Scotland Lip Cancer

Nicholas Bradley

2025-05-11

Introduction

In this analysis I estimate the risk of lip cancer in males in Scotland. I use data on the number of observed and expected lip cancer cases, and the proportion of population engaged in agriculture, fishing, or forestry (AFF) for each of the Scotland counties. First, I describe how to calculate and interpret the standardized incidence ratios (SIRs) in the Scotland counties. Then, I show how to fit the Besag-York-Mollié (BYM) model to obtain relative risk estimates and quantify the effect of the AFF variable. I also show how to calculate exceedance probabilities of relative risk being greater than a given threshold value. Results are shown by means of tables, static plots and interactive maps created with leaflet.

Examine the Data

```
class(Scotland)
```

```
## [1] "list"
```

```
names(Scotland)
```

```
## [1] "geo"             "data"            "spatial.polygon" "polygon"
```

Data Description

Scotland is a list object with the following elements:

geo: data frame with names and centroid coordinates (eastings/northings) of each of the counties, data: data frame with names, number of observed and expected lip cancer cases, and AFF values of each of the counties, spatial.polygon: SpatialPolygons object with the map of Scotland, polygon: polygon map of Scotland.

I can type head(scotland$data) to see the number of observed and expected lip cancer cases and the AFF values of the first counties.

```
head(Scotland$data)
```

```
##    county.names cases expected  AFF
## 1 skye-lochalsh     9      1.4 0.16
## 2  banff-buchan    39      8.7 0.16
## 3     caithness    11      3.0 0.10
## 4  berwickshire     9      2.5 0.24
## 5 ross-cromarty    15      4.3 0.10
## 6        orkney     8      2.4 0.24
```

The map of Scotland counties is given by the SpatialPolygons object called Scotland$spatial.polygon

```
map <- Scotland$spatial.polygon
map <- st_as_sf(map)
```

```
## Loading required package: sp

## Warning: package 'sp' was built under R version 4.3.3
```

```
plot(st_geometry(map))  # correctly plots an sf object
```

map does not contain information about the Coordinate Reference System (CRS), so we specify a CRS by assigning the corresponding proj4 string to the map. The map is in the projection OSGB 1936/British National Grid which has EPSG code 27700. The proj4 string of this projection can be seen in https://spatialreference.org/ref/epsg/27700/. I assign this proj4 string to map and set +units=km because this is the unit of the map projection.

```
map <- Scotland$spatial.polygon
proj4string(map) <- CRS("+proj=tmerc +lat_0=49 +lon_0=-2
  +k=0.9996012717 +x_0=400000 +y_0=-100000 +datum=OSGB36
  +units=km +no_defs")
```

```
## Warning in `proj4string<-`(`*tmp*`, value = new("CRS", projargs = "+proj=tmerc +lat_0=49 +lon_0=-2 +
## +proj=eqc +lat_ts=0 +lat_0=0 +lon_0=0 +x_0=0 +y_0=0 +datum=WGS84 +units=m +no_defs
## without reprojecting.
## For reprojection, use function spTransform
```

```r
map <- st_as_sf(map)  # now convert to sf
```

I wish to use the leaflet package to create maps. leaflet expects data to be specified in latitude and longitude using WGS84, so I transform map to this projection by first converting map from a sp to a sf object, and then using st_transform() to convert the coordinates.

```r
library(sf)
map <- st_as_sf(map)
map <- st_transform(map, crs = 4326)
```

Data Preparation

In order to analyze the data, I create a data frame called d with columns containing the counties ids, the observed and expected number of lip cancer cases, the AFF values, and the SIRs. Specifically, d contains the following columns:

county: id of each county, Y: observed number of lip cancer cases in each county, E: expected number of lip cancer cases in each county, AFF: proportion of population engaged in agriculture, fishing, or forestry, SIR: SIR of each county. I create the data frame d by selecting the columns of scotland$data that denote the counties, the number of observed cases, the number of expected cases, and the variable AFF. Then we set the column names of the data frame to c("county", "Y", "E", "AFF").

```r
d <- Scotland$data[, c("county.names", "cases", "expected", "AFF")]
names(d) <- c("county", "Y", "E", "AFF")
```

