Understanding the 2003 Spread of Avian Influenza in Netherlands

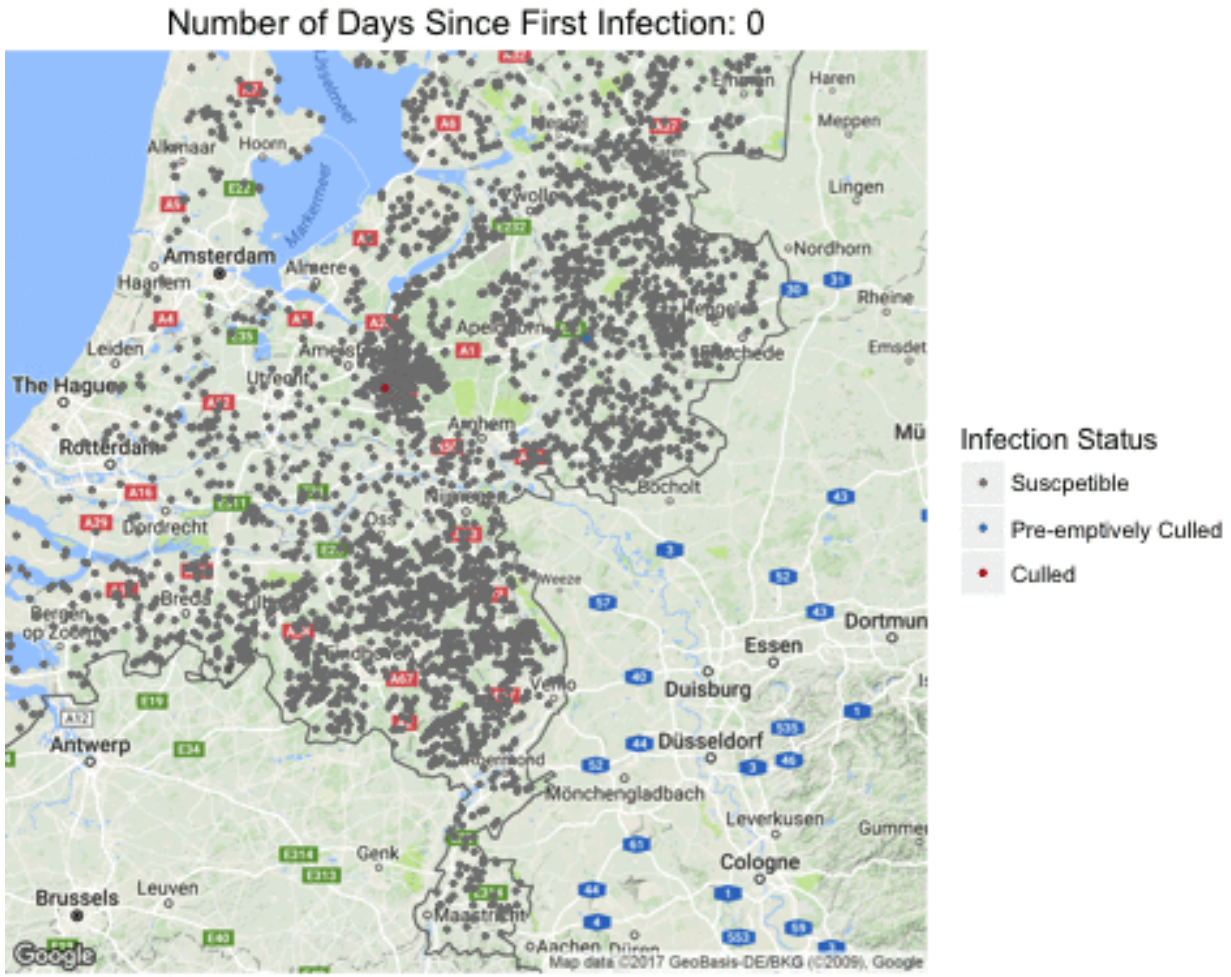
Rowland Seymour
School of Mathematical Sciences

