Short-term Traffic Flow Forecasting and Sequential Inference

Tiep Mai, Simon P. Wilson

Trinity College Dublin, Ireland.

TRINITY COLLEGE DUBLIN

Fondúireacht Eolaíochta Éireann
Science Foundation Ireland

Outline

- Bayesian inference and MCMC
- STFF with VARMA model
- Sequential inference for dynamic models
- A new spatial temporal model for STFF

Bayesian inference

- *Bayes's theorem*:

$$p_{\theta|y}(\theta) = \frac{p_\theta(\theta)p_{y|\theta}(y)}{p_y(y)} \propto p_\theta(\theta)p_{y|\theta}(y)$$

Bayesian inference

- *Bayes's theorem*:

$$p_{\theta|y}(\theta) = \frac{p_\theta(\theta) p_{y|\theta}(y)}{p_y(y)} \propto p_\theta(\theta) p_{y|\theta}(y)$$

- The likelihood $p_{y|\theta}(y)$: $cost = \theta \times dist + noise$

Bayesian inference

- *Bayes's theorem*:

$$p_{\theta|y}(\theta) = \frac{p_\theta(\theta)p_{y|\theta}(y)}{p_y(y)} \propto p_\theta(\theta)p_{y|\theta}(y)$$

- The likelihood $p_{y|\theta}(y)$: $cost = \theta \times dist + noise$
- The prior $p_\theta(\theta)$

Bayesian inference

- *Bayes's theorem*:

$$p_{\theta|y}(\theta) = \frac{p_\theta(\theta) p_{y|\theta}(y)}{p_y(y)} \propto p_\theta(\theta) p_{y|\theta}(y)$$

- The likelihood $p_{y|\theta}(y)$: $cost = \theta \times dist + noise$
- The prior $p_\theta(\theta)$
    - Expert opinion
    - Regularisation term

Bayesian inference

- *Bayes's theorem*:

$$p_{\theta|y}(\theta) = \frac{p_\theta(\theta)p_{y|\theta}(y)}{p_y(y)} \propto p_\theta(\theta)p_{y|\theta}(y)$$

- The likelihood $p_{y|\theta}(y)$: $cost = \theta \times dist + noise$
- The prior $p_\theta(\theta)$
  - Expert opinion
  - Regularisation term
- The posterior $p_{\theta|y}(\theta)$

Bayesian inference

- Analysing parameter θ:

$$\mathrm{E}_{\theta|y}(\theta) = \mu_{\theta|y} = \int \theta \; p_{\theta|y}(\theta) d\theta,$$

$$\mathrm{Var}_{\theta|y}(\theta) = \int (\theta - \mu_{\theta|y})^2 \; p_{\theta|y}(\theta) d\theta = \mathrm{E}_{\theta|y}[(\theta - \mu_{\theta|y})^2]$$

Bayesian inference

- Analysing parameter θ:

$$\mathrm{E}_{\theta|y}(\theta) = \mu_{\theta|y} = \int \theta \ p_{\theta|y}(\theta)d\theta,$$

$$\mathrm{Var}_{\theta|y}(\theta) = \int (\theta - \mu_{\theta|y})^2 \ p_{\theta|y}(\theta)d\theta = \mathrm{E}_{\theta|y}[(\theta - \mu_{\theta|y})^2]$$

- Let $\theta = (\theta_1, \theta_2)$. Marginalise out nuisance parameter θ_2:

$$p_{\theta_1|y}(\theta_1) = \int p_{\theta_1,\theta_2|y}(\theta_1, \theta_2)d\theta_2 = \mathrm{E}_{\theta_2|y}[p_{\theta_1|y,\theta_2}(\theta_1)]$$

Bayesian inference

- Analysing parameter θ:

$$\mathrm{E}_{\theta|y}(\theta) = \mu_{\theta|y} = \int \theta \; p_{\theta|y}(\theta) d\theta,$$

$$\mathrm{Var}_{\theta|y}(\theta) = \int (\theta - \mu_{\theta|y})^2 \; p_{\theta|y}(\theta) d\theta = \mathrm{E}_{\theta|y}[(\theta - \mu_{\theta|y})^2]$$

- Let $\theta = (\theta_1, \theta_2)$. Marginalise out nuisance parameter θ_2:

$$p_{\theta_1|y}(\theta_1) = \int p_{\theta_1,\theta_2|y}(\theta_1, \theta_2) d\theta_2 = \mathrm{E}_{\theta_2|y}[p_{\theta_1|y,\theta_2}(\theta_1)]$$

- Prediction:

$$p_{y_2|y}(y_2) = \int p_{y_2|\theta}(y_2) \; p_{\theta|y}(\theta) d\theta = \mathrm{E}_{\theta|y}[p_{y_2|\theta}(y_2)]$$

Bayesian inference

- Analysing parameter θ:

$$\mathrm{E}_{\theta|y}(\theta) = \mu_{\theta|y} = \int \theta \; p_{\theta|y}(\theta) d\theta,$$

$$\mathrm{Var}_{\theta|y}(\theta) = \int (\theta - \mu_{\theta|y})^2 \; p_{\theta|y}(\theta) d\theta = \mathrm{E}_{\theta|y}[(\theta - \mu_{\theta|y})^2]$$

- Let $\theta = (\theta_1, \theta_2)$. Marginalise out nuisance parameter θ_2:

$$p_{\theta_1|y}(\theta_1) = \int p_{\theta_1,\theta_2|y}(\theta_1, \theta_2) d\theta_2 = \mathrm{E}_{\theta_2|y}[p_{\theta_1|y,\theta_2}(\theta_1)]$$

- Prediction:

$$p_{y_2|y}(y_2) = \int p_{y_2|\theta}(y_2) \; p_{\theta|y}(\theta) d\theta = \mathrm{E}_{\theta|y}[p_{y_2|\theta}(y_2)]$$

- Integrations with no closed-form solutions

Monte Carlo methods

- By *the strong law of large number (SLLN)* and *the central limit theorem (CLT)*:

$$\mathrm{E}_x(f(x)) = \int f(x)\, p_x(x) dx \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i),$$

with $x_i \overset{iid}{\sim} p_x(\cdot)$.

