A Comparative Study of Techniques for Estimation and Inference of Nonlinear Stochastic Time Series

Dexter Barrows, B.Sc.

Masters Thesis Defence
McMaster University
April 2016

1. Framing
2. Stochastic SIR Model
3. Hamiltonian HMCMC
4. Iterated Filtering 2
5. Forecasting Frameworks
6. S-maps & Seasonal Outbreaks
7. Spatiotemporal Epidemics
8. Parallelism & Future Directions

Framing

1


SIResampling
ParticleMCMC
ParticleFilter
Dewdrop
SIS
SeasonalARIMA
EnsembleKF
GARMA
DewdropRegression
ASIR
IF
RPF
MIF2
HMM
RHF
GLM
ParticleFilter
Simplex
ABC
S-map
SGEnKF
KalmanFilter
SimplexProjection
MCMC
HamiltonianMCMC
ARIMA

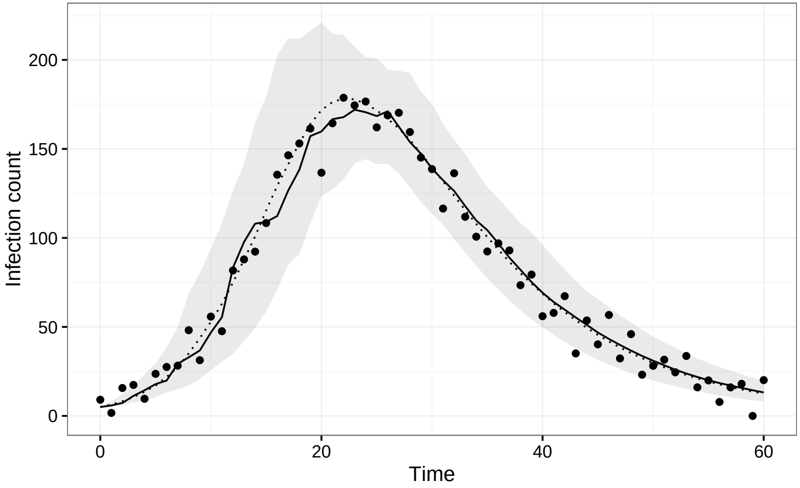
2 Stochastic SIR Model

Stochastic SIR model

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

$$+$$

$$\beta_{t+1} = \exp\left[\beta_t + \eta\left(\bar{\beta} - \beta_t\right) + \mathcal{N}(0, \sigma_{proc})\right]$$



Infection count
200
150
100
50
0
0
20
40
60
Time

3 Hamiltonian MCMC

MCMC

Iteratively construct Markov chain to approximate posterior

1. Choose starting parameter set
2. Generate N samples by
   2.1 Propose new sample
   2.2 Compute acceptance ratio
   2.3 Accept/reject sample

/* Select a starting point */
Input : Initialize $\theta^{(1)}$
for $i = 2 : N$ **do**
  /* Sample */
  $\theta^* \sim q(\cdot|\theta^{(i-1)})$
  $u \sim \mathcal{U}(0, 1)$
  /* Evaluate acceptance ratio */
  $r \leftarrow \frac{\mathcal{L}(\theta^*)p(\theta^*)}{\mathcal{L}(\theta)p(\theta)}$
  /* Step acceptance criterion */
  if $u < \min\{1, r\}$ **then**
    $\theta^{(i)} = \theta^*$
  else
    $\theta^{(i)} = \theta^{(i-1)}$

/* Samples from approximated posterior distribution */
Output: Chain of samples $(\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(N)})$

Hamiltonian Dynamics

Energy

Kinetic $$K(r) = \frac{1}{2} r^T M^{-1} r$$

Potential $$U(\theta) = -\log(\mathcal{L}(\theta)p(\theta))$$

Hamiltonian $$H(\theta, r) = U(\theta) + K(r)$$

Dynamics simulation
$$\frac{d\theta}{dt} = M^{-1} r$$
$$\frac{dr}{dt} = -\nabla U(\theta)$$

HMCMC algorithm

1. Choose starting parameter set
2. Generate N samples by
   2.1 Resampling moments
   2.2 Simulate Hamiltonian dynamics using Leapfrog integration
   2.3 Compute acceptance ratio
   2.4 Accept/reject sample