Nottingham High Performance Computing Conference

17^{th} April 2018

Table of Contents

Avian Influenza



Epidemic Modelling

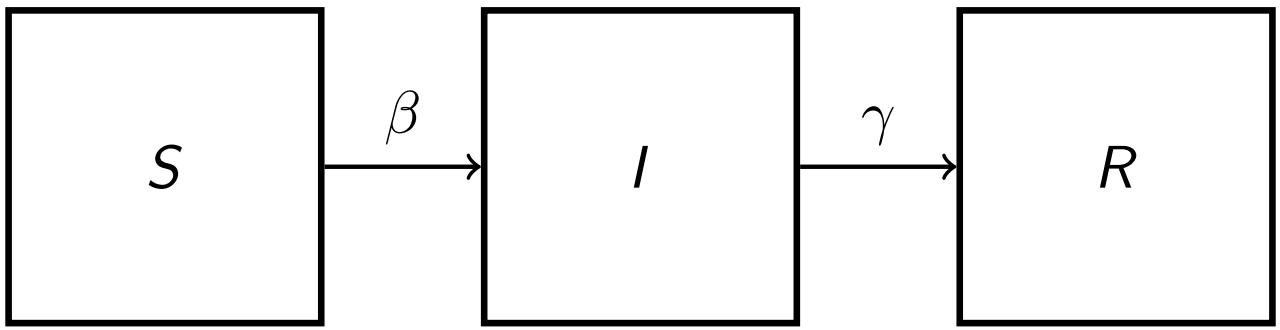


Figure: An SIR model.

- At any time farms are either **S**usceptible, **I**nfected or **R**emoved.
- We use Bayesian inference to infer the infection and removal rates from the available data - infection and removal times.
- For infection on farms, we assume the infection rate depends on the distance between farms.

Bayesian Inference

Combining Bayesian inference and Monte Carlo Markov Chain methods means we need to repeatedly evaluate out likelihood function

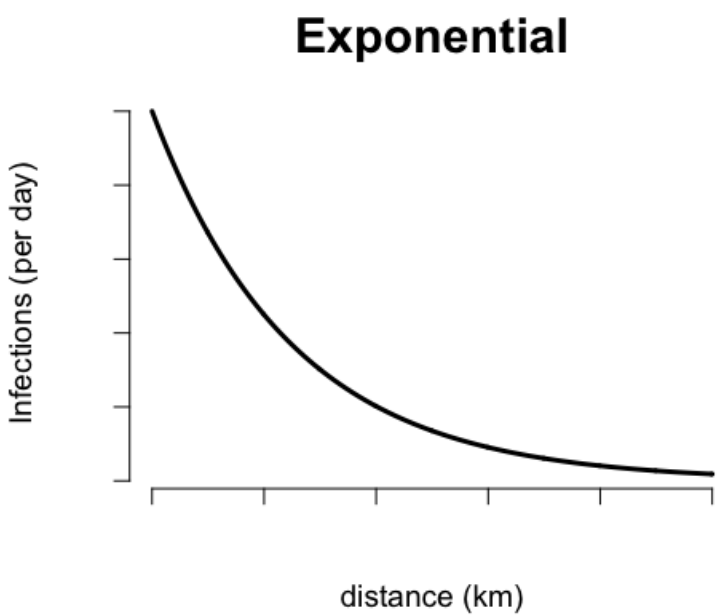
$$\pi(\mathbf{i}, \mathbf{r}|\boldsymbol{\beta}, \gamma) \propto \underbrace{\exp\Big(-\sum_{j=1}^{n}\sum_{k=1}^{N} \beta_{j,k}\big((r_j \wedge i_k) - (i_j \wedge i_k)\big)\Big)}_{\text{Total Infectious Pressure}}$$

$$\times \underbrace{\prod_{j=1}^{n}\Big(\sum_{k\in\mathcal{Y}_j} \beta_{k,j}\Big)}_{\text{Pressure infectives put on each susceptible}}$$

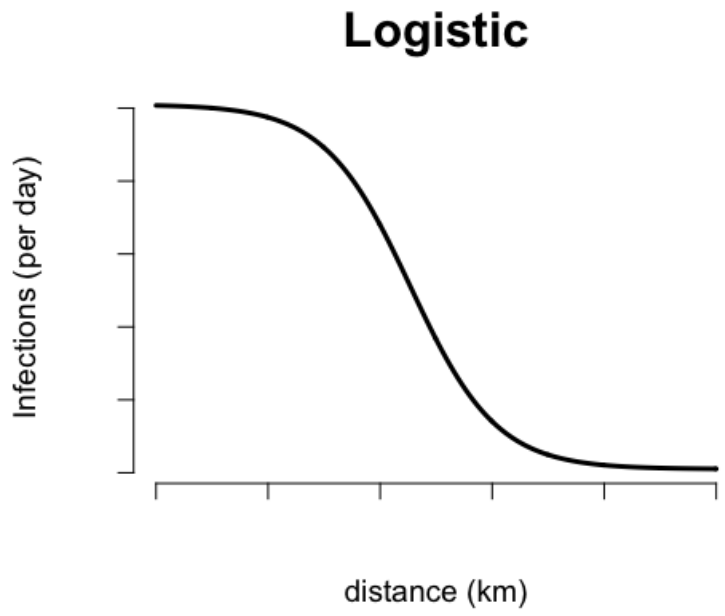
$$\times \underbrace{\prod_{j=1}^{n} f(r_j - i_j|\gamma)}_{\text{Infectious period distribution}} \quad .$$

The Infection Rates

We can model the infection rate as a function of the distance between two farms j and k.



