`trajectories`: Classes, methods and data analysis

Mehdi Moradi
Joint work with Edzer Pebesma and Jorge Mateu

Public University of Navarra
Department of Statistics, Informatics and Mathematics
Pamplona, Spain

Moving objects

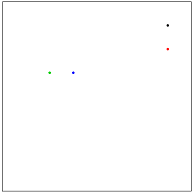
Classes

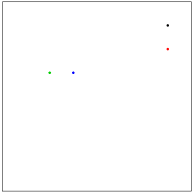
Methods

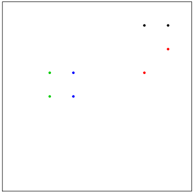
Exploratory data analysis

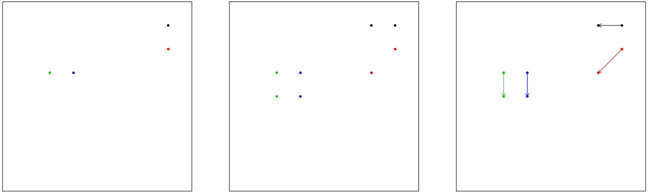
Summary

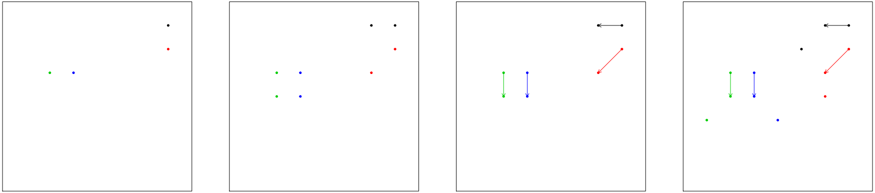
Moving objects

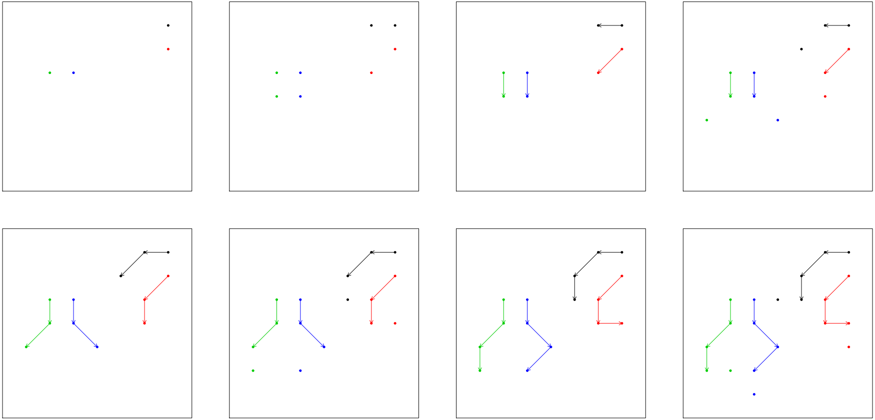












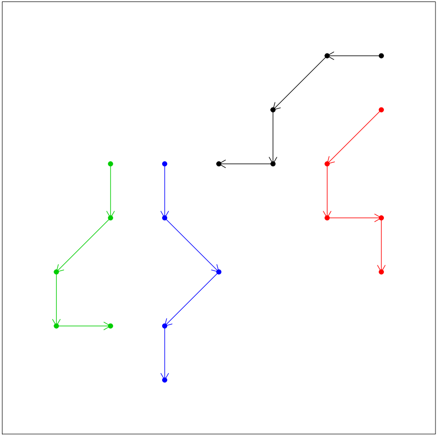


Figure: Trajectory pattern.

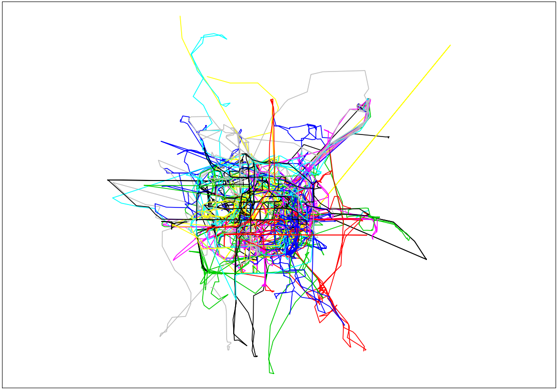


Figure: Movement of 50 taxis in Beijing (China) during the period of Feb. 2 to Feb. 8; 2008.

