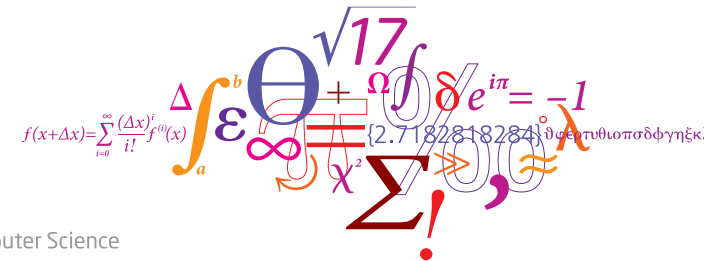
Automated and Early Detection of Disease Outbreaks

Kasper Schou Telkamp

Section for Dynamical Systems



DTU Compute
Department of Applied Mathematics and Computer Science

Outline

- Data
  - Focusing on STEC
- Novel outbreak detection algorithm
- Model formulas
  - Agegroup
  - Agegroup and seasonality
- Hierarchical Poisson Normal model
  - Implementation
  - Results
- Hierarchical Poisson Gamma model
  - Implementation
  - Results

Data

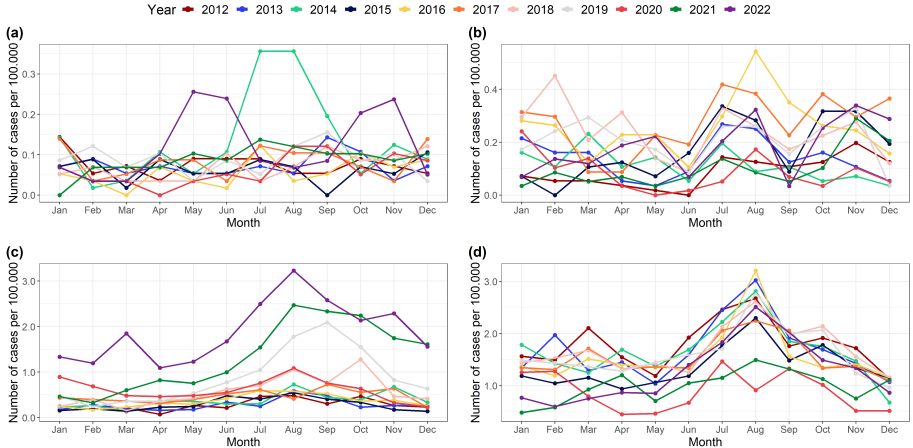
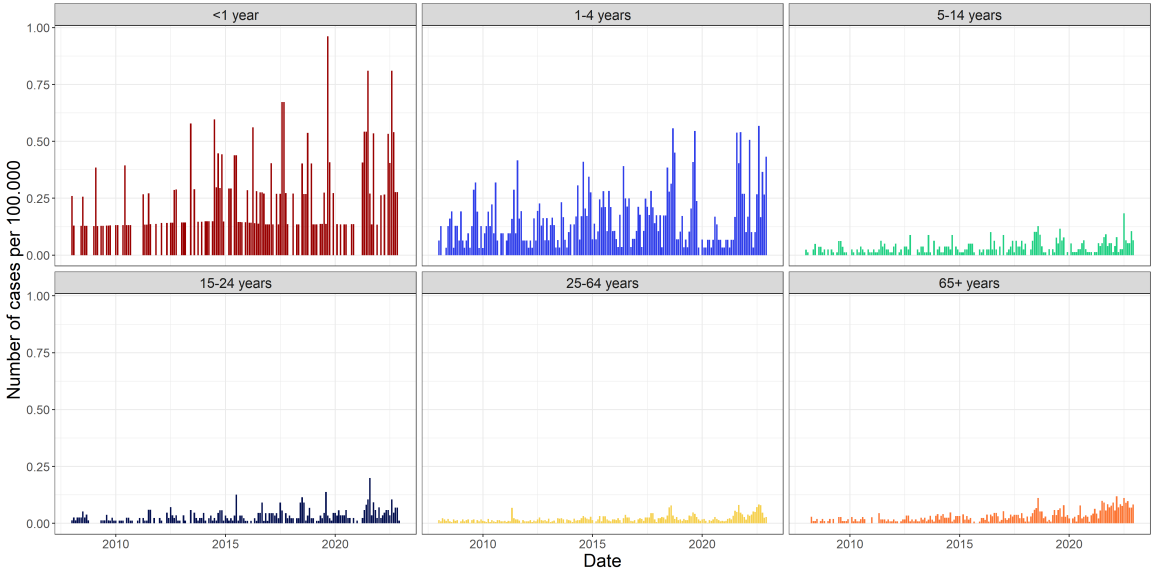


