STPP: Plotting, simulating and Analysing Spatio-Temporal Point Patterns

Edith Gabriel*, Barry Rowlingson° and Peter J Diggle°

*Université d'Avignon, France, °Lancaster University, UK.

July 21, 2011

Abstract: `stpp` is an R package for analysing, simulating and displaying space-time point patterns. It covers many of the models encountered in applications of point process methods to the study of spatio-temporal phenomena and propose an estimator of the space-time inhomogeneous K-function. In this paper we describe space-time point processes and introduce the package `stpp` to new users.

Keywords: epidemiology; inhomogeneous point patterns; spatial statistics; space-time point processes.

1 Introduction

A spatial point pattern is a set of data taking the form of a set of locations, irregularly distributed within a study region S, at which events have been recorded, for example the locations of trees in a naturally regenerated forest (Diggle, 2003).

An observed spatial point pattern can be modelled as a realisation of a spatial stochastic process represented by a set of random variables: $Y(S_m), S_m \subset S$, where $Y(S_m)$ is the number of events occurring in a sub-region S_m of S. Simulation of spatial point processes is mainly implemented in the R packages `spatstat` (Baddeley and Turner, 2005) and `splancs` (Rowlingson and Diggle, 1993).

Many spatial processes of scientific interest also have a temporal component that may need to be considered when modelling the underlying phenomenon (*e.g.* distribution of cases for a disease or assessment of risk of air pollution). Spatio-temporal point processes, rather than purely spatial point processes, must then be considered as potential models. There is an extensive literature on the analysis of point process data in time (e.g. Cox and Isham, 1980; Daley and Vere-Jones, 2003) and in space (e.g. Cressie, 1991; Diggle, 2003; Møller and Waagepetersen, 2003). Generic methods for the analysis of spatio-temporal point processes are less well established; see for example Diggle, 2006 and Gabriel and Diggle, 2009). There is, however, an extensive literature on the use of point process models in the specific field of seismology; see, for example, Zhuang, Ogata and Vere-Jones (2002) and references therein.

In this paper we first define such processes. Then, we present models for spatio-temporal point processes, some algorithms for their simulation, and statistical tools for their analysis.

2 Spatio-Temporal Point Processes

The events of a spatio-temporal point process form a countable set of points, $\mathcal{P} = \{(s_i, t_i) : i = 1, 2, ...\}$, in which $s_i \in S \subset \mathbb{R}^2$ is the location and $t_i \in T \subset \mathbb{R}$ is the time of occurrence of the ith event. In the following, $Y(A)$ denotes the number of events in a region A.

2.1 First-order and second-order properties

First-order properties describe the way in which the expected value of the process varies across the spatio-temporal domain of observation, $S \times T$, where typical S is a polygon and T a single closed interval. These properties are described by the intensity, $\lambda(s, t)$, of the process,

$$\lambda(s,t) = \lim_{|ds|\to 0, |dt|\to 0} \frac{\mathbb{E}\left[Y(ds, dt)\right]}{|ds||dt|},$$

where ds defines a small spatial region around the point s, $|ds|$ is its area, dt is a small interval containing time t, $|dt|$ is the length of this interval and $Y(ds, dt)$ refers to the number of events in ds in the interval of time dt. Thus, informally, $\lambda(s, t)$ is the mean number of events per unit volume at the location (s, t).

Second-order properties define one aspect of spatio-temporal dependence by describing the relationship between numbers of events in pairs of subregions within $S \times T$. The second-order spatio-temporal intensity is defined by:

$$\lambda_2\big((s_i,t_i),(s_j,t_j)\big) = \lim_{|Di|, |D_j|\to 0} \frac{\mathbb{E}\left[Y(D_i)Y(D_j)\right]}{|D_i||D_j|},$$

where $D_i = ds_i \times dt_i$ and $D_j = ds_j \times dt_j$ are small cylinders containing the points (s_i, t_i) and (s_j, t_j) respectively.

Other, essentially equivalent, descriptors of second-order properties include the covariance density,

$$\gamma\big((s_i,t_i),(s_j,t_j)\big) = \lambda_2\big((s_i,t_i),(s_j,t_j)\big) - \lambda(s_i,t_i)\lambda(s_j,t_j)$$

and the radial distribution function or point-pair correlation function

$$g\big((s_i,t_i),(s_j,t_j)\big) = \frac{\lambda_2\big((s_i,t_i),(s_j,t_j)\big)}{\lambda(s_i,t_i)\lambda(s_j,t_j)}.$$

(Cressie, 1991; Diggle, 2003). The covariance density is the point process analogue of the covariance function of a real-valued stochastic process. The pair correlation function can be interpreted informally as the standardised probability density that an event occurs in each of two small volumes centred on the points (s_i, t_i) and (s_j, t_j). For a spatio-temporal Poisson process (to be defined formally in Section 3.1), the covariance density is identically zero and the pair correlation function is identically 1. Larger or smaller values than these benchmarks therefore indicate informally how much more or less likely it is that a pair of events will occur at the specified locations than in a Poisson process with the same intensity.

2.2 Stationarity

Space-time first-order and second-order stationarity

A spatio-temporal point process $\{(s,t), s \in S, t \in T\}$ is first-order and second-order stationary:

- in space, if: $\lambda(t|s) \equiv \lambda(t)$ and $\lambda_2\big(s_i, s_j|t\big) = \lambda_2(s_i - s_j, t)$.
- in time, if: $\lambda(s|t) \equiv \lambda(s)$ and $\lambda_2\big(t_i, t_j|s\big) = \lambda_2(s, t_i - t_j)$.
- in both space and time, if: $\lambda(s,t) = \lambda$ and $\lambda_2\big((s_i,t_i),(s_j,t_j)\big) = \lambda_2(s_i - s_j, t_i - t_j)$.

If the process is also isotropic, $\lambda_2\big((s_i,t_i),(s_j,t_j)\big) = \lambda_2(u,v)$, where $u = \|s_i - s_j\|$ and $v = |t_i - t_j|$.

Space-time second-order intensity reweighted stationarity

A spatio-temporal point process is second-order intensity reweighted stationary and isotropic if its intensity function is bounded away from zero and its pair correlation function depends only on the spatio-temporal difference vector (u,v) (Baddeley, Møller and Waagepetersen, 2000).

2.3 Separability

A spatio-temporal point process is first-order separable if its intensity $\lambda(s,t)$ can be factorised as

$$\lambda(s,t) = m(s)\mu(t), \text{ for all } (s,t) \in S \times T.$$

Second-order separability of a spatio-temporal point process is typically defined only when the process is stationary, in which separability means that the covariance density $\gamma(u,v) = \lambda_2(u,v) - \lambda^2$ factorises as

$$\gamma(u,v) = \gamma_s(u)\gamma_t(v)$$

Note that in general, second-order separability is implied by, but does not imply, independence of the spatial and temporal component processes. Covariance separability of a process does not imply separability. However, a Poisson process has independent components if and only if it is first-order separable of the process itself.

2.4 Static and dynamic plotting of spatio-temporal point process data

The most effective form of display for a spatio-temporal point process data is an animation, repeated viewing of which may yield insights that are not evident in static displays. Nevertheless, static displays are sometime useful summaries. The `sttp` package includes four display functions that we illustrate using data on the locations and times (dates of reporting) of outbreaks of foot-and-mouth disease (FMD), a severe, highly communicable viral disease of farm livestock, during the UK 2001 FMD epidemic.

