How can we improve bus ETAs?

Using real-time position data to estimate road state

Tom Elliott and Thomas Lumley

Department of Statistics, University of Auckland, New Zealand

`tom.elliott@auckland.ac.nz`

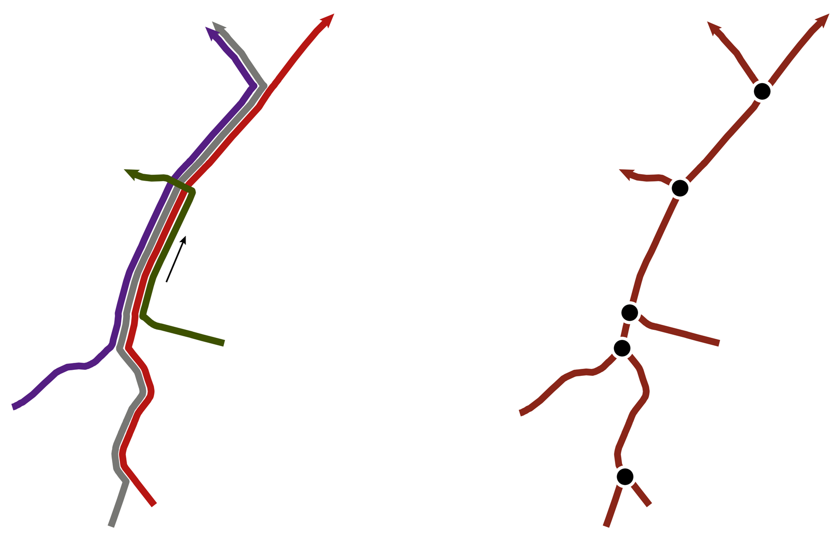
1. Introduction

- **real-time information (RTI)**: estimated arrival time (ETA), delays, cancellations
- helps commuters **plan journeys** and **improves their experience** [?] ... but **only if the information is reliable!**
- generating ETAs involves 1. a **real-time vehicle tracking** system (e.g., GPS), 2. a **vehicle state model** to process real-time noisy observations (e.g., Kalman filter [**?**, **?**, **?**], particle filter [**?**]), and 3. **travel time predictions**
- predictions often based on scheduled inter-stop travel times, occasionally historical data; however **real-time travel times** along intermediate roads would seem to be the best predictor

- **proposal**: an approach to modeling transit vehicles and network congestion to obtain reliable ETAs
- **test location**: Auckland, New Zealand, where Auckland Transport provides free, public transit API `https://dev-portal.at.govt.nz/`

2. GTFS transit network

- **GTFS**: API specification for transit data [**?**], 500+ locations worldwide
  - static: routes, **shapes**, stops, scheduled arrival/departure times
  - realtime: **vehicle locations**, arrival/departure delays
- **transit network** consists of intersections (nodes) and connecting road segments (edges)
- our general method for constructing network from raw GTFS data:
  1. Import raw GTFS shape data
  2. generate network of intersections (nodes) and road segments (edges) using adaptation of [**?**]
  3. express each route as a sequence of road segments
- Implementation in progress: `gtfsnetwork` R package



3. Vehicle state model

- estimate vehicle state $\boldsymbol{X}_k$ from a sequence of real-time GPS observations $\boldsymbol{Y}_k$
- **transition function** f describes behaviour of a bus: acceleration/deceleration and wait times at bus stops/intersections, with system noise Q_{k-1} (in vehicle speed)

$$\boldsymbol{X}_k = f(\boldsymbol{X}_{k-1}, w_k), \quad w_k \sim N(0, Q_{k-1})$$

