Bayesian Analysis of Traffic Flow Data

Workshop II: Traffic Estimation @ IPAM's Traffic Program

Vadim Sokolov Nicholas Polson

Argonne National Laboratory and University of Chicago

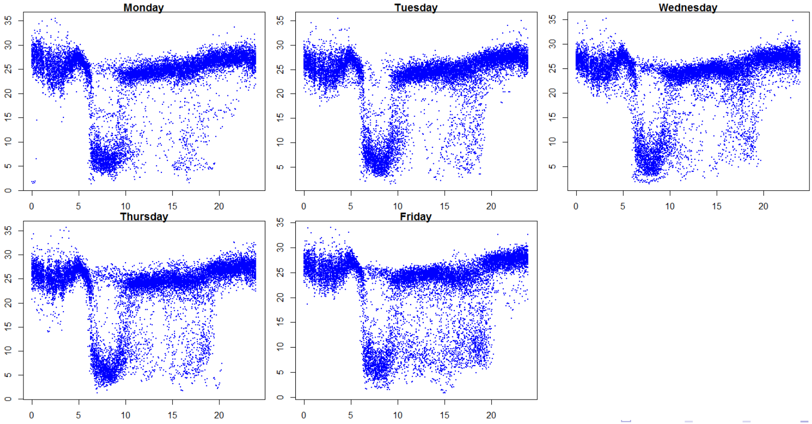
October 13, 2015

Overview

- Motivation
- LWR Model. How do we model traffic jams? Flux Function or Fundamental Diagram State-Space Representation
- Sequential Bayesian Learning Particle Filtering and Learning Methods
  On-line Tracking of Traffic Flow Densities
  Bayesian methods are flexible and fast
- Empirical Results: I-55 in Chicago
- Statistical Learning Models
- Sparseness and Robustness
- Liner and Non-linear

Motivation (Data)

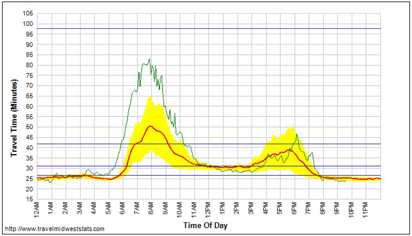
- Speed plots for 1 year
- 48 Wednesdays; 52 minus holidays and missing data days
- Measurements taken every 5 minutes (each plot has up to 14967 measurements)
- Speed is a mixture. There are surprises!



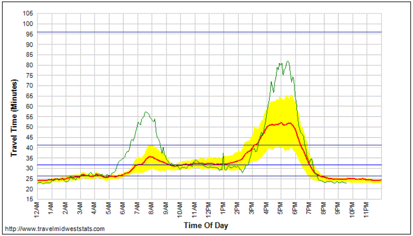
Non-recurrent traffic conditions

Weather

- Impact of light snow on travel times
- I-55 near Chicago on December 11, 2013
- 1.8 inches of snow



(a) North Bound

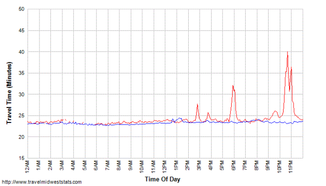


(b) South Bound

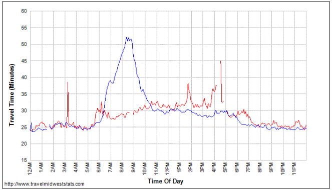
Non-recurrent traffic conditions

Special Events

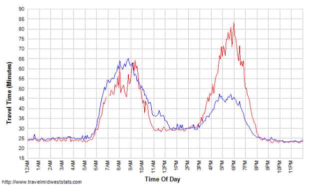
Impact of special events on I-55 north bound travel



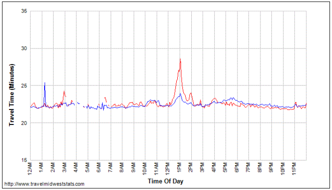
(a) NATO Summit on Sunday May 20, 2012



(b) NATO Summit on Monday May 21, 2012



(c) New York Giants at Bears on Thursday October 10, 2013

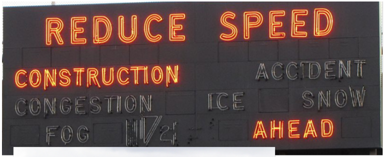


(d) Baltimore Ravens at Bears on Sunday November 10, 2013

Motivation

Real Time Traffic Management

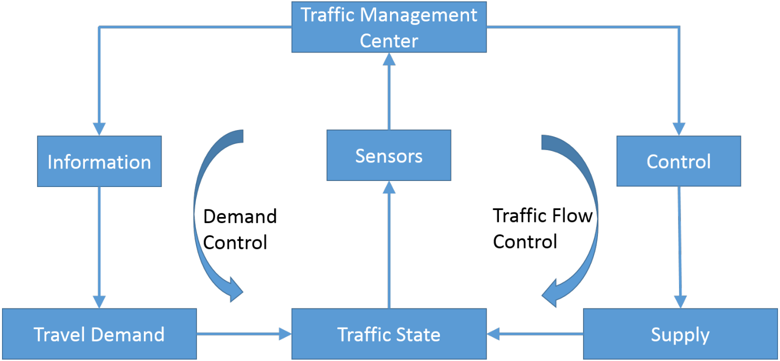
- Route guidance
- Ramp metering
- Adaptive traffic signals
- Speed harmonization
- Incident management
- Estimate current traffic conditions



¿¿

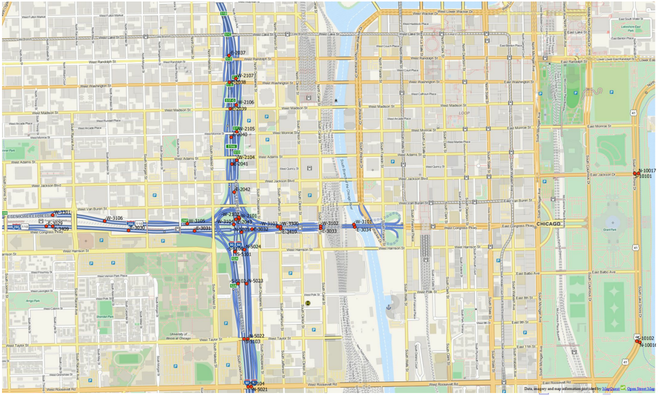
Motivation

Dynamic traffic and demand management (need 15-60 minute forecasts)

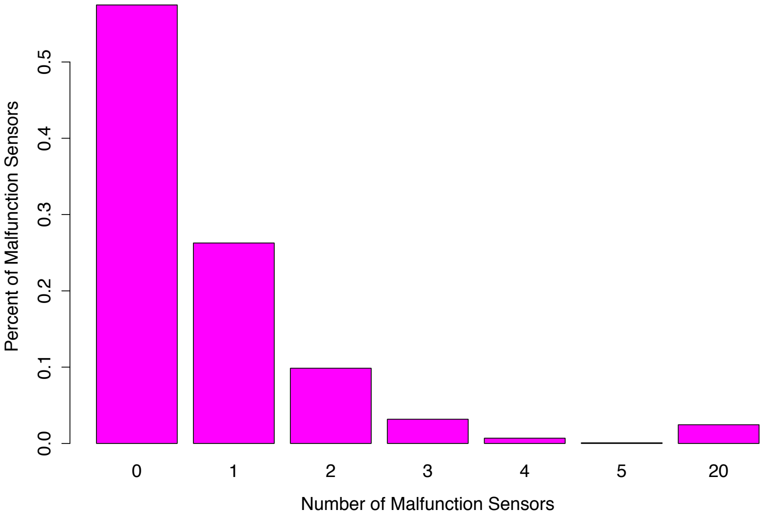


Data

- Loop detector (presence sensor)
- Speed, occupancy and flow, averaged over 5 minutes
- 1500 highway location around Chicago area
- Archived at Argonne since 2008
- Approx 50Mb per sensor (75Gb total)
- Missing/inaccurate measurements is an issue



System Health

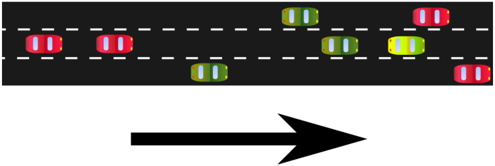


Single Lane Loop Detector

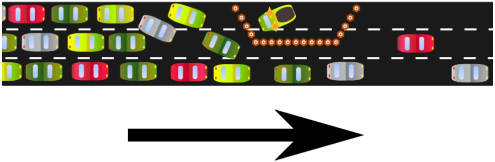
I-55 NB Google Street View

Macroscopic Traffic Flow

Free flow regime (no interactions between cars)



Congestion at bottlenecks



Fundamental Characteristics

- $q(x, t)$: flow [vehicles per hour]
- $\rho(x, t)$: density [vehicles per mile]
- $v(x, t)$: speed [miles per hour]

$$q(x, t) = v(x, t) \times \rho(x, t)$$

[vehicles per hour] = [miles per hour] × [vehicles per mile]

Discontinuities

- Macroscopic parameters (flow, density, speed) might have discontinuities
- Even when initial and boundary conditions are smooth
- Physical interpretation: traffic queue
- Shock wave (downstram density is different from upstream density)

LWR Traffic Flow Model

Lighthill-Whitham-Richards

$$\boxed{\frac{\partial \rho(x,t)}{\partial t} + \frac{\partial q(x,t)}{\partial x} = 0}$$

