Environmental Health Impact Assessment using R

Mapping Risks

In this session we will use `R` to do some disease mapping. We will work through an example of how to create basic maps in `R` by creating a map of Mexico City. We will then move on to an example of creating smoothed SMRs and plot them on a map by working 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 Mexico City split by municipalities (`cdmx.shp`, `cdmx.dbf`)
- population count and density for Mexico City split by municipalities (`cdmx.csv`)
- 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`).

Preliminaries

For this practical, 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.
- `rgdal` - Package to handle spatial objects.

As in Practical 1, we use the `install.packages()` function to download and install the packages that we need.

```r
# Installing required packages
install.packages("spdep")
install.packages("shapefiles")
install.packages("CARBayes")
install.packages("rgdal")
```

We use the `library()` function to load them into the `R` library.

```r
# Loading required packages into the library
library(spdep)
library(shapefiles)
library(CARBayes)
library(rgdal)
```

Before reading in any data for this practical you will need to ensure that you are in the correct folder. As explained in Practical 1, you can use the `setwd()` function

```r
setwd("Chosen_Directory_Path")
```

If you cannot get the `setwd()` to work, go to `Session > Set Working Directory > Choose Directory` in the toolbar on the top.

Remember, more information about any of the functions used here can be found by typing `help(function_name)` or `?function_name` into `R`.

Creating maps of Scotland

To create maps, we use something called 'shapefiles'. Shapefiles contain location, shape, and attributes of geographic features such as country borders. The files `Scotland_County.shp`, and `Scotland_County.dbf` contain the location, shape, and attributes of Scotland by county. These were obtained from http://www.gadm.org. The functions `read.shp()` and `read.dbf()` will read these shapefiles into R.

```
# Reading in borders
shp_Scotland_C <- read.shp(shp.name = "Scotland_County.shp")
dbf_Scotland_C <- read.dbf(dbf.name = "Scotland_County.dbf")
```

The file `Scotland_County.csv` contains the population of Scotland by county and we will use this to create maps. These are in csv format, so we use the `read.csv()` function.

```
# Read population data for Scotland
pop_Scotland_C <- read.csv('Scotland_County.csv', row.names = 1)
```

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

```
# Printing first six rows
head(pop_Scotland_C)
                   Name Pop2016 PopPercChange2016
1         Aberdeenshire  262190              0.09
2              Aberdeen  229840             -0.22
3                 Angus  116520             -0.33
4       Argyll and Bute   87130              0.28
5      Clackmannanshire   51350             -0.02
6 Dumfries and Galloway  149520             -0.10
```

We can see that this dataset contains the following variables:

- `Name` - County name,
- `Pop2016` - Population count in 2016,
- `PopPercChange2016` - Percentage change in population between 2015 and 2016.

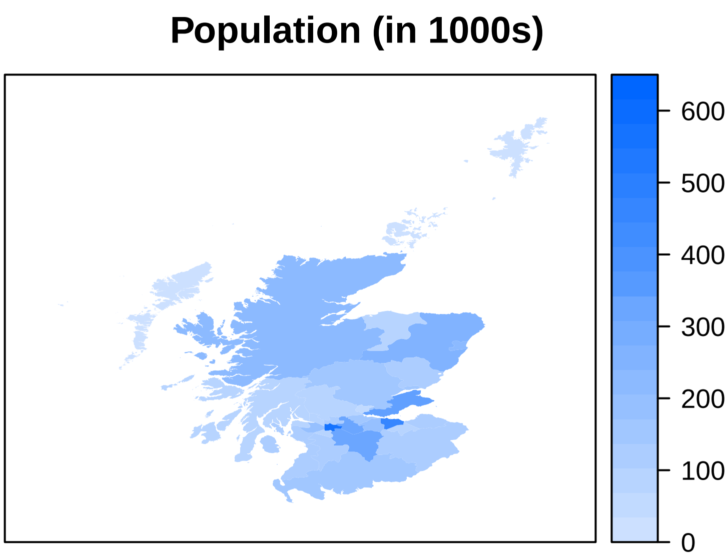
Lets create a map of population in 2016 for Scotland. To plot the population data in a map, we need to attach them to the shapefiles. The function `combine.data.shapefile()` allows us to combine shapefiles to plot later.

```
# Combining population data and the shapefile
Scotland_C <- combine.data.shapefile(data = pop_Scotland_C, # Dataset to attach
                                     shp  = shp_Scotland_C, # Shapefile
                                     dbf  = dbf_Scotland_C) # Database file
```