- **measurement function** h determines GPS coordinates for a known state using GTFS shape and distance traveled, so measurement model is

$$\boldsymbol{Y}_k = h(\boldsymbol{X}_k)$$

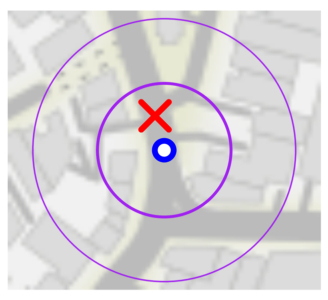
(we use an equirectangular projection g to work with geographic coordinates)

- **likelihood**: given $\hat{\boldsymbol{X}}_k$, define distance between $h(\hat{\boldsymbol{X}}_k)$ and $\boldsymbol{Y}_k$

$$\delta_k = d(h(\hat{\boldsymbol{X}}_k), \boldsymbol{Y}_k)$$

then δ_k^2 is the sum of two independent normal random variables with variance σ_y^2

$$\left(\delta_k^2 / \sigma_y^2\right) \sim \chi^2(2) \sim \text{Exp}(0.5)$$

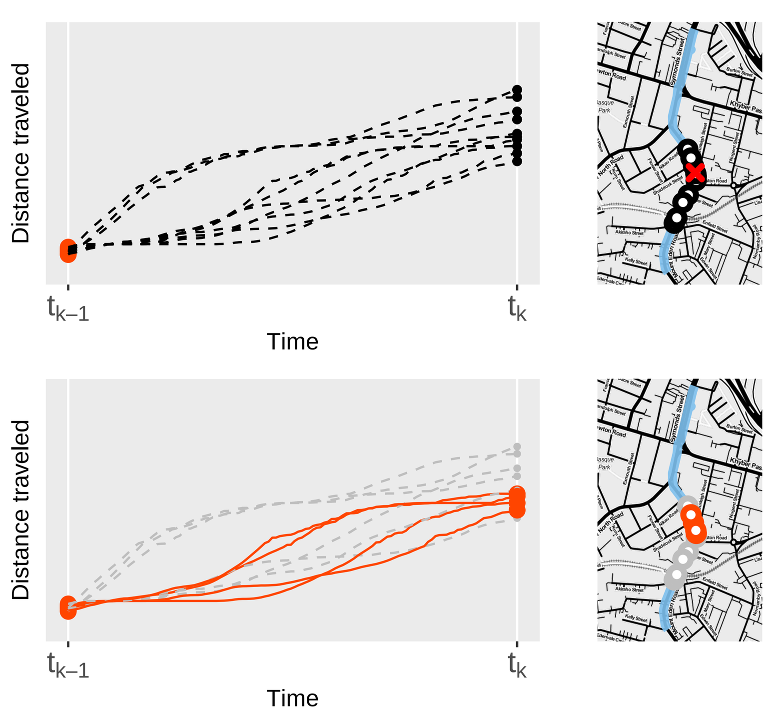


- **particle filter**: flexible estimation method approximating $\boldsymbol{X}_k$ using particles $(\boldsymbol{X}_k^{(i)})_{i=1}^N$
  1. **predict new state** by transitioning particles up to time t_k
  2. **evaluate likelihood** of each particle

$$p(\boldsymbol{Y}_k | \boldsymbol{X}_k^{(i)}) = 0.5 e^{-(\delta_k^{(i)})^2 / 2\sigma_y^2}$$

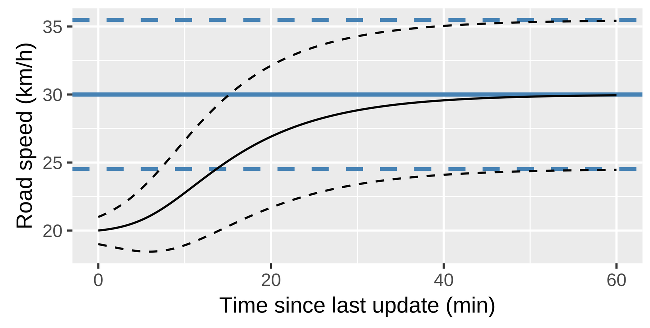
  3. **weighted resample with replacement**

$$w^{(i)} = \frac{p(\boldsymbol{Y}_k | \boldsymbol{X}_k^{(i)})}{\sum_{j=1}^N p(\boldsymbol{Y}_k | \boldsymbol{X}_k^{(j)})}$$

