INTENSITY ESTIMATION ON GEOMETRIC NETWORKS WITH PENALIZED SPLINES *

BY MARC SCHNEBLE¹ AND GÖRAN KAUERMANN²

¹Department of Statistics, Ludwig-Maximilians Universität Munich, marc.schneble@stat.uni-muenchen.de
²Department of Statistics, Ludwig-Maximilians Universität Munich, goeran.kauermann@stat.uni-muenchen.de

In the past decades, the growing amount of network data lead to many novel statistical models. In this paper, we consider so-called geometric networks. Typical examples are road networks or other infrastructure networks. Nevertheless, the neurons or the blood vessels in a human body can also be interpreted as a geometric network embedded in a three-dimensional space. A network-specific metric rather than the Euclidean metric is usually used in all these applications, making the analyses of network data challenging. We consider network-based point processes, and our task is to estimate the intensity (or density) of the process, which allows us to detect high- and low-intensity regions of the underlying stochastic processes. Available routines that tackle this problem are commonly based on kernel smoothing methods. This paper uses penalized spline smoothing and extends this towards smooth intensity estimation on geometric networks. Furthermore, our approach easily allows incorporating covariates, enabling us to respect the network geometry in a regression model framework. Several data examples and a simulation study show that penalized spline-based intensity estimation on geometric networks is a numerically stable and efficient tool. Furthermore, it also allows estimating linear and smooth covariate effects, distinguishing our approach from already existing methodologies.

1. Introduction. In statistical network analysis, a (static) network is usually considered as a graph that is characterized by a set of vertices which are connected by a set of, possibly weighted, edges (Kolaczyk and Csárdi, 2014). In this matter, the interest usually lies in the mutual relationship and the dependencies of the vertices, sometimes called “actors” (Snijders, 1996), that are induced by the edges. For a general overview of this research area, see e.g. Goldenberg et al. (2010). In the context of this paper, a network is rather considered as a geometric object embedded in a Euclidean space. We use the term “geometric network” and a typical example is a network of streets. The setting is that we observe a spatial point process on the network edges and focus is on estimating the intensity (or density) of this process.

Regarding the data structure, the question arises why one should analyze data points on a geometric network and not in the Euclidean space itself. To illustrate this, consider a point pattern that seems to be clustered in the plane. However, the points might be uniformly distributed on a network where many network segments are clustered within a small area. A typical example is the distribution of traffic accidents in an urban area (McSwiggan, Baddeley and Nair, 2017). Theoretically, such events can only occur on a network of streets which is often considered being embedded in the plane. Therefore, the statistical analyses of point patterns distributed across a Euclidean space and point patterns distributed only on a geometric network are tremendously different.

*We would like to thank the elite graduate program Data Science at LMU Munich and the Munich Center for Machine Learning (MCML) for funding.

Keywords and phrases: Intensity estimation of stochastic point processes, generalized additive models, geometric networks, penalized splines, poisson regression with offset, spatstat package
Due to the increasing availability of network-based data, the last 25 years have seen a broad range of literature concerned with network-based point processes. Amongst the first statistical analyses of spatial point patterns on a network were proposed by Okabe, Yomono and Kitamura (1995), Okabe and Yamada (2001) and Spooner et al. (2004). They all noted that in the context of geometric network data, the Euclidean distance needs to be replaced by the shortest path distance to respect the network geometry. This resulted e.g. in the geometrically corrected network K-function (Ang, Baddeley and Nair, 2012), a modified version of Ripley’s K-function in two dimensions. The network K-function can be used to analyze the correlation structure of point patterns on a network. Baddeley, Rubak and Turner (2015) discuss the topic in general and especially from an application point of view. Among other contributions, a huge library of functions is provided to create, manipulate and analyze both a point pattern on a linear network, embedded in the plane, and the network itself. Furthermore, Baddeley, Rubak and Turner (2015) also treat marked point processes, intensities of point processes depending on covariates and point processes on trees which are networks without loops. Marked point processes on directed linear networks are further discussed by Rasmussen and Christensen (2020) for various kinds of stochastic point processes. Finally, we would like to highlight Baddeley et al. (2020) who provide a broad overview on how to analyze point patterns on linear networks.

The focus of this paper is on intensity estimation in geometric networks. Borruso (2008) and Xie and Yan (2008) developed kernel density estimation on a network geometry which is performed by respecting the shortest path distance. However, both articles did not consider that around vertices with more than two adjoining segments, there is more network mass within a certain shortest path distance. Hence, this approach leads to biased estimates, especially if the point pattern is distributed according to a uniform distribution on the network. Okabe, Satoh and Sugihara (2009) solved this problem by introducing equal-split (dis-)continuous kernel density estimation. The idea is to split the mass of the kernel functions equally across all other segments that depart from a vertex when approaching this vertex from one side. The approach was refined in McSwiggan, Baddeley and Nair (2017). Instead of a finite sum of paths over the network, they consider an infinite sum leading to a diffusion estimate that can be computed via a heat equation on the network. Furthermore, Moradi, Rodríguez-Cortés and Mateu (2018) showed in their application that an extension of Diggle’s (Diggle, 1985) non-parametric edge-corrected kernel-based intensity estimator is superior to the equal-split discontinuous estimator that was proposed by Okabe, Satoh and Sugihara (2009). Most recently, Rakshit et al. (2019) proposed to perform kernel smoothing on networks making use of a two-dimensional kernel that is still robust against errors in network geometry. This approach is especially well suited in scenarios of vast networks.

All the models discussed so far are kernel-based and produce continuous intensity estimates. However, there are various non kernel-based approaches where the fitted intensity is not continuous. To begin with the special case of a river network, which results as a directed and acyclic geometric network, O’Donnell et al. (2014) used penalized piecewise constant functions for estimating the river flow, which can be interpreted as penalized splines (P-splines, Eilers and Marx, 1996) of order 0. The paper was reviewed by Rushworth et al. (2015) who implemented the theory in the R package smnet. Fused density estimation was proposed by Bassett and Sharpnack (2019) to estimate the density on a geometric network. This estimator is the solution to a total variation regularized maximum-likelihood problem. Another very recent method that does also not result in continuous estimates is the smoothed Voronoi estimate (Moradi et al., 2019). Therefore, the Voronoi estimate (Barr and Schoenberg, 2010) is computed several times while only retaining a fraction $f$ of the data points in each iteration. The smoothed Voronoi estimate is then equal to the average over the re-scaled intensities.
When referring again to kernel-based methods, Eilers and Marx (1996) argued that kernel density estimators of points on the real line suffer from boundary effects, e.g. if the domain of the data is not specified correctly. Instead, the authors proposed to estimate the density by making use of penalized splines in order to smooth the histogram that is created by binning the data with small bin widths. With time, this concept has been extended, among others, to allow for density estimation of multiple dimensional data (Currie, Durban and Eilers, 2006) or to represent the density as a mixture of weighted penalized spline densities (Schellhase and Kauermann, 2012). A comprehensive survey of penalized spline theory and its application is given in Eilers, Marx and Durbán (2015). Generally, penalized spline estimation has become a major workhorse in statistical modeling as demonstrated in Ruppert, Wand and Carroll (2003, 2009).

In this paper, we extend the penalized spline-based intensity estimation approach of Eilers and Marx (1996) to work on geometric networks. The focus is to estimate the intensity (or density) of a point process on the network, given realizations of this process as data. In contrast to the intensity estimation methods summarized above, our approach allows us to estimate the smooth baseline intensity and the effects of network internal and external covariates. The embedding in the context of generalized additive models further enables us to assess the uncertainty of both the network intensity and covariate effects. Besides this manuscript our major contribution is the implementation\(^1\) of our methodology in R (R Core Team, 2013, version 4.1.0) for linear networks, where a lot of the functionality is based on the family of spatstat packages (Baddeley, Rubak and Turner, 2015, version 2.2-0).

The remainder of this paper is structured as follows. Section 2 introduces some basic notation related to network graphs, geometric networks and stochastic point processes on networks. Section 3 treats our new methodology to estimate the intensity of a point process on a geometric network with penalized splines. This is followed by Section 4, the presentation of the networks and the data which we employ in this paper. Sections 5 - 7 cover applications to real data while Section 8 explores the performance of our model when fitted to simulated data. Section 9 concludes the paper and discusses possible supplements of our model.