The dataset `fmd`, included in `sttp` contains a three-column matrix of spatial locations and reported days (counted from 1st February) of FMD outbreaks in the county of Cumbria. Figure 1 shows a static display of the data consisting of locations in the left-hand panel and the cumulative distribution of the times in the right-hand panel. The left-hand panel shows a very uneven distribution which in the context of this data-set is of limited interest without knowledge of the spatial distribution of all of the farms at risk. The right-hand panel shows the characteristic

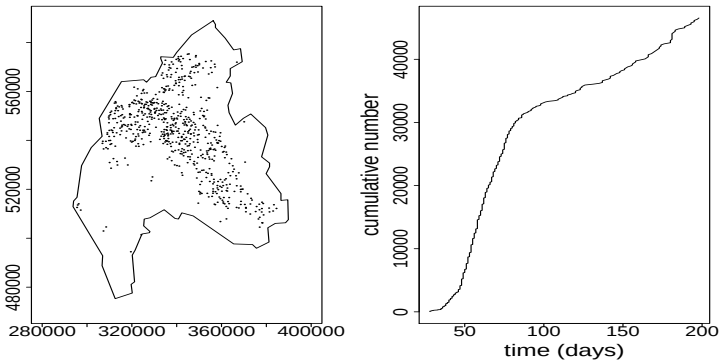


Figure 1: Static two-panel plot of data from the 2001 UK FMD epidemic in the county of Cumbria.

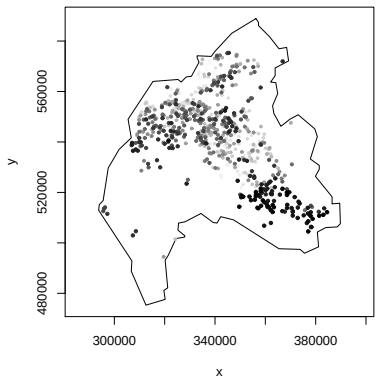


Figure 2: Static plot of data from the 2001 UK FMD epidemic. Time is treated as a quantitative mark; light grey/small dots correspond to the oldest events and dark grey/large dots correspond to the most recent events.

S-shape of an epidemic process. Such plot can be obtained in a single command after converting the dataset into an object of class 'stpp' as follows.

```
> library(stpp)
> data(fmd)
> data(northcumbria)
> fmd=as.3dpoints(fmd)
> plot(fmd, s.region=northcumbria)
```

Figure 2 shows an alternative static display in which the time is treated as a quantitative mark attached to each location, and the locations are plotted with the size and/or colour of the plotting symbol determined by the value of the mark. Such plot can be obtained as follows.

```
> plot(fmd, s.region=northcumbria, pch=19, mark=TRUE)
```

The function `animation` provides an animation of a space-time point pattern.

```
> animation(fmd,runtime=10,cex=0.5,s.region=northcumbria)
```

The approximate running time of the animation (in seconds) is set through the `runtime` parameter, although the animation may actually run more slowly than this, depending on the size of the data-set and the hardware configuration. If `runtime=NULL`, the animation is displayed as quickly as the data-set and hardware configuration allow.

A second form of dynamic display is provided by the `stan` function. This enables dynamic highlighting of time slices controlled by two arguments set by using sliders: when the 'time' slider is set to T and the 'width' slider to W, highlighted points are those whose time coordinate t satisfies $T - W < t < T$. Plotting of individual locations is controlled with the '`states`' parameter. This is a list of length three specifying how locations whose associated times fall before, within and after the time window are displayed. The use of sliders allows the user to track backward or forward in time at will.

```
> library(rgl)
> library(rpanel)
> stan(fmd, twid=1, bgpoly=northcumbria, bgframe=FALSE)
```

Repeated viewing of either of the graphic displays shows two main features of the epidemic. Firstly, there was a progressive movement of the epidemic's focus from its origin in the north of the county to the west, and later to the south-east. Secondly, the pattern consists predominantly of spatio-temporal spread between neighboring farms, but with occasional and apparently spontaneous infections occurring remotely from previously infected areas.

2.5 Space-time inhomogeneous K-function

For a second-order intensity reweighted stationary, isotropic spatio-temporal point process, the space-time inhomogeneous K-function (STIK-function) defined by Gabriel and Diggle (2009) is

$$K_{ST}(u,v) = 2\pi \int_0^v \int_0^u g(u',v')u' \,\mathrm{d}u' \,\mathrm{d}v',$$

where $g(u,v) = \lambda_2(u,v)/\left(\lambda(s,t)\lambda(s',t')\right)$, $u = \|s - s'\|$ and $v = |t - t'|$. They also provide a second definition which considers both past and future events,

$$K^*_{ST}(u,v) = 2\pi \int_{-v}^v \int_0^u g(u',v')u' \,\mathrm{d}u' \,\mathrm{d}v'.$$

Being an alternative characterization of the second-order properties of a point process, the STIK function can be used as a measure of the spatio-temporal aggregation or regularity of clustering. Indeed, for any inhomogeneous spatio-temporal Poisson process (see section 3.1) with intensity bounded away from zero, $K_{ST}(u,v) = \pi u^2 v$. Values of $K_{ST}(u,v)$ greater than $\pi u^2 v$ indicate aggregation at spatial and temporal separations less than u and v, whilst $K_{ST}(u,v) < \pi u^2 v$ indicates regularity. It can also be used to test for space-time clustering and space-time interaction (Gabriel and Diggle, 2009; Møller and Ghorbani, 2011).
The function `STIKhat` provides a non parametric estimation of the STIK function (as defined in Gabriel and Diggle, 2009). It has been applied to FMD data under the assumption of separability of the spatio-temporal intensity.

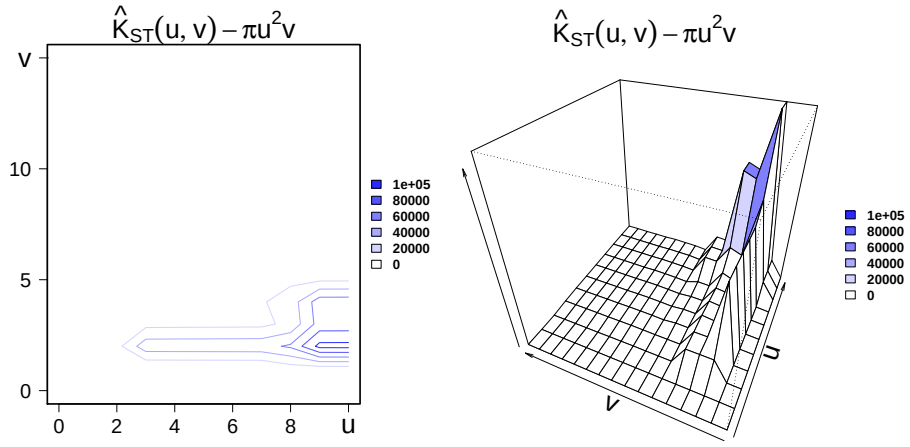


Figure 3: Contour plot (left) and perspective plot (right) of the STIK function estimated from FMD data.

```
> FMD=as.3dpoints(fmd[,1]/1000,fmd[,2]/1000,fmd[,3])

> # estimation of the temporal intensity
> M=density(FMD[,3],kernel="gaussian",n=200)
> mut=M$y[FMD[,3]]*dim(fmd)[1]

> # estimation of the spatial intensity
> h = mse2d(pts=FMD[,1:2], poly=northcumbria/1000, nsmse=100, range=5)
> hs=h$h[h$mse==min(h$mse)]
> require(spatialkernel)
> mhat <- lambdahat(pts=as.points(FMD[,1:2]), h=hs, gpts=as.points(FMD[,1:2]),
          poly = northcumbria/1000, edge = TRUE)$lambda

> # estimation of the STIK function
> u <- seq(0,10,by=1)
> v <- seq(0,15,by=1)
> stik <- STIKhat(xyt=FMD, s.region=northcumbria/1000,t.region=c(1,200),
          lambda=mhat*mut/dim(fmd)[1], dist=u, times=v, infectious=T)
```

We can plot the estimate by using the function `plotK`.

```
> # plotting the estimation
> plotK(stik)
> plotK(stik,persp=T,theta=-65,phi=35)
```

Figure 3 shows such plot for FMD data.