Figure: Epidemic curve showing the incidence per 100,000 in Denmark, 2012-2022, for the subset of diseases considered in this master thesis. (a) Listeriosis, (b) Shigellosis, (c) STEC, and (d) Salmonellosis.

Focusing on STEC



Novel outbreak detection algorithm

Algorithm 1: Test

Data: This text
Result: Hello
for $i = 1$ **to** T **do**
  Do something
end

Model formulas

$$\log(\lambda_{it}) = \boldsymbol{x}_{it}^T \boldsymbol{\beta} + \log(n_{it}), \quad i = 1, \ldots, k, \quad t = 1, \ldots, T \tag{1}$$

Agegroup

$$\log(\lambda_{it}) = \beta(ageGroup_i) + \log(n_{it}) \tag{2}$$

Agegroup and seasonality

$$\log(\lambda_{it}) = \beta(ageGroup_i)\beta_i + \sin\left(\frac{\pi \cdot monthInYear_t}{6}\right)\beta_{\sin} + \cos\left(\frac{\pi \cdot monthInYear_t}{6}\right)\beta_{\cos} + \log(n_{it}) \quad (3)$$

Hierarchical Poisson Normal model

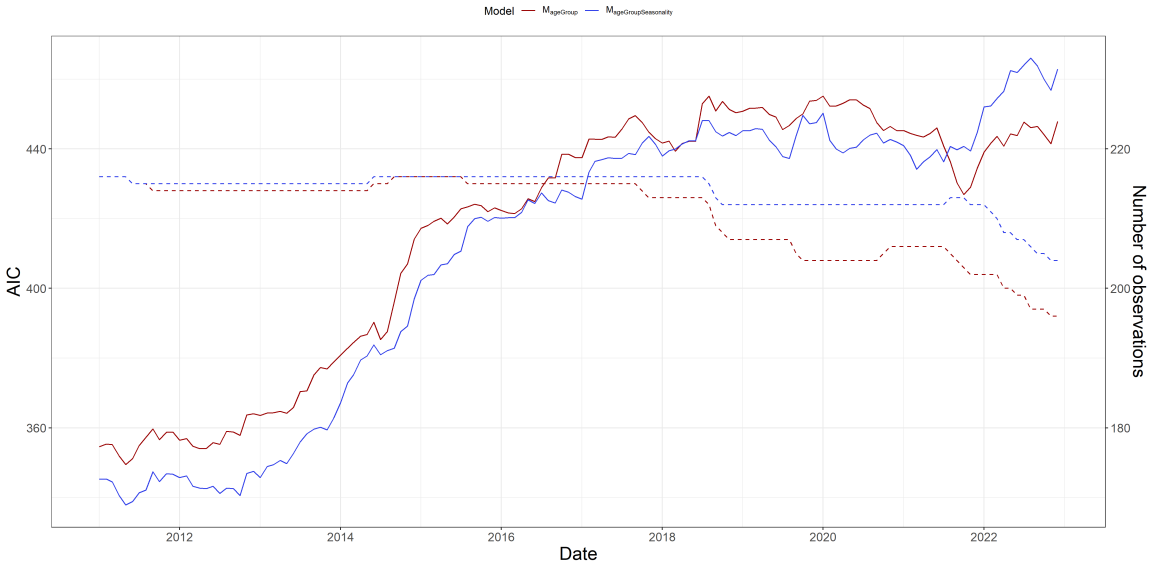
$$\boldsymbol{Y}|\boldsymbol{u} \sim \text{Pois}\left(\boldsymbol{\lambda}\exp(\boldsymbol{u})\right) \tag{4a}$$