2. Notation and Problem. Consider a set \( V = \{v_1, \ldots, v_W\} \) of \( W \in \mathbb{N} \) elements which we call vertices, where \( \mathbb{N} = \{1, 2, 3, \ldots\} \) denotes the set of positive integers. Further, let \( E = \{e_1, \ldots, e_M\} \subset V \times V \) be a set of \( M \in \mathbb{N} \) pairs \( e_m = (v_i, v_j) \) which we call edges. Putting these together leads to the network graph \( L = (V, E) \) and we denote \( L \) as the graph representation of the network defined by a set of vertices \( V \) and a set of edges \( E \). In this paper, we only consider undirected networks, i.e. there is an edge from \( v_i \) to \( v_j \) if and only if there is an edge from \( v_j \) to \( v_i \). An edge \( e \) is called incident to a vertex \( v \) if there is another vertex \( v_i \in V \) such that \( e = (v, v_i) \in E \). The degree of a vertex \( v \), denoted by \( \text{deg}(v) \) is defined as the count of edges which are incident to \( v \) and for our purpose we always remove a vertex \( v \) from \( V \) if \( \text{deg}(v) = 0 \).

A geometric network also typically exhibits a geometric representation as a subset of a Euclidean space \( \mathbb{R}^q \) for \( q \geq 2 \). In this case, the set of vertices \( V \) in the network graph representation can be viewed as a set \( V = \{v_1, \ldots, v_W\} \) of vectors with \( v_i \in \mathbb{R}^q \) for \( i = 1, \ldots, W \). Consequently, the edges can be viewed as a set \( E = \{e_1, \ldots, e_M\} \) of network segments with each \( e_m \subset \mathbb{R}^q \) being the connection between two vertices \( v_i \) and \( v_j \). More generally, such an edge can be described by the image set \( e_m = \nu_m([a_m, b_m]) \) of a parametric curve (Heuser, 2006) \( \nu_m : [a_m, b_m] \to \mathbb{R}^q \) with \( a_m < b_m, \nu_m(a_m) = v_i \) and \( \nu_m(b_m) = v_j \), where the length

\(^1\)The R code that can be used to reproduce the results in this paper can be downloaded from https://github.com/MarcSchneble/NetworkSplines.
of the curve segment $e_m$ is given by

$$d_m = |e_m| = \lim_{N \to \infty} \sum_{i=1}^{N} \|\nu_m(t_i) - \nu_m(t_{i-1})\|_q,$$

with $t_i = a_m + i(b_m - a_m)/N$ for $i = 1, \ldots, N$ and $\| \cdot \|_q$ denotes the Euclidean distance in $\mathbb{R}^q$. Thus, (1) already suggest to approximate a parametric curve by a number of $N$ straight line segments with endpoints $\nu_m(t_0), \ldots, \nu_m(t_N)$. This representation is used in the R package spatstat (Baddeley, Rubak and Turner, 2015), which allows to analyze linear networks in the plane.

We can now define the geometric representation of a network graph $L$ as $L = \bigcup_{m=1}^{M} e_m \subset \mathbb{R}^q$. Hence, there is a one-to-one correspondence between $L$ and $\mathcal{L}$ which we exploit consistently in this paper. According to the network graph representation, we also define a vertex degree for the geometric network representation, meaning that $\text{deg}(v)$ denotes the count of segments which have an endpoint equal to $v$. Furthermore, the lengths $d_m$ of the curves $e_m$ from (1) imply a metric $d_L : L \times L \to [0, \infty)$ on $L$. More precisely, $d_L(z_1, z_2)$ denotes the shortest path distance between two points $z_1, z_2$ on $L$ and with $[z_1; z_2] \subset L$ or with $[z_1; z_2] \subset \mathcal{L}$ we denote the corresponding path, where a round bracket indicates that an endpoint is not contained in the set. The total length of the geometric network is $|L| = \sum_{m=1}^{M} d_m$. If the network is not connected, i.e. the corresponding network graph $L$ consists of more than one connected component, we can use the extended metric (Beer, 2013) $d_L : L \times L \to [0, \infty) \cup \infty$. In this case, the same methodology can be applied unmodified.

Also note that the geometric representation $L$ is not necessarily unique which can be seen from the following consideration: Let $v$ be a vertex (in network graph representation) with exactly two incident edges $e_m = (v, v_i)$ and $e_n = (v, v_j)$, i.e. $\text{deg}(v) = 2$ and $(v_i, v_j) \notin E$. If we remove $v$ from $V$ as well as $e_m, e_n$ from $E$ but add the edge $e = (v_i, v_j)$ to $E$, the network graph representation of $L = (V, E)$ has changed. In the geometric representation, we can remove $v$ from $V$ as well as $e_m, e_n$ from $E$ and add the segment $e = e_m \cup e_n$ to $E$ which does not change $L$. To exemplify the notation introduced in this section, Figure 1 shows a small network as a network graph (left panel) and as a geometric network embedded in the plane (right panel). The figure also visualizes that the role of a vertex $v$ in the geometric

**Figure 1.** Two different representations of a network: Left panel: Network graph representation $L$. Right Panel: Geometric network representation $L$. 
network representation is merely being the endpoint of \( \deg(v) \) curves. Hence, in this network the vertex \( v_2 \) could be removed from \( L \) without changing its geometric representation.

We now consider the following setting, see also McSwiggan, Baddeley and Nair (2017). Let \( X \) be a stochastic point process on the geometric network \( L \) with continuous intensity \( \varphi_X : L \to [0, \infty) \). The expected number of points in a set \( K \subset L \) is then defined through

\[
\int_K \varphi_X(z) \, dz = \sum_{m=1}^M \int_{K \cap e_m} \varphi_X(z) \, d_{|m} z,
\]

where \( d_{|m} z \) denotes integration with respect to the curve \( e_m \). Our aim is to estimate the intensity of the point process \( X \) on \( L \) given that we observe realizations \( x_1, x_2, \ldots, x_n \) of this process. The point process \( X \) can equivalently be defined through a density function \( f_X : L \to [0, \infty) \). The probability that a random point \( X_i \sim f_X \) falls into a subset \( K \subset L \) is then given by \( \mathbb{P}(X_i \in K) = \int_K f_X(z) \, dz \).

3. Methodology.

3.1. B-Splines on a Network. First, we briefly review B-splines (compare Ruppert, Wand and Carroll, 2003 or Fahrmeir et al., 2013). To start, assume a simple point process \( X \) with intensity \( \varphi_X(z) \), where \( z \) is univariate and takes values in the bounded interval \([a, b]\). The goal is to estimate \( \varphi_X(z) \) in a smooth and flexible way. To do so, we approximate the logarithmized intensity \( \nu_X = \log \varphi_X \) through a B-spline basis representation \( \nu_X(z) = \sum_{j=1}^J \gamma_j B_j^l(z) \), where \( B_j^l(\cdot) \) are B-splines of order \( l \in \mathbb{N}_0 \) and \( \gamma = (\gamma_1, \ldots, \gamma_J)^\top \) is a vector of regression coefficients that needs to be estimated from the data. For the construction of B-splines, we use \( I \) interior knots \( a = \tau_1 < \cdots < \tau_I = b \). The \( J = I + l - 1 \) basis functions \( B_j^l(\cdot) \) are each locally supported on \( l + 2 \) adjacent knots and can be calculated recursively from lower order basis functions (De Boor, 1972). An important property of a B-spline basis is that \( \sum_{j=1}^J B_j^l(z) = 1 \) holds for \( z \in [a, b] \) and any order of B-splines \( l \). This property also needs to be respected in the geometric network case.

