

R-INLA Practical: spatial and spatio-temporal disease mapping

GEOMED 2019 Conference

Marta Blangiardo and Michela Cameletti

8/27/2019

1. Introduction

In this practical you will extend what already seen in the tutorial and estimate spatial and spatio-temporal disease mapping models. You will study salmonellosis disease in cattles in $n = 199$ swiss regions for the years 1991-2008 ($T = 18$). The data are included in the workspace `data_salmonellosis` which can be loaded in R via

```
load("data_salmonellosis.RData")
```

Typing

```
ls()
```

```
## [1] "data.offset" "data.salm"    "data.Y"       "map"
```

you will see several objects: `data.salm`, `data.Y`, `data.offset`, `map` (this is a `SpatialPolygonsDataFrame` object and will be used for mapping).

The main data file is `data.salm` and typing

```
str(data.salm)
```

```
## 'data.frame':   3582 obs. of  5 variables:
## $ Y          : int  1 3 0 1 0 1 1 0 0 0 ...
## $ offset     : num  28.4 28.4 28.4 28.4 28.4 28.4 28.4 28.4 28.4 28.4 ...
## $ region_id: int  1 1 1 1 1 1 1 1 1 ...
## $ time_id   : int  1 2 3 4 5 6 7 8 9 10 ...
## $ BEZIRKSNR: int  0 0 0 0 0 0 0 0 0 0 ...
```

you can see that the dataset is a `data.frame` with $199 \times 18 = 3582$ rows and five columns:

- `Y`: the number of salmonellosis cases for a combination of area/year
- `offset`: the number of herds in the region
- `region_id`: the area identifier
- `time_id`: the year identifier
- `BEZIRKSNR`: another region ID which matches the one included in the `map` (which you will need for mapping).

Note that this dataset is in **long** format, i.e. each region is repeated 18 times (as there are have 18 years); use `View(data.salm)` if you want to explore the data.

The workspace also contains `data.Y` and `data.offset` which are the cases and the offset in a `data.frame` format:

```
dim(data.Y)
```

```
## [1] 199  19
```

```

dim(data.offset)

## [1] 199 19

colnames(data.Y)

## [1] "1991"      "1992"      "1993"      "1994"      "1995"
## [6] "1996"      "1997"      "1998"      "1999"      "2000"
## [11] "2001"      "2002"      "2003"      "2004"      "2005"
## [16] "2006"      "2007"      "2008"      "BEZIRKSNR"

colnames(data.offset)

## [1] "1991"      "1992"      "1993"      "1994"      "1995"
## [6] "1996"      "1997"      "1998"      "1999"      "2000"
## [11] "2001"      "2002"      "2003"      "2004"      "2005"
## [16] "2006"      "2007"      "2008"      "BEZIRKSNR"

n = nrow(data.Y) #n. of areas
T = ncol(data.Y) - 1 #n. of years

```

Note that here the number of rows is given by the regions and the number of columns is given by the years (18) plus a column for the region BEZIRKSNR identifier.

DISCLAIMER: the code provided is one of the many ways to get the results. For advanced R users: feel free to ignore it and run your own code up to the INLA model.

2. Data preparation and exploration

Create a new `data.frame` which includes the region BEZIRKSNR identifier and the risk (probability) of salmonellosis given by Y/offset:

```

prob.salm = data.frame(BEZIRKSNR=data.Y$BEZIRKSNR,
                       p=data.Y[,1:T]/data.offset[,1:T]) #exclude the BEZIRKSNR col

```

Then transform the risk variables in categorical variables by using the `cut` functions and the following cutoffs: 0, 0.01, 0.5 and 1. Define a new `data.frame`:

```

prob.salm.cat = data.frame(lapply(prob.salm[,2:(T+1)],
                                   function(x) cut(x, c(0,0.01,0.5,1), include.lowest=TRUE)))
#Include area identifier in the dataframe
prob.salm.cat$BEZIRKSNR = prob.salm$BEZIRKSNR
colnames(prob.salm.cat)

## [1] "p.1991"      "p.1992"      "p.1993"      "p.1994"      "p.1995"
## [6] "p.1996"      "p.1997"      "p.1998"      "p.1999"      "p.2000"
## [11] "p.2001"      "p.2002"      "p.2003"      "p.2004"      "p.2005"
## [16] "p.2006"      "p.2007"      "p.2008"      "BEZIRKSNR"

```

Finally, plot the probabilities for three years (1991, 2000 and 2008):

```

#Merge prob.salm and the Switzerland shapefile
map.swiss = map
map.swiss@data = merge(map.swiss@data,
                       prob.salm.cat, by="BEZIRKSNR")

## Loading required package: sp

