

Practical 3: Spatial spatio-temporal models

4 March, 2019

In this practical you will use **R** as well as **OpenBUGS** to carry out a spatial and a spatio-temporal small area disease risk analysis.

In particular, you are going to model yearly hospital admissions for respiratory conditions (ICD-10 codes: J00-J99 and R09.1) in the Greater Glasgow and Clyde health board, for the period 2007 - 2011.

Scotland is divided into 14 health boards, and here we focus on the Greater Glasgow and Clyde health board, which contains the city of Glasgow and has a population of around 1.2 million people, during the period 2007 to 2011. This health board is split into $N = 271$ intermediate geographies (IG), which are a key geography for the distribution of small-area statistics in Scotland and contain populations of between 2,468 and 9,517 people.

The data used in this practical are freely available from the R package **CARBayesST**. A subset of these data is in the file `RESP_DATA.csv`.

Before starting the practical

Organise your work:

- Create a separate subdirectory in your home directory to save your files created during this practical (e.g. “C:/SpatialAnalysis2019/Practicals/Practical3”).
- Copy all the files from the blackboard to your subdirectory (created just above).
- You will use the following R packages: `rgdal`, `sp`, `spdep`, `ggplot2`, `viridis`, `foreign`, `R2OpenBUGS`, `mcmcplots`, `reshape`.
- If not installed yet, then install the required packages using the function `install.packages()`
- Then, load needed libraries:

```
library(rgdal)
library(sp)
library(spdep)
library(ggplot2)
library(viridis)
library(foreign)
library(R2OpenBUGS)
library(mcmcplots)
library(reshape)
```