Subsequently, we restrict ourselves to linear B-spline bases for simplicity of presentation. For simplicity of notation we drop the superscript \( l \) in the B-spline notation, i.e. we construct B-splines of order \( l = 1 \) on a geometric network \( L \). Such a basis can be constructed straightforwardly using the one-dimensional definitions from above. On every curve \( e_m \), which has endpoints \( v_i \) and \( v_j \), we specify an equidistant sequence of \( I_m \) knots \( v_i = \tau_{m,1}, \ldots, \tau_{m,I_m} = v_j \) with \( \tau_{m,k} \in e_m \) for \( k = 1, \ldots, I_m \), where \( d_L(\tau_{m,k}, \tau_{m,k-1}) = \delta_m \). Note that a knot which is equal to a vertex \( v \) is contained in the knot sequence of \( \deg(v) \) segments but it still represents the same knot. Other than in the one-dimensional setup, it is in general not possible to choose the set of knots to be equidistant on the entire geometric network \( L \) with respect to all curve lengths \( d_m \). However, we may choose a global knot distance \( \delta \) such that it is close to an equidistant allocation of knots on the entire geometric network. Let therefore \( \lceil \cdot \rceil \) denote the upwards rounded integer and \( \lfloor \cdot \rfloor \) the corresponding downwards rounded integer. We then define

\[
\delta_m = \begin{cases} 
  d_m / \lceil \frac{d_m}{\delta} \rceil, & \frac{d_m}{\delta} - \lfloor \frac{d_m}{\delta} \rfloor \leq 0.5 \\
  d_m / \lfloor \frac{d_m}{\delta} \rfloor, & \frac{d_m}{\delta} - \lfloor \frac{d_m}{\delta} \rfloor > 0.5 
\end{cases}
\]

which leads to curve-specific knot distances \( \delta_m \) which are as similar as possible for a given overall knot distance \( \delta \). Generally, we will choose \( \delta \) rather small such that the differences between the \( \delta_m \) are small and can be considered as negligible. This will become more clear later, when we also introduce a penalization component in the estimation.
Having the set of knots defined as above, we can construct a linear B-spline basis $B$ on the geometric network $L$. First, we use for every segment $e_m$ with endpoints $v_i$ and $v_j$ the equidistant sequence of knots $v_i = \tau_{m,1}, \ldots, \tau_{m,J_m} = v_j$ from above to construct $J_m = I_m - 2$ linear B-splines $B_{m,1}, \ldots, B_{m,J_m}$. These B-splines are defined accordingly to the univariate case by

$$B_{m,k}(z) = \frac{d_L(z, \tau_{m,k})}{\delta_m} \mathbb{1}_{[\tau_{m,k}, \tau_{m,k+1})}(z) + \frac{d_L(\tau_{m,k+2}, z)}{\delta_m} \mathbb{1}_{[\tau_{m,k+1}, \tau_{m,k+2})}(z)$$

for $z \in L$, $m = 1, \ldots, M$ and $k = 1, \ldots, J_m$. Therefore, the B-splines $B_{m,k}$ are only supported on $e_m$ and we denote with $B_e = \{B_{m,1}, \ldots, B_{m,J_m} \mid m = 1, \ldots, M\}$ the set of all these B-splines. We further require that $J_m \geq 1$ for all $m = 1, \ldots, M$ which is fulfilled if $\delta_m \leq \frac{d_m}{2}$ for all $m$. If $\delta_m$ from (2) does not fulfill this constraint, we set $\delta_m = \frac{d_m}{2}$.

In addition to the B-splines defined by (3) we construct a single B-spline around each vertex $v_i \in V$. Therefore, we consider the $\text{deg}(v_i)$ segments which have an endpoint equal to $v_i$ and we numerate them (without loss of generality) with $e_1, \ldots, e_{\text{deg}(v_i)}$. Again, without loss of generality, let $v_i = \tau_{1,1}, \ldots, \tau_{\text{deg}(v_i),1}$, i.e. we order the knots such that the first knot of every segment starting in $v_i$ equals $v_i$ itself, see Figure 2 as example. Then, we define the vertex specific B-spline $B_{(i)}$ for vertex $v_i$ by

$$B_{(i)}(z) = \mathbb{1}_{\{v_i\}}(z) + \sum_{k=1}^{\text{deg}(v_i)} \left[1 - \frac{d_L(v_i, z)}{\delta_k}\right] \mathbb{1}_{(v_i, \tau_{k,2})}(z).$$

for $z \in L$ and $i = 1, \ldots, W$. These B-splines have support $\text{supp}(B_{(i)}) = \bigcup_{k=1}^{\text{deg}(v_i)} [v_i; \tau_{k,2}) \subset L$, i.e. they are supported on $\text{deg}(v_i)$ segments. Note that all summands in (4) are nonnegative and at most one of the summands is positive. This set of B-splines is denoted with $B_e = \{B_{(1)}, \ldots, B_{(W)}\}$. Altogether, we specify the linear B-spline basis on $L$ by $B = B_e \cup B_v$ with dimension $J = |B| = \sum_{m=1}^M J_m + W$. For simplicity of presentation, we index from now on the B-spline Basis by $1, \ldots, J$ and by construction, it holds that $\sum_{j=1}^J B_j(z) = 1$ for $z \in L$. In Figure 2, we depict linear B-splines around the vertex $v_6$ with $\text{deg}(v_6) = 3$ of the network that is shown Figure 1.
3.2. **Intensity Estimation on a Network.** We can now easily adopt the density estimation approach proposed by Eilers and Marx (1996) for univariate data. On our geometric network $L$, we specify a bin width $h_m$ on every segment $e_m$ and then divide $e_m$ into $N_m = \frac{d_m}{h_m}$ bins of the same length such that $L$ is partitioned into $N = \sum_{m=1}^{M} \frac{d_m}{h_m}$ bins in total. As for the knot distances $\delta_m$, it is clear, that $h_m$ can not be the same for all curve segments of $L$. However, also the bin widths are chosen very small when performing intensity estimation with penalized splines. We therefore specify a small global bin width $h$ and define accordingly to (2)

$$h_m = \begin{cases} \frac{d_m}{\lceil \frac{d_m}{h} \rceil}, & \frac{d_m}{h} - \lceil \frac{d_m}{h} \rceil < 0.5 \\ \frac{d_m}{\lfloor \frac{d_m}{h} \rfloor}, & \frac{d_m}{h} - \lfloor \frac{d_m}{h} \rfloor \geq 0.5 \end{cases}.$$ 

If the left endpoint of $e_m$ is $v$, the bins are given by the $N_m$ subsets $[b_{m,k-1}; b_{m,k}) \subset e_m$ for $k = 1, \ldots, N_m$, where $b_{m,k} \in e_m$ satisfies $d_L(v, b_{m,k}) = kh_m$ and $b_{m,0} = v$. Each bin is characterized by its midpoint $z_{m,k}$ which satisfies $d_L(b_{m,k-1}, z_{m,k}) = d_L(z_{m,k}, b_{m,k})$.

Assume now that data on $n$ independently observed points $x_i$ of the point process $\mathcal{X}$ on the geometric network have been observed, with $i = 1, \ldots, n$. We assume that the observed points are not equal to the network’s vertices for identifiability reasons, i.e. each point lies on a single edge. We define with $y_{m,k} \in \mathbb{N} \cup \{0\}$ the number of observations which are contained in the $k$-th bin of the $m$-th segment, i.e.

$$y_{m,k} = \# \{x_i \in L \mid x_i \in [b_{m,k-1}; b_{m,k}), i = 1, \ldots, n\}$$

for $m = 1, \ldots, M$ and $k = 1, \ldots, N_m$. Based on our considerations for the point process $\mathcal{X}$ we assume a Poisson distribution for the counts $y_{m,k}$ such that we have

$$y_{m,k} \mid z_{m,k} \sim \text{Poi}(\lambda_{m,k}),$$

where $\lambda_{m,k}$ is approximated through

$$\lambda_{m,k} = \varphi(\mathcal{X}(z_{m,k})) \cdot h_m = \exp (\nu(\mathcal{X}(z_{m,k}) + \log h_m)), \tag{6}$$

We can consider $\log h_m$ as offset and aim to estimate $\nu(\mathcal{X})$ as continuous log-intensity for $z \in L$ treating the pairs $(y_{m,k}, z_{m,k})$ as independent observations from (5). Therefore, we replace $\nu(\mathcal{X})$ through the B-spline basis representation

$$\nu(\mathcal{X}) = \sum_{j=1}^{J} B_j(z) \gamma_j = \mathbf{B}(z) \gamma, \tag{7}$$

where $\mathbf{B}(z) = (B_1(z), \ldots, B_J(z))$ is a row vector consisting of the B-spline basis from above evaluated at $z \in L$ and $\gamma = (\gamma_1, \ldots, \gamma_J)^T$ is the vector of B-spline coefficients that needs to be estimated from the data $x_1, \ldots, x_n$. Imposing a penalty on the resulting Poisson likelihood leads to the penalized log-likelihood (constant terms are ignored)

$$\ell_{\mathcal{P}}(\gamma; \rho) = \sum_{m=1}^{M} \sum_{k=1}^{N_m} \left[ y_{m,k} \log \lambda_{m,k} - \lambda_{m,k} \right] - \rho \mathcal{P}(\gamma), \tag{8}$$

where $\mathcal{P}(\gamma)$ is a penalty which is defined in the next section and $\rho$ is the smoothing parameter. The estimation of the smoothing parameter is treated later in this section.