I can compute the SIR values as the ratio of the observed to the expected number of lip cancer cases.

```r
d$SIR <- d$Y / d$E
```

The first rows of the data frame d are the following:

```r
head(d)
```

```
##           county  Y   E  AFF      SIR
## 1 skye-lochalsh  9 1.4 0.16 6.428571
## 2  banff-buchan 39 8.7 0.16 4.482759
## 3     caithness 11 3.0 0.10 3.666667
## 4  berwickshire  9 2.5 0.24 3.600000
## 5 ross-cromarty 15 4.3 0.10 3.488372
## 6        orkney  8 2.4 0.24 3.333333
```

Adding data to map

The map with the counties map and the data d contains the Scotland counties in the same order. I join the data to the map as follows:

```r
map <- cbind(map, d)
```

The first part of the data by typing head(map). Here the first column corresponds to the row names of the data, and the rest of the columns correspond to the variables county, Y, E, AFF and SIR.

```r
head(map)
```

```
## Simple feature collection with 6 features and 5 fields
## Geometry type: MULTIPOLYGON
## Dimension:     XY
## Bounding box:  xmin: -6.789691 ymin: 55.63418 xmax: -1.791961 ymax: 59.27929
## Geodetic CRS:  WGS 84
##          county  Y   E  AFF      SIR                       geometry
## 1 skye-lochalsh  9 1.4 0.16 6.428571 MULTIPOLYGON (((-5.493299 5...
## 2  banff-buchan 39 8.7 0.16 4.482759 MULTIPOLYGON (((-2.272216 5...
## 3     caithness 11 3.0 0.10 3.666667 MULTIPOLYGON (((-3.524618 5...
## 4  berwickshire  9 2.5 0.24 3.600000 MULTIPOLYGON (((-2.36695 55...
## 5 ross-cromarty 15 4.3 0.10 3.488372 MULTIPOLYGON (((-4.043716 5...
## 6        orkney  8 2.4 0.24 3.333333 MULTIPOLYGON (((-3.400826 5...
```

Mapping SIRs

I can visualize the observed and expected lip cancer cases, the SIRs, as well as the AFF values in an interactive choropleth map using the leaflet package. I create a map with the SIRs by first calling leaflet() and adding the default OpenStreetMap map tiles to the map with addTiles(). Then I add the Scotland counties with addPolygons() where I specify the areas boundaries color (color) and the stroke width (weight). I fill the areas with the colors given by the color palette function generated with colorNumeric(), and set fillOpacity to a value less than 1 to be able to see the background map. I use colorNumeric() to create a color palette function that maps data values to colors according to a given palette. I create the function using the parameters palette with color function that values will be mapped to, and domain with the possible values that can be mapped. Finally, I add the legend by specifying the color palette function (pal) and the values used to generate colors from the palette function (values). I set opacity to the same value as the opacity in the map, and specify a title and a position for the legend

```r
library(tmap)
```

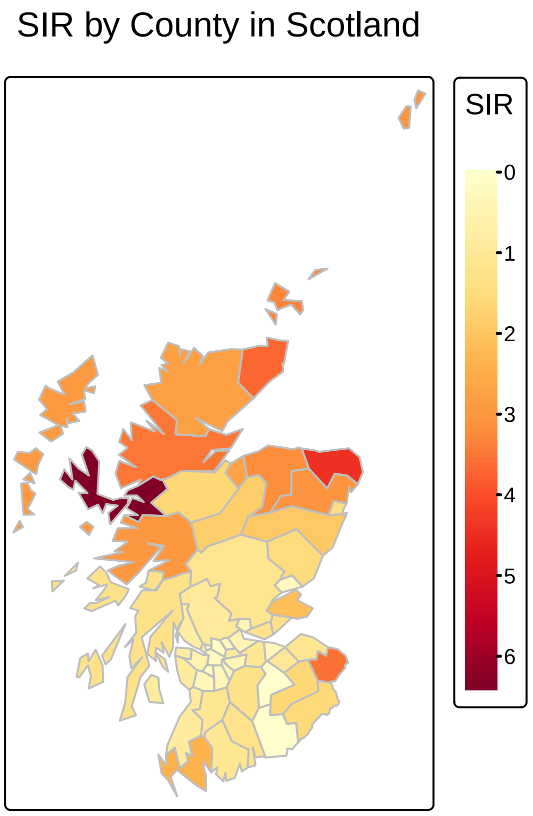
```
## Warning: package 'tmap' was built under R version 4.3.3
```

```r
tmap_mode("plot")  # static for PDF or image export
```

```
## i tmap mode set to "plot".
```

```r
tm_shape(map) +
  tm_fill(
    fill = "SIR",  # column name in your sf object
    fill.scale = tm_scale_continuous(values = "YlOrRd"),
    fill.legend = tm_legend(title = "SIR")
  ) +
  tm_borders(col = "grey") +
  tm_title("SIR by County in Scotland")
```

```
## [cols4all] color palettes: use palettes from the R package cols4all. Run
## `cols4all::c4a_gui()` to explore them. The old palette name "YlOrRd" is named
## "brewer.yl_or_rd"
## Multiple palettes called "yl_or_rd" found: "brewer.yl_or_rd", "matplotlib.yl_or_rd". The first one,
##
## [plot mode] fit legend/component: Some legend items or map compoments do not
## fit well, and are therefore rescaled.
## i Set the tmap option ‘component.autoscale = FALSE‘ to disable rescaling.
```



