Transport networks and road safety

Andrea Gilardi [1] Robin Lovelace [2]

[1]University of Milan - Bicocca

[2]University of Leeds - ITS

Who am I

- I'm Andrea Gilardi, a Ph.D. student in Statistics at the University of Milan - Bicocca.
- My main research interests lie in the field of spatial and spatiotemporal statistics, with a particular focus on point pattern events on a linear network. I really love coding in R.
- Now I'm in Leeds (I will leave next week unfortunately) working with Robin Lovelace on a review of existing packages for spatial network analysis in R (like `stplanr`) to improve my knowledge on several spatial topics and apply the methods to road safety research

Overview of the seminar

1. A

A few definitions

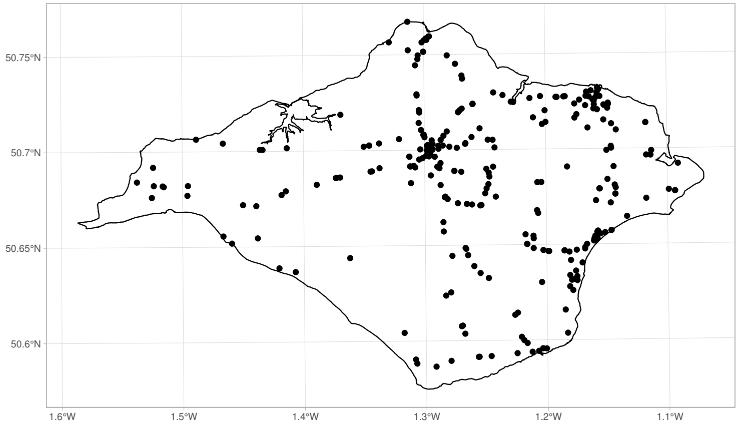
- Informally, a **Point Process** is a random mechanism whose outcomes are **Point Patterns**, i.e. a (finite) sequence of points in the space.
- Classical examples of point processes are: tree locations in a forest (the classic swedish pines data), animal nesting sites, ambulance interventions or, as in this seminar, car crashes.
- We will use these data to formalize the first steps we took towards the definition of a precise model that can be used to locate the most dangerous locations for car crashes (i.e. the black spots).

Car crashes data

- In the following part of this seminar, we will analyze data for car crashes that occurred in the Isle of Wight (UK) during 2018.
- We downloaded the data using the `stats19` package, which is a tool to help download, process and analyse the UK road collision data collected using the 'STATS19' form.
- These data are really rich and they include several additional information (like the severity of the crash, the weather, the light condition and several other markers) but, for the moment, we will focus only on the location of the events.

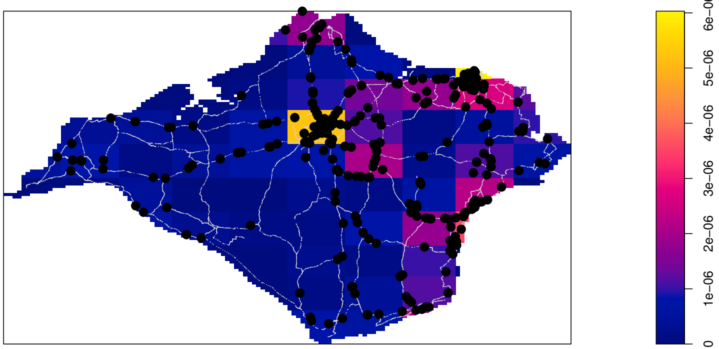
Car crashes data (cont)

This is a graphical representation of the car crashes occurred in the Isle of Wight (UK) during 2018. There are some clear patterns in the data that we need to take into account.



Point Processes on a Street Network

Car crashes represent a classical example of a point process occurring on a linear network and the usual statistical techniques (as the following quadratcount) are not valid.