If we replace $\gamma$ in (7) with the maximum-likelihood estimate $\hat{\gamma} = \arg\max_{\gamma} \ell_{\mathcal{P}}(\gamma; \rho)$, then $\hat{\nu}(\mathcal{X}) = \hat{\mathbf{B}}(z) \hat{\gamma}$ is an estimate of the log-intensity and thus $\hat{\varphi}(\mathcal{X}) = \exp (\hat{\nu}(\mathcal{X}))$ is an estimate of the intensity of the point process $\mathcal{X}$ for $z \in L$. Also note that for a given $n$ an estimate of the density of $\mathcal{X}$ is given by $\hat{f}(z) = \hat{\varphi}(z)/n$. 

INTENSITY ESTIMATION ON GEOMETRIC NETWORKS

7
3.3. Penalties on a Network. In order to control the smoothness of the intensity estimate and to overcome singularity issues, the penalty $P_r(\gamma)$ multiplied with the smoothing parameter $\rho$ is subtracted from the maximum likelihood criterion, which leads to the penalized log-likelihood (8). In the one-dimensional setting Eilers and Marx (1996) proposed to impose a penalty on the vector of coefficients $\gamma$ that is proportional to the $r$-th order differences of adjacent spline coefficients. The penalty is given by $\sum_{j=r+1}^{J}(\Delta^r \gamma_j)^2$ and for $r = 1$ we have $\Delta^1 \gamma_j = \gamma_j - \gamma_{j-1}$. Higher order penalty terms can be calculated recursively using $\Delta^r(\gamma_j) = \Delta^1(\Delta^{r-1} \gamma_j)$ starting with the first order differences $\Delta^1 \gamma_j$. It is straightforward to extend this idea to penalties on a network. Let $i, j, \ldots, J$ where $J$ is the dimension of the B-spline basis on the geometric network. According to the one-dimensional case, we are interested in the set of pairwise adjacent B-splines or their coefficients, respectively. Hence, we can view the B-splines on the geometric network $L$ itself as a network graph $L_B$ where $S_A$ is defined through a $J \times J$ adjacency matrix $A$. From the definition of the linear B-splines, it follows that $A(i, j) = 1$, if supp$(B(i)) \cap$ supp$(B(j)) \neq \emptyset$ and else $A(i, j) = 0$. In order to define penalties of arbitrary order, we need the $J \times J$ shortest path matrix $S_A$ where $S_A(i, j) = s$, if the B-Splines $B(i)$ and $B(j)$ have minimum distance $s$ in $L_B$. This all-pairs shortest path problem can be solved with complexity $O(J|A|)$ where $|A|$ is the number of non-zero entries in $A$ (Chan, 2012). For illustration, consider again Figure 2. Here, $B_{7,1}$ is adjacent to $B_{7,2}$ as well as $B(6)$ and the shortest path from $B_{7,2}$ to $B_{8,1}$ via $B_{7,1}$ and $B(6)$ has length 3 in $L_B$.

Now, let $D_1 = \{(i, j) \mid S_A(i, j) = 1, 1 \leq i < j \leq J\}$. According to Eilers and Marx (1996) we penalize neighboring coefficients. A first order penalty is then defined by

$$P_1(\gamma) = \sum_{D_1}(\gamma_i - \gamma_j)^2 = (D_1 \gamma)^\top (D_1 \gamma) = \gamma^\top K_1 \gamma,$$

where $D_1 \in \mathbb{Z}^{[D_1] \times J}$ and $K_1 = D_1^\top D_1 \in \mathbb{Z}^{J \times J}$ define the difference matrix and the resulting quadratic form according to the pairwise differences in (9). Further, let $D_2 = \{(i, k, j) \mid S_A(i, k) = 2, S_A(i, j) = S_A(k, j) = 1, 1 \leq i < j \leq J\}$.

Therewith, a second order penalty can be defined by

$$P_2(\gamma) = \sum_{D_2}((\gamma_i - \gamma_k) - (\gamma_k - \gamma_j))^2 = \sum_{D_2}(\gamma_i - 2\gamma_k + \gamma_j)^2 = (D_2 \gamma)^\top (D_2 \gamma) = \gamma^\top K_2 \gamma,$$

where $D_2 \in \mathbb{Z}^{[D_2] \times J}$ and again, $K_2 = D_2^\top D_2 \in \mathbb{Z}^{J \times J}$ results as matrix version from the sum in (10). For illustration, we revisit the B-splines which we depicted in Figure 2, but, restricted to the B-splines $B_1 = B_{7,2}, B_2 = B_{7,1}, B_3 = B(6), B_4 = B_{8,1}$ and $B_5 = B_{9,1}$. Thus, for $\gamma = (\gamma_1, \ldots, \gamma_5)$ the first- and second order penalties $P_1(\gamma)$ and $P_2(\gamma)$ are defined by the difference matrices

$$D_1 = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & 0 & -1 \end{pmatrix} \quad \text{and} \quad D_2 = \begin{pmatrix} 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{pmatrix},$$

respectively. By taking advantage of the shortest path matrix $S_A$ we can define penalties of any order $r$, but usually first and second order differences are used when applying penalized splines.

3.4. Estimation of the Smoothing Parameter. For the estimation of the smoothing parameter $\rho$, we apply the generalized Fellner-Schall method (Wood and Fasiolo, 2017) which is an iterative procedure to estimate the smoothing parameter in generalized additive models (Wood, 2017). The idea behind the Fellner-Schall method is to apply a mixed model approach...
and to optimize the log Laplace approximate marginal likelihood of the model with respect to the smoothing parameter. In each iteration step the estimated model parameters $\hat{\gamma}_\rho$ are obtained by maximizing the penalized log-likelihood (8) while treating $\rho$ from the previous cycle as fixed. Then, the update $\rho_{\text{new}}$ is calculated through

$$
\rho_{\text{new}} = \rho \frac{\text{tr}((\rho K_r) - K_r) - \text{tr}((B^\top W(\hat{\gamma}_\rho) B + \rho K_r)^{-1} K_r)}{\hat{\gamma}_\rho K_r \hat{\gamma}_\rho}
$$

where $\text{tr}(\cdot)$ denotes the trace operator and $(\rho K_r)^{-1}$ denotes a generalized inverse of $\rho K_r$. The calculation of $K_r$ is numerical demanding if the dimension of the parameter vector $\gamma$ is large. However, using standard linear algebra tools we can show that $\text{tr}((\rho K_r) - K_r) = \text{rk}(K_r)/\rho$, where $\text{rk}(\cdot)$ denotes the rank operator. Thus, the calculation of $K_r$ is not necessary. The design matrix $B \in \mathbb{R}^{N \times J}$ of the Poisson model is build by storing the row vectors $B(z_{m,k})$, which are defined accordingly to (7) for $m = 1, \ldots, M$ and $k = 1, \ldots, N_m$ as a matrix. Furthermore, $W(\hat{\gamma}_\rho) = \text{diag}(\hat{\lambda}_1, \ldots, \hat{\lambda}_N, \ldots, \hat{\lambda}_M, \ldots, \hat{\lambda}_{M,N})$ is a weight matrix, where $\hat{\lambda}_{m,k} = \exp(\gamma_{\chi}(z_{m,k}) + \log h_{m})$ is defined through (6). The matrix $B^\top W(\hat{\gamma}_\rho) B + \rho K_r$ is the Fisher information of our model and is therefore positive definite, which guarantees that $\rho_{\text{new}} > 0$ (Wood and Fasiolo, 2017). The iterative procedure stops, if $\rho_{\text{new}}$ in (11) differs only slightly from the previous $\rho$.

