IWR Summer School 2024 - Statistical Methods Computer Tutorial

Jerome Baron, Prasad Liyanage, Pascale Stiles (University Hospital Heidelberg, HIGH)

2024-09-24

Table of Contents

Prerequisites

For this tutorial you need to install the following software:

- R: https://www.r-project.org/
- RStudio: https://posit.co/download/rstudio-desktop/

We will be using the following R packages:

- dlnm
- dplyr
- ggplot2
- INLA
- mvmeta
- rgdal
- sf
- spdep
- splines

Packages only need to be installed once, but they need to be loaded at the start of each new session.

```r
install.packages("package_name") # only the first time
library(package_name)
```

The INLA package has a special installation, but then it is loaded like all the other packages.

```r
install.packages("INLA", repos = c(getOption("repos"), INLA =
"https://inla.r-inla-download.org/R/stable"), dep = TRUE) # only the first
time
library(INLA)
```

Part 1: Introduction

This tutorial will investigate the epidemiology and drivers of dengue cases in a district of Sri Lanka. First, we will use INLA to explore the spatial and temporal distributions of cases in the district. Then, we will investigate the lagged relationship between the Breteau Index (BI) and dengue cases. The tutorial will guide you through loading and exploring the data, fitting INLA models, conducting a Distributed Lag Non-linear Model (DLNM) analysis, and interpreting the results.

Part 1a: Load Packages

```
# modelling
library(dlnm)
library(INLA)
library(mvmeta)
library(splines)

# data management and visualization
library(dplyr)
library(ggplot2)
library(tidyr)

# spatial data
library(rgdal)
library(sf)
library(spdep)
```

Part 1b: Load Data

Understanding the structure of your data is crucial before any analysis. Here, we load a dataset that contains information on dengue cases (response variable) and the Breteau Index (exposure variable) from 2010 to 2018 in 10 sub-district health units. The dataset is organized in the log format which is needed for the meta-analytical approach. This data is a modified version of Dengue case data from the Southwest of Sri Lanka (Kalutara district). The variables are:

- `regnames`: short version of the spatial unit (Divisional secretariat)
- `pop`: population
- `Year`: year of data
- `time`: cumulative month numerical ID
- `season`: non-cumulative month numerical ID
- `cases`: number of dengue cases
- `PI`: Premises Index (number of premises positive for larvae per 100 premises)
- `BI`: Breteau Index (number of immature-stage positive containers per 100 premises)

```
dat <- read.csv("dengue_data_mod.csv", header = TRUE)
map <- st_read("RDHS_Kalutara_v2.shp")

## Reading layer `RDHS_Kalutara_v2' from data source
##   `C:\Users\pascale.stiles\Documents\Seafile\IDAlert_WP5_3\INLA
tutorial\RDHS_Kalutara_v2.shp'
```

```
##   using driver `ESRI Shapefile'
## Simple feature collection with 10 features and 6 fields
## Geometry type: POLYGON
## Dimension:     XY
## Bounding box:  xmin: 79.88602 ymin: 6.324922 xmax: 80.38194 ymax: 6.824916
## Geodetic CRS:  WGS 84
```

```
head(dat)
```

```
##   series regnames   pop Year time season cases   PI   BI
## 1      1      Aga 39346 2010    1      1     4  6.7  9.7
## 2      2      Aga 39346 2010    2      2     2 11.0 13.9
## 3      3      Aga 39346 2010    3      3     1 12.9 12.9
## 4      4      Aga 39346 2010    4      4     1 16.9 21.5
## 5      5      Aga 39346 2010    5      5     1 18.7 18.7
## 6      6      Aga 39346 2010    6      6     6 23.0 23.0
```

```
tail(dat)
```

```
##      series regnames    pop Year time season cases PI BI
## 1075   1075      Pan 246975 2018  103      7   146  4  4
## 1076   1076      Pan 246975 2018  104      8    97  5  5
## 1077   1077      Pan 246975 2018  105      9    32  3  3
## 1078   1078      Pan 246975 2018  106     10    33  5  5
## 1079   1079      Pan 246975 2018  107     11   100  3  4
## 1080   1080      Pan 246975 2018  108     12   146  3  4
```

Part 2: Exploratory Data Analysis

Part 2a: Seasonal Summary

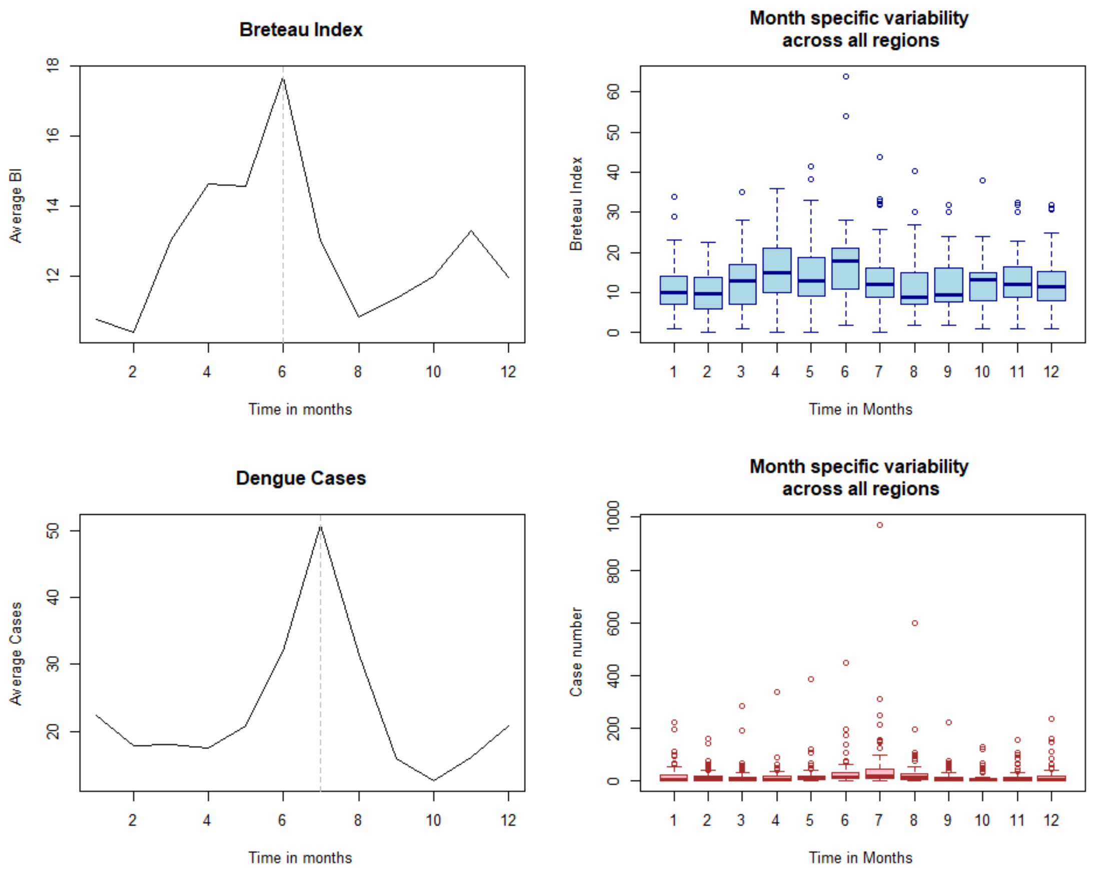
We first summarize the data by calculating the average number of dengue cases and BI values for each month to demonstrate the seasonality of the data.

```
seasonal_summary <- dat %>%
  group_by(season) %>%
  summarize(
    cases = mean(cases, na.rm = TRUE),
    BI = mean(BI, na.rm = TRUE)
  )
seasonal_summary
```

```
## # A tibble: 12 × 3
##    season cases    BI
##     <int> <dbl> <dbl>
##  1      1  22.3  10.8
##  2      2  17.8  10.4
##  3      3  18.0  13.0
##  4      4  17.4  14.6
##  5      5  20.9  14.6
##  6      6  32    17.7
##  7      7  50.9  13.0
##  8      8  31.5  10.8
## 9       9  16.0  11.4
## 10     10  12.5  12.0
## 11     11  16.0  13.3
## 12     12  20.9  12.0
```

We visualize the seasonal patterns of the Breteau Index and dengue cases. This is important to understand the temporal trends and potential lag dependencies of two variables.

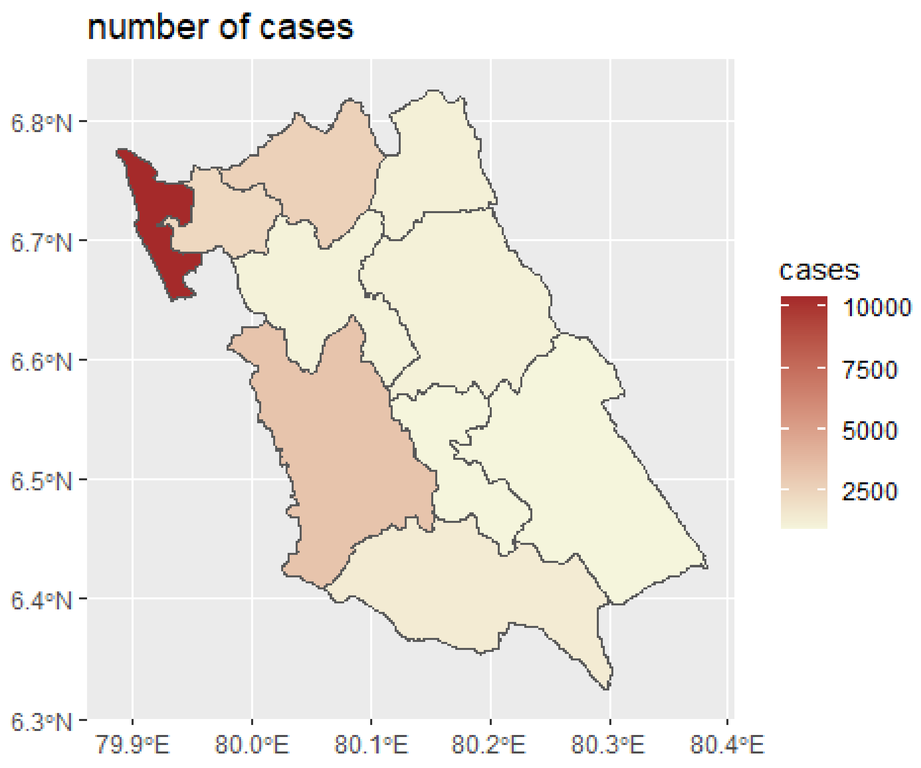
```
par(mfrow = c(2, 2))
plot(seasonal_summary$BI, type = "l",
     xlab = "Time in months",
     ylab = "Average BI",
     main = "Breteau Index")
abline(v = 6, lty = 2, col = "gray")

boxplot(BI ~ season,
        data = dat,
        main = "Month specific variability
across all regions",
        xlab = "Time in Months",
        ylab = expression(paste("Breteau Index")),
        col = "lightblue",
        border = "darkblue")

plot(seasonal_summary$cases, type = "l",
     xlab = "Time in months",
     ylab = "Average Cases",
     main = "Dengue Cases")
abline(v = 7, lty = 2, col = "gray")

boxplot(cases ~ season,
        data = dat,
        main = "Month specific variability
across all regions",
        xlab = "Time in Months",
        ylab = expression(paste("Case number")),
        col = "pink",
        border = "brown")
```



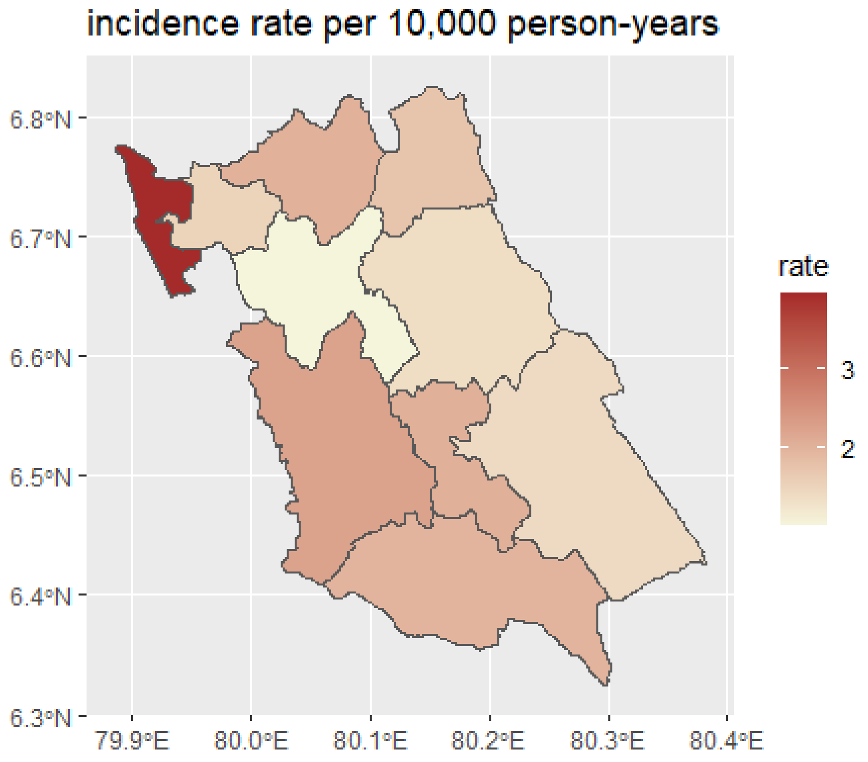
Part 2b: Spatial Summary

Mapping out the cases helps us identify where there are clusters of cases.

```
dat_agg <- aggregate(dat[, c("cases", "pop")], by = list(dat$regnames), FUN =
sum)
dat_agg$rate <- dat_agg$cases * 10000 / dat_agg$pop
map2 <- merge(map, dat_agg, by.x = "reg_names", by.y = "Group.1")

ggplot(map2) +
  geom_sf(aes(fill = cases)) +
  scale_fill_gradient(low = "beige", high = "brown") +
  ggtitle("number of cases")
```



```r
ggplot(map2) +
  geom_sf(aes(fill = rate)) +
  scale_fill_gradient(low = "beige", high = "brown") +
  ggtitle("incidence rate per 10,000 person-years")
```



Part 3: Fixed and Mixed Effect Models in INLA

Part 3a: Model Fitting and Likelihood Families

Based on the exploratory data analysis, we can see that there may be some temporal trends to account for in the dengue case data. We'll start with a basic example using INLA. INLA models are formulated with syntax very similar to GLM or GLMER models in R.

```
formula <- outcome ~ variable1 + variable2 + ... + f(random_effect, model =
"functionname")

model <- inla(formula,
              data = dataframe, # define the data
              family = "familyname", # set the likelihood family
              # Default model family is "gaussian"
              # Other families in epidemiology include "binomial", "poisson",
"nbinomial", "zeroinflated..."
              # Depending on the model you might need to add other arguments
such as:
                # Ntrials = n for a binomial model
                # offset = n for count data models with a population
              control.predictor = list(compute = TRUE), # computes fitted
values (predictions)
              control.compute = list(dic = TRUE) # computes DIC as the model
fit parameter
              # Other model fit parameter options are waic and cpo
              )
```