Classes to handle movement data in `trajectories` R package:

- **Track:** Single track followed by a moving object.
- **Tracks:** Collection of tracks followed by a single moving object.
- **TracksCollection:** Collection of tracks followed by a group of moving objects.
    - Inner/nested classes: list, data.frame, SpatialPoints, xts/zoo, POSIXct/POSIXt.

Method	Operation
dim	number of spatial points
summary	Summarises the internal information
proj4string	projection attributes
coordinates	coordinates of spatial locations
coordnames	coordinate names of fixes
bbox	The box which contains the objects
stbox	The spatio-temporal box which contains the objects
aggregate	Spatially aggregate track properties (coercing fixes to points)
compare	Compares two Track objects: for the common time period
dists	distance matrix with distances for each pair of tracks
downsample	Remove fixes from a Track
frechetDist	Fréchet distance between two Track objects
stcube	Draw a space-time cube
stplot	Create trellis plot for TracksCollection objects
cut	Obtain ranges of space and time coordinates

- Simulation techniques (see functions `rTrack, rTracks, rTracksCollection`) and fitting ARIMA models (see function `auto.arima.Track`).

trajectories: analysing movement data

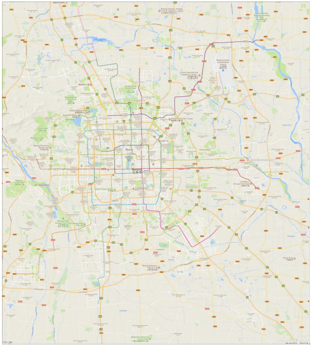


Figure: Beijing (China) and its surrounding.

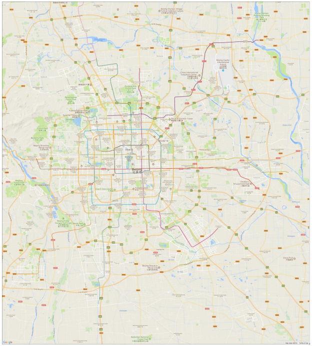


Figure: Beijing (China) and its surrounding.

- 10,357 taxis,
- During one week (Feb. 2 a Feb. 8, 2008),
- **cleaned data:**
  - 5642 taxis,
  - The total number of points is more than 10 millions,
  - Total distance passed by taxis reaches 7 millions Km.

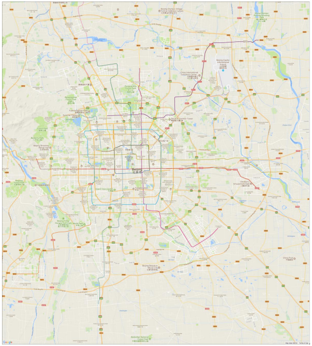


Figure: Beijing (China) and its surrounding.

- 10,357 taxis,
- During one week (Feb. 2 a Feb. 8, 2008),
- **cleaned data:**
  - 5642 taxis,
  - The total number of points is more than 10 millions,
  - Total distance passed by taxis reaches 7 millions Km.

- `install.packages("taxidata", repos = "http://pebesma.staff.ifgi.de" ,type = "source")`

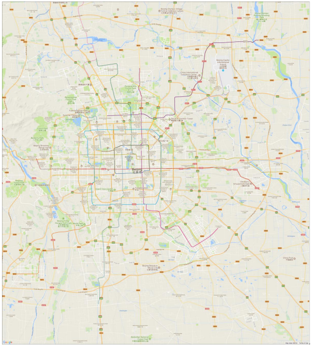


Figure: Beijing (China) and its surrounding.

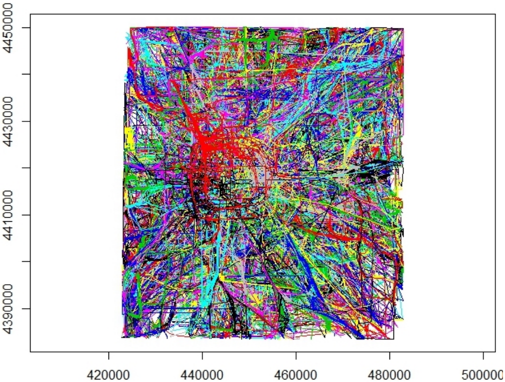


Figure: Taxis movement pattern.

