Session 2 Practical: Spatial modelling in R

Duncan Lee

---

1. Introduction

This practical session will show you how to undertake spatial modelling in R.

Aims of the session

By the end of this session you should be able to:

- Read in and format spatial data ready for analysis.
- Fit spatial correlation models using the `CARBayes` software package.
- Check the adequacy of the fitted model.
- Summarise the results from a fitted spatial model.

Motivating example

We consider part of the data analysed by Kavanagh, Lee, and Pryce (2016), which focused on the decentralisation of poverty in the four largest Scottish Cities, namely: Aberdeen, Dundee, Edinburgh, and Glasgow. Data were collected for the 2011 census, and are available at the datazone (DZ) resolution. Datazones are the smallest spatial units at which data about populations are released in Scotland, and between 500 and 1,000 people live in each datazone. These datazones lie within large administrative units called Health Boards, of which there are 14 in Scotland. The data are stored in `Scotland data.csv`, and contain the following columns.

- `DZ` - A unique code identifying each datazone.
- `HB` - The health board that the datazone lies within.
- `Easting` - The east/west coordinate (in metres) of the central point of the datazone.
- `Northing` - The north/south coordinate (in metres) of the central point of the datazone.
- `urban` - A six-level categorical variable denoting how urban or rural each datazone is. Here a 1 denotes urban while a 6 denotes rural, see http://www.gov.scot/Topics/Statistics/About/Methodology/UrbanRuralClassification.
- `JSA2011` - The number of people of working age claiming Job Seekers Allowance (JSA) in each datazone in 2011. JSA is a benefit paid to people with no job, hence is a measure of poverty.

- `workpop2011` - The total number of people of working age in each datazone in 2011.

The aim here is to model these data and compute an estimate and a 95% uncertainty interval for the dissimilarty index, a measure of the level of segregation between rich and poor populations.

2. Reading in and formatting spatial data

This section is a reminder from earlier about how to read in and format spatial data in `R`. The first task is to read in the following two data elements.

- The data set on poverty in Scotland stored in `Scotland data.csv`.
- The shapefiles for datazones in Scotland, which are contained in `datazone.shp` and `datazone.dbf`.

However, before reading in these data put all these files in the same directory as the `R` code file you are doing the analysis in, and then set the working directory to the current directory. This latter step is done via the `Rstudio Session` menu, and then selecting `Set Working Directory` and then `To Source File Location`. Once this is done you can read in all three data sets using the following code:

```
## Read in the data and fix the rownames to the datazone codes
dat <- read.csv(file="Scotland data.csv")
rownames(dat) <- dat$DZ
dat$DZ <- NULL


## Read in the shapefile
library(shapefiles)
```

```
## Loading required package: foreign
```

```
##
## Attaching package: 'shapefiles'
```

```
## The following objects are masked from 'package:foreign':
##
##     read.dbf, write.dbf
```

```
shp <- read.shp(shp.name = "datazone.shp")
dbf <- read.dbf(dbf.name = "datazone.dbf")
```

The data set can then be subsetted to Glasgow using the code:

```
gla <- dat[dat$HB=="Greater Glasgow & Clyde" & dat$urban==1, ]
```

Then the 3 data sets are combined into a `spatialPolygonsDataFrame` using the code:

```
library(CARBayes)
```

```
## Loading required package: MASS
```

```
## Loading required package: Rcpp
```

```
library(sp)
sp.gla <- combine.data.shapefile(data=gla, shp=shp, dbf=dbf)
class(sp.gla)
```

```
## [1] "SpatialPolygonsDataFrame"
## attr(,"package")
## [1] "sp"
```

Then add the raw proportions to the `sp.gla` object using the following code:

```
sp.gla@data$propJSA2011 <- sp.gla@data$JSA2011 / sp.gla@data$workpop2011
```

Then the neighbourhood matrix $\mathbf{W}$ used in the spatial correlation models and the `listw` object representation of it used in computing Moran's I statistic are computed using the code:

```
library(spdep)
```

```
## Loading required package: Matrix
```

```
W.nb <- poly2nb(sp.gla, row.names = rownames(sp.gla@data))
W <- nb2mat(W.nb, style = "B")
W.list <- nb2listw(W.nb, style = "B")
```

This provides all the objects we need to undertake the spatial modelling of these data.