In this first step, we will consider a simple fixed effect model using month as the fixed effect and number of cases as the outcome. We will look at two different likelihood families. These will be the Poisson and negative binomial families, which are commonly used for count data. The `summary()` function gives you an overall summary of the model with parameters for the variables and model fit. You can see the full list of model likelihood families by calling `names(inla.models()$likelihood)` and details about each family can be seen by calling `inla.doc("familyname")`. Similarly, you can see the full list of random effect functions by calling `names(inla.models()$latent)`, with details about each function at `inla.doc("funtionname")`.

```
formula1 <- cases ~ as.factor(season)
m.inla.1 <- inla(formula1,
                 offset = log(pop),
                 data = dat,
                 family = "poisson",
                 control.predictor = list(compute = TRUE),
                 control.compute = list(dic = TRUE))
# summary(m.inla.1)

formula2 <- cases ~ as.factor(season)
m.inla.2 <- inla(formula2,
                 offset = log(pop),
                 data = dat,
                 family = "nbinomial",
                 control.predictor = list(compute = TRUE),
                 control.compute = list(dic = TRUE))
# summary(m.inla.2)
```

The DIC is interpreted in the same way as the AIC, where a lower value indicates a better-fitting model.

```
m.inla.1$dic$dic

## [1] 10766.71

m.inla.2$dic$dic

## [1] 8199.496
```

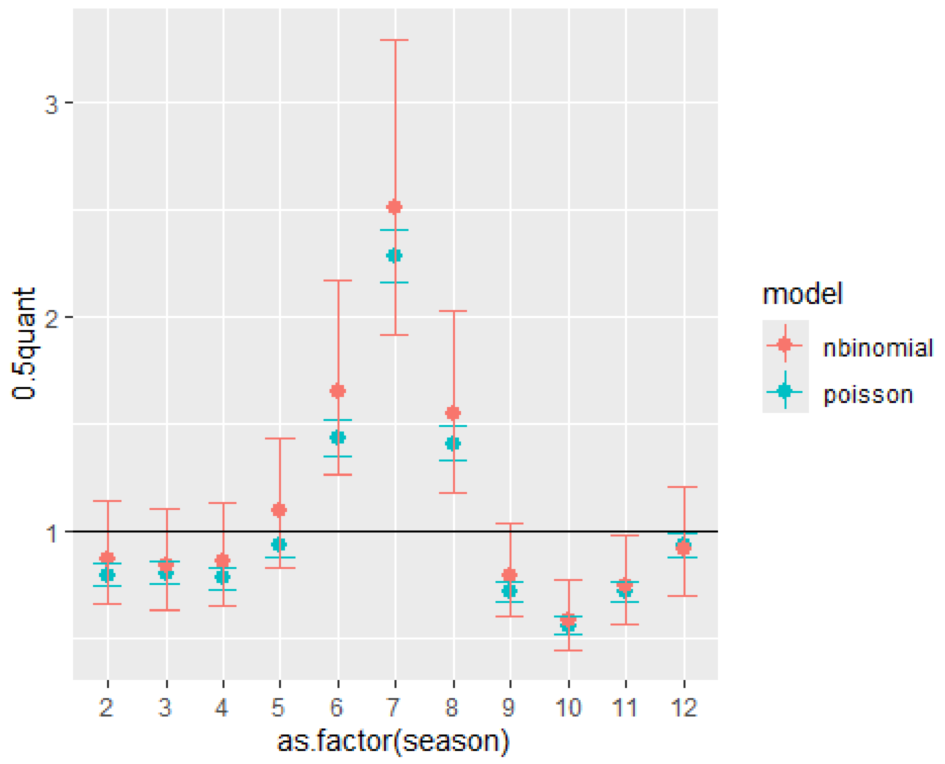
Here we can see from the DICs that the negative binomial model fits better. However, it is good practice to check which distribution fits your outcome data best before proceeding with modelling. This can be done using functions from the `fitdistrplus` package such as: `descdist()` and `fitdist()`.

Part 3b: Interpreting Fixed Effects

Remember that Bayesian models do not yield p-values but rather a posterior distribution of likely values the parameters will take. INLA produces a summary of the posterior estimates for each fixed effect coefficient, including the intercept, which is stored in `summary.fixed`. The median and 95% credible interval (2.5% and 97.5% percentiles) are presented here. The marginal values are stored in `marginals.fixed`, which is a named list with the values for each coefficient, where x is the estimated value and y is the density of that value. The summary and marginals for random effects, predictions, and hyperparameters are also stored like this.

We can plot the median and 95% credibility interval for the fixed effect (we'll remove `season == 1` from the figure as this is actually the intercept or referent group, not the effect for the month of January). The parameter values are exponentiated to be more interpretable as rate ratios. So for instance, `season == 7` (July) has a 2.5 times higher rate of cases compared to the reference of `season == 1` (January). In exponentiated form, values above 1 imply higher rates, and values below 1 imply lower rates. If the exponentiated 95% credible interval includes one, we would consider this a non-significant difference from the referent category (e.g., `season == 5`).

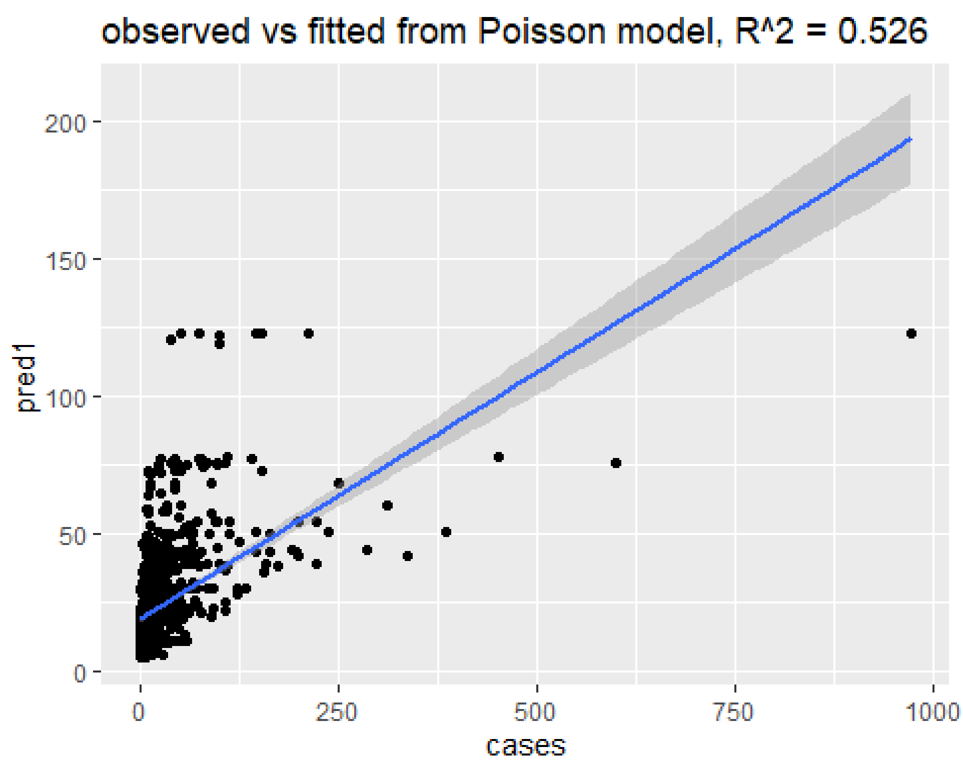
```
x1 <- as.data.frame(exp(m.inla.1$summary.fixed))
x1$season <- seq(1:12)
x1$model <- "poisson"
x2 <- as.data.frame(exp(m.inla.2$summary.fixed))
x2$season <- seq(1:12)
x2$model <- "nbinomial"

x <- rbind(x1, x2)

ggplot(x[x$season != 1,], aes(x = as.factor(season), y = `0.5quant`, ymin =
`0.025quant`, ymax = `0.975quant`)) +
  geom_pointrange(aes(col = model)) +
  geom_errorbar(width = 0.5, aes(col = model)) +
  geom_line(aes(col = model)) +
  geom_hline(aes(yintercept = 1))
```



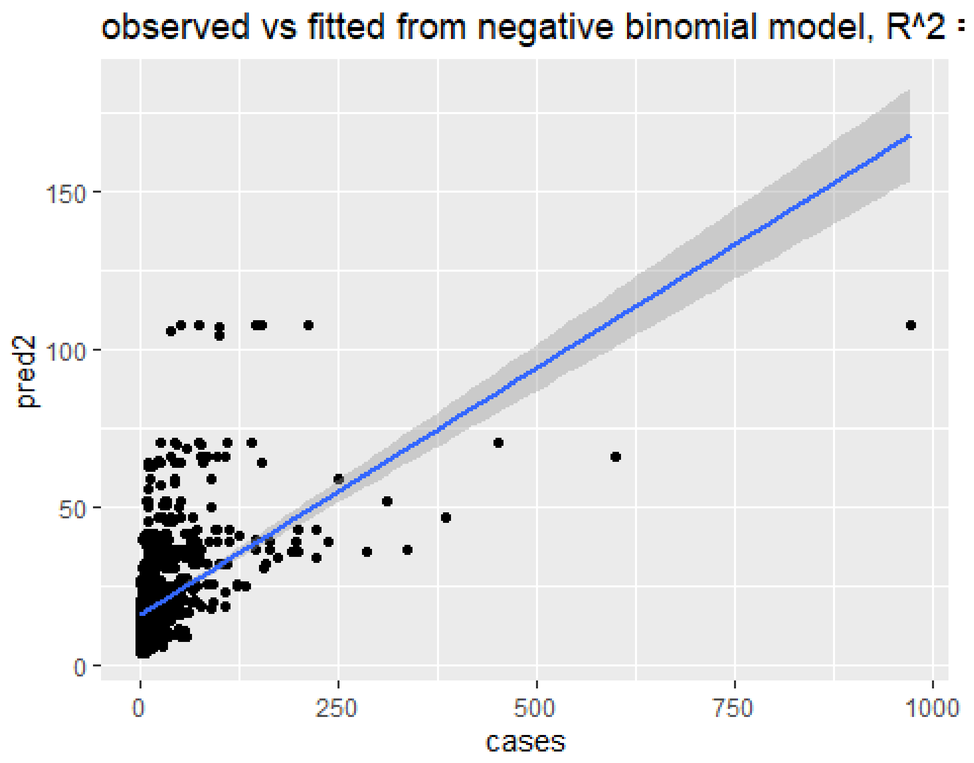
Part 3c: Fitted vs. Predicted Values

We can obtain estimates of the predicted values in the `summary.fitted.values` tab of the model object. These can be compared to the observed values to assess the predictive performance of a model. With binomial models, this can be done through ROC curve analysis. In this case, we will use simple correlation between the median fitted values of both models and the observed values. We'll round them as these are case data, so decimals don't make logical sense.

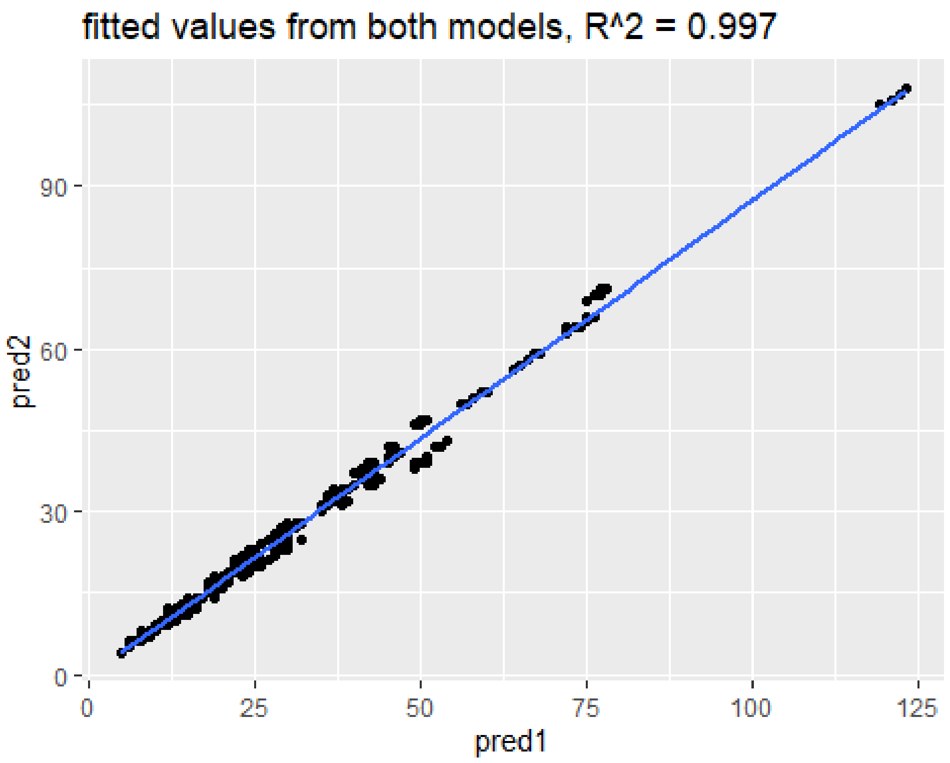
```
dat$pred1 <- round(m.inla.1$summary.fitted.values$`0.5quant`, 0)
dat$pred2 <- round(m.inla.2$summary.fitted.values$`0.5quant`, 0)

ggplot(dat, aes(cases, pred1)) +
  geom_point() +
  geom_smooth(method="lm") +
  ggtitle(paste0("observed vs fitted from Poisson model, R^2 = ",
round(cor(dat$cases, dat$pred1), 3)))
```



```
ggplot(dat, aes(cases, pred2)) +
  geom_point() +
  geom_smooth(method="lm") +
  ggtitle(paste0("observed vs fitted from negative binomial model, R^2 = ",
round(cor(dat$cases, dat$pred2), 3)))
```



```
ggplot(dat, aes(pred1, pred2)) +
  geom_point() +
  geom_smooth(method="lm") +
  ggtitle(paste0("fitted values from both models, R^2 = ",
round(cor(dat$pred1, dat$pred2), 3)))
```



As we can see, neither model is very good at predicting the observed outcome, but they come up with nearly identical predictions to each other. These are very simple models which don't account for spatial and/or temporal trends nor any other explanatory variables, so this isn't surprising.