Monte Carlo methods

- By *the strong law of large number (SLLN)* and *the central limit theorem (CLT)*:

$$\mathrm{E}_x(f(x)) = \int f(x)\; p_x(x)dx \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i),$$

  with $x_i \overset{iid}{\sim} p_x(\cdot)$.

- Sampling methods

Monte Carlo methods

- By *the strong law of large number (SLLN)* and *the central limit theorem (CLT)*:

$$\mathrm{E}_x(f(x)) = \int f(x)\; p_x(x)dx \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i),$$

with $x_i \overset{iid}{\sim} p_x(\cdot)$.

- Sampling methods
- *Markov chain Monte Carlo (MCMC)*

Monte Carlo methods

- By *the strong law of large number (SLLN)* and *the central limit theorem (CLT)*:

$$\mathrm{E}_x(f(x)) = \int f(x)\ p_x(x)dx \approx \frac{1}{n}\sum_{i=1}^{n} f(x_i),$$

with $x_i \overset{iid}{\sim} p_x(\cdot)$.

- Sampling methods
- *Markov chain Monte Carlo (MCMC)*
  - Construct a Markov chain that admits $p_x(x)$ as a stationary density
  - Starting at an initial point x_s, explore $p_x(x)$ by Markov chain transition density
  - Use trace points as samples from $p_x(x)$

MCMC

- *Metropolis-Hastings*

MCMC

- *Metropolis-Hastings*
  - Each movement from x_c to x_n has an acceptance rate $\alpha(x_c, x_n)$
  - $\alpha(x_c, x_n)$ is dependent on $p_x(x_n)/p_x(x_c)$
  - Movement distance and acceptance rate

MCMC

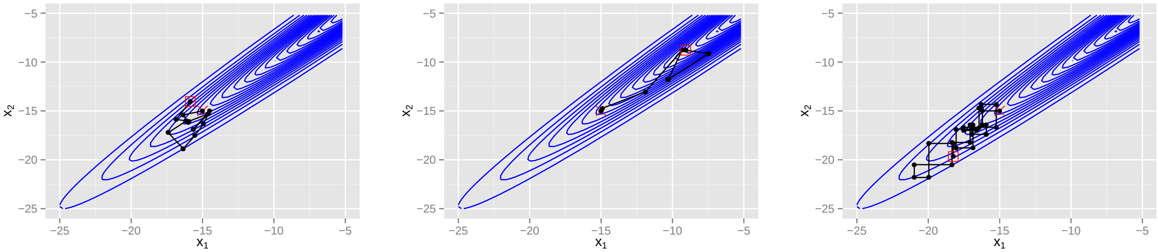
- *Metropolis-Hastings*
  - Each movement from x_c to x_n has an acceptance rate $\alpha(x_c, x_n)$
  - $\alpha(x_c, x_n)$ is dependent on $p_x(x_n)/p_x(x_c)$
  - Movement distance and acceptance rate
- *Gibbs sampling* on $p_{x_1,x_2}(x_1, x_2)$ with $x = (x_1, x_2)$

MCMC

- *Metropolis-Hastings*
  - Each movement from x_c to x_n has an acceptance rate $\alpha(x_c, x_n)$
  - $\alpha(x_c, x_n)$ is dependent on $p_x(x_n)/p_x(x_c)$
  - Movement distance and acceptance rate
- *Gibbs sampling* on $p_{x_1,x_2}(x_1, x_2)$ with $x = (x_1, x_2)$
  - Sample from $p_{x_1|x_2}(x_1)$
  - Sample from $p_{x_2|x_1}(x_2)$

MCMC and variable correlation

- Variable correlation



Summary

- Bayesian inference

Summary

- Bayesian inference
- Problems that are formulated as integrations w.r.t. the posterior

Summary

- Bayesian inference
- Problems that are formulated as integrations w.r.t. the posterior
- Monte Carlo methods

Summary

- Bayesian inference
- Problems that are formulated as integrations w.r.t. the posterior
- Monte Carlo methods
- MCMC and variable correlation

Introduction

- Intelligent Transportation System applications: route optimisation, system performance optimisation, accident detection

Introduction

- Intelligent Transportation System applications: route optimisation, system performance optimisation, accident detection
- Short-term Traffic Flow Forecasting (STFF)

Introduction

- Intelligent Transportation System applications: route optimisation, system performance optimisation, accident detection
- Short-term Traffic Flow Forecasting (STFF)
- Common approaches: ARMA and DLM

Introduction

- Intelligent Transportation System applications: route optimisation, system performance optimisation, accident detection
- Short-term Traffic Flow Forecasting (STFF)
- Common approaches: ARMA and DLM
- A new model, based on VARMA:

Introduction

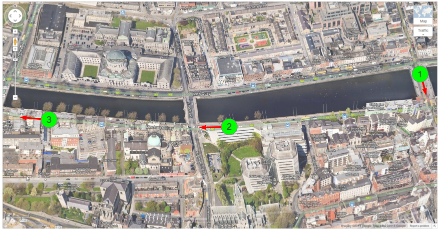
- Intelligent Transportation System applications: route optimisation, system performance optimisation, accident detection
- Short-term Traffic Flow Forecasting (STFF)
- Common approaches: ARMA and DLM
- A new model, based on VARMA:
  - Spatial dependency of upstream and downstream flows