3 Models

3.1 Poisson process

Homogeneous Poisson process

The homogeneous Poisson process represents the simplest possible stochastic mechanism for the generation of spatio-temporal point patterns. It is rarely plausible as a model for data, but provides a benchmark of complete spatio-temporal randomness (CSTR). Informally, in a realisation of a homogenous Poisson process on any spatio-temporal region $S \times T$, the events form an independent random sample from the uniform distribution on $S \times T$. More formally, the homogeneous Poisson process is defined by the following postulates, which correspond to the definition of CSTR:

1. For some $\lambda > 0$, the number $Y(S \times T)$ of events within the region $S \times T$ follows a Poisson distribution with mean $\lambda|S|T$, where T is the length of the time interval T.

2. Given $Y(S \times T) = n$, the n events in $S \times T$ form an independent random sample from the uniform distribution on $S \times T$.

The first-order and second-order intensities of a homogeneous Poisson process reduce to constants, $\lambda(s,t) = \lambda$ and $\lambda_2\big((s_i,t_i),(s_j,t_j)\big) = \lambda^2$. Hence, as stated in Section 2.1, the covariance density is identically zero, the pair correlation function identically 1, and the STIK function is $K_{ST}(u,v) = \pi u^2 v$.

Simulation

To generate a homogeneous Poisson point pattern in $S \times T$ a two step procedure can be used.

1. *Simulate the number of events $n = Y(S \times T)$ occurring in $S \times T$ according to a Poisson distribution with mean $\lambda|S|T$.*

2. *Sample each of the n locations and n times according to a uniform distribution on S and on T respectively.*

Inhomogeneous Poisson process

The inhomogeneous Poisson process is the simplest non-stationary point process. It is obtained replacing the constant intensity λ of a homogeneous Poisson process by a spatially and/or temporally varying intensity function $\lambda(s,t)$. Inhomogeneous Poisson processes are defined by the following postulates:

1. The number $Y(S \times T)$ of events within the region $S \times T$ follows a Poisson distribution with mean $\int_S \int_T \lambda(s,t)\,\mathrm{d}t\,\mathrm{d}s$.

2. Given $Y(S \times T) = n$, the n events in $S \times T$ form an independent random sample from distribution on $S \times T$ with probability density function $f(s,t) = \lambda(s,t) / \int_S \int_T \lambda(s',t')\,\mathrm{d}t'\,\mathrm{d}s'$.

For any Poisson process, with intensity $\lambda(s,t)$, the second-order intensity is $\lambda_2\big((s_i,t_i),(s_j,t_j)\big) = \lambda(s_i,t_i)\lambda(s_j,t_j)$, hence the covariance density is identically zero, the pair correlation function identically 1, and the STIK fucntion $K_{ST}(u,v) = \pi u^2 v$ as in the homogeneous case.

Simulation

To generate a realisation of an inhomogeneous Poisson process in $S \times T$, we use a thinning algorithm as follows. For a given intensity function $\lambda(s,t)$:

1. *Define an upper bound λ_{max} for the intensity function $\lambda(s,t)$.*
2. *Simulate a homogeneous Poisson process with intensity λ_{max}.*
3. *"Thin" the simulated process according to the following way:*
   (a) *Compute $p = \lambda(s,t)/\lambda_{max}$ for each point (s,t) of the homogeneous Poisson process.*
   (b) *Generate a sample u from the uniform distribution on $(0,1)$.*
   (c) *Retain the locations for which $u \leq p$.*

Examples

Poisson processes are simulated by the function `rpp`. Realisations are simulated in a region $S \times T$, where S is a polygon and T is an interval; default is the unit cube. For a homogeneous Poisson process, the intensity is specified by a constant. For example, the sequence of commands

```
> hpp1 = rpp(lambda=200, nsim=5, replace=FALSE)
> stan(hpp1$xyt[[2]])
```

generates five realisations of the Poisson process with intensity $\lambda = 200$ in the unit cube and displays the second realisation.

The sequence of commands

```
> data(northcumbria)
> hpp2 = rpp(npoints=1000, s.region=northcumbria, t.region=c(1,500),
 discrete.time=TRUE)
> polymap(northcumbria)
> animation(hpp2$xyt, s.region=hpp2$s.region)
```

generates and displays a realisation of the Poisson process with intensity $\lambda = 1000/(|S|T)$ conditioned to produce exactly 1000 points in the region $S \times T$, where S is the county of Cumbria and $T = [1, 500]$. The argument `npoints` specifies the fixed value of the number of events to be generated. Simulated times are restricted to integers (set by the `discrete.time` parameter), and coincident times are allowed (set by the `replace` argument).

The function `rpp` can also generate realisations of inhomogeneous Poisson processes. The intensity is then specified either by a function of the coordinates and times, $\lambda(x,y,t,...)$, or by a pixel image if the `lambda` parameter is a character. For example, the sequence of commands

```
> lbda1 = function(x,y,t,a){a*exp(-4*y) * exp(-2*t)}
> ipp1 = rpp(lambda=lbda1, npoints=200, a=1600/((1-exp(-4))*(1-exp(-2))))
> stan(ipp1$xyt)
```

generates 200 points of the Poisson process with intensity $\lambda(x, y, t) = a \exp(-4y - 2t)$ in the unit cube. When the `npoints` argument is omitted, the number of points is not fixed by the user but is generated by a Poisson distribution with mean $\iint_S \int_T \lambda(x, y, t, ...) \, \mathrm{d}t \, \mathrm{d}x \, \mathrm{d}y$. Realisations can also be generated when the intensity is specified by a spatio-temporal intensity array. In the following example, we estimate the spatial and temporal intensities of the `fmd` data by kernel smoothing (Diggle, 1985; Silverman, 1986; Berman and Diggle, 1989) and display the realisation superimposed on a grey-scale image of the spatial intensity estimate.

```
> data(fmd)
> data(northcumbria)
> h = mse2d(as.points(fmd[,1:2]), northcumbria, nsmse=30, range=3000)
> h = h$h[which.min(h$mse)]
> Ls = kernel2d(as.points(fmd[,1:2]), northcumbria, h, nx=100, ny=100)
> Lt = dim(fmd)[1]*density(fmd[,3], n=200)$y
> Lst=array(0,dim=c(100,100,200))
> for(k in 1:nt) Lst[,,k] <- Ls$z*Lt[k]/dim(fmd)[1]
> ipp2 = rpp(lambda="m", Lambda=Lst, s.region=northcumbria,
  t.region=c(1,200), discrete.time=TRUE)
> image(Ls$x, Ls$y, Ls$z, col=grey((1000:1)/1000)); polygon(northcumbria)
> animation(ipp2$xyt, add=TRUE, cex=0.5, runtime=15)
```

3.2 Poisson cluster process

We define a Poisson cluster process as follows (Neyman and Scott, 1958).