I can examine the map of SIRs and see which counties in Scotland have SIR equal to 1 indicating observed counts are the same as expected counts, and which counties have SIR greater (or smaller) than 1, indicating observed counts are greater (or smaller) than expected counts. This map gives a sense of the lip cancer risk across Scotland. However, SIRs may be misleading and insufficiently reliable in counties with small populations. In contrast, model-based approaches enable to incorporate covariates and borrow information from neighboring counties to improve local estimates, resulting in the smoothing of extreme values based on small sample sizes. In the next section, I show how to obtain disease risk estimates using a spatial model with the R-INLA package.

Modelling

In this section I specify the model for the data, and detail the required steps to fit the model and obtain the disease risk estimates using R-INLA.

Model

I specify a model assuming that the observed counts, Y_i, are conditionally independently Poisson distributed:

$$Y_i \sim \text{Poisson}(E_i \theta_i), \quad i = 1, \dots, n$$

where E_i is the expected count and θ_i is the relative risk in area i. The logarithm of θ_i is expressed as

$$\log(\theta_i) = \beta_0 + \beta_1 \cdot \text{AFF}_i + u_i + v_i$$

Where β_0 is the intercept that represents the overall risk, β_1 is the coefficient of the AFF covariate, u_i is a spatial structured component modelled with a CAR distribution $u_i \mid u_{-i} \sim \mathcal{N}\left(\bar{u}_{\delta_i}, \frac{\sigma_u^2}{n_{\delta_i}}\right)$, and v_i is an unstructured spatial effect defined as $v_i \sim \mathcal{N}(0, \sigma_v^2)$. The relative risk $theta_i$, quantifies whether area i has higher $\theta_i > 1$ or lower $\theta_i < 1$ risk than the average risk in the population

Neighbourhood Matrix

I create the neighborhood matrix needed to define the spatial random effect using the poly2nb() and the nb2INLA() functions of the spdep package. First, I use poly2nb() to create a neighbors list based on areas with contiguous boundaries. Each element of the list nb represents one area and contains the indices of its neighbors. For example, nb[[2]] contains the neighbors of area 2.

```r
library(spdep)
```

```
## Loading required package: spData

## Warning: package 'spData' was built under R version 4.3.3

## To access larger datasets in this package, install the spDataLarge
## package with: 'install.packages('spDataLarge',
## repos='https://nowosad.github.io/drat/', type='source')'
```

```r
library(INLA)
```

```
## Warning: package 'INLA' was built under R version 4.3.3

## Loading required package: Matrix

## This is INLA_24.03.29 built 2024-03-29 13:04:47 UTC.
##  - See www.r-inla.org/contact-us for how to get help.
##  - List available models/likelihoods/etc with inla.list.models()
##  - Use inla.doc(<NAME>) to access documentation

##
## Attaching package: 'INLA'

## The following object is masked _by_ '.GlobalEnv':
##
##     Scotland
```

```r
nb <- poly2nb(map)
```

```
## Warning in poly2nb(map): some observations have no neighbours;
## if this seems unexpected, try increasing the snap argument.

## Warning in poly2nb(map): neighbour object has 4 sub-graphs;
## if this sub-graph count seems unexpected, try increasing the snap argument.
```

```
head(nb)
```

```
## [[1]]
## [1]  5  9 19
##
## [[2]]
## [1]  7 10
##
## [[3]]
## [1] 12
##
## [[4]]
## [1] 18 20 28
##
## [[5]]
## [1]  1 12 19
##
## [[6]]
## [1] 0
```

