# Trajectory Regression on Networks

Tsuyoshi Idé (IBM Research – Tokyo)

Joint work with Prof. Masashi Sugiyama

JFFoS 2012, Nice, France

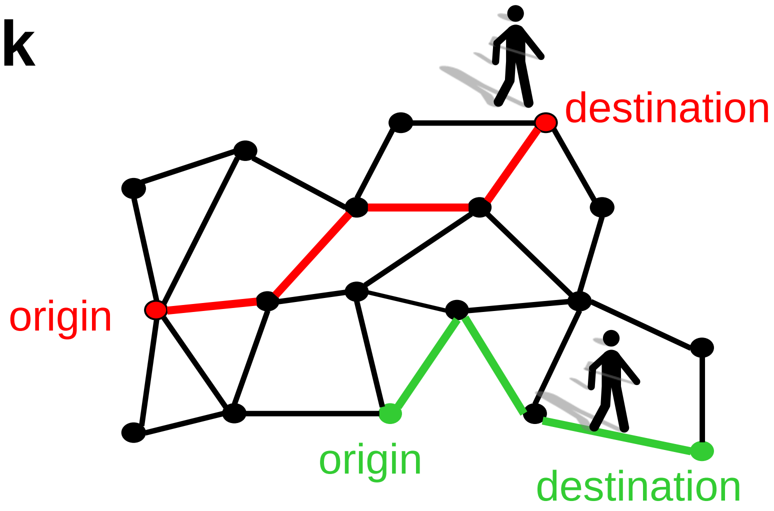
# Problem: Predict the "cost" of an arbitrary path on networks

□ **Input: arbitrary path on a network**

– A sequence of adjacent link IDs

□ **Output: total cost of the path**

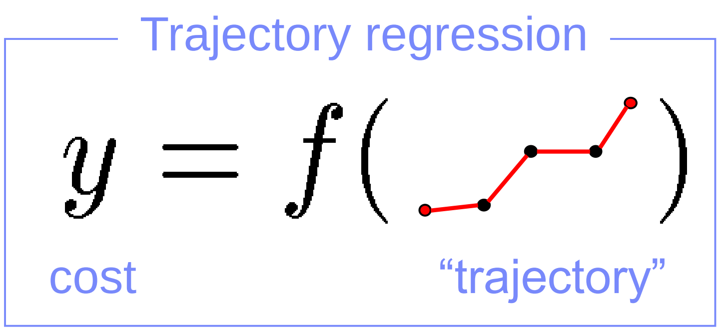
– Scalar (e.g. travel time)




□ **Training data:**

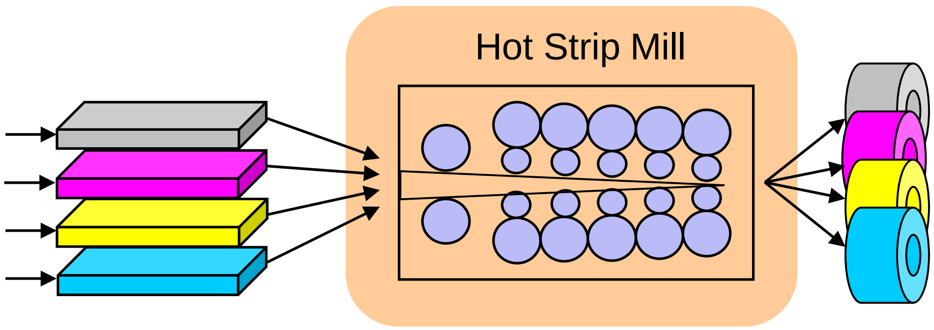
$$\mathcal{D} \equiv \{(x^{(n)}, y^{(n)}) | n = 1, 2, ..., N\}$$