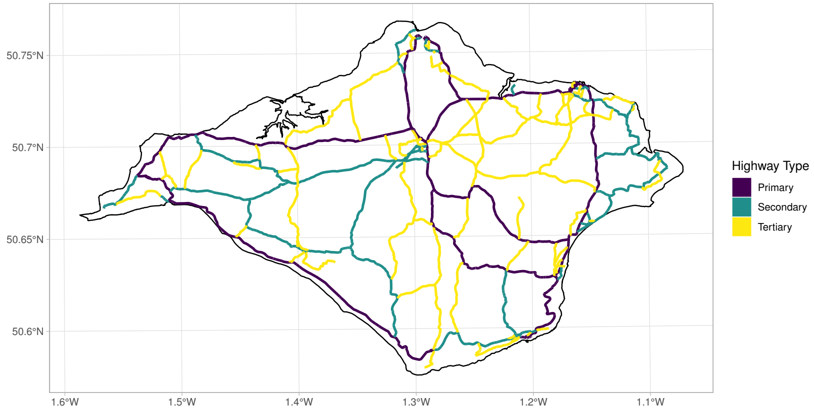


Street Networks

- The road network we use is built using OpenStreetMap data.
- OpenStreetMap is a project that aims at building a free and editable map of the World with an open-content license.
- The basic components of OpenStreetMap data are called *elements* and they consist of:
  - *nodes*: representing points on the earth surface;
  - *ways*: which is an ordered list of nodes;
  - *relations*: which is a list of nodes, ways and other relations. Each member has additional information that describe its relationship with the other elements. Roads, turn restrictions and administrative boundaries are usually described as relations.

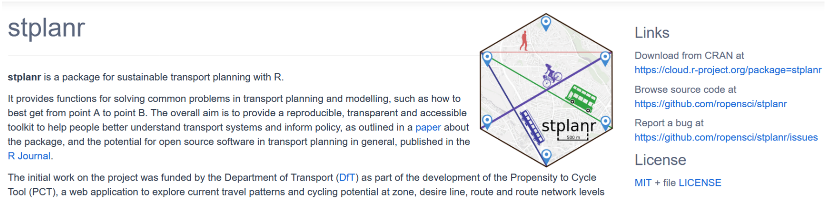
Street Network in the Isle of Wight

This is a graphical representation of the main roads in the Isle of Wight.



Representing street networks as graphs

- Typically roads are represented purely as geographic objects, but they are also graphs
- A few ways of representing spatial networks have been developed, including in the R package `stplanr`
- That is, roughly speaking, how Robin and I met



stplanr

stplanr is a package for sustainable transport planning with R.

It provides functions for solving common problems in transport planning and modelling, such as how to best get from point A to point B. The overall aim is to provide a reproducible, transparent and accessible toolkit to help people better understand transport systems and inform policy, as outlined in a paper about the package, and the potential for open source software in transport planning in general, published in the R Journal.

The initial work on the project was funded by the Department of Transport (DfT) as part of the development of the Propensity to Cycle Tool (PCT), a web application to explore current travel patterns and cycling potential at zone, desire line, route and route network levels

Links

Download from CRAN at
https://cloud.r-project.org/package=stplanr

Browse source code at
https://github.com/ropensci/stplanr

Report a bug at
https://github.com/ropensci/stplanr/issues

License

MIT + file LICENSE

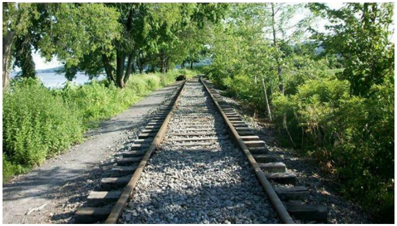
`stplanr` - **networks**

- Broadly speaking, let's say that a street network is a network whose nodes and edges are associated with geographical elements in the space.
- In the `stplanr` representation of a street network, the edges are the ways that were download from OSM while the vertexes are the starting and ending node of each way.
- This representation implies that two or more edges are *connected* if and only if they share one or more boundary point.

Problems...

Theories and definitions are fine but obviously the data we face in the wild world is quite different. We will discuss three probems: **roundabouts** (i.e. circular ways), **overpasses** (i.e. intersecting ways that are not really connected due to a vertical grade of separation) and (some) **street intersections**.

