Efficient Data Augmentation for Fitting Stochastic Epidemic Models to Prevalence Data[2]

STAT517 Final Report

Sarayu Gundlapalli
saru07g@uw.edu

1 Summary of the main research problem

The main research problem addressed in this paper is developing an efficient method for fitting stochastic epidemic models to partially observed prevalence data. SEM models describe the dynamics of an epidemic as a disease spreads through a population, but obtaining complete information about the time evolution of the epidemic is often not feasible. The absence of complete information gives rise to a complicated latent variable problem, especially for larger populations.

The key challenges are:

1. For epidemic models with stochastic dynamics, the likelihood function that relates the model parameters to the observed prevalence data is analytically intractable, especially for large populations.
2. Standard data augmentation MCMC methods that introduce latent individual-level disease histories suffer from convergence issues when little or no individual-level data is available.

The paper aims to develop a Bayesian data augmentation Markov chain Monte Carlo (MCMC) algorithm that can estimate the parameters of stochastic epidemic models from time series of disease prevalence counts, without needing any individual-level data.
The algorithm involves constructing subject-path proposals using a time-inhomogeneous continuous-time Markov process (CTMC) and sampling the exact infection and recovery times using efficient methods like modified rejection sampling and uniformization-based sampling.

2 Contrasting the main contribution with algorithms from lectures

The SIR model describes the time evolution of an epidemic in terms of the disease histories of individuals as they transition through three states — susceptible (S), infected/infectious (I), and recovered (R). The data are sampled from a latent epidemic process, $\mathbf{X} = \{\mathbf{X}_1, \ldots, \mathbf{X}_N\}$, that evolves continuously in time as individuals get infected. The state space of this process is $\mathbf{S} = \{S, I, R\}^N$, the cartesian product of N state labels taking values in {S, I, R}. The state space of a single subject, $\mathbf{X}_j$, is {S, I, R}.

In the subject-path proposal framework described, the main objective is to sample the trajectory (path) of a single subject over time, conditioned on observed data, using a time-inhomogeneous continuous-time Markov process (CTMC). This process captures the evolution of the disease state of the subject, with the state space and transition rates determined by the disease histories of other individuals in the population.

Main Contribution:

The MCMC algorithm for constructing a subject path proposal proceeds:

1. HMM step: The disease state of the subject under consideration is sampled at observation times, conditional on the data and disease history of other subjects.

For sampling a sequence of disease states at the observation times, the emission probability is written as:

$$Y_l|X_j(t_l), I_{t_l}^{(-j)}, \rho \sim Binomial(\mathbb{I}(X_j(t_l) = I) + I_{t_l}^{(-j)}, \rho). \quad (1)$$

The data are conditionally independent of one another, given $\mathbf{x}$ and θ, which induces a hidden Markov model(HMM) over the joint distribution $\mathbf{X}$ and $\mathbf{Y}$.

2. Discrete-time skeleton step: Conditional on the states sampled in the HMM step, the states are sampled at times when the time-inhomogenous CTMC rates change. The problem is reduced to the time-homogeneous case by first sampling the disease state at the intermediate event times when the CTMC rates change, and then sampling the full path within each inter-event interval.
3. Event time step: The exact times of transition events, conditional on the sequence of states sampled in the previous steps, are sampled.

2.1 Constrasting the Metropolis-Hastings algorithm in the paper and class:

Both the paper method and the lecture notes method utilize Markov Chain Monte Carlo (MCMC) techniques, albeit in different contexts and with varying levels of complexity. Both approaches rely on generating a proposed sample and computing an acceptance ratio based on the target and proposal distributions.

- Context:
  - The MCMC method described in class is a standard Metropolis-Hastings algorithm, a general MCMC method for sampling from a target distribution. It involves approximating integrals by generating samples from a posterior distribution.
  - The paper discusses using the Metropolis-Hastings step within a data augmentation framework for epidemic models. The proposal involves constructing complete subject-path proposals and deciding on their acceptance using the Metropolis-Hastings algorithm.
- Distribution:
  - The proposal distribution $g(\theta^{'}|\theta)$ is mentioned explicitly as needing to be symmetric, with Gaussian being a common choice. This simplicity in the proposal mechanism contrasts with the more complex target distribution considerations in the paper.
  - In the paper, the target distribution is complex and involves the product of several terms reflecting the likelihood of observed data given the proposed subject path, the prior distribution of the subject path, and other model-specific factors. This distribution is neither Markovian nor analytically tractable.