```

```

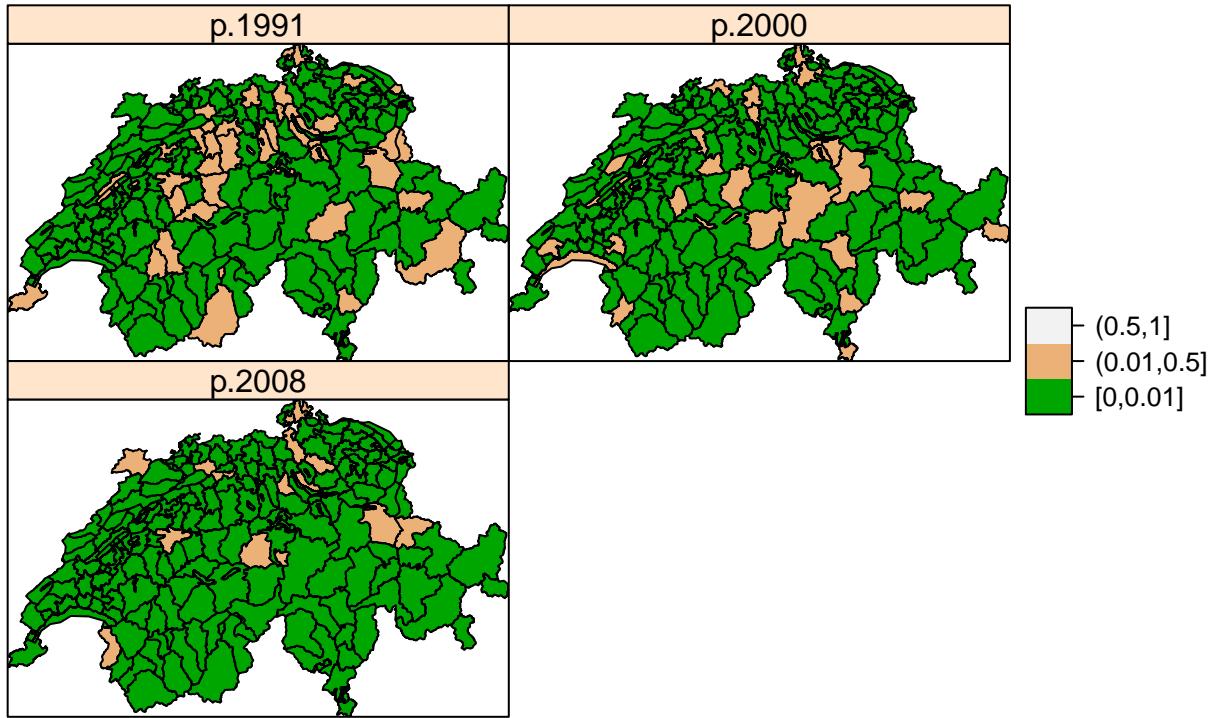
head(map.swiss@data)

##   BEZIRKSNR Cnt_BEZIRK      p.1991      p.1992      p.1993      p.1994
## 1          0            3 (0.01,0.5] (0.01,0.5] [0,0.01] (0.01,0.5]
## 2         100           2 [0,0.01]  [0,0.01]  [0,0.01] (0.01,0.5]
## 3         101          14 [0,0.01]  [0,0.01] (0.01,0.5]  [0,0.01]
## 4         102          24 [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]
## 5         103          23 [0,0.01]  [0,0.01] (0.01,0.5]  [0,0.01]
## 6         104          22 (0.01,0.5] (0.01,0.5] (0.01,0.5] (0.01,0.5]
##               p.1995      p.1996      p.1997      p.1998      p.1999      p.2000      p.2001
## 1  [0,0.01] (0.01,0.5] (0.01,0.5] [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]
## 2  [0,0.01]  [0,0.01]  [0,0.01] [0,0.01] (0.01,0.5]  [0,0.01]  [0,0.01]
## 3  [0,0.01] (0.01,0.5]  [0,0.01] [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]
## 4 (0.01,0.5]  [0,0.01]  [0,0.01] [0,0.01]  [0,0.01] (0.01,0.5]  [0,0.01]
## 5 (0.01,0.5]  [0,0.01]  [0,0.01] [0,0.01]  [0,0.01] (0.01,0.5]  [0,0.01]
## 6 (0.01,0.5] (0.01,0.5] (0.01,0.5] [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]
##               p.2002      p.2003      p.2004      p.2005      p.2006      p.2007
## 1  [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]
## 2 (0.01,0.5]  [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]
## 3  [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01] (0.01,0.5]
## 4  [0,0.01]  [0,0.01]  [0,0.01] (0.01,0.5]  [0,0.01]  [0,0.01]
## 5  [0,0.01] (0.01,0.5]  [0,0.01]  [0,0.01] (0.01,0.5] (0.01,0.5]
## 6  [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]  [0,0.01]
##               p.2008
## 1  [0,0.01]
## 2  [0,0.01]
## 3 (0.01,0.5]
## 4  [0,0.01]
## 5 (0.01,0.5]
## 6  [0,0.01]

spplot(map.swiss,
       c("p.1991", "p.2000", "p.2008"),
       col.regions= terrain.colors(3),
       as.table=TRUE,
       main = "Risk of salmonellosis (Y/offset)")

```

Risk of salmonellosis (Y/offset)



#3. Purely spatial disease mapping model Now, focussing on year 1991, specify a BYM model as the one described in slide 40 of the lecture and Section 4 of the tutorial.

