Statistics in Environmental Research (BUC Workshop Series) IV
Problem sheet 1

Website: `http://www.stat.ubc.ca/~gavin/STEPIBookNewStyle/course_bath.html`

Spatial modelling with INLA - Part 1

The aims of this practical are to get a general understanding of

- how to fit simple spatial point process models
- including covariates in a spatial model and model choice
- using constructed covariates

1. Preliminary things:

   ```
   library(spatstat)
   library(mvtnorm)
   library(lattice)
   library(mgcv)
   library(pixmap)
   library(numDeriv)
   library(fields)
   library(INLA)
   ```

   Specify the working directory (the path will be different on your machine; *change accordingly*): Enter folder `/home/inla_workshop/day 2/data`

   Also source a file with functions that you need later:

   `source(/home/inla_workshop/day 2/data/functions.r)`

2. A simple point process model

   We initially fit a simple model to a plant data set from Australia. The structure of the model forms the basic structure of many other more complicated models. Hence all the other models may be regarded as generalisations of this simple example.

   The original data set comprises the locations of 67 different species collected in 22m x 22m plot. Here we focus on one of the plant species, *Andersonia heterophylla*, an erect or ascending shrub, 0.1 - 0.5 m in height, grows in sandy, very nutrient poor soils.

A plot of the plants reveals that the pattern is inhomogeneous. This might have been caused by varying soil conditions or by how the species is spreading. Clearly, if covariate data were available we would want to include these in a model.

However, here no covariates are available that could be explicitly included in a model to explain this inhomogeneity (e.g. soil conditions). We therefore fit a model that takes into account the spatial large scale variability of the pattern, assuming that this is the result of an underlying environmental trend even in the absence of data on this. The approach we take here fits a smooth surface to the data that reflects this trend without making any assumptions on the parametric form of the trend.

(a) Reading in and gridding the data

We read in the data as:

```
paul<- read.delim("paul.txt")
# type 5 is Andersonia heterophylla
data<-paul[paul$type=="5",]
x=data$x/10
y=data$y/10
```

We transform the data into a point pattern object (using several commands from the library `spatstat`, for details check the library help files). Ignore the warning about duplicated points.

```
x.area=22
x.win=owin(c(0, x.area),c(0, x.area))

data.pp=ppp(x,y,window=x.win)
plot(data.pp, main= " Andersonia heterophylla")
```

We now need to transform the data, i.e. construct a grid with 30 x 30 cells

```
nrow=30
ncol=nrow
x.grid=quadrats(x.win,ncol,nrow)
```

and count the number of points in each grid cell; note that this will be our response variable.

```
count.grid=quadratcount(data.pp, tess=x.grid)
plot(count.grid)
```

(b) Running a first model

We have to transform the grid of counts into a vector (and we now use the notation from the slides for the response variable):

```
Y = as.vector(count.grid)
```

The number of grid cells

```
n = ncol*nrow
```

And calculate the area of the grid cells:

```
cell.area<-x.area^2/n
E<-rep(cell.area, n)
```

INLA requires separate indices for each of the spatial effect and the error term.

```
I = 1:n
J = 1:n
```

We have to specify a prior for the spatial effect

```
prior.spat=c(1,0.00005) #Default!
hyper.spat=list(prec=list(param=prior.spat))
```

We can no specify the model formula

```
formula = Y ~ 1+f(I, model="rw2d", nrow=nrow, ncol=ncol,
      hyper=hyper.spat)+f(J, model="iid")
```

and run the model (this should take only a few seconds at most)

```
result=inla(formula,data=data.frame(Y,I,J),
family="poisson",E=E, verbose=TRUE,
control.compute=list(dic=TRUE))
```

We can look at a summary and a plot of the results

```
summary(result)
plot(result)
```

The estimated intercept

```
result$summary.fixed
```

the (posterior mean of the) spatial effect

```
f.spat=result$summary.random$I$mean
```

plot it

```
im.matrix(matrix(f.spat, ncol, nrow))
```

the error term

```
f.unstruct=result$summary.random$J$mean
im.matrix(matrix(f.unstruct, nrow, ncol))
```

the resulting fit (compare with original pattern) scaled up by the area.

```
fit=E*result$summary.fitted.values$mean
im.matrix(matrix(fit, nrow, ncol))
```

Exercise

- Fit a model without the spatial or the error term to the same data and compare using the DIC.
- Fit similar models to another spatial pattern from the data set in `paul.txt` and discuss the results.
- Investigate the effect of changing the prior for the spatial effect on the smoothness of the estimated effect. Do this by changing the shape and scale parameters of the prior.