$$\boldsymbol{u} \sim \text{N}(\boldsymbol{0}, I\sigma^2) \tag{4b}$$

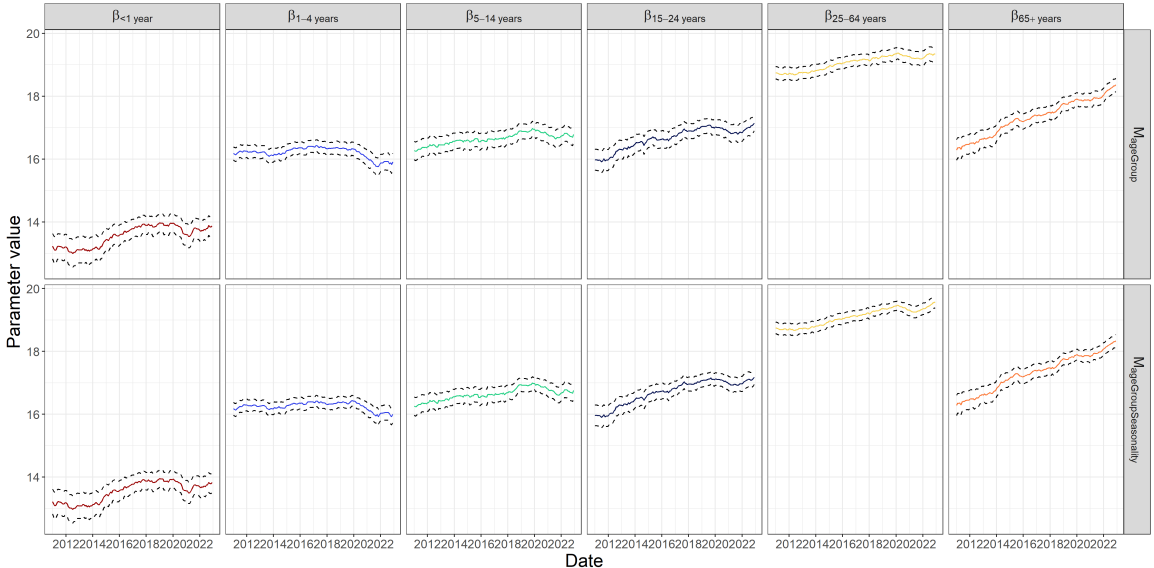
Implementation

```
#include <TMB.hpp>              // Links in the TMB libraries

template<class Type>
Type objective_function<Type>::operator() ()
{
  DATA_VECTOR(y);                                 // Data vector transmitted from R
  DATA_VECTOR(x);                           // Data vector transmitted from R
  DATA_MATRIX(X);                           // Design matrix transmitted from R

  PARAMETER_VECTOR(u);                          // Random effects

  // Parameters
  PARAMETER_VECTOR(beta);          // Parameter value transmitted from R
  PARAMETER(log_sigma_u);                   // Parameter value transmitted from R

  vector<Type> lambda  = exp(X*beta-log(x)+u);
  Type sigma_u = exp(log_sigma_u);

  int nobs = y.size();
  Type mean_ran = Type(0);

  Type f = 0;                               // Declare the "objective function"
  for(int t=0; t < nobs; t++){
    f -= dnorm(u[t],mean_ran,sigma_u,true);
    f -= dpois(y[t],lambda[t],true);
  }

  return f;
}
```