The adjacency matrix in the inla format is already available through the object `switzerland.graph` saved in your directory (note that for this case study it is not possible to create the adjacency matrix from the `map.swiss` object - as shown in the tutorial - because there is a problem in the geographic structure of the polygons).

```
swiss.adj.path = "switzerland.graph"
```

Then define the `formula` and run INLA (use a `logGamma(1,0.1)` prior for both the precisions):

```
library(INLA)
```

```
## Loading required package: Matrix
## This is INLA_18.07.12 built 2018-07-12 11:07:12 UTC.
## See www.r-inla.org/contact-us for how to get help.
## To enable PARDISO sparse library; see inla.pardiso()

formula = Y ~ 1 + f(region_id,
                      model="bym",
                      graph=swiss.adj.path,
                      hyper=list(prec.unstruct=list(prior="loggamma",param=c(1,0.1)),
                                 prec.spatial=list(prior="loggamma",param=c(1,0.1)))

output1 = inla(formula,
               family = "poisson",
               data = data.salm[data.salm$time_id==1], #select just the data of the first year
               offset = log(offset),
               control.compute=list(dic=TRUE))
summary(output1)
```

```

## 
## Call:
## c("inla(formula = formula, family = \"poisson\", data = data.salm[data.salm$time_id == ", " 1, ]")
## 
## Time used:
##   Pre-processing    Running inla Post-processing      Total
##             2.4585          3.1557          0.1714        5.7856
## 
## Fixed effects:
##           mean     sd 0.025quant 0.5quant 0.975quant   mode   kld
## (Intercept) -4.7003 0.2145     -5.1595  -4.6862    -4.3216 -4.6567 6e-04
## 
## Random effects:
## Name   Model
## region_id BYM model
## 
## Model hyperparameters:
##           mean     sd 0.025quant
## Precision for region_id (iid component) 1.890  1.103   0.6519
## Precision for region_id (spatial component) 9.373 10.401   0.6338
##           0.5quant 0.975quant   mode
## Precision for region_id (iid component)     1.604     4.776 1.208
## Precision for region_id (spatial component) 6.227    36.953 1.720
## 
## Expected number of effective parameters(std dev): 33.62(10.72)
## Number of equivalent replicates : 5.918
## 
## Deviance Information Criterion (DIC) .....: 272.29
## Deviance Information Criterion (DIC, saturated) ....: 168.82
## Effective number of parameters .....: 33.18
## 
## Marginal log-Likelihood: -106.46
## Posterior marginals for linear predictor and fitted values computed

```

Now you want to map the spatial random effects. To do this you first bring back $z_i = u_i + v_i$ to the natural scale (from the logarithmic one), compute the posterior mean of z_i and create the corresponding categorical variable:

```
names(output1$ marginals.random)
```

```
## [1] "region_id"
```

```
length(output1$ marginals.random$region_id)
```

```
## [1] 398
```

#Remember that the first n rows include information on z=u+v

#Go back to the natural scale

```
exp.z = unlist(lapply(output1$ marginals.random$region_id[1:n],
                      function(x) inla.emarginal(exp,x)))
range(exp.z)
```

```
## [1] 0.7994582 7.7038390
```

#Create a new dataframe

```
data.exp.z = data.frame(BEZIRKSNR=data.Y$BEZIRKSNR,
```

```

exp.z=exp.z)
#Add also the categorical variable
data.exp.z$exp.z.cat = cut(data.exp.z$exp.z,
                           c(0.88,0.99,1.01,1.2,7.8),
                           include.lowest=T)
head(data.exp.z)

##          BEZIRKSNR      exp.z  exp.z.cat
## index.1      0 2.014090  (1.2,7.8]
## index.2     100 1.286439  (1.2,7.8]
## index.3     101 1.417821  (1.2,7.8]
## index.4     102 1.265068  (1.2,7.8]
## index.5     103 1.162087 (1.01,1.2]
## index.6     104 7.703839  (1.2,7.8]

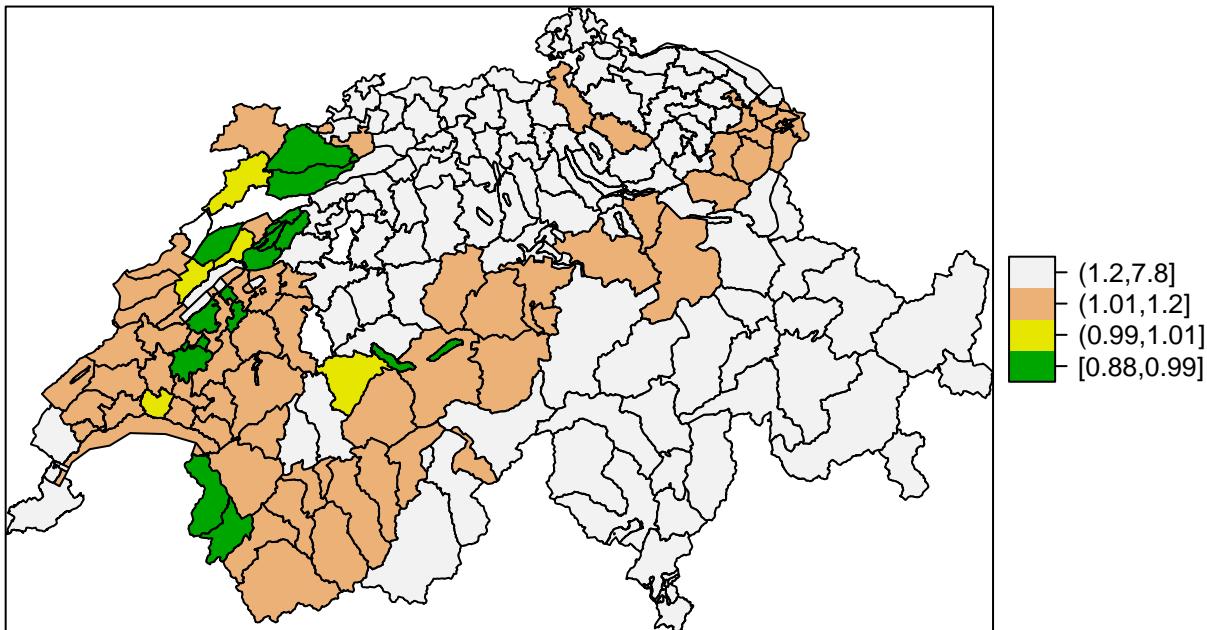
```