Algorithm 1 Trajectory pattern $S = \{s_1, \dots, s_n\}$ as series of spatial point patterns

1: Retrieve the range of time of S using the function range in trajectories.
2: Create a regular sequence of time based on the obtained range and a given timestamp using the function tsqTracks. Let say $T = \{t_1, t_2, \dots, t_m\}$ is the obtained regular sequence of time.
3: Reconstruct each $s_i, i = 1, \dots, n$, using the function reTrack and based on the sequence of time T.
4: The corresponding locations (points) to each $t_i, i = 1, \dots, m$, is a spatial point pattern and is denoted by $\mathbf{x}_i, i = 1, \dots, m$.

- See function `as.Track.ppp`

Distance analysis

The average pairwise distances for point pattern $\mathbf{x}_i, i = 1, \ldots, m$ is then of the form

$$\overline{D}_i = \frac{2}{n_i(n_i - 1)} \sum_{h \neq k} \|x_i^k - x_i^h\|, \qquad 1 \leq h, k \leq n_i. \qquad (1)$$

- See function `avedistTrack`

Distance analysis

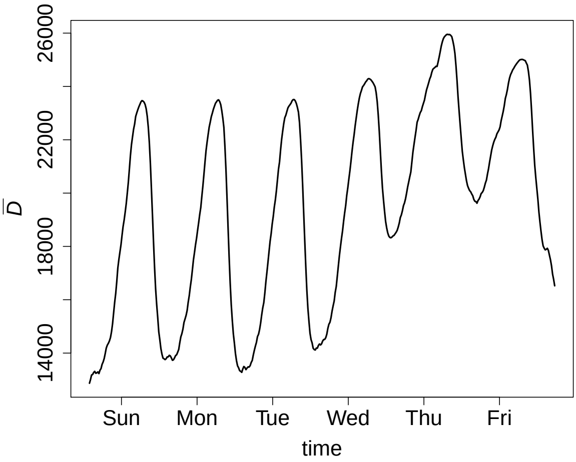


Figure: Average pairwise distance between taxis in Beijing (China). Within the period 2-8, Feb 2008.

Distance analysis

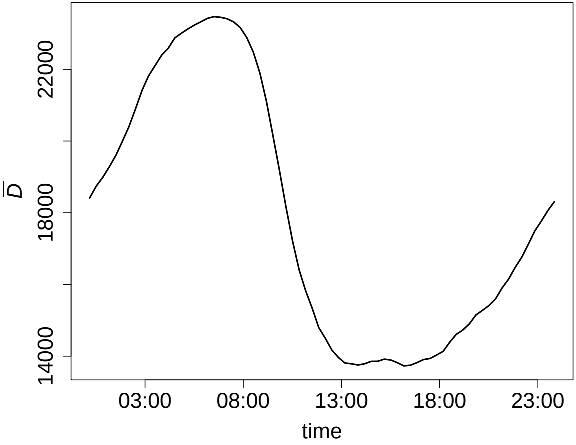


Figure: Average pairwise distance between taxis in Beijing (China). During 3*rd* of Feb 2008.

Assume a situation where the data points $\{x_1, x_2, \ldots, x_n\}$ are marked by numeric values $\{z_1, z_2, \ldots, z_n\}$. Then the smoothed value at an arbitrary location $u \in W$, W is the observed window of our data points, is

$$Q(u) = \frac{\sum_{j=1}^{n} z_j \xi_j}{\sum_{j=1}^{n} \xi_j}, \tag{2}$$

where

$$\xi_j = \frac{1}{(\|u - x_j\|)^p}, \tag{3}$$

that p is an integer (Baddeley and Turner, 2005; Baddeley et al., 2015).