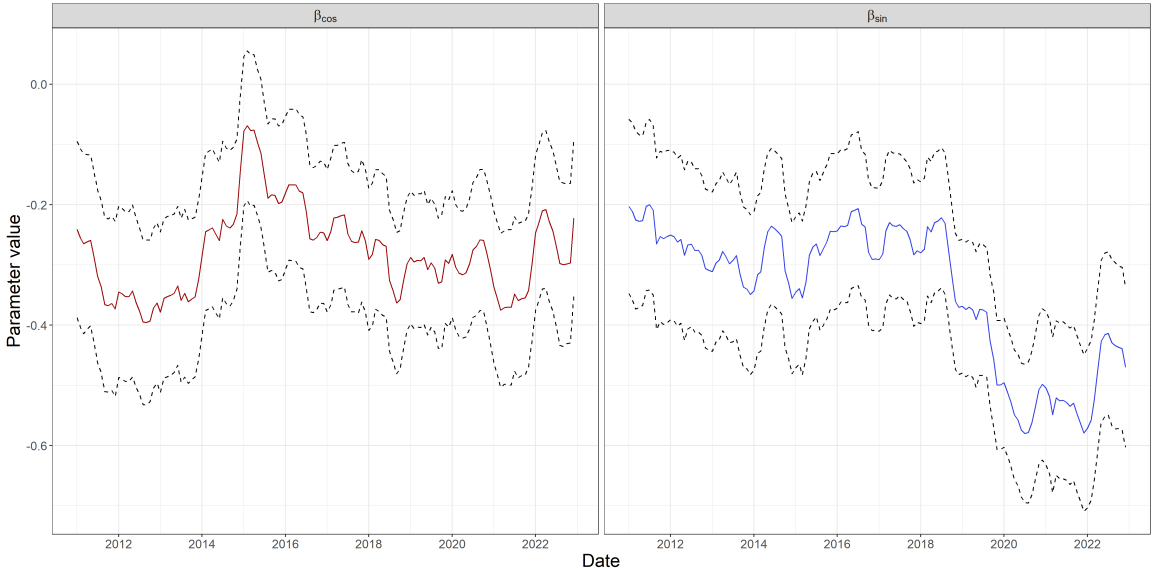
Model performance



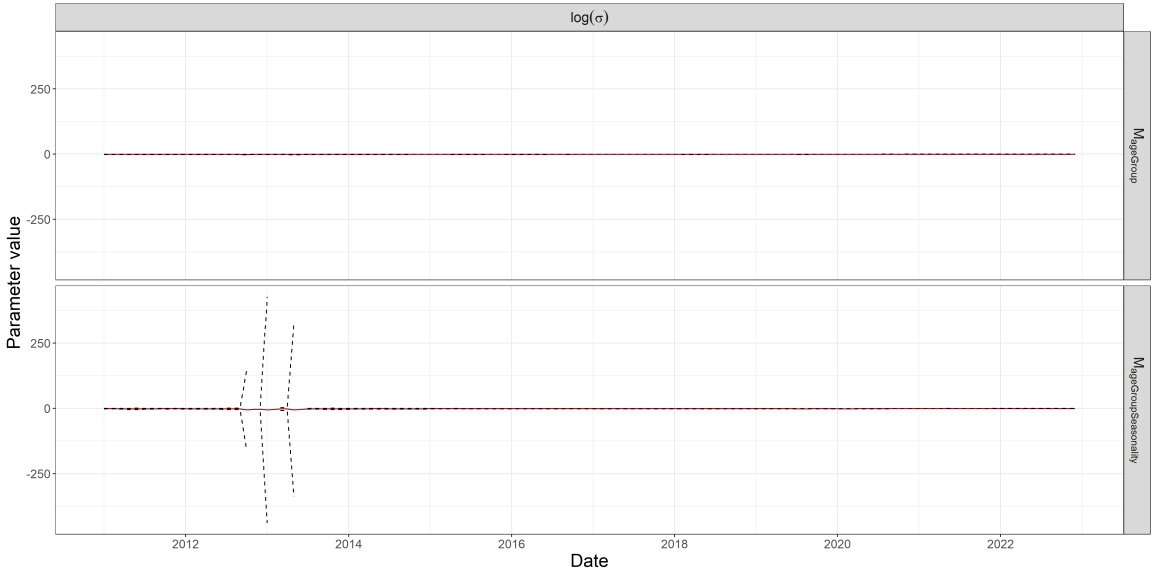
Agegroup parameters



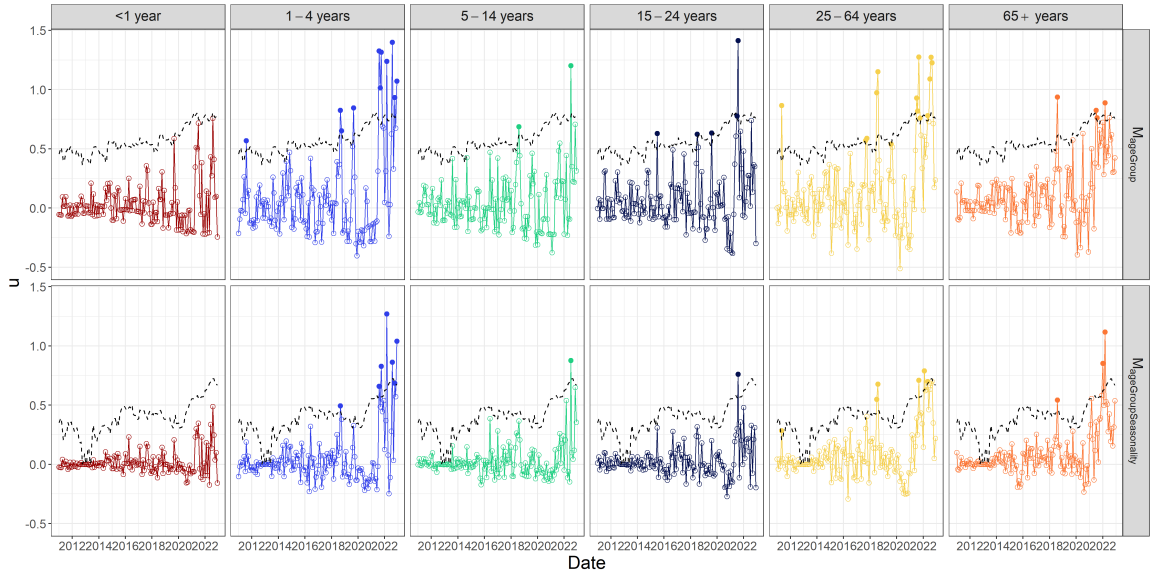
Seasonality parameters