Then merge the new dataframe `data.exp.z.c` with the data from the shapefile and finally plot the categorical variable:

```

map.swiss@data = merge(map.swiss@data,
                       data.exp.z,
                       by="BEZIRKSNR")
spplot(obj=map.swiss,
       zcol="exp.z.cat",
       col.regions=terrain.colors(4),
       as.table=TRUE)

```



It is also possible to compute the probability $p(z_i > 1 | \mathbf{y})$ (or equivalently $p(u_i + v_i > 0 | \mathbf{y})$) using the built-in function `inla.pmargin`:

```

#Compute the probabilities
threshold = 0
prob.z = unlist(lapply(output1$ marginals.random$region_id[1:n],
                      function(x) 1-inla.pmargin(threshold,x)))
#Create a new dataframe
data.prob.z = data.frame(BEZIRKSNR=data.Y$BEZIRKSNR,

```

```

prob.z=prob.z)
head(data.prob.z)

##      BEZIRKSNR    prob.z
## index.1      0 0.7280784
## index.2    100 0.5145844
## index.3    101 0.5531607
## index.4    102 0.4903404
## index.5    103 0.4688317
## index.6    104 0.9933848

```

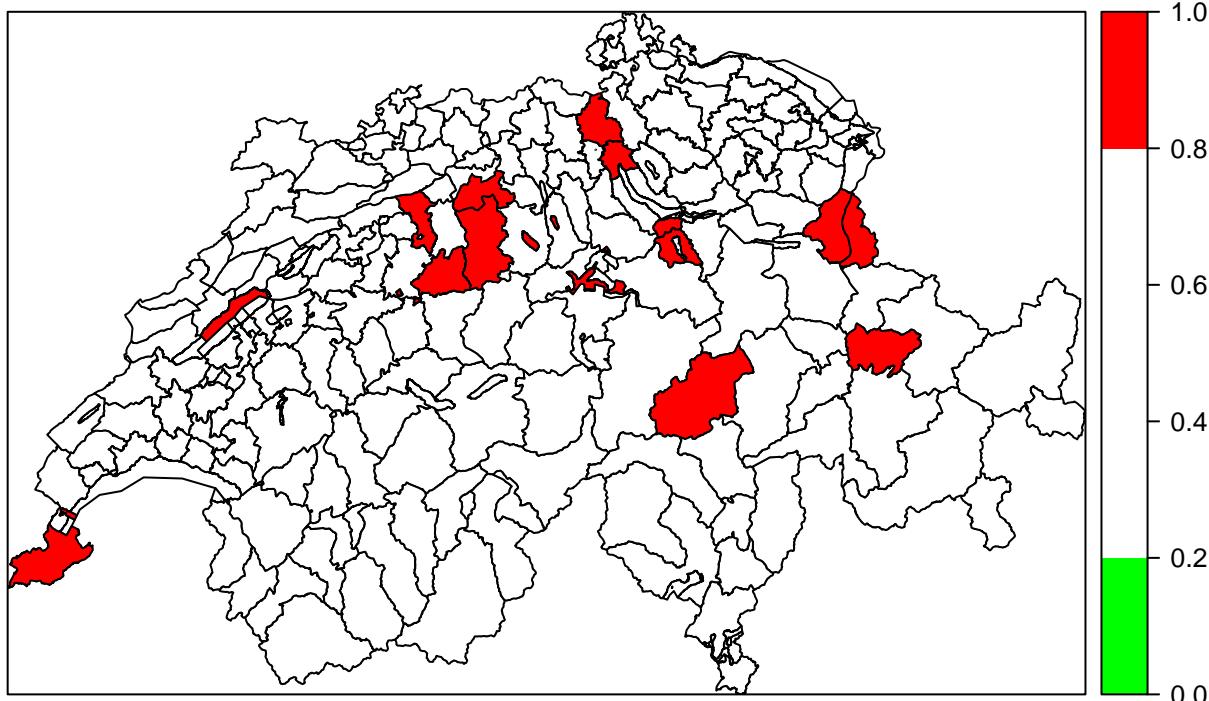
Then you merge the new dataframe `data.prob.z.cat1` with the data from the shapefile and finally plot the categorical variable with the probabilities:

```

prob.cutoff = c(0,0.2,0.8,1)
map.swiss@data = merge(map.swiss@data,
                       data.prob.z,
                       by="BEZIRKSNR")
spplot(obj=map.swiss,
       zcol="prob.z",
       col.regions=c("green","white","red"),
       at=prob.cutoff,
       as.table=TRUE,
       main="BYM model - exp(z)")

```

BYM model – $\exp(z)$



Note the same purely spatial model could be estimated independently for all the available years.