/* Select a starting point */

Input : Initialize $\theta^{(1)}$

for $i = 2 : N$ **do**

  /* Resample moments */

  for $i = 1 : n$ **do**

    $r(i) \leftarrow \mathcal{N}(0, 1)$

  /* Leapfrog initialization */

  $\theta_0 \leftarrow \theta^{(i-1)}$

  $r_0 \leftarrow r - \nabla U(\theta_0) \cdot \varepsilon/2$

  /* Leapfrog intermediate steps */

  for $j = 1 : L - 1$ **do**

    $\theta_j \leftarrow \theta_{j-1} + M^{-1} r_{j-1} \cdot \varepsilon$

    $r_j \leftarrow r_{j-1} - \nabla U(\theta_j) \cdot \varepsilon$

  /* Leapfrog last steps */

  $\theta^* \leftarrow \theta_{L-1} + M^{-1} r_{L-1} \cdot \varepsilon$

  $r^* \leftarrow \nabla U(\theta_L) \cdot \varepsilon/2 - r_{L-1}$

  /* Evaluate acceptance ratio */

  $r = \exp\left[H(\theta^{(i-1)}, r) - H(\theta^*, r^*)\right]$

  /* Sample */

  $u \sim \mathcal{U}(0, 1)$

  /* Step acceptance criterion */

  if $u < \min\{1, r\}$ **then**

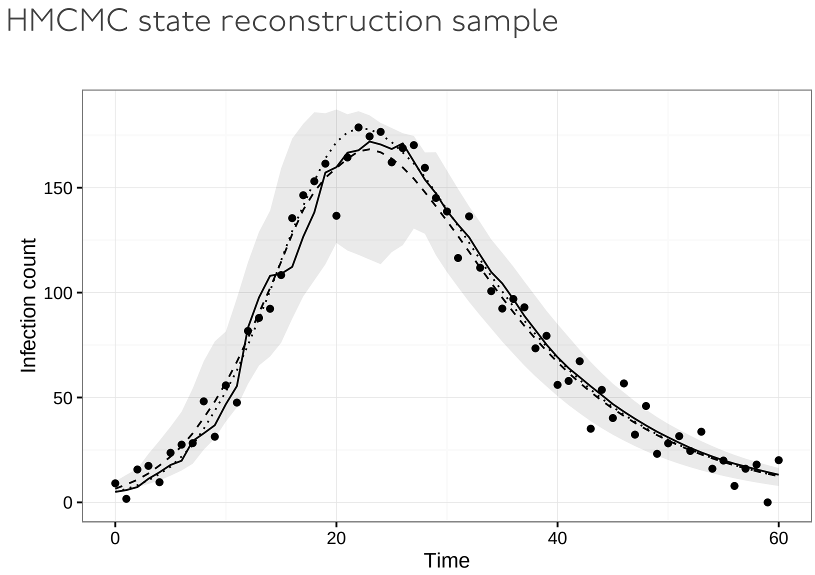
    $\theta^{(i)} = \theta^*$

  else

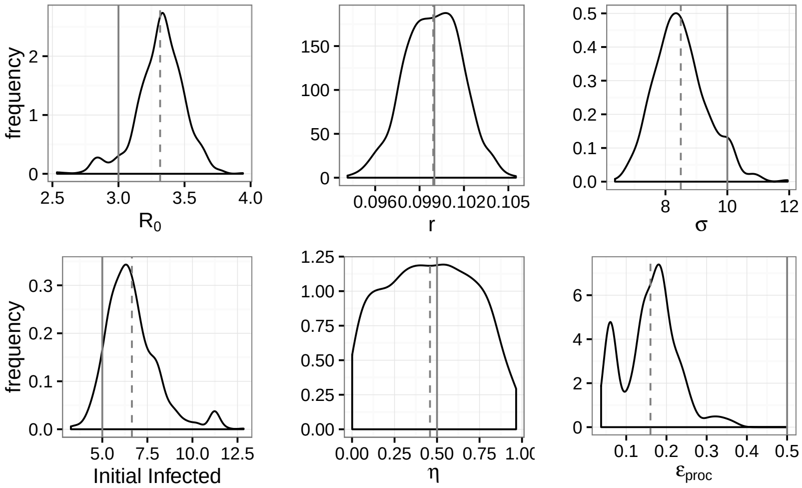
    $\theta^{(i)} = \theta^{(i-1)}$

/* Samples from approximated posterior distribution */

Output: Chain of samples $(\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(N)})$


HMCMC state reconstruction sample
Infection count
Time
0
50
100
150
0
20
40
60

HMCMC parameter estimation kernels sample



Iterated Filtering 2

4

Basic Particle Filter

Iterative prediction-update cycle prunes particle cohort of poor parameter estimates

1. Initialize particles with parameter sets
2. For each data point
   - 2.1 Evolve particle states
   - 2.2 Weight via likelihood
   - 2.3 Resample proportional to weights

/* Select a starting point */
Input : Observations $D = y_1, y_2, \ldots, y_T$, initial particle distribution P_0 of size J