Relation between density and flow (fundamental diagram)

$$q(x,t) = q^*(\rho(x,t))$$

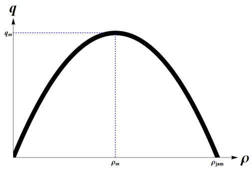
Combined with fundamental diagram

$$\frac{\partial \rho(x,t)}{\partial t} + \frac{\partial q^*(\rho(x,t),x)}{\partial x} = 0$$

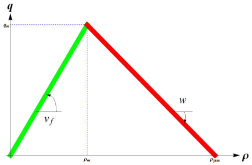
- One dimensional conservation law
- Properly models shock wave path
- Homogeneous road segment
- Key postulate: there is relation between flow q and density ρ

Examples

Fundamental diagram (flow-density relation):

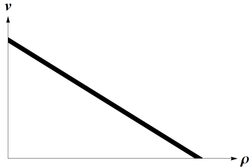


Greenshield's

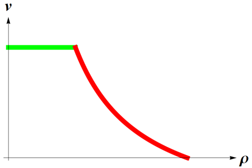


Triangular

Speed-density relation:



Greenshield's



Triangular

Solving the Conservation Law Equation

Godunov Scheme

- Discontinuous solution (finite differences are not applicable)
- Godunov Scheme solves Riemann problem for each 2 consecutive sells

Riemann = Cauchy problem (initial value problem) with initial conditions that have a single discontinuity

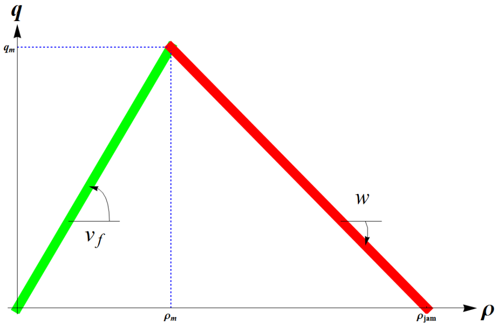
$$\rho_0(x) = \begin{cases} \rho_l, \; x < 0 \\ \rho_r, \; x > 0 \end{cases} \tag{1}$$

For th Riemann problem the speed of the of the shock wave propagation is given by

$$w = \frac{q(\rho_l) - q(\rho_r)}{\rho_l - \rho_r}$$

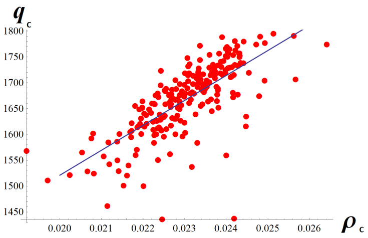
LWR Model Parameters

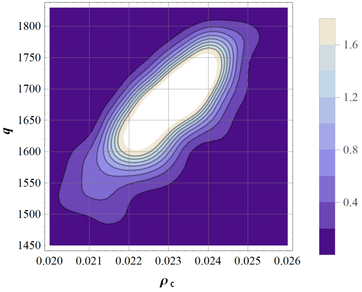
Fundamental diagram + conservation law

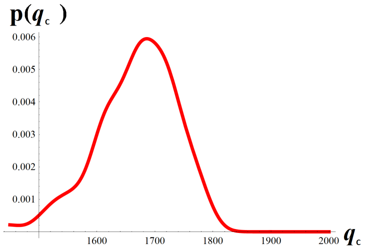


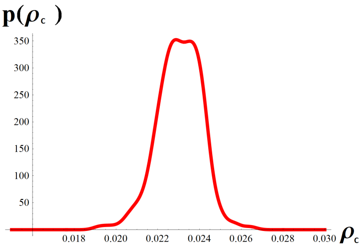
ρ_c - critical density
q_c - critical flow (capacity)
ρ_{jam} - maximal possible density

Example: I-55 Chicago







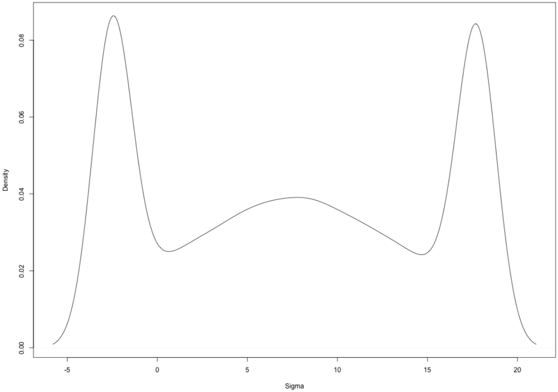


Wave Speed Propagation is a Mixture Distribution

Shock wave propagation speed is a mixture, when calculated using Godunov scheme

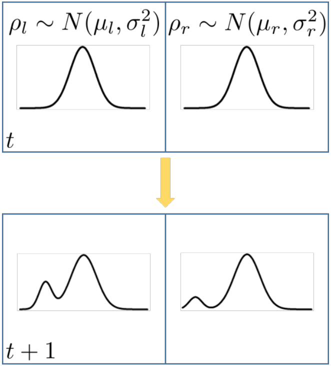
$$w = \frac{q(\rho_l) - q(\rho_r)}{\rho_l - \rho_r} \quad \left[\frac{mi}{h}\right] = \left[\frac{veh}{h}\right]\left[\frac{mi}{veh}\right].$$

Assume $\rho_l \sim TN(32, 16, 0, 320)$ and $\rho_r \sim TN(48, 16, 0, 320)$
$q_c = 1600\ veh/h$, $\rho_c = 40\ veh/mi$, and $\rho_{jam} = 320\ veh/mi$



Traffic Flow Speed Forecast is a Mixtrue Dsitribution

Theorem: The solution (including numerical) to the LWR model with stochastic initial conditions is a mixture distribution.



A moment based filters such as Kalman Filter or Extended Kalman Filter would not capture the mixture.

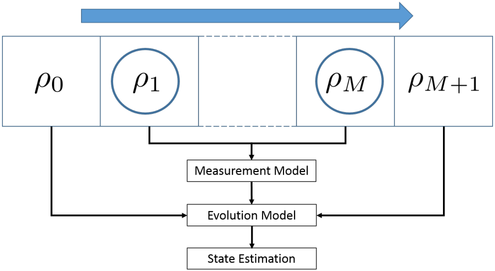
Problem at Hand

The Parameter Learning and State Estimation Problem

- Goal: given sparse sensor measurements, find the distribution over traffic state and underlying traffic flow parameters $p(\theta_t, \phi | y_1, y_2, ..., y_t)$; $\phi = (q_c, \rho_c)$
- Parameters of the evolution equation (LWR) are stochastic
- Distribution over state is a mixture
- Can't use moment based filters (KF, EKF,...)

Data Assimilation

State Space Representation



State space formulation allows to combine knowledge from analytical model with the one from field measurements, while taking model and measurement errors into account

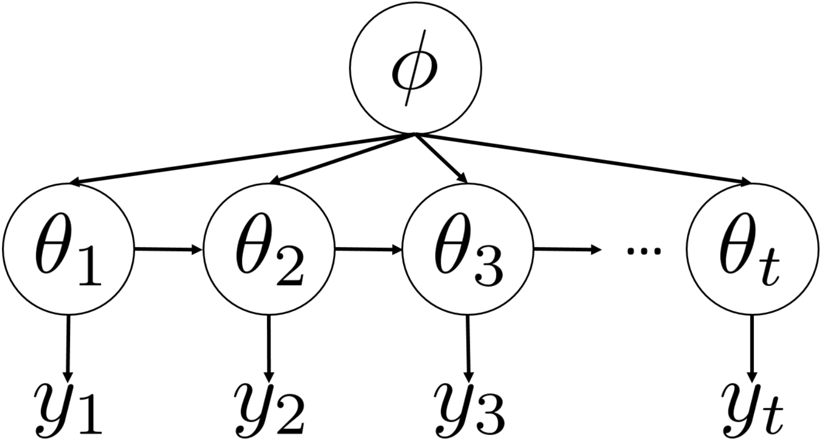
State Space Representation

- State vector $\theta_t = (\rho_{1t}, \ldots, \rho_{nt})$
- Boundary conditionals ρ_{0t} and $\rho_{(n+1)t}$
- Underlying parameters $\phi = (q_c, \rho_c)$ are stochastic

$$\text{Observation: } y_{t+1} = H\theta_{t+1} + v; \;\; v \sim N(0, V) \tag{2}$$
$$\text{Evolution: } \theta_{t+1} = f_\phi(\theta_t) + w; \;\; w \sim N(0, W) \tag{3}$$

$H : \mathbb{R}^M \to \mathbb{R}^k$ in the measurement model. $\phi = (q_c, \rho_c, \rho_{max})$.
Parameter priors: $q_c \sim N(\mu_q, \sigma_c^2)$, $\rho_c = Uniform(\rho_{min}, \rho_{max})$

Particle Parameter Learning

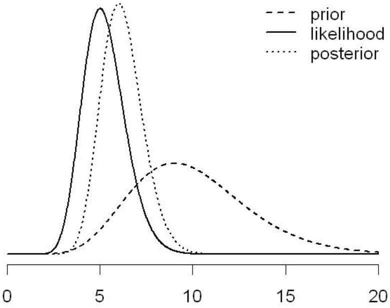


Streaming Data

How do Parameter Distributions change in Time?