We use downloaded population data to create the map here. If you have your own data, we can use that later. To plot the map, we use the `spplot()` function.

```
# Scaling population counts (to 1000s
Scotland_C$Pop2016 <- Scotland_C$Pop2016/1000

# Creating map of population counts in Scotland
spplot(obj  = Scotland_C, # Spatial object to be plotted
       zcol = c("Pop2016"), # Choice of the column  the object you are plotting.
       main = "Population (in 1000s)", # Plot title
       at   = seq(0,650, length.out=20), # Break points for legend
       col = 'transparent', # Colour for borders
       col.regions = hsv(0.6, seq(0.2, 1, length.out=20), 1)) # Create a set of colours
```



Activities

- Carefully change the above code to create a map of the percentage change in population between 2015 and 2016 in Scotland.
- In the folder `Data` there are also shapefiles for Scotland split by administrative boundary (`Scotland_Admin.shp`, `Scotland_Admin.dbf`, `Scotland_Admin.csv`). Add your own data to the `Scotland_Admin.csv` file and carefully change the above code to create your own map of Scotland by administrative boundary.

COPD in England

We now look at example into the hospital admission rates for chronic obstructive pulmonary disease (COPD) in England between 2001–2010. In England, there are 324 local authority administrative areas each with an observed and expected number of cases. The expected numbers were calculated using indirect standardisation by applying the age–sex specific rates for the whole of England to the age–sex population profile of each of the areas.

Reading in data and shapefiles

To create SMR maps, 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 these shapefiles into `R`.

```
# Reading in borders
shp <- read.shp(shp.name="englandlocalauthority.shp")
dbf <- read.dbf(dbf.name="englandlocalauthority.dbf")
```

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

```
# Reading in observed numbers of hospital admissions in England by local authority
observed <- read.csv(file="copdmortalityobserved.csv", row.names=1)

# Reading in expected numbers of hospital admissions in England by local authority
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.

```
# Printing first six 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

# Printing first six 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.

```
# 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
```

We can see that `observed` has the following variables:

- `name` - Name of local authority,
- `Y20XX` - Observed number of hospital admissions for COPD in the year 20XX.

We can see that `expected` has the following variables:

- `Y20XX` - Expected number of hospital admissions for COPD (calculated using indirect standardisation) in the year 20XX.

Activities

- Does it look like `R` has read in the data correctly?
- Are there any extreme values in our dataset?
- Can you find which local authorities have the smallest and largest observed counts in England in 2010? HINT: Use `subset()`

Modelling the Raw SMRs

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

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

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

To change attributes of a dataset such as the column names, we use the `names()` function.

```r
# Altering 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
# Printing first six rows of raw SMRs
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 raw SMRs
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?

Mapping the Raw SMRs

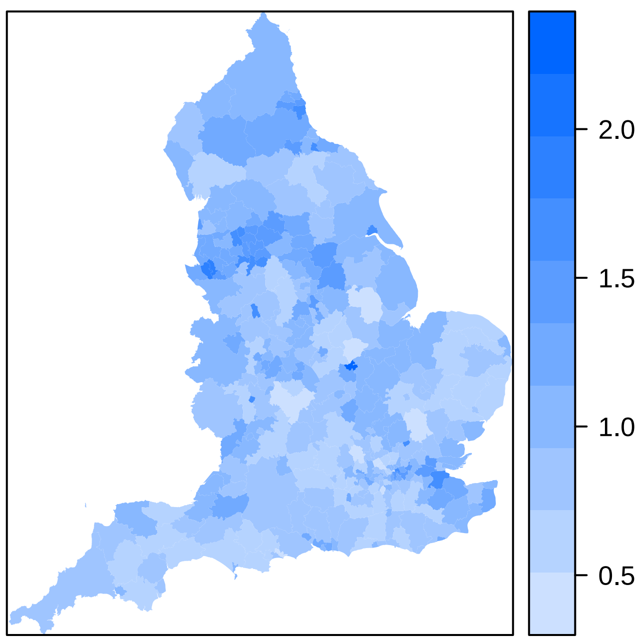
To plot these SMRs to a map, we need to attach them to the shapefiles. The function `combine.data.shapefile()` allows us to combine shapefiles to plot later.