Introduction

- Intelligent Transportation System applications: route optimisation, system performance optimisation, accident detection
- Short-term Traffic Flow Forecasting (STFF)
- Common approaches: ARMA and DLM
- A new model, based on VARMA:
    - Spatial dependency of upstream and downstream flows
    - Multi-step-ahead prediction

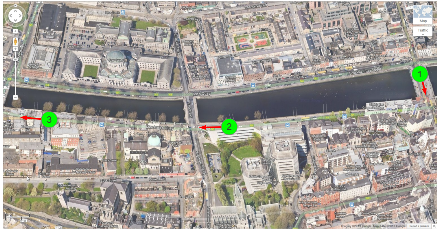
Vector Autoregressive Moving Average Model (VARMA)

- Traffic flows z_t of daily pattern with period s



Vector Autoregressive Moving Average Model (VARMA)

- Traffic flows z_t of daily pattern with period s



- Data preprocessing:

$$y_{SD;t} = \nabla_s z_t = z_t - z_{t-s},$$

$$y_{DSD;t} = \nabla \nabla_s z_t = (z_t - z_{t-s}) - (z_{t-1} - z_{t-s-1}),$$

$$y_{MP;t} = z_t - m_t.$$

Vector Autoregressive Moving Average Model (VARMA)

- Use VARMA with noise $e_t \overset{iid}{\sim} N(0, \Sigma_e)$ on residual y_t:

$$(y_t - \beta) - \sum_{j=1}^{p} \Phi_j (y_{t-j} - \beta) = e_t + \sum_{j'=1}^{q} \Theta_{j'} e_{t-j'}$$

Vector Autoregressive Moving Average Model (VARMA)

- Use VARMA with noise $e_t \overset{iid}{\sim} N(0, \Sigma_e)$ on residual y_t:

$$(y_t - \beta) - \sum_{j=1}^{p} \Phi_j (y_{t-j} - \beta) = e_t + \sum_{j'=1}^{q} \Theta_{j'} e_{t-j'}$$

- Sparse form of VARMA, e.g $\Phi_j = 0$ if $j \neq 1, s$.

Vector Autoregressive Moving Average Model (VARMA)

- Use VARMA with noise $e_t \overset{iid}{\sim} N(0, \Sigma_e)$ on residual y_t:

$$(y_t - \beta) - \sum_{j=1}^{p} \Phi_j (y_{t-j} - \beta) = e_t + \sum_{j'=1}^{q} \Theta_{j'} e_{t-j'}$$

- Sparse form of VARMA, e.g $\Phi_j = 0$ if $j \neq 1, s$.
- Network structure for variable dependency:

$$\Phi_1 = \begin{pmatrix} a & 0 & 0 \\ b & c & 0 \\ c & d & e \end{pmatrix}$$

VARMA-MCMC

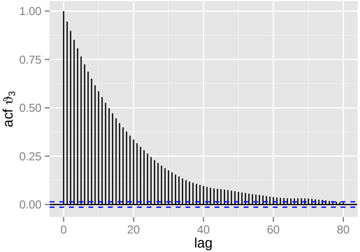
- MCMC on $(\varPhi, \varTheta, \beta, \varSigma_e)$

VARMA-MCMC

- MCMC on $(\Phi, \Theta, \beta, \Sigma_e)$
  - Sample β from $p_{\beta|\Phi,\Theta,\Sigma_e,y_{1:n}}(\cdot)$ from normal distribution
  - Sample Σ_e from $p_{\Sigma_e|\Phi,\Theta,\beta,y_{1:n}}(\cdot)$ from inverse Wishart distribution
  - Sample Φ from $p_{\Phi|\Theta,\beta,\Sigma_e,y_{1:n}}(\cdot)$ from normal distribution
  - Use Metropolis-Hastings on $p_{\Theta|\Phi,\beta,\Sigma_e,y_{1:n}}(\cdot)$

VARMA-MCMC

- MCMC on $(\Phi, \Theta, \beta, \Sigma_e)$
  - Sample β from $p_{\beta|\Phi,\Theta,\Sigma_e,y_{1:n}}(\cdot)$ from normal distribution
  - Sample Σ_e from $p_{\Sigma_e|\Phi,\Theta,\beta,y_{1:n}}(\cdot)$ from inverse Wishart distribution
  - Sample Φ from $p_{\Phi|\Theta,\beta,\Sigma_e,y_{1:n}}(\cdot)$ from normal distribution
  - Use Metropolis-Hastings on $p_{\Theta|\Phi,\beta,\Sigma_e,y_{1:n}}(\cdot)$
- Variable correlation between Φ and Θ

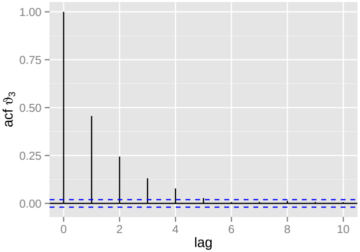


VARMA-MCMC

- Marginalisation on Φ: $p_{\Theta|\beta,\Sigma_e,y_{1:n}}(\cdot)$
- Adaptive MCMC proposal on Θ

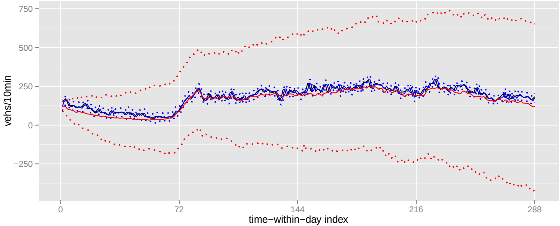
VARMA-MCMC

- Marginalisation on Φ: $p_{\Theta|\beta,\Sigma_e,y_{1:n}}(\cdot)$
- Adaptive MCMC proposal on Θ

