Bayesian Hierarchical Models

Practical 2: Bayesian disease mapping

Introduction

In this session we will work through an example of creating raw and smoothed SMRs, using both MCMC and R-INLA, and plot them on a map.

Preliminaries

We need the following packages

- `spdep` - Package to create spatial objects (such as neighbourhood matrix).
- `shapefiles` - Package to read and write shapefiles.
- `CARBayes` - Package to fit spatial GLMMs.
- `INLA` - Package to perform full Bayesian analysis of Latent Gaussian Models using Integrated Nested Laplace Approximations.
- `rgdal` - Package to handle spatial objects.
- `rgeos` - Package to handle spatial objects.

Make sure that these packages are downloaded and installed in `R`. We use the `require()` function to load them into the `R` library.

```
require(spdep)
require(shapefiles)
require(CARBayes)
require(INLA)
require(rgdal)
require(rgeos)
```

Data

We will work with data on hospital admissions for chronic obstructive pulmonary disease (COPD) for England between 2001–2010. All data required for this practical can be found in the folder `Data`. You will need the following files

- shapefiles and information for England split by local authorities (`englandlocalauthority.shp`, `englandlocalauthority.dbf`)
- observed numbers of hospital admissions by local authority (`copdmortalityobserved.csv`)
- expected numbers of hospital admissions by local authority (`copdmortalityobserved.csv`).

The observed and expected COPD counts in England by local authority, are in csv format, so we use the `read.csv()` function to read these into `R`.

```r
# Reading in the observed counts
observed <- read.csv(file="copdmortalityobserved.csv", row.names=1)

# Reading in the expected counts
expected <- read.csv(file="copdmortalityexpected.csv", row.names=1)
```

To check that the data has been read into `R` correctly, we can use the `head()` function, which prints the first six rows of a dataset.

```r
# Viewing the first 6 rows of the observed counts
head(observed)
                        name Y2001 Y2002 Y2003 Y2004 Y2005 Y2006 Y2007
00AA      City of London LB     2     0     3     1     1     1     5
00AB Barking and Dagenham LB   100   100   122    93   136    97    91
00AC               Barnet LB   110   102   106    89    99    97    72
00AD               Bexley LB   109   113   113    96   113    97    94
00AE                Brent LB    69    89    70    59    61    48    53
00AF              Bromley LB   120   129   135   124   128   117   120
     Y2008 Y2009 Y2010
00AA     1     0     1
00AB    96   101    78
00AC    84    78    89
00AD    89    93    93
00AE    46    55    43
00AF   106   107   113

# Viewing the first 6 rows of the expected counts
head(expected)
          E2001     E2002      E2003      E2004      E2005      E2006
00AA   2.648915   2.68106   2.727112   2.749562   2.808655   2.915977
00AB  63.946730  63.41700  62.567863  61.444884  60.677119  59.678672
00AC 121.795213 121.91534 122.451050 123.201898 124.449563 125.982868
00AD  90.201336  91.24645  91.949050  92.754781  93.674540  94.598593
00AE  76.876437  77.18529  78.017980  78.967493  80.422828  81.785325
00AF 131.182934 132.30521 133.257442 134.520920 136.441229 137.382528
          E2007      E2008      E2009      E2010
00AA   3.021586   3.114696   3.237998   3.237998
00AB  58.487583  57.701932  57.250524  57.250524
00AC 127.088805 128.825149 131.374946 131.374946
00AD  95.447131  96.832061  97.651369  97.651369
00AE  83.651266  85.265264  87.089119  87.089119
00AF 138.634021 139.508507 140.634084 140.634084
```

