markov chains for β and S_0.

Exercise 1. Run the above code yourself and compare the estimates of β and S_0 to the maximum likelihood estimates using the code below:

```
# for simplicity set the initial values for optim equal to the postior means
# but try additional starting values to prove to yourself that they don't matter
init<-c(floor(mean(S0.sub)),mean(beta.sub))
# Note: I'm using the argument "control=list(fnscale=-1 . . "
# which tells optim to maximize rather than minimize the likelihood.
mle.fit<-optim(init,likelihood,method="L-BFGS-B",
lower=c(sum(I)+10,1e-4),upper=c(1e4,4e-4),
control=list(fnscale=-1,ndeps=c(1,1e-5)),vecs=vecs)
```

Looking at the output of the chains, we see that the posterior means are very close (almost indistinguishable) from the maximumlikelihood estimates for S_0

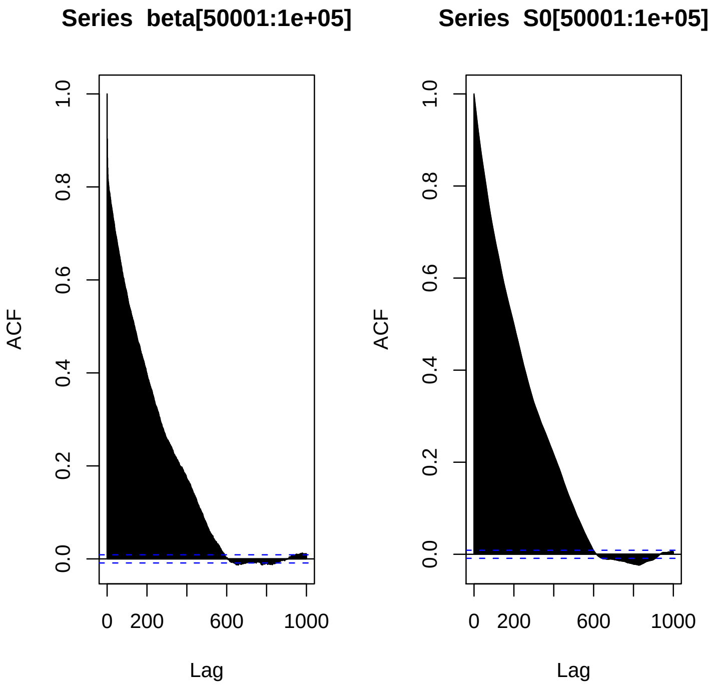

Figure 5: Autocorrelation plots for β and S_0 chains.

and β. That is because as we have formulated the model, there is no information in the prior distributions, so all the information must come from the likelihood. So, while instructive as an example of the algorithms, we've basically taken the long route to get a solution that we could have done using standard maximum likelihood. Now lets look at what we can do to make use of the the Bayesian framework.

5 Making Use of Bayes: Auxiliary data

In the previous example, we used the machinery of Bayes, but applied it to an example that could have more easily been solved with standard maximum likelihood. Now we will extend the analysis to take advantage of the flexibility of Bayesian methods. One challenge to classical likelihood methods is that there is often no standard way to incorporate data from multiple sources. Bayesian methods, however, provide a fairly convenient structure for incorporating additional information through the prior.