Online Dynamic Learning

- Real-time surveillance
- Bayes means sequential updating of information
- Update posterior density $p(\boldsymbol{\theta}, \boldsymbol{\phi} \mid y_t)$ with every new observation $(t = 1, \ldots, T)$ - "sequential learning"



Bayes theorem:

$$p(\boldsymbol{\theta}, \boldsymbol{\phi} \mid y^t) \propto p(y_t \mid \boldsymbol{\theta})\, p(\boldsymbol{\theta} \mid y^{t-1})$$

Particle Filter (Resample-Propagate)

Construct an essential state vector θ_{t+1}.

$$
\begin{aligned}
p(\theta_{t+1}|y^{t+1}) &= \int p(\theta_{t+1}|\theta_t, y_{t+1})\ d\mathbb{P}(\theta_t|y^{t+1}) \\
&\propto \int \underbrace{p(\theta_{t+1}|\theta_t, y_{t+1})}_{propagate} \overbrace{\underbrace{p(y_{t+1}|\theta_t)}_{resample}\ d\mathbb{P}(\theta_t|y^t)}
\end{aligned}
$$

Bayes Rule

$$
p(y_{t+1}, \theta_{t+1}|\theta_t) = p(y_{t+1}|\theta_t)\, p(\theta_{t+1}|\theta_t, y_{t+1}).
$$

- Given a particle approximation to $p^N(\theta_t|y^t)$

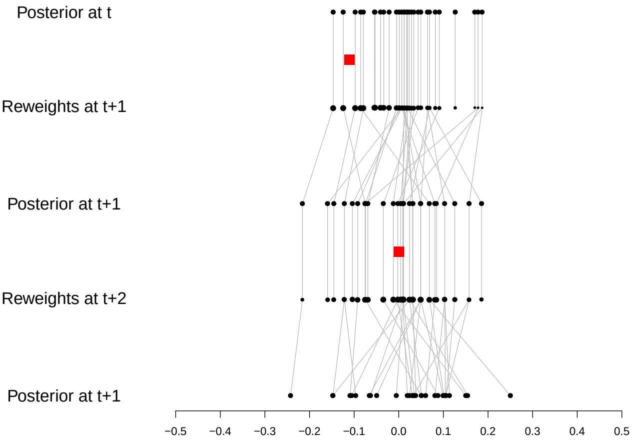
$$
\begin{aligned}
p^N(\theta_{t+1}|y^{t+1}) &\propto \sum_{i=1}^{N} p\left(y_{t+1}|\theta_t^{(i)}\right) p\left(\theta_{t+1}|\theta_t^{(i)}, y_{t+1}\right) && (4) \\
&= \sum_{i=1}^{N} w_t^{(i)} p\left(\theta_{t+1}|\theta_t^{(i)}, y_{t+1}\right), && (5)
\end{aligned}
$$

where

$$
w_t^{(i)} = \frac{p\left(y_{t+1}|\theta_t^{(i)}\right)}{\sum_{i=1}^{N} p\left(y_{t+1}|\theta_t^{(i)}\right)}.
$$

- Essentially a mixture Kalman filter

Resample – Propagate



Particle Parameter Learning

Given particles (a.k.a. random draws) $(\theta_t^{(i)}, \phi^{(i)}, s_t^{(i)})$, $i = 1, \ldots, N$

$$p(\theta_t | y_{1:t}) = \frac{1}{N} \sum_{i=1}^{N} \delta_{\theta^{(i)}} \ .$$

- First resample $(\theta_t^{k(i)}, \phi^{k(i)}, s_t^{k(i)})$ with weights proportional to $p(y_{t+1}|\theta_t^{k(i)}, \phi^{k(i)})$ and $s_t^{k(i)} = S(s_t^{(i)}, \theta_t^{k(i)}, y_{t+1})$ and then propogate to $p(\theta_{t+1}|y_{1:t+1})$ by drawing $\theta_{t+1}^{(i)}$ from $p(\theta_{t+1}|\theta_t^{k(i)}, \phi^{k(i)}, y_{t+1})$, $i = 1, \ldots, N$.
- Next we update the sufficient statistic as

  $$s_{t+1} = S(s_t^{k(i)}, \theta_{t+1}^{(i)}, y_{t+1}),$$

  for $i = 1, \ldots, N$, which represents a deterministic propogation.
- Finally, parameter learning is completed by drawing $\phi^{(i)}$ using $p(\phi|s_{t+1}^{(i)})$ for $i = 1, \ldots, N$.

Algorithm

These ingredients then define a particle filtering and learning algorithm for the sequence of joint posterior distributions $p(\theta_t, \phi | y_{1:t})$:

Step 1. (Resample) Draw an index $k_t(i) \sim Mult_N\left(w_t^{(1)}, \ldots, w_t^{(N)}\right)$,

where the weights are given by $w_t^{(i)} \propto p(y_{t+1}|(\theta_t, \phi)^{(i)})$, for $i = 1, \ldots, N$

Step 2. (Propagate) Draw $\theta_{t+1}^{(i)} \sim p\left(\theta_{t+1}|\theta_t^{k_t(i)}, y_{t+1}\right)$ for $i = 1, \ldots, N$.

Step 3. (Update) $s_{t+1}^{(i)} = S(s_t^{k_t(i)}, \theta_{t+1}^{(i)}, y_{t+1})$

Step 4. (Replenish) $\phi^{(i)} \sim p(\phi|s_{t+1}^{(i)})$

There are a number of efficiency gains from such an approach, e.g. it does not suffer from degeneracy problems associated with traditional propagate-resample algorithms when y_{t+1} is an outliers.

Resample-Propagate Steps (LWR model)

$$\text{Resample: } p(y_{t+1}|\theta_{t+1},\phi) \sim N(H_{t+1}\theta_{t+1}, V_t)$$
$$\text{Propagate: } p(\theta_{t+1}|\theta_t,\phi) \sim N(f_\phi(\theta_t), W_t).$$

Therefore, we have distribution

$$p(y_{t+1}|\theta_t,\phi) \sim N(H_{t+1}f_\phi(\theta_t), H_{t+1}^T W_t H_{t+1} + V_t)$$

For propagation of θ_{t+1}, we use Bayes' rule

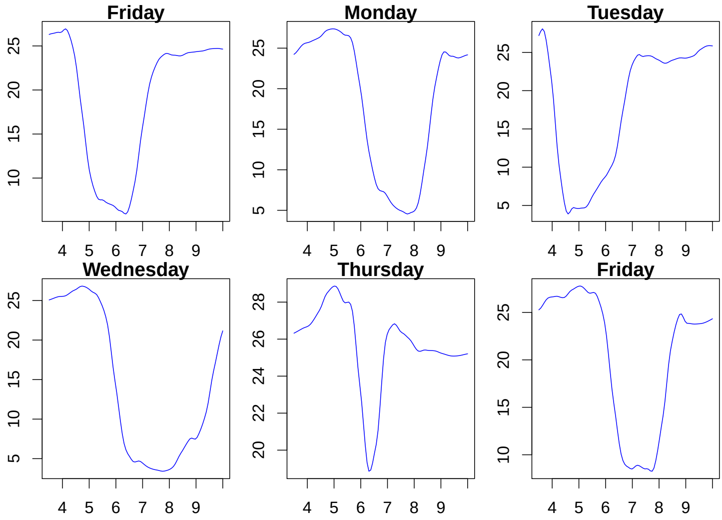
$$p(\theta_{t+1}|\theta_t,\phi,y_{t+1}) \sim N(\mu_{t+1}, C_{t+1})$$

mean and variance follow the Kalman recursion

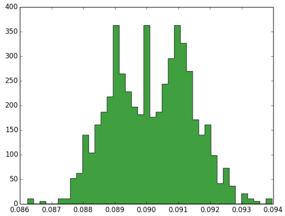
$$\begin{aligned}
\text{Forecast: } & \mu_f = f_\phi(\theta_t), \;\; C_f = W_{t+1} \\
\text{Kalman Gain: } & K = C_f H_{t+1}^T (H_{t+1} C_f H_{t+1}^T + V_{t+1})^{-1} \\
\text{Measurement Assimilation: } & \mu_{t+1} = \mu_f + K(y_{t+1} - H_{t+1}\mu_f) \\
& C_{t+1} = (I - KH_{t+1})C_f
\end{aligned}$$

Typical Morning Peak Period Pattern

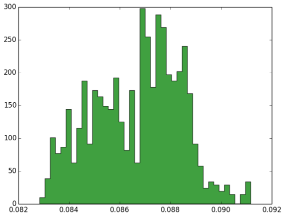
February 2009, I-55 NB @ Cicero



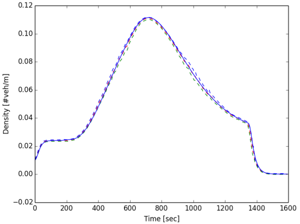
Empirical Results I (state estimation)



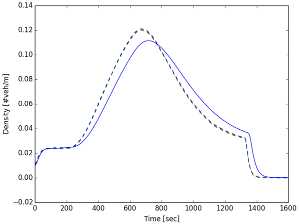
Mixture at t=890s



Mixture at t=910s

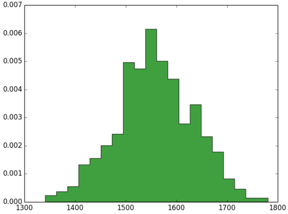


With Parameter Learning

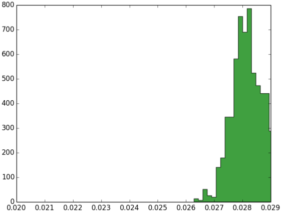


Without Parameter Learning

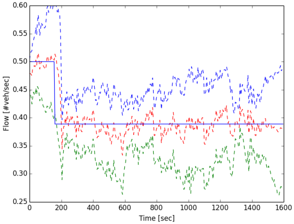
Empirical Results II (parameter learning)

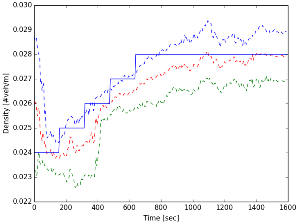


Posterior for q_c

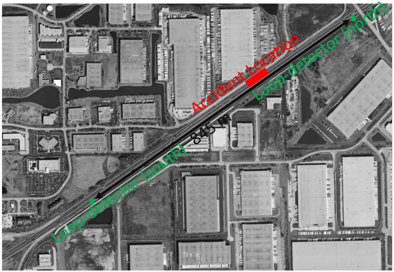


Posterior for ρ_c





Accident Photos



(a)Accident and loop detector locations

(b) Image of the accident from the roadside camera

Real-time accident modeling

- Identify a drop in capacity (critical flow) due to an accident
- May 9, 2014 a semi-tractor trailer caught fire at 6:40 am on interstate highway I-55
- The police shut down the southbound lanes
- Will the filter pick up the "rubbernecking" effect on the northbound lanes?