3. Modelling spatial data

3.1. Checking for spatial correlation

The first step in a spatial regression problem is to fit a simple model to the data that does not allow for spatial correlation, because the available covariates may capture all the spatial correlation in the data being modelled. To check this one must examine the residuals from a model assuming independence for spatial correlation. If no spatial correlation is present then the analysis may stop there, but if spatial correlation is present it should be modelled. However, in this example there are no covariates, and thus we have to check whether the data themselves contain spatial correlation before fitting a spatial correlation model. Recall that for the 2011 data the **spatial binomial logistic model** is given by:

$$\begin{aligned} \texttt{JSA2011}_k &\sim \text{Binomial}(\texttt{workpop2011}_k, \theta_k) \\ \ln\left(\frac{\theta_k}{1-\theta_k}\right) &= \beta_0 + \phi_k, \end{aligned} \tag{1}$$

where β_0 is an intercept term and $(\phi_1, \ldots, \phi_K)$ is the vector of spatially correlated random effects. The model will thus represent the estimated proportions in poverty $(\theta_1, \ldots, \theta_K)$ on the logit scale (e.g. $\ln(\frac{\theta}{1-\theta})$) as spatially correlated, so before modelling we have to check that the simple proportions $p_k = \texttt{JSA2011}_k/\texttt{workpop2011}_k$ (denoted by `propJSA2011`) are spatially correlated. This can be done by computing Moran's I statistic using the code below:

```
W.list <- nb2listw(W.nb, style = "B")
moran.mc(x = sp.gla@data$propJSA2011, listw = W.list, nsim = 10000)
```

```
##
##  Monte-Carlo simulation of Moran I
##
## data:  sp.gla@data$propJSA2011
## weights: W.list
## number of simulations + 1: 10001
##
## statistic = 0.45265, observed rank = 10001, p-value = 9.999e-05
## alternative hypothesis: greater
```

The result shows these proportions are substantially spatially correlated (p-value < 0.05), meaning that a spatial correlation model is appropriate.

3.2. Fitting the spatial model

Recall from the previous lecture that we fit the spatial correlation model (1) to the data. The set of random effects $\boldsymbol{\phi} = (\phi_1, \ldots, \phi_K)$ are modelled by the CAR prior proposed by Leroux, Lei, and Breslow (2000), which is given by

$$\phi_k | \boldsymbol{\phi}_{-k}, \mathbf{W} \sim \text{N}\left(\frac{\rho \sum_{j=1}^K w_{kj}\phi_j}{\rho \sum_{j=1}^K w_{kj} + 1 - \rho}, \frac{\tau^2}{\rho \sum_{j=1}^K w_{kj} + 1 - \rho}\right), \tag{2}$$

where here ρ is a spatial dependence parameter with $\rho = 0$ corresponding to independence and $\rho = 1$ corresponding to strong spatial correlation. This model can be fitted in a Bayesian setting using Markov chain Monte Carlo (MCMC) simulation, using the `S.CARleroux()` function from the `CARBayes` package. This function requires (at a minimum) the following arguments.

- `formula` - specifies the covariates and offset to include in the model.
- `family` - the data likelihood model to fit, in this case a binomial logistic model.

- `data` - where the data (response, covariates, etc) are stored.
- `trials` - the total number of trials for each binomial data point, which here is the `workpop2011` variable.
- `W` - the neighbourhood matrix $\mathbf{W}$.
- `burnin` - the number of samples to throw away as the burnin period.
- `n.sample` - the total number of samples to generate.
- `thin` - only save every `thin` MCMC sample to reduce their correlation.