To familiarise ourselves with the data, we can summarise it using the `summary()` function. This will allow us to check for anomalies in our data.

```r
# Summarising the observed counts
summary(observed)
              name          Y2001            Y2002            Y2003
 Adur CD        :  1   Min.   :  2.00   Min.   :  0.00   Min.   :  3.00
 Allerdale CD   :  1   1st Qu.: 35.00   1st Qu.: 38.00   1st Qu.: 38.00
 Amber Valley CD:  1   Median : 50.00   Median : 52.00   Median : 52.00
 Arun CD        :  1   Mean   : 68.01   Mean   : 69.63   Mean   : 73.44
```

```
 Ashfield CD    :  1   3rd Qu.: 83.50   3rd Qu.: 80.75   3rd Qu.: 83.25
 Ashford CD     :  1   Max.   :445.00   Max.   :438.00   Max.   :480.00
 (Other)        :318
     Y2004            Y2005            Y2006            Y2007
 Min.   :  1.00   Min.   :  1.00   Min.   :  1.00   Min.   :  5.00
 1st Qu.: 35.00   1st Qu.: 37.00   1st Qu.: 35.00   1st Qu.: 37.00
 Median : 49.50   Median : 51.00   Median : 49.00   Median : 50.00
 Mean   : 66.67   Mean   : 69.37   Mean   : 67.07   Mean   : 68.17
 3rd Qu.: 81.25   3rd Qu.: 80.50   3rd Qu.: 81.00   3rd Qu.: 79.00
 Max.   :428.00   Max.   :395.00   Max.   :428.00   Max.   :456.00

     Y2008            Y2009            Y2010
 Min.   :  1.00   Min.   :  0.00   Min.   :  1.00
 1st Qu.: 37.00   1st Qu.: 36.00   1st Qu.: 38.00
 Median : 51.00   Median : 50.00   Median : 51.00
 Mean   : 71.40   Mean   : 67.04   Mean   : 68.81
 3rd Qu.: 84.25   3rd Qu.: 78.00   3rd Qu.: 81.25
 Max.   :463.00   Max.   :394.00   Max.   :441.00
```

```
# Summarising the expected counts
summary(expected)
     E2001             E2002             E2003             E2004
 Min.   :  2.649   Min.   :  2.681   Min.   :  2.727   Min.   :  2.75
 1st Qu.: 39.066   1st Qu.: 39.456   1st Qu.: 39.849   1st Qu.: 40.60
 Median : 51.766   Median : 52.671   Median : 53.487   Median : 54.29
 Mean   : 62.944   Mean   : 63.589   Mean   : 64.139   Mean   : 64.72
 3rd Qu.: 74.292   3rd Qu.: 74.974   3rd Qu.: 74.701   3rd Qu.: 74.02
 Max.   :370.913   Max.   :371.271   Max.   :369.861   Max.   :368.87
     E2005             E2006             E2007             E2008
 Min.   :  2.809   Min.   :  2.916   Min.   :  3.022   Min.   :  3.115
 1st Qu.: 41.646   1st Qu.: 42.497   1st Qu.: 43.203   1st Qu.: 44.262
 Median : 54.765   Median : 55.506   Median : 56.552   Median : 57.522
 Mean   : 65.440   Mean   : 66.180   Mean   : 67.022   Mean   : 67.950
 3rd Qu.: 75.003   3rd Qu.: 75.260   3rd Qu.: 75.790   3rd Qu.: 76.935
 Max.   :368.565   Max.   :367.838   Max.   :368.026   Max.   :368.291
     E2009             E2010
 Min.   :  3.238   Min.   :  3.238
 1st Qu.: 45.062   1st Qu.: 45.062
 Median : 58.077   Median : 58.077
 Mean   : 68.901   Mean   : 68.901
 3rd Qu.: 78.166   3rd Qu.: 78.166
 Max.   :368.940   Max.   :368.940
```

Activities

- Does it look like `R` has read in the data correctly?
- Are there any strange values in our dataset?

To create maps of SMRs, we need to read in the relevant shapefiles. The files `englandlocalauthority.shp` and `englandlocalauthority.dbf` contain the location, shape, and attributes of English local authorities. The functions `read.shp()` and `read.dbf()` will read shapefiles into `R`.

```r
# Read in shapefiles for England split by local authorities
shp <- read.shp(shp.name="englandlocalauthority.shp")
shp$shp[[23]]$points <- shp$shp[[23]]$points[-c(460, 461, 462, 463, 464, 465), ]

# Read in information for England split by local authorities
dbf <- read.dbf(dbf.name="englandlocalauthority.dbf")
dbf$dbf <- dbf$dbf[ ,c(2,1,3:7)]
```

