The STPP package

Peter J Diggle

School of Health and Medicine, Lancaster University

and

Department of Epidemiology and Population Health, University of Liverpool

Overview

Package developed by Edith Gabriel (Avignon) with support from Barry Rowlingson and Peter Diggle (Lancaster)

- **static and dynamic plotting**
- **second-order summary statistics**
- **simulation of spatio-temporal point processes**

Gabriel E., Rowlingson B., Diggle P. (2013) stpp: An R package for plotting, simulating and analysing Spatio-Temporal Point Patterns. *Journal of Statistical Software*, **53, 1–29.**

Static plotting

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

Plot function accepts a number of optional arguments

```
?plot.stpp
plot(fmd, s.region=northcumbria,
         mark=TRUE,mark.cexmin=0.2,mark.cexmax=1)
```

Dynamic plotting

```
animation(fmd, runtime = 10, cex = 0.5, s.region = northcumbria)
?animation
animation(fmd, runtime = 30, cex = 0.5, s.region = northcumbria,
   incident="red",prevalent="gray")
```

- **run-time argument is approximate**
- **more sophisticated animations available elsewhere (examples at `www.lancs.ac.uk/staff/diggle`)**

Dynamic plotting (2)

- can add animation to an existing plot (useful for context)

```
library(splancs)
image(xg,yg,zg,col=terrain.colors(10))
polymap(northcumbria,lwd=3,add=TRUE)
animation(fmd, runtime = 30, cex = 0.5, s.region=northcumbria,
   incident="red",prevalent="blue",add=TRUE)
```

- can also examine time-slices

```
?stan
stan(fmd, bgpoly = northcumbria, bgframe = FALSE)
```

Second-moment properties

```
#
# estimate marginal spatial and temporal intensities
#
library(KernSmooth)
FMD <- as.3dpoints(fmd[, 1] / 1000, fmd[, 2] / 1000, fmd[,3])
Northcumbria <- northcumbria / 1000
Mt <- density(FMD[ ,3], n = 1000)
mut <- Mt$y[findInterval(FMD[ ,3], Mt$x)] * dim(FMD)[1]
h <- mse2d(as.points(FMD[,1:2]), Northcumbria,
         nsmse = 50,range = 4)
h <- h$h[which.min(h$mse)]
Ms <- kernel2d(as.points(FMD[ ,1:2]), Northcumbria, h = h,
         nx = 5000,ny = 5000)
```

```
#
# estimate inhomogeneous K-function
#
atx <- findInterval(x = FMD[ ,1], vec = Ms$x)
aty <- findInterval(x = FMD[ ,2], vec = Ms$y)
mhat <- NULL
for (i in 1:length(atx)) mhat <- c(mhat, Ms$z[atx[i],aty[i]])
u <- seq(0, 10, by = 1)
v <- seq(0, 15, by = 1)
?STIKhat
stik <- STIKhat(xyt = FMD, s.region = Northcumbria,
         t.region = c(1, 200),lambda = mhat * mut / dim(FMD)[1],
         dist = u,times = v, infectious = TRUE)
plotK(stik)
```

```
#
# estimate pair correlation function
#
?PCFhat
g <- PCFhat(xyt = FMD, lambda = mhat * mut / dim(FMD)[1],
         dist = 1:20,times = 1:20, s.region = Northcumbria,
         t.region = c(1,200))
plotPCF(g)
```

Exercise. Experiment with the AEGISS dataset:

- **go to** `http://www.lancs.ac.uk/staff/diggle`
- **clink on** **spatial point pattern data-sets**
- **download the three AEGISS data-files**

Simulation: homogeneous Poisson process

```
data("northcumbria")
set.seed(6713)
hpp1 <- rpp(lambda = 1, nsim = 2,npoints=500)
hpp<-as.3dpoints(hpp1$xyt[[1]])
plot(hpp,mark=TRUE,mark.cexmin=0.2,mark.cexmax=1,pch=19)
times<-hpp[,3]
plot(sort(times),1:length(times),pch=19,cex=0.5)
```

Simulation: inhomogeneous Poisson process

```
set.seed(91772)
n.factor<-1500
lambda1 <- function(x, y, t, a){a*exp(-2.5*y)*exp(-1.5*t)}
ipp1 <- rpp(lambda = lambda1, npoints = 250,
   a = n.factor/((1-exp(-2.5))*(1-exp(-1.5))))
ipp<-as.3dpoints(ipp1$xyt)
dim(ipp)
#
# adjust value of n.factor to generate more,or fewer, points
#
plot(ipp,mark=TRUE,mark.cexmin=0.2,mark.cexmax=1,pch=19)
times<-ipp[,3]
plot(sort(times),1:length(times),pch=19,cex=0.5)
```

Simulation: simple inhibition

```
set.seed(81325)
inh1 <- rinter(npoints = 200, thetas = 0, deltas = 0.025,
         thetat = 0,deltat = 0.001, inhibition = TRUE)
inh<-as.3dpoints(inh1$xyt)
animation(inh,runtime = 30, cex =1.3,incident="red",
         prevalent="pink3")
#
# increase inhibtion distance
#
set.seed(81325)
inh2 <- rinter(npoints = 200, thetas = 0, deltas = 0.05,
         thetat = 0, deltat = 0.001, inhibition = TRUE)
inh<-as.3dpoints(inh2$xyt)
animation(inh,runtime = 30, cex =1.3,incident="red",
         prevalent="pink3")
```

Simulation: contagion

```
data(northcumbria)
cont1 = rinter(npoints=250,nsim=2,s.region=northcumbria,
         t.region=c(1,200),thetas=0,deltas=5000,
         thetat=0,deltat=10,recent=1,inhibition=FALSE)
cont<-as.3dpoints(cont1$xyt[[1]])
animation(cont,runtime=20,cex=0.5,s.region=northcumbria)
cont<-as.3dpoints(cont1$xyt[[2]])
animation(cont,runtime=20,cex=0.5,s.region=northcumbria)
```

Simulation: writing your own models

```
?rinter
hs<-function(d,theta,delta) {
   result<-theta
   if (d<delta) result<-result+(1-theta)*(1-d/delta)
   result
   }

set.seed(556137)
fmdsim1= rinter(npoints=250,hs=hs,s.region=northcumbria,
         t.region=c(1,200),thetas=0.05,deltas=10000,
         thetat=0, deltat=1, recent=1, inhibition=FALSE)
sim<-as.3dpoints(fmdsim1$xyt)
animation(sim, runtime = 30, cex = 0.5, s.region = northcumbria)
```

Warning. Some of the simulation functions are very slow