- $x^{(n)}$ : *n*-th trajectory (or path)
- $y^{(n)}$ : *n*-th cost

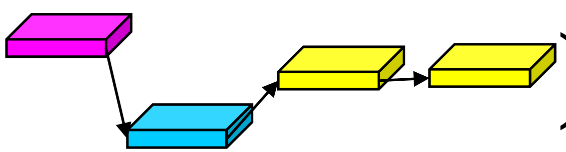


# Why is this problem interesting?

- **This is a new regression problem**
  – The input is not a vector
- **This task appears in several real problems**
  – Travel-time prediction on road networks
  – Energy consumption prediction in steel rolling process



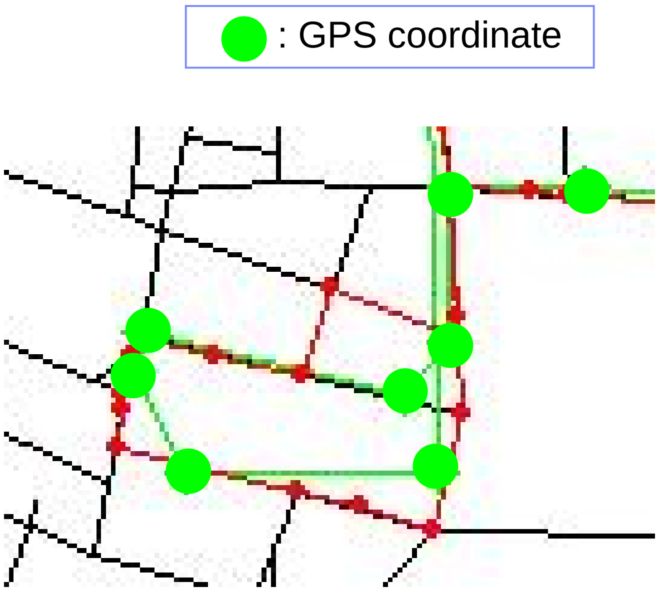

Total amount of electricity over 30 min can be measured

Energy = $f($  $)$

# We focus on travel-time prediction

- **GPS produces a sequence of sparse and noisy coordinate points**
  - Very hard to measure the cost of individual links precisely
- **Total cost can be precisely measured even in that case**
  - Simply computed as the time difference between O and D:

$$t_{\mathrm{destination}} - t_{\mathrm{origin}}$$




Brakatsoulas et al., "On map-matching vehicle tracking data," Proc. VLDB '05, pp.853--864

# Agenda

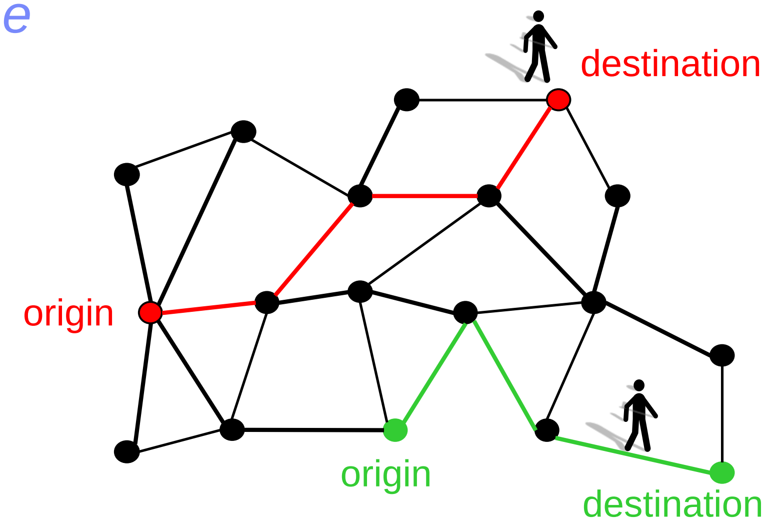
- **Problem setting**
- **Formulation**
- **Relationship with Gaussian process regression**
- **Experiments**