3. A model with covariates We now model a rainforest data set. Some extraordinarily detailed multi-species maps are being collected in tropical forests as part of an international effort to gain greater understanding of these ecosystems. These data comprise the locations of all trees with diameters at breast height (dbh) 1 cm or greater, a measure of the size of the trees (dbh), and the species identity of the trees. The data usually amount to several hundred thousand trees in large (25 ha or 50 ha) plots that have not been subject to any sustained disturbance such as logging. The spatial distribution of these trees is likely to be determined by both spatially varying environmental conditions and local dispersal. In our example we will only look at species association with environmental covariates, our current paper however discusses taking into account local dispersal as well.

   The model is fitted to a data set from a 50 ha forest dynamics plot on Barro Colorado Island (BCI) and consists of the spatial pattern formed by a total of 3887 trees of the species *Beilschmiedia pendula Lauraceae*. We fit a similar model as before but we include two (normalised) covariates, elevation and gradient, for each grid cell.

   Again we start by reading in the data and gridding it; read in the data it is a big data set and takes a while to load up.

```
nrow=50
ncol=100
n<-nrow*ncol

BCIData <- read.delim("Routput.txt")
species<-BCIData[BCIData$sp=="beilpe",]
spec.string="Beilschmiedia p.L."

x<-species$gx[is.na(species$gx)==F]
y<-species$gy[is.na(species$gy)==F]
x.win=owin(c(0,1000),c(0,500))
data.pp<-ppp(x,y, window=x.win)
plot(data.pp, main=spec.string)
Area=rep(1,nrow*ncol)
```

   Generate the count grid

```
x.grid=quadrats(x.win,ncol,nrow)
count.grid=quadratcount(data.pp, tess=x.grid)
```

Find midpoints of each cell

```
mid.p=midpoints.func(x.win,nrow,ncol)
```

We also need to read in the covariate data, elevation and gradient. This is done in the file `read_cov.txt` (check details if you are interested but it depends on the format of the data and is hence not that interesting here):

```
source("read_cov.txt")
```

Plot the centred elevation (as stored in `elev.mean`)

```
im.matrix(matrix(elev.mean,nrow,ncol))
```

Plot the centred gradient (as stored in `grad.mean`)

```
dev.new()
im.matrix(matrix(grad.mean,nrow,ncol))
```

The response

```
Y=as.vector(count.grid)
```

the explanatory variables

```
Z1=as.vector(elev.mean)
Z2=as.vector(grad.mean)
```

the grid cell index for the spatial effects

```
I = 1:n
J = 1:n
```

Use the log-gamma prior for the precision

```
param.spat=list(prec=list(prior="pc.prec",param=c(u=1,alpha=0.01)))
```

And we fit a model with the two covariates, a spatially structured effect and an error term, as before.

```
formula = Y ~ 1 + Z1 + Z2 + f(I, model="rw2d", nrow=nrow,
        ncol=ncol, hyper=param.spat, scale.model=T)+ f(J, model="iid")

result.rain = inla(formula, data=data.frame(Y,Z1,Z2, I, J), family = "poisson",
        E=Area, verbose = TRUE , control.compute=list(dic=TRUE))
```

This might take a few minutes (96 seconds when I last ran it). If you want to use a quick and dirty (i.e. slightly less exact) method use

```
result.rain2 = inla(formula, data=data.frame(Y,Z1,Z2, I,J), family = "poisson",
        E=Area, verbose = TRUE , control.compute=list(dic=TRUE),
        control.inla = list(strategy = "gaussian", int.strategy = "eb"))


# Posterior estimates
# Compare result.rain and result.rain2

summary(result.rain)

# The structured and unstructured spatial effects
im.matrix((matrix(result.rain$summary.random$I$mean, nrow, ncol)))
dev.new()
im.matrix((matrix(result.rain$summary.random$J$mean, nrow, ncol)))
```

Exercise

- Investigate the effect of changing the prior (by adjusting `u`) for the spatial effect on the smoothness of the estimated effect.
- The matrix `cov.all` contains (log-transformed) soil covariates along with the two topography covariates we used before. `cov.string` tells you the names of the covariates. Take a look at these (column no) using `im.matrix(matrix(cov.all[,no],nrow,ncol))`.
- Find a suitable model for the given species.