Part 3d: Exercises

Since observations are repeated within each spatial location, we will now add a random effect for the location (`regnames`) in the negative binomial model. This is a simple random effect with no correlation structure.

```r
formula3 <- cases ~ as.factor(season) +
  f(regnames, model = "iid")
m.inla.3 <- inla(formula3,
                 offset = log(pop),
                 data = dat,
                 family = "nbinomial",
                 control.predictor = list(compute = TRUE),
                 control.compute = list(dic = TRUE))
# summary(m.inla.3)
```

1. Compare the DICs of Model 2 and Model 3. Which one indicates a better fit?

Part 4: Temporal Correlation Structures

When looking at the parameters for month as a fixed effect in the models above, there appears to be a seasonal trend. This implies that there are similarities between months that are closer to each other in time than months further apart, which can be better captured by including month as a random effect with a correlation structure. Time can be modeled in two main ways: 1) in a unidirectional way (such as for years or cumulative time) and 2) in a cyclical way (such as for months or seasons). In our data we have 3 time variables: year (2010-2018, variable `Year`), month (1-12, variable `season`), and month-year (1-108, variable `time`). Months are nested within years but can also be considered cyclical if we want to capture seasonality.

Part 4a: Non-Cyclic Temporal Effect

We will start by applying one of the more simple forms of latent effects on the month variable by using an autoregressive model of order 1 (AR1). Other models that can be considered are:

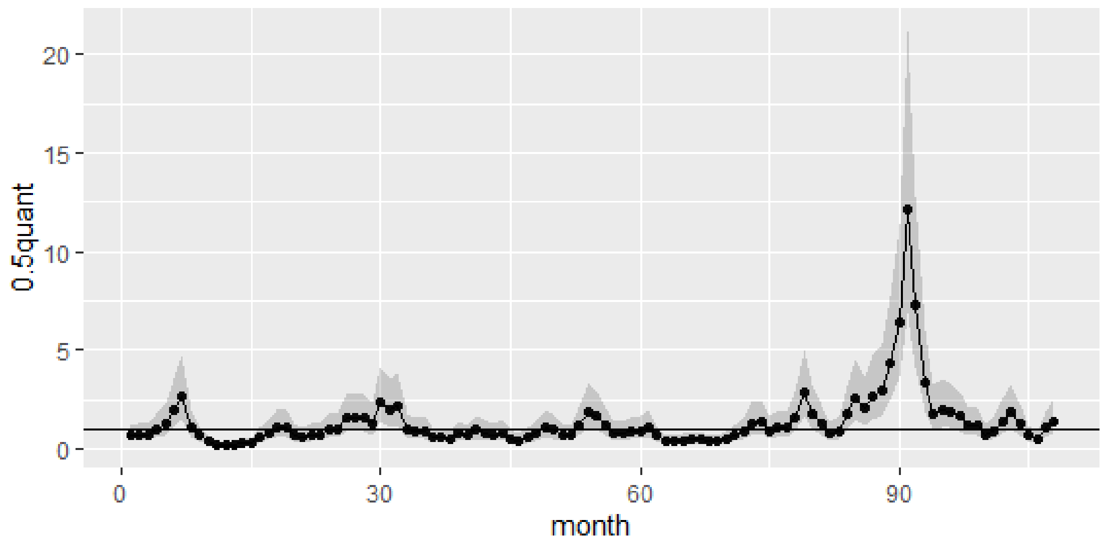
- ARp (autoregressive model of order *p*)
- RW1 (random walk of order 1)
- RW2 (random walk of order 2)

AR and RW function differently mathematically, but essentially they act as smoothing functions by relating the value of time t to past time points (the order being the number of past time points being considered - the higher that is, the more smooth the output will be)

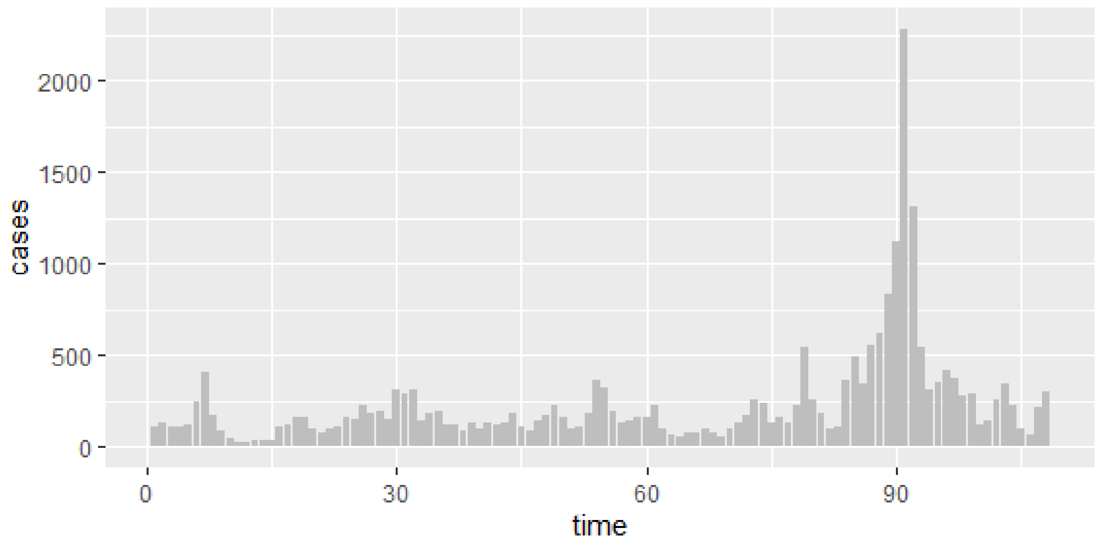
```r
formula4 <- cases ~ 1 +
  f(time, model = "ar1") +
  f(regnames, model = "iid")

m.inla.4 <- inla(formula4,
                 offset = log(pop),
                 data = dat,
                 family = "nbinomial",
                 control.predictor = list(compute = TRUE),
                 control.compute = list(dic = TRUE))
# summary(m.inla.4)
```

Let’s plot the monthly effects and compare them to the number of cases per month:

```r
x4 <- as.data.frame(exp(m.inla.4$summary.random$time))
x4$month <- seq(1:108)
x4$model <- "ar1"

ggplot(x4, aes(x = month, y = `0.5quant`, ymin = `0.025quant`, ymax =
`0.975quant`)) +
  geom_point() +
  geom_ribbon(alpha = 0.2) +
  geom_line() +
  geom_hline(aes(yintercept = 1))
```



```r
ggplot(dat, aes(time, cases)) +
  geom_col(fill = "grey")
```



This AR1 model tends to overfit the data. The temporal effect nearly exactly matches how cases are distributed in time, and doesn't really give us an interpretation as to how time is related to cases.

Part 4b: Exercises

1. Please fit a model where the temporal term for `time` follows non-cyclic random walk of order 1. Which model fits better?
2. What criteria did you use to make this determination?

Part 4c: Cyclic Temporal Effect

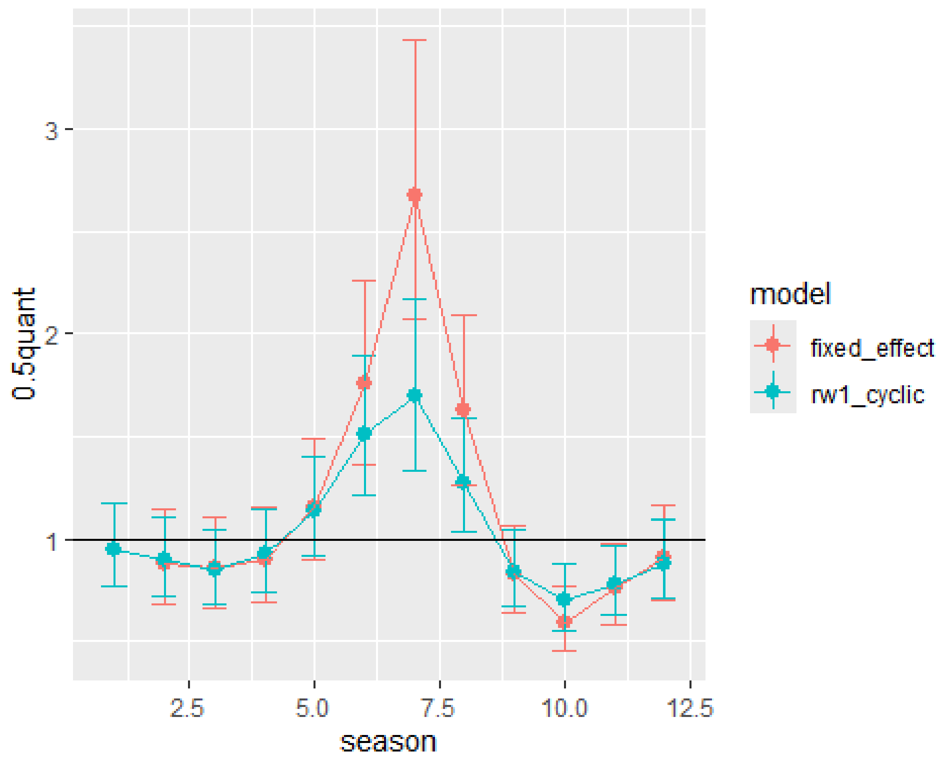
However, if we want to capture seasonality, we need to consider the cyclic nature of time (i.e., December is a neighbor of January). We also have to consider that month is nevertheless nested within year (so essentially this is to account for the fact that although

June 2010 and June 2018 are both the month of June from a seasonal perspective, they are different individual periods in time, and there is also a year effect). To do this we have a cyclical structured random effect (using random walk of order 1) on the `season` variable and non-structured random effects on the `Year` and `time` variables.

```
formula5 <- cases ~ 1 +
  f(season, model = "rw1", cyclic = TRUE) +
  f(Year, model = "iid") +
  f(time, model = "iid") +
  f(regnames, model = "iid")

m.inla.5 <- inla(formula5,
                 offset = log(pop),
                 data = dat,
                 family = "nbinomial",
                 control.predictor = list(compute = TRUE),
                 control.compute = list(dic = TRUE))
# summary(m.inla.5)
```

Now we can compare the random effect of `season` to the fixed effect of `season` from Model 3, since both of these also have location as a non-structured random effect. Remember that the fixed effect compares to a referent category, so we need to remove the estimates from the first row as they do not represent estimates for the effect of January but rather the intercept and so it is meaningless for the comparison here.

```
x3 <- as.data.frame(exp(m.inla.3$summary.fixed))
x3$season <- seq(1:12)
x3$model <- "fixed_effect"
x3[1, c(1:7)] <- NA

x5 <- as.data.frame(exp(m.inla.5$summary.random$season))
x5$season <- seq(1:12)
x5$model <- "rw1_cyclic"

x <- rbind(x3, x5[, c(2:10)])

ggplot(x, aes(x = season, y = `0.5quant`, ymin = `0.025quant`, ymax =
`0.975quant`)) +
  geom_pointrange(aes(col = model)) +
  geom_errorbar(width = 0.5, aes(col = model)) +
  geom_line(aes(col = model)) +
  geom_hline(aes(yintercept = 1))
```



More complex structures can be built if you also want to consider and explain the effect of year which can also include a correlation structure.

Part 4d: Exercises

1. Why do you think the seasonal effect is weaker in the cyclical model (Model 5) than when month was a fixed effect (Model 3)?

Part 5: Spatial Correlation Structures

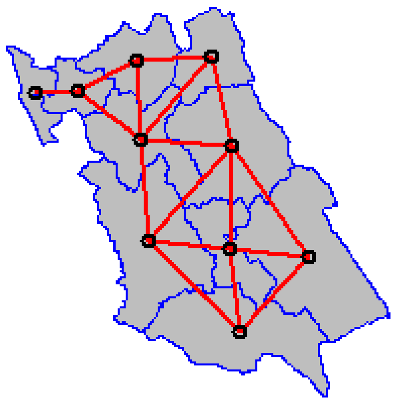
Part 5a: Adjacency Matrix

For spatial models using areal data, we need to create an adjacency matrix so the model understands how the spatial units correspond to each other.

First you create a list using `poly2nb()` that lists the indices of all neighbors for each polygon. We set “queen” contiguity (the default), which means that a single shared point is enough to be considered neighbors (i.e., a corner). This is a first-order equal-weight contact matrix, which means that only direct contacts are considered and all contacts are given the same value. Other contact matrices are possible, such as “rook” (when corner contacts aren’t considered neighbors), second-order (i.e., the neighbor of your neighbor) or weighted (e.g., the length of the shared border, distance between centroids, or using a movement/contact network).

Then, the `nb2INLA()` function takes the adjacency values and creates a graph in the format that INLA requires. We also need to create a numerical spatial ID variable in the data that matches the order of the spatial units in the map.

```
map$ID <- seq(1:nrow(map))
dat2 <- merge(dat, map[, c("reg_names","ID")], by.x = "regnames", by.y =
"reg_names")

map.mat <- poly2nb(map, queen = TRUE)
plot(map$geometry, col = "gray", border = "blue")
xy <- st_coordinates(st_centroid(map))
plot(map.mat, xy, col = "red", lwd = 2, add = TRUE)
```



```
map.mat[[2]] # polygon 2 is a neighbor of polygons 1, 3, 5, and 6 in the
map.mat list

## [1] 1 3 5 6

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

Part 5b: Model Fitting

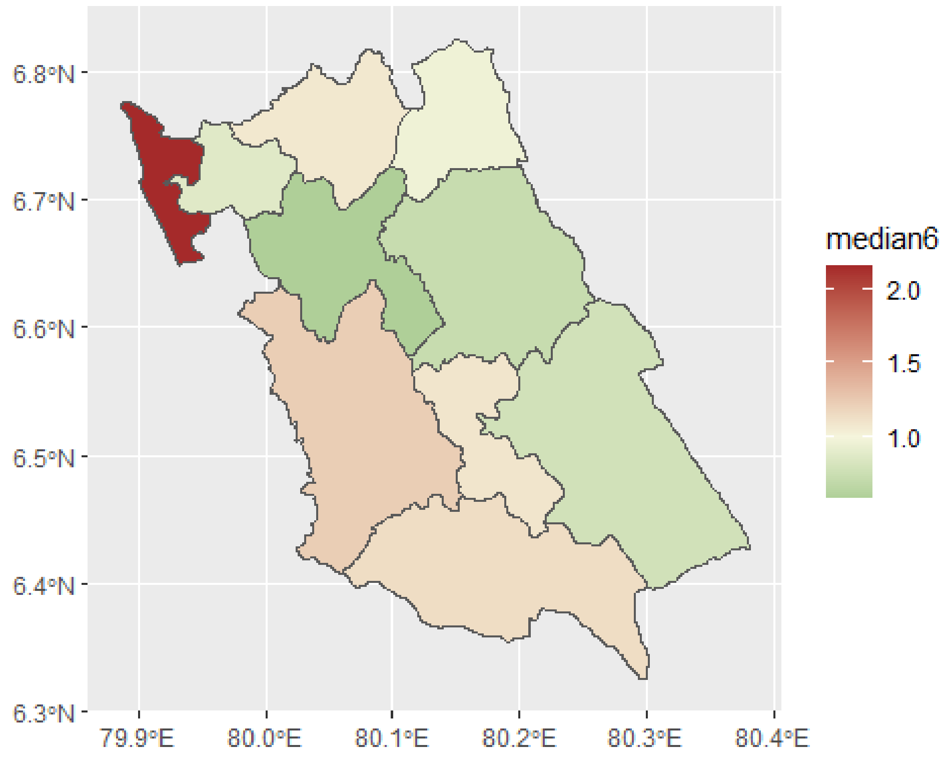
The most common spatial models are `besag` and `bym`. The `besag` model only accounts for the structured spatial correlation, so a second `iid` random effect also needs to be added to account for the unstructured correlation. The `bym` model accounts for both structured and unstructured correlation, so it can be used alone. More information on these models can be found by running `inla.doc("bym")`.