While it is difficult to get independent data on the transmission rate in

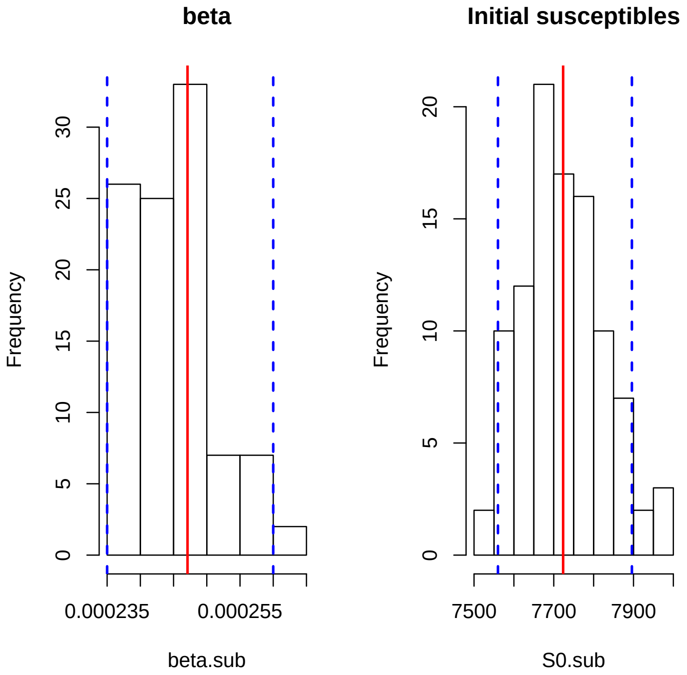

Figure 6: Histograms of subsampled β and S_0 chains. Solid red lines are the mean of the posterior. Dashed blue lines are 90% confidence limits.

Niamey, we can use knowledge of the sytem to express our prior information about the initial suscpetible population size S_0. Vaccination coverage in Niamey is reported to be around 70%. As a part of the reactive campaign in 2004, MSF performed a vaccination survey to estiamte coverage before and after their campaign in the three Communes of Niamey, which we can use to estimate the vaccination coverage for each commune.

Recall that measles is highly episodic in Niamey; measles went locally extinct following an outbreak in 2001 and there were very few cases between late 2001 and late 2003. Additionally, there were 3 years of outbreaks from 1999-2001 **??**. Thus, we can reasonably assume that the susceptible population was very small (near 0) following the 2001 outbreak, and the susceptible population at the start of 2003-2004 outbreak is due to children born while measles was locally extinct. We can then estimate the susceptible population size as the number of births, discounted by the vaccination rate. Population size and birth rate for Niamey are reasonably well known and we can use the 2004 vaccination survey, conducted by MSF, to estimate the vaccination rate.

To do so, we first need to look at the vaccination data; the data are coded by

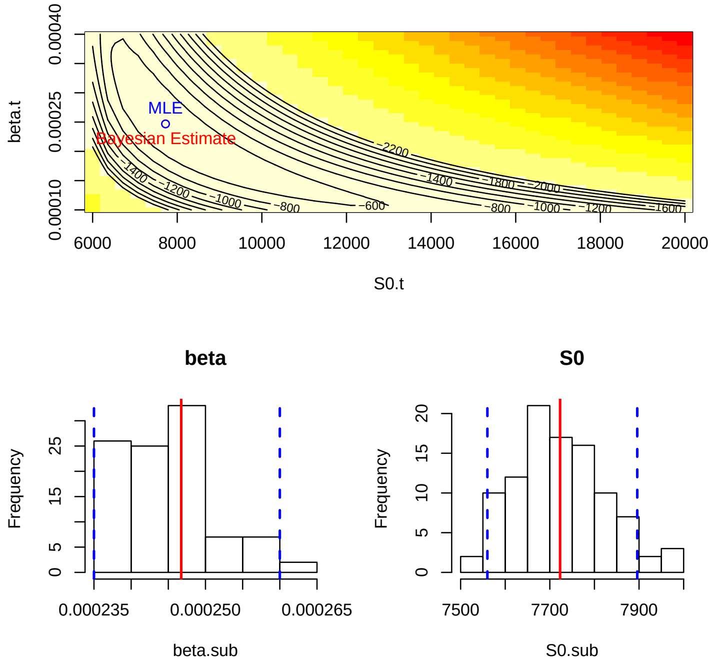

Figure 7: Top panel is the point estimate for β and S_0 on the likleihood surface for the chain binomial. Lower panels are the marginal posteriors for β and S_0 . Solid red lines are the mean of the posterior. Dashed blue lines are 90% confidence limits.