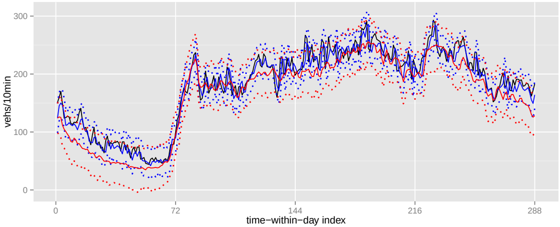


Prediction results

DSD

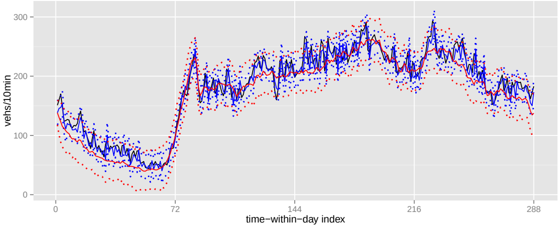


SD

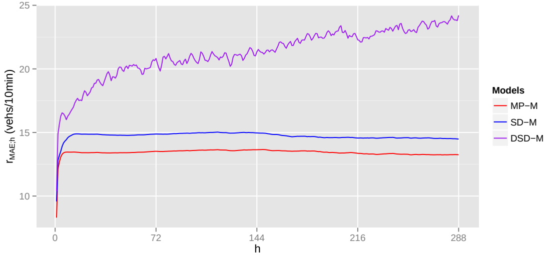


Prediction results

MP

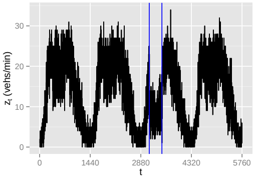


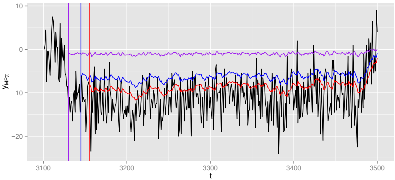
MAE

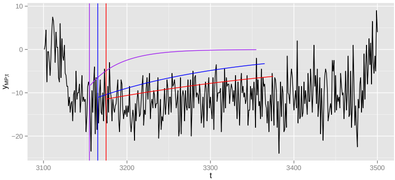


Problems

- Traffic data with incidents

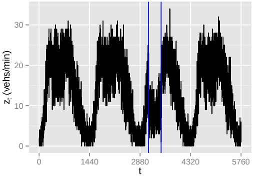


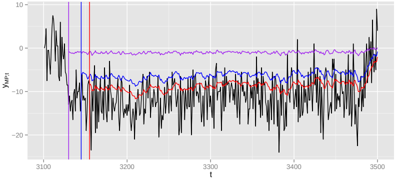


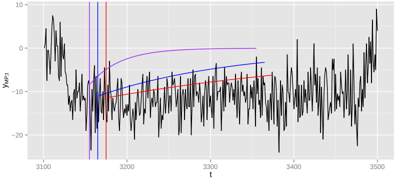


Problems

- Traffic data with incidents



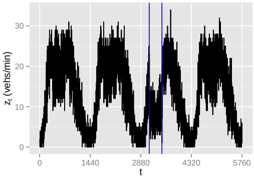


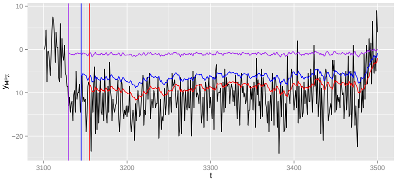


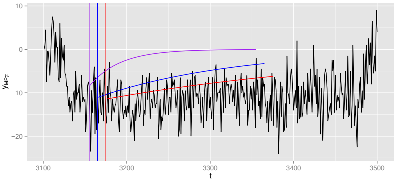
- Scalable to network expansion

Problems

- Traffic data with incidents







- Scalable to network expansion
- Sequential inference

Functional approximation iterLap

- Functional approximation $\widetilde{p}_x(x)$ of $p_x(x)$: why?

Functional approximation iterLap

- Functional approximation $\widetilde{p}_x(x)$ of $p_x(x)$: why?
- *(Bornkamp, 2011)* mixture of normal distributions approximation *iterLap*:

$$p_x(x) \approx \widetilde{p}_{n_c;x}(x) = \sum w_i N(x|\mu_i, Q_i)$$

Functional approximation iterLap

- Functional approximation $\widetilde{p}_x(x)$ of $p_x(x)$: why?
- *(Bornkamp, 2011)* mixture of normal distributions approximation *iterLap*:

$$p_x(x) \approx \widetilde{p}_{n_c;x}(x) = \sum w_i N(x|\mu_i, Q_i)$$

- An iterative process

Functional approximation iterLap

- Functional approximation $\widetilde{p}_x(x)$ of $p_x(x)$: why?
- *(Bornkamp, 2011)* mixture of normal distributions approximation *iterLap*:

$$p_x(x) \approx \widetilde{p}_{n_c;x}(x) = \sum w_i N(x|\mu_i, Q_i)$$

- An iterative process
  - i normal components at iteration i

Functional approximation iterLap

- Functional approximation $\widetilde{p}_x(x)$ of $p_x(x)$: why?
- *(Bornkamp, 2011)* mixture of normal distributions approximation *iterLap*:

$$p_x(x) \approx \widetilde{p}_{n_c;x}(x) = \sum w_i N(x|\mu_i, Q_i)$$

- An iterative process
  - i normal components at iteration i
  - Start at a point $x_{s;i}$, maximise the residual function $x_{m;i} = \arg\min_x \{h_{i;x}\}$