/* Setup */
Initialize particle cohort by sampling $(p^{(1)}, p^{(2)}, \ldots, p^{(J)})$ from P_0

for $t = 1 : T$ **do**

  /* Evolve */
  for j = 1:J **do**
    $X_t^{(j)} \leftarrow f_1(X_{t-1}^{(j)}, \theta^{(j)})$

  /* Weight */
  for j = 1:J **do**
    $w^{(j)} \leftarrow P(y_t | X_t^{(j)}, \theta^{(j)}) = f_2(X_t^{(j)}, \theta^{(j)})$

  /* Normalize */
  for j = 1:J **do**
    $w^{(j)} \leftarrow w^{(j)} / \sum_1^J w^{(j)}$

  /* Resample */
  $p^{(1:J)} \leftarrow \text{sample}(p^{(1:J)}, \text{prob} = w, \text{replace} = \text{true})$

/* Samples from approximated posterior distribution */
Output: Cohort of posterior samples $(\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(J)})$

Iterated Filtering 2

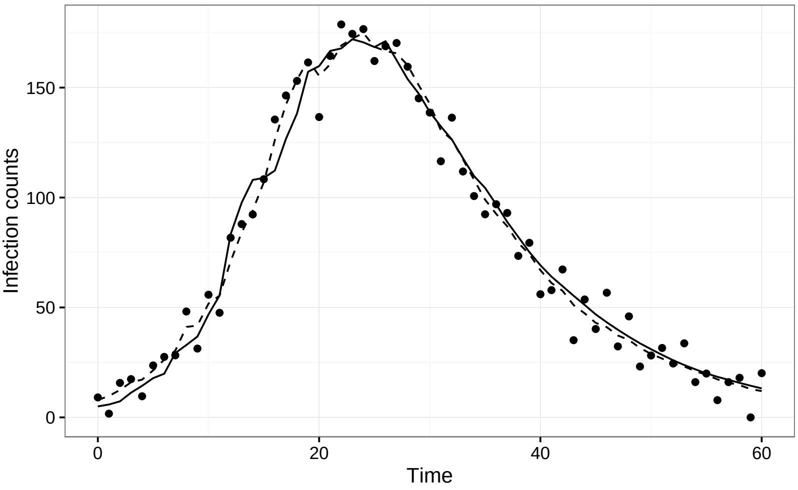
Treat parameter estimates as stochastic processes

Multiple passes through data (a la Data cloning)

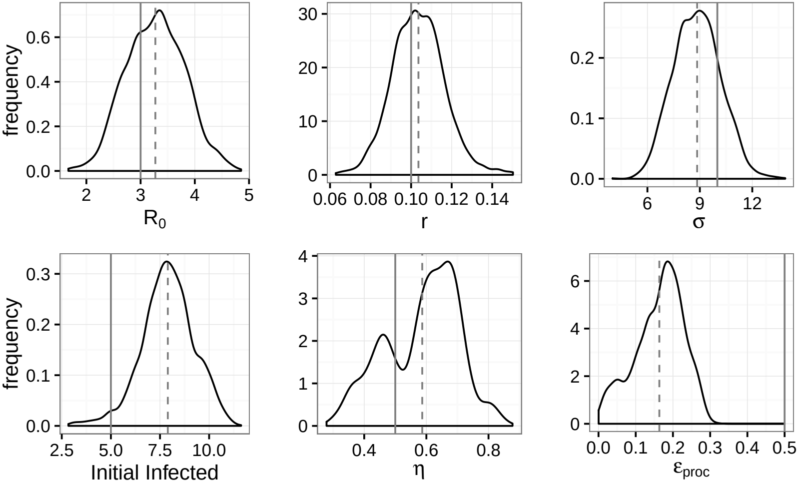
/* Setup */
Initialize particle cohort by sampling $(p^{(1)}, p^{(2)}, \ldots, p^{(J)})$ from P_0
/* Particle seeding distribution */
$\Theta \leftarrow P_0$
for $m = 1 : M$ **do**
    /* Pass perturbation */
    for j = 1:J **do**
        $p^{(j)} \sim h(\Theta^{(j)}, \sigma_m)$
    for $t = 1 : T$ **do**
        for j = 1:J **do**
            /* Iteration perturbation */
            $p^{(j)} \sim h(p^{(j)}, \sigma_m)$
            /* Evolve */
            $X_t^{(j)} \leftarrow f_1(X_{t-1}^{(j)}, \theta^{(j)})$
            /* Weight */
            $w^{(j)} \leftarrow P(y_t | X_t^{(j)}, \theta^{(j)}) = f_2(X_t^{(j)}, \theta^{(j)})$
        /* Normalize */
        for j = 1:J **do**
            $w^{(j)} \leftarrow w^{(j)} / \sum_1^J w^{(j)}$
        /* Resample */
        $p^{(1:J)} \leftarrow \text{sample}(p^{(1:J)}, \text{prob} = w, \text{replace} = \text{true})$
    /* Collect particles for next pass */
    for $j = 1 : J$ **do**
        $\Theta^{(j)} \leftarrow p^{(j)}$