Then, I use nb2INLA() to convert this list into a file with the representation of the neighborhood matrix as required by R-INLA. Then I read the file using the inla.read.graph() function of R-INLA, and store it in the object g which I will later use to specify the spatial random effect.

```
nb2INLA("map.adj", nb)
g <- inla.read.graph(filename = "map.adj")
```

Inference Using INLA

The model includes two random effects, namely, u_i for modeling the spatial residual variation, and v_I for modeling unstructured noise. I need to include two vectors in the data that denote the indices of these random effects. I call idareau the indices vector for u_i, and idareav the indices vector for v_I. I set both idareau and idareav equal to $1, ...n$, where n is the number of counties. In the example, $n = 56$ and this can be obtained with the number of rows in the data (nrow(map)).

```
map$idareau <- 1:nrow(map)
map$idareav <- 1:nrow(map)
```

I specify the model formula by including the response in the left-hand side, and the fixed and random effects in the right-hand side. The response variable is Y and I use the covariate AFF. Random effects are defined using $f()$ with parameters equal to the name of the index variable and the chosen model. For u_i, I use model = "besag" with neighborhood matrix given by g. I also use option scale.model = TRUE to make the precision parameter of models with different CAR priors comparable. For v_i, i chose model = "iid".

```
formula <- Y ~ AFF +
  f(idareau, model = "besag", graph = g, scale.model = TRUE) +
  f(idareav, model = "iid")
```

I fit the model by calling the inla() function. I specify the formula, family ("poisson"), data, and the expected counts (E). I also set control.predictor equal to list(compute = TRUE) to compute the posteriors of the predictions.

```r
res <- inla(formula,
  family = "poisson", data = map, E = E,
  control.predictor = list(compute = TRUE),
  control.compute = list(return.marginals.predictor = TRUE)
)
```