1. Parents form a Poisson process with intensity $\lambda_p(s, t)$.
2. The number of offspring per parent is a random variable N_c with mean m_c, realised independently for each parent.
3. The positions and times of the offspring relative to their parents are independently and identically distributed according to a trivariate probability density function $f(\cdot)$
4. The final process is composed of the superposition of the offspring only.

Simulation

To generate a Poisson cluster point process in $S \times T$ we use the following three-step procedure:

1. *Simulate a Poisson process of parent points with intensity $\lambda_p(s, t)$ in $S' \supset S$ to avoid edge effects (thus, the contributions of offspring falling in S from parents outside S are not lost).*
2. *For each simulated parent, generate a random number n_c of offspring from a Poisson distribution with mean m_c.*
3. *Generate the spatio-temporal displacements the offspring from their parent as independent realisations from the trivariate distribution with density $f(\cdot)$.*

Examples

The function `rpcp` generates points around a number of parents points generated by `rpp`. Their spatial and temporal distributions can be chosen among "uniform", "normal" and "exponential" using the `cluster` argument. This can be either a single value if the distribution in space and time is the same, or a vector of length two, giving first the spatial distribution of children and then the temporal distribution. The parameter `dispersion` corresponds to a scale parameter. It equals twice the standard deviation of location of children relative to their parent for a normal distribution of children; the mean for an exponential distribution and half range for an uniform distribution. By default, `edge="larger.region"`. The function generates the Poisson cluster process within a larger region but return only those points that fall within $S \times T$. If `edge="without"` the process is generated only in $S \times T$ and will have an artificially reduced intensity near the boundary of S. By default, the larger spatial region is the convex hull of `s.region` enlarged by the spatial related value of `dispersion` and the larger time interval is `t.region` enlarged by the temporal related value of `dispersion`. One can over-ride default using the 2-vector parameter `larger.region`.

In the following example, parents are generated by a homogeneous spatio-temporal Poisson process with intensity $\lambda = n_p/(|S|T)$, where S is the boundary of Cumbria, $T = [1, 365]$ and $n_p = 50$ is the number of parents. Each parent gives birth to a series of offspring; the number of children per parent follows a Poisson distribution with mean `mc`.

```
> data(northcumbria)
> pcp1 <- rpcp(nparents=50, mc=10, s.region=northcumbria, t.region=c(1,365),
  cluster=c("normal","exponential"), dispersion=c(5000,5))
> animation(pcp1$xyt, s.region=pcp1$s.region, t.region=pcp1$t.region,runtime=5)
```

The sequence of commands

```
> lbda <- function(x,y,t,a){a*exp(-4*y) * exp(-2*t)}
> pcp2 <- rpcp(nparents=50, npoints=250, cluster="normal", lambda=lbda,
  a=2000/((1-exp(-4))*(1-exp(-2))))
> stan(pcp2$xyt)
```

generates a realisation of the Poisson cluster process in the unit cube and displays the realisation. Here, the parent process is Poisson with intensity $\lambda(x, y, t) = a \exp(-4y - 2t)$ and the offspring are normally dispersed both in space and in time.

3.3 Interaction processes

Inhibition process

Inhibition processes either prevent (strict inhibition) or make unlikely the occurrence of pairs of close events, to patterns that are more regular in space and/or in time than a Poisson process of the same intensity.

In a strict inhibition process, let δ_s denote the minimum permissible distance between events and λ_s the spatial intensity of the process. The proportion of the plane covered by non-overlapping discs of radius $\delta_s/2$ is $\rho = \lambda_s \pi \delta_s^2/4$, which we call the packing density. The maximum achieveable packing density is for a pattern of points in a regular triangular lattice

at spacing δ_s, for which $\rho = \sqrt{3}/2 \approx 0.87$. Depending on exactly how the points are generated, even this value of δ_s may not be feasible; for example, if the points are placed sequentially at random in a large region S, the maximum achieveable packing density is approximately 0.55 (Tanemura, 1979).
Simple sequential inhibition processes in space and time are defined by the following algorithm. Consider a sequence of m events (s_i, t_i) in $S \times T$. Then,

1. s_1 and t_1 are uniformly distributed in S and T respectively.
2. At the kth step of the algorithm, $k = 2, ..., m$, s_k is uniformly distributed on the intersection of S with $\{s : \|s - s_j\| \geq \delta_s, j = 1, \ldots, k-1\}$ and t_k is uniformly distributed on the intersection of T with $\{t : |t - t_j| \geq \delta_t, j = 1, \ldots, k-1\}$.

To obtain a larger class of inhibition processes than the one defined above, we extend condition 2. of the above algorithmic definition by introducing functions $p_s(u)$ and $p_t(v)$ which together determine the probability that a point at location s and time t will be accepted as an event of the process, according to the following algorithm.

1. s_1 and t_1 are uniformly distributed in S and T respectively.
2. At the kth step of the algorithm, $k = 2, ..., m$,
   (a) Generate uniformly a location $s \in S$ and a time $t \in T$.
   (b) Generate $u_s \sim \mathcal{U}[0,1]$ and $u_t \sim \mathcal{U}[0,1]$.
   (c) If $\|s - s_j\| \geq \delta_s$ for all $j = 1, \ldots, k-1$, then set $p_s = 1$.
   Otherwise compute $p_s = g_s\left(h_s\left((\|s - s_j\|)_{j=l,\ldots,k-1}, \theta_s, \delta_s\right), r\right)$
   (d) If $|t - t_j| \geq \delta_t$ for all $j = 1, \ldots, k-1$, then set $p_t = 1$.
   Otherwise compute $p_t = g_t\left(h_t\left((|t - t_j|)_{j=l,\ldots,k-1}, \theta_t, \delta_t\right), r\right)$
   (e) If $u_s < p_s$ and $u_t < p_t$, then keep s and t.

The functions g_s and g_t can be chosen among "min", "max" and "prod" and depend on a parameter r. This parameter allows us to consider either the product of terms $h_s(\cdot)$ and $h_t(\cdot)$ from all previous events or only the most recent. The functions h_s and h_t depend on the distance between locations and times respectively. They are monotone, increasing, tend to 1 when the distance tends to infinity and satisfy $0 \leq h(\cdot) \leq 1$. Currently the following functions are implemented:

- step: $h(x) = \begin{cases} 1, & \text{if } x > \delta \\ \theta, & \text{otherwise} \end{cases}$, $\theta \in [0,1]$
- gaussian: $h(x) = \begin{cases} 1, & \text{if } x > \delta + \theta/2 \\ \exp\left\{-\frac{(x-\delta-\theta/2)^2}{2(\theta/8)^2}\right\}, & \text{if } \delta < x \leq \delta + \theta/2 \\ 0, & \text{if } x \leq \delta \end{cases}$, $\theta \geq 0$.