Variance parameter



Outbreak detection



Hierarchical Poisson Gamma model

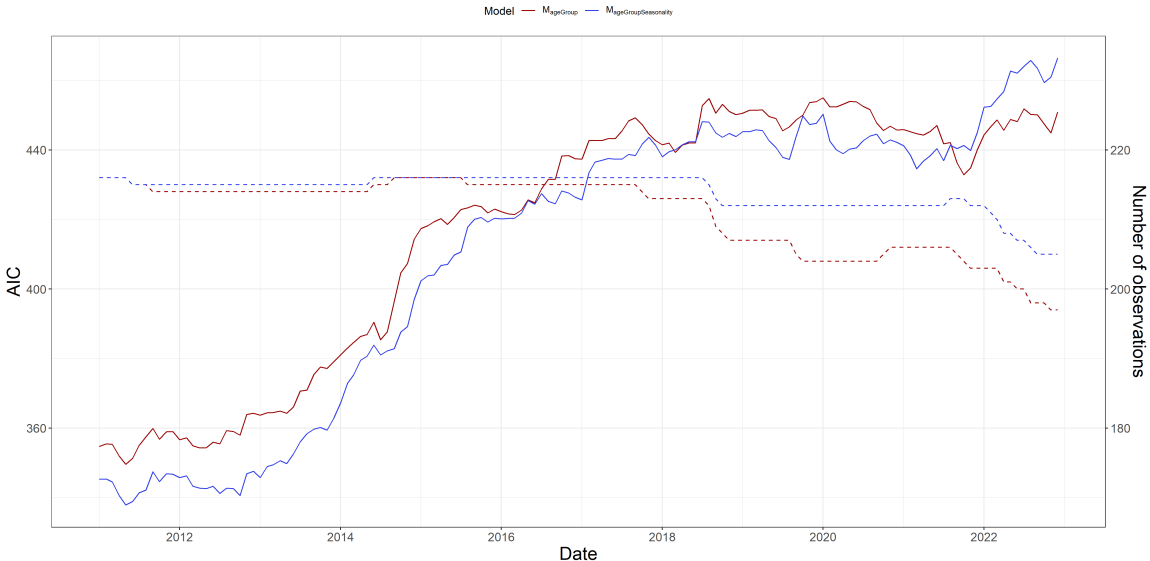
$$\boldsymbol{Y}|\boldsymbol{u} \sim \text{Pois}(\boldsymbol{\lambda u}) \tag{5a}$$