Functional approximation iterLap

- Functional approximation $\widetilde{p}_x(x)$ of $p_x(x)$: why?
- *(Bornkamp, 2011)* mixture of normal distributions approximation *iterLap*:

$$p_x(x) \approx \widetilde{p}_{n_c;x}(x) = \sum w_i N(x|\mu_i, Q_i)$$

- An iterative process
    - i normal components at iteration i
    - Start at a point $x_{s;i}$, maximise the residual function $x_{m;i} = \arg\min_x \{h_{i;x}\}$
    - Add a new normal component with mean $x_{m;i}$

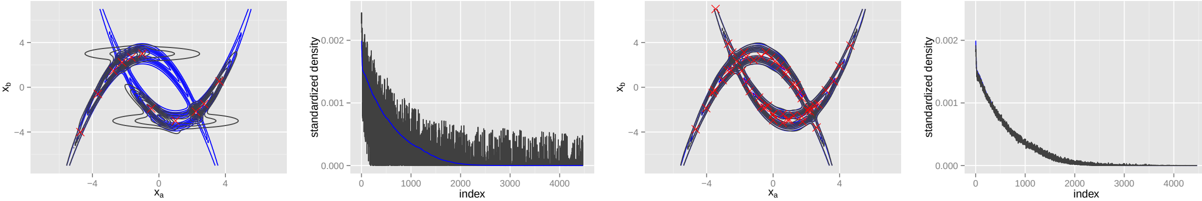
iterLap: Modifications

- Modifications: stopping criteria, starting points at each iteration, residual function.

iterLap: Modifications

- Modifications: stopping criteria, starting points at each iteration, residual function.
- Example:

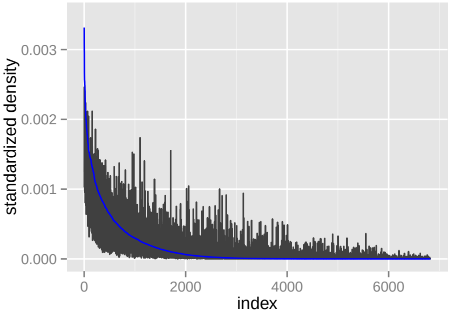
$$p_x(x) = 0.5N(x_a|\mu = -1, \sigma^2 = 6)N(x_b|\mu = -0.5(x_a+1)^2 + 3, \sigma^2 = 2)$$
$$+ 0.5N(x_a|\mu = 1, \sigma^2 = 6)N(x_b|\mu = 0.5(x_a-1)^2 - 3, \sigma^2 = 2).$$

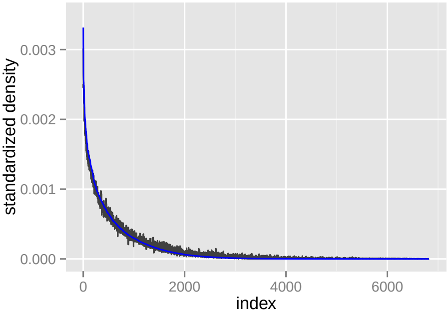


iterLap: Modifications

- Another example with $\dim(x_a) = \dim(x_b) = 1$, $\dim(x_c) = 4$:

$$p_x(x) = N(x_a|\mu_a, \Sigma_a = \sigma_a^2.I)N(x_b|\mu_b = A(x_a - \mu_a) + b, \Sigma_b = \sigma_b^2.I)$$
$$N(x_c|\mu_c = C(x_1 - \mu_a, x_b - \mu_b)^2, \Sigma_c = \sigma_c^2.I)$$





Particle filter and parameter learning

- Dynamic model (DM)

$$y_t \sim p_{y_t|x_t,\varphi}(\cdot)$$
$$x_t \sim p_{x_t|x_{t-1},\varphi}(\cdot)$$

Particle filter and parameter learning

- Dynamic model (DM)

$$y_t \sim p_{y_t|x_t,\varphi}(\cdot)$$
$$x_t \sim p_{x_t|x_{t-1},\varphi}(\cdot)$$

- Filtering problem: update from $p_{\varphi,x_t|y_{1:t}}(\cdot)$ to $p_{\varphi,x_{t+1}|y_{1:(t+1)}}(\cdot)$

Particle filter and parameter learning

- Dynamic model (DM)

$$
\begin{aligned}
y_t &\sim p_{y_t|x_t,\varphi}(\cdot) \\
x_t &\sim p_{x_t|x_{t-1},\varphi}(\cdot)
\end{aligned}
$$

- Filtering problem: update from $p_{\varphi,x_t|y_{1:t}}(\cdot)$ to $p_{\varphi,x_{t+1}|y_{1:(t+1)}}(\cdot)$
- Linear equations and $\varphi = \emptyset$: *Kalman filter*

Particle filter and parameter learning

- Dynamic model (DM)

$$
\begin{aligned}
y_t &\sim p_{y_t|x_t,\varphi}(\cdot) \\
x_t &\sim p_{x_t|x_{t-1},\varphi}(\cdot)
\end{aligned}
$$

- Filtering problem: update from $p_{\varphi,x_t|y_{1:t}}(\cdot)$ to $p_{\varphi,x_{t+1}|y_{1:(t+1)}}(\cdot)$
- Linear equations and $\varphi = \emptyset$: *Kalman filter*
- Non-linear equations and $\varphi = \emptyset$: *Particle filter*

