

GET: Hotspot detection on a linear network

Mari Myllymäki

Natural Resources Institute Finland (Luke)

Tomáš Mrkvička

University of South Bohemia

Stanislav Kraft

University of South Bohemia

Vojtěch Blažek

University of South Bohemia

Michal Konopa

University of South Bohemia

Abstract

This vignette describes and shows how the methodology proposed by [Mrkvička, Kraft, Blažek, and Myllymäki \(2023\)](#) for detecting hotspots on a linear network can be performed using the R package **GET** ([Myllymäki and Mrkvička 2024](#)).

Keywords: false discovery rate, hotspot, linear network, Monte Carlo test, road accidents, R, spatial point pattern.

1. Practical description of hotspots computation

The first step involves import data to R. The crashes are recorded as a point pattern, thus x and y coordinates together with window range must be provided. The same holds for crossroads that forms the vertices of the linear network. The edges of the linear network are provided in the form of matrix, where first column corresponds to the order of the crossroad where the edge starts and the second column corresponds to the order of the ending crossroad. Further, the covariate can be imported to R from tiff or raster format.

When all files are prepared, the analysis can move to R with use of **spatstat**, **GET** and **parallel** packages. This vignette provides an example of hotspots detection that can be easily customized.

1.1. Estimating Poisson point pattern

The function `pois.lppm()`, can be used to estimate the inhomogeneous Poisson point process model on linear network. This function provides the `firstordermodel`, i.e. the regression model of dependence of crashes on the spatial covariates, `EIP`, i.e. estimated inhomogeneous intensity from the data and `secondorder`, i.e. estimation of the inhomogeneous K -function. The plot of the `secondorder` provides diagnostics, if the model is adequate for the data. If the estimated K -function lies close to the theoretical line, the data does not report any clustering, and the function `hotspots.poislpp()` can be used for final hotspots detection. If the estimated K -function does not lie close to the theoretical line, and it is above, the data report clustering, and the a clustered point pattern model must be fitted to the data and hotspots detected using this clustered model instead. The important input parameters to be specified for the function `hotspots.poislpp()` are `PP`, i.e., the point pattern used for estimation, `formula`, i.e., the linear regression formula specified as usually in R having `PP` on the right hand side of the formula (i.e., as the response variable), `data`, i.e., the object from which the formula takes the data.

1.2. Estimating Matérn cluster point pattern

The function `MatClust.lppm()`, can be used to estimate the Matérn cluster point pattern with inhomogeneous cluster centers on linear network. This function provides the same outputs as the `pois.lppm()` and further estimated parameters α and R . The `secondorder` provides again the diagnostics for checking if the clustered model is appropriate. The sample K -function must be close to the K -function of the estimated model (green line). If it is not the case the searching grid for parameters α and R that is input in the function must be manipulated to get the a closer result. If the estimated model is adequate one can proceed to the hotspot detection with the use of the function `hotspots.MatClustlpp()`. Remark here, that for the estimation of the second order structure a smaller data can be used than for the estimation of the first order structure in order to save the computation time, since the second order is a local characteristics. Thus the input to this function can contain, in addition to the full data in `PP` that is used for first order estimation, a subwindow `subwin` to specify a smaller part of the full data for second order estimation. Furthermore, `valpha`, i.e., vector of proposed alphas which should be considered in the optimization, `vr`, i.e., vector of proposed R s which should be considered in the optimization must be provided. The user can also specify how many cores should be used in the computation by parameter `ncores`.

1.3. Hotspot detection under the Poisson assumption

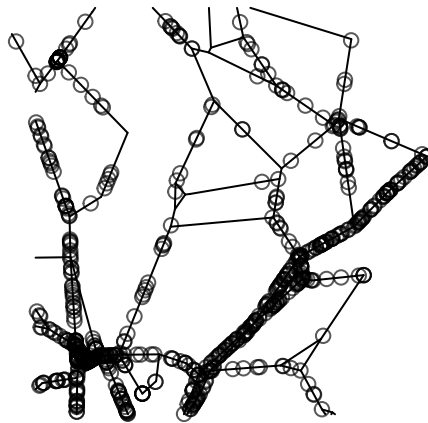
If the Poisson assumption is checked, the hotspots can be detected using the function `hotspots.poislpp()`. The plot of results contains the locations of determined hotspots together with their sizes. A parameter `sigma` must be provided in this function. It determines the bandwidth of the kernel used in the inhomogeneous intensity estimation. This parameter should be carefully selected with respect to the size of the window. It represents how much smoothing is applied on the intensity, it is too big, the inhomogeneity will be blurred away. If it is too small, the intensity will react on every event and the inhomogeneity will be too crazy. The `nsim` parameter specifies the number of simulations to perform the envelope. It should be as large as possible. Usually, the default of 10000 is fine. The argument `ncores` can be used to specify how many cores should be used for the computation.