Creating Raw SMRs

Now that we have read in the data, we can calculate raw SMRs,

$$\text{SMR} = \frac{\text{observed}}{\text{expected}}$$

```r
# Creating raw SMRs
SMR_raw <- observed[ ,-1]/expected
```

To change the column names, we use the `names()` function.

```r
# Changing column names
names(SMR_raw) <- c("SMR2001", "SMR2002", "SMR2003", "SMR2004", "SMR2005",
                    "SMR2006", "SMR2007", "SMR2008", "SMR2009", "SMR2010")
```

It is important that we check that no errors have occurred at any stages, so we check by summarising the results using the `head()` and `summary()` functions.

```r
# Viewing the first 6 rows of SMR_raw
head(SMR_raw)
       SMR2001   SMR2002   SMR2003   SMR2004   SMR2005   SMR2006   SMR2007
00AA 0.7550261 0.0000000 1.1000648 0.3636943 0.3560423 0.3429382 1.6547601
00AB 1.5638016 1.5768644 1.9498828 1.5135516 2.2413721 1.6253713 1.5558858
00AC 0.9031554 0.8366462 0.8656520 0.7223915 0.7955030 0.7699460 0.5665330
00AD 1.2084078 1.2384043 1.2289415 1.0349871 1.2063043 1.0253852 0.9848384
00AE 0.8975442 1.1530694 0.8972291 0.7471429 0.7584911 0.5869024 0.6335828
00AF 0.9147531 0.9750183 1.0130766 0.9217897 0.9381329 0.8516367 0.8655884
       SMR2008   SMR2009   SMR2010
00AA 0.3210586 0.0000000 0.3088328
00AB 1.6637225 1.7641760 1.3624329
00AC 0.6520466 0.5937205 0.6774503
00AD 0.9191171 0.9523676 0.9523676
00AE 0.5394928 0.6315370 0.4937471
00AF 0.7598103 0.7608397 0.8035037

# Summarising SMR_raw
summary(SMR_raw)
    SMR2001          SMR2002          SMR2003          SMR2004
 Min.   :0.3883   Min.   :0.0000   Min.   :0.3616   Min.   :0.2778
 1st Qu.:0.7900   1st Qu.:0.8272   1st Qu.:0.8519   1st Qu.:0.7636
 Median :0.9496   Median :1.0168   Median :1.0209   Median :0.9266
 Mean   :1.0349   Mean   :1.0508   Mean   :1.0895   Mean   :0.9812
```

```
 3rd Qu.:1.2526   3rd Qu.:1.2364   3rd Qu.:1.3071   3rd Qu.:1.1858  
 Max.   :1.9861   Max.   :2.2181   Max.   :2.2483   Max.   :1.9811  
    SMR2005          SMR2006          SMR2007          SMR2008      
 Min.   :0.3326   Min.   :0.3429   Min.   :0.3509   Min.   :0.3211  
 1st Qu.:0.7592   1st Qu.:0.7415   1st Qu.:0.7533   1st Qu.:0.7695  
 Median :0.9573   Median :0.9101   Median :0.9305   Median :0.9404  
 Mean   :1.0126   Mean   :0.9726   Mean   :0.9743   Mean   :1.0069  
 3rd Qu.:1.2083   3rd Qu.:1.1586   3rd Qu.:1.1679   3rd Qu.:1.1979  
 Max.   :2.2414   Max.   :2.0805   Max.   :1.8528   Max.   :2.0567  
    SMR2009          SMR2010      
 Min.   :0.0000   Min.   :0.3088  
 1st Qu.:0.7452   1st Qu.:0.7682  
 Median :0.8777   Median :0.9337  
 Mean   :0.9328   Mean   :0.9639  
 3rd Qu.:1.0934   3rd Qu.:1.1335  
 Max.   :1.8507   Max.   :2.3856  
```

Activities

- Does it look like the SMRs have been estimated correctly?
- Are there any strange values?