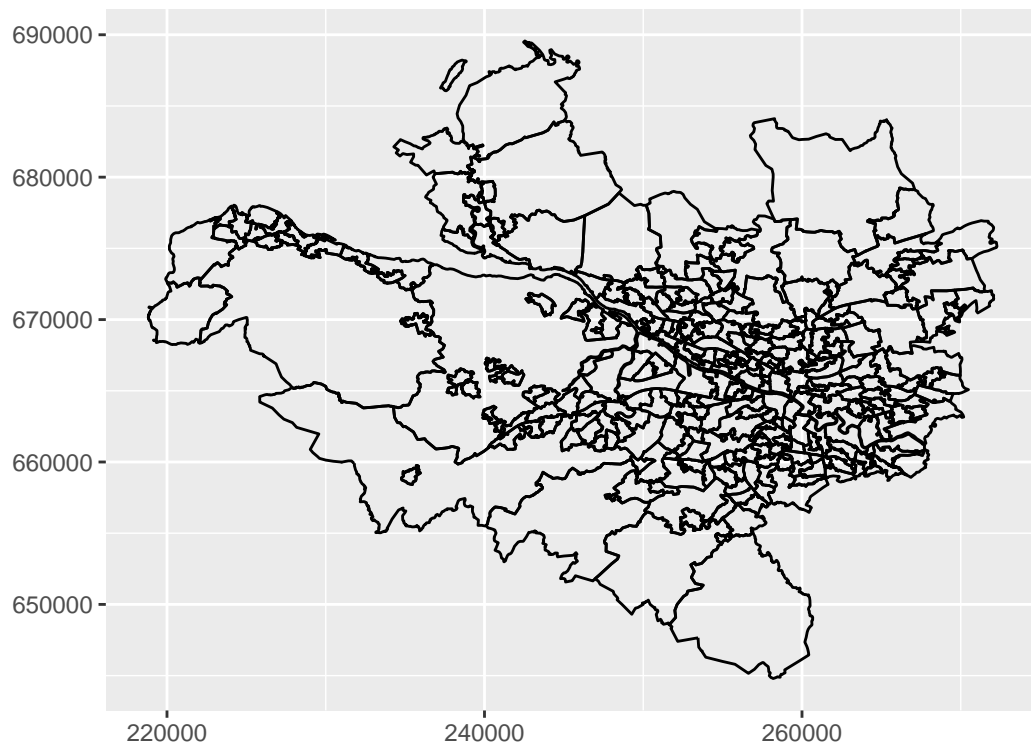
Data

- Import the `.csv` file with the data and call the `data.frame` object as *RESP_DATA*. Print the first rows of the data.
- Import the shape file *GGHB* using the function `readOGR` and call the object as *GGHB*.
- Then, examine the spatial object *GGHB*: the class and the names of the columns of the data frame (to see what it contains). Finally make a simple plot
- Now, plot it using `ggplot2` package. This requires the use of the `fortify` statement that returns a `data.frame` object.

```
GGHB.df <- fortify(GGHB)
```

```
## Regions defined for each Polygons
```

```
ggplot(data <- GGHB.df, aes(x = long, y = lat, group = group)) + geom_path() +  
  theme(axis.title.x = element_blank(),  
        axis.title.y = element_blank(),  
        axis.line = element_blank())
```



Spatial model

In order to use the same data structure for both the space-only model and later the space-time model, a new set of data is formed by aggregating both the observed and expected counts over

time. A Poisson-log linear model is then fitted, assuming a BYM model for the random effects. The code for the model is in the file *RESP_MODEL_S.txt*.

- Open the file *RESP_MODEL_S.txt* and add the statement in your code to compute the residual relative risks `exp(U[i] + V[i])` and call it as *residRR*.
- Define the neighbours and create the weight matrices. The `car.normal` function requires the following spatial information:

- a list of neighbours supplied as a vector called `adj`,
- a list of weights called `weights` pertaining to the list of neighbours,
- a vector called `num` that gives the number of neighbours for each area.

To do so, convert the polygons to a list of neighbours using the function `poly2nb`, then convert it in OpenBUGS format using the function `nb2WB`

```
shp_nb <- poly2nb(GGHB) # Convert the polygons to a list of neighbours
nbWB <- nb2WB(nb=shp_nb) # Convert this list to OpenBUGS format
names(nbWB)
```

```
## [1] "adj"      "weights" "num"
```

- Aggregate observed and expected cases over geographical areas (see Practical 1)

```
RESP_DATAagg <- aggregate(RESP_DATA[3:ncol(RESP_DATA)],
                           by=list(RESP_DATA$SP_ID), FUN="sum")
colnames(RESP_DATAagg) <- c("SP_ID", "observed", "expected")
```

- Compute the SMRs (here they are the standardized morbidity ratios)

```
RESP_DATAagg$SMR <- RESP_DATAagg$observed / RESP_DATAagg$expected
```

- Map the SMRs using `ggplot2` package. For the map use the following breakpoints [min,0.4], (0.4-0.6], (0.6-0.8], (0.8,1], (1,1.2], (1.2-1.4], (1.4-1.6], (1.6-max]. Color palettes used for this plot are obtained from the package `viridis`