3 Propose a simplified version of the main contribution

To create a simplified version of the model described in the paper, we can focus on a basic yet essential aspect of the MCMC framework used for estimating the parameters of a stochastic epidemic model. The simplified model will still utilize the core idea of data augmentation and parameter estimation but with reduced complexity.

- The simplified model reduces complexity by estimating only the transmission rate (β), assuming a known recovery rate (γ), and disregarding other parameters.

- We implement a basic version of the MCMC method, specifically the Metropolis-Hastings algorithm, for parameter updates (likelihood calculations in the original model are based on detailed simulations that account for the time-varying nature of the epidemic).

- The original model uses CTMC to account for time-varying parameters but we use a basic MCMC model as we assume parameters are constant throughout the study.

Data:

1. Use a simple SIR (Susceptible-Infected-Recovered) model for the epidemic. This model divides the population into three compartments based on their disease status: susceptible (S), infected (I), and recovered (R).

2. Assume you have total infection counts over time, this removes the need to differentiate between new and existing infections daily.
3. Estimate the key parameter of the SIR model using the MCMC method - the transmission rate (β), assuming that the recovery rate (γ) is fixed over time.

Implementation:

1. Make an initial guess for β.
2. Given the total number of infections on the day **t** and assuming a fixed γ, calculate the expected number of infections for the day **t+1** using the SIR model equations with the current guess for β.
3. Compare this expected number of infections to the observed number of infections to compute the likelihood.
4. Propose a new value for β, via a simple distribution (like a normal distribution).
5. Go through the same process and decide whether to accept or reject the new β using the Metropolis-Hastings acceptance ratio, based on the likelihood value.
6. After sufficient iterations, analyze the distribution of accepted β values to estimate the most likely transmission rate and its uncertainty.

4 Derive and code the simplified version of the main contribution.

SIR Model:

The SIR model divides the population into three compartments:

- S for Susceptible: individuals who can contract the disease.
- I for Infected: individuals who have contracted the disease and can transmit it to susceptible individuals.
- R for Recovered (or Removed): individuals who have recovered from the disease and are assumed to be immune, or have died, and thus are not part of the susceptible or infected populations anymore.
  The differential equations for the SIR model are[3]:

$$\frac{dS}{dt} = -\frac{\beta SI}{N}$$
$$\frac{dI}{dt} = \frac{\beta SI}{N} - \gamma I$$
$$\frac{dR}{dt} = \gamma I$$

  where N = S + I + R is the total population, assumed to be constant, β is the transmission rate, and γ is the recovery rate.

Metropolis-Hastings Algorithm for β:

1. Start with an initial guess for $\beta = \beta^0$ (for each iteration i)
2. Generate β^* from a proposal distribution, $\beta^* = \beta^{(i+1)}$
3. Calculate the acceptance ratio $A = \frac{P(\beta^*|Data)}{P(\beta^{(i+1)}|Data}$
4. Accept or reject the proposed β^* based on A, updating $\beta^{(i)}$

Algorithm:

1. Initialize Metropolis-Hastings Algorithm
2. Likelihood calculation: Calculate likelihood as the negative log-likelihood of the observed and predicted number of infected individuals.

3. Metropolis-Hastings Sampling (as mentioned above)
4. Estimate the transmission rate as the mean of effective samples.

Check Appendix5 for code

5 Design a synthetic data example and experiment with the code of your simplified model.

We design a data sample for the SIR model, initializing the SIR components and time counts[1]5. The differential equations are defined and the epidemic dynamics are simulated over a specified period. The data is visualized in Figure1.