Particle filter and parameter learning

- Dynamic model (DM)

$$
\begin{aligned}
y_t &\sim p_{y_t|x_t,\varphi}(\cdot) \\
x_t &\sim p_{x_t|x_{t-1},\varphi}(\cdot)
\end{aligned}
$$

- Filtering problem: update from $p_{\varphi,x_t|y_{1:t}}(\cdot)$ to $p_{\varphi,x_{t+1}|y_{1:(t+1)}}(\cdot)$
- Linear equations and $\varphi = \emptyset$: *Kalman filter*
- Non-linear equations and $\varphi = \emptyset$: *Particle filter*
- Non-linear equations and $\varphi \neq \emptyset$
    - *Liu and West, 2001. Fearnhead, 2002. Carvalho, 2010.*

Sequential inference for dynamic models

- Consider a sub-class of the dynamic model:

$$y_t \sim p_{y_t|x_t,\varphi}(\cdot),$$
$$x_t = Gx_{t-1} + h + u_t,$$

where $u_t \sim N(0, \Sigma_u = Q_u^{-1})$ and φ is a parameter vector (may include G, h and Q_u).

Sequential inference for dynamic models

- Consider a sub-class of the dynamic model:

$$y_t \sim p_{y_t|x_t,\varphi}(\cdot),$$
$$x_t = Gx_{t-1} + h + u_t,$$

where $u_t \sim N(0, \Sigma_u = Q_u^{-1})$ and φ is a parameter vector (may include G, h and Q_u).

- At time t, there is an approximated $\widetilde{p}_{\varphi,x_t|y_{1:(t-1)}}(\cdot)$.
- Use iterLap to approximate $\widetilde{p}_{\varphi,x_t|y_{1:t}}(\varphi, x_t) \propto \widetilde{p}_{\varphi,x_t|y_{1:(t-1)}}(\varphi, x_t) p_{y_t|x_t,\varphi}(y_t) \approx \sum_i w_i N_i(\varphi, x_t|\mu_{i;\varphi,x_t}, Q_{i;\varphi,x_t})$.
- Use the linear state equation and gaussian noise to evolve from $\widetilde{p}_{\varphi,x_t|y_{1:t}}(\cdot)$ to $\widetilde{p}_{\varphi,x_{t+1}|y_{1:t}}(\cdot)$:

$$\widetilde{p}_{\varphi,x_{t+1}|y_{1:t}}(\varphi, x_{t+1}) = \int \widetilde{p}_{\varphi,x_t|y_{1:t}}(\cdot) N(x_{t+1}|Gx_t + h, Q_u) dx_t.$$

Sequential inference for dynamic models

- Example:

$$y_t = (b_1.x_t + b_2 \sin(2\pi t/100) + b_3.\cos(2\pi t/50) + b_4)^2 + v_t,$$
$$x_t = a x_{t-1} + u_t,$$

where $u_t \overset{iid}{\sim} N(0, \sigma_u^2)$, $v_t \overset{iid}{\sim} N(0, \sigma_v^2)$ with $a^\star = 0.8$, $b^\star = (1.0, -2.0, 3.0, 15.0)$, $\tau_u^\star = \log(\sigma_u^{-2}) = \log(1.2^{-2})$, $\tau_v^\star = \log(\sigma_v^{-2}) = \log(15^{-2})$.

Sequential inference for dynamic models

- Example:

$$y_t = (b_1.x_t + b_2 \sin(2\pi t/100) + b_3.\cos(2\pi t/50) + b_4)^2 + v_t,$$
$$x_t = a x_{t-1} + u_t,$$

where $u_t \overset{iid}{\sim} N(0, \sigma_u^2)$, $v_t \overset{iid}{\sim} N(0, \sigma_v^2)$ with $a^\star = 0.8$, $b^\star = (1.0, -2.0, 3.0, 15.0)$, $\tau_u^\star = \log(\sigma_u^{-2}) = \log(1.2^{-2})$, $\tau_v^\star = \log(\sigma_v^{-2}) = \log(15^{-2})$.

- Identifiability problem, e.g. $x = y + z$

Sequential inference for dynamic models

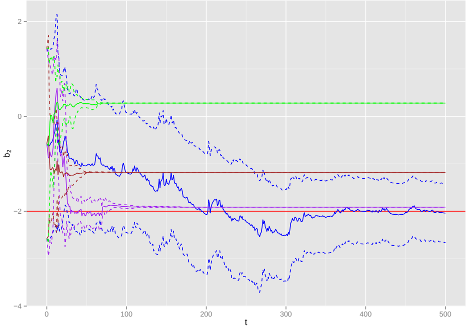
- Example:

$$
\begin{aligned}
y_t &= (b_1.x_t + b_2 \sin(2\pi t/100) + b_3.\cos(2\pi t/50) + b_4)^2 + v_t, \\
x_t &= a x_{t-1} + u_t,
\end{aligned}
$$

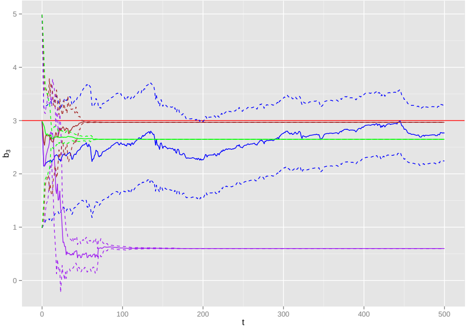
  where $u_t \overset{iid}{\sim} N(0, \sigma_u^2)$, $v_t \overset{iid}{\sim} N(0, \sigma_v^2)$ with $a^\star = 0.8$, $b^\star = (1.0, -2.0, 3.0, 15.0)$, $\tau_u^\star = \log(\sigma_u^{-2}) = \log(1.2^{-2})$, $\tau_v^\star = \log(\sigma_v^{-2}) = \log(15^{-2})$.