# Representing the total cost using latent variables $\{f_e\}$

$$y(x) = \sum_{e \in x} \boxed{C_e}$$

cost of link *e*

for all links included input path *x*




**Our goal is to find cost deviation $\{f_e\}$ from the baseline**

$$C_e \equiv l_e(\phi_e^0 + f_e)$$

Link length (known)

Baseline unit cost

(known from e.g. legal speed limit)

**Cost deviation per unit length from the baseline**

# Setting two criteria to determine the value of $\{f_e\}$

- **The observed total cost must be reproduced well**
  - Minimize:

$$\sum_{n=1}^{N}\left(y^{(n)} - \boxed{\sum_{e\in \boldsymbol{x}^{(n)}} C_e(f_e)}\right)^2$$

Estimated cost for a path $x^{(n)}$

Observed cost for $x^{(n)}$

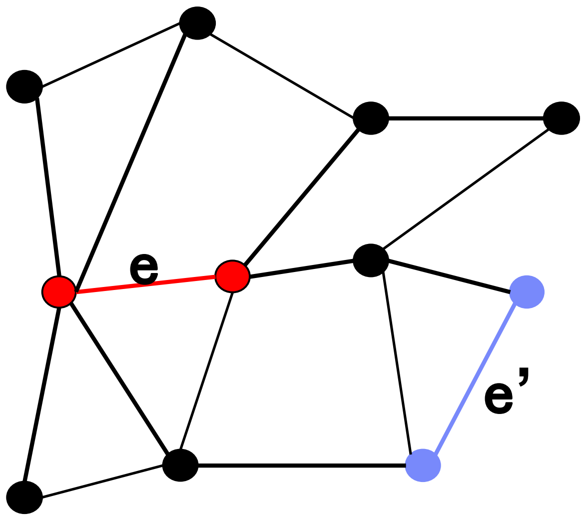
- **Neighboring links should have similar values of cost deviation**
  - While keeping this constant

$$\sum_{e=1}^{M}\sum_{e'=1}^{M} S_{e,e'}\,|f_e - f_{e'}|^2$$

Similarity between link e and e'

# Example of definition of link similarities

$$S_{e,e'} \equiv \begin{cases} \omega^{1+d(e,e')}, & d(e,e') \leq d_0 \\ 0, & \text{otherwise} \end{cases}$$




*d* = (# of hops between edges)

In this case, *d*(*e*,*e*')=2

# Final objective function to be minimized

$$\Psi(\boldsymbol{f}|\lambda) = \sum_{n=1}^{N} \left( y^{(n)} - \sum_{e \in \boldsymbol{x}^{(n)}} c_e(f_e) \right)^2 + \frac{\lambda}{2} \sum_{e=1}^{M} \sum_{e'=1}^{M} S_{e,e'} |f_e - f_{e'}|^2$$

- "Predicted cost should be close to observed values"
- "Neighboring links should take similar values"
- ***S*** is the similarity matrix between links

# This optimization problem can be analytically solved

## Matrix representation of the objective

$$\Psi(\boldsymbol{f}|\lambda) = \left\|\boldsymbol{y}_N - \mathsf{Q}^\top \boldsymbol{f}\right\|^2 + \lambda \boldsymbol{f}^\top \mathsf{L}\, \boldsymbol{f}$$

Graph Laplacian induced by the link-similarity Matrix

$$\mathsf{L}_{i,j} \equiv \delta_{i,j} \sum_{k=1}^{M} S_{i,k} - S_{i,j}$$

"Data matrix" of the trajectories

$$\mathsf{Q} \equiv [\boldsymbol{q}^{(1)}, ..., \boldsymbol{q}^{(N)}] \in \mathbb{R}^{M \times N}$$

$$q_e^{(n)} = \begin{cases} l_e, & \text{for } e \in \boldsymbol{x}^{(n)} \\ 0, & \text{otherwise} \end{cases}$$