Theory

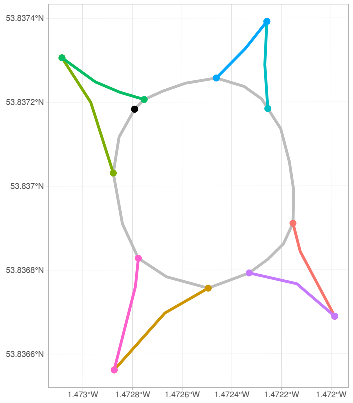


Real Data

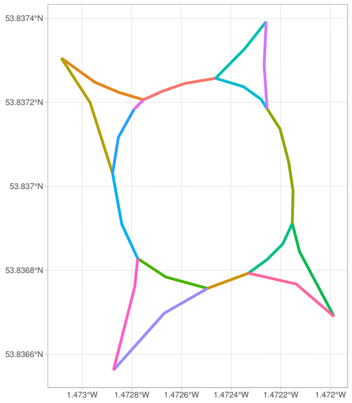
Roundabouts, i.e. circular ways

The roundabout on the left is unroutable by `stplanr`-definition of spatial network since the roundabout is not connected to the other edges.

Before



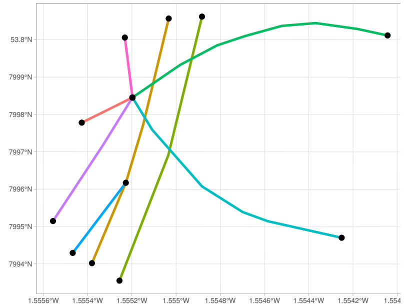
After



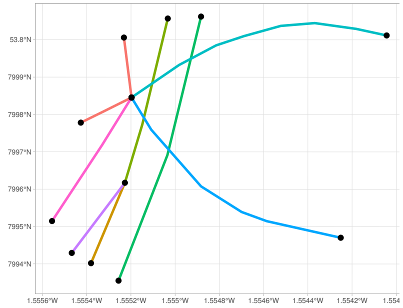
Bridges, overpasses and underpasses

Even if we break up a street network (unroutable on the left) we must be sure not to ruin overpasses and underpasses relations.

Before



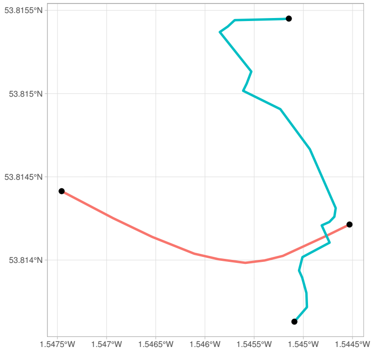
After



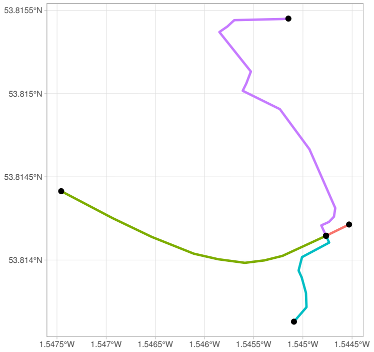
Streets intersections

There are also some cases where two streets intersects and they don't share any vertex.

Before

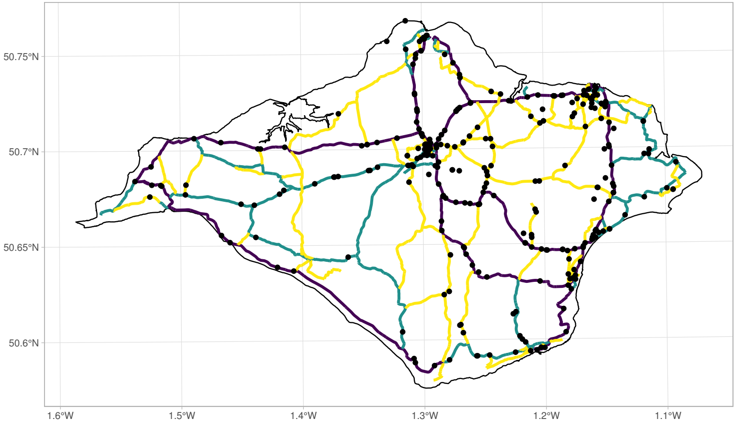


After



Fixing the street networks

We developed a new function (called `rnet_breakup_vertices` and stored in `stplanr` package) to fix all these problems and this is the result.