3. Spatio-temporal disease mapping

Now specify a spatio-temporal dynamic model for the salmonellosis data. The model you want to use is the following (see page 62 of the lecture slides):

$$\begin{aligned}
 y_{it} &\sim \text{Poisson}(\phi_{it} E_{it}) \\
 \log(\phi_{it}) &= \beta_0 + u_i + v_i + \lambda_t + \gamma_t \\
 v_i &\sim \text{Normal}(0, \sigma_v^2) \\
 \mathbf{u} &\sim \text{ICAR}(\mathbf{W}, \sigma_u^2) \\
 \gamma_t &\sim \text{Normal}(0, \sigma_\gamma^2) \\
 \lambda_t &\sim \text{RW1}(\sigma_\lambda^2)
 \end{aligned}$$

The model includes, besides the BYM components, two temporal effects (unstructured (`model=iid`) and structured (`model=rw1`)). A logGamma(1,0.1) prior is specified for the four precisions.

To estimate this model you have to create a new temporal identifier as you need to define two time specific components in the model and INLA does not allow to assign more than one `f()` to any identifier:

```

data.salm$time_id2 = data.salm$time_id
formula.ST1 = Y ~ 1+ f(region_id,
                        model="bym",
                        graph=swiss.adj.path,
                        hyper=list(prec.unstruct=list(prior="loggamma",param=c(1,0.1)),
                                   prec.spatial=list(prior="loggamma",param=c(1,0.1)))) +
  f(time_id, model="iid",
    hyper=list(prec=list(prior="loggamma",param=c(1,0.1)))) +
  f(time_id2, model="rw1",
    hyper=list(prec=list(prior="loggamma",param=c(1,0.1))))
  
```

Then run the model

```

output2 = inla(formula.ST1,
               family="poisson",
               data=data.salm,
               offset=log(offset),
               control.compute=list(dic=TRUE))
summary(output2)

##
## Call:
## c("inla(formula = formula.ST1, family = \"poisson\", data = data.salm, ", "      offset = log(offset")
##
## Time used:
##   Pre-processing    Running inla Post-processing          Total
##             3.1523        28.0390         0.3577       31.5490
##
## Fixed effects:
##           mean     sd 0.025quant 0.5quant 0.975quant    mode kld
## (Intercept) -5.404 0.1188     -5.6439  -5.4016     -5.1766 -5.3966  0
##
## Random effects:
## Name Model
## region_id BYM model
## time_id IID model
  
```

```

## time_id2 RW1 model
##
## Model hyperparameters:
##                                     mean      sd 0.025quant
## Precision for region_id (iid component) 1.017  0.205    0.6737
## Precision for region_id (spatial component) 4.584  4.620    0.7884
## Precision for time_id                  19.803 11.534    5.6987
## Precision for time_id2                 15.127  8.828    4.4525
##                                     0.5quant 0.975quant   mode
## Precision for region_id (iid component) 0.9976     1.476  0.9596
## Precision for region_id (spatial component) 3.2093    16.645  1.7979
## Precision for time_id                  17.1800    49.350 12.7610
## Precision for time_id2                 13.0808   37.918  9.7547
##
## Expected number of effective parameters(std dev): 145.74(4.515)
## Number of equivalent replicates : 24.58
##
## Deviance Information Criterion (DIC) .....: 3672.45
## Deviance Information Criterion (DIC, saturated) ....: 2390.67
## Effective number of parameters .....: 142.23
##
## Marginal log-Likelihood: -1889.31
## Posterior marginals for linear predictor and fitted values computed

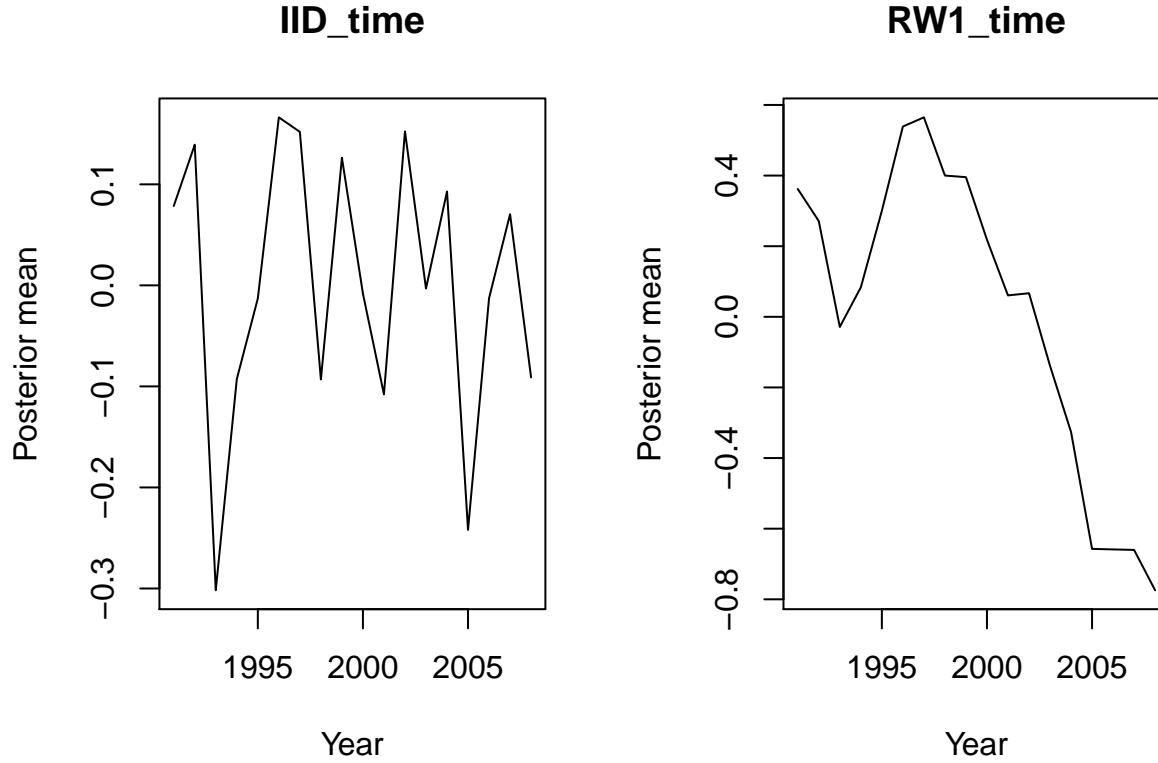
```