Model Setup

- Road segment under study is between two sensors: length is 845 meters
- $h = 845/4 = 211$ meters
- $\tau = 5$ minutes (satisfies the Courant-Friedrichs-Lewy condition)
- Prior on road capacity: $q_m \sim U[1440, 1560]$ veh/h
- Critical density is fixed at $\rho_m = 0.025$ veh/m
- $\sigma_{meas} = 0.2 \times 10^{-2}$ veh/m
- $\sigma_{evolution} = 0.1 \times 10^{-2}$ veh/m

Regularization

- To address the problem of model identification
- Use relation between free flow speed, capacity and critical density, namely $v_f = q_c/\rho_c$ to regularize
- $v_f \approx 17\ m/s$

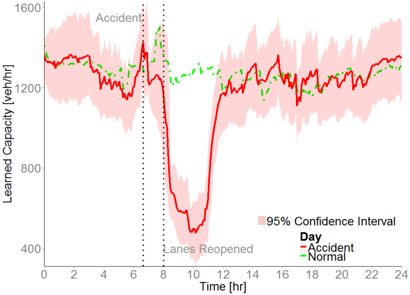
Our particle weights are regularized by

$$w_t^{(i)} = \frac{p\left(y_{t+1}|(\theta_t,\phi)^{(i)}\right)\boxed{\varphi(q_c^{(i)}/\rho_c^{(i)}, v_f, \sigma_{v_f})}}{\sum_{i=1}^N \left[p\left(y_{t+1}|(\theta_t,\phi)^{(i)}\right)\varphi(q_c^{(i)}/\rho_c^{(i)}, v_f, \sigma_{v_f})\right]},$$

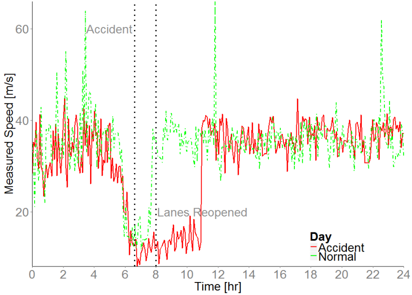
where φ is the p.d.f of the normally distributed variable. The prior error standard deviation was set at $\sigma_{v_f} = 5\ m/s$. Choice of both v_f and σ_{v_f} is based on empirical observations.

Learned Road Capacity

- 15 minutes between traffic flow speed reverts to a normal level and capacity recovers.
- time it takes the algorithm to learn the capacity



(a) Learned capacity of the road segment



(b) Measured speed at the south loop-detector

Learned Road Capacity

- captures the effect of capacity degradation as a result of the accident
- 95% Bayes credible intervals demonstrate that uncertainty about the estimate is larger during normal operating mode and lower during the periods of capacity degradation and recovery.
- slope of the speed curve on an accident day is much steeper than the slope of the learned capacity curve
- there is some delay associated with the learning process

.

Problems With Flow Models

- Need to know fundamental diagram for each road segment under study
- Need to deal with data imputations
- Works very well with flow/density measurements, need to tweak for speed measurements (GPS)
- Assumptions on the shape of fundamental diagram do not always hold (histeria, arterials)

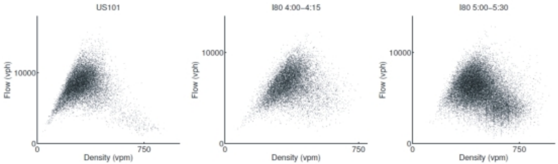


Image Source: `https://www.ocf.berkeley.edu/~argote/research.html`

Statistical Learning

Models:

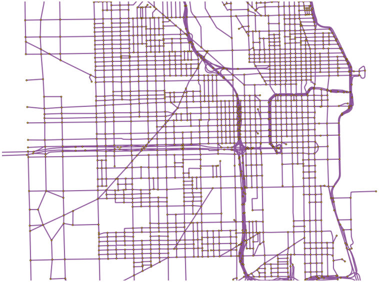
1. Non parametric regression (nearest neighbors)
2. Autoregression (linear)
3. Deep learning (non-linear regression)

Model requirements:

- Robustness (data is noisy)
- Scalability (data sets are large)
- Forecast accuracy (Robustness)

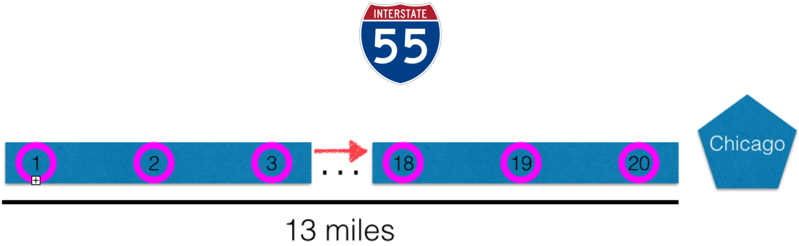
Large Road Graph

- Graph of the roads might contain millions of vertexes (road segments)
- Measured data is in Tb
- Need scalable models

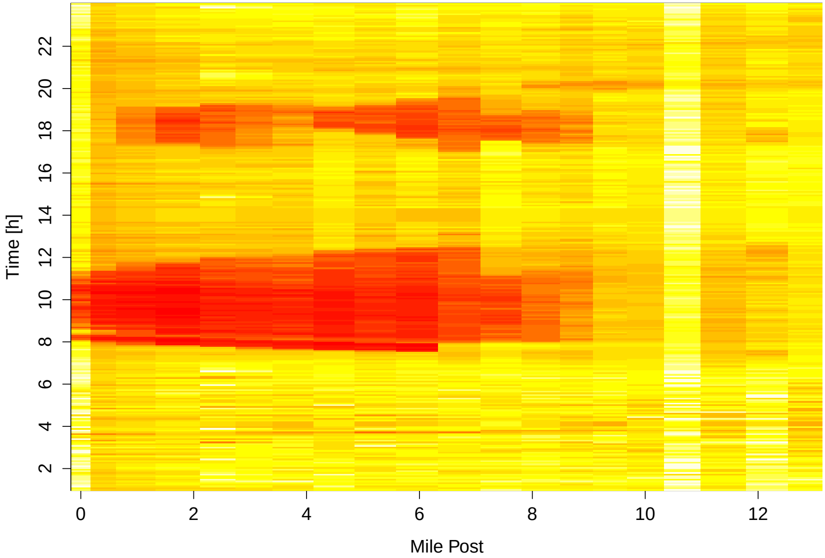


Study Area

- 20 loop detector sensors
- 13 miles of I-55 north bound (towards the city)
- downstream is always congested in the morning
- upstream might be uncontested
- used data from 2009



Patterns in flow data

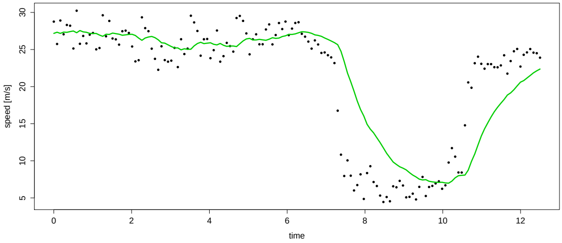


Robustness

Measurement data is noisy with many anomalies.
Potential solutions:

- Smooth data (filter out unnecessary variations and anomalies)
  - Iterative Exponential Smoothing
  - Median Smoothing
  - Trend Filtering
- Use robust loss functions, when possible, i.e. l_1 norm or equivalently Huber's fit, when computation is an issue.

Iterative Exponential Smoothing



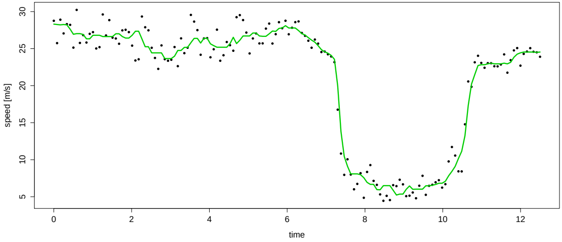
$$x_n = (1-\beta)x_{n-1} + \beta x_n$$

where $x_0 = 0$, and β is the smoothing parameter, with $0 < \beta < 1$.

$$x_n = \beta \sum_{i=1}^{n} (1-\beta)^{n-1} x_i.$$

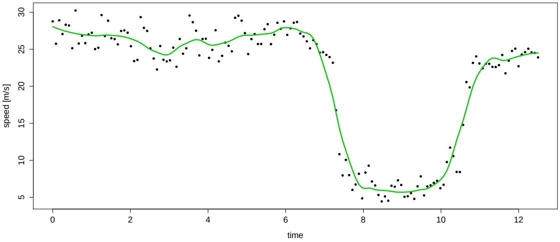
- $+$: Computationally Efficient
- $+$: Works well for arterials
- - : Does not work work well for processes that quickly change regimes (highways)