Spatial modelling with INLA - Part 2

In this practical we will look at some more complex spatial models.

The aims of this practical are to get a general understanding of how to fit

- joint models/ models with several likelihoods
- marked point processes (qualitatively marked point patterns, quantitatively marked point patterns)
- models of replicated point patterns

1. Preliminary things (same as before):

   ```
   library(spatstat)
   library(mvtnorm)
   library(lattice)
   library(mgcv)
   library(pixmap)
   library(numDeriv)
   library(fields)
   library(INLA)
   ```

   specify the working directory (the path will be different on your machine; *change accordingly*):

   ```
   setwd("/home/inla\_workshop/day 2/data")
   ```

   read in some functions that we will need later

   ```
   source("functions.r")
   ```

2. A joint model of two species

   The first model we fit is a model of two rainforest species (*Protium tenuifolium* and *Protium panamense*) where we model the species jointly. This means we have two separate likelihoods. In this case these are both Poisson likelihoods; we will see in the next example that it is also possible to have different likelihoods and more than two likelihoods within the same model.

   We consider a joint model here with a shared spatial effect in order to differentiate between co-occurrence as the result of a shared environmental preference that cannot be accounted for by the covariates and local spatial interaction. In a simpler option would be to simply model a single species given the other species. However, if this is done it is not easily possible to distinguish the two potential explanations for the species co-occurrence.

(a) Reading in and gridding the data

We read in the data as:

```
nrow=50
ncol=100
n=nrow*ncol

BCIData <- read.delim("Routput.txt")

species1<-BCIData[BCIData$sp=="protte",]
spec.string1="Protium tenuifolium"

species2<-BCIData[BCIData$sp=="protpa",]
spec.string2="Protium panamense"
```

We transform the data into two point pattern objects before

```
species<-species1
x<-species$gx[is.na(species$gx)==F]
y<-species$gy[is.na(species$gy)==F]
x.win=owin(c(0,1000),c(0,500))
cell.area=100
data1.pp<-ppp(x,y, window=x.win)
E1<-rep(cell.area, n)

species<-species2
x<-species$gx[is.na(species$gx)==F]
y<-species$gy[is.na(species$gy)==F]
x.win=owin(c(0,1000),c(0,500))
data2.pp<-ppp(x,y, window=x.win)
E2<-rep(cell.area, n)
E<-c(E1,E2)
```

and count the number of points in each grid cell; note that these will be our two response variables.

```
x.grid=quadrats(x.win,ncol,nrow)
count1.grid=quadratcount(data1.pp, tess=x.grid)
count2.grid=quadratcount(data2.pp, tess=x.grid)
```

The midpoints

```
mid.p=midpoints.func(x.win,nrow,ncol)
```

We read in the soil covariate data along with elevation and gradient.

```
source("read_cov.txt")
```

In this model we work with the principal components of the covariates (just for simplicity)

```
pc=princomp(cov.all,cor=T)
pc.log=pc$scores[,1:3]
```

(b) Running a joint model

We have to generate two vectors of response variables for the joint model with NAs in the appropriate places. Anything related to species 1 has values in the first n entries and NAs elsewhere and anything related to species 2 has NAs in the first n entries and values in $(n+1,,2n)$

```
nothing = rep(NA, n)

y = as.vector(count1.grid)
yNA = as.vector(c(count1.grid, nothing))
z = as.vector(count2.grid)
zNA = as.vector(c(nothing, count2.grid))
```

and put them in a matrix (B for both)

```
B= matrix(c(yNA,zNA), ncol=2)
```

We have to generate the index vectors for the spatial effects for each species; for species 1

```
i.spat1 = c(1:n, nothing)
j.error1 = c(1:n, nothing)
```

for species 2

```
i.spat2 = c(nothing,1:n)
j.error2 = c(nothing,1:n)
```

The covariates also have to be stored in stacked vectors

```
pc.1<-pc.log[,1]
pc.2<-pc.log[,2]
PC11<-c(pc.1, nothing)
PC21<-c(pc.2, nothing)
PC12<-c(nothing, pc.1)
PC22<-c(nothing, pc.2)
```