Figure 4 gives two examples. The left-hand panel displays the step function for $\delta = 4$ and $\theta = 0.3$ whilst the right-hand panel shows the Gaussian function for $\delta = 2$ and $\theta = 9$.

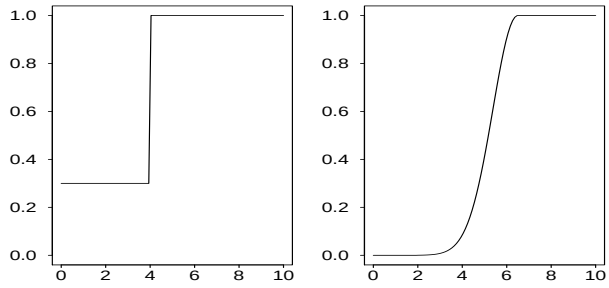


Figure 4: Inhibition functions implemented in `rinter`; *left*: step, *right*: gaussian.

Contagious process

A simple contagious processes in space and time can be defined algorithmically as follows. Consider a sequence of m events (s_i, t_i) in $S \times T$. Then,

1. s_1 and t_1 are uniformly distributed in S and T respectively.
2. At the kth step of the algorithm, given $\{(s_j, t_j), j = 1, \ldots, k-1\}$, s_k is uniformly distributed on the intersection of S and the circle of center s_{k-1} and radius δ_s, whilst t_k is uniformly distributed on the intersection of T and the segment $[t_{k-1}, t_{k-1} + \delta_t]$.

As in the case of inhibitory processes, we enlarge class of contagious processes by introducing functions p_s and p_t, which depends on $\|s - s_j\|$ and $|t - t_j|$ respectively. The kth step of this algorithm is

1. Generate uniformly a location $s \in S$ and a time $t \in T$.
2. Generate $u_s \sim \mathcal{U}[0,1]$ and $u_t \sim \mathcal{U}[0,1]$.
3. If $\|s - s_j\| < \delta_s$ for all $j = 1, \ldots, k-1$, then set $p_s = 1$.
   Otherwise compute $p_s = g_s\left(h_s\left(\left(\|s - s_j\|\right)_{j=l,\ldots,k-1}, \theta_s, \delta_s\right), r\right)$
4. If $|t - t_j| < \delta_t$ for all $j = 1, \ldots, k-1$, then set $p_t = 1$.
   Otherwise compute $p_t = g_t\left(h_t\left(\left(|t - t_j|\right)_{j=l,\ldots,k-1}, \theta_t, \delta_t\right), r\right)$
5. If $u_s < p_s$ and $u_t < p_t$, then keep s and t.

The functions h_s and h_t depend on the distance between locations and times respectively. They are monotone, decreasing, tend to 1 when the distance tends to 0 and satisfy $0 \leq h(\cdot) \leq 1$. The implemented h functions are the following.

- step: $h(x) = \begin{cases} 1, & \text{if } x \leq \delta \\ \theta, & \text{otherwise} \end{cases}$, $\theta \in [0,1]$
- gaussian: $h(x) = \begin{cases} 1, & \text{if } x \leq \delta \\ \exp\left\{-\frac{(x-\delta)^2}{2(\theta/8)^2}\right\}, & \text{otherwise} \end{cases}$, $\theta \geq 0$.

Figure 5 gives two examples. The left-hand panel displays the step function for $\delta = 4$ and $\theta = 0.3$ and the right-hand panel shows the Gaussian function for $\delta = 2$ and $\theta = 9$.

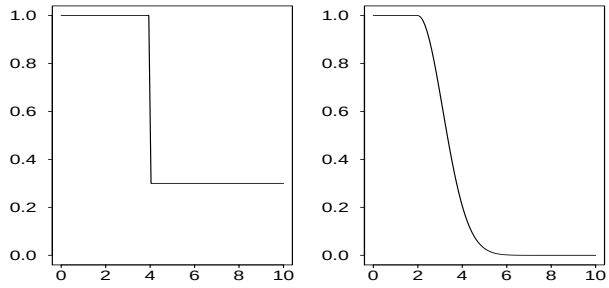


Figure 5: Contagious functions implemented in `rinter`; *left*: step, *right*: gaussian.

Examples

The function `rinter` generate both inhibition and contagious processes, differentiated by the parameter `inhibition`.

The simple sequential inhibition process is obtained by choosing "min" for g functions, "step" for h functions and 0 for θ parameters. The commands

```
> inh1 = rinter(npoints=200, thetas=0, deltas=0.05, thetat=0, deltat=0.001,
 inhibition=TRUE)
> stan(inh1$xyt)
```

generate one realisation of this process in the unit cube and display the realisation.

Similarly, the commands

```
> data(northcumbria)
> cont1 = rinter(npoints=250, s.region=northcumbria, t.region=c(1,200), thetas=0,
  deltas=7500, thetat=0, deltat=10, recent=1, inhibition=FALSE)
> plot(cont1$xyt, pch=19, s.region=cont1$s.region, mark=TRUE, mark.col=4)
> animation(cont1$xyt, s.region=cont1$s.region, t.region=cont1$t.region,
  incident="red", prevalent="lightgreen", runtime=15, cex=0.8)
```

generate one realisation of the simple contagious process in a given spatio-temporal region and display the realisation. The simple contagious process is given using "step" for h functions, 0 for θ parameters and $r = 1$.

The user can also call their own functions h_s and h_t, which can combine inhibitory and contagious elements provided that the functions only depend on d (spatial or temporal distance between points), θ and δ. This is illustrated in the following example.

```
> # defining and plotting hs and ht
> hs = function(d,theta,delta,mus=0.1){
>  res=NULL
>  a=(1-theta)/mus
>  b=theta-a*delta
>  for(i in 1:length(d))
>  {
>  if (d[i]<=delta) res=c(res,theta)
```

```
>  if (d[i]>(delta+mus)) res=c(res,1)
>  if (d[i]>delta & d[i]<=(delta+mus)) res=c(res,a*d[i]+b)
>  }
>  return(res)}

> ht = function(d,theta,delta,mut=0.3){
>  res=NULL
>  a=(1-theta)/mut
>  b=theta-a*delta
>  for(i in 1:length(d))
>  {
>  if (d[i]<=delta) res=c(res,theta)
>  if (d[i]>(delta+mut)) res=c(res,1)
>  if (d[i]>delta & d[i]<=(delta+mut)) res=c(res,a*d[i]+b)
>  }
>  return(res)}

> d=seq(0,1,length=100)
> plot(d, hs(d0.2,0.1,0.1), xlab="", ylab="", type="l", ylim=c(0,1), lwd=2, las=1)
> lines(d, ht(d,0.1,0.05,0.3), col=2, lwd=2)
> legend("bottomright", col=1:2, lty=1, lwd=2, bty="n", cex=2,
  legend=c(expression(h[s]),expression(h[t])))

> # generating the inhibition process
> inh2 = rinter(npoints=100, hs=hs, gs="min", thetas=0.2, deltas=0.1, ht=ht,
gt="min", thetat=0.1, deltat=0.05, inhibition=TRUE)
> animation(inh2$xyt,runtime=15,cex=0.8)
```

3.4 Infectious processes

SIR models are widely used to describe the progress of an epidemic (see, for example, Becker, 1989). These models consider that at any time, each member of a population will be in one of three states: susceptible (S), infectious (I) and recovered or removed (R). To each infected individual at a time t there corresponds an infection rate $h(t)$ which depends on three parameters: a latent period α, the maximum infection rate β and the infection period γ.