Modeling of crashes on street networks

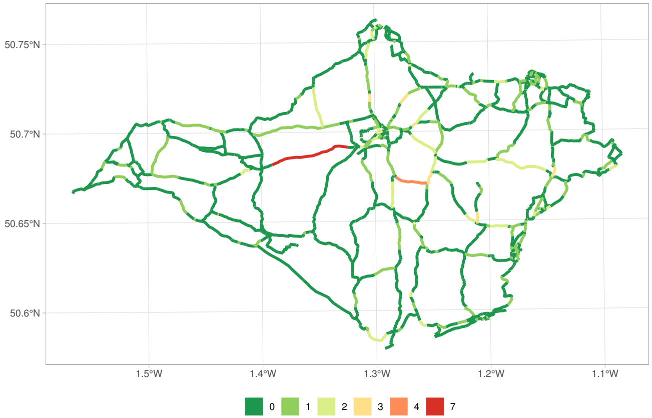
Summary of the models

Counting crashes on street networks

- Following the same ideas behind the quadratcounts on the plane (and some statistical theories) we wrote a few function to count the number of car crashes occurring on each edge of the street network.
- R can only represent exactly integers numbers and fractions whose denominator is a power of two (source), so none of the car crashes (whose coordinates are represented as double numbers with a 53 binary digits accuracy) lie exactly on the street network.
- For that reason we wrote a few R function to match each car crash with the nearest edge on the network and count the occurrences.

The first road risk measure

This is the result. You should note that we excluded all car crashes that were farther than 100m from the nearest network edge (33 crashes).

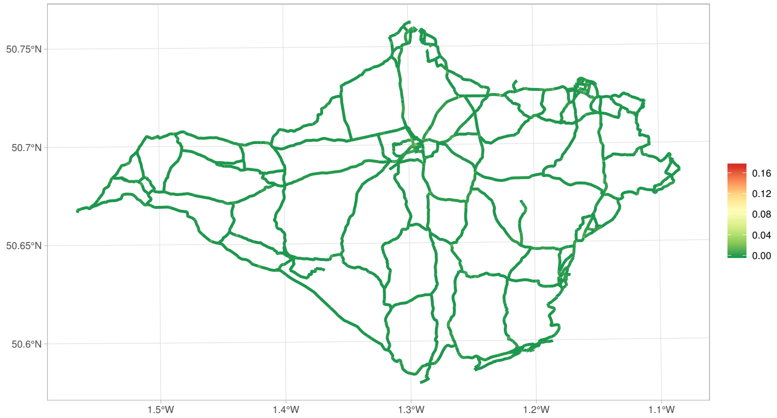


Problems with the raw counts measure

- There are some clear problems with the previous risk measure and, the most important one, is that we are comparing ways with very different length (i.e. different *exposure*).
- There are two possible solutions: 1) rebuild the network cutting and pasting ways in such a way that they all have approximately the same length; 2) estimate the number of car crashes per meter.
- For the moment we are working with the second solution since the other one is much more difficult to implement.

Car crashes per meter

This is the result and, then again, there are some obvious problems: there are a few car crashes occurring in very small road segments that artificially inflate this risk index.



Local smoothing on a linear network

- Let n be the number of (non overlapping) edges in the street network, $x_i,\ i = 1, \ldots, n$ the number of car crashes occurring in each edge and $l_i,\ i = 1 \ldots n$ the length of each way. The ratio

$$y_i = \frac{x_i}{l_i},\ i = 1, \ldots, n$$

  represent the number of car crashes per meter.

- Let $\delta_{i,p}$ represent the set of all neighbours of each street i up to order p. If $p = 0$ then $\delta_{i,p}$ includes only the i-th street segment; if $p = 1$ then $\delta_{i,1}$ includes the street segment i and its neighbours[1]; if $p = 2$ then $\delta_{i,2}$ includes the street segment, its neighbours and the neighbours of its neighbours and so on...

[1] In the `stplanr` representation of street networks, two ways are neighbours if they share one boundary point

Local smoothing on a linear network

- Now we can perform a local smoothing of y_i, i.e. the number of car crashes per meter, as

$$\tilde{y}_i = \frac{1}{m_i} \sum_{j \in \delta_{i,p}} y_j$$

  where m_i represent the number of elements included in $\delta_{i,p}$.