```
formula6 <- cases ~ as.factor(season) +
  f(ID, model = "bym", graph = map.x)
m.inla.6 <- inla(formula6,
                 offset = log(pop),
```

```r
                    data = dat2,
                    family = "nbinomial",
                    control.predictor = list(compute = TRUE),
                    control.compute = list(dic = TRUE))
# summary(m.inla.6)
```

Now let's map the random effects. Because the bym model has both structured and unstructured correlation, the length of the summary.random dataframe is double the length of the original dataframe, with the first half (1:n) corresponding to the structured effects and the second half (n+1:n+n) corresponding to the unstructured effects. Thus you want to map out the first half as it is more interesting to interpret the spatial effect on the outcome. In this map, we see the highest rate ratio in the northwest, which we expected from mapping out the case counts.

```r
x6 <- as.data.frame(exp(m.inla.6$summary.random$ID[1:10, ]))

map$median6 <- exp(m.inla.6$summary.random$ID[1:10,]$`0.5quant`)

ggplot(map) +
  geom_sf(aes(fill = median6)) +
  scale_fill_gradient2(low = "forestgreen", mid = "beige", high = "brown",
midpoint = 1)
```



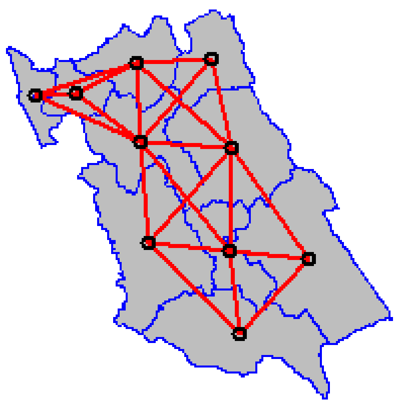
Part 5c: Exercises

1. Now let us create another proximity matrix. This time we will rely not just on direct contact but also distance proximity. Here, if two spatial units are not in direct contact but have boundary points within 5000m of each other, then they are still considered

neighbors. This can be especially useful when you have a large imbalance in spatial unit sizes, or islands that are not in contact with any other location on the map, even if they are close by.

2. Using the new matrix, fit a new model and compare the spatial effect to Model 6.

```
# Change the projection so we can measure the distance in meters
# This is not necessary, but it's easier to interpret than if we left it in a
projection that uses degrees
map_m <- st_transform(map, crs = 5234)

map.mat2 <- poly2nb(map_m, snap = 5000, queen = TRUE)
plot(map_m$geometry, col = "gray", border = "blue")
xy <- st_coordinates(st_centroid(map_m))
plot(map.mat2, xy, col = "red", lwd = 2, add = TRUE)
```



```
map.mat2[[2]] # polygon 9 is added to the neighbors of polygon 2 in the new
map.mat2 list compared to map.mat

## [1] 1 3 5 6 9

nb2INLA("map.adj", map.mat2)
map.x2 <- inla.read.graph(filename = "map.adj")
```

Part 6: Spatio-Temporal Interactions

In this final section, we will combine both spatial and temporal random effects. When this is done, we need to consider the interaction of space and time.

Part 6a: Types of Spatio-Temporal Interactions

Spatio-temporal interactions can take four types. Which type you choose will depend on the hypothesis you are testing and/or prior knowledge of the system you are modelling. We will focus on the simplest Type 1 interaction here, but see below for more detailed information on the four types.

1. Type 1 interactions: This is an interaction between the nonstructured spatial and nonstructured temporal random effects. It is implemented by adding an IID random effect for each unique space-time observation.

```r
formulaI = y ~ x +
  f(spaceID, model = "bym", graph = graph.adj) +
  f(timeID, model = "rw1") +
  f(timeID2, model = "iid") +
  f(spacetimeID, model = "iid")
```

2. Type 2 interactions: This interaction is formed between the structured temporal and unstructured spatial random effects. Each spatial unit observes temporal correlation within the unit, but the temporal correlation structure is independent from the temporal structure in neighboring units. For example, use this spatio-temporal interaction when there is a seasonal effect in each state that is independent of the seasonal effect in the neighboring states. It is implemented by adding a time group to an IID spatial random effect and defining the model of the group to be the same as the original structured temporal random effect.

```r
formulaII = y ~ x +
  f(spaceID, model = "bym", graph = graph.adj) +
  f(timeID, model = "rw1") +
  f(timeID2, model = "iid") +
  f(spaceID2, model = "iid", group = timeID3, control.group = list(model =
"rw1"))
```

3. Type 3 interactions: This interaction is formed between the structured spatial and unstructured temporal random effects. Each time point observes spatial correlation within the time point, but the spatial correlation structure is independent from the spatial structure in neighboring time points. For example, use this spatio-temporal interaction when there is spatial effect between states in a given year that is independent of the spatial effect in previous years. It is implemented by adding a space group to an IID temporal random effect and defining the model of the group to be the same as the original structured spatial random effect.

```r
formulaIII = y ~ x +
  f(spaceID, model = "bym", graph = graph.adj) +
  f(timeID, model = "rw1") +
  f(timeID2, model = "iid") +
  f(timeID3, model = "iid", group = spaceID2, control.group = list(model =
"besag", graph = graph.adj))
```

4. Type 4 interactions: This is the most complicated type of interaction, formed between the structured temporal and structured spatial random effects. The temporal correlation structure within each spatial unit is dependent on the temporal

structure in neighboring spatial units and the spatial structure at each time point is influenced by its neighbors. For example, use this spatio-temporal interaction when there is a seasonal effect in each state that influences the seasonal effects observed in neighboring states. It is implemented by adding a time group to a besag structured spatial random effect and defining the model of the group to be the same as the original structured temporal random effect.

```
formulaIV = y ~ x +
  f(spaceID, model = "bym", graph = graph.adj) +
  f(timeID, model = "rw1") +
  f(timeID2, model = "iid") +
  f(spaceID2, model = "besag", graph = graph.adj,
    group = timeID3, control.group = list(model = "rw1"))
```

Part 6b: Model Fitting

Type 1 interactions are done by creating a new numerical index variable that is a sequence from `1:nrow(data)` and including this as an uncorrelated IID term in the model formula. We also add an random walk model of order 1 to `Year` to represent potential long-term trends.

```
dat2$IDspacetime <- seq(1, nrow(dat2), 1)

formula7 <- cases ~ 1 +
  f(season, model = "rw1", cyclic = TRUE) +
  f(Year, model = "rw1") +
  f(time, model = "iid") +
  f(ID, model = "bym", graph = map.x) +
  f(IDspacetime, model = "iid")

m.inla.7 <- inla(formula7,
                 offset = log(pop),
                 data = dat2,
                 family = "nbinomial",
                 control.predictor = list(compute = TRUE),
                 control.compute = list(dic = TRUE))
# summary(m.inla.7)
```

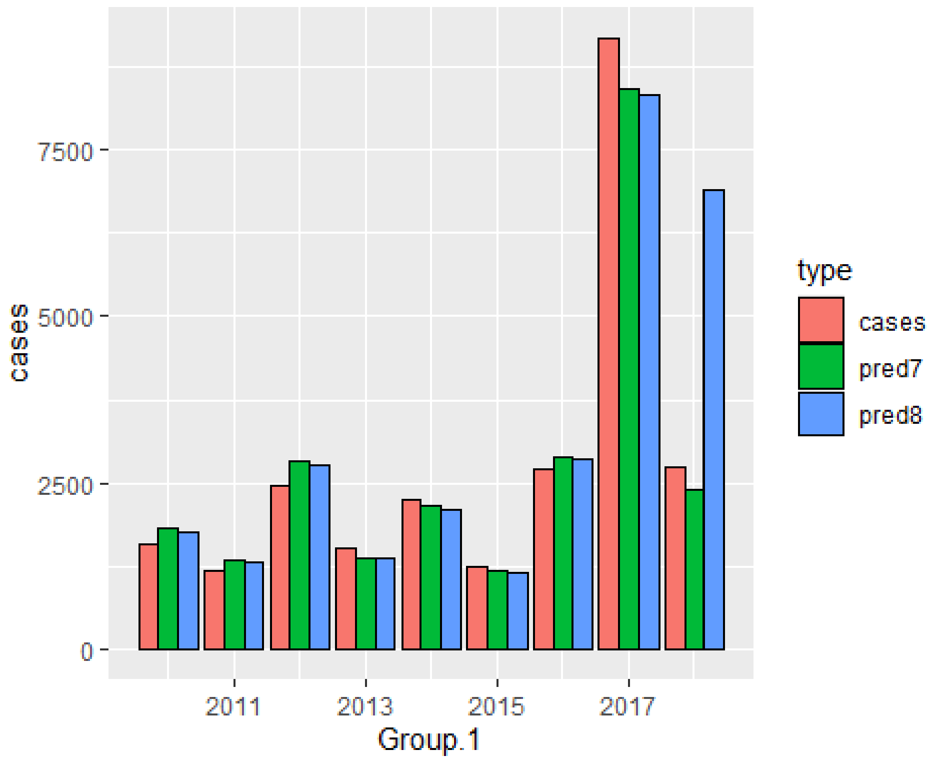
Part 6c: Exercises

1. Compare the DICs from Models 6 and 7.
2. (Optional) Please compare the fitted vs. observed values in Model 7 and produce a plot for month and a map to summarize the different random effects in the model.

Part 6d: Getting Predictions (Optional)

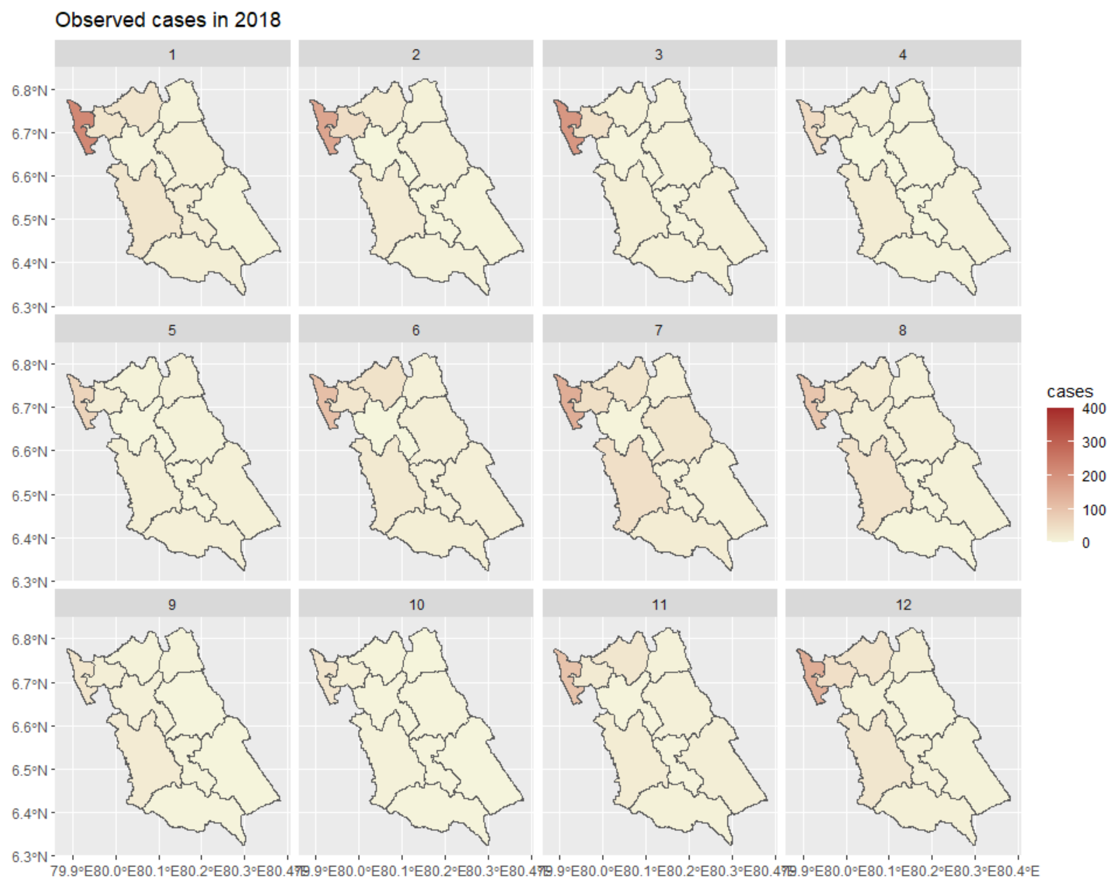
INLA can also compute predictions for observations without outcome data. To test this, let us create a new outcome variable in which we have no case data for 2018.