- Identifiability problem, e.g. $x = y + z$
- Assume that $b_1^\star$, $b_4^\star$ and $\tau_v^\star$ are known. Estimate $(a, b_2, b_3, \tau_u^\star, x_t)$:

Sequential inference for dynamic models

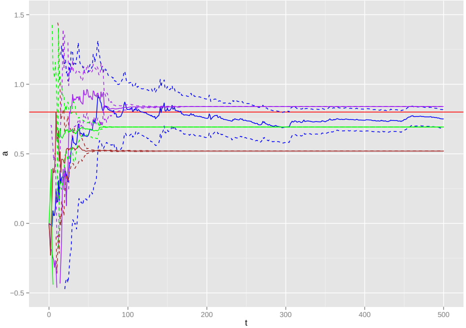


$p_{b_2|y_{1:t}}(\cdot)$

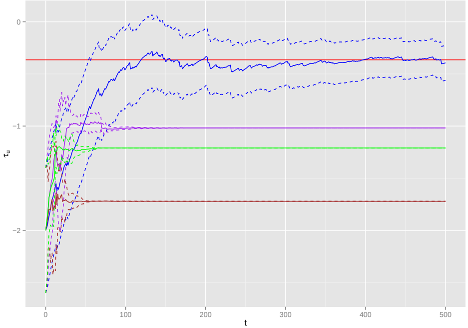


$p_{b_3|y_{1:t}}(\cdot)$

Sequential inference for dynamic models

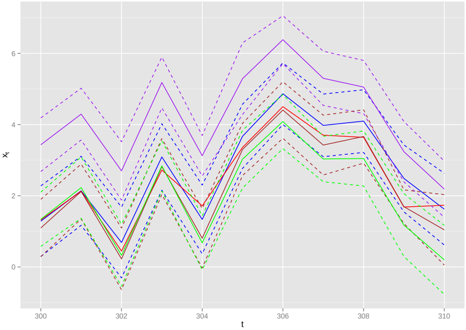


$p_{a|y_{1:t}}(\cdot)$



$p_{\tau_u|y_{1:t}}(\cdot)$

Sequential inference for dynamic models



$$p_{x|y_{1:t}}(\cdot)$$

A spatial-temporal model for STFF

- A model robust to traffic incident. Use the number of vehicles within a link, together with traffic flows

A spatial-temporal model for STFF

- A model robust to traffic incident. Use the number of vehicles within a link, together with traffic flows
- Information loss in multi-step-prediction

A spatial-temporal model for STFF

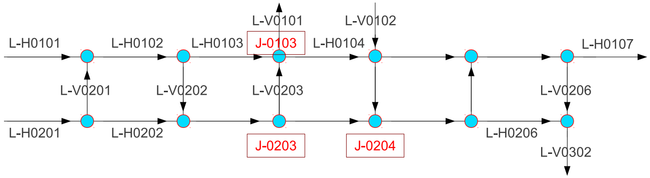
- A model robust to traffic incident. Use the number of vehicles within a link, together with traffic flows
- Information loss in multi-step-prediction
- Scalability

A spatial-temporal model for STFF

- A model robust to traffic incident. Use the number of vehicles within a link, together with traffic flows
- Information loss in multi-step-prediction
- Scalability
- Sequential inference

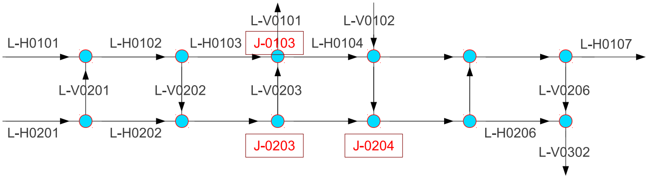
A spatial-temporal model for STFF

- The general model consists 4 sub-models:



A spatial-temporal model for STFF

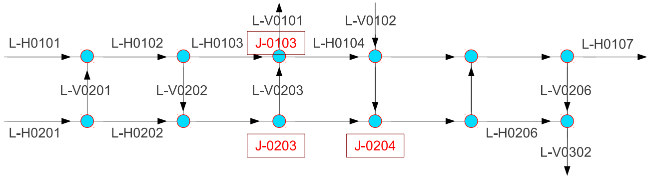
- The general model consists 4 sub-models:



- Link-outflow sub-model: $z_{2;t} = f_a(z_{1;t-1}, \nu_{t-1}, \Omega_a)$

A spatial-temporal model for STFF

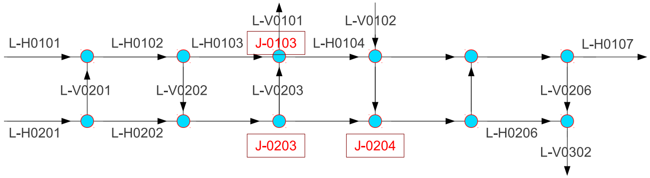
- The general model consists 4 sub-models:



- Link-outflow sub-model: $z_{2;t} = f_a(z_{1;t-1}, \nu_{t-1}, \Omega_a)$
- Junction sub-model: $\zeta_{2;t} = f_b(x_{b;t}, \zeta_{1;t}, \Omega_b)$ and $x_{b;t} = g_b(x_{b;t-1}, \Omega_b)$

A spatial-temporal model for STFF

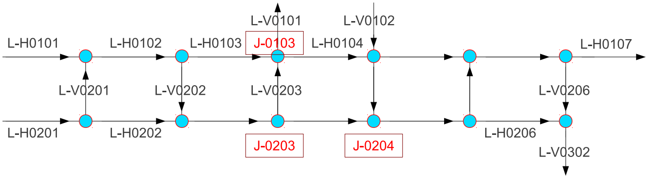
- The general model consists 4 sub-models:



- Link-outflow sub-model: $z_{2;t} = f_a(z_{1;t-1}, \nu_{t-1}, \Omega_a)$
- Junction sub-model: $\zeta_{2;t} = f_b(x_{b;t}, \zeta_{1;t}, \Omega_b)$ and $x_{b;t} = g_b(x_{b;t-1}, \Omega_b)$
- Root-inflow sub-model: $z_{1;t} = f_c(x_{c;t}, \Omega_c)$ and $x_{c;t} = g_c(x_{c;t-1}, \Omega_c)$

A spatial-temporal model for STFF

- The general model consists 4 sub-models:



- Link-outflow sub-model: $z_{2;t} = f_a(z_{1;t-1}, \nu_{t-1}, \Omega_a)$
- Junction sub-model: $\zeta_{2;t} = f_b(x_{b;t}, \zeta_{1;t}, \Omega_b)$ and $x_{b;t} = g_b(x_{b;t-1}, \Omega_b)$
- Root-inflow sub-model: $z_{1;t} = f_c(x_{c;t}, \Omega_c)$ and $x_{c;t} = g_c(x_{c;t-1}, \Omega_c)$
- The conservation equation for the number of vehicles: $\nu_t = \nu_{t-1} + z_{1;t} - z_{2;t} + e_{\nu;t}$

Link-outflow sub-model

- Use a polynomial spline

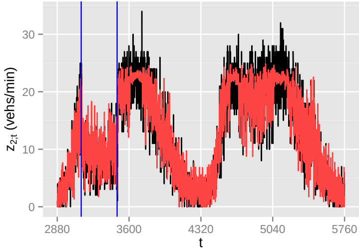
$$z_{2;t} = \sum_{i=0}^{2} \alpha_i z_{1;t-1}^{i} + \sum_{i'} \alpha_{2+i'} (z_{1;t-1} - a_{i'})_+^2 + \sum_{j=0}^{2} \beta_j \nu_{t-1}^{j} + \sum_{j'} \beta_{2+j'} (\nu_{t-1} - b_{j'})_+^2 + e_t$$

Link-outflow sub-model

- Use a polynomial spline

$$z_{2;t} = \sum_{i=0}^{2} \alpha_i z_{1;t-1}^{i} + \sum_{i'} \alpha_{2+i'}(z_{1;t-1} - a_{i'})_{+}^{2} + \sum_{j=0}^{2} \beta_j \nu_{t-1}^{j} + \sum_{j'} \beta_{2+j'}(\nu_{t-1} - b_{j'})_{+}^{2} + e_t$$

- A model is fitted with the first two-day data. Prediction results for the last two day:



Junction sub-model

- A dynamic model with noises $v_t \overset{iid}{\sim} N(0, \Sigma_v = \sigma_v^2 I)$ and $u_t \overset{iid}{\sim} N(0, \Sigma_u = Q_u^{-1})$:

$$\zeta_{2;t} = A_t(x_t)\zeta_{1;t} + v_t$$

$$x_t = x_{t-1} + u_t$$

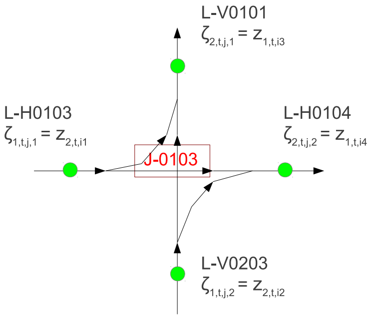
Junction sub-model

- A dynamic model with noises $v_t \overset{iid}{\sim} N(0, \Sigma_v = \sigma_v^2 I)$ and $u_t \overset{iid}{\sim} N(0, \Sigma_u = Q_u^{-1})$:

$$\zeta_{2;t} = A_t(x_t)\zeta_{1;t} + v_t$$
$$x_t = x_{t-1} + u_t$$

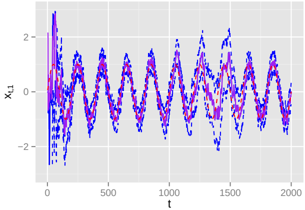
- A is the turning rate matrix:

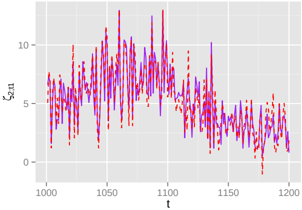
$$B_t = \begin{pmatrix} x_{t,1} & x_{t,2} \\ 0 & 0 \end{pmatrix}$$
$$A_{t,i,i'} = \frac{\exp(B_{t,i,i'})}{\sum_k \exp(B_{t,k,i'})}$$



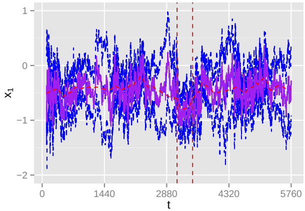
Junction sub-model: results

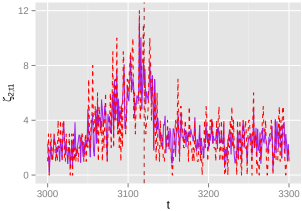
- Simulated dataset:





- VISSIM dataset:





Conclusion

- STFF with VARMA models and MCMC
- iterLap and sequential inference
- STFF with incidental traffic pattern

Conclusion

- STFF with VARMA models and MCMC
- iterLap and sequential inference
- STFF with incidental traffic pattern
- Thank you!
- Questions?