/* Samples from approximated posterior distribution */
Output: Cohort of posterior samples $(\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(J)})$

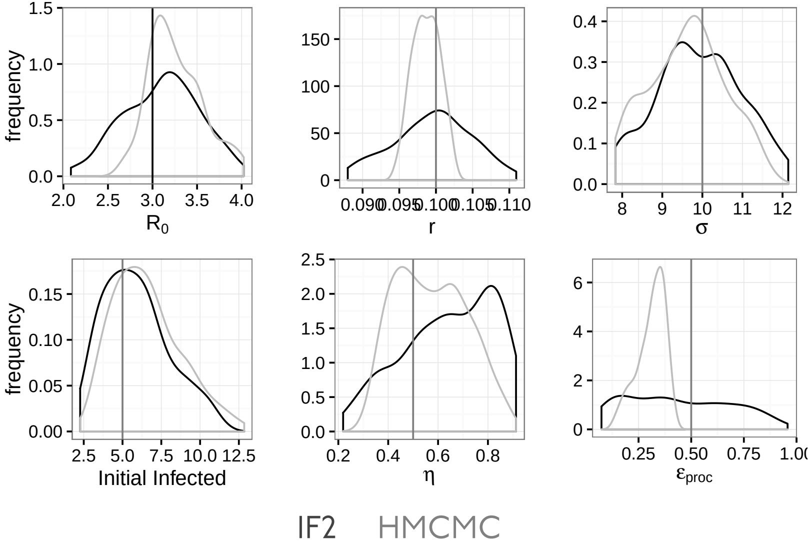
IF2 state reconstruction sample



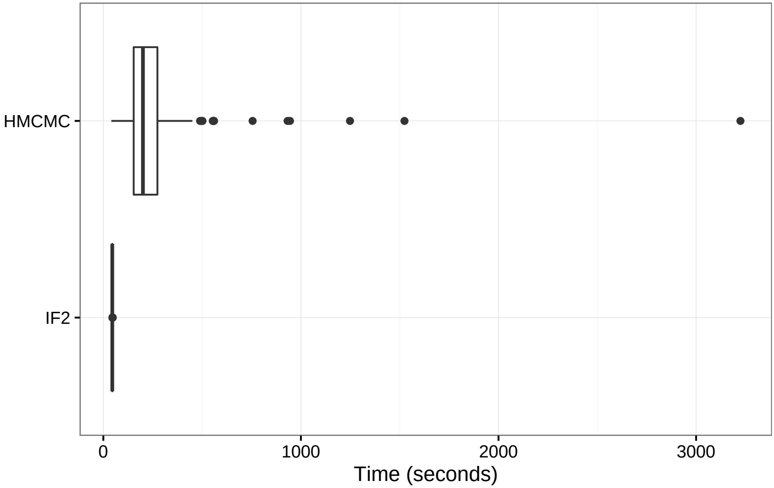
IF2 parameter estimation kernels sample



Mean parameter estimation distributions



Running times



5

Forecasting Frameworks

IF2 + Parametric bootstrapping + Forward simulation

- Provides additional samples from posterior distribution
- Forward simulation using states point estimate, posterior samples

Input : Forward simulator $S(\theta)$, data set D

/* Initial fit */
$\theta^* \leftarrow \mathrm{IF2}(D)$

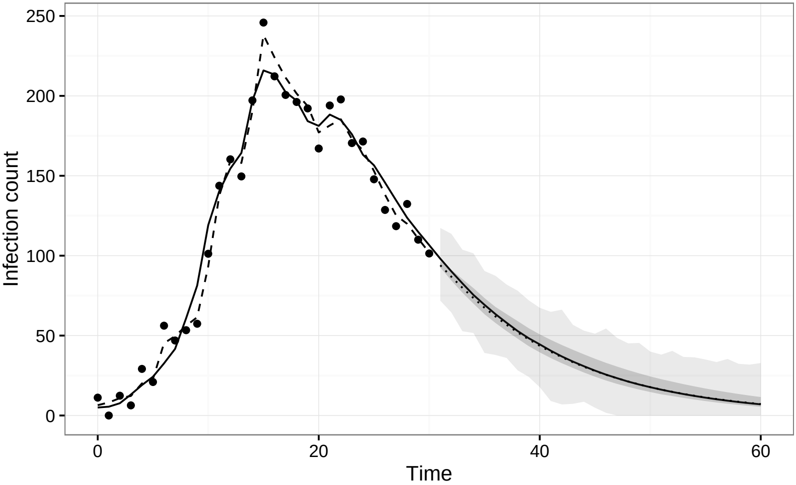
/* Generate artificial data sets */
for $i = 1 : M$ **do**
  $D_i \leftarrow S(\theta^*)$