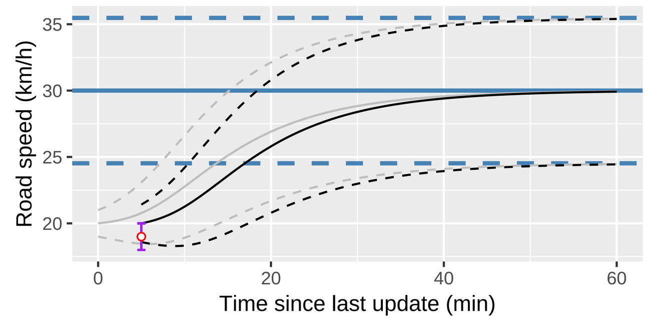


4. Network state model

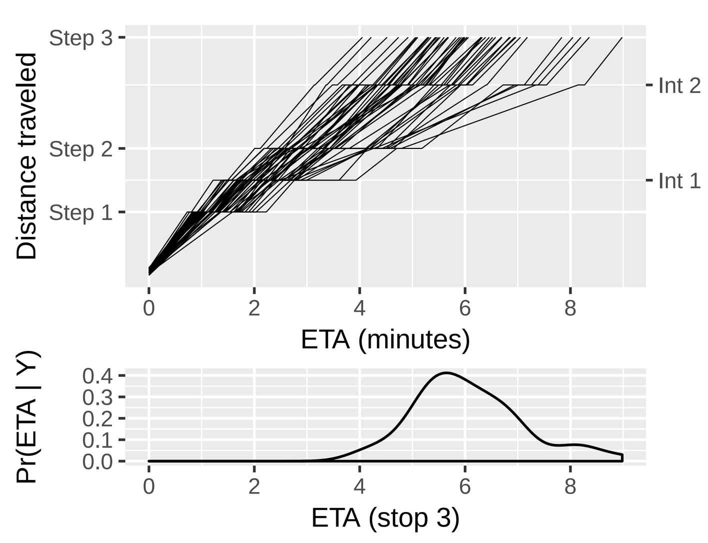
- estimate **real-time network state** from vehicle speeds, and forecast future states for ETAs
- β_r^j is average **speed of buses** along road segment j at time t_r
- forecasted state converges towards **historical mean** (i.e., the prior)



- update as vehicles traverse network using extended Kalman filter algorithm
  - observation $\hat{s}_t$ is mean speed of particles along segment
  - measurement error r_t^2 is variance of particle speeds



5. Predicting arrival time



- **simulate particle trajectories** using forecasted segment speeds
- obtain **distribution of arrival times**, $(A_j^i)_{i=1}^N$ for each stop j
- provide commuters with summary statistics of distribution, for example
  - a **point estimate** of 5 minutes
  - a **prediction interval** of 4–8 minutes
- we want ETAs that **decrease with time** while also **minimising** $\Pr(A_j < \hat{A}_j | \boldsymbol{X}_k)$

6. Conclusion

- segmenting routes allows vehicles to share travel times with others using the same roads
- real-time vehicle and network state models combine real-time and historical data to predict arrival time
- real-time C++ implementation takes up to 20 seconds on an 8-core Virtual Machine with 4000 particles per vehicle (of which there can be 1000+ at peak times)

7. Future work

- tune particle filter to perform better, faster resampling, to allow increasing N
- improve network state model:
  - **non-constant segment speed**: speed varies by time and distance along road
  - **additional covariates**: adjacent segments, yesterday's traffic, weather, etc.
- stop- and intersection-wait time models to estimate and quantify wait time uncertainty
- find optimal point and interval estimates for ETAs