A part of the analysis, as will be described below, uses a subset of the `roadcrash` data, because the computations of inhomogeneous K -function and density can be rather computational.

Here we define the subwindow and plot the pattern living in the subwindow.

```
R> subwin <- owin(c(-760000, -740000), c(-1160000, -1140000))
R> plot(PPfull[, subwin], main="Road crashes: subpattern")
```

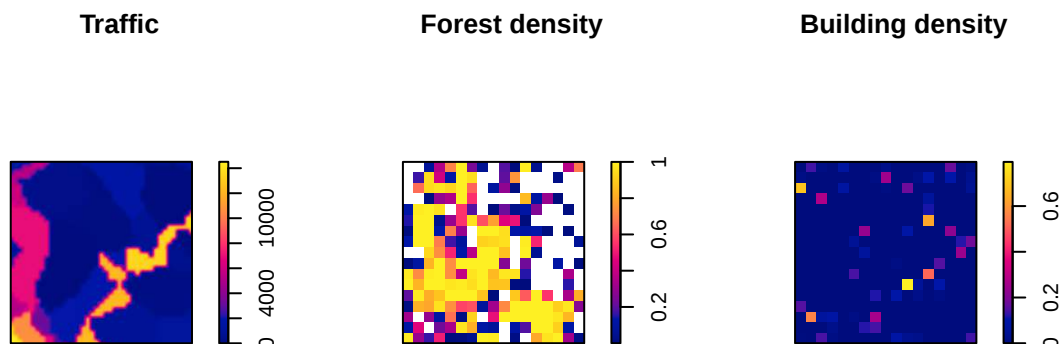
Road crashes: subpattern



[Mrkvička *et al.* \(2023\)](#) had a total of 9 spatially defined covariates. In our example here and available in `roadcrash` in **GET** are three covariates, namely average traffic volume (number of vehicles per 24 hours), forest density and building density in the cell.

The following plots show these covariates in the subwindow defined above.

```
R> par(mfrow=c(1,3))
R> plot(roadcrash$Traffic[subwin], main="Traffic")
R> plot(roadcrash$ForestDensity[subwin], main="Forest density")
R> plot(roadcrash$BuildingDensity[subwin], main="Building density")
```



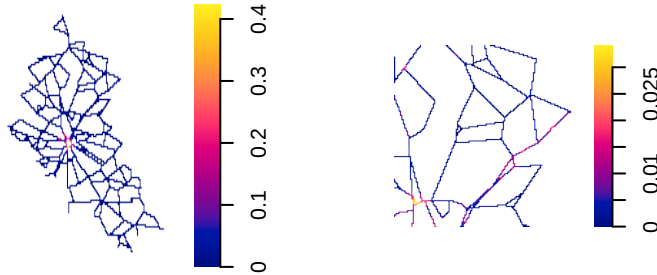
4. Non-parametric intensity estimate

A non-parametric density estimate of the point pattern on a linear network can be obtained using the function `density.lpp()` of the **spatstat** package.

A parameter `sigma` must be provided in this function. It is the same parameter `sigma` that was already discussed above in Section 1.3, i.e., it determines the bandwidth of the kernel used in the inhomogeneous intensity estimation. The argument `distance` specifies what type of kernel to use in the linear network. In our hotspot detection, we use here a two-dimension kernel specified by `distance="euclidean"` because computation of the density with this kernel is relatively fast. We set the smoothing bandwidth `sigma` that small that two roads are very unlikely closer than two times `sigma` apart from each other. Thus, the intensity estimate at a certain location on the linear network is computed merely from the crashes at that location.

```
R> densi <- density.lpp(PPfull, sigma = 250, distance="euclidean")
R> densi2 <- density.lpp(PPfull[, subwin], sigma = 250, distance="euclidean")
R> par(mfrow=c(1,3))
R> plot(densi, main="Intensity of crashes: full window")
R> plot(densi2, main="Intensity of crashes: subwindow")
```

Intensity of crashes: full wind | Intensity of crashes: subwind



5. Fitting the inhomogeneous Poisson process

The simplest point process model for road crashes is the (inhomogeneous) Poisson process with intensity

$$\rho_{\beta}(u) = \kappa \exp(z(u)\beta^T), \quad u \in L, \quad (1)$$

where L is a linear network, $z = (z_1, \dots, z_k)$ is a vector of covariates and $\beta = (\beta_1, \dots, \beta_k)$ is a regression parameter. This process can be fitted using the **spatstat** package. We fit the model using the full **roadcrash** data.