Algorithm 2 Trajectory pattern $S = \{s_1, \ldots, s_n\}$ as series of segment patterns

1: Follow the steps in Algorithm 1.
2: Using each two consecutive point patterns, say $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$, $i = 1, \ldots, m-1$, make a segment pattern by connecting the current location of each taxi to its previous location.
3: Call the segment patterns $\mathbf{y}_1, \ldots, \mathbf{y}_{m-1}$.

- See function `as.Track.arrow`

Convert obtained segment patterns $\mathbf{y}_1, \ldots, \mathbf{y}_{m-1}$ from Algorithm 2 to $m-1$ marked point patterns in which points are the mid-points of segments and marks are the length of segments.

- See function `idw.Track`

Movement smoothing

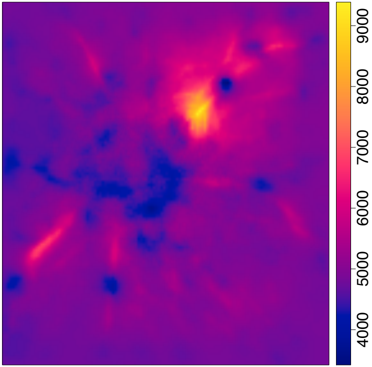


Figure: Movement (per 20 minuets and longer than 1km) smoothing for taxi data in Beijing (China).

Intensity estimation

We next propose an average intensity estimate so that we first convert a trajectory pattern S to spatial point patterns $\mathbf{x}_1, \ldots, \mathbf{x}_m$, and then we estimate the intensity of each pattern $\mathbf{x}_i, i = 1, \ldots, m$, denoted by $\widehat{\lambda}_1, \ldots, \widehat{\lambda}_m$, finally the estimated intensity of trajectory pattern S is of the form

$$\overline{\lambda}(u) = \frac{1}{m} \sum_{i=1}^{m} \widehat{\lambda}_i(u), \qquad u \in W, \tag{4}$$

where $\widehat{\lambda}_i(\cdot), i = 1, \ldots, m$, can be obtained from any non-parametric or parametric method.

- See function `density.list`

Intensity estimation

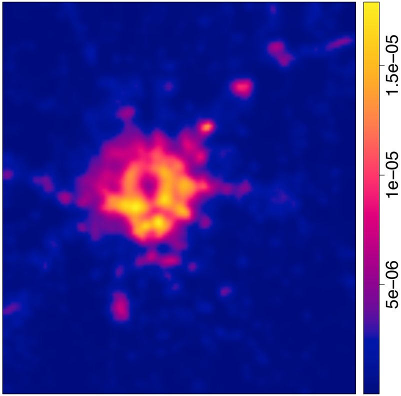


Figure: Estimated intensity of the trajectory pattern of taxi data in Beijing (China).

Intensity estimation

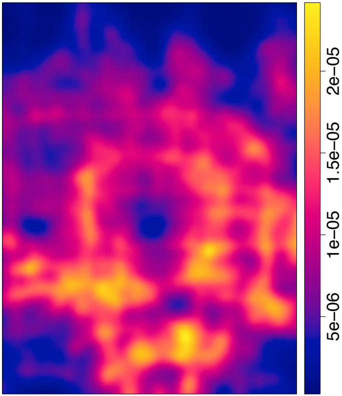


Figure: Focus on the metropolitan area of Beijing (China).

Chi-squared maps

Let $\lambda_1^{\text{est}} = \widehat{\lambda}_1, \ldots, \lambda_m^{\text{est}} = \widehat{\lambda}_m$ and

$$\lambda_j^{\text{exp}}(u) = \frac{\sum_{i=1}^{m} \lambda_i^{\text{est}}(u) \sum_{\nu \in W} \lambda_j^{\text{est}}(\nu)}{\sum_{i=1}^{m} \sum_{\nu \in W} \lambda_i^{\text{est}}(\nu)}, \qquad 1 \leq j \leq m. \qquad (5)$$

We call $\lambda_j^{\text{exp}}(u)$ the expected intensity at location u and time t_j. Define the χ^2 statistics as

$$\chi_j^2(u) = \frac{\lambda_j^{\text{est}}(u) - \lambda_j^{\text{exp}}(u)}{\sqrt{\lambda_j^{\text{exp}}(u)}}, \qquad 1 \leq j \leq m. \qquad (6)$$