Results

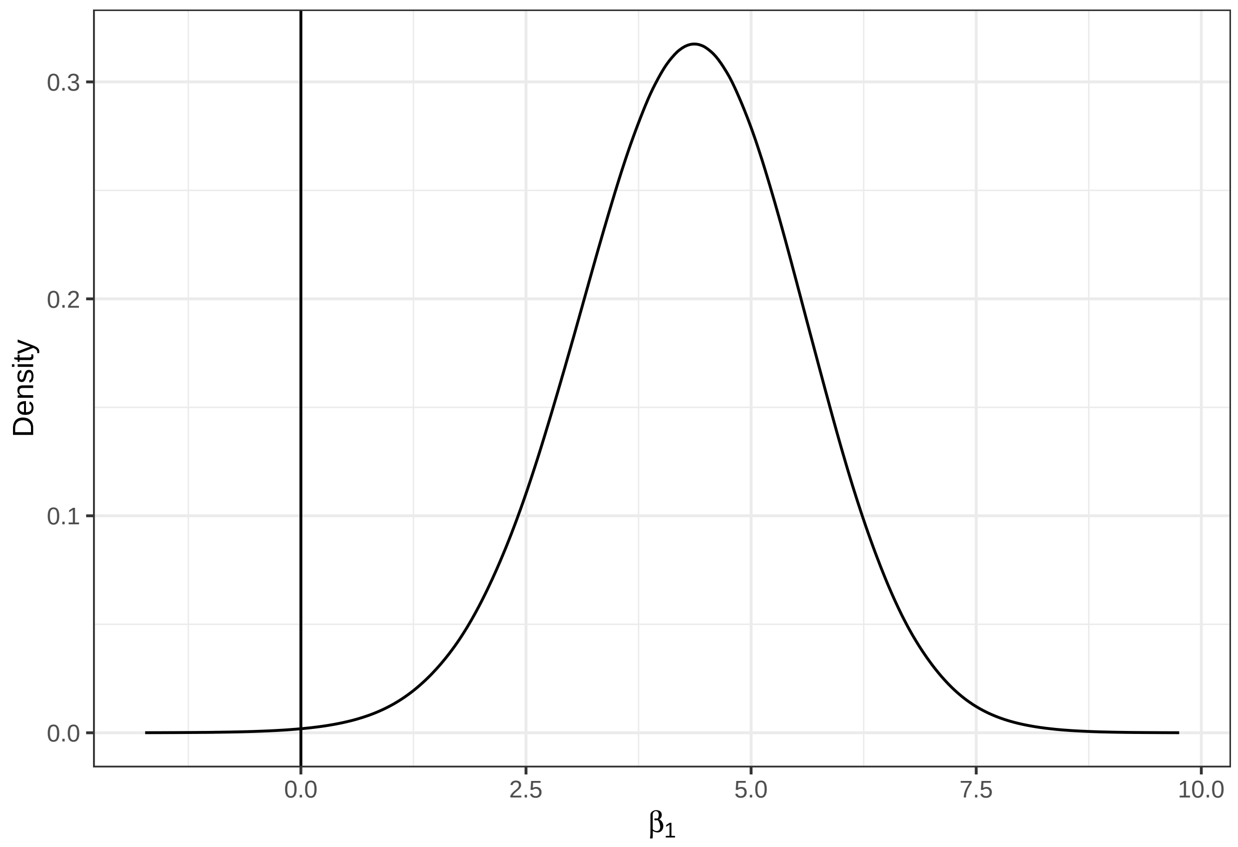
I can inspect the results object res by typing summary(res).

```r
summary(res)
```

```
## Time used:
##     Pre = 0.353, Running = 0.278, Post = 0.32, Total = 0.951
## Fixed effects:
##               mean    sd 0.025quant 0.5quant 0.975quant   mode kld
## (Intercept) -0.306 0.120     -0.538   -0.306     -0.068 -0.306   0
## AFF          4.309 1.277      1.740    4.329      6.761  4.329   0
##
## Random effects:
##   Name     Model
##     idareau Besags ICAR model
##    idareav IID model
##
## Model hyperparameters:
##                             mean       sd 0.025quant 0.5quant 0.975quant    mode
## Precision for idareau     4.17     1.44       2.07     3.93       7.68    3.49
## Precision for idareav 21889.53 22068.48    1877.11 15190.03   80499.13 5293.69
##
## Marginal log-Likelihood:  -189.69
##  is computed
## Posterior summaries for the linear predictor and the fitted values are computed
## (Posterior marginals needs also 'control.compute=list(return.marginals.predictor=TRUE)')
```

I observe the intercept $\hat{beta}_0$ = -0.305 with a 95% credible interval equal to (-0.5386, -0.0684), and the coefficient of AFF is $\hat{beta}_1 = 4.330$ with a 95% credible interval equal to (1.7435, 6.7702). This indicates that AFF increases lip cancer risk. I can plot the posterior distribution of the AFF coefficient by first calculating a smoothing of the marginal distribution of the coefficient with inla.smarginal(), and then using the ggplot() function of the ggplot2 package

```r
library(ggplot2)
marginal <- inla.smarginal(res$marginals.fixed$AFF)
marginal <- data.frame(marginal)
ggplot(marginal, aes(x = x, y = y)) + geom_line() +
  labs(x = expression(beta[1]), y = "Density") +
  geom_vline(xintercept = 0, col = "black") + theme_bw()
```



Mapping Relative Risks

The estimates of the relative risk of lip cancer and their uncertainty for each of the counties are given by the mean posterior and the 95% credible intervals which are contained in the object res$summary.fitted.values. Column mean is the mean posterior and 0.025quant and 0.975quant are the 2.5 and 97.5 percentiles, respectively.

```
head(res$summary.fitted.values)
```

```
##                          mean        sd 0.025quant 0.5quant 0.975quant     mode
## fitted.Predictor.01 4.902162 1.4662225   2.670577 4.688646   8.360921 4.294729
## fitted.Predictor.02 4.354579 0.6774164   3.178333 4.302484   5.827964 4.200188
## fitted.Predictor.03 3.503599 1.0226183   1.920782 3.362958   5.893038 3.098957
## fitted.Predictor.04 3.009818 0.8996793   1.631739 2.881594   5.124480 2.643587
## fitted.Predictor.05 3.273662 0.7539202   2.046451 3.188881   4.986116 3.026870
## fitted.Predictor.06 2.885776 0.9260771   1.490383 2.746828   5.080702 2.491782
```