Vector of the trajectory costs

$$\boldsymbol{y}_N \equiv [y^{(1)}, y^{(2)}, ..., y^{(N)}]^\top$$

## Analytic solution

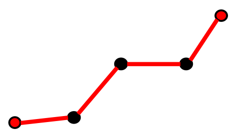
$$\boldsymbol{f} = \left[\mathsf{Q}\mathsf{Q}^\top + \lambda \mathsf{L}\right]^{-1} \mathsf{Q}\boldsymbol{y}_N$$

Lambda is determined by cross-validation

# Agenda

- **Problem setting**
- **Formulation**
- **Relationship with Gaussian process regression**
- **Experiments**

# The link-Laplacian matrix bridges the two approaches

$y = f($  $)$

**Kernel-based** [Idé-Kato 09]

**Proposition:**

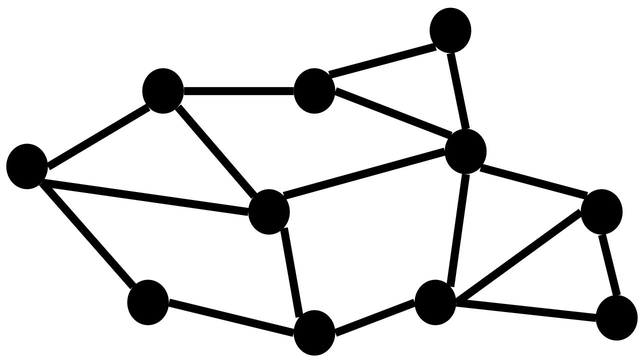
These two approaches are equivalent if the kernel matrix is chosen as

$$\mathsf{K}_{n,n'} = \boldsymbol{q}^{(n)\top} \mathsf{L}^{\dagger} \boldsymbol{q}^{(n')}$$

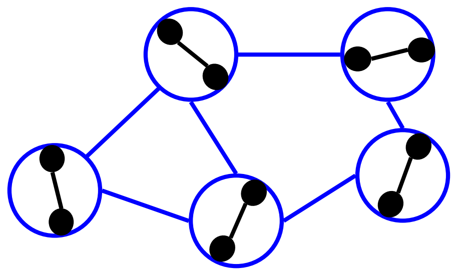
**Feature-based** (this work)

# Practical implications of the link-Laplacian kernel

- **The proposition suggests a natural choice of kernel**
  - In the kernel regression approach, the choice of kernel is based on just an intuition
  - Since a link-similarity is much easier to define, our approach is more practical than the kernel method



(a) Road network



(b) Link network

# Agenda

- **Problem setting**
- **Formulation**
- **Relationship with Gaussian process regression**
- **Experiments**

# Traffic simulation data on real road networks

- **Network data**
  - Synthetic 25x25 square grid
  - Downtown Kyoto
- **Trajectory-cost data**
  - Generated with IBM's agent-based traffic simulator
  - Data available on the Web
- **Method compared**
  - Legal: perfectly complies with the legal speed limit
  - GPR: Gaussian process regression with a string kernel
  - RETRACE: proposed method