Median Filter



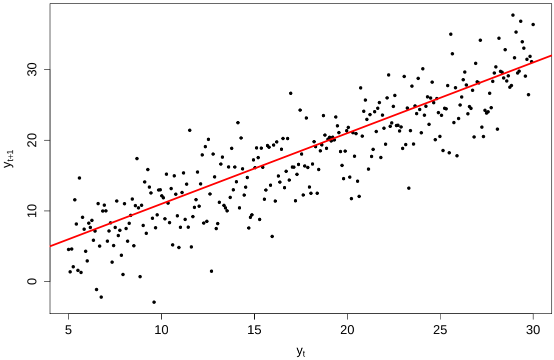
- +: Computationally Efficient
- +: Captures quick changes in regimes
- -: Does not perform well on arterials

Trend Filter



- +: Computationally Efficient
- +: Captures quick changes in regimes
- +: Accurately captures up/down trend, even short ones
- -: Does not perform well on arterials
- -: Slopes might be underestimated

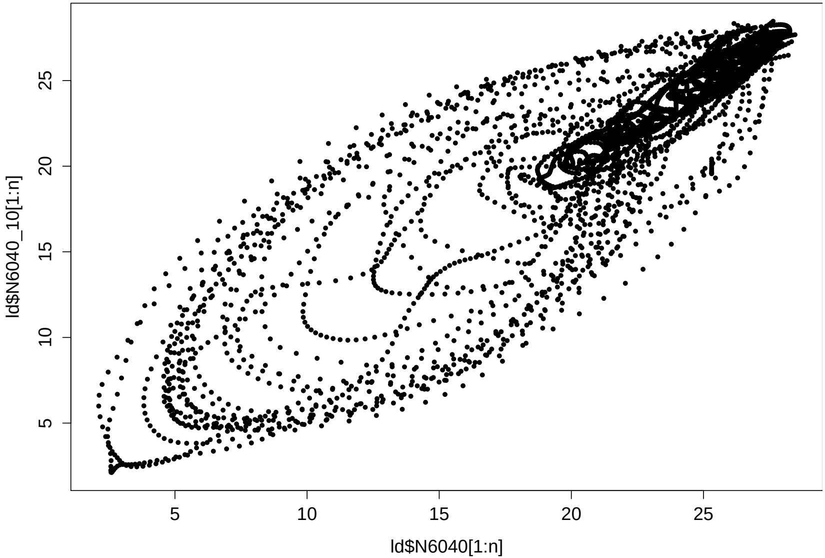
Autoregression



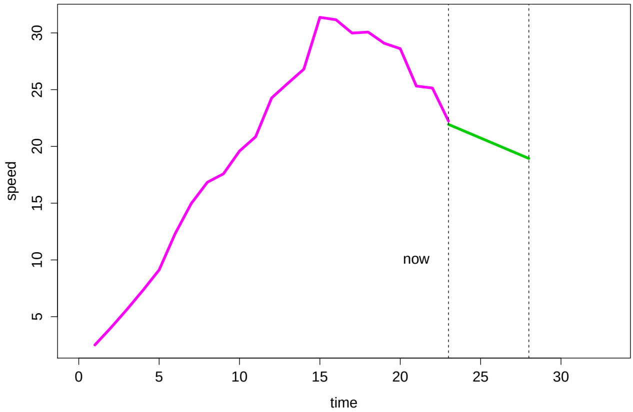
Forecast:

$$x_i^t = \sum_{j \in N(i)} \sum_{l=1}^{L} a_{ij}^l x_j^{t-l}$$

Autoregression



k-Nearest Neigbours



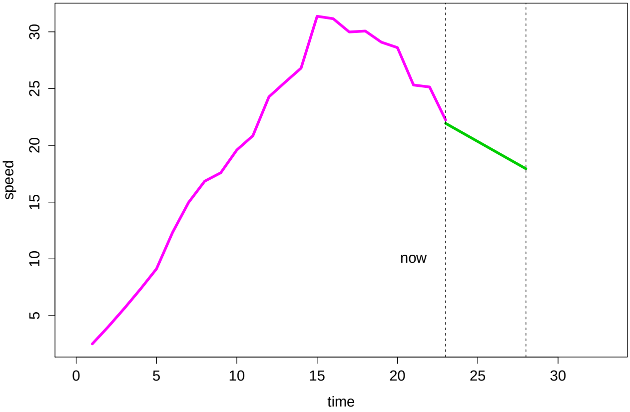
Forecast #1

Forecast:

$$x_i^t = \frac{1}{k} \sum_{\tau \in N_k(x_i^t)} x_i^\tau$$

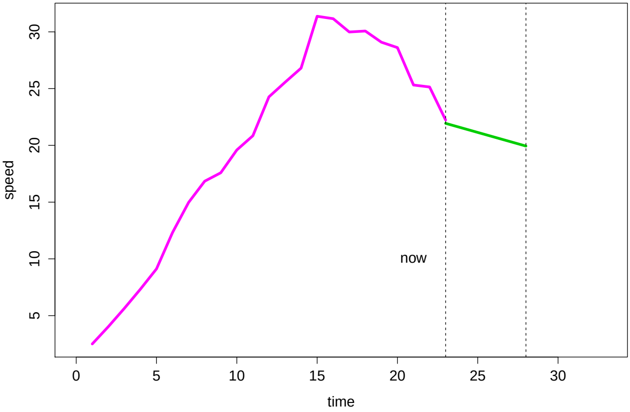
Neighbors both in space and time

k-Nearest Neigbours



Forecast #2

k-Nearest Neigbours



Forecast #3

Choice of predictors: Heuristics

- Geometric neighbors: might end up including many unnecessary edges (especially in city centers)
- Topological neighbors: can miss an important predictor (i.e. frontage road or arterial part of a corridor)

Atomated choice of predictors

- Forward step-wise: at each step add the predictor that leads to smallest error
- Backward step-wise: at each step remove least significant predictor (with smallest *p*-value)
- Forward stage-wise: update the coefficient for the predictor most correlated with error
- Regularization

Regularizartion

Ivanov regularization

$$\min_{x \in \mathbb{R}} ||Ax - b||_2^2 \quad \text{s.t.} \quad ||x||_l \leq k$$

Morozov regularization

$$\min_{x \in \mathbb{R}} ||x||_l \quad \text{s.t.} \quad ||Ax - b||_2^2 \leq \tau$$

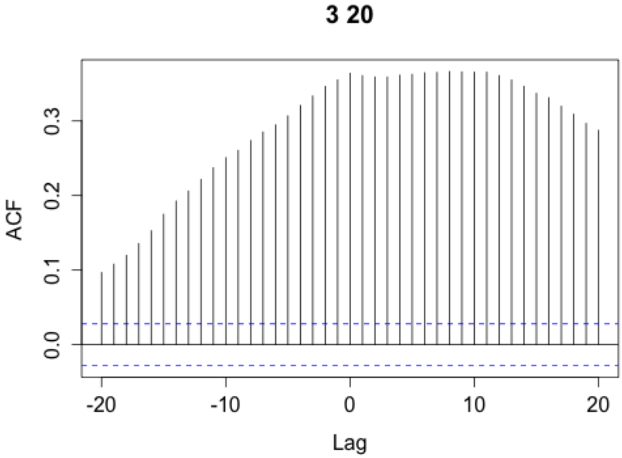
Here τ reflects the so called noise level, i.e. an estimate of the error which is made during the measurment of b.

Tikhonov regularization

$$\min_{x \in \mathbb{R}} ||Ax - b||_2^2 + \lambda ||x||_l$$

- Tikhonov regularization with $l = 1$ is lasso
- Tikhonov regularization with $l = 2$ is ridge regression
- lasso + ridge = elastic net

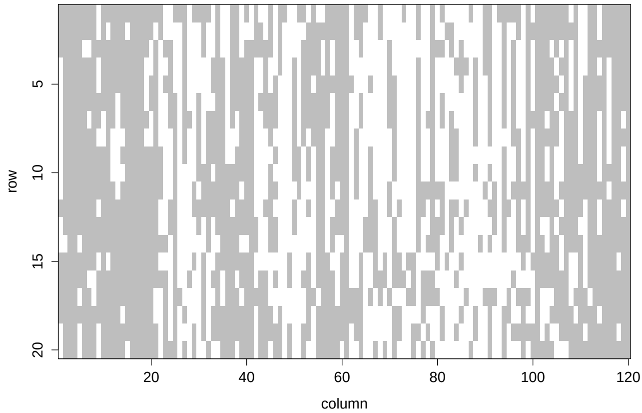
Choosing time lag L



The max at $L = 6$

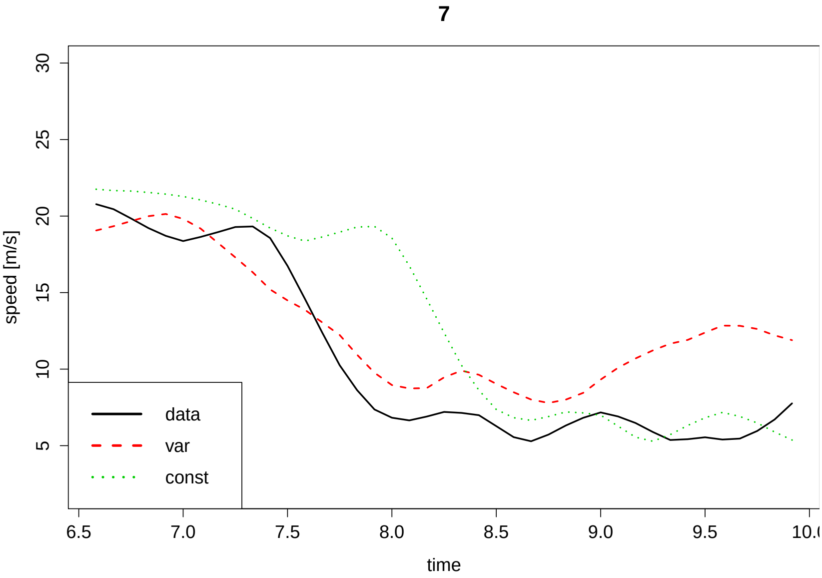
Lasso

Sparsity pattern of matrix A ($L = 6$, i.e. 120 predictors):