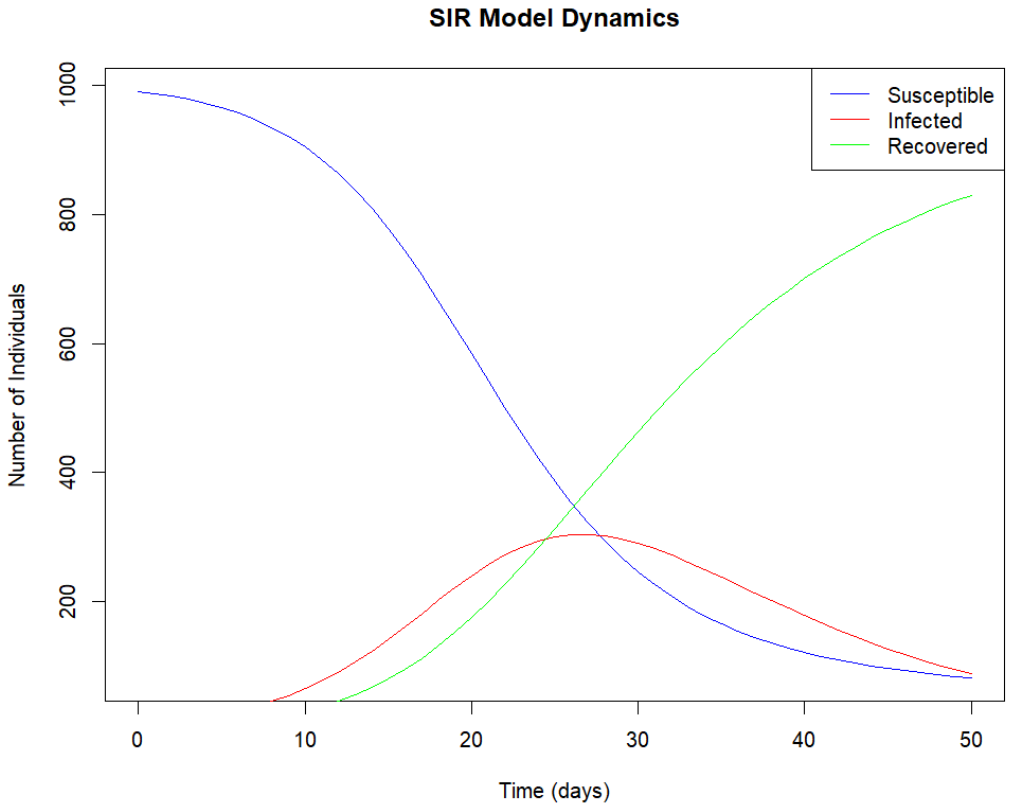

Figure 1: SIR model over a period of time.

This figure represents the progression of an infectious disease through a closed population over a period of 50 days, as modeled by the SIR model. The intersection point of the susceptible and infected lines indicates the peak of the epidemic, after which the infection rate slows down as fewer individuals are left susceptible to infection.

The parameters, transmission rate (β), and recovery rate (γ), determine the speed and breadth of the epidemic spread.

Results:

Figure 2 shows the distribution of the effective sample values of the transmission rate (β) obtained from the Metropolis-Hastings algorithm. The highest bars of the histogram show the most sampled β values, indicating where the true value might be.

The histogram indicates that the majority of the samples are clustered around a narrow range of values, which suggests that the algorithm has converged to a specific region in the parameter space.

The convergence around the true value (β_{true} =0.3) in the synthetic dataset is a good sign that the algorithm is performing well.1

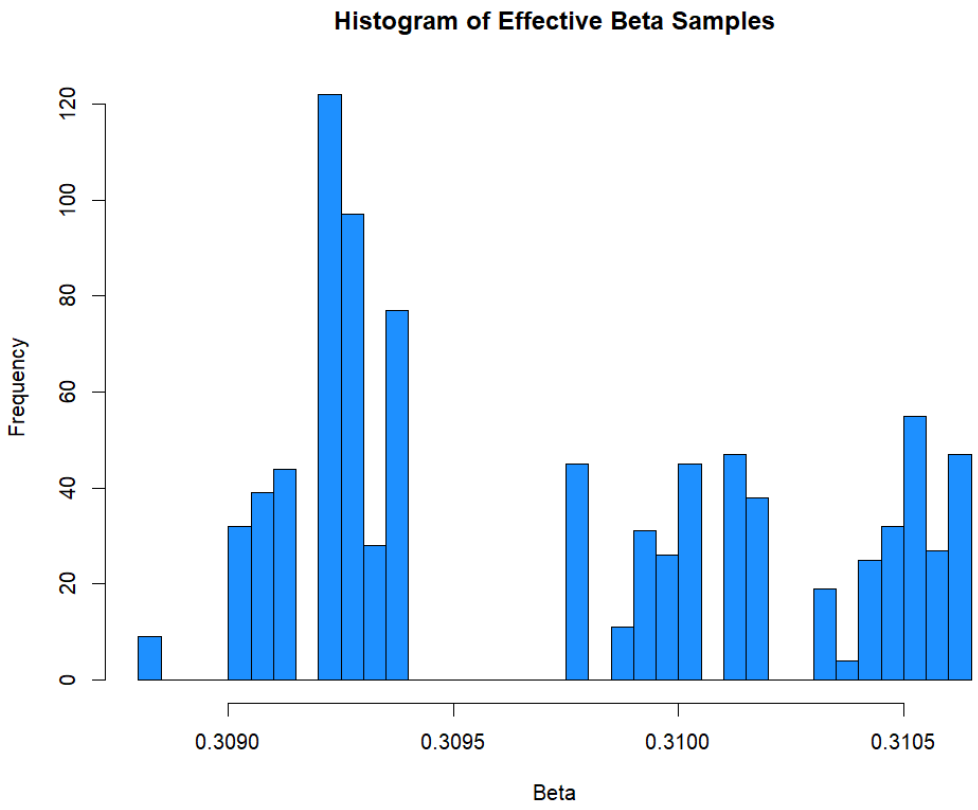

Figure 2: histogram: distribution of the effective sample values of the transmission rate (β) obtained from the Metropolis-Hastings algorithm.