(a) $\beta_{j,k} = \beta_0 \exp\{-\beta_1 d_{j,k}\}$

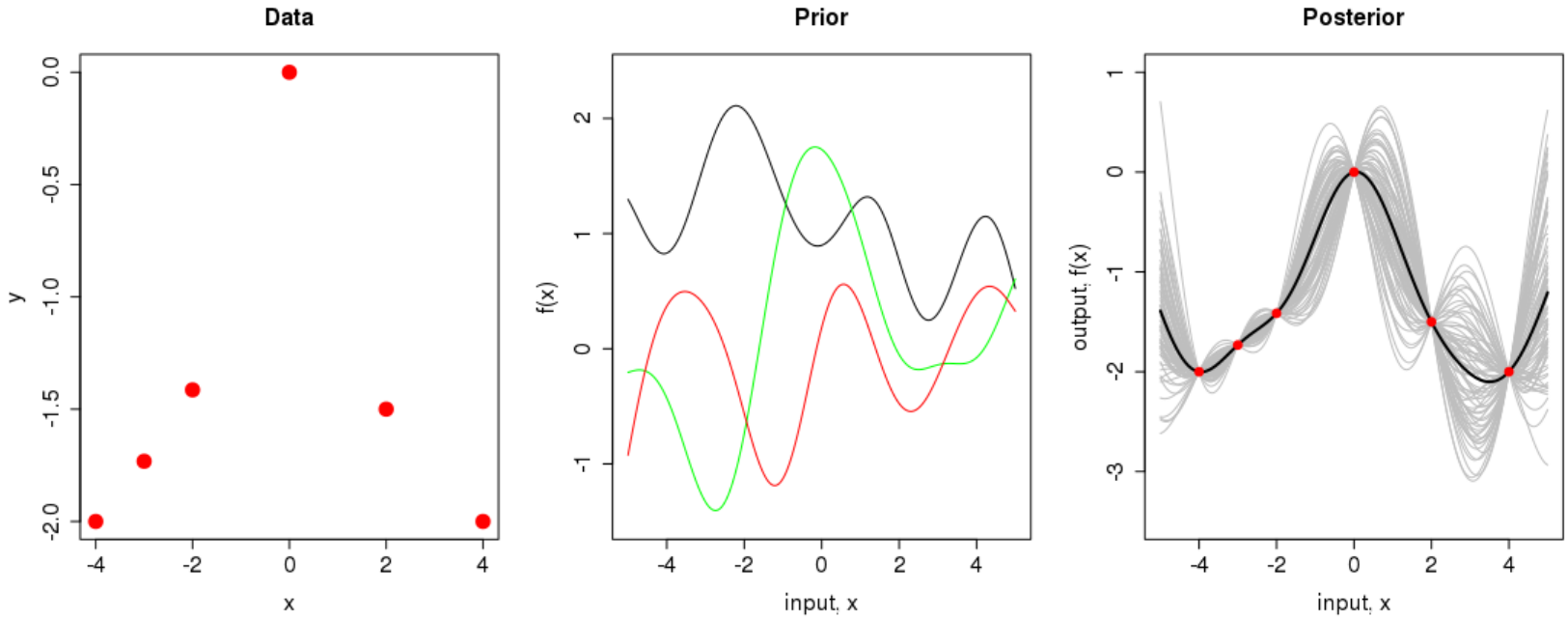


(b) $\beta_{j,k} = \frac{\beta_0}{1+d_{j,k}^{\beta_1}}$

This can be very restrictive, as we have to specify the exact form. Instead, we can use non-parametric methods.

Nonparametric Infection Rates

- We can use Gaussian Processes to estimate the infection rate nonparametrically.
- We construct a covariance matrix, giving how each farm is related.
- Inference algorithms require us to invert and decompose these matrices.



Computation Difficulties

The size of the avian influenza data set can cause difficulties.

- There were 5,539 poultry farms in the Netherlands.
- We only observe the culling dates.
- Over 1,200 farms were culled without knowing the infection status.

The MCMC algorithm also creates computational difficulties.