Commune, with 0= not vaccinated, and 1= vaccinated prior to the campaign:

```
vac<-read.csv("niamey.vacc.csv",header=T)
# split data into 3 communes
vacc.commune<-split(vac[,3],vac[,2])
# count number of children sampled in each commune
samp.size<-sapply(vacc.commune,length)
# rate of vaccination
p.vacc<-sapply(vacc.commune,mean,na.rm=T)
```

The total population size for Commune 1 in 2003 is given as 335,122 and the number of live births (correcting for infant mortality) in the inter-epidemic interval is given as 22,432 (Note: this assumes that children born in the 6 months prior to the 2003-2004 outbreak are still protected by maternal immunity). The vaccination rate for Commune 1 from the survey is 0.678; which gives an estimate of the susceptible population size of 7,223. The population sizes and births for

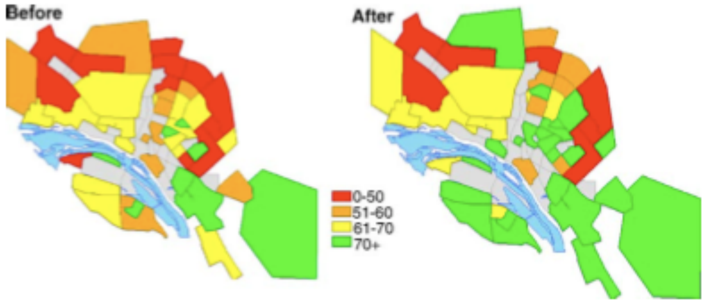

Figure 8: Vaccine coverage across Niamey, Niger. From Dubray et al. 2006. Vaccine.

each commune are given below:

Commune	*PopulationSize*	*Births*
1	335,122	22,432
2	297,836	19,936
3	100,206	6,708

So, based on our assumptions about the susceptible population, we might choose a prior distribution that has a mean of 7,223; say a Poisson distribution with mean of 7,223.

Exercise 2. Change the prior distribution for S_0 from the uninformative uniform distribution, to a prior that reflects our hypothesis that S_0 is about 7,223. Specifically, use a Poisson prior, with mean equal to 7,223. How does this affect the estiamtes of S_0 and β?

Alternatively, we can incorporate the additional information from the vaccine survey directly, and fit a full model, with both sets of data, simultaneously. First we need to re-parameterize the likelihood for the time series of cases in terms of the births and vaccination rate:

```
likelihood<-function(par,vecs){
        p.vacc<-par[1]
        beta<-par[2]
        I<-vecs$I
        S0<-vecs$births*(1-p.vacc)
        S<-floor(S0-cumsum(I[1:(length(I)-1)]))
        p<-1-exp(-beta*(I[1:(length(I)-1)]))
        L<-dbinom(I[2:length(I)],S,p,log=T)
        Lik<-sum(L,na.rm=TRUE)
}
```

Note that this is just the original likelihood, reparameterized with p.vaccc $=$ S_0/births. This new formulation is useful, because we have the vaccine survey data that provides and estimate of the vaccinated proportion, and the uncer-

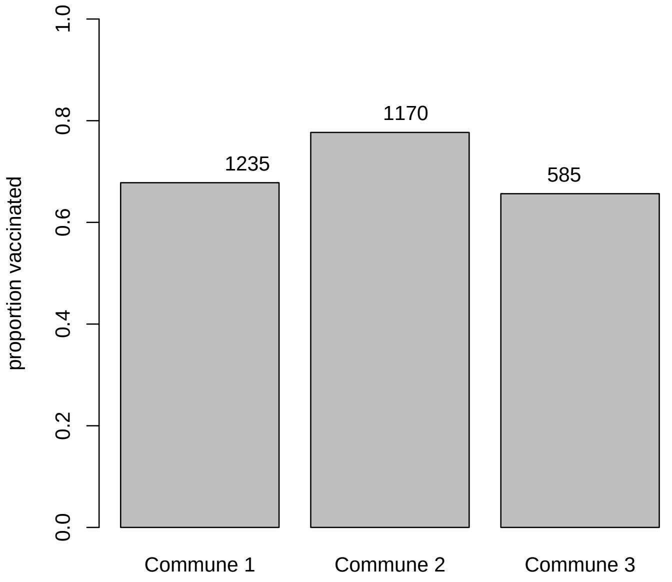

Figure 9: Proportion vaccinated in 3 Communes of Niamey prior to the 2003-2004 outbreak. Numbers above give the sample size in each commune.