```
dat2$cases2 <- dat2$cases
dat2[dat2$Year == 2018, ]$cases2 <- NA

formula8 <- cases2 ~ 1 +
  f(season, model = "rw1", cyclic = TRUE) +
  f(Year, model = "rw1") +
  f(time, model = "iid") +
  f(ID, model = "bym", graph = map.x) +
  f(IDspacetime, model = "iid")

m.inla.8 <- inla(formula8,
                 offset = log(pop),
                 data = dat2,
                 family = "nbinomial",
                 control.predictor = list(compute = TRUE),
                 control.compute = list(dic = TRUE))
# summary(m.inla.8)

dat2$pred7 <- round(m.inla.7$summary.fitted.values$`0.5quant`, 0)
dat2$pred8 <- m.inla.8$summary.fitted.values$`0.5quant`
dat2[dat2$Year != 2018,]$pred8 <- round(dat2[dat2$Year != 2018,]$pred8, 0)
dat2[dat2$Year == 2018,]$pred8 <- round(exp(dat2[dat2$Year == 2018,]$pred8),
0)

dat2_agg <- aggregate(dat2[, c("cases", "pred7", "pred8")], by =
list(dat2$Year), FUN = sum)
dat2_long <- gather(dat2_agg, type, cases, cases:pred8)

ggplot(dat2_long, aes(Group.1, cases)) +
  geom_col(aes(fill = type), position = "dodge", col = "black")
```

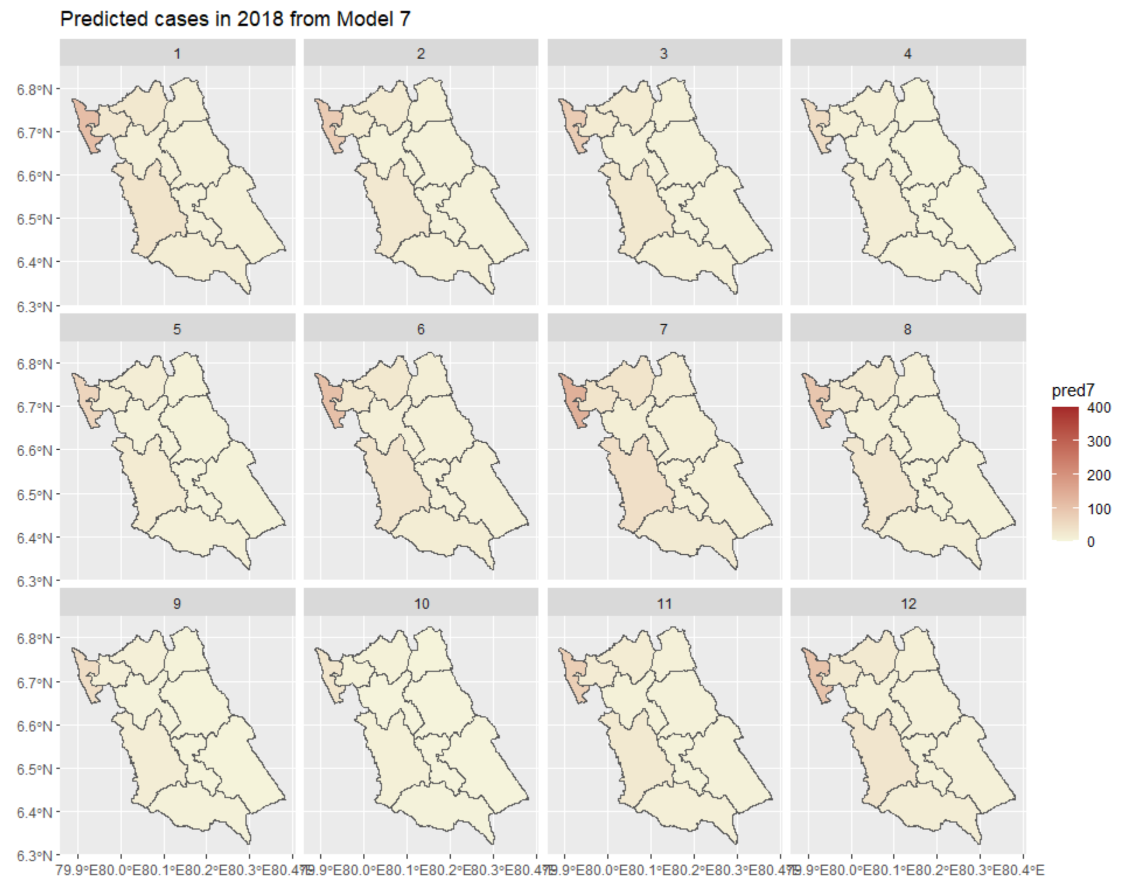


```
map3 <- merge(map, dat2[dat2$Year == 2018, c("regnames", "cases", "pred7",
"pred8", "season")], by.x = "reg_names", by.y = "regnames")
```

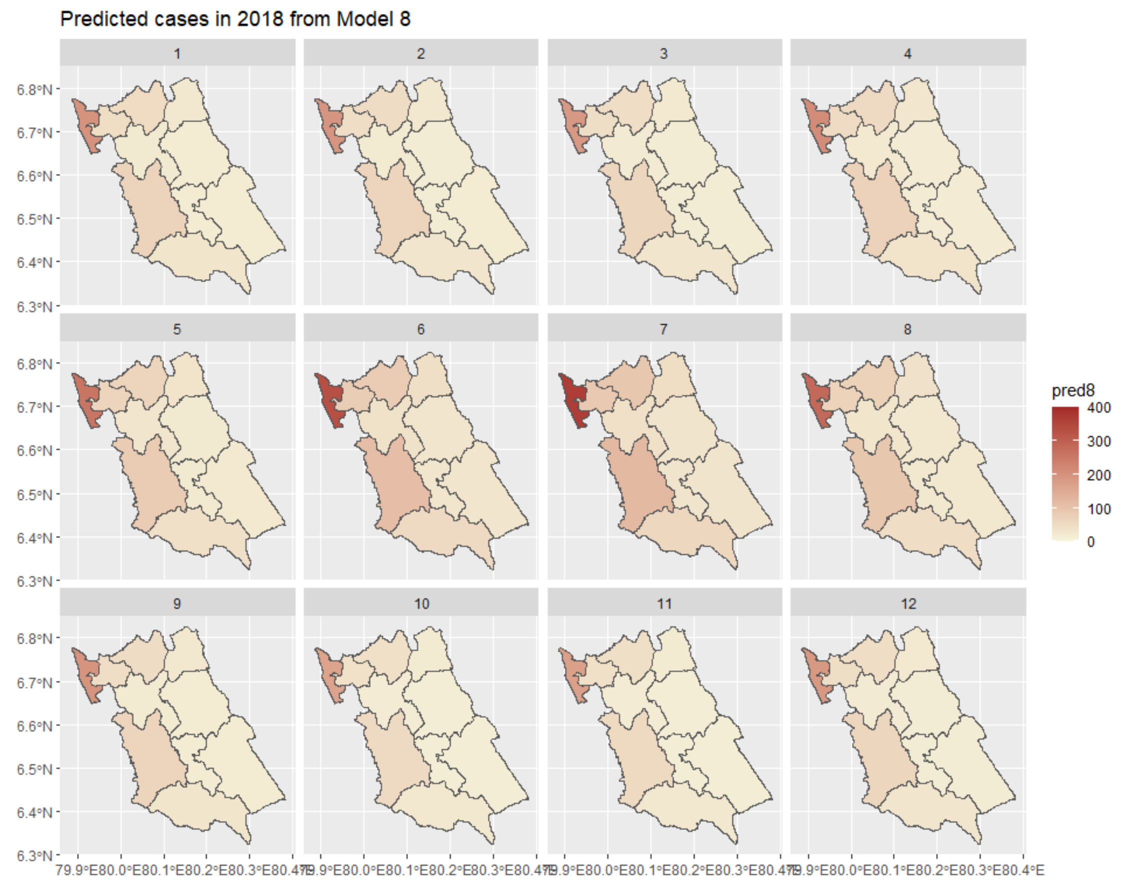
```r
ggplot(map3) +
  geom_sf(aes(fill = cases)) +
  facet_wrap(~season) +
  labs(title = "Observed cases in 2018") +
  scale_fill_gradient(low="beige",high="brown", limits = c(0, 400))
```



```r
ggplot(map3) +
  geom_sf(aes(fill = pred7)) +
  facet_wrap(~season) +
  labs(title = "Predicted cases in 2018 from Model 7") +
  scale_fill_gradient(low="beige",high="brown", limits = c(0, 400))
```



```r
ggplot(map3) +
  geom_sf(aes(fill = pred8)) +
  facet_wrap(~season) +
  labs(title = "Predicted cases in 2018 from Model 8") +
  scale_fill_gradient(low="beige",high="brown", limits = c(0, 400))
```



Part 7: DLNM for a Single Region

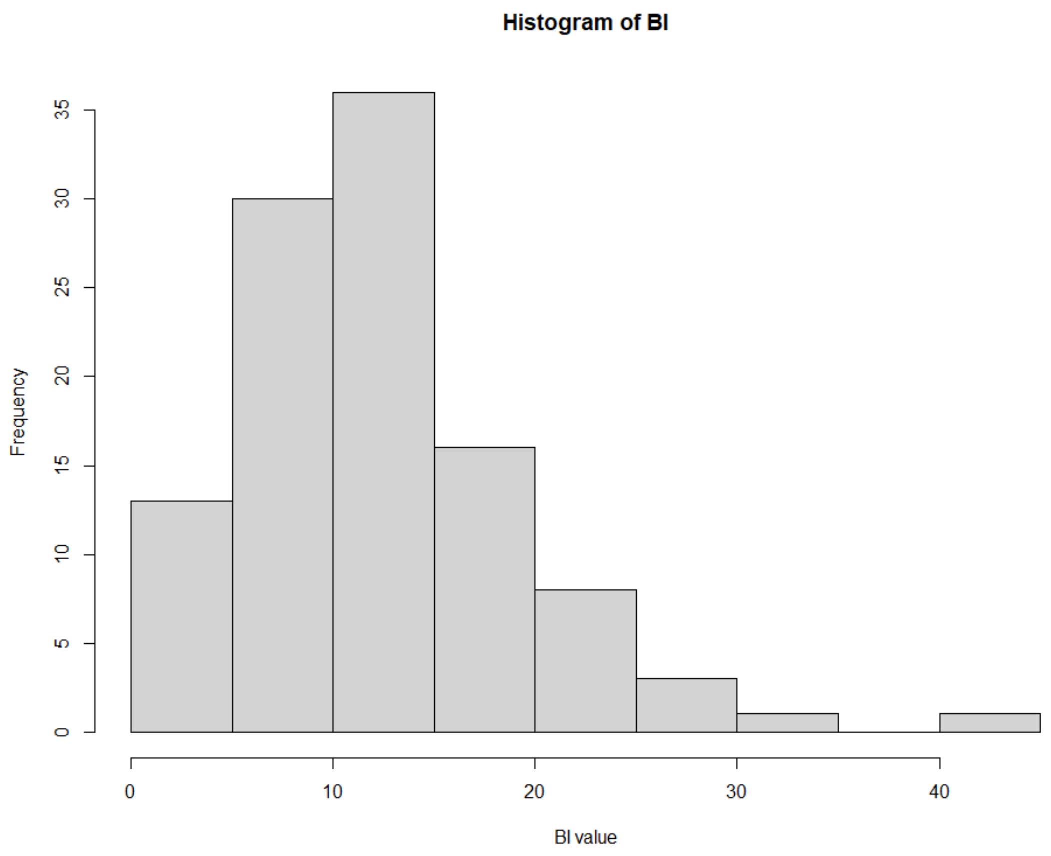
We will focus on a specific region, "Aga", to illustrate the modelling process.

```
reg1 <- subset(dat, regnames == "Aga")
summary(reg1)
```

```
##      series          regnames              pop             Year
##  Min.   :  1.00   Length:108         Min.   :39346   Min.   :2010
##  1st Qu.: 27.75   Class :character   1st Qu.:39831   1st Qu.:2012
##  Median : 54.50   Mode  :character   Median :40283   Median :2014
##  Mean   : 54.50                      Mean   :40135   Mean   :2014
##  3rd Qu.: 81.25                      3rd Qu.:40498   3rd Qu.:2016
##  Max.   :108.00                      Max.   :40607   Max.   :2018
##       time             season          cases              PI
##  Min.   :  1.00   Min.   : 1.00   Min.   : 0.000   Min.   : 0.900
##  1st Qu.: 27.75   1st Qu.: 3.75   1st Qu.: 3.000   1st Qu.: 7.225
##  Median : 54.50   Median : 6.50   Median : 6.000   Median :10.000
##  Mean   : 54.50   Mean   : 6.50   Mean   : 8.204   Mean   :10.532
##  3rd Qu.: 81.25   3rd Qu.: 9.25   3rd Qu.: 9.250   3rd Qu.:13.000
##  Max.   :108.00   Max.   :12.00   Max.   :90.000   Max.   :24.600
##        BI             pred1            pred2
##  Min.   : 0.90   Min.   : 5.000   Min.   : 4.000
##  1st Qu.: 8.15   1st Qu.: 6.750   1st Qu.: 6.000
##  Median :11.85   Median : 7.500   Median : 6.000
##  Mean   :12.72   Mean   : 9.019   Mean   : 7.852
##  3rd Qu.:15.22   3rd Qu.: 9.750   3rd Qu.: 8.750
##  Max.   :43.80   Max.   :20.000   Max.   :18.000
```

Before modelling, we examine the frequency distribution of the Breteau Index in the selected region.

```
par(mfrow = c(1, 1))
hist(reg1$BI, xlab = "BI value",
     main = "Histogram of BI")
```



Part 7a: Cross-Basis Function Definition

Define Lag Period and Spline Functions We define the cross-basis function for the Breteau Index using a natural cubic spline and specify the desired lag period. The cross-basis function allows us to examine the relationship between the Breteau Index and dengue cases over time.

lag: Defines the lag period: As an integer scalar, it specifies the maximum lag. As a vector of length 2, it defines the lag range (e.g., from minimum to maximum lag). The lag period should be defined based on evidence, considering the biologically plausible temporal dimension to avoid introducing noise into the model.

argvar: The predictor (exposure-response relationship). arglag: The lag (lag-response relationship).

fun: Common choices for fun are represented by ns and bs from package splines

It is important to select appropriate function and knot positioning to select the best fit model. It will be useful to explore the frequency distribution of the exposure variable(shown above) when defining the knot positioning.

```
lag <- c(0, 3)
boundpi <- range(reg1$PI) # set boundaries for the cross-basis

argvarpi <- list(fun = "ns", knots = c(15), df = 2, cen = 0)
arglagpi <- list(fun = "ns", knots = c(2), df = 2)

suppressWarnings({
  cbpi <- crossbasis(reg1$BI, lag = lag, argvar = argvarpi, arglag =
arglagpi)
})

summary.crossbasis(cbpi)

## CROSSBASIS FUNCTIONS
## observations: 108
## range: 0.9 to 43.8
## lag period: 0 3
## total df:  6
##
## BASIS FOR VAR:
## fun: ns
## knots: 15
## intercept: FALSE
## Boundary.knots: 0.9 43.8
##
## BASIS FOR LAG:
## fun: ns
## knots: 2
## intercept: TRUE
## Boundary.knots: 0 3
```

Part 7b: Model Fitting

We fit a quasi-Poisson model to account for overdispersion in the data, which is common in count data such as dengue cases.

Here we define natural spline function to the indicator variable 'time' to model the seasonality and long term trends. You may adjust the spline function and degree of freedom depending on the nature of the data presented. We have used 2 df per year (generally can go up to 6 df per year) depending on the structure of the time varying confounders and seasonal patterns.

Offset (offset = log(pop)): The offset term adjusts the model for differences in population size across observations. By taking the logarithm of the population (log(pop)), the model ensures that the outcome (cases) is adjusted per unit population, making the estimates comparable across regions or time periods with different population sizes.