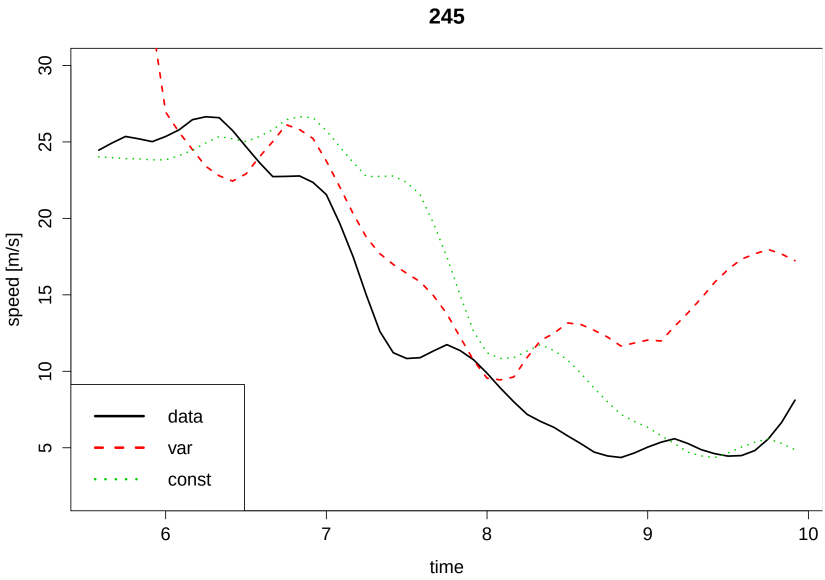


80% of entries are zeros

40 minute forecat with VAR (lasso fit)

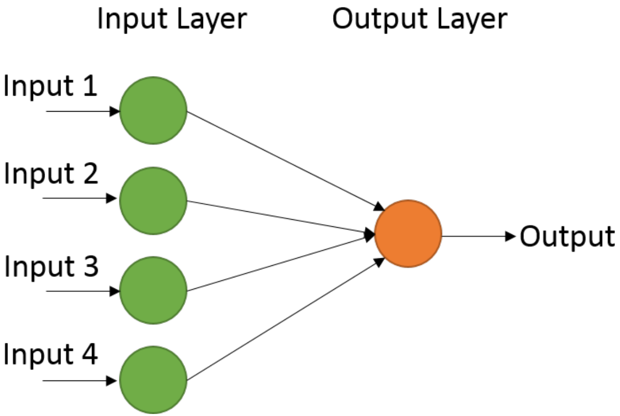


40 minute forecat with VAR (lasso fit)



Neural network models

Simplest version: linear regression

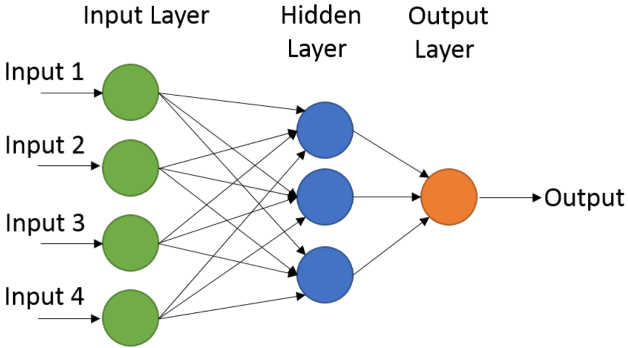


Neural network models

- Coefficients attached to predictors are called weights.
- Forecasts are obtained by a linear combination of inputs.
- Weights selected using a learning algorithm that minimises a cost function.

Neural network models

Nonlinear regression model with one hidden layer



Neural network models

- A multilayer feed-forward network where each layer of nodes receives inputs from the previous layers.
- Inputs to each node combined using linear combination.
- Result modified by nonlinear function before being output

Neural network models

Inputs to hidden neuron j linearly combined:

$$z_j = b_j + \sum_{i=1}^{i} w_{ij} x_i$$

Modified using nonlinear function such as a tanh:

$$s(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

or Rectified Linear

$$s(x) = \max(0, s)$$

This tends to reduce the effect of extreme input values, thus making the network somewhat robust to outliers

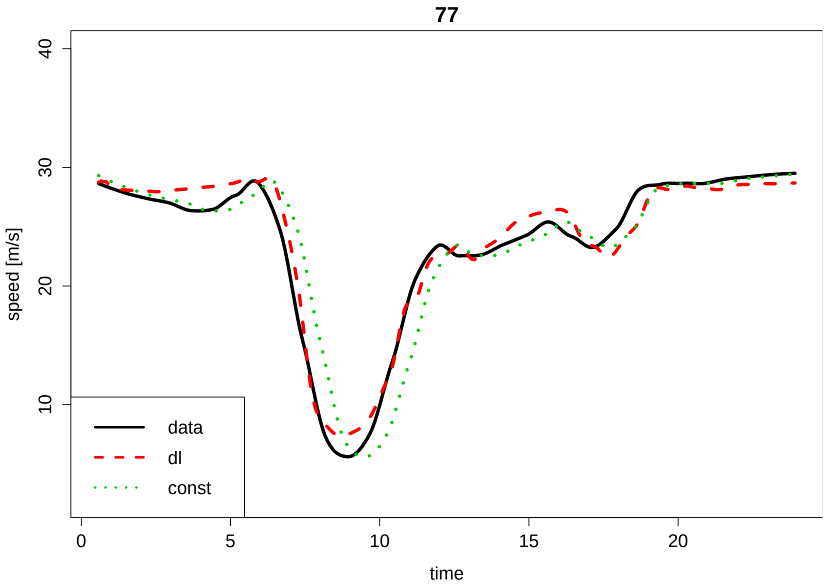
Neural network models

- Weights take random values to begin with, which are then updated using the observed data.
- There is an element of randomness in the predictions. So the network is usually trained several times using different random starting points, and the results are averaged.
- Number of hidden layers, and the number of nodes in each hidden layer, must be specified in advance.

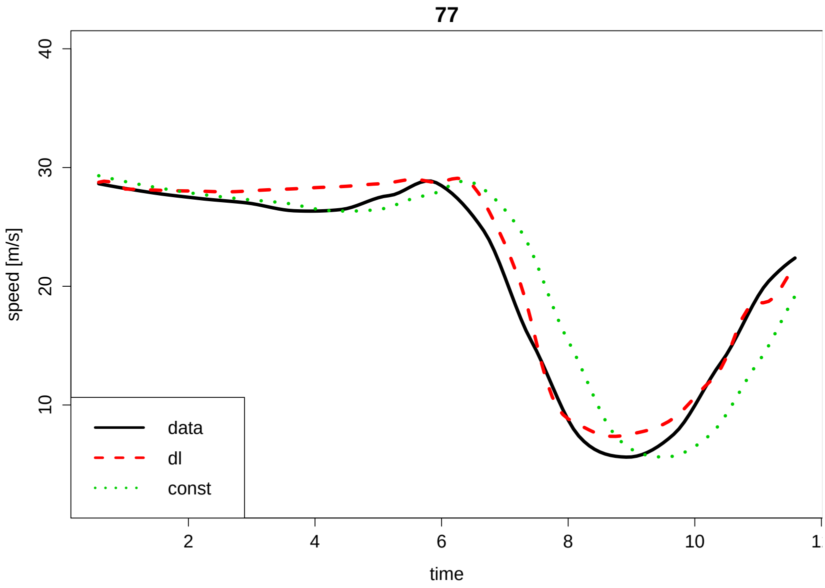
Neural Network models for TIme Series

- Lagged values of the time series can be used as inputs to a neural network.
- Model with no hidden layers is equivalent to an VAR(p) model but without stationarity restrictions.
- Seasonal inputs, with seasonality m: $(x_{t-1}, x_{t-2}, ..., x_{t-p}, x_{t-m}, x_{t-2m}, ..., x_{t-Pm})$ and k neurons in the hidden layer.
- This model is equivalent to an VAR$(p, 0, 0)(P, 0, 0)_m$ model but without stationarity restrictions.

Deep Learning



Deep Learning

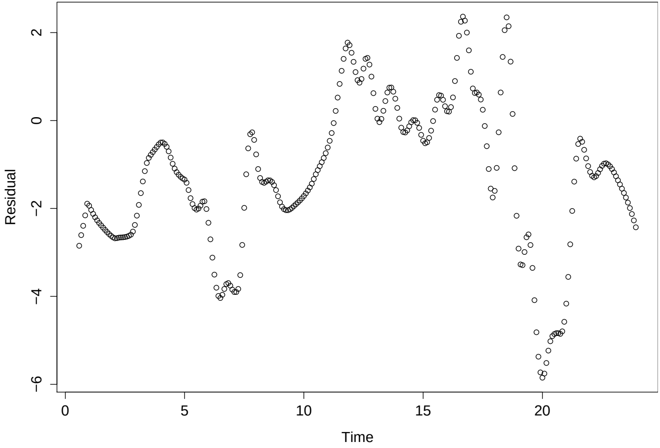


Work in Progress

$$\text{Observation: } y_t = H^T x_t + \gamma^T z_t + v_t, \;\; v_t \sim N(0, V_t \Sigma_t)$$
$$\text{Evolution: } x_t = F_{\alpha_t}(x_{t-1}) + w_t, \;\; w_t \sim N(0, W_t \Sigma_t)$$
$$p(\alpha_{ti} = 1) = \frac{\exp(c + \nu \alpha_{t-1i} + \upsilon Z_{ti})}{1 + \exp(c + \nu \alpha_{t-1i} + \upsilon Z_{ti})}$$

- F_{α_t} is a statistical learning model
- $y_t = (y_{t1}, \dots, y_{tq})^T$, the q-vector of observations at time t
- v_t, the q-vector of observational errors at time t
- $x_t = [x_{t2}, \dots, x_{t2}] \in \mathbb{R}^{\times}$ the matrix whose columns are the state vectors of the individual routes
- α_t is the special event indicator
- Z_{ti} regressors can include x_t and y_t and known forecasts (weather forecast, scheduled road work,...)