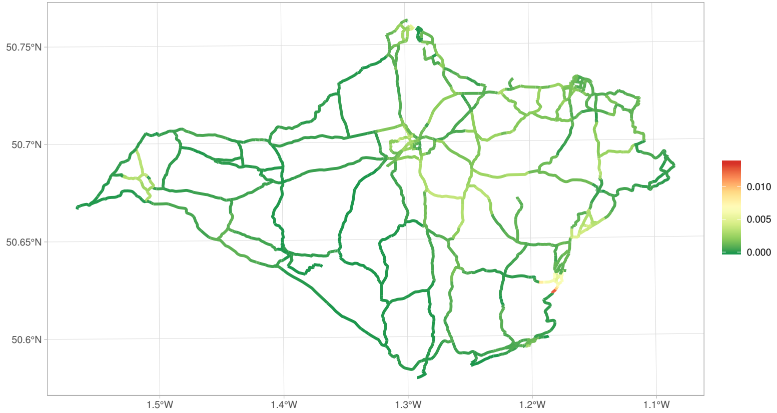
- The value of p is an input for the procedure and represent the degree of the smoothing: small values represent a local smoothing while higher values create a "global" smoothing. For example if we take $p = n$ then every segment is linked with the same value, i.e. $\tilde{y} = \frac{1}{n} \sum_{i=1}^{n} y_i \ \forall i = 1, \dots, n$.

- We'll see later a few methods that can be used to decide the value of p.

- The procedure was coded using the `igraph` package.

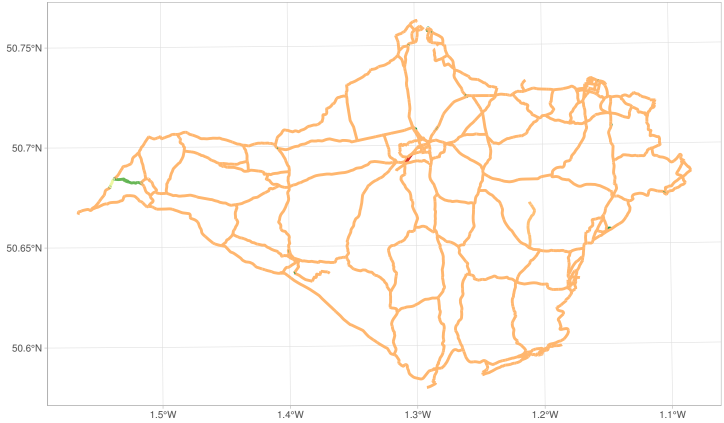
Smoothing on a linear network

This is the result with $p = 6$ and now it's possible to extract some information from the plot.



Why network cleaning is important?

This is the same procedure applied to the street network without the cleaning and with $p = n$. It's clear that there is something wrong.



Drawbacks

- Nevertheless, even if we work with a proper and clean network, the smoothing procedure has several drawbacks since it is nothing more than an exploratory data analysis (EDA) procedure.
- We did not define any precise statistical model which implies that we can't quantify any uncertainty measure on the risk indexes. Moreover, inference and testing procedures are not clearly defined.
- In the last part of this seminar I'm going to present a statistical approach for the estimation of risk index on a street network which is based on Emprical Bayes Estimators.

Setting up the terminology

- Let n be the number of (non overlapping) edges in the street network and x_i be the number of car crashes occurring in each edge. We assume that

  $$x_i \sim \text{Poisson}(l_i \theta_i)$$

  where l_i represents the exposure on each segment and θ_i is the risk measure of the ith street segment.

- There are several possible ways to measure the exposure of each segment and, for the moment, we are going to assume that l_i represents the length of the ith segment.

Setting up the terminology (cont)

- We can now define
$$y_i = \frac{x_i}{l_i}$$
which is just the ratio between the number of car crashes and the length of the edge and, under the previous assumptions, represent the maximum likelihood estimates of θ_i.

- It's possible to prove that $\text{var}[y_i] = \frac{\theta_i}{l_i}$, which implies that the lower is the exposure, the more inaccurate is the estimate y_i. Moreover, it's possible to prove that, taken together, $\{y_i,\ i = 1, \ldots, n\}$ are not necessarily the best[2] estimates of $\{\theta_i,\ i = 1, \ldots, n\}$.