```
# Combining Raw SMRs and the shapefile
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 which colours the local authorities by the SMR estimate.

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

# Creating map of Raw SMRs in England in 2010
spplot(obj  = SMRspatial_raw,    # Spatial object to be plotted
       zcol = c("SMR2010"),  # Choice of the column  the object you are plotting.
       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

- Do you notice anything about this plot?
- Are there any extreme values?
- If so, do you believe that these are the truth or perhaps sampling error?

Modelling the smoothed SMRs

To calculate the smoothed SMRs, we first need to create a 'neighbourhood' structure. The functions `poly2nb()` and `nb2mat()` can be used to create this.

```
# 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.

The function `S.CARleroux()` allows us to use this neighbourhood structure and performs a Bayesian analysis, to create a smoothed set of observed values as discussed in the lecture.

```
# Running smoothing model
model <- S.CARleroux(formula=observed$Y2010~offset(log(expected$E2010)), # 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)
```

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

```
# Creating a dataset with smoothed SMRs in 2010
SMR2010 <- model$fitted.values / expected$E2010
SMR_smooth <- as.data.frame(SMR2010, row.names = rownames(observed))
```

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

```
# Printing first six rows of smoothed SMRs
head(SMR_smooth)
       SMR2010
00AA 0.9969077
00AB 1.2598926
00AC 0.6855420
00AD 0.9729904
00AE 0.5986103
00AF 0.8603609

# Summarising smoothed SMRs
summary(SMR_smooth)
    SMR2010
 Min.   :0.5458
 1st Qu.:0.7944
 Median :0.9229
 Mean   :0.9645
 3rd Qu.:1.0810
 Max.   :1.7414
```

Activities

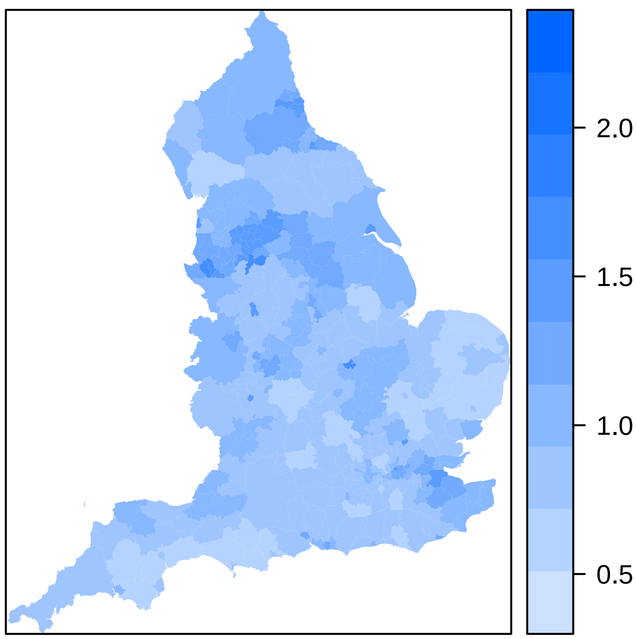
- Does it look like the SMRs have been estimated correctly?
- Are there any extreme values?

Mapping the Smoothed SMRs

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

```
# Combining smoothed SMRs and the shapefile
SMRspatial_smooth <- combine.data.shapefile(data = SMR_smooth, # Dataset to attach
                                            shp  = shp,        # Shapefile
                                            dbf  = dbf)        # Database file

# Creating map of Raw SMRs in England in 2010
spplot(obj  = SMRspatial_smooth,     # Spatial object to be plotted
       zcol = c("SMR2010"),          # Choice of the column  the object you are plotting.
       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 plot?
- Are there any extreme values?
- Are there any differences between the smoothed and raw estimates?

Repeat this analysis for another year to see if the results are similar. Carefully go through the previous sections and change any references from 2010 to any year that you wish between 2001–2009.