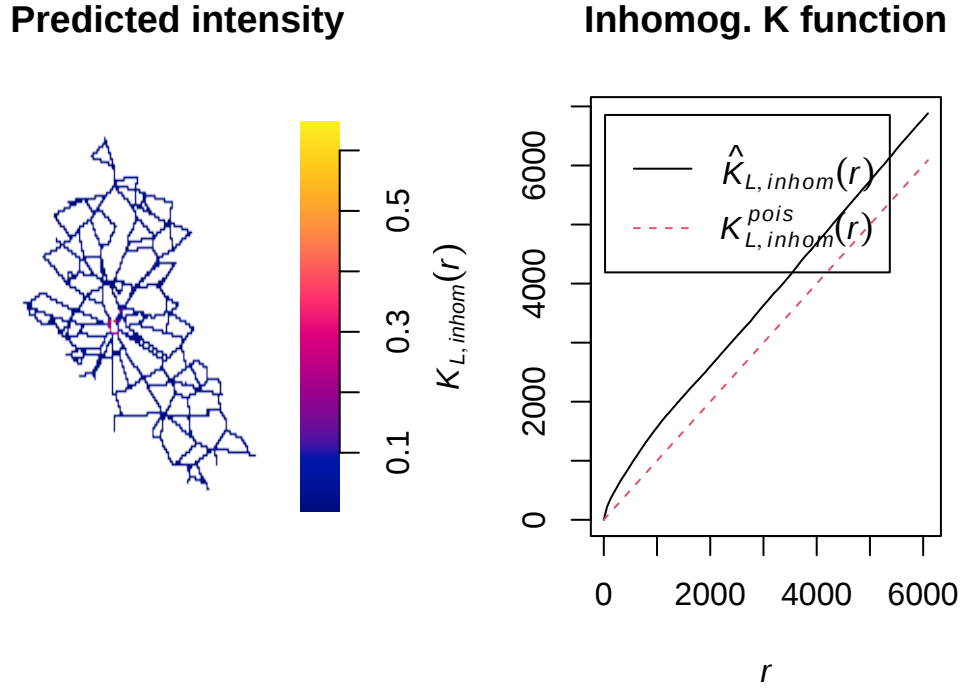
The function `pois.lppm()` both fits the model for the intensity as well as provides predicted point process intensity and the inhomogeneous K -function estimated from a given point pattern `PP` using the estimated intensity surface. (This is rather fast, taking a bit more than 10 seconds on a normal laptop; `system.time` is used to take the time only.)

```
R> myformula <- PP ~ Traffic + ForestDensity + BuildingDensity
R> system.time(
  Poi <- pois.lppm(PP=PPfull, formula=myformula, data=roadcrash)
)

   user  system elapsed 
12.83   0.05   12.92
```

Both the predicted point process intensity and the inhomogeneous K -function with theoretical Poisson K function can be plotted:

```
R> par(mfrow=c(1,2))
R> plot(Poi$EIP, main="Predicted intensity")
R> plot(Poi$secondorder, main="Inhomog. K function")
```



Here the inhomogeneous K -function estimated from the data lies above the theoretical line for the Poisson process and suggests clustering of points.

6. Fitting the Matern cluster process on a linear network

Mrkvička *et al.* (2023) considered instead of the Poisson process the Matern cluster point process with inhomogeneous cluster centers. This process is more suitable for clustered data. It can be estimated in two steps according to its construction following Mrkvička, Muška, and Kubečka (2014). In first step, the first order intensity function is estimated through Poisson likelihood. This was done above, i.e., the object EIP contains the estimated intensity. In second step, the second order interaction parameters α (mean number of points in a cluster) and R (cluster radius) are estimated through minimum contrast method. Unfortunately, working with cluster processes on linear networks is rather consuming and therefore they are currently not covered by the **spatstat** package. Thus, we have used the inhomogeneous K -function and the minimum contrast and grid search methods to find the optimal parameters. We implemented functions for simulating the Matern cluster process on a linear network LL with pre-specified centers (function `rMatClustlpp`) and for fitting the Matern cluster process on a point pattern on a linear network (function `MatClust.lppm`).