- It needs to run for about 50,000 iterations.
- This can take several days.

Table of Contents

Nonparametric Inference

For each outbreak, we want to infer the infection rate, infection times, and length of infectious period. To do this we use MCMC and for each iteration, we

- suggest changing 200 infection times.
- decompose the covariance matrix and suggest new parameters to control the covariance.
- propose a change to the infection rate.

At each of the steps we recompute the likelihood function. This takes about 6 days for 50,000 iterations.

Parallel Computing

We can split the infection rate into three distinct parts, and then split each part between multiple threads.

$$\pi(\mathbf{i}, \mathbf{r}|\boldsymbol{\beta}, \gamma) \propto \underbrace{\exp\Big(-\sum_{j=1}^{n}\sum_{k=1}^{N} \beta_{j,k}\big((r_j \wedge i_k) - (i_j \wedge i_k)\big)\Big)}_{\text{Different values of } j \text{ can go to different threads.}}$$

$$\times \underbrace{\prod_{j=1}^{n}\Big(\sum_{k\in\mathcal{Y}_j} \beta_{k,j}\Big)}_{\text{Different values of } j \text{ can go to different threads}}$$

$$\times \underbrace{\prod_{j=1}^{n} f(r_j - i_j|\gamma)}_{\text{Different values of } j \text{ can go to different threads}} .$$

With 16 threads, this reduces the time to about 2.5 days.

Simulation Study Results

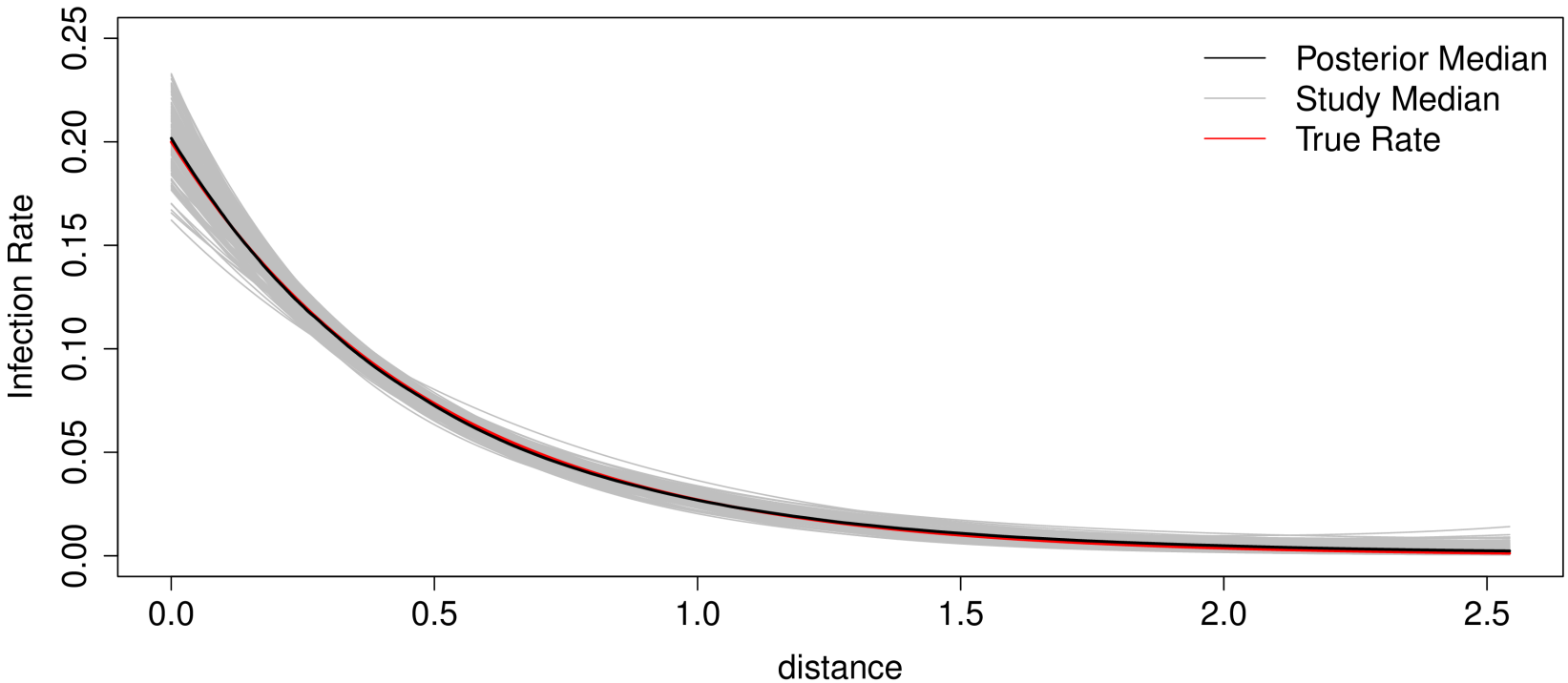


Figure: Nonparametric estimates for infection rate.