I add these data to map to be able to create maps of these variables. I assign mean to the estimate of the relative risk, and 0.025quant and 0.975quant to the lower and upper limits of 95% credible intervals of the risks.

```
map$RR <- res$summary.fitted.values[, "mean"]
map$LL <- res$summary.fitted.values[, "0.025quant"]
map$UL <- res$summary.fitted.values[, "0.975quant"]
```

Then, I show the relative risk of lip cancer in an interactive map using leaflet. In the map, I add labels that appear when the mouse hovers over the counties showing information about observed and expected counts, SIRs, AFF values, relative risks, and lower and upper limits of 95% credible intervals. The map created is shown in Figure 6.5. I observe counties with greater lip cancer risk are located in the north of Scotland, and counties with lower risk are located in the center. The 95% credible intervals indicate the uncertainty in the risk estimates.

```r
library(tmap)

library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.3.3
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

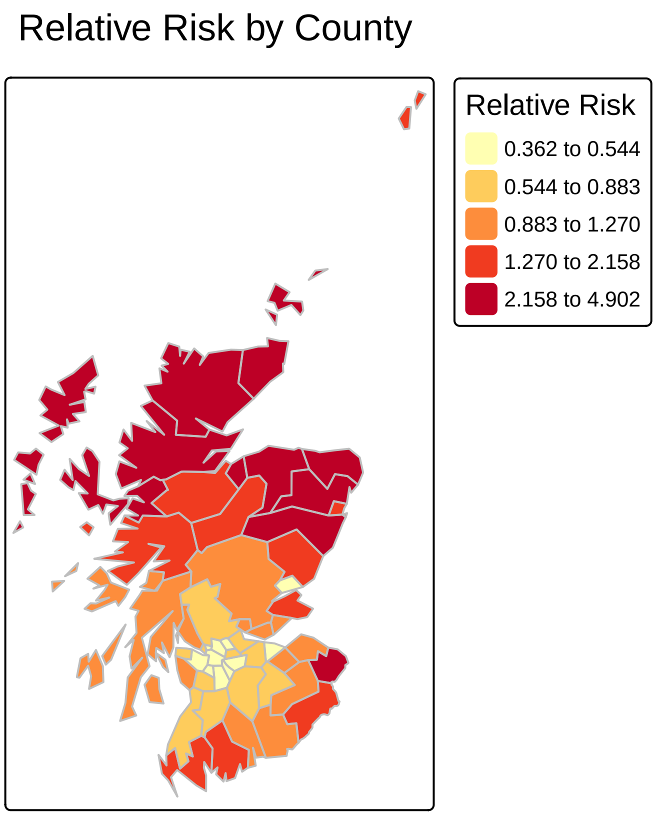
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
# Make sure map_clean does not include NA
map_clean <- map %>% dplyr::filter(!is.na(RR))

tmap_mode("plot")
```

```
## i tmap mode set to "plot".
```

```r
tm_shape(map_clean) +
  tm_fill(
    fill = "RR",
    fill.scale = tm_scale_intervals(
      values = "brewer.yl_or_rd",
      n = 5,
      style = "quantile"
    ),
    fill.legend = tm_legend(title = "Relative Risk")
  ) +
  tm_borders(col = "grey") +
  tm_title("Relative Risk by County")
```