[2]with a squared loss function

Classical and Bayesian statistics

- The classical statistical setting is based on the assumption that θ_i, the parameter that we want to estimate, is a fixed and unknown quantity and all the inferential procedures are based on the *repeated sampling principle* (i.e. we imagine it's possible to get other samples $\{x_i,\ i = 1, \ldots, n\}$ which are generated under the same random mechanism).

- Instead, in a bayesian framework, the sample is fixed while the parameters θ_i represent a random quantity that should be specified as a *random variable* with its *prior distribution*. We can include in the *prior distirbution* all the previous knowledge we have on θ_i, like its mean, median, modal quantity or range of variation. The statisticians then use the Bayes theorem to combine the

Setting up the terminology (cont)

- Now we are going to adopt a bayesian mindset specifying θ_i as a random variable with mean μ_i and variance σ_i^2, without any explicit functional form.

- So this is the framework

$$x_i|\theta_i \sim \text{Poisson}(l_i\theta_i) \text{ and } \theta_i \sim (\mu_i, \sigma_i^2).$$

- If we define y_i in the same manner as before, then $\mathbb{E}[y_i|\theta_i] = \theta_i$ and $\text{var}[y_i|\theta_i] = \frac{\theta_i}{l_i}$. By the law of iterated expected values

$$\mathbb{E}[y_i] = \mathbb{E}[\mathbb{E}[y_i|\theta_i]] = \mu_i$$

and

$$\text{var}[y_i] = \text{var}[\mathbb{E}[y_i|\theta_i]] + \mathbb{E}[\text{var}[y_i|\theta_i]] = \sigma_i^2 + \frac{\mu_i}{l_i} = \nu_i$$

Shrinkage Estimator

- It's possible to prove that the best linear Bayesian estimator for $\{\theta_i,\ i = 1, \dots, n\}$ is given by the shrinkage estimator:

$$\hat{\theta}_i = P_i y_i + (1 - P_i)\mu_i,\ i = 1, \dots, n$$

where $P_i = \frac{\sigma_i^2}{\nu_i}$.

- We can see that $\hat{\theta}_i$ is a weighted average of the prior mean, μ_i and y_i, the raw estimate for θ_i. The weights P_i are proportional to the exposure, which means that the grater is l_i, and the more important is y_i in the estimation of θ_i

An Empirical Bayes Approach

- There are two problems now. The first problem is that the number of parameters in the model is greater than the number of edges in the network (which makes the model unidentifiable). We can solve it by assuming that $\mu_i = \mu \; \forall i$ and $\sigma_i^2 = \sigma^2 \; \forall i$.

- The second problem is that we don't know μ and σ^2 but, using the James-Stein approach, we can estimate them using the marginal distribution of Y (since we proved that $\mathbb{E}[y_i] = \mu$ and $\text{var}[y_i] = \sigma^2$). This is called an Empirical Bayes Approach.

An Empirical Bayes Approach (cont)

- If we don't specify any functional form for the prior of θ, we can still use the methods of moment for the estimation of μ and σ^2.

- We will use

$$\tilde{\mu} = \frac{\sum_{i=1}^{n} l_i y_i}{\sum_{i=1}^{n} l_i},$$

$$\tilde{\sigma}^2 = \max\left\{0, S^2 - \frac{\tilde{\mu}}{\bar{l}}\right\},$$

and

$$\tilde{P}_i = \frac{\tilde{\sigma}^2}{\tilde{\sigma}^2 + \frac{\tilde{\mu}}{l_i}}$$

and the empirical version of the shrinkage estimator is

$$\hat{\theta}_i = \tilde{P}_i y_i + (1 - \tilde{P}_i)\tilde{\mu}$$

A spatial version of Empirical Bayes

- The problem now is that we are completely ignoring the spatial component of the model (i.e. if we randomly shuffle all the counts x_i the $\hat{\theta}_i$ doesn't change).