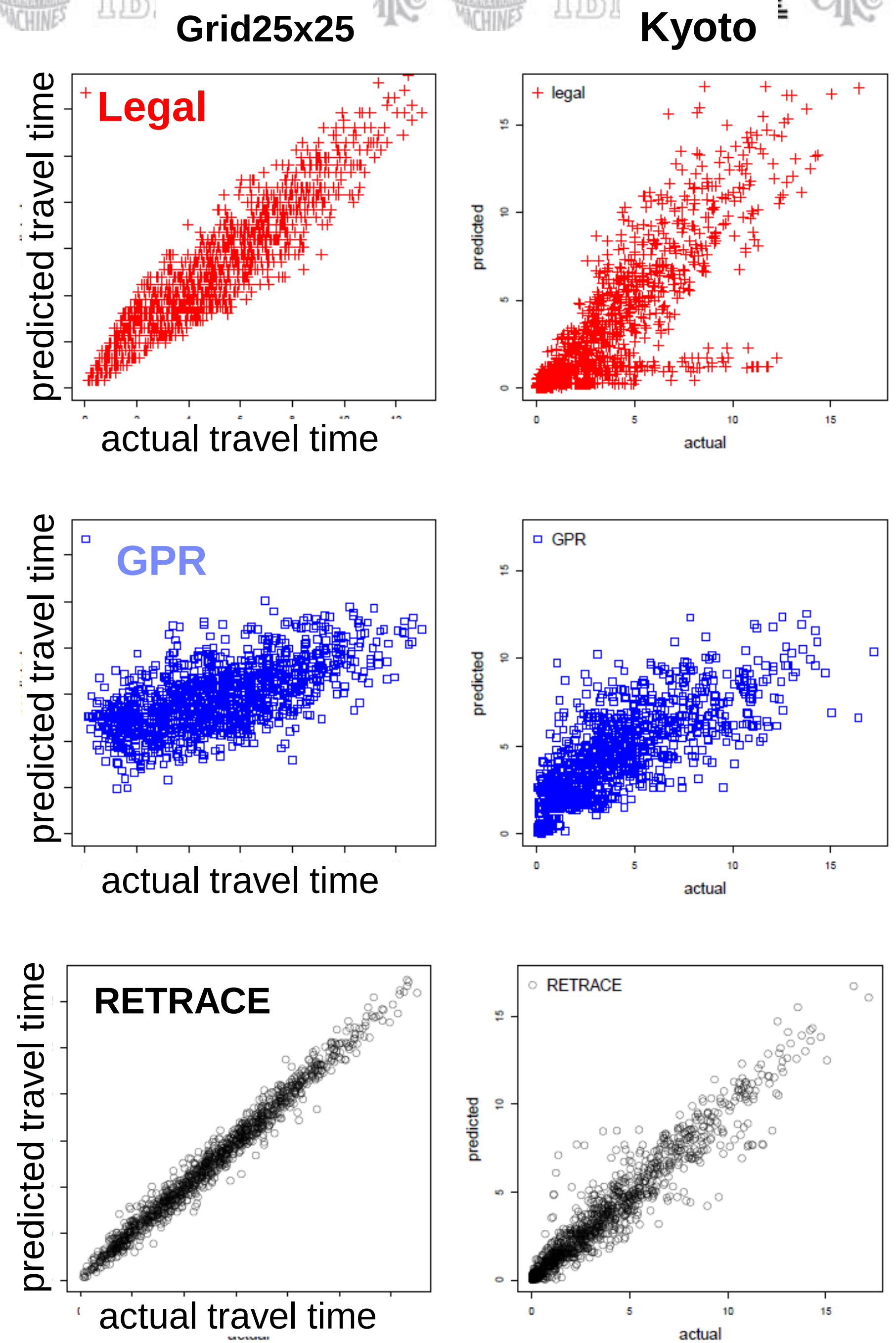

Figure 2: Downtown Kyoto City map of Kyoto data.

Table 1: Summary of data.

| | Grid25x25 | Kyoto |
|---|---|---|
| # nodes | 625 | 1 518 |
| # links | 2 400 | 3 478 |
| # generated paths | 1 200 | 1 739 |

# Proposed methods gave the best performance

- **Legal**
  - Does not reproduce traffic congestion in Kyoto
- **GPR**
  - Worst
  - Due to the choice of kernel that can be inconsistent to the network structure
- **RETRACE** (proposed)
  - Clearly the best
  - Outliers still exist: cannot handle dynamic changes of traffic



# Concluding remarks

- **Summary**
  - Proposed the RETRACE algorithm for trajectory regression
  - RETRACE has an analytic solution that is easily implemented
  - Gave an interesting insight to the relationship with the GPR approach

- **Future work**
  - To test the algorithm using probe-car data
  - To study how to improve the scalability

For technical details:
T. Ide and M. Sugiyama, “Trajectory Regression on Road Networks,” Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence (AAAI-11), pp.203-208, 2011.