In the procedure, we estimate the parameters R and α of the Matern cluster process using the inhomogeneous K -function (with the estimated Poisson process intensity). We consider

a range of possible values of the parameters R and α . For each value of R and α , we compute the difference of the observed K -function from the "theoretical" K -function of the model, computed from the average of `nsim` (by default 10) simulation from the model. Simulations are used, because the theoretical K -function is not known for the Matern cluster process on the linear network. We then find out which of these possible values of parameters R and α lead to the smallest difference between the observed and "theoretical" K -functions.

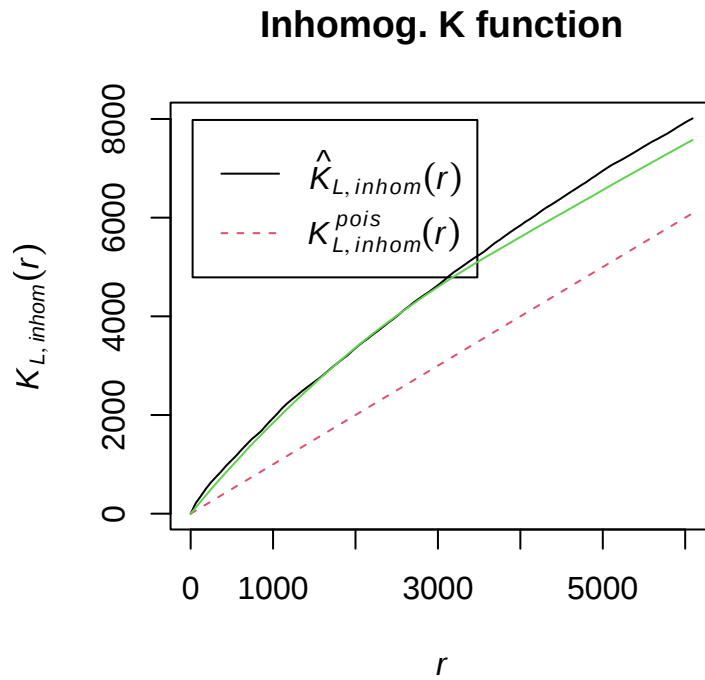
Remark here that the first order structure is estimated below from the full pattern `PPfull` (provided to the `MatClust.lppm()` function in the argument `PP`), whereas the second order structure is estimated from the pattern observed in the subwindow that is provided in the argument `subwin`. The second order structure has a limited range; therefore, estimating it only from the subpattern is useful for time saving.

```
R> valpha <- seq(5, 30, by=5)
R> vR <- seq(250, 2500, by=500)
R> myformula <- PP ~ Traffic + ForestDensity + BuildingDensity
R> system.time( # Took about 1,2 minutes on a laptop
  MatCl <- MatClust.lppm(PP=PPfull, formula=myformula, subwin=subwin,
    valpha=valpha, vR=vR, data=roadcrash, ncores = 1)
)
```

```
user  system elapsed
65.30   1.67   67.21
```

These results can be viewed, by plotting the observed K (solid line) and theoretical Poisson line (dashed line) and adding the K -function of the estimated Matern cluster process estimated from `nsim` (here 10) simulations.

```
R> # The observed K, and theoretical Poisson line
R> plot(MatCl$secondorder, main="Inhomog. K function")
R> # The Matern Cluster process K from nsim (here 10) simulations
R> # with chosen values of alpha and R
R> lines(x=MatCl$secondorder$r, y=MatCl$MCsecondorder, col=3)
```



The chosen parameter values are given in `alpha` and `R`.

```
R> MatCl$alpha
```

```
[1] 30
```

```
R> MatCl$R
```

```
[1] 1750
```