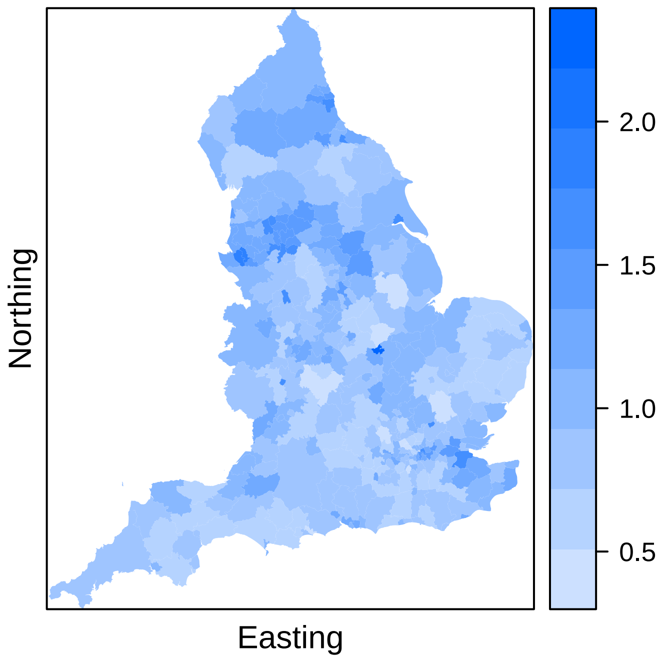
To plot these SMRs on a map, we need to attach them to the shapefile. To achieve this, we can use the `combine.data.shapefile()` function.

```
# Attaching the raw SMRs to the the shapefiles
SMRspatial_raw <- combine.data.shapefile(data = SMR_raw, # Dataset to attach
                                         shp  = shp,     # Shapefile
                                         dbf  = dbf)     # Database file
```

Now that the estimates are attached to the shapefile, the function `spplot()` allows us to create a map with colours representing the magnitude of the SMRs in each of the local authorities.

```
# Breaks for legend of plot
range <- seq(min(SMR_raw$SMR2010)-0.01, max(SMR_raw$SMR2010)+0.01, length.out=11)

# Creating a map of 2010 smoothed estimates
spplot(obj  = SMRspatial_raw,    # Spatial object to be plotted
       zcol = c("SMR2010"),  # Choice of the column  the object you are plotting.
       xlab = "Easting",     # Label of the x column
       ylab = "Northing",    # Label of the y column
       at   = range,         # Break points for colours
       col  = "transparent", # Choice of colour
       col.regions = hsv(0.6, seq(0.2, 1, length.out=10), 1)) # Create a set of colours
```



Activities

- What do you notice about this plot?
- Are there any extreme values?
- If so, do you believe that these represent the truth or are due to sampling error?

Creating Smoothed SMRs using MCMC

In this part, we will implement a Poisson log-normal model with spatial effects using MCMC. To calculate the smoothed SMRs, we first need to create a 'neighbourhood' structure, otherwise known as an adjacency matrix. The functions `poly2nb()` and `nb2mat()` can be used to create this adjacency matrix.

```
# Creates the neighbourhood
W.nb <- poly2nb(SMRspatial_raw, row.names = rownames(SMRspatial_raw))
# Creates a matrix for following function call
W.mat <- nb2mat(W.nb, style="B")
```

Here, we use 'first neighbours' to define our structure, so any local authority that shares a border with another are considered neighbours.

We now create a dataframe which we will use to model the 2010 smoothed SMRs.

```
# Extracting observed values
obs <- observed$Y2010

# Extracting expected values
exp <- expected$E2010

# Creating a dataset to input to INLA
dat <- data.frame(obs, exp)
```

The function `S.CARleroux()` allows us to fit a Poisson log-normal model with spatial effects using MCMC. This function will use this neighbourhood structure and perform a Bayesian analysis, to create a smoothed set of observed values as discussed in the lecture.

```
# Model Formula
formula <- obs ~ 1 + offset(log(exp))

# Calculating smoothed SMRs using MCMC
model <- S.CARleroux(formula = formula,  # Model Formula
                     family = "poisson", # Choosing Poisson Regression
                     W = W.mat,          # Neighbourhood matrix
                     burnin = 20000,     # Number of burn in samples
                     n.sample = 100000,  # Number of MCMC samples
                     thin = 10,
                     fix.rho = TRUE,
                     rho = 1,
                     data = dat)
```