It is possible to plot the posterior mean of the two temporal effects:

```

par(mfrow=c(1,2))
plot(1991:2008,
      output2$summary.random$time_id$mean,
      type="l", main="IID_time",ylab="Posterior mean",xlab="Year")
plot(1991:2008,
      output2$summary.random$time_id2$mean,
      type="l", main="RW1_time",ylab="Posterior mean",xlab="Year")

```



```
par(mfrow=c(1,1))
```

It would be also possible to compute the **sum** of the two temporal effect as the total temporal trend $\gamma_t + \lambda_t$. To do this you need to use **inla.make.lincombs** function (see for example Section 7.1.1 of the INLA book) to create a linear combination of two random effects.

As done in the previous Section 2., it is also possible to plot for this model the residual spatial random effects ($z_i = u_i + v_i$).

4. Spatio-temporal disease mapping with space-time interaction

You will now run a spatio-temporal model with a type I interaction (see page 63-64 of the lecture slides).

$$\begin{aligned}
y_{it} &\sim \text{Poisson}(\phi_{it} E_{it}) \\
\log(\phi_{it}) &= \beta_0 + u_i + v_i + \lambda_t + \gamma_t \\
v_i &\sim \text{Normal}(0, \sigma_v^2) \\
\mathbf{u} &\sim \text{ICAR}(\mathbf{W}, \sigma_u^2) \\
\gamma_t &\sim \text{Normal}(0, \sigma_\gamma^2) \\
\lambda_t &\sim \text{RW1}(\sigma_\lambda^2) \\
\delta_{it} &\sim N(0, \sigma_\delta^2)
\end{aligned}$$

A logGamma(1,0.1) prior is specified for the five precisions. To run this model you need to specify another identifier for the interaction, going from 1 to the total number of rows in **data.salm**:

```
#ID
region_time_id = seq(1,nrow(data.salm))
```

```

#Formula
formula.intI = Y ~ 1 + f(region_id,
                           model="bym",
                           graph=swiss.adj.path,
                           hyper=list(prec.unstruct=list(prior="loggamma", param=c(1,0.1)),
                                      prec.spatial=list(prior="loggamma", param=c(1,0.1)))) +
                           f(time_id, model="iid",
                             hyper=list(prec=list(prior="loggamma", param=c(1,0.1)))) +
                           f(time_id2, model="rw1",
                             hyper=list(prec=list(prior="loggamma", param=c(1,0.1)))) +
                           f(region_time_id, model="iid",
                             hyper=list(prec=list(prior="loggamma", param=c(1,0.1)))))

#Run INLA
output3 = inla(formula.intI,
                family="poisson",
                data=data.salm,
                offset=log(offset),
                control.compute=list(dic=TRUE))
summary(output3)

##
## Call:
## c("inla(formula = formula.intI, family = \"poisson\", data = data.salm, ", "      offset = log(offset)
##
## Time used:
##   Pre-processing    Running inla Post-processing        Total
##             3.0853          66.6302          0.7425         70.4581
##
## Fixed effects:
##               mean     sd 0.025quant 0.5quant 0.975quant    mode kld
## (Intercept) -5.674 0.1252    -5.9267  -5.6718    -5.434 -5.6676  0
##
## Random effects:
## Name Model
## region_id BYM model
## time_id   IID model
## time_id2  RW1 model
## region_time_id IID model
##
## Model hyperparameters:
##                               mean     sd 0.025quant
## Precision for region_id (iid component) 1.134 0.2501 0.7266
## Precision for region_id (spatial component) 3.726 3.3619 0.7079
## Precision for time_id                  20.218 12.1121 5.9322
## Precision for time_id2                 17.420 10.0258 4.9915
## Precision for region_time_id           1.623 0.2396 1.2164
##                               0.5quant 0.975quant    mode
## Precision for region_id (iid component) 1.105     1.704 1.049
## Precision for region_id (spatial component) 2.749     12.572 1.624
## Precision for time_id                  17.310     51.643 12.792
## Precision for time_id2                 15.193     43.071 11.323
## Precision for region_time_id           1.599     2.157 1.549
##