3.5. Quantification of Uncertainty. Since (6) directly relates to the setting of a generalized additive model we can use already existing GAM theory in order to assess the uncertainty of our predictions. Therefore, we follow the functionality of the mgcv package (Wood, 2017, Version 1.8-33) in R which produces standard errors based on the Bayesian posterior covariance matrix $V = V(\hat{\gamma})$ of the model parameters. Treating the smoothing parameter $\rho$ as fixed this covariance matrix is given as the inverse of the Fisher information from above, i.e.

$$
V = (B^\top W(\hat{\gamma}_\rho) B + \rho K_r)^{-1}.
$$

3.6. Intensity Depending on Covariates. The methodology from above can easily be extended to allow the intensity to depend on one or several covariates, distinguishing between two kinds of covariates. Firstly, we denote purely network-related covariates as internal covariates. Examples are longitude/latitude, distance to the nearest vertex or the location on the network segment. In applications with road networks, an internal covariate could also be the direction or the kind of a road. Technically, internal covariates can always be associated with a network segment index $m$ and a respective bin index $k$. Secondly, we denote covariates that are not directly related to the network geometry as external covariates. Examples are time, weather conditions, but also the kind of a crime when the point pattern represents the spatial distribution of crimes along a network of streets.

Let $x_1, \ldots, x_C$ be the set of $C \in \mathbb{N}$ covariates to be considered in the model which are already suitably transformed, if required. According to the above considerations the index set $\{1, \ldots, C\}$ disjointly splits into sets $\mathcal{I}$ and $\mathcal{E}$ referring to internal/external covariates, i.e. $\mathcal{I} \cup \mathcal{E} = \{1, \ldots, C\}$. Further, if $\mathcal{E} \neq \emptyset$ let $U$ be the number of unique combinations of the outcomes of the external covariates, otherwise we set $U = 1$. By introducing an additional index $u = 1, \ldots, U$ we change (6) to

$$
\lambda_{m,k,u} = \exp \left( \nu_{\chi}(z_{m,k}) + \sum_{c=1}^{C} s_c(x_{c,m,k,u}) + \log h_{m} \right),
$$

where $s_c(\cdot)$ is the influence function of the covariate $x_c$ which is defined below and the function $\nu_{\chi}$ now serves as smooth baseline log-intensity. In (12), $x_{c,m,k,u}$ is the value of
the covariate $x_c$ measured at location $z_{m,k}$ on the network if the value of $x_c$ in this bin corresponds to the $u$-th unique combination of all external covariates. This notation suggests that an external covariate can also be network related, and indeed, we can e.g. model varying weather conditions over time at different locations of the network. The function $s(c(\cdot))$ determines whether the $c$-th covariate is modeled (log-)linearly or as a smooth term. In the former case $s_c(x_{c_{m,k,u}}) = \beta_c x_{c_{m,k,u}}$, where $\beta_c$ is the corresponding parameter that needs to be estimated. In the latter case the covariate $x_c$ has a B-spline basis representation $s_c(x_{c_{m,k,u}}) = \sum_{j=1}^{J_c} \gamma^{(c)}_j B^j_c (x_{c_{m,k,u}})$, where the objective is to estimate the parameter vector $\gamma_c = (\gamma^{(c)}_1, \ldots, \gamma^{(c)}_{J_c})$, see Section 3.1 for details. Moreover, the spline functions $s_c$ are centered around zero for ensuring identifiability of smooths effects. Altogether, the former parameter vector $\gamma_c$ from Section 3.2 now extends to a parameter vector $\theta$ which further includes the parameters $\beta_c$ and B-spline coefficients $\gamma_c$.

Suppose now that indices $S \subset \{1, \ldots, C\}$ represent covariates, internal or external, which are modeled smoothly. Therefore, we need further $|S|$ penalties $P^{(s)}_{\gamma_s}$ and smoothing parameters $\rho_s$ associated with each of these smooth terms (Section 3.3), where $\rho$ denotes the vector containing these smoothing parameters. Therefore, the log-likelihood that we now need to maximize is given by

$$\ell_P(\theta, \rho) = \sum_{m=1}^{M} \sum_{k=1}^{N_m} \sum_{u=1}^{U} \left[ y_{m,k,u} \log(\lambda_{m,k,u}) - \lambda_{m,k,u} \right] - \rho P_{\gamma} (\gamma) - \sum_{s \in S} \rho_s P^{(s)}_{\gamma_s} (\gamma_s),$$

where $y_{m,k,u}$ denotes the count of observations in the $k$-th bin of the $m$-th curve segment with covariate combination $u$ and $\rho P_{\gamma} (\gamma)$ is the same penalty as in (8). The smoothing parameters $\rho_s$ associated with the smooth functions can also be updated using the Fellner-Schall method from Section 3.4. In particular, we can update many smoothing parameters at practically no additional costs. Moreover, quantification of uncertainty of linear and smooth covariate effects easily extends by taking the resulting inverse penalized Fisher information, as discussed in the previous subsection.

4. Networks and Data. In this section, we introduce three geometric networks that we use throughout the rest of this paper. We further visualize point processes living on these networks and describe properties of the networks, such as the count of edges. In our R implementation all networks are represented as an instance of the class linnet in the R package spatstat (Baddeley, Rubak and Turner, 2015), i.e. all these networks have a representation as a linear network.

**FIG 3.** Left panel: The Chicago crimes network. Right panel: Major roads in the CBD of Melbourne, Australia, allocated to a street network. The circles show the location of parking bays with installed in-ground sensors.
First, we consider the Chicago crimes network, which has already been treated in many papers before, see e.g. Rakshit et al. (2019). This network is available as an object named `chicago` in the R package `spatstat` and shown in the left panel of Figure 3. Various symbols visualize 116 crimes recorded over two weeks in the year 2002 in Hyde Park, Chicago, subdivided into seven kinds of crimes. Therefore, these data represent a marked point process on this geometric network. Most of the crimes seem to occur in the northeastern and northwestern parts of the map extract. A summary of the geometric network itself is given in Table 1.

Secondly, we employ data from the City of Melbourne, Australia. Between August 2011 and May 2012, the city installed in-ground sensors underneath around 4,600 out of more than 20,000 on-street parking lots in the city center of Melbourne. These sensors are capable of recording the arrival time and the departing time of a car to the second. We take data from the period June-August 2019 and consider a subset of 1,618 on-street parking lots with installed in-ground sensors, which are all located in the Central Business District (CBD) of Melbourne and which are released at least once a day on average. Altogether the database compresses 1,907,941 events where one event is defined as the release of a parking lot. Our goal is to detect regions in this area where the occupancy of parking lots fluctuates most, also concerning the time of the day. Therefore, we first need to specify a geometric network where the point process is living on. This network is constructed by only including major streets in this area as well as side streets in which on-street parking lots with sensors are located. In Figure 3 we show the location of the considered on-street parking lots on the corresponding geometric network. A summary of the network is given in Table 1.

The third and endmost data example are road accidents recorded on a network of highways (state highways, interstate highways and US highways) in the southern part of Montgomery County, Maryland, which borders on the District of Columbia in the north. We take data related to 14,571 traffic collisions from the years 2015-2019, which occurred between 6 am and 10 pm. The locations of these incidents on the underlying road network of highways are shown in Figure 4. Here, we already see that the network of highways is denser in the southern part of the map extract with many south-north routes originating from Washington, D.C. Note that we excluded Maryland Route 200 from the network since traffic collisions occurring on this highway are not included in the database. Besides the location of each collision, the dataset includes covariates such as the type of the highway (internal) or the date and time (external) of the incident. Moreover, we added the direction of a street section as a further internal covariate. Again, a summary of the network itself is provided in Table 1.

<table>
<thead>
<tr>
<th>Network</th>
<th>Hyde Park, Chicago</th>
<th>CBD, Melbourne</th>
<th>South Montgomery County</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unit Resolution Source</td>
<td>feet all streets</td>
<td>meters only major streets</td>
<td>kilometers only highways</td>
</tr>
<tr>
<td></td>
<td>spatstat package own representation own representation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>V</td>
<td>$</td>
<td>338</td>
</tr>
<tr>
<td>$</td>
<td>E</td>
<td>$</td>
<td>503</td>
</tr>
<tr>
<td>$</td>
<td>L</td>
<td>$</td>
<td>31,150,21</td>
</tr>
</tbody>
</table>

Table 1

Summary of the geometric networks covered in this paper.