We define an infectious process in space and time as follows. Consider a sequence of m events $\{(s_i, t_i), i = 1, \ldots, m\}$ in $S \times T$. Then,

1. s_1 and t_1 have distributions f_s and f_t in S and T respectively.
2. Given $\{(s_j, t_j), j = 1, \ldots, k-1\}$, s_k is either radially symmetrically distributed around s_{k-1} or is Poisson distributed with intensity $\lambda(s)$, and t_k is either uniformly or exponentially distributed from t_{k-1}. We denote by f_s and f_t the distribution of s_j and t_j respectively.

I DON'T UNDERSTAND THE ABOVE DEFINITION. HOW CAN A LOCATION S_K HAVE A POISSON DISTRIBUTION?

WE MAY WANT TO GENERATE EVENTS FROM A CONTAGIOUS PROCESS WHICH REFLECTS POPULATION DENSITY (AS GIVEN IN EXAMPLE, SEE inf2 EXEMPLE)

Simulation

The algorithm used in the `stpp` package to simulate an infectious processes is as follows.

Step 0

1. Set t_0 and generate randomly in S a location s_0.

Step k

2. Compute $h_k(t) = \frac{1}{\int_T h(u)\,\mathrm{d}u} h(t|t_{k-1}, \alpha, \beta, \gamma)$ and $\mu_k(t) = \sum_{j=l}^{k} h_j(t)$
3. Generate $u_k \sim \mathcal{U}[0,1]$.
4. Generate t_k from f_t and s_k from f_s.
5. If $\begin{cases} \|s_k - s_j\| \geq \delta_s, & \text{for an inhibition process} \\ \|s_k - s_j\| < \delta_s, & \text{for a contagious process} \end{cases}$, $j = 1, \ldots, k-1$, then set $p_k = 1$.
   Otherwise, compute $p_k = g\left(\mu_k(t_0, \ldots, t_k), r\right)$.
6. If $u_k < p_k$, then keep (s_k, t_k).

The function g can be chosen amongst "min", "max" and "prod" and depends on the parameter r which allows us to consider either all previous events or only the most recent. The spatial distribution f_s can be chosen among:

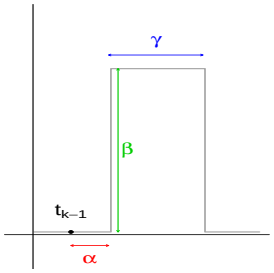
- uniform: $s_k = (x_k, y_k)$, where $x_k = x_{k-1} + \mathcal{U}[-d_s, d_s]$ and $y_k = y_{k-1} + \mathcal{U}[-d_s, d_s]$.
- gaussian: $s_k = (x_k, y_k)$, where $x_k = x_{k-1} + \mathcal{N}(0, d_s/2)$ and $y_k = y_{k-1} + \mathcal{N}(0, d_s/2)$.
- exponential: $s_k = (x_k, y_k)$, where $x_k = x_{k-1} \pm \mathcal{E}xp(1/d_s)$ and $y_k = y_{k-1} \pm \mathcal{E}xp(1/d_s)$.
- poisson: $s_k \sim \mathcal{P}oiss\left(\lambda(s)\right)$.

The temporal distribution f_t can be chosen among:

- uniform: $t_k = t_{k-1} + \mathcal{U}[0, d_t]$.
- exponential: $t_k = t_{k-1} + \mathcal{E}xp(1/d_t)$.

The infection rate h depends on t_{k-1}, α (latent period), $\beta \in [0,1]$ (maximum infection rate) and γ (infection period). It can be chosen among:

- step: $h(t) = \begin{cases} \beta, & \text{if } t_{k-1} + \alpha \leq t \leq t_{k-1} + \alpha + \gamma \\ 0, & \text{otherwise} \end{cases}$,



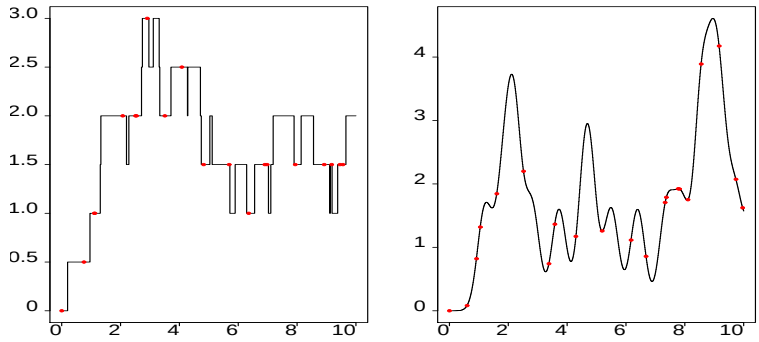


Figure 6: Illustration of $\mu(t)$ for h as a step function (left) or a gaussian function (right).

- gaussian: $h(t) = \beta \exp\left\{-\big(t - (t_{k-1} + \alpha + \gamma/2)\big)^2/2(\gamma/8)^2\right\}$.

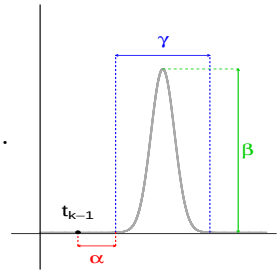


Figure 6 illustrates $\mu(t)$ for h as a step function (left) or a gaussian function (right), with $\alpha = 0.2$, $\beta = 0.7$ and $\gamma = 2$. Dots correspond to the times of events.

Examples

The function `rinfec` generates infectious processes defined by an infection rate function $h(\alpha, \beta, \gamma)$ (parameters `h`, `alpha`, `beta` and `gamma`) and inhibition or contagious processes (differentiated by the parameter `inhibition`). Spatial and temporal distribution are specified through the parameters `s.distr`, `t.distr` and `maxrad`, where `maxrad` is a 2-vector defining the spatial and temporal radiation respectively. The spatial distribution may be "gaussian", "exponential" or "poisson", and the temporal distribution may be "uniform" or "exponential". The probability of acceptance of a new point is computed by a function $g(h(\cdot), r)$ chosen among "min", "max" and "prod". Parameter `recent` allows to consider either all or the r most recent events.

The sequence of commands

```
> inf1 = rinfec(npoints=100, alpha=0.1, beta=0.6, gamma=0.5, maxrad=c(0.075,0.5),
  t.region=c(0,50), s.distr="uniform", t.distr="uniform", h="step", g="min",
  recent="all", inhibition=TRUE)
> animation(inf1$xyt, cex=0.8, runtime=10)
```

generates one realisation of an infectious/inhibition process and displays the realisation over the spatial intensity estimate.