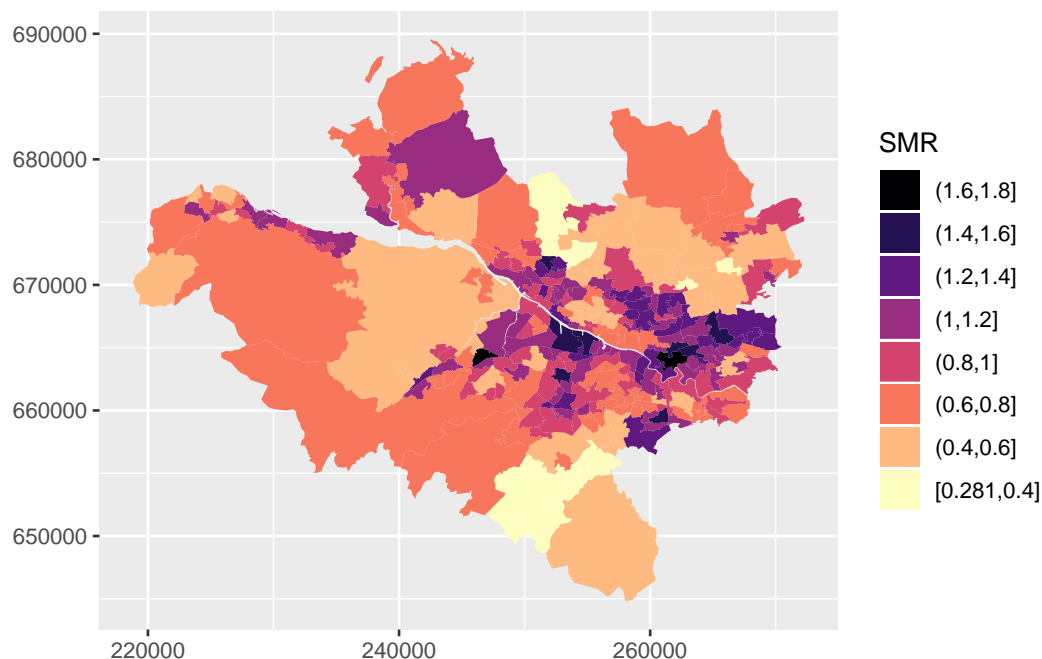
```
RESP_DATAagg$SMRcat <- cut(RESP_DATAagg$SMR,
                           breaks=c(min(RESP_DATAagg$SMR), 0.4, 0.6, 0.8, 1, 1.2, 1.4, 1.6, max(RESP_DATAagg$SMR)),
                           labels=c("min-0.4", "0.4-0.6", "0.6-0.8", "0.8-1", "1-1.2", "1.2-1.4", "1.4-1.6", "1.6-max"))

GGHB.dbf <- read.dbf("GGHB.dbf") # read.dbf is a function from package foreign
GGHB.dbf$id <- 0:270

map.smr <- merge(GGHB.dbf, RESP_DATAagg, by="SP_ID", all.x=TRUE)
```

```
map.smr2 <- merge(GGHB.df, map.smr, by="id", all.x=TRUE)

ggplot(data = map.smr2, aes(long, lat, group = group, fill = SMRcat)) +
  geom_polygon() +
  coord_equal() + labs(x = "", y = "", fill = "SMRcat") +
  scale_fill_viridis(
    option = "magma",
    name = "SMR",
    discrete = T,
    direction = -1,
    guide = guide_legend(
      title.position = 'top',
      reverse = T
    )
  )
```



- Fit the hierarchical Poisson log-linear model in **OpenBUGS** via **R**. Remember to follow all the usual steps:
 1. Prepare the data
 2. Set the initial values (2 chains). Remember that for the spatially structured random effects (U), we need initial values with a sum-to-zero constraint (suggestion: you could set U in chain 1 as $U=c(\text{rep}(0.1,134),\text{rep}(0.2,1),\text{rep}(-0.1,136))$, and in chain 2 as $U=c(\text{rep}(0.5,134),\text{rep}(1,1),\text{rep}(-0.5,136))$).
 3. Set the parameters to be monitored. Among them, remember to set up (i) the posterior residual relative risks for all the areas, and (ii) the posterior probability (PP) that the residual relative risks is > 1 for all the areas, as later on we are going to map these quantities.

4. Specify the MCMC setting. Here, we say we want to run two chains for 7,000 iterations, discarding the first 4,000 values, and thinning the remaining values by 5.
 5. Run the MCMC simulations calling OpenBUGS from R using the function `bugs()`. Denominate the bugs object as `modelResp.sim`
- Visualise the main diagnostic plots using `mcmcplot`.
 - Inspect the Rhat (the potential scale reduction factor) from the posterior summary statistics. Remember from Practical 1 that a Rhat close to 1.0 for all parameters indicates a good mixing (if we find values higher than the 1.1 threshold, we should increase the nb of iterations).

```
modelResp.sim$summary[, "Rhat"]
modelResp.sim$summary[2:272, "Rhat"] # e.g. Rhat for RR
```

- Obtain the posterior summary statistics (mean, sd, and 95% credible intervals) of the parameters of interest

```
modelResp.sim$summary[, c(1, 2, 3, 7)] # mean, sd, and 95% credible intervals
```

- Map the area level posterior residual RRs and the posterior probability (PP) that the residual RRs > 1 using `ggplot2` package

For the map of posterior residual RRs use the following breakpoints [min,0.4], (0.4-0.6], (0.6-0.8], (0.8,1], (1,1.2], (1.2-1.4], (1.4-1.6], (1.6-max].