---

2 The massive amounts of data that these sensors produce and many more data related to parking in the City Melbourne are available for gratuitous download at [https://data.melbourne.vic.gov.au/](https://data.melbourne.vic.gov.au/).

3 Data on car crashes can be downloaded from [https://data.montgomerycountymd.gov/Public-Safety/Crash-Reporting-Incidents-Data/bhju-22kf](https://data.montgomerycountymd.gov/Public-Safety/Crash-Reporting-Incidents-Data/bhju-22kf).
5. The Chicago Crimes Network. To begin with the analysis of the Chicago crimes data, we neglect the kind of crime. When estimating the intensity of crimes with our approach, we set $\delta = 10$, $h = 2$ feet and make use of a second-order penalty. We compare the intensity fit with two baseline methods which are both kernel-based and implemented as function `density.lpp` in the `spatstat` package. Firstly, we fit a kernel-based model according to McSwiggan, Baddeley and Nair (2017) which computes the estimates by solving a heat equation on the network. We refer to this as method 1. This method exclusively relies on the shortest path distance as the metric. The bandwidth $\sigma$ of the kernel smoother is selected via likelihood cross-validation using the function `bw.lpp1` while setting the argument `distance = "path"`, yielding an optimal bandwidth of $\sigma = 158.49$ feet. Secondly, we fit another kernel-based model, now using an adaptive two-dimensional smoothing kernel as proposed by (Rakshit et al., 2019). Here, Scott’s rule of thumb (Scott, 2015), which is for linear networks implemented within the function `bw.scott.iso`, yields an optimal bandwidth of $\sigma = 119.55$ feet. We refer to this as method 2. Note that the ratio of the two optimal bandwidths which we obtained for methods 1 and 2, respectively, are in line with the general arguments of Rakshit et al. (2019), equation 18.

The top plot of Figure 5 shows the intensity estimate when employing the penalized spline-based approach. The lower plots show the fitted intensity when method 1 (bottom left) or method 2 (bottom right), respectively, is used. We find that the high-intensity regions on the network are similarly located for all three methods. However, we see the following two major deviations. First, the penalized spline-based method yields higher intensity estimates for the region on top in the middle of the plot. Otherwise, the estimate is akin to the estimate, resulting when fitting method 1 to the data. Second, we consider the fitted intensity with method 2 in the area, marked by the rectangle in the top right corner of the map. It can be seen that in this area the kernel smoother, which is based on the Euclidean distance, estimates a distinctly higher estimate when compared to the other two methods based on the shortest path distance. This might be caused by the data located in the top right corner of the map. These data are close to the rectangle with respect to the Euclidean distance, but the shortest-path distance is larger by a factor of around three.
We repeat the penalized spline intensity estimation from above but now include the kind of crime as a covariate in our model. For comparison, we also fit a Poisson process on the network with the kind of crime as a single covariate and taking the estimate resulting from method 1 from above as offset, for details see McSwiggan (2019). Therefore, we employ the function `lppm` from the `spatstat` package. The resulting parameter estimates on the log-scale of both models, including 95\% confidence intervals, are shown in Figure 6. We see that the effects and their respective standard errors are very similar in both models. However, our model has the advantage that the nonparametric baseline intensity and covariate effects can be estimated within one model. This can be extended to work with multiple covariates if available. When the baseline intensity shall be estimated with a kernel smoother, covariate effects can generally be estimated employing an alternating two-step approach, see e.g. Kauermann (2002) for asymptotic results on the real line. We stress that this is not provided within any of the functions in the `spatstat` package.

6. On-Street Parking in Melbourne, Australia. Our goal in this example is to detect locations in the street network of the CBD of Melbourne where the occupation of on-street parking lots fluctuates most. Therefore, we define an event to be the clearing of an on-street parking lot, and the point process that we observe now has a spatial as well as a temporal structure. However, in areas with more allocated parking lots, there is per se a higher chance of finding a cleared lot. Therefore, we first need to estimate the intensity $\varphi_Z$ of parking lots
FIG 6. Parameter estimates of linear effects and 95% confidence intervals for the kind of crime in the Chicago crimes network, when fitting a penalized spline model with covariates or a Poisson process with constant baseline intensity.

FIG 7. (Baseline) intensity of on-street parking lots (in parking lots per meter) in the CBD of Melbourne. Left panel: Intensity fitted without covariates. Right panel: Baseline intensity of a fit including a covariate which considers closeness to a vertex. Both plots only show (baseline) intensities ≥ 0.1 on a logarithmic color-scale with the underlying network being visualized in grey.

on the network of streets, where \( Z \) is the point process already visualized in the right panel of Figure 3.

The fitted intensity \( \hat{\varphi}_Z \) of allocated parking lots using the penalized spline-based approach is depicted in the left panel of Figure 7. Here, the intensity is visualized on a logarithmic scale, and for reasons of presentation we only show areas on the network where the intensity of on-street parking lots is expected to be at least 0.1 parking lots per meter. However, we also find that the intensity around street crossings is throughout lower than 0.1, which is reasonable when considering the allocation of parking lots shown in Figure 3. Therefore, we fit the model again with a dichotomous network internal covariate, which has a value of 1 if a location on the network is closer than 20 meters to a vertex and 0 else. The resulting baseline intensity is shown in the right panel of Figure 7, again only showing baseline intensities being 0.1 or higher on a logarithmic color scale. We now find that around many intersections, the baseline intensity exceeds the value 0.1. The estimated effect of the internal covariate is \(-1.87\) on the log-scale with a standard error of 0.12. Consequently, we must multiply the baseline intensity by a factor \( \exp(-2.35) \approx 0.15 \) in order to get the intensity estimates at areas closer than 20 meters to a vertex. Note that the overall intensity is not continuous by including this covariate anymore even though the baseline intensity is still continuous.
We now look at cleared parking lots, see Section 4 for details. The data are considered as results of the clearing point process $Y$, whose log-baseline intensity $\varphi_Y$ is again estimated using penalized spline smoothing. We further include a smooth effect $s(t)$ for the time of the day, where the estimate on the log-scale is shown in the left panel of Figure 8. The effect yields an increasing intensity towards the evening, followed by a rapid decrease of the intensity after 6 pm. To quantify, the intensity of the clearing process $Y$ drops by factor $\exp(-0.4 - 0.2) \approx 0.55$, i.e. by more than a half, within two hours. Note that due to the large amounts of data, the confidence bands of the smooth effect $s(t)$ are very narrow.

Overall we want to determine the ratio $\varphi_X = \varphi_Y / \varphi_Z$, which expresses the expected fluctuation rate of parking lots along the network. However, we are only interested in the fluctuation rate where we expect a reasonable number of parking lots, here the locations where $\hat{\varphi}_Z(z) \geq 0.1$, see Figure 7. Here, we make use of $\hat{\varphi}_Z$ which results from the fit when accounting for the distance to the nearest vertex. The baseline intensities $\hat{\varphi}_X(z)$ are normalized such that they can be interpreted as the expected hourly fluctuation rate of a parking lot, which is located at $z \in L$. In order to get the expected fluctuation rate at a specific time of the day $t$, $\hat{\varphi}_X(z)$ needs to be multiplied by factor $\exp(s(t))$. The estimates of the fluctuation process $X$ are shown in right panel of Figure 8, where spots with $\varphi_X(z) \geq 5$ are surrounded by a black circle. We find that high fluctuation rates occur, especially in the southwestern part of the CBD.

7. Car Crashes in Montgomery County, Maryland. As a start, we determine the penalized spline-based intensity fit of traffic collisions on the network of highways shown in
<table>
<thead>
<tr>
<th>Effect</th>
<th>Estimate (s.e.)</th>
<th>Relative risk</th>
<th>95% CI of relative risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interstate highway</td>
<td>-1.560 (0.196)</td>
<td>0.21</td>
<td>[0.14, 0.31]</td>
</tr>
<tr>
<td>US highway</td>
<td>0.597 (0.222)</td>
<td>1.82</td>
<td>[1.18, 2.81]</td>
</tr>
<tr>
<td>East-west</td>
<td>-0.085 (0.110)</td>
<td>0.92</td>
<td>[0.74, 1.14]</td>
</tr>
<tr>
<td>Southeast-northwest</td>
<td>0.059 (0.112)</td>
<td>1.06</td>
<td>[0.85, 1.32]</td>
</tr>
<tr>
<td>Southwest-northeast</td>
<td>-0.435 (0.158)</td>
<td>0.65</td>
<td>[0.47, 0.88]</td>
</tr>
<tr>
<td>Distance to intersection in km</td>
<td>-1.970 (0.466)</td>
<td>0.14</td>
<td>[0.06, 0.35]</td>
</tr>
</tbody>
</table>

**Table 2**

Summary of estimated fixed effects on the log-scale including standard errors, relative risk = \( \exp(\text{estimate}) \) and the 95% confidence interval (CI) of the relative risk.