```
model <- glm(cases ~ cbpi + ns(time, df = 10 * 2), family = quasipoisson(),
offset = log(pop), reg1)
# summary(model)
```

Part 7c: Model Evaluation

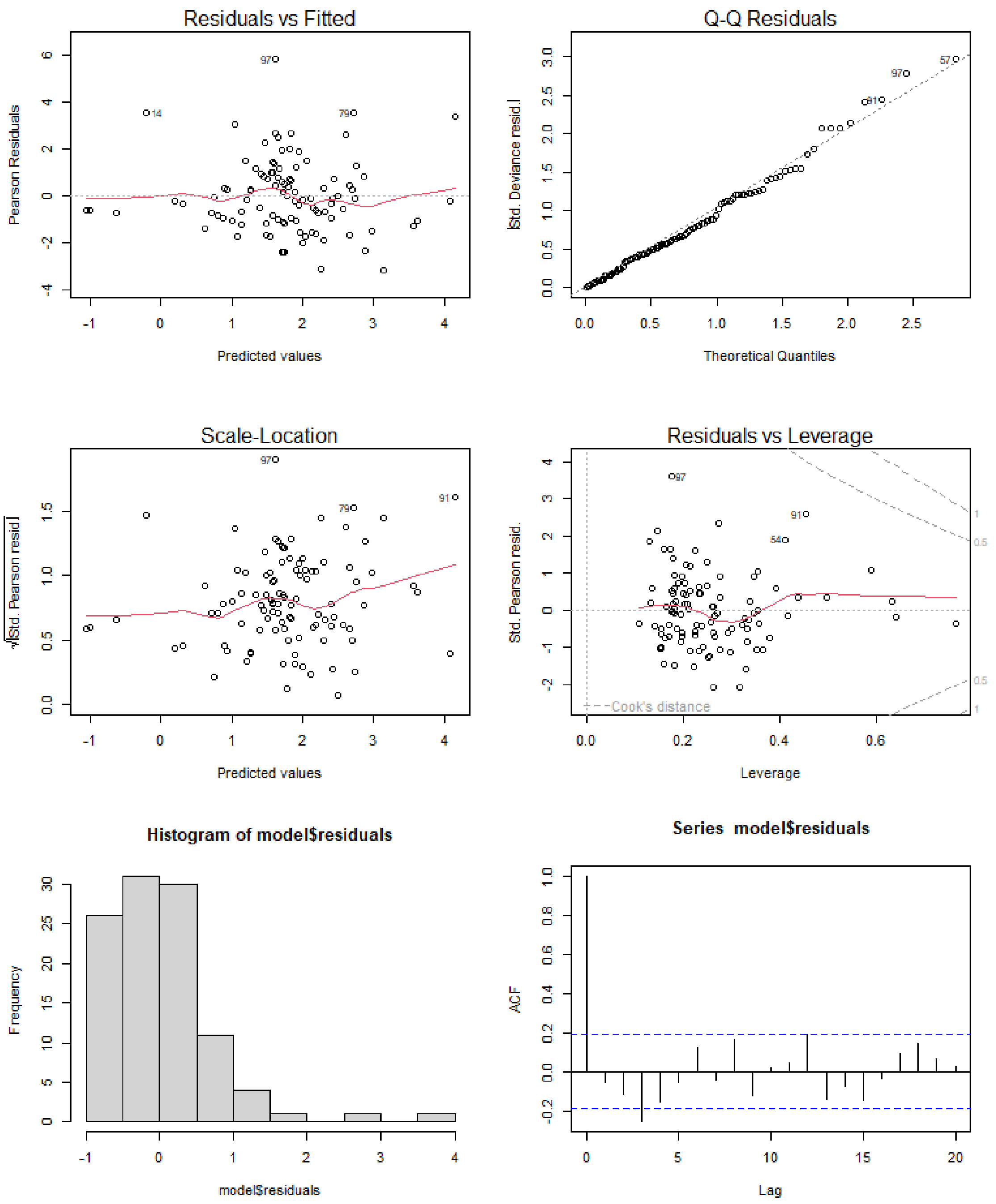
Goodness of Fit and Residual Analysis We assess the goodness of fit of the model by observing aic and plotting residuals and examining their distribution.

Function to compute Q-AIC in Quasi-poisson models

```
fqaic <- function(model) {
  loglik <- sum(dpois(model$y,model$fitted.values,log=TRUE))
  phi <- summary(model)$dispersion
  qaic <- -2*loglik + 2*summary(model)$df[3]*phi
  return(qaic)
}

fqaic(model)

## [1] 779.1978
```

Create diagnostic plots

```
par(mfrow = c(3, 2))
plot(model)
hist(model$residuals)
acf(model$residuals)
```



Part 8: Evaluating the Lag Space

Part 8a: Prediction Using crosspred

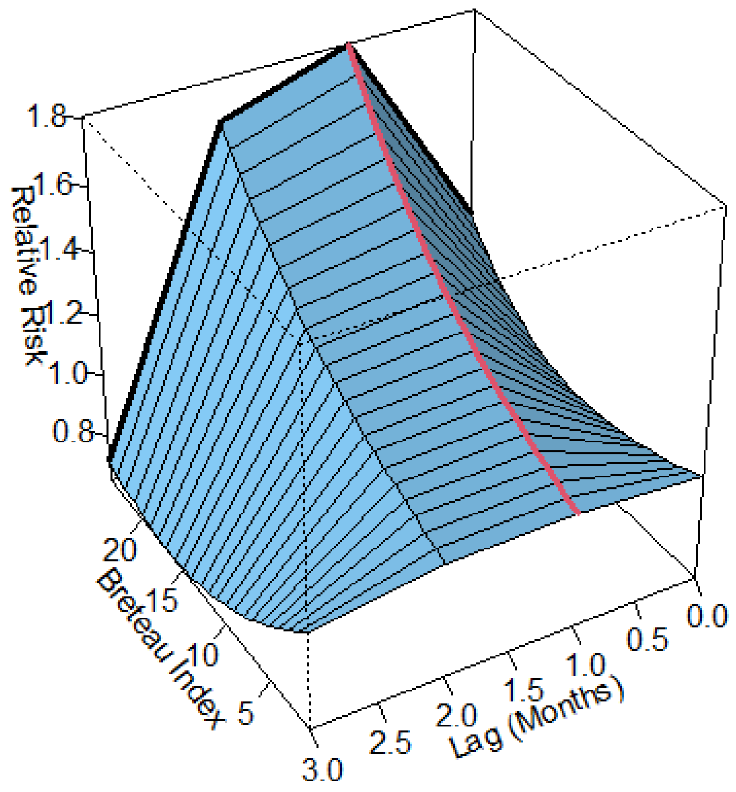
We use the `crosspred` function to predict the relative risk of dengue cases across the space of the exposure (BI) and the and lag.

'cen' specifies the centering value, then used as a reference for relative risk estimation. The centering value is set by default to the mid-range. Here we used 0 as the centering value

```
cp <- crosspred(cbpi, model, from = boundpi[1], to = boundpi[2], by = 1, cen
= 0)
```

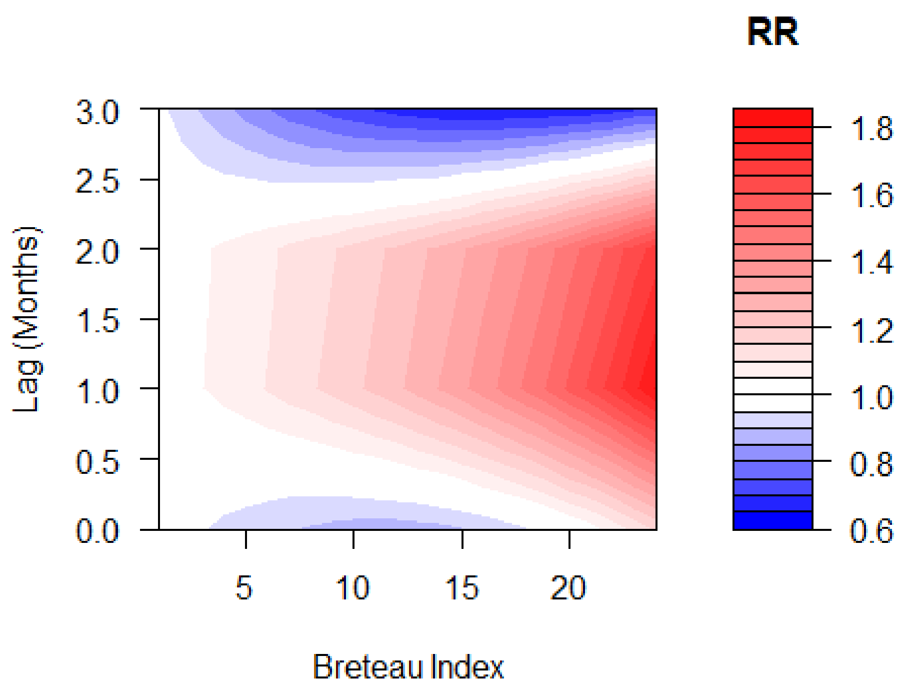
plot the exposure response space

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

d3 <- plot(cp, xlab = "Breteau Index", zlab = "Relative Risk", ylab = "Lag
(Months)",
           phi = 35, theta = 240, ltheta = 170, shade = 0.2)
mtext(text = "", cex = 1.5)

lines(trans3d(x = 24, y = 0:3, z = cp$matRRfit[as.character(24),],
              pmat = d3), lwd = 3, lty = 1, col = 1)
lines(trans3d(x = cp$predvar, y = 1, z = cp$matRRfit[ ,"lag1"],
              pmat = d3), lwd = 3, col = 2)
```



Contour plot for the exposure response space

```
plot(cp,"contour", key.title = title("RR"),
     plot.title = title("", xlab = expression("Breteau Index"), ylab = "Lag
(Months)"))
```



Part 8b: Unidimensional Summaries

We obtain lag-specific and predictor-specific summaries using the crossreduce function.

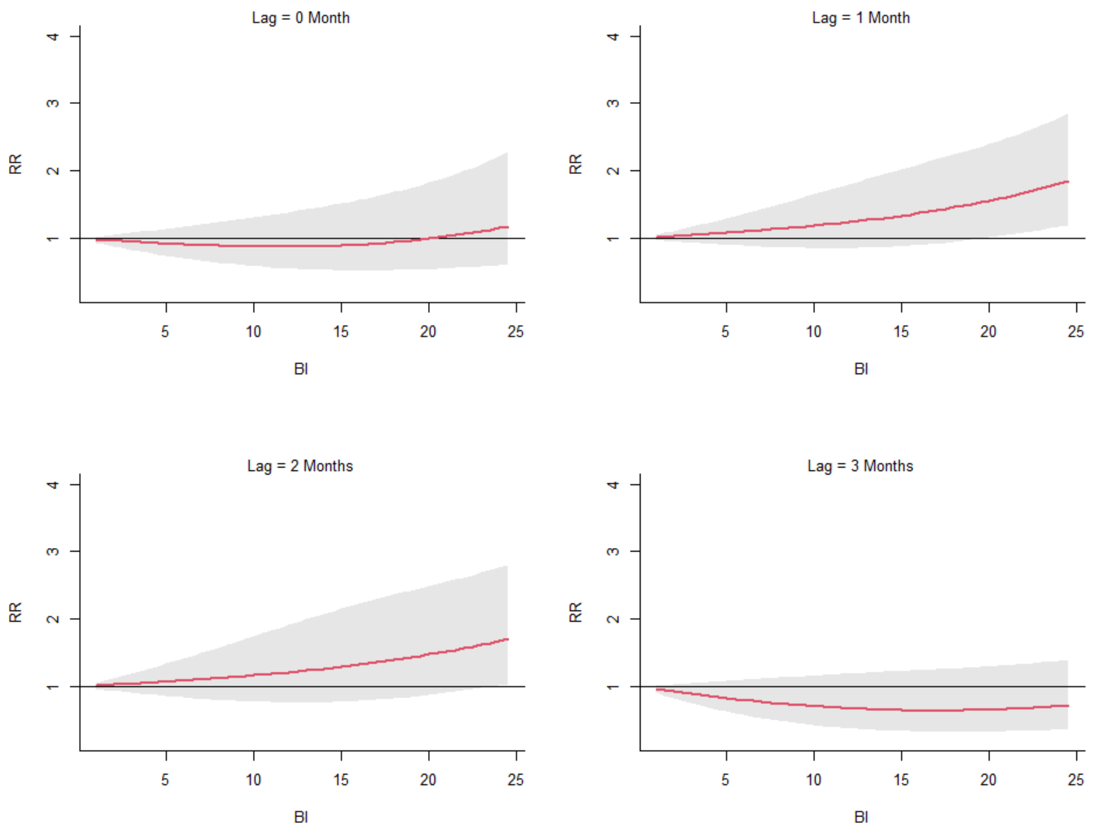
Lag specific summaries

```
crlag0 <- crossreduce(cbpi, model, type = "lag", value = 0, from =
boundpi[1],
                     to = boundpi[2], bylag = 0.2, cen = 0)
crlag1 <- crossreduce(cbpi, model, type = "lag", value = 1, from =
boundpi[1],
                      to = boundpi[2], bylag = 0.2, cen = 0)
crlag2 <- crossreduce(cbpi, model, type = "lag", value = 2, from =
boundpi[1],
                      to = boundpi[2], bylag = 0.2, cen = 0)
crlag3 <- crossreduce(cbpi, model, type = "lag", value = 3, from =
boundpi[1],
                      to = boundpi[2], bylag = 0.2, cen = 0)
```

plot the lag specific associations

```
par(mfrow = c(2, 2))
plot(crlag0, xlab = "BI", ylab = "RR", col = 2, ylim = c(.2, 4), lwd = 2)
mtext(text = "Lag = 0 Month", cex = 0.8)
```

```r
plot(crlag1, xlab = "BI", ylab = "RR", col = 2, ylim = c(.2, 4), lwd = 2)
mtext(text = "Lag = 1 Month", cex = 0.8)

plot(crlag2, xlab = "BI", ylab = "RR", col = 2, ylim = c(.2, 4), lwd = 2)
mtext(text = "Lag = 2 Months", cex = 0.8)

plot(crlag3, xlab = "BI", ylab = "RR", col = 2, ylim = c(.2, 4), lwd = 2)
mtext(text = "Lag = 3 Months", cex = 0.8)
```