Since we want to estimate separate offsets for each of the species we have to generate a factor vector with two levels indicating the two outcome variables.

```
mu = as.factor(c(rep(1,n), rep(2,n)))
```

and the priors

```
param.cc=list(prec=list(param=c(15,0.005)))
param.spat=list(prec=list(param=c(80,10)))
```

The call to R-INLA to run the full model is specified as

```
data=list(B=B, i.spat1=i.spat1, i.spat2=i.spat2, j.error1=j.error1,
      j.error2=j.error2, E=E, PC11=PC11, PC21=PC21,
     PC12=PC12, PC22=PC22)

formula = B ~ mu -1 +
f(i.spat1, model="rw2d", nrow=nrow, ncol=ncol, hyper= param.spat)+
f(i.spat2, copy="i.spat1", fixed=F)+
PC11 + PC21 + PC12 +PC22
```

In the call to inla we now also have a vector of families. They are both the same here but this can of course vary. Also note that since we assume a joint spatial use `f(i.spat2, copy="i.spat1", fixed=F)` to make sure that the same spatial effect is used to explain large scale aggregation for both species.

```
result= inla(formula, data=data, family =c( "poisson","poisson"),
     E=E,verbose = TRUE ,
     control.compute=list(dic=TRUE,return.marginals=FALSE),
     control.inla = list(strategy = "gaussian", int.strategy = "eb"))
```

Note that to tell R-INLA to not calculate a joint offset -1 needs to be added to the formula.

Plotting the spatial effects- note that these are the same!

```
im.matrix((matrix(result$summary.random$i.spat1$mean, nrow, ncol)))
dev.new()
im.matrix((matrix(result$summary.random$i.spat2$mean, nrow, ncol)))
```

3. A joint model of marks and pattern

   The data have been collected in a study conducted at the Koala Conservation Centre on Phillip Island, near Melbourne, Australia. For each of 915 trees within a reserve enclosed by a koala-proof fence, information on the leaf chemistry and on the frequency of koala visits has been collected.

   The leaf chemistry is summarised in a measure of the palatability of the leaves. Palatability is assumed to depend on the intensity of the point pattern. In addition, "frequency marks" describe for each tree the diurnal tree use by individual koalas collected at monthly intervals between 1993 and March 2004. The frequency marks' are assumed to depend on the intensity of the point pattern as well as on the leaf marks.

   We fit a joint model to the pattern and the two different marks where we assume that they share a common spatial effect. We now have three different likelihoods and two different types of likelihoods.

   This model is fitted to a subset of the data only. This is only done to avoid spending too much time on formatting the data in this practical. The model can be easily fitted to the entire irregular area and we are happy to pass on the code.

(a) Read in the data stored in workspace `EucData.Rdata`

```
load("EucData.Rdata")
```

define the subwindow and subpatterns

```
sub.window<-owin(c(348140, 348250), c(5739140, 5739200))
sub.food.pp<-FOOD.ppp[sub.window]
sub.koala.pp<-KOALA.ppp[sub.window]
```

log the data so that they are close to normal

```
leave.marks<-log(sub.food.pp$marks)
koala.marks<-sub.koala.pp$marks

plot(unmark(data.ppp))
sub.pp<-data.ppp[sub.window]
points(sub.pp, col=3)
pp<-sub.pp #with leave marks
n.points<-length(pp$x)
```

a small data transformation

```
new.x<-pp$x- min( pp$x)
new.y<-pp$y- min( pp$y)
pp$x<-new.x
pp$y<-new.y
x.dim=c(0,106,0,60)
new.window<-owin(c(x.dim[1], x.dim[2]), c(x.dim[3], x.dim[4]))
pp$window<-new.window
plot(unmark(pp))

nrow<-20
ncol<-35

x.p=unmark(pp)
x.leave=x.p%mark%leave.marks
x.koala=x.p%mark%koala.marks
```

plot of subpattern

```
plot(matrix(c(x.koala$x,x.koala$y),nrow=x.koala$n,ncol=2),xlab="",
     ylab="",main="")
```