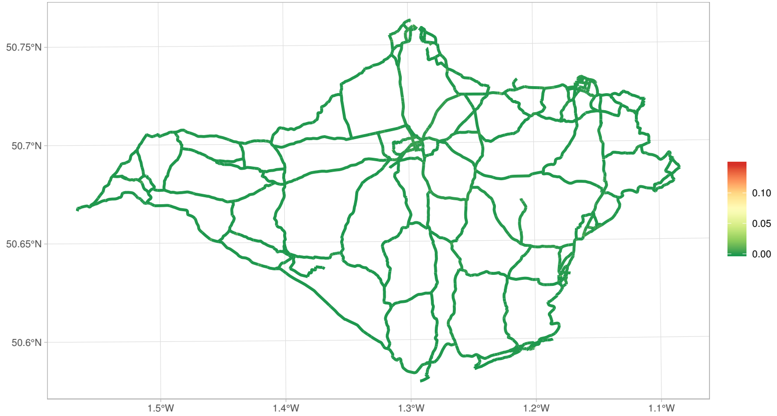
- We can fix that using the same approach as before, i.e. we can define a set $\delta_{i,p}$ which represent all the neghbours of the ith edge up to order p and modify all the previous formula just to take into account just the edges included in $\delta_{i,p}$. For example

$$\tilde{\mu}_{i,1} = \frac{\sum_{j \in \delta_{i,p}} y_j l_j}{\sum_{j \in \delta_{i,p}} l_j}$$

  is the empirical estimate of μ for the ith edge considering all neighbours up to order 1.

Problems . . .

We tried to apply that methodology to our data with several possible choices of p but it soon became clear that it didn't work (as you can see from the following figure with $p = 6$). But why?



The lengths of the OSM ways data

- The problem is that $\hat{\theta}_i$ is a convex combination between the number of car crashes per meter and a local average measure where the weights are defined by $P_i = \frac{\sigma_i^2}{\sigma_i^2 + \frac{\mu_i}{l_i}}$.

- In our case it occurred (by chance) that a few car crashes happened really close to the short edges in the network while all the other surrounding ways remained "empty". This means that $P_i \to 1$, $\hat{\theta}_i \to y_i$ and our methodology fails to solve that problem.

- We need an algorithm to merge the shortest z% of segments with its nearest and shortest neighbour. We choose $z = 15\%$.

Merging the shortest segments

We just started working on this topic, so please don't judge me . . .

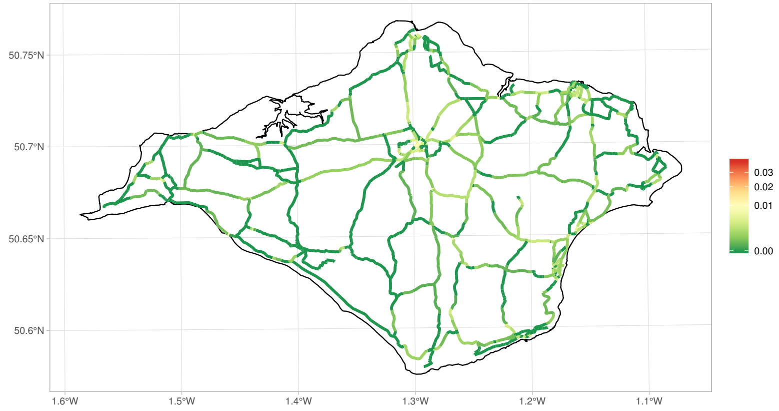
Merging the shortest segments

This is the pseudo-code:

```
1: Set minimal length threshold.
2: Extract the sfc from the sf object and sort it according to its length.
3: Estimate the neighborhood structure.
4: while Any segment is longer than threshold do
5:     for $i = 1$ to $length(sfc)$ do
6:         if $st\_length(sfc[i]) > threshold$ then
7:             Merge the small segment with its nearest and shortest neighbour;
8:             Remove the old segments and add the new one;
9:             Rebuild the neighborhood structure and sort the sfc;
10:            BREAK.
11:        end if
12:    end for
13: end while
```

The final result

This is the result with $p = 1$. I think that the methodology is really good but, to be honest, I used a square-root transformation of the empirical bayes estimates just to improve the graphical representation.



Future Work

- Redefine l_i
- Consider the risk of the estimation of l_i

TODO

- Add citations and fix bibliography
- Add credits to memes, OSM and beamer theme
- highlight important word (i.e. ABC)
- RSP meme with hyperref