tainty in the vaccine rate, through the likelihood. The likelihood for the vaccination rate is a simple binomial, and we can write a simple function for the likelihood (with the same inputs as the time series likelihood) as :

```
likelihood.vacc<-function(par,vecs){
        p.vacc<-par[1]
        x<-sum(vecs$vacc,na.rm=T)
        N<-sum(!is.na(vecs$vacc))
        dbinom(x,N,p.vacc,log=T)
}
```

Because the two datasets are independent (an assumption we are implicitly making) the likelihood for the vaccinated proportion is simply the product of the likelihoods for the two datasets. Thus, we can modify the above code as follows:

```
#first read in the data
dat<-read.csv("commune.measles.csv",header=F)
```

```
#read in the niamey vaccination by commune
vac<-read.csv("niamey.vacc.csv",header=T)
vacc.commune<-split(vac[,3],vac[,2])
samp.size<-sapply(vacc.commune,length)
#select the commune 1 data on cases and vaccination
I<-dat[,1]
vacc<-vacc.commune[[1]]
# note that now "vecs" contains the timeseries
#AND the vaccine survey data, and the births
vecs<-list(I=I,vacc=vacc,births=22432)
#set the number of MCMC iterations
iter<-100000
#set storage vectors
p.vacc<-numeric(iter)
beta<-numeric(iter)
#choose initial values
p.vacc[1]<-.6
beta[1]<-1e-4
#set step sizes for proposing new candidate values of parameters
p.vacc.step<-.001
beta.step<-1e-6
for(i in 2:iter){

        p.vacc.tmp<-p.vacc[i-1]+sample(c(-1,1),1)*p.vacc.step
        numer<- likelihood(c(p.vacc.tmp,beta[i-1]),vecs) +
           likelihood.vacc(p.vacc.tmp,vecs) + dbeta(p.vacc.tmp,1,1,log=T)
        denom<- likelihood(c(p.vacc[i-1],beta[i-1]),vecs) +
           likelihood.vacc(p.vacc[i-1],vecs) + dbeta(p.vacc[i-1],1,1,log=T)
        ifelse(runif(1)<exp(numer-denom),p.vacc[i]<-p.vacc.tmp,
        p.vacc[i]<-p.vacc[i-1])

        beta.tmp<-0
        while(beta.tmp<1e-9){
            beta.tmp<-beta[i-1]+sample(c(-1,1),1)*beta.step}
        numer<- likelihood(c(p.vacc[i],beta.tmp),vecs) +
           dbeta(beta.tmp,1,1,log=T)
        denom<- likelihood(c(p.vacc[i],beta[i-1]),vecs) +
           dbeta(beta[i-1],1,1,log=T)
        ifelse(runif(1)<exp(numer-denom),beta[i]<-beta.tmp,
           beta[i]<-beta[i-1])

        #cat(i,".\n")
}
```

Note that now the prior for S_0 has been replaced with an expression for P_{vacc}. The new expression is the likelihood for vaccinated proportion, given the vacccine survey data times a flat prior for the vaccinated proportion (the distribution beta(1,1) has uniform density on the interval 0,1). Likelihood time prior is the posterior, thus, the posterior for the vaccination data provides a prior for P_{vacc}, with a distribution that reflects the uncertainty in the estimate from the vaccine data.

We can then run the same diagnostic plots for the chains as before, and estimate the point and interval estiamtes in the same way.