Avian Influenza Results

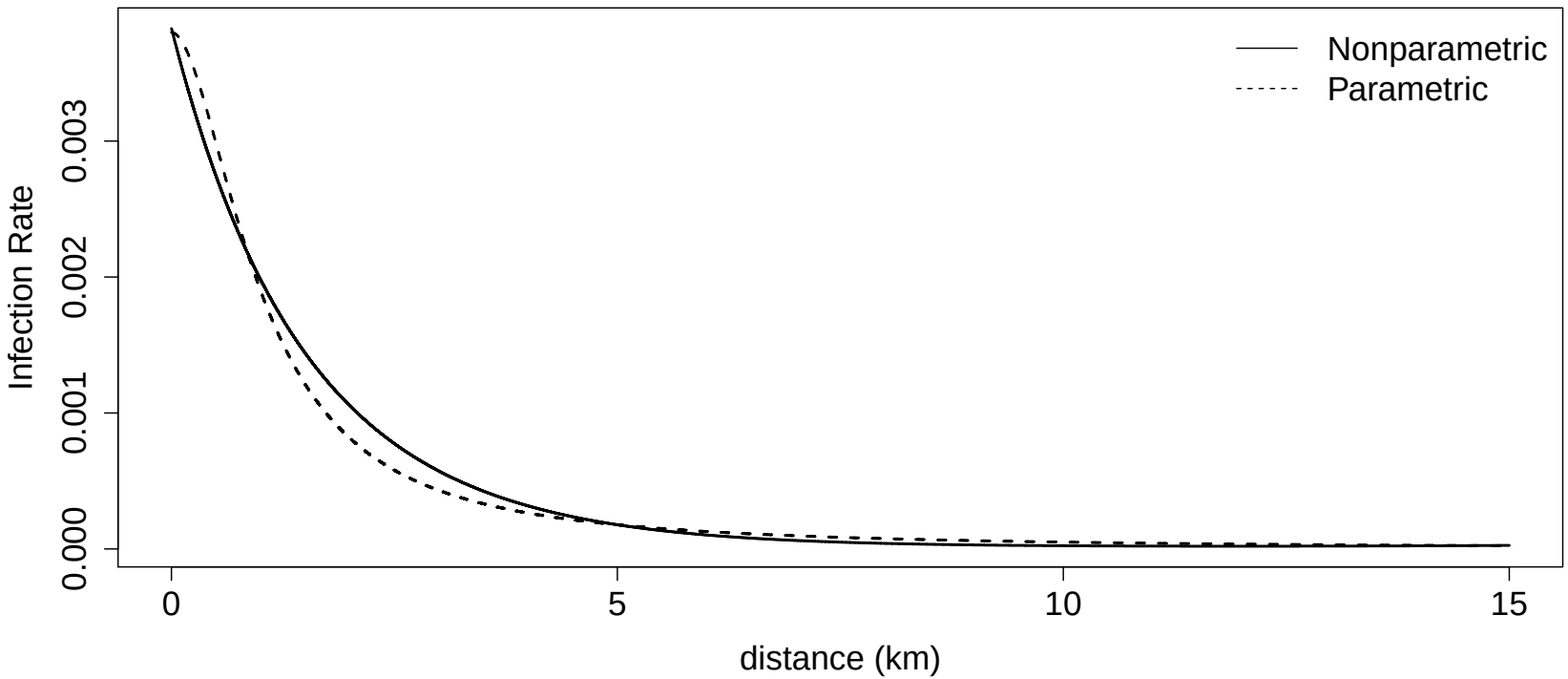


Figure: Nonparametric Infection Rate (Solid) compared to parametric estimate (dashed) for the Avian Influenza data set.

Table of Contents

Tips

1. A little bit of bash can go a long way
    1. Use for and while loops and move around files.
    2. Very helpful for batch jobs.
2. OpenMP is a easy way to parallelise C/C++/Fortran.
    1. Simple to add into code.
    2. Only need to change the number of cores on the HPC.
3. Use the HPC for simulation studies
    1. Use `#PBS -J 1-250` to run the code 250 times, and refer to them using `${PBS_ARRAY_INDEX}`.
    2. Useful for making sure your code gives the right answer.