The new smoothed values can be extracted from the model output and divided by the expected values to allow comparison between the areas.

```
# Calculating the 2010 smoothed SMRs
SMR2010_MCMC <- model$fitted.values / expected$E2010

# Putting 2010 smoothed SMRs into a dataframe
SMR_smooth_MCMC <- as.data.frame(SMR2010_MCMC, row.names = rownames(observed))
```

Again, we check that no errors have occurred, by summarising the results using the `head()` and `summary()` functions.

```
# Looking at the first 6 rows of SMR_smooth_MCMC
head(SMR_smooth_MCMC)
     SMR2010_MCMC
00AA    0.9993103
00AB    1.2582397
00AC    0.6865305
00AD    0.9719282
00AE    0.5994937
00AF    0.8581860

# Summarising SMR_smooth_MCMC
summary(SMR_smooth_MCMC)
  SMR2010_MCMC
 Min.   :0.5444
 1st Qu.:0.7936
 Median :0.9242
 Mean   :0.9648
 3rd Qu.:1.0802
 Max.   :1.7286
```

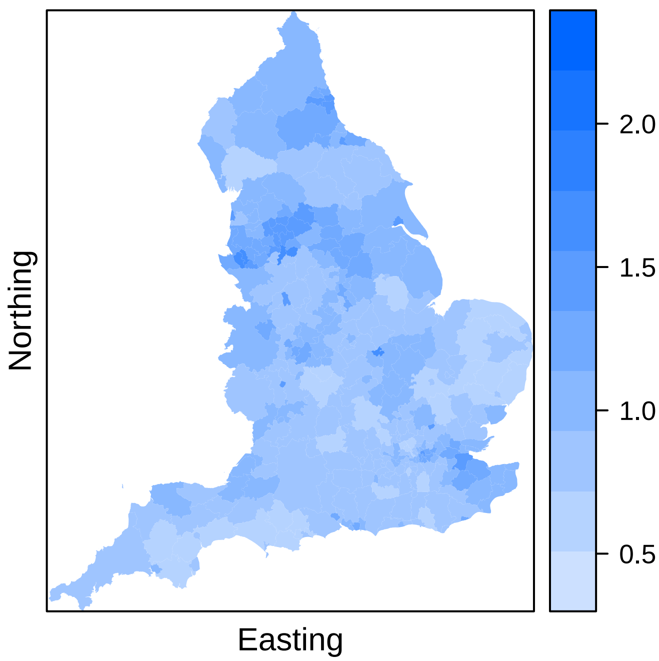
Activities

- Does it look as thought the smooothed SMRs have been estimated correctly?

As before, we attach the smoothed SMRs to the shapefile using the `combine.data.shapefile()` function and create a map using the `spplot()` function.

```
# Attaching the Smoothed SMRs extimated from MCMC to a shapefile
SMRspat_smooth_MCMC <- combine.data.shapefile(data = SMR_smooth_MCMC, # Dataset to attach
                                              shp  = shp,          # Shapefile
                                              dbf  = dbf)          # Database file

# Creating a map of 2010 smoothed SMRs extimated from MCMC
spplot(obj  = SMRspat_smooth_MCMC,    # Spatial object to be plotted
       zcol = c("SMR2010_MCMC"),         # Choice of the column  the object you are plotting.
       xlab = "Easting",             # Label of the x column
       ylab = "Northing",            # Label of the y column
       at   = range,                 # Break points for colours
       col  = "transparent",         # Choice of colour
       col.regions = hsv(0.6, seq(0.2, 1, length.out=10), 1)) # Create a set of colours
```



Activities

- What do you notice about this new map?
- Are there any differences between the smoothed and raw estimates?