When the spatial distribution is Poisson, its intensity can be defined by a function $\lambda(x, y, t, ...)$ or by matrix. The following example illustrates the case of an intensity defined by a matrix, here corresponding to the kernel estimate of the `fmd` dataset. The commands

```
> data(fmd)
> data(northcumbria)
> h = mse2d(as.points(fmd[,1:2]), northcumbria, nsmse=30, range=3000)
> h = h$h[which.min(h$mse)]
> Ls = kernel2d(as.points(fmd[,1:2]), northcumbria, h, nx=50, ny=50)
> inf2 = rinfec(npoints=100, alpha=4, beta=0.6, gamma=20, maxrad=c(12000,20),
  s.region=northcumbria, t.region=c(1,2000), s.distr="poisson", t.distr="uniform",
  h="step", g="min", recent=1, lambda=Ls$z, inhibition=FALSE)
> image(Ls$x, Ls$y, Ls$z, col=grey((1000:1)/1000)); polygon(northcumbria,lwd=2)
> animation(inf2$xyt, add=TRUE, cex=0.7, runtime=15)
```

generate one realisation of an infectious/contagious process in a given space-time region and display the realisation.

3.5 Log-Gaussian Cox processes

Cox processes are doubly stochastic point processes formed as inhomogeneous Poisson processes with stochastic intensity. Such processes were introduced by Cox (1955) in one temporal dimension. Their definition in space and time is:

1. $\{\Lambda(s,t) : s \in S, t \in T\}$ is a non-negative-valued stochastic process.
2. Conditional on $\{\Lambda(s,t) = \lambda(s,t) : s \in S, t \in T\}$, the events form an inhomogeneous Poisson process with intensity $\lambda(s,t)$.

Suppose that $Z = \{Z(s,t); s \in S, t \in T\}$ is a real-valued Gaussian process with mean $\mu(s,t) = \mathbb{E}[Z(s,t)]$ and covariance function $c\big((s_i,t_i),(s_j,t_j)\big) = \mathrm{Cov}\big(Z(s_i,t_i), Z(s_j,t_j)\big)$. If the intensity function is defined by $\Lambda(s,t) = \exp\{Z(s,t)\}$, then the corresponding process Y is a Log-Gaussian Cox process.

Simulation

The simulation of log-Gaussian Cox processes in the `stpp` package uses the following the same thinning algorithm as was used for inhomogeneous Poisson processes, but preceded by a simulation of the underlying Gaussian process. For a given covariance function $c\big((s_i,t_i),(s_j,t_j)\big)$ and a given mean $\mu(s,t)$ for the Gaussian process:

1. *Generate a realisation of a Gaussian field, with covariance function* $c\big((s_i,t_i),(s_j,t_j)\big)$ *and mean* $\mu(s,t)$.
2. *Define* $\lambda(s,t) = \exp\{Z(s,t)\}$ *and an upper bound* λ_{max} *for* $\lambda(s,t)$.
3. *Simulate a homogeneous Poisson process with intensity* λ_{max}.
4. *Thin the simulated process as follows:*
   (a) *Compute* $p = \lambda(s,t)/\lambda_{max}$ *for each point* (s,t) *of the homogeneous Poisson process.*
   (b) *Generate an sample* u *from a uniform distribution.*
   (c) *Retain the locations for which* $u \leq p$.

We have implemented exact simulation of stationary Gaussian random fields on a $n_1 \times n_2 \times n_3$ grid, based on circulant embedding of the covariance function (see *e.g.* Chan *et al.*, 1997) as follows.

Let $\bar{n} = \prod_{k=1}^{3} n_k$. If we write Z as an $\bar{n}$-vector, its covariance matrix is a $\bar{n} \times \bar{n}$ symmetric non-negative definite block Toeplitz matrix. It can be embedded into a $\bar{m} \times \bar{m}$-symmetric block circulant matrix $\mathbf{C}$ with $\bar{m} = \prod_{k=1}^{3} m_k$ and $m_k \geq 2(n_k - 1)$ and we have $\mathbf{C} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^*$, where $\mathbf{\Lambda}$ is a diagonal matrix with eigenvalues of $\mathbf{C}$, $\mathbf{Q}$ is an unitary matrix with columns being eigenvectors of $\mathbf{C}$ and $\mathbf{Q}^*$ is the conjugate transpose of $\mathbf{Q}$. By construction $\mathbf{C}$ is a covariance matrix of a stationary process, say $\tilde{Z}$. Because (i) eigenvalues of $\mathbf{C}$ is the Fast Fourier Transform (FFT) of any row of $\mathbf{C}$, (ii) eigenvectors of $\mathbf{C}$ are independent on $\mathbf{C}$ and (iii) $\mathbf{Qz}$ is the FFT of $\mathbf{z}$, then if $\mathbf{C}$ is non-negative definite $\tilde{Z}$ can be defined by $\tilde{Z} \equiv \mathbf{Q}\mathbf{\Lambda}^{1/2}\mathbf{Q}^* X$, $X \sim \mathcal{N}(0, \mathbf{I})$.

Thus, the algorithm to generate Z is :

1. Compute the smallest embedding length $m_k = 2^{g_k} \geq 2(n_k - 1)$.
2. Define the first row of the circulant matrix $\mathbf{C}$.
3. Take the FFT of the first row of $\mathbf{C}$ to get the vector of eigenvalues of $\mathbf{C}$, say λ.
4. (a) If $\min_{l=1,\ldots,\bar{m}} \lambda_l < 0$, then increase all g_k by 1, set $m_k = 2^{g_k}$, and return to step 2.
   (b) Otherwise, define the diagonal matrix $\mathbf{\Lambda} = \mathrm{diag}(\lambda)$.
5. Generate $X \sim \mathcal{N}(0, \mathbf{I}_{\bar{m}})$.
6. Define $\tilde{X}$ as the FFT of X.
7. Define $\mathbf{z}$ as $\mathbf{\Lambda}^{1/2}\tilde{X}$.
8. Define $\tilde{Z}$ as the FFT of $\mathbf{z}$.
9. Consider the real part of $\tilde{Z}$ and extract $n_1 \times n_2 \times n_3$ consecutive elements to form the realization Z.

We have implemented separable, stationary covariance functions of the form

$$c(h,t) = c_s(h)c_t(t), h \in S, t \in T, \tag{1}$$

where $c_s(h)$ and $c_t(t)$ are purely spatial and purely temporal covariance functions, respectively. For convenience, we considered isotropic models, $c_s(h) = c(\|h\|)$ or $c_t(t) = c(|t|)$, including the Matérn class, the Cauchy class and the wave class (see Table 1). The Matérn covariance is defined in terms of the modified Bessel function K_ν.

We have also implemented non-separable covariance functions of the form

$$c(h,t) = \psi(t)^{-\alpha_6}\phi\left(\frac{h}{\psi(t)}\right), \quad \text{Gneiting } \textit{et al.} \text{ (2005)}, \tag{2}$$

where $\phi(r), r \geq 0$ is a completely monotone function and $\psi(r), r \geq 0$ is a positive function with a completely monotone derivative. This model depends on 6 parameters α_1 to α_6. The function ϕ defines:

Class	Functional form
Exponential	$c(r) = \sigma^2 \exp(-r)$, $\sigma \geq 0$
Stable	$c(r) = \sigma^2 \exp(-r^\alpha)$, $\alpha \in [0, 2]$, $\sigma \geq 0$
Matérn	$c(r) = \sigma^2 \frac{(\alpha r)^\nu}{2^{\nu-1}\Gamma(\nu)} K_\nu(\alpha r)$, $\nu > 0$, $\alpha > 0$, $\sigma \geq 0$
Cauchy	$c(r) = \sigma^2 (1 + r^2)^{-\alpha}$, $\alpha > 0$, $\sigma \geq 0$
Wave	$c(r) = \sigma^2 \frac{\sin(r)}{r}$ if $r > 0$, $c(0) = 1$, $\sigma \geq 0$

Table 1: *Some classes of isotropic covariance functions.*

- the stable model $\phi(r) = \exp(-r^{\alpha_1})$, if $\alpha_2 = 1$,
- the cauchy model $\phi(r) = (1 + r^2)^{-\alpha_1}$, if $\alpha_2 = 2$.