/* Fit to new data sets */
for $i = 1 : M$ **do**
  $\theta_i \leftarrow \mathrm{IF2}(D_i)$

/* Simulate forward to produce forecasts */
for $i = 1 : M$ **do**
  $F_i \leftarrow S(D, \theta_i)$

Output: Forecast trajectories $F_1, F_2, \ldots, F_M$

IF2 parametric bootstrapping forecast



HMCMC + State reconstructions + Forward simulation

- Reconstruct states using latent process noise samples
- Simulate forward

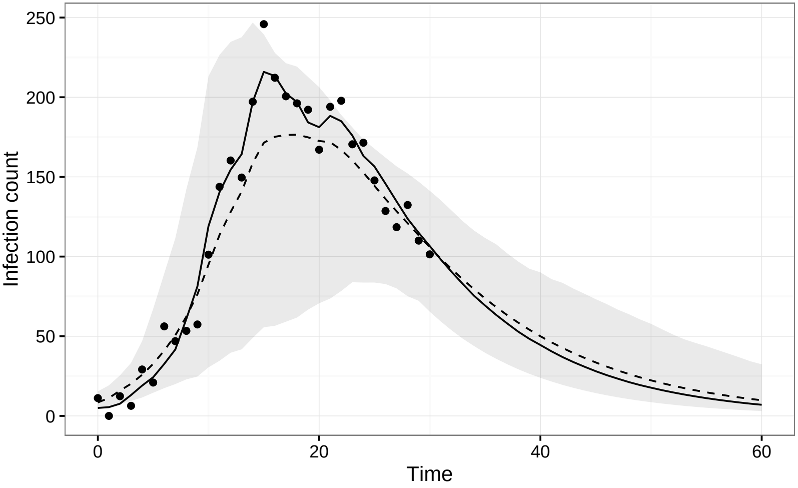
/* Sample from the posterior */
Input : Posterior samples $\theta_1, \theta_2, \ldots, \theta_M$

/* Reconstruct state estimates */
for $i = 1 : M$ **do**
    $D_i \leftarrow S(\theta_i)$

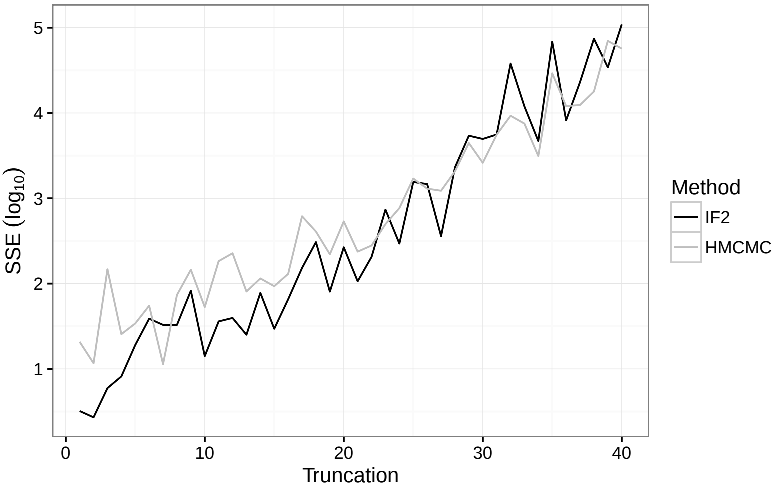
/* Simulate forward to produce forecasts */
for $i = 1 : M$ **do**
    $F_i \leftarrow S(D_i, \theta_i)$