Grid, reduced window

```
x=x.p$x
y=x.p$y
```

```
eps=c((max(x)-min(x))/nrow/2,(max(y)-min(y))/ncol/2)
x.red=ppp(x,y,window=owin(c(min(x),max(x)),c(min(y),max(y))))
x.win=x.red$window
```

Midpoints and grid

```
x.x=matrix(rep(seq(min(x.red$x)+eps[1],max(x.red$x)-eps[1],length=nrow),ncol),
     ncol,nrow,byrow=T)
x.y=matrix(rep(seq(min(x.red$y)+eps[2],max(x.red$y)-eps[2],length=ncol),nrow),
     ncol,nrow)
x.ppp=ppp(x.x,x.y,window=x.win)
x.grid=quadrats(x.win,nrow,ncol)
```

Counts

```
x.count=quadratcount(x.red, tess=x.grid)
x.count=x.count[ncol:1,] # Oriented as x.p
```

The gridded point pattern

```
y.p=as.vector(x.count)
```

For the marks we need to find the mean mark in each cell. This is SLOW, but general

```
f.grid.mean=function(z,x.grid)
{
grid.mean.list=by(z,x.grid,FUN=function(z){mean(z$marks)})
grid.mean=matrix(as.numeric(grid.mean.list),ncol,nrow,byrow=T)
grid.mean=grid.mean[ncol:1,]
return(grid.mean)
}


leave.grid=f.grid.mean(x.leave,x.grid)
koala.grid=f.grid.mean(x.koala,x.grid)
leave.m<-as.vector(leave.grid)
koala.m<-as.vector(koala.grid)
```

(b) Fitting a model

As above

```
n=ncol*nrow
nothing<-rep(NA,n)
```

As above we need to define separate indices for the different levels of the model.

```
i1 = c(seq(1:n),nothing,nothing)
i2 = c(nothing, 1:n, nothing)
i3 = c(nothing, nothing, 1:n)
i = rep(1:n, 3)
```

```
j = c(1:n, nothing, nothing)
k = c(nothing, 1:n, nothing)
l = c(nothing, nothing, 1:n)
```

and separate offsets

```
mu = as.factor(c(rep(1,n), rep(2,n), rep(3,n)))
```

and separate response vectors and a response matrix.

```
response.pattern<-c(y.p, nothing, nothing)
response.leave<-c(nothing, leave.m, nothing)
response.koala<-c(nothing, nothing, koala.m)
response.matrix<-matrix(c(response.pattern, response.leave, response.koala),
     byrow=F, ncol=3)
y=response.matrix
```

since we assume that the leave chemistry impacts on the koala visits we also have a covariate that only impacts on the third outcome.

```
cov.1<-c(nothing, nothing, leave.grid)
```

fitting the model

```
cell.area<-(max(x.p$x)/nrow)*(max(x.p$y)/ncol)
E<-rep(cell.area, 3*n)

prior.a=10
prior.b=0.001

formula = y ~ mu -1 + cov.1 +
f(i1, model="rw2d", nrow=nrow,ncol=ncol, param=c(prior.a, prior.b))+
f(i2, copy="i1", fixed= FALSE, param=c(1,0.5)) +
f(i3, copy="i1", fixed=FALSE, param=c(1,0.5)) +
f(j, model="iid", initial = 6, fixed=TRUE) +
f(k, model="iid", initial = 6, fixed=TRUE) +
f(l, model="iid", initial = 6, fixed=TRUE)

data=list(y=y,i1=i1,i2=i2,i3=i3,j=j,k=k,l=l,cov.1=cov.1)
result = inla(formula, data=data, family = c("poisson","normal","poisson"),
E=E, verbose = TRUE , control.compute=list(dic=TRUE),
control.inla = list(strategy = "gaussian", int.strategy = "eb"),
control.fixed = list(prec.intercept = 0.01))

summary(result)
```

Plotting the estimated spatially structured effect and the spatially unstructured effect

```
im.matrix((matrix(result$summary.random$i1$mean,nrow,ncol)))
dev.new()
im.matrix((matrix(result$summary.random$j$mean,nrow,ncol)))
dev.new()
im.matrix((matrix(result$summary.random$k$mean,nrow,ncol)))
dev.new()
im.matrix((matrix(result$summary.random$l$mean,nrow,ncol)))
```