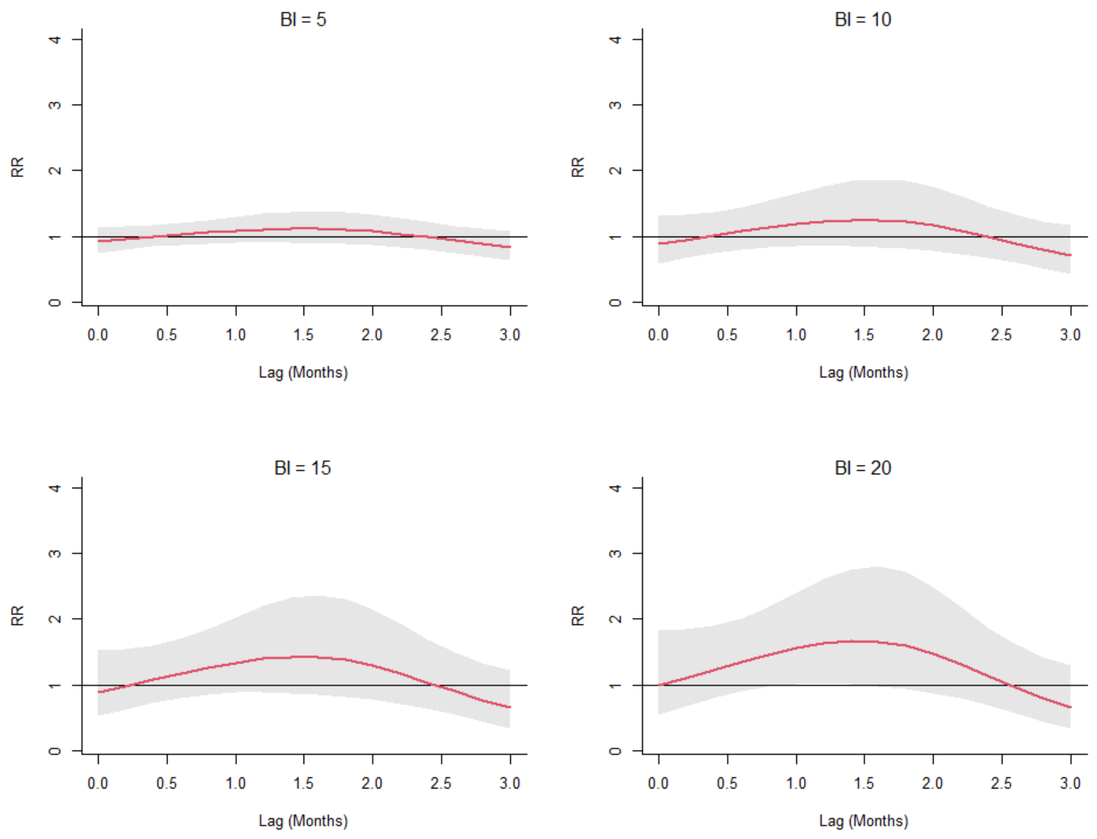


Predictor specific summaries

```r
crvar_5 <- crossreduce(cbpi, model, type = "var", value = 5, from =
boundpi[1],
                       to = boundpi[2], bylag = 0.2, cen = 0)
crvar_10 <- crossreduce(cbpi, model, type = "var", value = 10, from =
boundpi[1],
                        to = boundpi[2], bylag = 0.2, cen=0)
crvar_15 <- crossreduce(cbpi, model, type = "var", value = 15, from =
boundpi[1],
                        to = boundpi[2], bylag = 0.2, cen = 0)
crvar_20 <- crossreduce(cbpi, model, type = "var", value = 20, from =
boundpi[1],
                        to = boundpi[2], bylag = 0.2, cen = 0)
```

Plot the predictor specific summaries

```
par(mfrow = c(2, 2))

plot(crvar_5, xlab = "Lag (Months)", ylab = "RR", col = 2, ylim = c(0.1, 4),
lwd = 2)
mtext(text = paste("BI = 5", sep = ""), cex = 1)

plot(crvar_10, xlab = "Lag (Months)", ylab = "RR", col = 2, ylim = c(0.1, 4),
lwd = 2)
mtext(text = paste("BI = 10", sep = ""), cex = 1)

plot(crvar_15, xlab = "Lag (Months)", ylab = "RR", col = 2, ylim = c(0.1, 4),
lwd = 2)
mtext(text = paste("BI = 15", sep = ""), cex = 1)

plot(crvar_20, xlab = "Lag (Months)", ylab = "RR", col = 2, ylim = c(0.1, 4),
lwd = 2)
mtext(text = paste("BI = 20", sep = ""), cex = 1)
```



Part 9: Second stage hierarchical meta-analysis (Optional)

Part 9a: Individual Models for Each Region

The code conducts a two-stage analysis to assess the association between the BI and dengue cases across different regions. The analysis is structured into two main stages: the first stage involves fitting models for each region, and the second stage involves meta-analyzing the results from all regions.

Define a vector for dub-district health units

```
regions <- as.character(unique(dat$regnames))
```

CREATE A LIST WITH THE REGIONAL SERIES

```
data <- lapply(regions, function(x) dat[dat$regnames == x,])
names(data) <- regions
m <- length(regions)
```

define PI ranges

```
rangespi <- t(sapply(data, function(x)range(x$PI, na.rm = TRUE)))
```

Model Specification:

The predictor space uses a quadratic spline (`ns()`) with specified knots and degrees of freedom (`df`) to model the effect of BI. The lag space uses a natural cubic spline with specified knots and degrees of freedom to model lagged effects of BI on dengue cases. We used a similar approach to the single region model.

Define parameters for cross-basis matrix

```
lag <- c(0, 3)

boundpi <- colMeans(rangespi)
#varknots <- boundr[1] + diff(boundr)/4*(1:2)
argvarpi <- list(type = "ns", knots = c(15), df = 1, cen = 0)
#arglagpi <- list(type = "ns", df = 3)
arglagpi <- list(type = "ns", knot = c(2), df = 2)
```

Built objects where results will be stored

Overall cumulative summaries

```
coefall <- matrix(NA, length(data), 2, dimnames = list(regions, paste("b",
seq(2), sep = "")))
```

Predictor-specific summaries

```
coef0 <- matrix(NA, length(data), 2, dimnames = list(regions, paste("b",
seq(2), sep = "")))
coef1 <- coef2 <- coef3 <- coef0
```

(Co)variance matrices

```
vcovall <- vector("list", length(data))
names(vcovall) <- regions
vcov0 <- vcov1 <- vcov2 <- vcov3 <- vcovall
```

Q-AIC

```r
qaic <- 0

####################################################################
# Loop over each sub-region health units

# LOOP FOR CITIES
# WARNING FOR PREDICTION BEYOND BOUNDARIES SUPPRESSED
system.time({
  for(i in seq(data)) {

    # LOAD
    sub <- data[[i]]

    # DEFINE THE CROSS-BASES
    suppressWarnings({
      cbpi <- crossbasis(sub$BI,lag = lag, argvar = argvarpi, arglag =
arglagpi)
    })

    # RUN THE FIRST-STAGE MODELS

    mfirst <- glm(cases ~ cbpi+ns(time, df = 10*2), family = quasipoisson(),
offset = log(pop), sub)

    ####################################################################
    # REDUCTION TO SUMMARY ASSOCIATIONS

    # TO OVERALL CUMULATIVE SUMMARY
    suppressWarnings({
      crall <- crossreduce(cbpi, mfirst, from = boundpi[1], to = boundpi[2],
by = 2, cen = 0)

    })
    # TO PREDICTOR-SPECIFIC SUMMARY
    suppressWarnings({
      crlag0 <- crossreduce(cbpi, mfirst, type = "lag", value = 0, cen = 0)
      crlag1 <- crossreduce(cbpi, mfirst, type = "lag", value = 1, cen = 0)
      crlag2 <- crossreduce(cbpi, mfirst, type = "lag", value = 2, cen = 0)
      crlag3 <- crossreduce(cbpi, mfirst, type = "lag", value = 3, cen = 0)

    })

    ####################################################################
    # STORE THE RESULTS

    # OVERALL CUMULATIVE SUMMARY FOR THE MAIN MODEL
    coefall[i,] <- coef(crall)
    vcovall[[i]] <- vcov(crall)
    # PREDICTOR-SPECIFIC SUMMARY FOR MAIN MODEL
    coef0[i,] <- coef(crlag0)
    coef1[i,] <- coef(crlag1)
    coef2[i,] <- coef(crlag2)
    coef3[i,] <- coef(crlag3)

    vcov0[[i]] <- vcov(crlag0)
    vcov1[[i]] <- vcov(crlag1)
    vcov2[[i]] <- vcov(crlag2)
    vcov3[[i]] <- vcov(crlag3)


    # Q-AIC
    qaic[i] <- fqaic(mfirst)


  }
})
```

```
##    user  system elapsed 
##    0.12    0.00    0.12
```

```
####################################################################
# GRAND Q-AIC
sum(qaic)
```

```
## [1] 10929.18
```

Part 9b: Multivariate Meta-Analysis

Select the estimation method method
Estimation method: "fixed" for fixed-effects models, "ml" or "reml" for random-effects models fitted through (restricted) maximum likelihood, "mm" for random-effects models fitted through method of moments, and "vc" for random-effects models fitted through variance components.

Here we use ml to perform the random effect meta-analysis

```
method <- "ml"
```

Model specification for the overall cumulative association (the cumulative lag effect)

```
mvall <- mvmeta(coefall ~ 1, vcovall, method = method)
# summary(mvall)
```

model specifications for the predictor specific summaries

```
mv0 <- mvmeta(coef0 ~ 1, vcov0, method = method)
mv1 <- mvmeta(coef1 ~ 1, vcov1, method = method)
mv2 <- mvmeta(coef2 ~ 1, vcov2, method = method)
mv3 <- mvmeta(coef3 ~ 1, vcov3, method = method)

# # examine the model summaries
# summary(mv0)
# summary(mv1)
# summary(mv2)
# summary(mv3)
```

Generate a table for model summaries

```
ftab1 <- function(m, mref = NULL) {
  # HETEROGENEITY AND IC STATS
  q <- qtest(m)
  het <- c(q$Q[1], q$df[1], q$pvalue[1], (q$Q[1]-q$df[1])/q$Q[1]*100, AIC(m),
BIC(m))
}

tab_mv <- matrix(NA, 5, 6)
colnames(tab_mv) <- c("Q", "df", "p", "I-square", "AIC", "BIC")
rownames(tab_mv) <- c("Overall", "Lag 0 Month", "Lag 1 Months",
                      "Lag 2 Month", "Lag 3 Months")

tab_mv[1, ] <- ftab1(mvall)
tab_mv[2, ] <- ftab1(mv0)
tab_mv[3, ] <- ftab1(mv1)
tab_mv[4, ] <- ftab1(mv2)
tab_mv[5, ] <- ftab1(mv3)

tab_mv

##                     Q df            p I-square      AIC      BIC
## Overall      39.58059 18 2.379663e-03 54.52316 91.46158 96.44025
## Lag 0 Month  32.48422 18 1.925456e-02 44.58848 44.97992 49.95858
## Lag 1 Months 51.44989 18 4.543131e-05 65.01450 41.34152 46.32018
## Lag 2 Month  49.27284 18 9.715447e-05 63.46872 47.40175 52.38041
## Lag 3 Months 49.85551 18 7.937440e-05 63.89567 50.05764 55.03630
```

Create bases for prediction

Bases of temperature and lag used to predict, equal to that used for estimation The code below is creating bases (reference points) for the prediction of Relative Risks (RR) across different values of the Breteau Index (BI) and time lags. These bases are used to exposure lag response associations.

```
#defines a sequence of values from the lower to the upper bound of the
Breteau Index range
xvar <- seq(boundpi[1], boundpi[2], by = 1)

#generates the basis matrix for the Breteau Index (xvar) using the attributes
(argvar) from the previously #created cross-basis object (cbpi). The onebasis
function
bvar <- do.call("onebasis", c(list(x = xvar), attr(cbpi, "argvar")))

#generates the basis matrix for the lag periods (xlag) using the attributes
(arglag) from the cross-basis #object (cbpi)
```

```r
xlag <- 0:3
blag <- do.call("onebasis", c(list(x = xlag), attr(cbpi, "arglag")))
```

Extract region-specific first-stage summaries

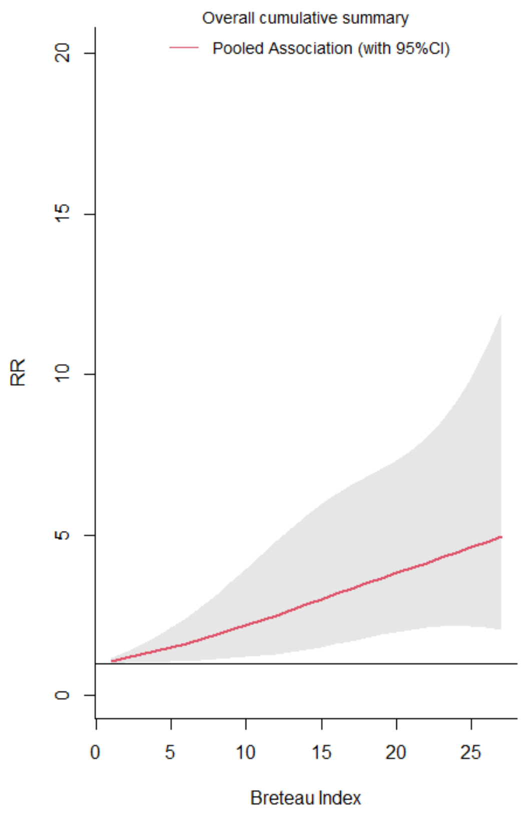
```r
regall <- apply(coefall, 1, function(x) exp(bvar %*% x))
```

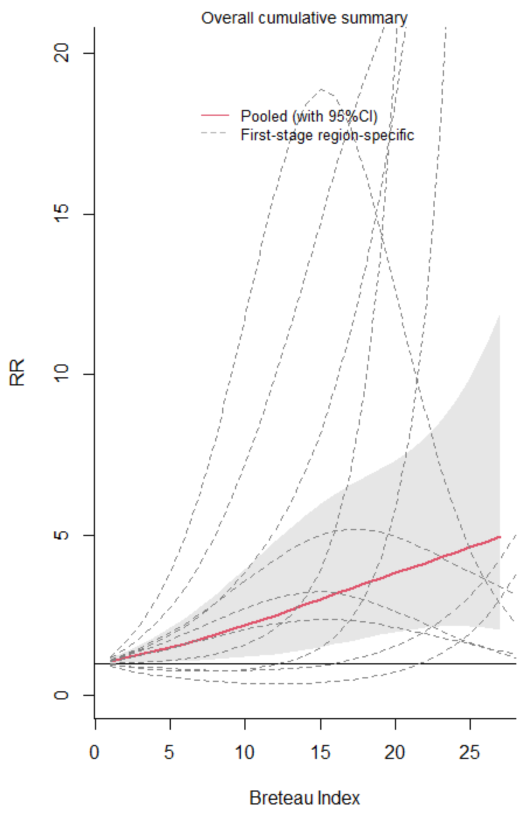
Overall cumulative summary association for main model

```r
cpall <- crosspred(bvar, coef = coef(mvall), vcov = vcov(mvall),
                   model.link = "log", by = 1, from = boundpi[1], to =
boundpi[2])
```

plot overall cumulative summary and region specific estimates

```r
par(mfrow = c(1,2))

plot(cpall, type = "n", ylab = "RR", ylim = c(0.1, 20), xlab = "Breteau
Index")
lines(cpall, col = 2, lwd = 2)
legend ("top", c("Pooled Association (with 95%CI)"),
        lty = c(1, 2), lwd = 1.5, col = c(2, grey(0.7)), bty = "n", inset =
0, cex = 0.85)
mtext(text = paste("Overall cumulative summary", sep = ""), cex = 0.9)

# plot the division specific associations

plot(cpall, type = "n", ylab = "RR", ylim = c(0.1, 20), xlab = "Breteau
Index")
lines(cpall, col = 2, lwd = 2)
matplot(regall, type = "l", col = grey(0.5), lty = 2, add = TRUE)
legend ("top", c("Pooled (with 95%CI)", "First-stage region-specific"),
        lty = c(1,2), lwd = 1.5, col = c(2, grey(0.7)), bty = "n", inset =
0.1, cex = 0.8)
mtext(text = paste("Overall cumulative summary", sep = ""), cex = 0.9)
```