Output: Forecast trajectories $F_1, F_2, \ldots, F_M$

HMCMC parametric bootstrapping forecast



Forecast accuracy comparison



6

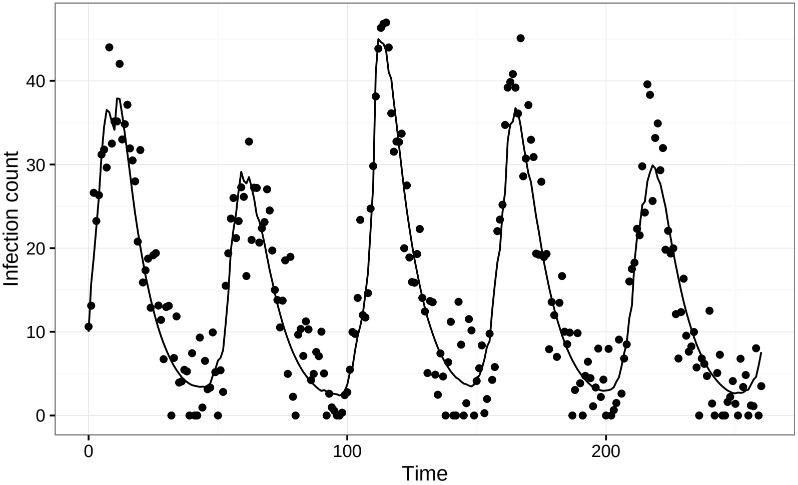
S-maps & Seasonal Outbreaks

Stochastic SIRS model

$$\frac{dS}{dt} = -\beta SI + \alpha R$$
$$\frac{dI}{dt} = \beta SI - \gamma I$$
$$\frac{dR}{dt} = \gamma I - \alpha R$$

$$+$$

$$\beta_{t+1} = \exp\left[\beta_t + \eta\left(\bar{\beta} - \beta_t\right) + \mathcal{N}(0, \sigma_{\text{proc}})\right]$$


Infection count
0
10
20
30
40
0
100
200
Time

S-map

- Construct global mapping of time-lagged vectors (the library) to future states
- Weightings are used to penalize poorly matching library vectors

/* Select a starting point */

Input : Time series $x_1, x_2, \ldots, x_T$, embedding dimension E, distance penalization θ, forecast length L, predictor vector $\mathbf{x_t}$