True β	Estimated β
0.3	0.3097342

Table 1: Estimation of β

References

[1] J. Fintzi. https://github.com/fintzij/bdaepimodel/tree/master, Feb 2022.

[2] J. W. Jonathan Fintzi, Xiang Cui and V. N. Minin. Efficient data augmentation for fitting stochastic epidemic models to prevalence data. *Journal of Computational and Graphical Statistics*, 26(4):918–929, 2017. PMID: 30515026.

[3] D. Smith and L. Moore. The sir model for spread of disease - the differential equation model | mathematical association of america, Dec 2004.

Appendix

Synthetic data example:

```
library(deSolve)
library(ggplot2)

# Parameters
S <- 990
I <- 10
R <- 0
N <- S + I + R
beta_true <- 0.3
gamma_fixed <- 0.1
days <- 50  # Time period

# SIR model differential equations
sir_equations <- function(time, state, parameters) {
  with(as.list(c(state, parameters)), {
    dS <- -beta * S * I / N
    dI <- beta * S * I / N - gamma * I
    dR <- gamma * I
    return(list(c(dS, dI, dR)))
  })
}

# Generate synthetic data
initial_state <- c(S = S, I = I, R = R)
parameters <- c(beta = beta_true, gamma = gamma_fixed)
times <- seq(0, days, by = 1)
out <- ode(y = initial_state, times = times, func = sir_equations,
           parms = parameters)
data <- as.data.frame(out)

plot(data$time, data$S, type="l", col="blue", xlab="Time (days)",
ylab="Number of Individuals", main="SIR Model Dynamics")
lines(data$time, data$I, col="red")
lines(data$time, data$R, col="green")
legend("topright", legend=c("Susceptible", "Infected",
"Recovered"), col=c("blue", "red", "green"), lty=1)
```

Simplified model:

```
beta_initial <- 0.1  # Initial guess for beta
sigma_beta <- 0.02  # SD of proposal distribution
n_iterations <- 1000  # Number of iterations
sampled_betas <- numeric(n_iterations)  # To store beta samples
sampled_betas[1] <- beta_initial  # Set the first sample to the initial guess

# Function to calculate likelihood (negative log-likelihood here for simplicity)
calculate_likelihood <- function(beta, data, N) {
  S <- data$S[1]
  I <- data$I[1]
  R <- data$R[1]
  likelihood <- 0

  for (t in 2:nrow(data)) {
    # Predicted new infections
    new_infections <- beta * S * I / N
```

```
    # Update compartments
    S <- S - new_infections
    I <- I + new_infections - (gamma_fixed * I)
    R <- R + (gamma_fixed * I)
    # Calculate likelihood: simple difference for demonstration
    likelihood <- likelihood - abs(data$I[t] - I)
  }

  return(likelihood)
}

set.seed(123)  # For reproducibility
for (i in 2:n_iterations) {
  current_beta <- sampled_betas[i - 1]
  proposed_beta <- rnorm(1, mean = current_beta, sd = sigma_beta)

  # Calculate likelihoods
  likelihood_current <- calculate_likelihood(current_beta, data, N)
  likelihood_proposed <- calculate_likelihood(proposed_beta, data, N)

  # Acceptance ratio
  acceptance_ratio <- exp(likelihood_proposed - likelihood_current)

  # Accept or reject
  if (runif(1) < acceptance_ratio) {
    sampled_betas[i] <- proposed_beta
  } else {
    sampled_betas[i] <- current_beta
  }
}

# Analysis and plot of sampled beta values
effective_samples <- sampled_betas[-(1:100)]  # Assuming first 100
                                               as initial "burn-in values"

# Plotting histogram of effective beta samples
hist(effective_samples, breaks = 50, main = "Histogram of
Effective Beta Samples", xlab = "Beta", col = "dodgerblue")

# Estimating beta as the mean of effective samples
beta_estimate <- mean(effective_samples)
cat("Estimated Beta:", beta_estimate, "\n")
```