Deep Learning Errors are not White Noise



Potential solution add an ARIMA model for the residuals (whiten them).
In this specific case ARIMA(2,1,1) whitens the errors.

Quick Comparison of Computational Costs

	Training	Forecast
KNN	fast	slow
VAR	fast	fast
DL	slow	fast

Discussion

Traffic Flow Model (LWR) Flux function. Traffic jam and prediction

Bayesian Data Analysis Natural approach. Works even though $f_\phi(\theta_t)$ only numerically available Hard to build random walk Metropolis-Hastings algorithms to deal with this problem. Parameter learning helps avoiding estimation bias, can be used for event identification

Statistical Learning A viable competitor to physics-based models. Can do inference with PF, even for non-linear hierarchical models (DL + Logistic Regression + ARIMA)

ADDITIONAL SLIDES

Mixture Kalman Filter For Traffic

$$
\begin{aligned}
\text{Observation:}\ & y_{t+1} = Hx_{t+1} + \gamma^T z_{t+1} + v_{t+1},\ \ v_{t+1} \sim N(0, V_{t+1}) \\
\text{Evolution:}\ & x_{t+1} = F_{\alpha_{t+1}} x_t + (1 - F_{\alpha_{t+1}})\mu + \alpha_t \beta_t + \omega_1 \\
& \beta_{t+1} = \max(0, \beta_t + \omega_2) \\
\text{Switching Evolution:}\ & \alpha_{t+1} \sim p(\alpha_{t+1} | \alpha_t, Z_t)
\end{aligned}
$$

where z_t is an exogenous variable that effects the sensor model, μ is an average free flow speed

$$\alpha_t \in \{0, 1, -1\}$$

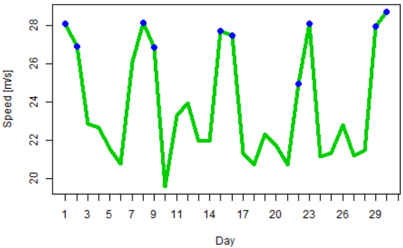
$$\omega = (\omega_1, \omega_2)^T \sim N(0, W),\ \ v \sim N(0, V)$$

$$F_{\alpha_t} = \begin{cases} 1, & \alpha_t \in \{1, -1\} \\ F, & \alpha_t = 0 \end{cases}$$

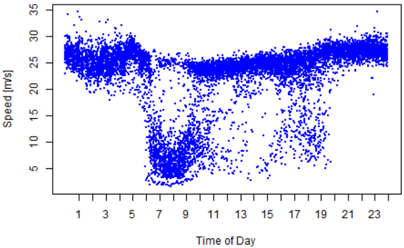
No boundary conditions estimation is needed. Not capacity/critical density is needed.

Choosing $p(\alpha_{t+1}|\alpha_t, Z_t)$

- Weekends are different from weekdays
- Time of day matters
- Weather effects on traffic
- Special Events



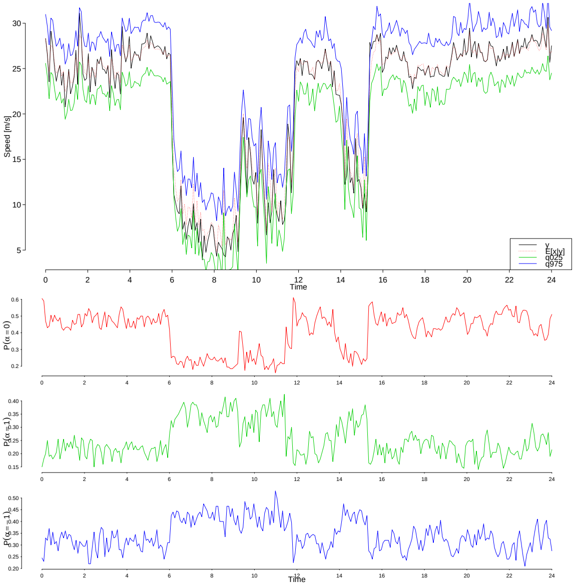
(a) Average Speed per day



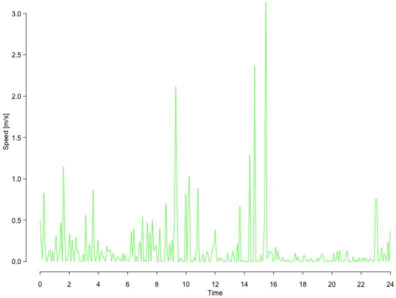
(b) Speed on Wednesdays

Non-parametric regression seems to work well (or choose your favorite machine learning algorithm)

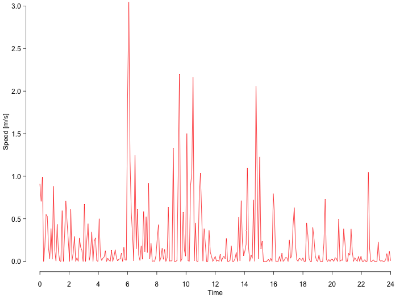
Filtered speed and α (flow regime)



Filtered β (rate of degradation/recovery)



(a) Recovery

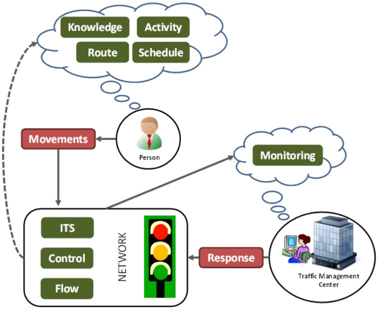


(b) Degradation

POLARIS

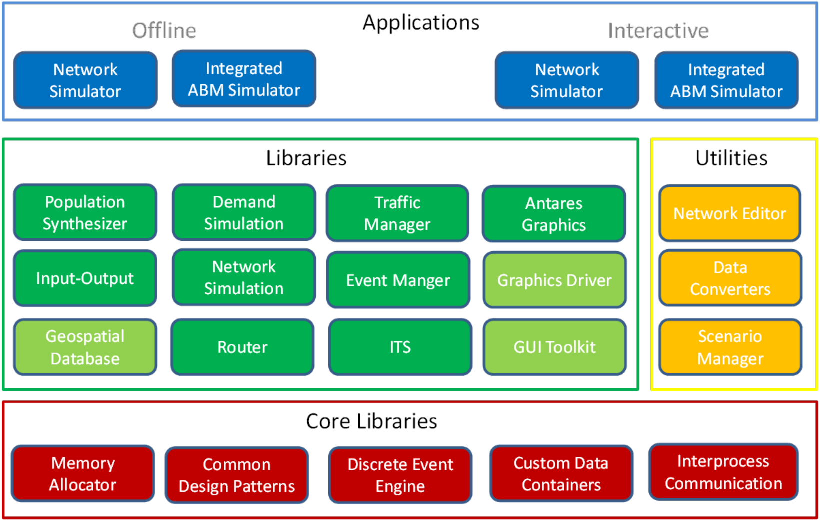
Transport System Modeling Tool

- Agent based Approach
- Traveler is in the center
- All aspects of the traveler's day are modeled explicitly in a single model
- Integrated Network and Demand

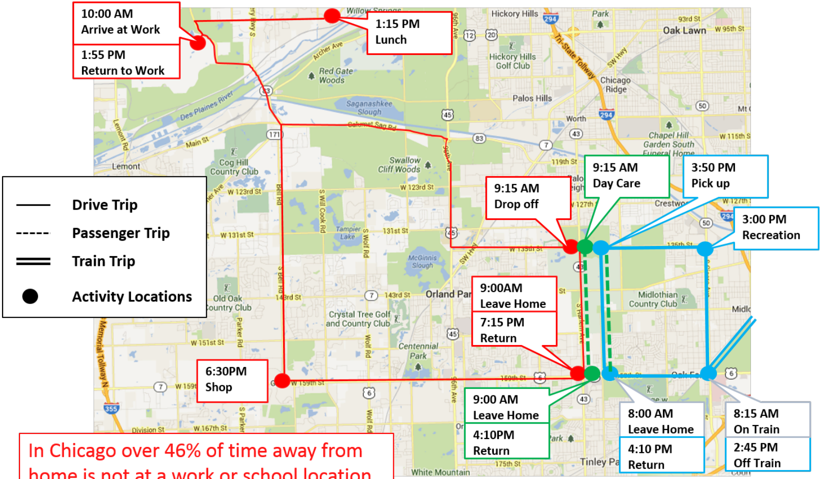


- Physical laws that govern dynamics of traffic flow Newells model
- Managed Lanes
- Controlled intersections
- Traveler information systems
- Traffic management
- Multimodal travel

POLARIS Stacks



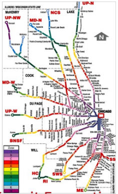
Travel Demand



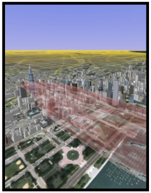
In Chicago over 46% of time away from home is not at a work or school location