For the map of the probabilities PP that the residual RRs > 1 use the following breakpoints [0,0.2], (0.2-0.8], (0.8-1].

```
resRR_PP <- data.frame(resRR=modelResp.sim$mean$residRR,
                       PP=modelResp.sim$mean$PP,
                       SP_ID=RESP_DATAagg[,1])
```

```
# breakpoints
summary(modelResp.sim$mean$residRR)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.3668  0.7727  1.0528  1.0649  1.3186  2.1774
```

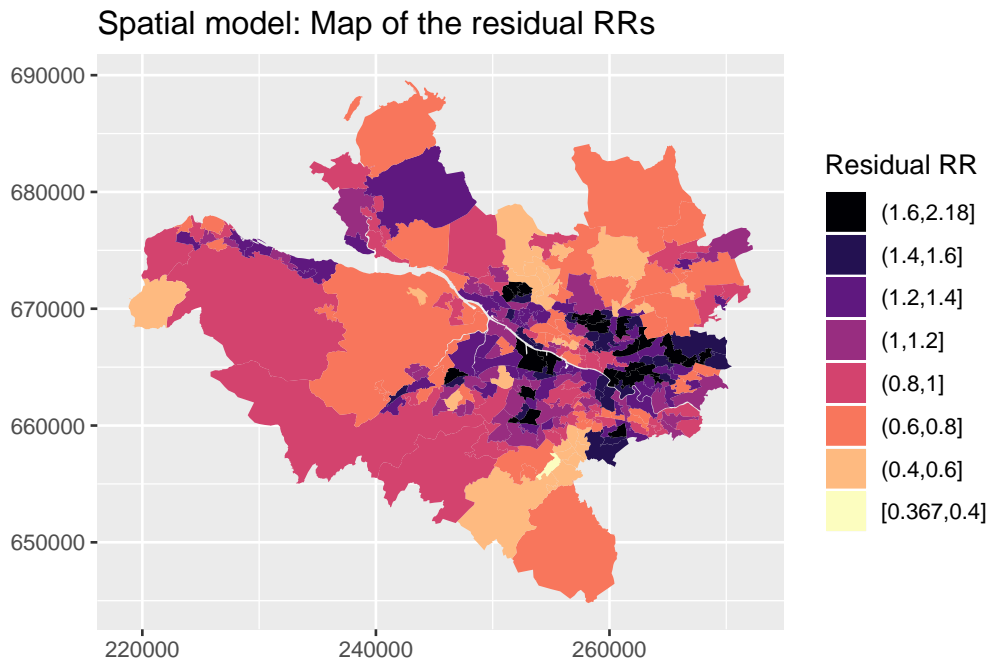
```
resRR_PP$resRRcat <- cut(resRR_PP$resRR, breaks=c(min(resRR_PP$resRR),
          0.4, 0.6, 0.8, 1, 1.2, 1.4, 1.6,
          max(resRR_PP$resRR)),include.lowest = T)

resRR_PP$PPcat <- cut(resRR_PP$PP, c(0, 0.2, 0.8, 1.00), include.lowest = TRUE)

map.RR.PP <- merge(GGHB.dbf, resRR_PP, by="SP_ID", all.x=TRUE)
```

```
map.RR.PP2 <- merge(GGHB.df, map.RR.PP, by="id", all.x=TRUE)

# Map posterior residual RRs
ggplot(data = map.RR.PP2, aes(long, lat, group = group, fill = resRRcat)) +
  geom_polygon() +
  ggtitle("Spatial model: Map of the residual RRs") +
  coord_equal() + labs(x = "", y = "", fill = "resRRcat") +
  scale_fill_viridis(
    option = "magma",
    name = "Residual RR",
    discrete = T,
    direction = -1,
    guide = guide_legend(
      title.position = 'top',
      reverse = T
    )
  ))
```