/* Construct library $\{x_i\}$ */

for $i = E : T$ **do**

  $x_i = (x_i, x_{i-1}, \ldots, x_{i-E-1})$

/* Construct mapping from library vectors to predictions */

for $i = 1 : (T_E + 1)$ **do**

  for $j = 1 : E$ **do**

    $A(i, j) = w(||x_i - \mathbf{x_t}||)x_i(j)$

for $i = 1 : (T_E + 1)$ **do**

  $b(i) = w(||x_i - \mathbf{x_t}||)y_i$

/* Use SVD to solve the mapping system, Ac = b */

$\mathrm{SVD}(Ac = b)$

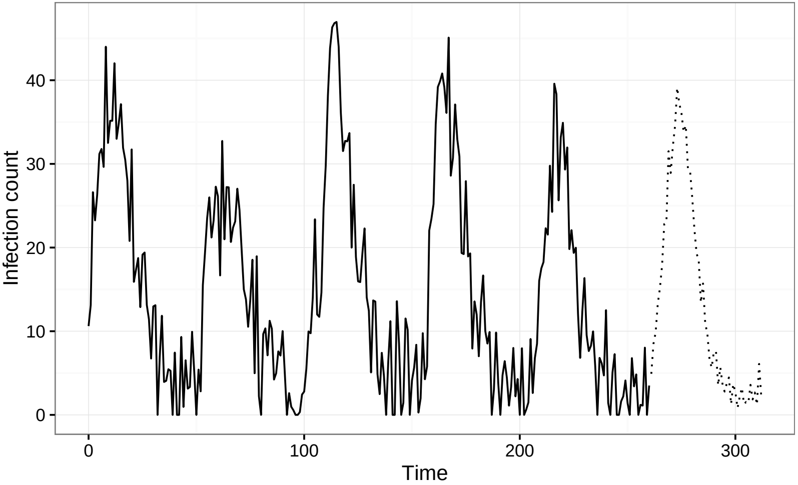
/* Compute forecast */

$\hat{y_t} = \sum_{j=0}^{E} c_t(j)\mathbf{x_t}(j)$

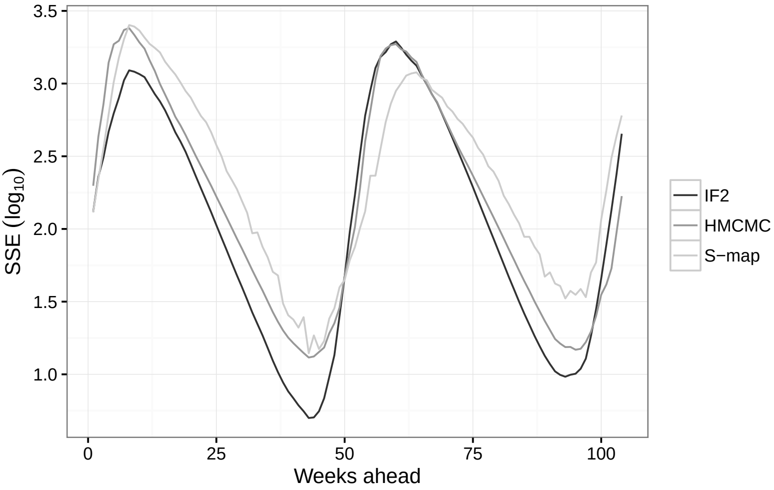
/* Forecasted value in time series */

Output: Forecast $\hat{y_t}$

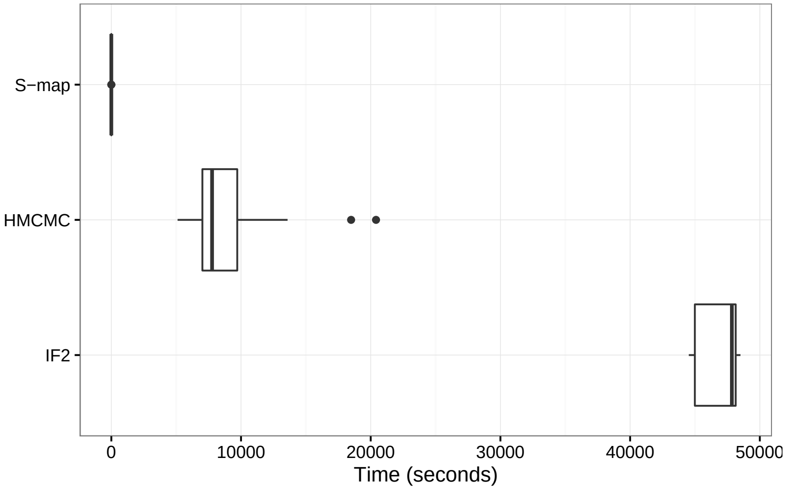
S-map forecast



SIRS model forecasting error



SIRS model forecasting runtimes



S-map: 316,000x faster than IF2, 61,800x faster than HMCMC

7

Spatiotemporal Epidemics

Stochastic Spatial SIR model

$$\frac{dS_i}{dt} = -\left(1 - \phi\frac{M}{M+1}\right)\beta_i S_i I_i - \frac{\phi}{M+1} S_i \sum_{j=1}^{M} \beta_{ij} I_j$$

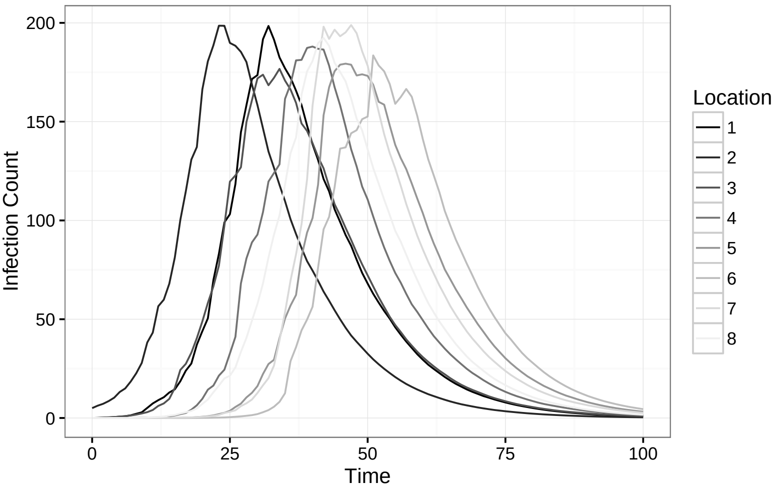
$$\frac{dI_i}{dt} = \left(1 - \phi\frac{M}{M+1}\right)\beta_i S_i I_i + \frac{\phi}{M+1} S_i \sum_{j=1}^{M} \beta_{ij} I_j - \gamma I_i$$

$$\frac{dR_i}{dt} = \gamma I_i$$

$$+$$

$$\beta_{i,t+1} = \exp\left[\beta_{i,t} + \eta\left(\bar{\beta} - \beta_{i,t}\right) + \mathcal{N}(0, \sigma_{proc})\right]$$