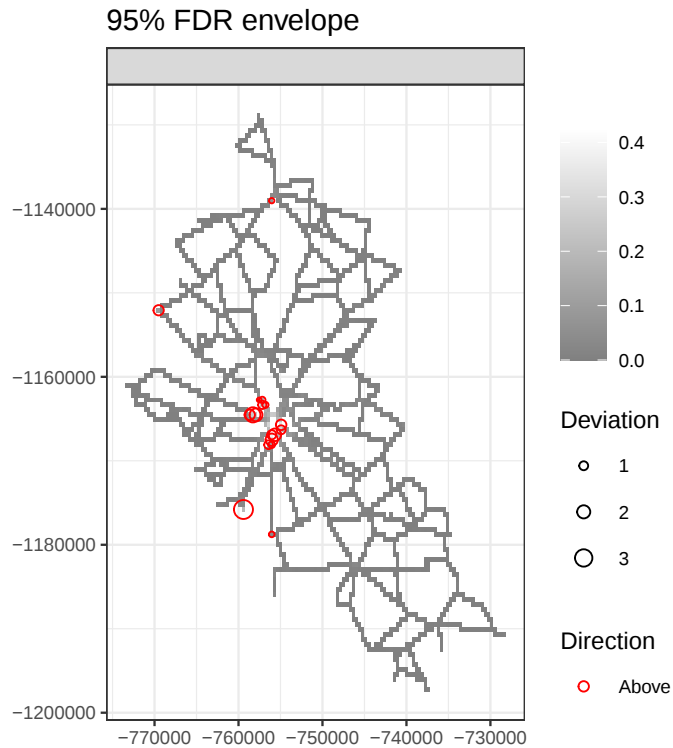
7. False discovery rate envelopes

To find the hotspots of road crashes that are not explained by the covariates, we first generate `nsim` simulations from the fitted Matérn cluster process and estimate the intensity for each of the simulated patterns. We note that we estimate the intensity here similarly as above for the observed pattern. This computation takes a bit of time (using 4 cores on a normal laptop took about 25-30 minutes).

```
R> nsim <- 10000
R> system.time(
  res <- hotspots.MatClustlpp(PP=PPfull, formula=myformula,
                             R=MatCl$R, alpha=MatCl$alpha,
                             data = roadcrash, sigma=250, nsim=nsim, ncores=4)
)
```

The FDR envelope (Mrkvička and Myllymäki 2023) is computed within `hotspots.MatClustlpp()` using the function `fdr_envelope()` of the **GET** package. Because we are only interested in locations where the intensity is higher than expected, the test is done alternative to "greater".

```
R> plot(res) + scale_radius(range = 0.5 * c(1, 6))
```



The size of the cluster is indicated by the circles. The circle radius is proportional to the size of the deviation of the observed intensity from the upper bound of the FDR envelope divided by the difference of the upper FDR envelope and the centre of the envelope. Thus, the size is a measure of relative exceedance.

Acknowledgements

We thank Mikko Kuronen for helping to make the code faster.

References

- Mrkvička T, Kraft S, Blažek V, Myllymäki M (2023). “Hotspot Detection on a Linear Network in the Presence of Covariates: A Case Study on Road Crash Data.” [doi:http://dx.doi.org/10.2139/ssrn.4627591](https://doi.org/10.2139/ssrn.4627591).
- Mrkvička T, Muška M, Kubečka J (2014). “Two Step Estimation for Neyman-Scott Point Process with Inhomogeneous Cluster Centers.” *Statistics and Computing*, **24**(1), 91–100. [doi:10.1007/s11222-012-9355-3](https://doi.org/10.1007/s11222-012-9355-3).

Mrkvička T, Myllymäki M (2023). “False Discovery Rate Envelopes.” *Statistics and Computing*, **33**, 109. doi:10.1007/s11222-023-10275-7.

Myllymäki M, Mrkvička T (2024). “**GET**: Global Envelopes in R.” *Journal of Statistical Software*, **111**(3), 1–40. doi:10.18637/jss.v111.i03.

Affiliation:

Mari Myllymäki

Natural Resources Institute Finland (Luke)

Latokartanonkaari 9

FI-00790 Helsinki, Finland

E-mail: mari.myllymaki@luke.fi

URL: <https://www.luke.fi/en/experts/mari-myllymaki>

and

Tomáš Mrkvička

Faculty of Agriculture and Technology

University of South Bohemia,

Studentská 1668

37005 České Budějovice, Czech Republic

E-mail: mrkvicka.toma@gmail.com

URL: https://www.jcu.cz/en/university/staff/person?identita=Mrkvicka_Tomas_6571