$$\boldsymbol{u} \sim \text{G}(\mathbf{1}/\phi, \phi) \tag{5b}$$

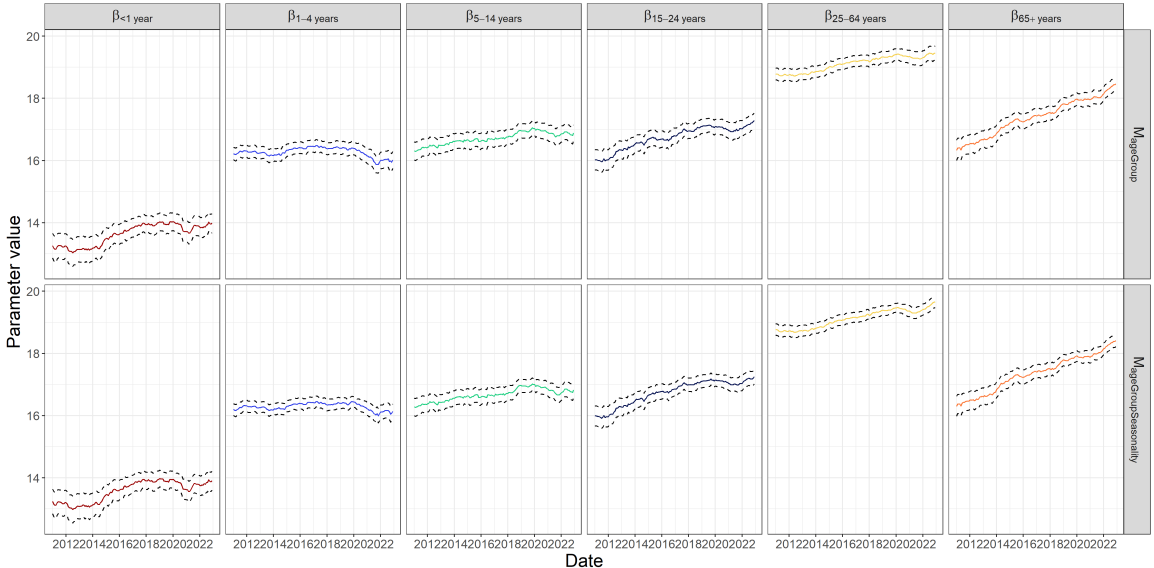
Implementation

```cpp
#include <TMB.hpp>              // Links in the TMB libraries

template<class Type>
Type objective_function<Type>::operator() ()
{
  DATA_VECTOR(y);                                 // Data vector transmitted from R
  DATA_VECTOR(x);                           // Data vector transmitted from R
  DATA_MATRIX(X);                           // Design matrix transmitted from R

  // Parameters
  PARAMETER_VECTOR(beta);          // Parameter value transmitted from R
  PARAMETER(log_phi_u);              // Parameter value transmitted from R

  vector<Type> lambda  = exp(X*beta-log(x)); // Construct the model parameters
  Type phi_u = exp(log_phi_u); // ... and the model parameters

  Type r = 1/phi_u; // Construct the size
  vector<Type> p = 1/(lambda*phi_u+1); // ... and the probability parameter

  Type f = -sum(dnbinom(y, r, p,true)); // Calculate the "objective function"

  return f;
}
```