Some Applications

Urban Planning
(Chicago Lakeside)



Transit Evacuation
(Chicago)

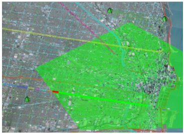


Highway Evacuation
(Chicago Downtown)

CHICAGO 2016
CANDIDATE CITY

Special Events

Chemical Depot Evacuation



Regional Evacuation
(Chicago Metropolitan)

Planning Traffic Management
and Operations

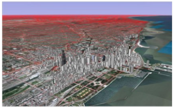
POLARIS and Autonomie

Energy of Transport System; Connected and Automated Vehciles

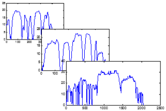
Define fleet
(e.g. from MA3T, Vision)

Transportation Simulation

Define network, control & demand



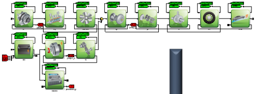
POLARIS



Speed/grade profiles extraction/generation

Powertrain Simulation

Define Powertrains

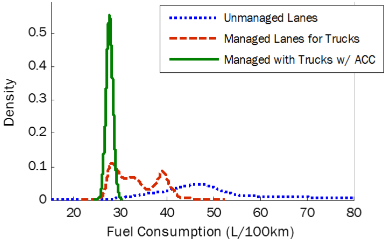


Energy consumption of the transportation network

This tool is designed to evaluate *in a real-world context*: Powertrain technologies, Vehicle Automation, Market Studies, Infrastructure changes, ITS, Modal Changes, etc.

Results

- Reduced capacity for vehicle traffic leads to increase in energy consumed by cars
- Removing slow accelerating trucks from traffic flow allows to improve system energy consumption by 7.8 %



- I-90 (CBD to O'hare)
- 17.5K vehicles simulated, 7.5% HDV
- Scenarios:
  - Unmanaged
  - Managed: left lane for trucks only
  - Managed + ACC: all trucks have adaptive cruise control

- Using Adaptive Cruise Control to allow platoons allows further improve energy consumed by system by another 10%
- The new simulation framework allows to assess a large number of scenarios in a matter of hours

Solving the Conservation Law Equation

Godunov Scheme

Discontinuous solutions (finite differences are not applicable)

- $$\rho_i^{n+1} = \rho_i^n + \frac{\tau}{h}\left(q_G(\rho_{i-1}^n, \rho_i^n) - q_G(\rho_i^n, \rho_{i+1}^n)\right)$$

  where ρ_i^n density value at $x = ih$, $t = n\tau$

  $$q_G(\rho_l, \rho_r) = \begin{cases} q(\rho_l), & \rho_r < \rho_l \leq \rho_m \\ q(\rho_c), & \rho_r \leq \rho_m \leq \rho_l \\ q(\rho_r), & \rho_m \leq \rho_r < \rho_l \\ \min(q(\rho_l), q(\rho_r)), & \rho_l < \rho_r \end{cases} \tag{6}$$

- To include boundary conditions (in flow and out flow).

  $$\rho_0^{n+1} = \rho_0^n + \frac{\tau}{h}\left(q_G(\rho_{-1}^n, \rho_0^n) - q_G(\rho_0^n, \rho_1^n)\right), \text{ with } \rho_{-1}^n = \frac{1}{\tau}\int_{(n-1/2)\tau}^{(n+1/2)\tau} \rho(0,t)dt$$

  and

  $$\rho_M^{n+1} = \rho_0^n + \frac{\tau}{h}\left(q_G(\rho_{M-1}^n, \rho_M^n) - q_G(\rho_M^n, \rho_{M+1}^n)\right), \text{ with } \rho_{M+1}^n = \frac{1}{\tau}\int_{(n-1/2)\tau}^{(n+1/2)\tau} \rho(L,t)dt$$

- Courant-Friedrichs-Lewy type condition: $\tau \leq \frac{h}{|v_{max}|}$, guarantees stability

Godunov Schema

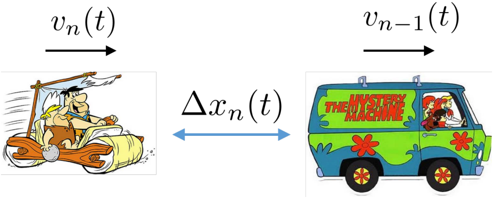
Solves Riemann problem for each 2 consecutive cells.
Riemann = Cauchy problem (initial value problem) with initial conditions that have a single discontinuity

$$\rho_0(x) = \begin{cases} \rho_l, & x < 0 \\ \rho_r, & x > 0 \end{cases} \tag{7}$$

For th Riemann problem the speed of the of the shock wave propagation is given by

$$w = \frac{q(\rho_l) - q(\rho_r)}{\rho_l - \rho_r}$$

Microscopic Simulation (Car Following)



General Motors (GM) based car-following models: acceleration is a response to the stimulus (force from interaction)

$$\dot{v}(t) = \alpha \frac{v_n(t)^m}{(\Delta x_n(t))^l} \left(v_{n-1}(t-\tau) - v_n(t-\tau)\right)$$

α, l, m: parameters of the model

Simulation Classification

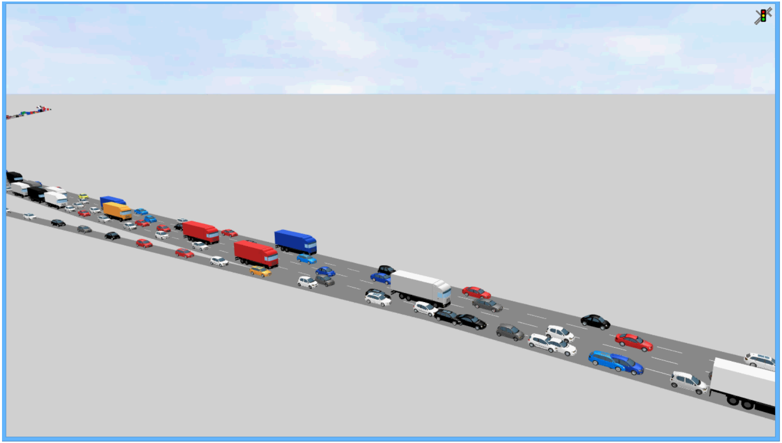
The simulation depends on the granularity of the representation

- Macroscopic
  - Space is continuous/discretized
  - Time is discrete
- Microscopic
  - Individual vehicles are represented
  - Individual interactions between vehicles are modeled

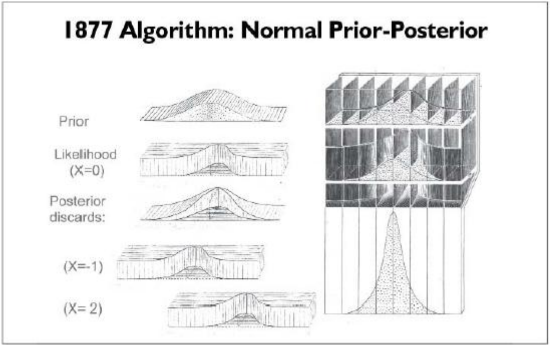
Microscopic Simulation

- Car Following
  - Distance-based models (stimulus is a function of Δx)
  - Psycho-physical (reaction is function of Δv, and Δx)
  - Other acceleration models (might be different for free flow vs congested regime)
- Lane changing
- Gap acceptance

Microscopic Simulation Demonstration (I-90 NB in Chicago)

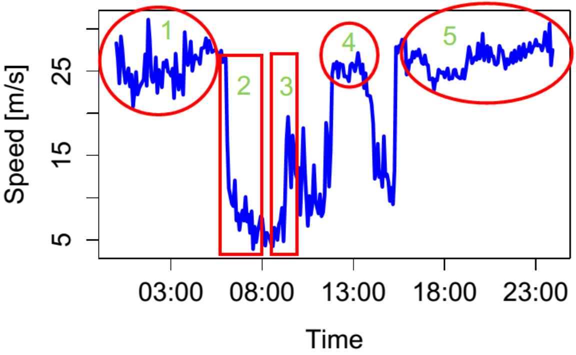


Galton 1877: First Particle Filter



Example

of one day speed profile from May 14, 2009 (Thursday)



LWR Model Derivation

Use the cumulative flow $N(x,t)$, number of vehicles that pass location x by time t. Then the conservation law can be derived by evaluating

$$\frac{\partial N}{\partial t} = q(x,t), \ \frac{\partial N}{\partial x} = -\rho(x,t)$$

Assuming that N is smooth

$$\frac{\partial^2 N}{\partial x \partial t} = \frac{\partial^2 N}{\partial t \partial x}$$

Streaming Data

Online Learning

Construct an essential state vector θ_{t+1}.

$$p(\theta_{t+1}|y^{t+1}) = \int p(\theta_{t+1}|\theta_t, y_{t+1}) \; d\mathbb{P}(\theta_t|y^{t+1})$$

$$\propto \int \underbrace{p(\theta_{t+1}|\theta_t, y_{t+1})}_{propagate} \overbrace{\underbrace{p(y_{t+1}|\theta_t)}_{resample} \; d\mathbb{P}(\theta_t|y^t)}$$

1. *Re-sample* with weights proportional to $p(y_{t+1}|\theta_t^{(i)})$ and generate $\{\theta_t^{\zeta(i)}\}_{i=1}^N$
2. *Propagate* with $\theta_{t+1}^{(i)} \sim p(\theta_{t+1}|\theta_t^{\zeta(i)}, y_{t+1})$ to obtain $\{\theta_{t+1}^{(i)}\}_{i=1}^N$

Parameters: $p(\theta|\theta_{t+1})$ drawn "offline"