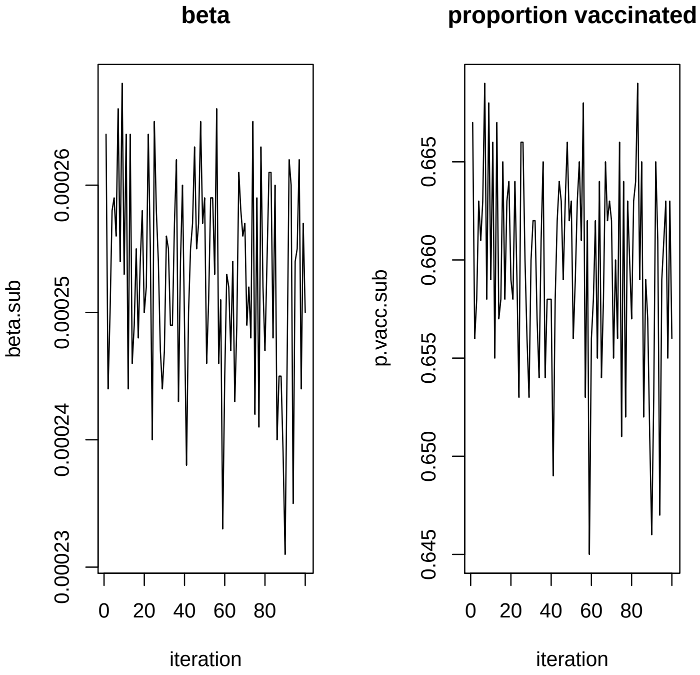

Figure 10: Markov chains for β and P_{vacc}.

We're not terribly interested in the proportion vaccinated, so we can translate the proportion vaccinated P_{vacc} into the initial suscpetible population as $22432 * (1 - P_{vacc})$.

Now we can compare the point estimates and confidence intervals for the three approaches to fit the Commune 1 data.

Exercise 3. Use the code above to estimate the initial suscpetible population size and the transmission rate for Commune 2.

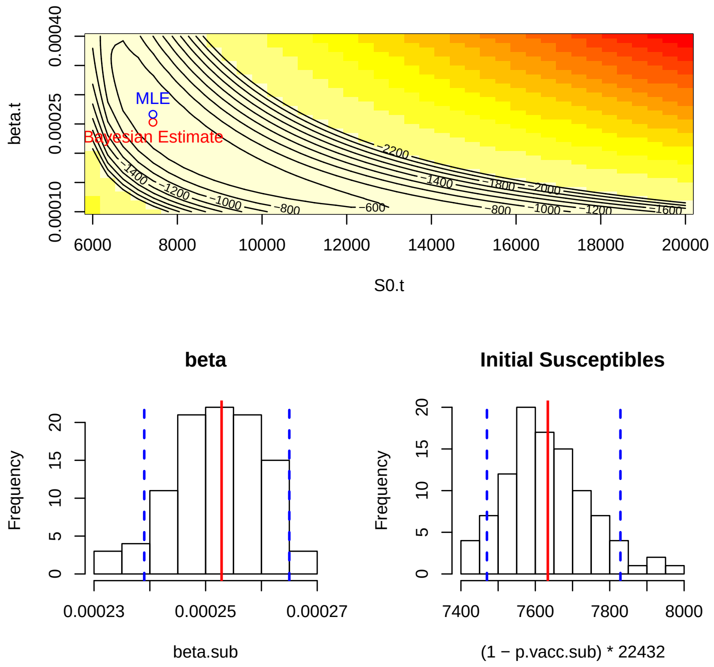

Figure 11: Top panel is the point estimate for β and S_0 on the likleihood surface for the chain binomial. Lower panels are the marginal posteriors for β and S_0 . Solid red lines are the mean of the posterior. Dashed blue lines are 90% confidence limits.

6 Making Use of Bayes: Missing Observations

The fitting the chain binomial model using maximum likelihood cannot handle missing observations, because the likelihood of for the number of cases at time t is dependent on the number in time $t-1$. Bayesian methods, however, can easily overcome this problem by simply treating the missing ovservation as an additional parameter to be estimated. We can explore this using the data for Commune 3, where no cases were reported in the second epidemic week. Clearly, from the chain binomial model, we can't have cases in week 3 if there are no cases to seed infection in week 2, so there must have been some cases that went unreported.

Following the same logic as above, there were 6708 births in Commune 3.

We need to rewrite the likelihood for the time series of cases in terms of hte new parameter, I_2:

```
likelihood<-function(par,vecs){
```