---

Figure 10. Smooth effect of the time of the day on the exp-scale.

Figure 4, where we do not include covariates in the model. In the left panel of Figure 9 we illustrate the fitted intensity, where the unit of the intensity estimate is traffic collisions per kilometer and year. We see that some routes exhibit throughout very high intensities, most of them originating from Washington, D.C. To name a few of them, these are US highway 29 or Maryland state highways 97 and 355. On the two interstate highways in this area with numbers 270 and 495, there seems to be a relatively low risk of traffic collisions, i.e. there are only a few crashes per kilometer of highway.

As a next step, we include covariates in the model as proposed in Section 3.6. Firstly, these are two categorical covariates, namely the type of highway (categories: state, interstate, or US highway) as well as the direction of the highway (categories: south-north, east-west, southeast-northwest, southwest-northeast), with the first category always representing the respective reference category. Secondly, we include a linear effect for the distance in kilometers to the nearest intersection with another highway. Finally, we include a smooth effect \( s(t) \) for the time of the day \( t \) as we already did in Section 6.

The fitted baseline intensity, when including covariates, is shown in the right panel of Figure 9. We do not have such a clear picture as before when fitting the intensity without covariates, suggesting that covariate effects now explain a large part of the variance. The resulting estimates of the fixed linear effects are listed in Table 2. Indeed, the relative risk of a traffic incident is five times lower on an interstate highway and nearly twice as large on a US highway when compared to Maryland state highways. Both effects are significant on the 95% confidence level. The effect of interstate highways might be associated with the type of construction as these have several lanes with different driving directions being structurally separated. Moreover, interstate highways are usually connected with other highways
through several ramps to avoid contact with oncoming traffic. Concerning the direction of the highways, there seems to be a significant difference of routes proceeding from southwest to northeast when controlling for the other effects included in the model. To quantify the effect, the relative risk of observing a traffic collision is 35% lower when compared to south-north highways. Lastly, we can infer that high-risk areas in the fit without covariates are mainly located close to intersections since the relative risk decreases by 86% ($\approx\exp(-1.97)$) per kilometer distance to the nearest intersection. Finally, we depict in Figure 10 the estimated smooth effect of the time of the day. Here, we see that the significant risk of observing a traffic accident varies enormously with the hour of the day, where we see a peak in the morning between 8 am and 10 am as well as in the afternoon between 2 pm and 6 pm. In the latter period, the relative risk of observing a road accident is more than twice as large as at noon.

8. Simulation Study.

8.1. Integrated Squared Error. We start by employing data simulated on the Chicago crimes network in order to explore the performance of penalized spline based intensity estimation on geometric networks, also with respect to two methods which we briefly discussed in Section 5 above. Therefore, we specify intensity functions $\varphi_{X_n}$ on the Chicago network which satisfy $\int_L \varphi_{X_n}(z) \, dz = n$ and for each of the sample sizes $n = 100, 200, 500, 1000$ we simulate $R = 100$ point processes $X_n = \{x_1, \ldots, x_n\}$. We quantify the prediction error of the $r$-th simulation with sample size $n$ through

$$ISE(\widehat{\varphi}_{X_n}; r) = \frac{1}{n^2} \int_L \left( \widehat{\varphi}_{X_n}(z; r) - \varphi_{X_n}(z) \right)^2 \, dz$$

where $\widehat{\varphi}_{X_n}(\cdot; r)$ denotes the estimate of $\varphi_{X_n}$ based on the $r$-th sample. That is, we quantify the prediction error through the integrated squared error (ISE) between the the estimated density $\hat{f}_X(z) = \widehat{\varphi}_{X_n}(z)/n$ and the true density $f_X(z) = \varphi_{X_n}(z)/n$, which enables us to compare the ISE for different sample sizes. Denoting with $E[\cdot]$ the sample mean in the following, the mean integrated squared error (MISE) estimated from a sample of size $n$ is given by $MISE(\widehat{\varphi}_{X_n}) = E[ISE(\widehat{\varphi}_{X_n}; r)] = \frac{1}{R} \sum_{r=1}^R ISE(\widehat{\varphi}_{X_n}; r)$. Moreover, denoting with $E[\widehat{\varphi}_{X_n}(z; r)] = \frac{1}{R} \sum_{r=1}^R \widehat{\varphi}_{X_n}(z; r)$ the point-wise sample mean of the intensity estimate, we can express the MISE as the sum of the integrated variance (IVar) and the integrated squared bias (ISBias), i.e.

$$MISE(\widehat{\varphi}_{X_n}) = IVar(\widehat{\varphi}_{X_n}) + ISBias(\widehat{\varphi}_{X_n})$$

(13)

$$= \frac{1}{n^2} \int_L E\left[ \left( \widehat{\varphi}_{X_n}(z; r) - E[\widehat{\varphi}_{X_n}(z; r)] \right)^2 \right] \, dz + \frac{1}{n^2} \int_L \left( E[\widehat{\varphi}_{X_n}(z; r)] - \varphi_{X_n}(z) \right)^2 \, dz.$$

We now carry out the process described above for two intensity functions. First, the intensity is chosen to be proportional to the intensity estimate fitted with the kernel method based on the shortest path distance from below, compare the bottom left panel of Figure 5. Second, we simulate point processes $\mathcal{X}$ according to

$$\varphi_{\mathcal{X}}(z) \propto \begin{cases} 1, & z \in e_m \text{ and } m \equiv 0 \pmod{10} \\ 0, & \text{else} \end{cases},$$

(14)

which means that we simulate data according to a uniform distribution on line segments whose edge index is completely divisible by 10 and on all other edges the probability of
observing a datum is zero. Therefore, (14) specifies an intensity function where the data are clustered on some edges of the network and the intensity is not smooth but discontinuous. When fitting the simulated data with one of the three models, we use the same hyperparameters or strategies to determine the hyperparameters as described in Section 5 when fitting the original data.

In Figure 11 we show a boxplot of the resulting ISEs\(^4\) when fitting the intensity of simulated point patterns with different sample sizes making use of the three models discussed above. The left panel of Figure 11 illustrates that the distribution of the ISEs is very similar for a given sample size if the data are simulated according to the smooth intensity function shown in the bottom left panel of Figure 5, and there is an apparent reduction of the ISE if the sample size increases. When simulating from the discontinuous intensity function (14), the ISE is generally more than ten times larger when compared to the first example, see the right panel of Figure 11. In this situation, penalized spline-based estimation is favored against the two kernel-based methods in terms of ISE. Moreover, the two kernel-based methods perform similarly for small sample sizes, but the estimate based on the Euclidean distance shows only a slight reduction of the ISE if the sample size increases. Therefore, we can conclude that if the actual intensity is sufficiently smooth, all the three considered methods exhibit similar estimation errors. However, the penalized spline-based method shows more robustness towards misspecified smoothness when compared to the two considered kernel-based methods.

Figure 12 shows the same simulation results as in Figure 11, but now in terms of the MISE as the decomposition of IVar and ISBias. We see that in both settings, i.e. with data simulated from a smooth intensity function and a discontinuous intensity function, respectively, our method seems to solve the bias-variance trade-off reasonably well. Note that this is, as already known from penalized spline smoothing on the real line (Fahrmeir et al., 2013), achieved by the optimal choice of the smoothing parameter. In this matter, overestimation or underestimation of the smoothing parameter leads to higher bias and less variability or less

\(^4\)It is essential to note that for computing the ISEs, we have used the development version 2.2-1.003 of the \texttt{spatstat.linnet} package.
bias and more variability, respectively. In some settings of our simulation study, it can be seen that the two kernel-based methods do not solve the bias-variance trade-off well. When considering estimates of the method based on the shortest path distance in the first setting, most of the portion of the MISE can be attributed to the IVar if the sample size is large. On the other hand, in the second setting the major portion of the MISE can be attributed to the ISBias, where the same holds for the kernel methods based on the Euclidean distance.