Creating Smoothed SMRs using R-INLA

We will now implement a Poisson log-normal model with spatial effects using `R-INLA`. To calculate the smoothed SMRs in `R-INLA`, we first need to convert the 'neighbourhood' structure from the previous exmaple to a format that can be used with `R-INLA` format. We do this using the `nb2INLA()` function.

```r
# Create an INLA format neighbourhood structure
nb2INLA("UK.adj",W.nb)
```

We also need to create IDs for the areas to enable use to match our data to the neighbourhood structure. We add these IDs to our dataset.

```r
# Creating an ID to match observations ot area
dat$ID <- 1:nrow(observed)
```

The function `inla()` allows us to fit a Poisson log-normal model with spatial effects using `R-INLA`. We specify the random effects within `f()`. Here, we specify that we are using an 'besag' (iCAR) model and include the neighbourhood structure.

```r
# Formula for Poisson log-normal model with spatial effects
formula <- obs ~ 1 + f(ID, model="besag", graph="UK.adj")

# Fitting the model with R-INLA
mod <- inla(formula,
            family = "poisson",
            data = dat,
            E = exp, # Expected Values
            control.predictor=list(compute=TRUE)) # Create predictions
```

Note that here we do not include the expected values within the model formula. They are treated as an offset and specified within the the `inla()` function.

The smoothed values estimated using `R-INLA` can be extracted from the model output.

```r
# Extracting the 2010 smoothed SMRs
SMR2010_INLA <- mod$summary.fitted.values$mean

# Putting 2010 smoothed SMRs into a dataframe
SMR_smooth_INLA <- as.data.frame(SMR2010_INLA,
                                 row.names = rownames(observed))
```

Again, we check that no errors have occurred, by summarising the results using the `head()` and `summary()` functions.

```r
# Looking at the first 6 rows of SMR_smooth_INLA
head(SMR_smooth_INLA)
     SMR2010_INLA
00AA    1.0118323
00AB    1.2692888
00AC    0.6902161
00AD    0.9769350
00AE    0.6035849
00AF    0.8632745

# Summarising SMR_smooth_INLA
summary(SMR_smooth_INLA)
  SMR2010_INLA
 Min.   :0.5543
 1st Qu.:0.8035
```

```
 Median :0.9317
 Mean   :0.9727
 3rd Qu.:1.0884
 Max.   :1.7349
```

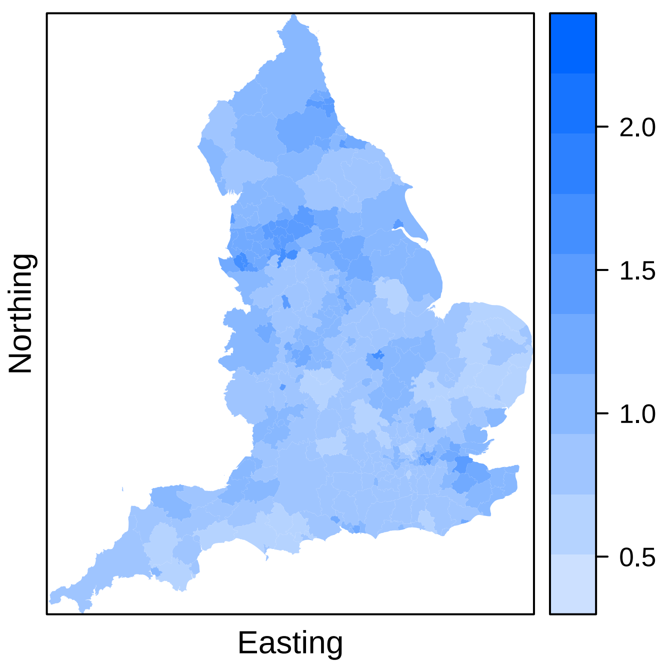
Activities

- Does it look like the SMRs have been estimated correctly using R-INLA?
- Are there any strange values?

Similarly to before, we attach the values of the smoothed SMRs estimated using R-INLA to the shapefile using the `combine.data.shapefile()` function and create a map using the `spplot()` function.

```
# Attaching the Smoothed SMRs extimated from INLA to a shapefile
SMRspat_smooth_INLA <- combine.data.shapefile(data = SMR_smooth_INLA, # Dataset to attach
                                              shp  = shp,         # Shapefile
                                              dbf  = dbf)         # Database file

# Creating a map of 2010 smoothed SMRs extimated from R-INLA
spplot(obj  = SMRspat_smooth_INLA,  # Spatial object to be plotted
       zcol = c("SMR2010_INLA"),    # Choice of the column  the object you are plotting.
       xlab = "Easting",            # Label of the x column
       ylab = "Northing",           # Label of the y column
       at   = range,                # Break points for colours
       col  = "transparent",        # Choice of colour
       col.regions = hsv(0.6, seq(0.2, 1, length.out=10), 1)) # Create a set of colours
```



Activities

- What do you notice about this new map?
- Are there any differences between the smoothed and raw estimates?