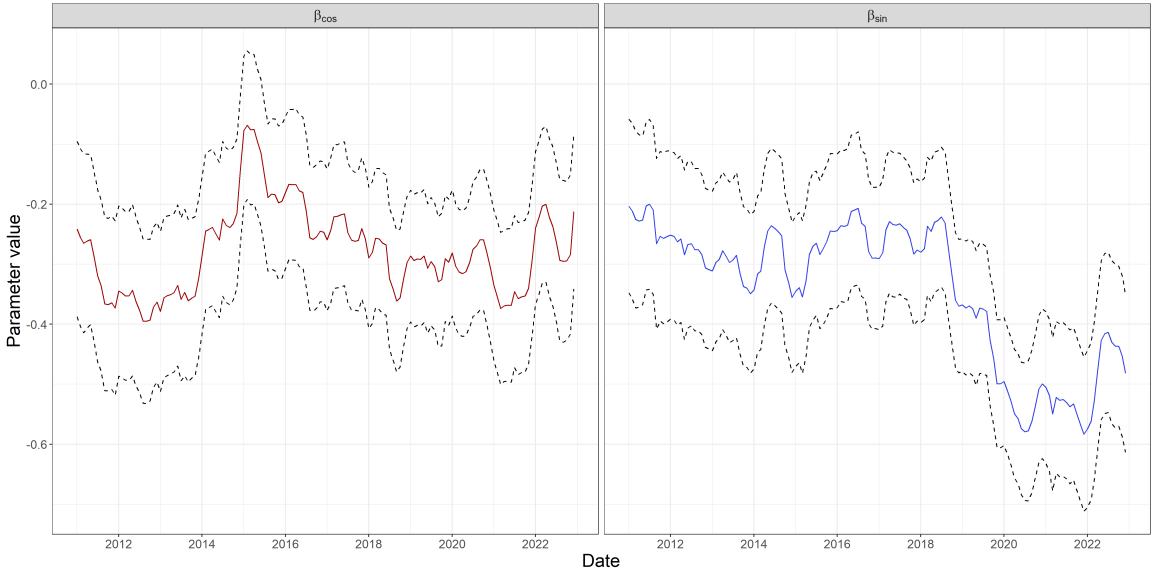
Model performance



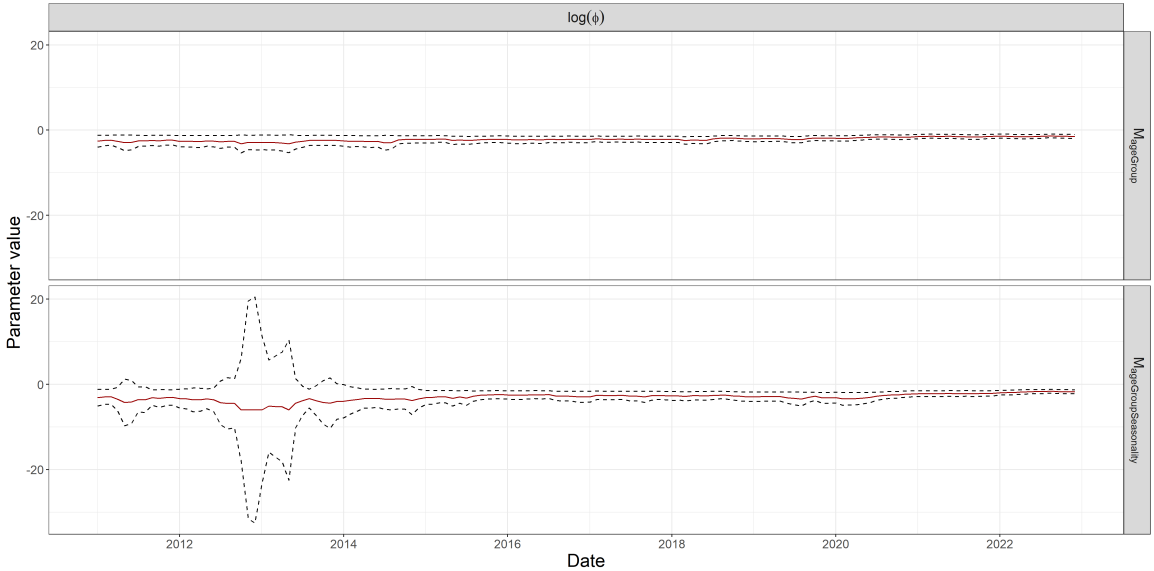
Agegroup parameters



Seasonality parameters



Variance parameter



Outbreak detection