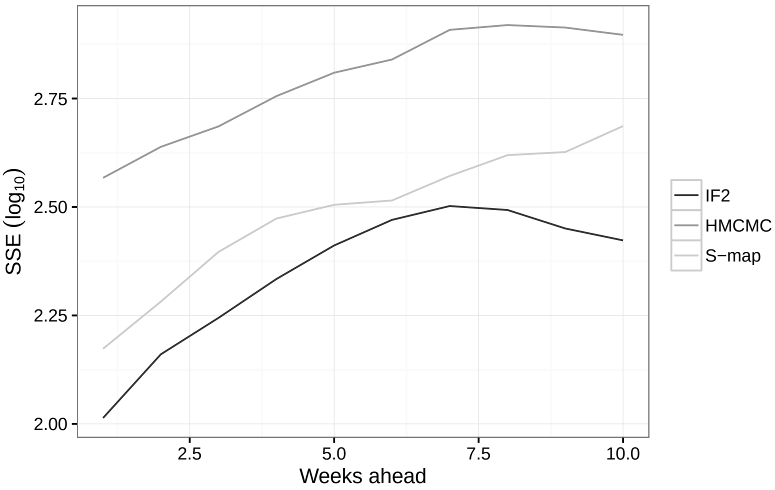
Stochastic spatial SIR model simulation (ring topology)

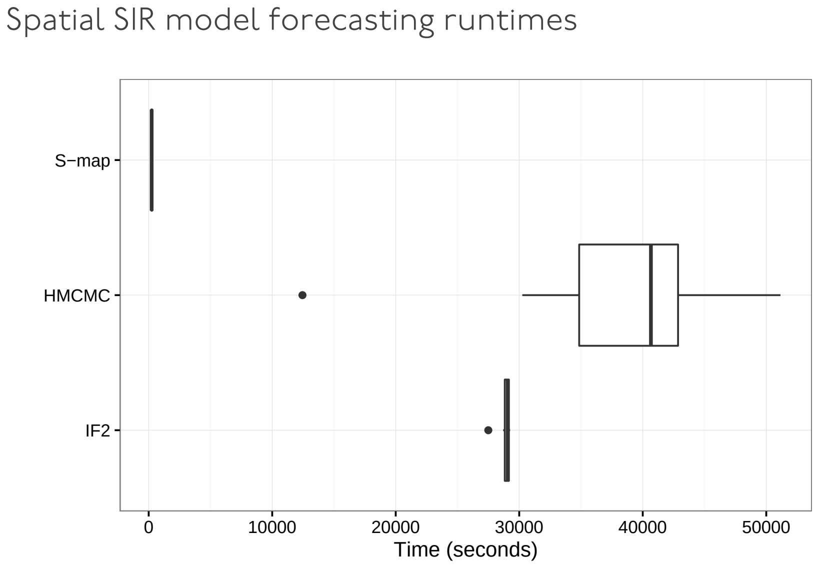


Dewdrop Regression

- “Stitching” together multiple short time series into single library
- Requires scaling

Stochastic SIR model forecasting error




Spatial SIR model forecasting runtimes
S−map
HMCMC
IF2
0
10000
20000
30000
40000
50000
Time (seconds)

8

Parallelism & Future Directions

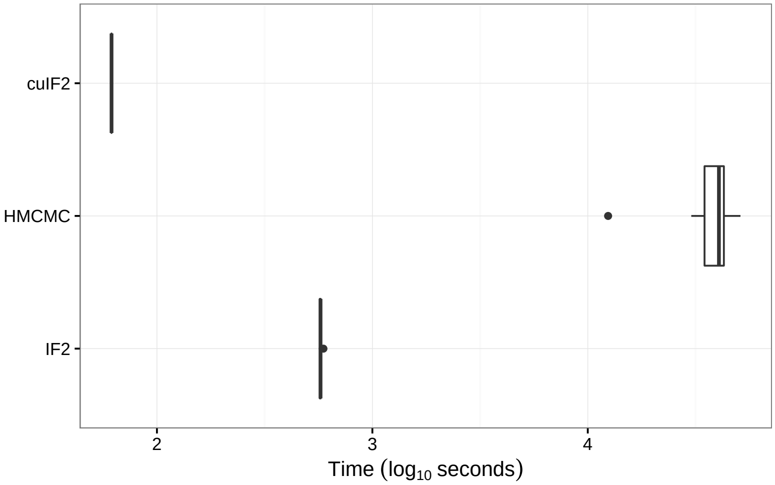
Conclusions

- IF2 produces superior forecasts in all scenarios
- S-mapping runs orders of magnitude faster than other methods

More than Moore

- Moore's Law is ceasing to hold
- Focus now on distributed computing
- MCMC-based methods are resistant to parallelization
    - Chain construction requires iterative dependence
- IF2 exhibits high parallel potential
    - Preliminary CUDA (GPU-accelerated) implementation - `cuIF2`

Spatial SIR model fitting times



`cuIF2`: 9.33x faster than IF2, 617x faster than HMCMC

THE BEST THESIS DEFENSE IS A GOOD THESIS OFFENSE.