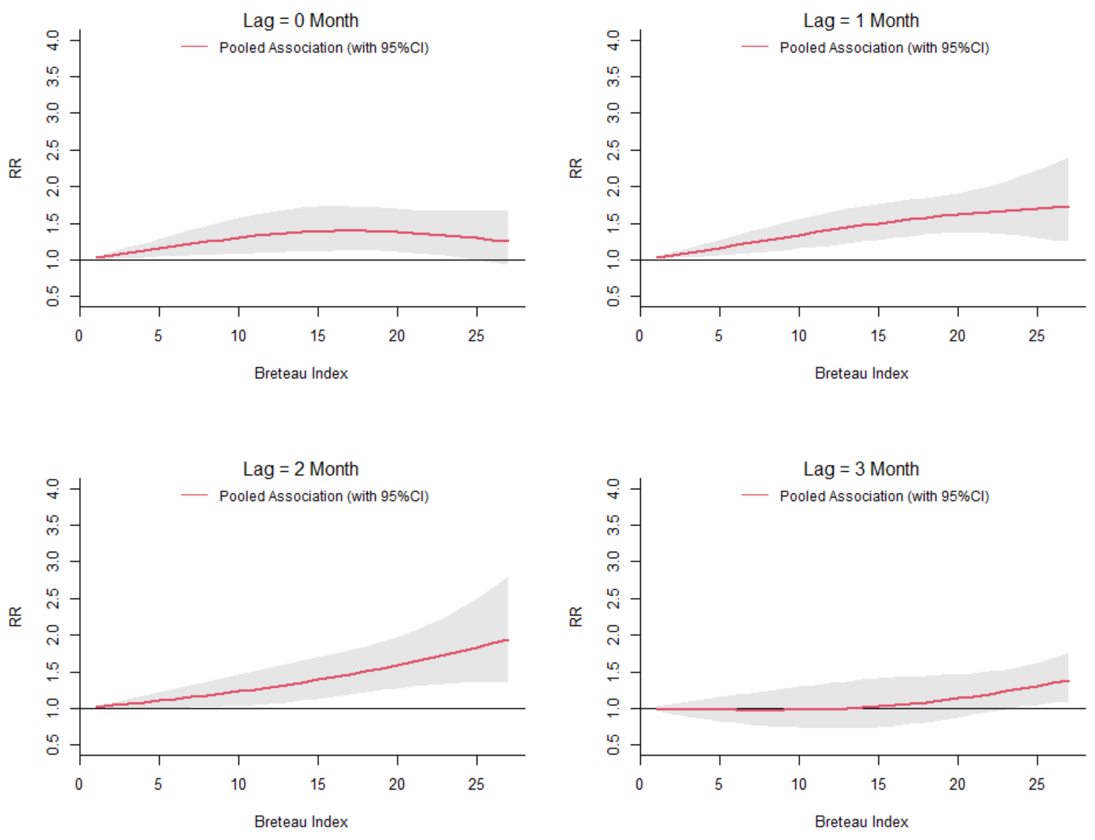
Define lag-specific summaries

```
cp0 <- crosspred(bvar, coef = coef(mv0), vcov = vcov(mv0),
                 model.link = "log", by = 1, from = boundpi[1], to =
boundpi[2])
cp1 <- crosspred(bvar, coef = coef(mv1), vcov = vcov(mv1),
                 model.link = "log", by = 1, from = boundpi[1], to =
boundpi[2])
cp2 <- crosspred(bvar, coef = coef(mv2), vcov = vcov(mv2),
                 model.link = "log", by = 1, from = boundpi[1], to =
boundpi[2])
cp3 <- crosspred(bvar, coef = coef(mv3), vcov = vcov(mv3),
                 model.link = "log", by = 1, from = boundpi[1], to =
boundpi[2])
```

Plot predictor specific summaries

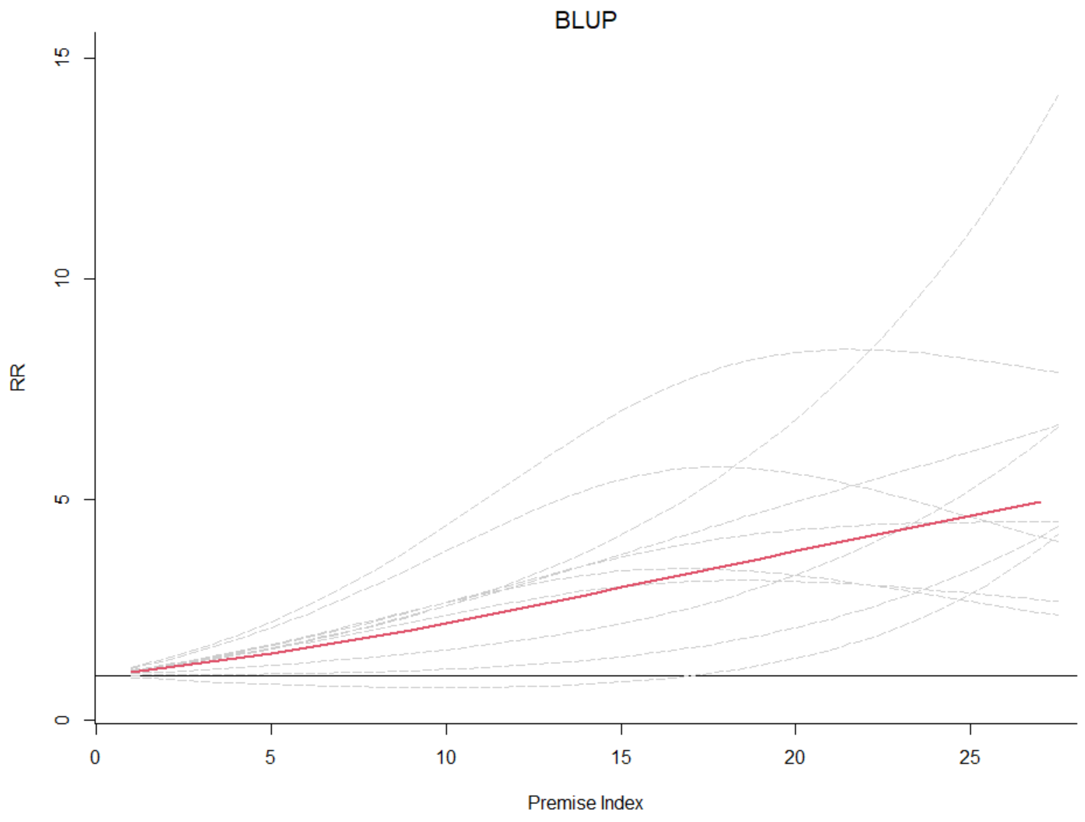
```
par(mfrow = c(2, 2))
mar = c(0, 0, 0, 0)

plot(cp0, type = "n", ylab = "RR", ylim = c(0.5, 4), xlab = "Breteau Index")
lines(cp0, col = 2, lwd = 2)
legend("top", c("Pooled Association (with 95%CI)"),
       lty = c(1, 2), lwd = 1.5, col = c(2, grey(0.7)), bty = "n", inset = 0,
cex = 0.9)
```

```
mtext(text = paste("Lag = ", 0, " Month", sep = ""), cex = 1)


plot(cp1, type = "n", ylab = "RR", ylim = c(0.5, 4), xlab = "Breteau Index")
lines(cp1, col = 2, lwd = 2)
legend("top", c("Pooled Association (with 95%CI)"),
       lty = c(1, 2), lwd = 1.5, col = c(2, grey(0.7)), bty = "n", inset = 0,
cex = 0.9)
mtext(text = paste("Lag = ", 1, " Month", sep = ""), cex = 1)

plot(cp2, type = "n", ylab = "RR", ylim = c(0.5, 4), xlab = "Breteau Index")
lines(cp2, col = 2, lwd = 2)
legend("top", c("Pooled Association (with 95%CI)"),
       lty = c(1, 2), lwd = 1.5, col = c(2, grey(0.7)), bty = "n", inset = 0,
cex = 0.9)
mtext(text = paste("Lag = ", 2, " Month", sep = ""), cex = 1)

plot(cp3, type = "n", ylab = "RR", ylim = c(0.5, 4), xlab = "Breteau Index")
lines(cp3, col = 2, lwd = 2)
legend("top", c("Pooled Association (with 95%CI)"),
       lty = c(1, 2), lwd = 1.5, col = c(2, grey(0.7)), bty = "n", inset = 0,
cex = 0.9)
mtext(text = paste("Lag = ", 3, " Month", sep = ""), cex = 1)
```



Part 9c: BLUP Estimation

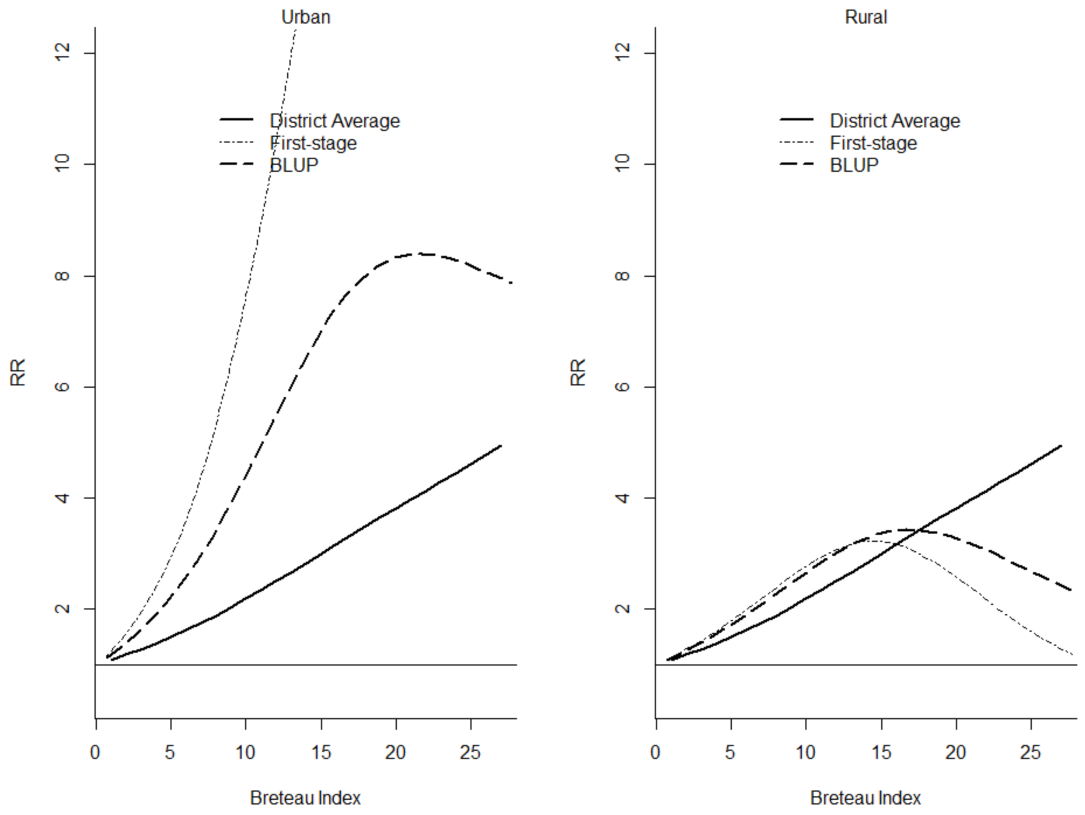
```
blup <- blup(mvall, vcov = TRUE)
plot(cpall, type = "n", ci = "n", ylab = "RR", ylim = c(0.5, 15), xlab =
"Premise Index")
for(i in seq(m)) {
  # WARNING FOR PREDICTION BEYOND BOUNDARIES SUPPRESSED
  suppressWarnings(
    lines(crosspred(bvar, coef = blup[[i]]$blup, vcov = blup[[i]]$vcov,
                    model.link = "log", from = boundpi[1], to = boundpi[2]),
col = grey(0.8), lty = 5)
  )
}
lines(cpall, "overall", col = 2, lwd = 2)
mtext("BLUP", cex = 1.3)
```



Part 9d: Regional summaries with BLUP

```
par(mfrow=c(1,2))
mar=c(0,0,0,0)

plot(cpall,type="n",ci="n",ylab="RR",ylim=c(.5,12),xlab="Breteau Index")
lines(cpall,col=1,lwd=2)
lines(xvar,exp(bvar%*%coefall[4,]),lty=4)
lines(xvar,exp(bvar%*%blup[[4]]$blup),lty=5,lwd=2)
mtext("Urban",cex=1)
```

```
legend("top",c("District Average","First-stage","BLUP"),lty=c(1,4,5),
       lwd=c(2,1,2),cex=1,inset=0.1,bty="n")


plot(cpall,type="n",ci="n",ylab="RR",ylim=c(.5,12),xlab="Breteau Index")
lines(cpall,col=1,lwd=2)
lines(xvar,exp(bvar%*%coefall[9,]),lty=4)
lines(xvar,exp(bvar%*%blup[[9]]$blup),lty=5,lwd=2)
mtext("Rural",cex=1)
legend("top",c("District Average","First-stage","BLUP"),lty=c(1,4,5),
       lwd=c(2,1,2),cex=1,inset=0.1,bty="n")
```



Resources

- https://www.r-inla.org/home
- https://groups.google.com/g/r-inla-discussion-group
- Books on INLA
  - Bayesian inference with INLA: https://becarioprecario.bitbucket.io/inla-gitbook/index.html
  - Bayesian Regression Modeling with INLA: https://julianfaraway.github.io/brinlabook/index.html
  - Spatial and Spatio-temporal Bayesian Models with R-INLA: https://sites.google.com/a/r-inla.org/stbook/ (not free online)

  - Advanced Spatial Modeling with Stochastic Partial Differential Equations Using R and INLA: https://becarioprecario.bitbucket.io/spde-gitbook/
  - Geospatial Health Data: Modelling and Visualization with R-INLA and Shiny: https://www.paulamoraga.com/book-geospatial/index.html
  - Dynamic Time Series Models using R-INLA: An Applied Perspective: https://ramanbala.github.io/dynamic-time-series-models-R-INLA/
- Papers on INLA: https://www.r-inla.org/learnmore/papers
- Gasparrini A, Armstrong B, Kenward MG. Distributed lag non-linear models. Stat Med. 2010;29: 2224–2234. doi:10.1002/sim.3940
- Gasparrini A, Armstrong B. Reducing and meta-analysing estimates from distributed lag non-linear models. BMC Med Res Methodol. 2013;13: 1–10. doi:10.1186/1471-2288-13-1
- Gasparrini A, Armstrong B, Kenward MG. Multivariate meta-analysis for non-linear and other multi-parameter associations. Stat Med. 2012;31: 3821–3839. doi:10.1002/sim.5471
- Liyanage P, Tozan Y, Overgaard HJ, Aravinda Tissera H, Rocklöv J. Effect of El Niño–Southern Oscillation and local weather on Aedes vector activity from 2010 to 2018 in Kalutara district, Sri Lanka: a two-stage hierarchical analysis. Lancet Planet Health. 2022;6: e577–e585. doi:10.1016/S2542-5196(22)00143-7