Finally, we also explore the effect of the bin width $h$ and the dimension $J$ of the B-spline basis, which is determined by the knot distance $\delta$. We vary $\delta$ with 5, 10 and 20 feet, the global bin width $h$ is chosen to be the half, a fifth or a tenth of $\delta$, respectively. Here, the analysis is restricted on the sample size $n = 200$ and $\varphi_X$ according to the bottom left panel of Figure 5. The results in Figure 13 show the ISEs with $R = 100$ simulations for each configuration. We find that the ISE hardly varies with different choices of $\delta$ and $h$. Thus, as long as $\delta$ and $h$ are small enough, we can not considerably increase the prediction performance by reducing these two hyperparameters. This result is in line with the motivating arguments in Eilers and Marx (1996) and corresponds to the general results for penalized spline smoothing as derived in Kauermann and Opsomer (2011).

In order to find a proper value for $\delta$ in general, it is often helpful to start with $\delta \approx \frac{1}{2} \min_m d_m$ which is for the Chicago network given by 4.7 feet. This ensures that there is at least one segment-specific B-spline on each $e_m$ and that the segment-specific knot distances $\delta_m$ are of similar size, see Section 3.1. In the above simulation study on the Chicago network, $\delta$ can be increased by at least factor four without loss of prediction performance which is the merit of the penalization. Moreover, since we are operating with linear B-splines, there is no benefit by setting the global bandwidth $h$ to an unnecessarily small value, which is also supported by the simulation results shown in Figure 13.

8.2. Estimation of Covariate Effects. We now want to study the performance of the model extension which we have elaborated in Section 3.6. Therefore, we first define an intensity function $\varphi_{X}$ of a point process $X$ on the Chicago street network from above according
Integrated squared error (scaled by factor 10,000) of the penalized spline based intensity estimate depending on $\delta$ (left panel: $\delta = 5$, middle panel: $\delta = 10$, right panel: $\delta = 20$) and the global bin width $h$. Data are simulated according to the bottom left panel of Figure 5 with $n = 200$ sample points.

Left panel: Parameter estimates when simulating with internal covariates $t_p$ and $x$ setting $\beta_{t_p} = 2$ and $\beta_x = 1$. Right panel: Parameter estimates when simulating with external covariates time and type, i.e., $\beta = (\beta_0, \beta_{t_p}, \beta_{type})^T$, and setting $\beta^{(1)} = (2, 1, 1)^T$, $\beta^{(2)} = (1, -1, 1)^T$, $\beta^{(3)} = (1, 0, -1)^T$, and $\beta^{(4)} = (4, 1, 1)^T$.

Finally, we simulate again according to the intensity function $\varphi_X$ which yields from the shortest path dependent kernel based intensity estimate of the Chicago crimes data. However, the data shall now also dependent on two external covariates. Therefore, we first draw a sample of size 10 from a time dependent covariate $x_t \sim \mathcal{N}(0, 1)$. Secondly, we also consider a dichotomous covariate $x_d$ with values “A” and “B”. Thus, there are

$$\varphi_X(z) \propto \exp \left(2 \cdot t_p + x\right),$$

where $t_p \in [0, 1]$ measures the relative location of the point $z$ on its line segment, with $t_p = 0$ or $t_p = 1$ meaning that $z$ is located on one of its endpoints. Furthermore, $x$ is the $x$-coordinate (in 1000 feet) of $z$ in the plane. Note that this intensity function is not continuous and data simulated according to (15) are clustered towards the right end of each line segment. We simulate $R = 100$ point patterns of sample sizes $n = 100, 200, 500, 1000$ according to (15) and estimate the intensity including $t_p$ and $x$ as linear covariates. These covariates are both internal covariates with effect sizes $\beta_{t_p} = 2$ and $\beta_x = 1$ on the log-scale. The resulting parameter estimates are shown as boxplots in Figure 14. We see that the estimates of both effects are slightly biased towards zero, while the bias is generally larger for the effect of $t_p$ and decreases when the sample size increases. Likewise, for both effects the variances decrease with increasing sample size.
Future. We find that these estimates are generally unbiased. Note that the baseline intensity is chosen to be the highest in scenario 4, which results in the lowest variances of the estimates when compared to scenarios with lower baseline intensity.

9. Discussion and further Work. In this article, we developed a new method for estimating the intensity (or density) of a stochastic process living on a geometric network. We exploited and extended penalized spline estimation to work on a subset of connected curves, denoted as geometric networks. A benefit of this model is its inherent simplicity and the relatedness to well-elaborated statistical concepts such as penalized spline smoothing and generalized additive models. Note that by our definition, an interval \([a, b]\) is a special case of a geometric network \(L\) with \(|E| = 1\) and \(|V| = 2\).

This paper shows that our methodology works for point processes on two-dimensional linear networks embedded in the plane, a particular case of a geometric network. In the future, we plan to implement intensity estimation on geometric networks in general, which is straightforward when considering our general derivations. More precisely, we want to enable to represent geometric networks as the union of parametric curves (see Section 2), also in higher dimensional spaces, and provide visualizations of fitted intensities in two and three dimensions.

As seen in the simulation study and the application examples, the penalization also compensates for non-equidistant knots and bin widths on different segments of \(L\). However, these differences can be made as small as desired by reducing \(\delta\) and \(h\). In the end, this leads to a trade-off between accuracy and computational effort. In a Euclidean space, the penalties used for estimation with B-splines are often based on derivatives of the smoother. However, in a geometric network the question arises how one could define differentiability of a function \(f\) at vertices \(v\) with \(\deg(v) > 2\). Adapting the penalization technique of Eilers and Marx (1996) circumvents this question and proves to be the right choice for our setting.

We envisage many more generalizations and extensions of our method. First, the linear penalized spline approach could be extended to work with higher-order penalized splines, particularly with quadratic or cubic penalized splines. Therewith, the estimated intensities could become even smoother along the network. However, B-splines of order two or higher in Euclidean spaces are differentiable. Therefore, as stated above, it would be much more complicated to construct network-based B-splines of order two or higher around vertices \(v\) with \(\deg(v) > 2\).

Furthermore, if we drop the assumption that the network graph \(L\) should not be directed, we need the geometric representation \(L\) to be possibly directed as well. This means that a curve \(e_m\) additionally is equipped with a direction if \(e_m = (v_i, v_j)\) is a directed edge from \(v_i\) to \(v_j\) but there is no edge from \(v_j\) to \(v_i\). In this case, the distance measure \(d_L\) from above does not define a metric any more since then, \(d_L(z_1, z_2) = d_L(z_2, z_1)\) for \(z_1, z_2 \in L\) does not hold in general. This extension of the model could especially be applied to the Maryland road accident data to investigate whether the intensity varies with the direction of the lane. However, we consider this to be beyond the scope of this paper and aim to tackle this in the future.

\[ U = 20 \] unique combinations of these two covariates. We further introduce a parameter vector \(\beta = (\beta_0, \beta_1, \beta_2)^\top\) of effects on the log-scale, where \(\beta_0\) scales the baseline intensity, \(\beta_1\) is the linear effect of \(x_t\) and \(\beta_2\) is the effect of \(x_d = B\) with respect to \(x_d = A\). Then, for each covariate combination \(u = 1, \ldots, U\) we draw the sample size \(n_u\) from a Poisson distribution with parameter \(\mu_u = \exp(\beta_0 + \beta_1 \cdot x_{t,u} + \beta_2 \cdot 1\{x_{d,u} = B\})\). In the end, we have simulated \(n = \sum_{u=1}^U n_u\) data points on the network. The right panel of Figure 14 shows boxplots of the parameter estimates when conducting this simulation study with \(\beta^{(1)} = (2, 1, 1)^\top, \beta^{(2)} = (1, -1, 1)^\top, \beta^{(3)} = (1, 0, -1)^\top\) and \(\beta^{(4)} = (4, 1, 1)^\top\), respectively. We find that these estimates are generally unbiased. Note that the baseline intensity is chosen to be the highest in scenario 4, which results in the lowest variances of the estimates when compared to scenarios with lower baseline intensity.
REFERENCES


R Core Team (2013). R: A language and environment for statistical computing.