Comparison of MCMC and R-INLA for smoothing risks

We have calculated two sets of smoothed estimates using MCMC and R-INLA and we are particularly interested in whether the resutls differ between the two methods. Firstly, we create a dataframe to enable us to match the results from the different methods.

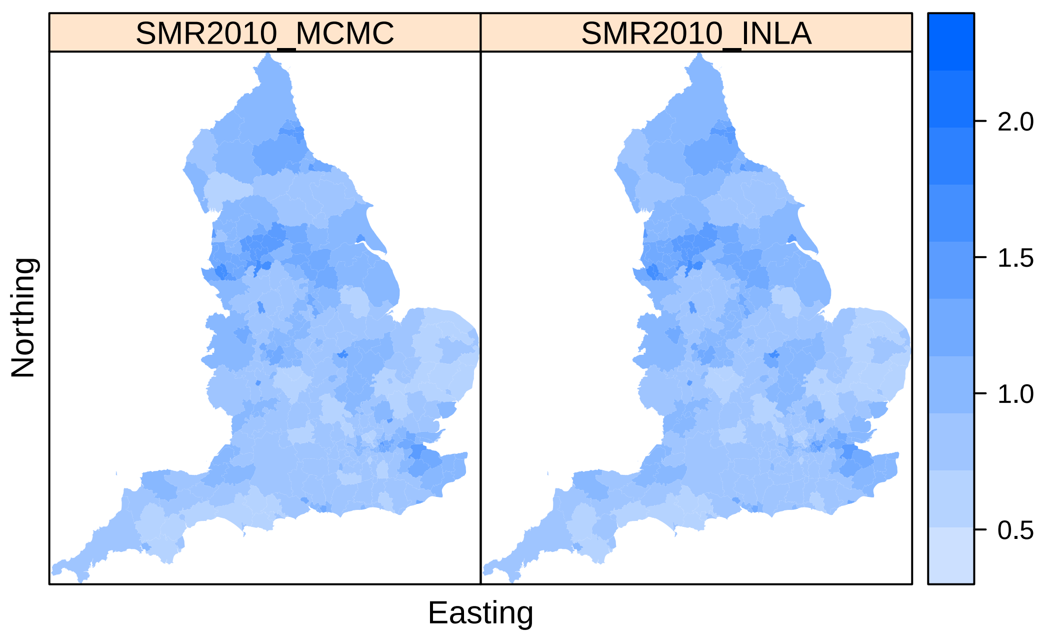
```
# Creating a dataframe with the raw and smoothed estimates of MCMC and INLA to analyse
smoothed_2010 <- data.frame(SMR2010_MCMC,
                            SMR2010_INLA,
                            row.names = rownames(observed))
```

We can now estimate the the differences between the results. We do this both using summary statistics and visualisation.

```
# Calculating a summary of the differences between MCMC and INLA
summary(smoothed_2010$SMR2010_MCMC - smoothed_2010$SMR2010_INLA)
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-0.023460 -0.009477 -0.007517 -0.007914 -0.005504 -0.001533

# Attaching the Smoothed SMRs extimated from INLA to a shapefile
SMRspat_smooth <- combine.data.shapefile(data = smoothed_2010, # Dataset to attach
                                         shp  = shp,       # Shapefile
                                         dbf  = dbf)       # Database file

# Creating a map of 2010 smoothed SMRs extimated from R-INLA
spplot(obj  = SMRspat_smooth,  # Spatial object to be plotted
       xlab = "Easting",            # Label of the x column
       ylab = "Northing",           # Label of the y column
       at   = range,                # Break points for colours
       col  = "transparent",        # Choice of colour
       col.regions = hsv(0.6, seq(0.2, 1, length.out=10), 1)) # Create a set of colours
```



If you have time, you may want to repeat this analysis for another year. Carefully go through the previous sections using any year that you wish, between 2001–2009.