```

```

## Expected number of effective parameters(std dev): 495.13(35.32)
## Number of equivalent replicates : 7.234
##
## Deviance Information Criterion (DIC) .....: 3387.19
## Deviance Information Criterion (DIC, saturated) ....: 2105.41
## Effective number of parameters .....: 434.22
##
## Marginal log-Likelihood: -1831.36
## Posterior marginals for linear predictor and fitted values computed

It is possible to map the posterior mean of the interaction  $\delta_{it}$  where  $i = 1, \dots, n$  and  $t = 1, \dots, T$ :

#Extract the posterior mean (log scale) and create a new data frame
delta.intI = data.frame(delta=output3$summary.random$region_time_id$mean,
                        year=data.salm$time_id,
                        ID.area=data.salm$region_id)
dim(delta.intI)

## [1] 3582    3

head(delta.intI)

##      delta year ID.area
## 1  0.31852263    1      1
## 2  1.16193664    2      1
## 3 -0.09983847    3      1
## 4  0.41458621    4      1
## 5 -0.15261009    5      1
## 6  0.29121551    6      1

#Transform into categories
delta.intI$delta.cat = cut(delta.intI$delta,
                           c(-0.75,-0.05,0.05,3.1),
                           include.lowest=T)

#Reshape the dataframe so that the dimension is 199 (n) X 18 (T)
delta.intI.wide = reshape(delta.intI,direction="wide",
                           timevar = "year",
                           idvar = "ID.area",
                           drop = "delta")
dim(delta.intI.wide)

## [1] 199  19

#Remove the first column twith the station ID
delta.intI.wide = delta.intI.wide[,-1]
#Change the column names
colnames(delta.intI.wide) = paste("delta.",seq(1991,2008),sep="")
#Add the BEZIRKSNR identifier
delta.intI.wide$BEZIRKSNR = data.Y$BEZIRKSNR

```

For mapping it is necessary to merge the shapefile datatable with the new dataframe `delta.intI.wide`:

```

map.swiss@data = merge(map.swiss@data,
                      delta.intI.wide,
                      by="BEZIRKSNR")
spplot(obj=map.swiss,
       zcol=c("delta.1991","delta.2000","delta.2008"),

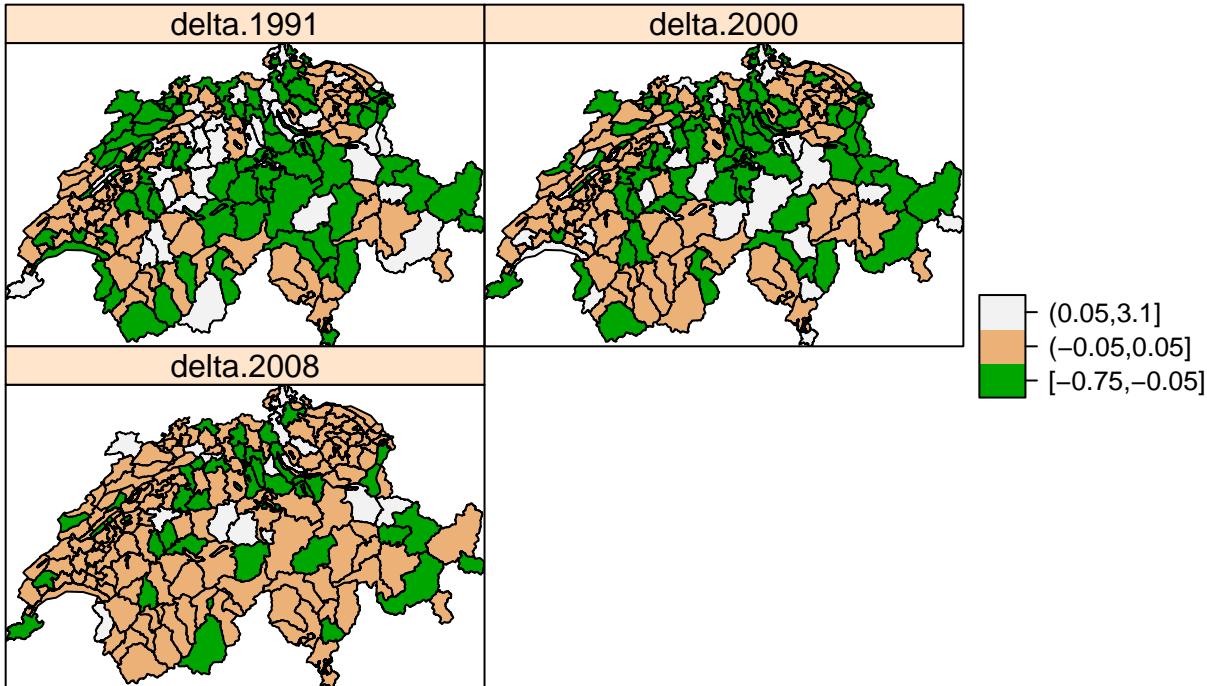
```

```

col.regions= terrain.colors(3),
main="Type I interaction",
as.table=TRUE)