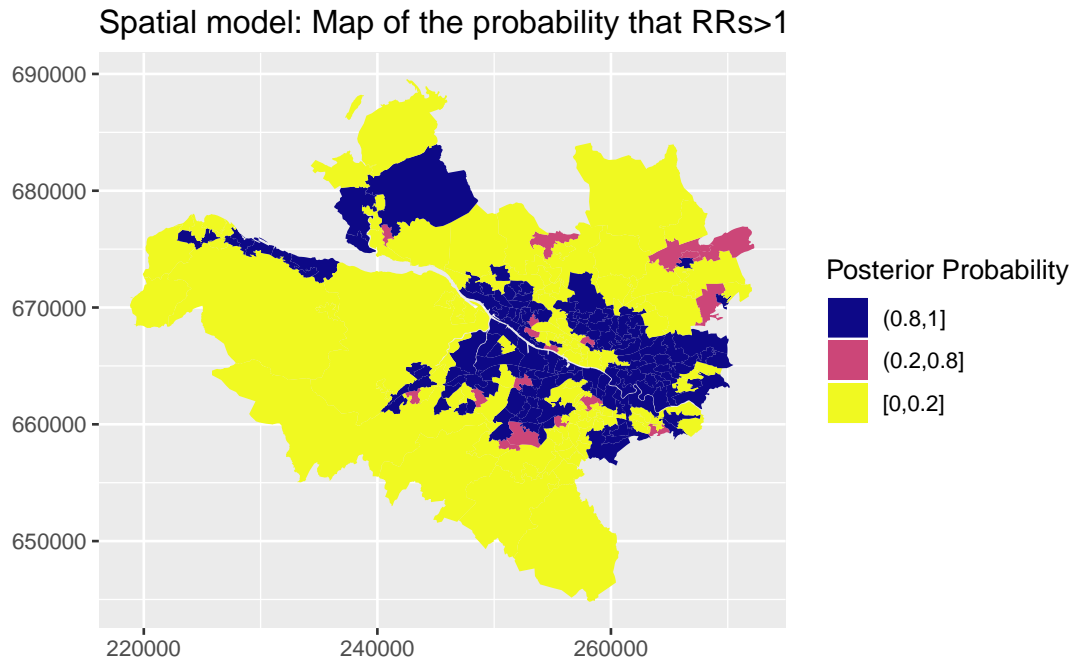


```
# Map posterior probability that the residual RRs are > 1
ggplot(data = map.RR.PP2, aes(long, lat, group = group, fill = PPcat)) +
  geom_polygon() +
  ggtitle("Spatial model: Map of the probability that RRs>1") +
  coord_equal() + labs(x = "", y = "", fill = "PPcat") +
  scale_fill_viridis(
    option = "plasma",
    name = "Posterior Probability",
    discrete = T,
```

```

direction = -1,
guide = guide_legend(
  title.position = 'top',
  reverse = T
))

```



Spatio-temporal model

Now, we extend the above analysis to a separable space-time model without interactions. For the temporal component, we use the RW(1) prior. To fit this model in **OpenBUGS**, we exploit the fact that the RW(1) prior can be represented as an intrinsic CAR prior where the neighbours for each time point t are the preceding ($t - 1$) and subsequent ($t + 1$) time points with special conditions for the first and last time points. Hence we can use the `car.normal` distribution in **OpenBUGS** to fit this model.

- Open the model *RESP_MODEL_ST.csv* and make sure you understand the specification of the weight matrix and the adjacency matrix for the RW(1) prior for the temporal random effects \mathbf{x}_i .
- Run the space-time model and carry out appropriate checks for convergence. In particular:
 - (a) Prepare the data, taking into account that now the data have not only a spatial but also a temporal dimension. To prepare the data to be used for this analysis in **OpenBUGS** via **R**, we need to reshape the original data to obtain a matrix of dimension 271 (number of areas) by 5 (number of years).

```

# prepare observed data
Observed <- RESP_DATA[,1:3]
Observed <- reshape(Observed, idvar = "SP_ID",
                    timevar = "year", direction = "wide")
Observed <- as.matrix(Observed[,c(2:6)])

# prepare expected data
Expected <- RESP_DATA[,c(1,2,4)]
Expected <- reshape(Expected, idvar = "SP_ID",
                    timevar = "year", direction = "wide")
Expected <- as.matrix(Expected[,c(2:6)])

dataST <- list("N" = 271, # nb of areas
              "T" = 5, # nb of years
              "O" = Observed, # observed nb of cases
              "E" = Expected, # expected nb of cases
              "adj" = nbWB$adj, "weights" = nbWB$weights,
              "num" = nbWB$num)