Exceedance Probabilities

I can also calculate the probabilities of relative risk estimates being greater than a given threshold value. These probabilities are called exceedance probabilities and are useful to assess unusual elevation of disease risk. The probability that the relative risk of area 1 is higher than a value c can be written as $P(\theta_i > c)$. This probability can be calculated by substracting $P(\theta_i \leq c)$ to 1 as follows:

$$P(\theta_i > c) = 1 - P(\theta_i \leq c)$$

In R-INLA, the probability $P(\theta_i \leq c)$ can be calculated using the inla.pmarginal() function with arguments equal to the marginal distribution of $theta_i$ and the threshold value c. Then, the exceedance probability $P(\theta_i > c)$ can be calculated by substracting this probability to 1. Marg is the marginal distribution of the predictions, and c is the threshold value. The marginals of the relative risks are in the list res$marginals.fitted.values, and the marginal corresponding to the first county is res$marginals.fitted.values[[1]]. In the example, I can calculate the probability that the relative risk of the first county exceeds 2, $P(\theta_i > 2)$ as follows:

```
marg <- res$marginals.fitted.values[[1]]
1 - inla.pmarginal(q = 2, marginal = marg)
```

```
## [1] 0.9985181
```

To calculate the exceedance probabilities for all counties, I can use the sapply() function passing as arguments the list with the marginals of all counties (res$marginals.fitted.values), and the function to calculate the exceedance probabilities *inla.pmarginal()).sapply()returnsavectorofthesamelengthasthelistres*marginals.fitted.values with values equal to the result of applying the function 1- inla.pmarginal() to each of the elements of the list of marginals.

```
exc <- sapply(res$marginals.fitted.values,
FUN = function(marg){1 - inla.pmarginal(q = 2, marginal = marg)})
```

Then I can add the exceedance probabilities to the map and create a map of the exceedance probabilities with tmap.

```
map$exc <- exc
```

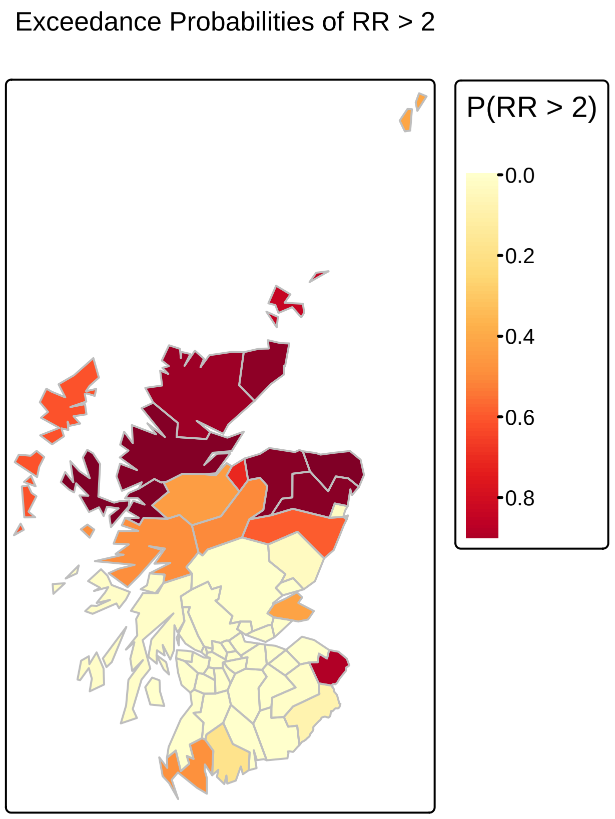
```
# Static plotting mode
tmap_mode("plot")
```

```
## i tmap mode set to "plot".
```

```
# Ensure no missing values in 'exc'
map_clean <- map %>% dplyr::filter(!is.na(exc))

tm_shape(map_clean) +
  tm_fill(
    fill = "exc",
    fill.scale = tm_scale_continuous(
      values = "brewer.yl_or_rd"
    ),
    fill.legend = tm_legend(title = "P(RR > 2)")
  ) +
  tm_borders(col = "grey") +
  tm_title("Exceedance Probabilities of RR > 2")
```

```
## [plot mode] fit legend/component: Some legend items or map compoments do not
## fit well, and are therefore rescaled.
## i Set the tmap option ‘component.autoscale = FALSE‘ to disable rescaling.
```



This map provides evidence of excess risk within individual areas. In areas with probabilities close to 1, it is very likely that the relative risk exceeds 2, and areas with probabilities close to 0 correspond to areas where it is very unlikely that the relative risk exceeds 2. Areas with probabilities around 0.5 have the highest uncertainty, and correspond to areas where the relative risk is below or above 2 with equal probability. I observe that the counties in the north of Scotland are the counties where it is most likely that relative risk exceeds 2.