```

Type I interaction



The same approach could be adopted to map for each available year the posterior probability that δ exceeds the threshold 0:

```

#Extract the probability
prob.delta.vec = unlist(lapply(output3$marginals.random$region_time_id,
                               function(x){1 - inla.pmarginal(threshold, x)}))

#Create a new dataframe
prob.delta.intI = data.frame(prob.delta=prob.delta.vec,
                             year=data.salm$time_id,
                             ID.area=data.salm$region_id)

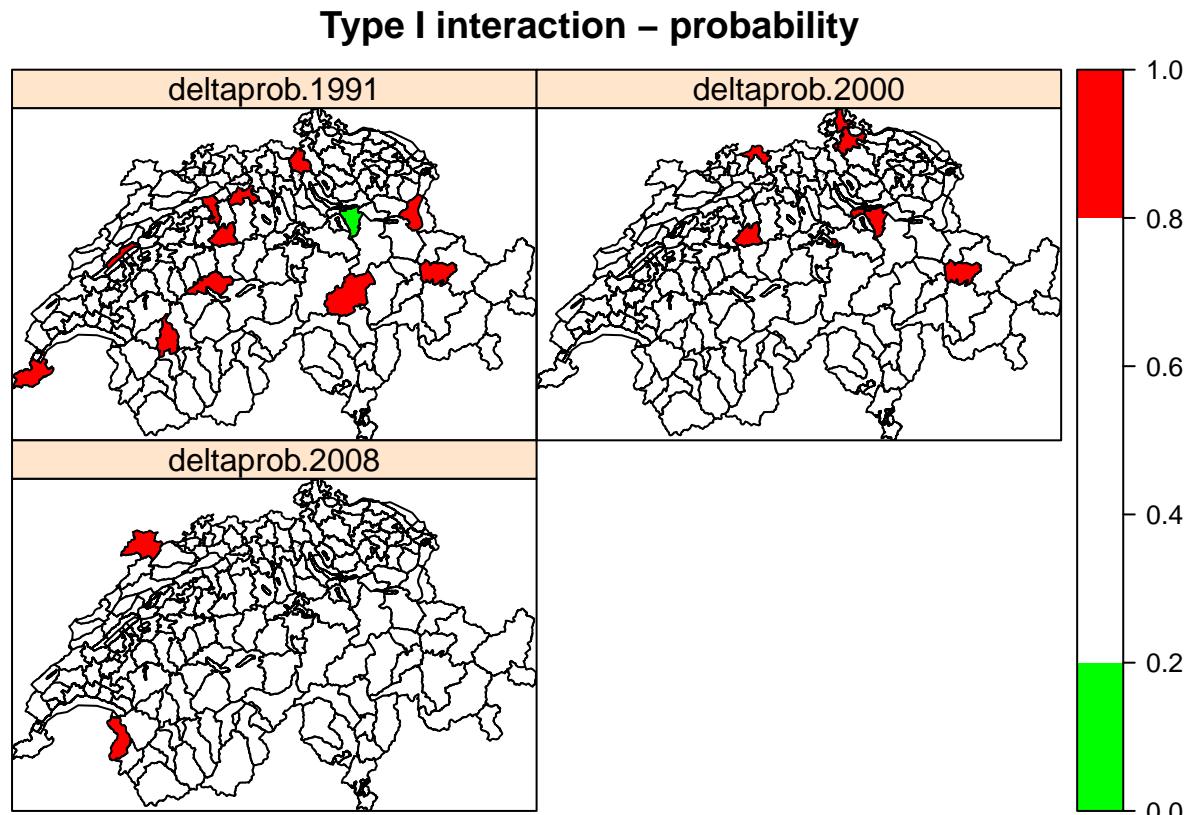
#Reshape the dataframe so that the dimension is 199 (n) X 18 (T)
prob.delta.intI.wide = reshape(prob.delta.intI,direction="wide",
                                timevar = "year",
                                idvar = "ID.area")

#Remove the first column twith the station ID
prob.delta.intI.wide = prob.delta.intI.wide[,-1]
#Change the column names
colnames(prob.delta.intI.wide) = paste("deltaprob.",seq(1991,2008),sep="")
#Add the BEZIRKSNR identifier
prob.delta.intI.wide$BEZIRKSNR = data.Y$BEZIRKSNR

#Merge before mapping
map.swiss@data = merge(map.swiss@data, prob.delta.intI.wide, by="BEZIRKSNR")

```

```
spplot(obj=map.swiss,
      zcol=c("deltaprob.1991","deltaprob.2000","deltaprob.2008"),
      col.regions=c("green","white", "red"),
      at=prob.cutoff,
      main="Type I interaction - probability",
      as.table=TRUE)
```



Finally we compare the dic of the two spatio-temporal models (output2: spatio-temporal model with bym + 2 temporal effects, output3: spatio-temporal model with bym and space-time type I interaction)

```
output2$dic$dic
```

```
## [1] 3672.448
```

```
output3$dic$dic
```

```
## [1] 3387.19
```

It seems that the model with the interaction is preferable.