```

- Set the initial values (2 chains)
- Set the parameters to be monitored. Among them, remember to set up (i) the spatial residual RRs, and (ii) the temporal residual RRs, as later on we are going to plot these quantities.
- Specify the MCMC setting as in the spatial model.
- Run the MCMC simulations calling `OpenBUGS` from R. Denominate the bugs object as `modelRespST.sim`.
- Visualise the main diagnostic plots using `mcmcplot`.
- Map the spatial residual RRs (`residRR`) with `ggplot2` package using the following breakpoints [min,0.4], (0.4-0.6], (0.6-0.8], (0.8,1], (1,1.2], (1.2-1.4], (1.4-1.6], (1.6-max]
- Plot the time-series of the temporal residual RRs (`residRRtm`)

```

attach.bugs(modelRespST.sim)

# Preparing posterior estimates for the plot:
# median and 95% credible intervals of teporal residuals
Median.tm<-data.frame(apply(residRRtm, 2,
                           quantile, probs=0.50)) # Post. median
Low.tm<-data.frame(apply(residRRtm, 2,
                        quantile, probs=0.025)) # Lower bound
Up.tm<-data.frame(apply(residRRtm, 2,
                       quantile, probs=0.975)) # Upper bound

# combining posterior estimates in a data frame

```



```

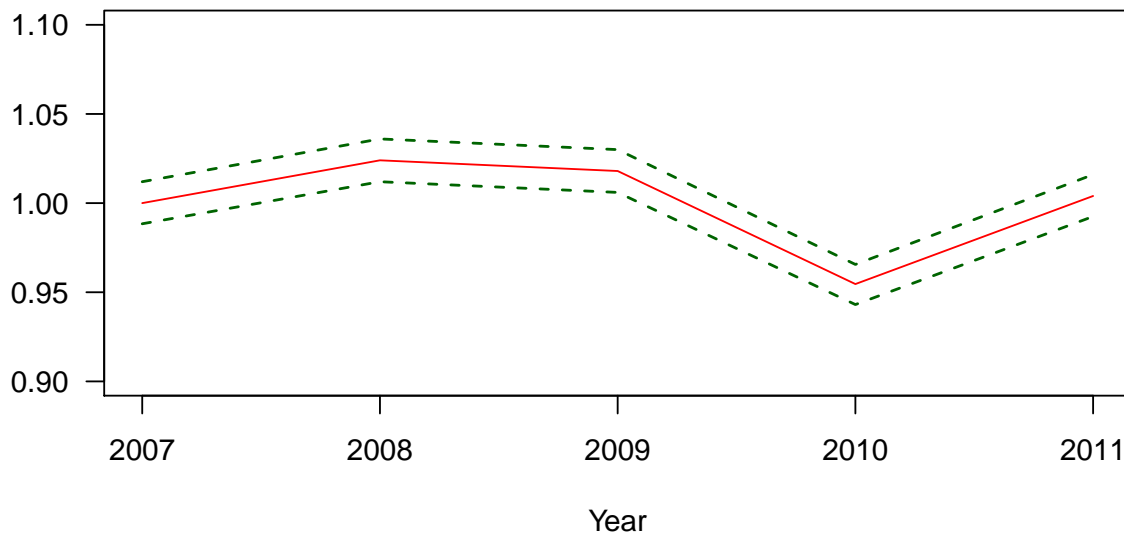
Year <- format(seq(as.Date("2007/01/01"), as.Date("2011/12/31"),
                  by = "year"), "%Y")

Data.tm.plot <- as.data.frame(cbind(Year, Median.tm,
                                   Low.tm, Up.tm))

colnames(Data.tm.plot)<-c("Year", "Post.median", "Low", "Up")

# plot the posterior median with the 95% credible intervals
plot(Year, Data.tm.plot$Post.median, xlab="Year", las=1,
     ylab="", cex=1, col="red", type="l",
     pch=1, ylim=c(0.9, 1.1))
points(Year, Data.tm.plot$Low, type="l",
       col="darkgreen", lwd=1.5, lty=2)
points(Year, Data.tm.plot$Up, type="l",
       col="darkgreen", lwd=1.5, lty=2)

```



We observe that the temporal pattern of respiratory hospital admission shows a downward trend from 2007 to 2010, but newly rises up in 2011.