Chi-squared maps

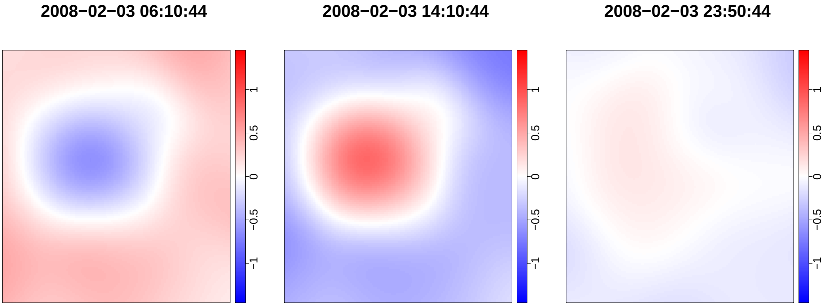
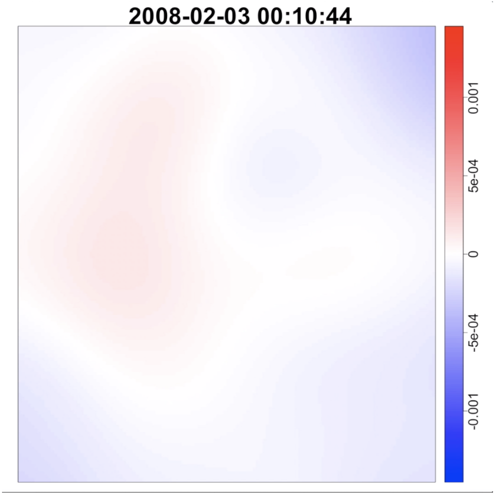


Figure: Chi-squared maps (values are multipled by 1000). *Left*: Morning, *Middle*: afternoon, *Right*: Night.

- See function `chimaps`

Chi-squared maps



Is there any interaction?

We denote the estimated K-functions for $\mathbf{x}_1, \ldots, \mathbf{x}_m$ by $\widehat{K}_1, \ldots, \widehat{K}_m$. An average version of K-functions is of the form

$$\overline{K}(r) = \frac{1}{m} \sum_{i=1}^{m} \widehat{K}_i(r), \qquad r \geq 0. \tag{7}$$

We look at minimum and maximum values of the estimated K-functions as

$$\widehat{K}^{\text{high}}(r) = \max_r [\widehat{K}_i(r)], \qquad \widehat{K}^{\text{low}}(r) = \min_r [\widehat{K}_i(r)], \qquad i = 1, \ldots, m.$$

- See function `Kinhom.Track`

K-function

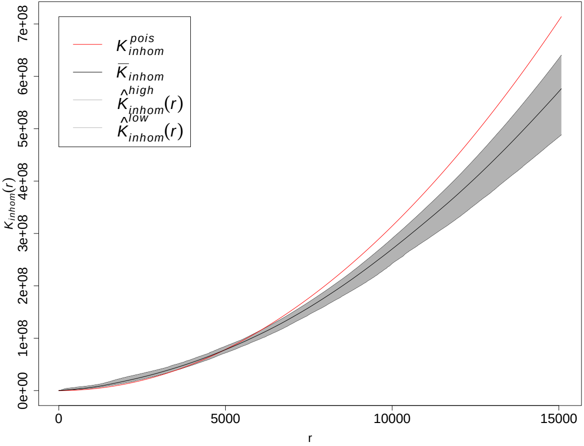


Figure: Variability area of K-function.

Summary

What R package `trajectories` provides:

- Classes and methods.
- Simulation techniques and model fitting.
- Distance analysis.
- Movement smoothing.
- Intensity estimation.
- Chi-squared maps.
- Second-order summary statistics.

References I

Baddeley, A., Rubak, E., and Turner, R. (2015). *Spatial Point Patterns: Methodology and Applications with R*. CRC Press.

Baddeley, A. and Turner, R. (2005). Spatstat: an R package for analyzing spatial point patterns. *Journal of Statistical Software*, 12(6):1–42.

Thanks for your attention

" ***Q&A*** "

Get in touch

- Email: m2.moradi@yahoo.com
- twitter: @m2_moradi
- github: @Moradii