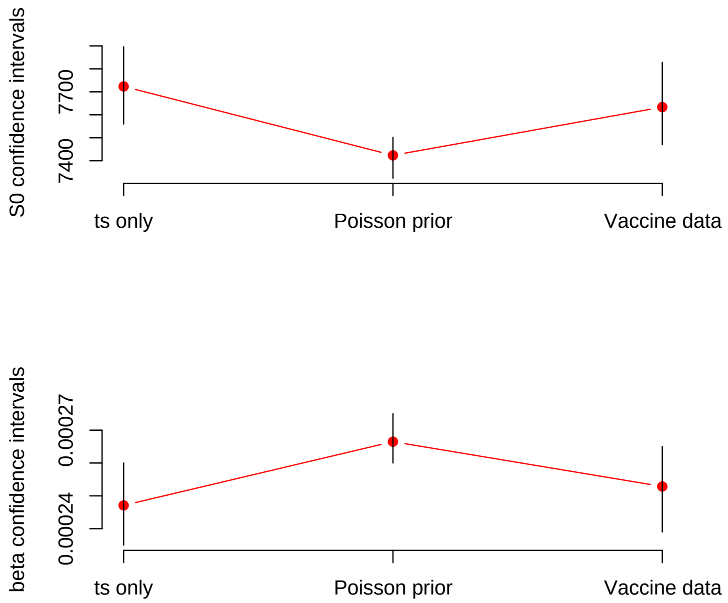

Figure 12: Confidence intervals for S_0 (top) and β (bottom) for three examples.

```
        p.vacc<-par[1]
        beta<-par[2]
        I2<-par[3]
        I<-vecs$I
        I[2]<-I2
        S0<-vecs$births*(1-p.vacc)
        S<-floor(S0-cumsum(I[1:(length(I)-1)]))
        p<-1-exp(-beta*(I[1:(length(I)-1)]))
        L<-dbinom(I[2:length(I)],S,p,log=T)
        Lik<-sum(L,na.rm=TRUE)
}
```

. . . and include an addition step in our loop to sample from the new parameter. Here I include a Poisson prior on the unobserved cases at time step 2, with a mean of 5.

```
dat<-read.csv("commune.measles.csv",header=F)
# note that the epidemic in Commune 3 did not start until the 4th biweek
I<-dat[4:16,3]
```

```
vac<-read.csv("niamey.vacc.csv",header=T)
vacc.commune<-split(vac[,3],vac[,2])
vacc<-vacc.commune[[3]]
vecs<-list(I=I,vacc=vacc,births=6708)
iter<-100000
p.vacc<-numeric(iter)
beta<-numeric(iter)
I2<-numeric(iter)
p.vacc[1]<-.5
beta[1]<-1e-4
I2[1]<-1
p.vacc.step<-.001
beta.step<-5e-6
I2.step<-1
for(i in 2:iter){
I2.tmp<-I2[i-1]+sample(c(-1,1),1)*I2.step
numer<- likelihood(c(p.vacc[i-1],beta[i-1],I2.tmp),vecs) +
    dpois(I2.tmp,5,log=T)
denom<- likelihood(c(p.vacc[i-1],beta[i-1],I2[i-1]),vecs) +
    dpois(I2[i-1],5,log=T)
ifelse(runif(1)<exp(numer-denom),I2[i]<-I2.tmp,I2[i]<-I2[i-1])

p.vacc.tmp<-p.vacc[i-1]+sample(c(-1,1),1)*p.vacc.step
numer<- likelihood(c(p.vacc.tmp,beta[i-1],I2[i]),vecs) +
    likelihood.vacc(p.vacc.tmp,vecs) + dbeta(p.vacc.tmp,1,1,log=T)
denom<- likelihood(c(p.vacc[i-1],beta[i-1],I2[i]),vecs) +
    likelihood.vacc(p.vacc[i-1],vecs) + dbeta(p.vacc[i-1],1,1,log=T)
ifelse(runif(1)<exp(numer-denom),
    p.vacc[i]<-p.vacc.tmp,p.vacc[i]<-p.vacc[i-1])


beta.tmp<-0
while(beta.tmp<1e-9){beta.tmp<-beta[i-1]+sample(c(-1,1),1)*beta.step}
numer<- likelihood(c(p.vacc[i],beta.tmp,I2[i]),vecs) +
    dbeta(beta.tmp,1,1,log=T)
denom<- likelihood(c(p.vacc[i],beta[i-1],I2[i]),vecs) +
    dbeta(beta[i-1],1,1,log=T)
ifelse(runif(1)<exp(numer-denom),
    beta[i]<-beta.tmp,beta[i]<-beta[i-1])

#cat(i,".\n")

}
```

Exercise 4. Run the above analysis, evaluate the diagnostic plots, and obtain point and interval estimates of the parameters.