The function ψ satisfies:

- $\psi^2(r) = (r^{\alpha_3} + 1)^{\alpha_4}$, if $\alpha_5 = 1$,
- $\psi^2(r) = (\alpha_4^{-1} r^{\alpha_3} + 1)/(r^{\alpha_3} + 1)$, if $\alpha_5 = 2$,
- $\psi^2(r) = -\log(r^{\alpha_3} + 1/\alpha_4)/\log \alpha_4$, if $\alpha_5 = 3$

The range of the parameters describing the model (2) are given in Table 2.

α_1	α_2	α_3	α_4	α_5	α_6
$[0, 2]$	1	$(0, 2]$	$(0, 1]$	1	$[2, \infty)$
$(0, \infty)$	2	$(0, 2]$	$(0, 1]$	1	$[2, \infty)$
$[0, 2]$	1	$(0, 2]$	$(0, 1]$	2	$[2, \infty)$
$(0, \infty)$	2	$(0, 2]$	$(0, 1]$	2	$[2, \infty)$
$[0, 2]$	1	$(0, 2]$	$(0, 1]$	3	$[2, \infty)$
$(0, \infty)$	2	$(0, 2]$	$(0, 1]$	3	$[2, \infty)$

Table 2: *Range of the parameters defining Equation (2).*

Examples

The function `rlgcp` generates realisations of a log-Gaussian Cox process. The covariance of the Gaussian process may be separable or not, as specified by the parameter `separable`. The parameter `model` is a vector of length 1 or 2 specifying the model(s) of covariance of the Gaussian random field. If `separable=TRUE` and `model` is of length 2, then the elements of `model` define the spatial and temporal covariances, respectively. When `separable=TRUE` and `model` is of length 1, the spatial and temporal covariances belongs to the same class of covariances, choice for which are "matern", "exponential", "stable", "cauchy" and "wave".

When `separable=FALSE`, `model` must be of length 1 and is either "gneiting" or "cesare". In all cases, parameters of the covariance models are defined in the vector `param` and the mean and variance of the Gaussian process are specified through the parameters `mean.grf` and `var.grf`.

The thinning algorithm used to generate the space-time pattern depends on the space-time intensity $\Lambda(x, y, t)$ which is a evaluated on a `nx`×`ny`×`nt` grid. The larger the grid size, the slower

are the simulations. Simulation time is also longer when the parameter `exact` is `TRUE`, providing an exact simulation rather than an approximation.

The sequence of commands

```
> lgcp1 <- rlgcp(npoints=200, nx=50, ny=50, nt=50, separable=FALSE,
  model="gneiting", param=c(1,1,1,1,1,2), var.grf=1, mean.grf=0)
> N <- lgcp1$Lambda[,,1]
> for(j in 2:(dim(lgcp1$Lambda)[3])){N <- N+lgcp1$Lambda[,,j]}
> image(N,col=grey((1000:1)/1000));box()
> animation(lgcp1$xyt, cex=0.8, runtime=10, add=TRUE, prevalent="orange")

> lgcp2 <- rlgcp(npoints=200, nx=50, ny=50, nt=50, separable=TRUE,
 model="exponential",param=c(1,1,1,1,1,2), var.grf=2, mean.grf=-0.5*2)
> N <- lgcp2$Lambda[,,1]
> for(j in 2:(dim(lgcp2$Lambda)[3])){N <- N+lgcp2$Lambda[,,j]}
> image(N,col=grey((1000:1)/1000));box()
> animation(lgcp2$xyt, cex=0.8, runtime=10, add=TRUE, prevalent="orange")
```

generates a realisation of each of two log-Gaussian Cox processes, one with a non-seperable and one with a separable covariance structure, and displays the realisations superimposed on a grey-scale image of the spatial intensity.

References

[1] A. Baddeley, J. Møller and R. Waagepetersen 2000. Non- and semi-parametric estimation of interaction in inhomogeneous point patterns. *Statistica Neerlandica*, **54**, 329–350.

[2] A. Baddeley and R. Turner 2005. spatstat: An R Package for Analyzing Spatial Point Patterns. *Journal of Statistical Software*, **12**, Issue 6.

[3] N.G. Becker 1989. *Analysis of Infectious Disease Data*. Boca Raton: Chpman and Hall/CRC

[4] M. Berman and P. Diggle 1989. Estimating weighted integrals of the second-order intensity of a spatial point process, *Journal of the Royal Statistical Society*, B, **51**, 81–92.

[5] G. Chan, A. Wood 1997. An algorithm for simulating stationary Gaussian random fields. *Applied Statistics, Algorithm Section*, **46**, 171–181.

[6] N. Cressie 1991. *Statistics for spatial data*, New-York, Wiley.

[7] D. Cox 1955. Some statistical methods related with series of events (with discussion). *Journal of the Royal Statistical Society*, B, **17**, 129–164.

[8] D. Cox and V. Isham 1980. *Point Processes*, Chapman and Hall, London.

[9] D. Daley and D. Vere-Jones 2003. *An Introduction to the Theory of Point Processes*, Second Edition, Springer, New York.

[10] P. Diggle 1985. A kernel method for smoothing point process data. *Applied Statistics*, **34**, 138–147.

[11] P. Diggle 2003. *Statistical analysis of spatial point patterns*, second edition. Arnold.

[12] P. Diggle 2006. Spatio-temporal point processes: methods and applications, In *Semstat2004*, B. Finkenstadt, L. Held and V. Isham (eds), 1–45, London: CRC Press.

[13] P. Diggle and E. Gabriel 2010. *Spatio-temporal point processes*, in Handbook of Spatial Statistics, Chapman and Hall/CRC Handbooks of Modern Statistical Methods, pp 449–461.

[14] E. Gabriel and P. Diggle 2009. Second-order analysis of inhomogeneous spatio-temporal point process data. *Statistica Neerlandica*, **63**, 43–51.

[15] J. Møller and R. Waagepetersen 2003. *Statistical inference and simulation for spatial point processes*, Monographs on Statistics an Applied Probability 100. Chapman & Hall.

[16] J. Møller and M. Ghorbani. *Second-order analysis of structured inhomogeneous spatio-temporal point processes.* Research Report R-2010-11, Department of Mathematical Sciences, Aalborg University

[17] J. Neyman and E. Scott 1958. Statistical approach to problems of cosmology (with discussion). *Journal of the Royal Statistical Society*, B, **20**, 1–43.

[18] B. Rowlingson and P. Diggle 1993. Splancs: Spatial Point Pattern Analysis Code in S-Plus. *Computers and Geosciences*, **19**, 627–655.

[19] B. Silverman 1986. Density Estimation. London: Chapman and Hall.

[20] M. Tanemura 1979. On random complete packing by discs. *Annals of the Institute of Statistical Mathematics*, **31**, 351–365.

[21] J. Zhuang, Y. Ogata and D. Vere-Jones 2002. Stochastic declustering of space-time earthquake occurrences. *Journal of the American Statistical Association*, **97**, 369–380