With these arguments the number of samples used for estimating the parameters in the model is `(n.sample - burnin) / thin`. The model can be fitted using the following code, where the `print()` function prints a summary of the model to the screen. The `verbose=FALSE` argument stops the function updating the user on its progress, which is purely done to make this document look nice! I recommend setting `verbose=TRUE` (the default value) so you can see how long the function has left to run.

```
model1 <- S.CARleroux(formula=JSA2011~1, family="binomial", data=sp.gla@data,
    trials=sp.gla@data$workpop2011, W=W, burnin=10000, n.sample=60000, thin=50,
    verbose=FALSE)
print(model1)
```

```
##
## #################
## #### Model fitted
## #################
## Likelihood model - Binomial (logit link function)
## Random effects model - Leroux CAR
## Regression equation - JSA2011 ~ 1
## Number of missing observations - 0
##
## ############
## #### Results
## ############
## Posterior quantities and DIC
##
##               Median   2.5%   97.5% n.sample % accept n.effective
## (Intercept) -3.0405 -3.056 -3.0257     1000     35.2        1000
## tau2         1.2451  1.130  1.3849     1000    100.0        1000
## rho          0.9383  0.870  0.9811     1000     45.7        1000
##             Geweke.diag
## (Intercept)        -0.3
## tau2                0.2
## rho                -0.3
##
## DIC =  7889.078       p.d =  970.693       Percent deviance explained =  98.71
```

The output from the `print()` function is split into 2 sections. The first section `Model fitted`

displays the model that has been fitted, which includes the choice of covariates, the data likelihood model and the random effects model. The second section presents the results, which includes both parameter summaries for key parameters and overall model fit criteria such as the Deviance Information Criterion (`DIC`, Spiegelhalter et al. (2002)) with the effective number of parameters (`p.d`), and the percentage of the deviance explained by the model. The summary table of the key model parameters (all parameters except the random effects ϕ) contains the following information:

- `Median` - point estimate for the parameter, which is the median of the samples generated.
- `(2.5%, 97.5%)` - 95% credible interval for the parameter.
- `n.sample` - the number of post burnin samples generated.
- `% accept` - the acceptance probability for the Markov chain.
- `n.effective` - the effective number of independent samples generated, as the set of samples generated are correlated.
- `Geweke.diag` - the convergence diagnostic for the samples proposed by Geweke (1992), which is in the form of a Z-score. Values within the interval (-1.96, 1.96) are indicative of convergence.

Tip - When fitting a model in a Bayesian setting, 95% uncertainty intervals are called **95% credible intervals** and not 95% confidence intervals.

The three parameters highlighted in above summary have the following interpretations:

- `(Intercept)` - is the average (across Glasgow) estimated proportion of people in poverty on the logit scale.
- `tau2` - is the amount of spatial variation in the data. The larger the value the higher the spatial variation in the estimated proportion in poverty across Glasgow.
- `rho` - is the level of spatial correlation in the proportion in poverty across Glasgow. A value of zero corresponds to no spatial correlation (independence), while a value close to 1 corresponds to strong spatial correlation.

The fitted model object `model1` is an `R list` object, which contains the following elements as shown via the `summary()` function.

```
summary(model1)
```

```
##                      Length Class      Mode
## summary.results        21   -none-     numeric
## samples                 6   -none-     list
## fitted.values        1161   -none-     numeric
## residuals               3   data.frame list
## modelfit                7   -none-     numeric
## accept                  4   -none-     numeric
## localised.structure     0   -none-     NULL
## formula                 3   formula    call
## model                   2   -none-     character
## X                    1161   -none-     numeric
```

A description of the key elements in this list is given below.

- `summary.results` - the summary table of results produced when using the `print()` function.
- `samples` - a list of the parameter samples generated by the model.
- `fitted.values` - a vector of fitted values (in this case the fitted values for the binomial model are $\texttt{workpop2011}_k \times \hat{\theta}_k$, where $\hat{\theta}_k$ is the estimated proportion for area k).
- `residuals` - a matrix with 3 different types of residuals.
- `modelfit` - a vector containing model fit criteria including the DIC and the percentage deviance explained.

3.3. Checking convergence

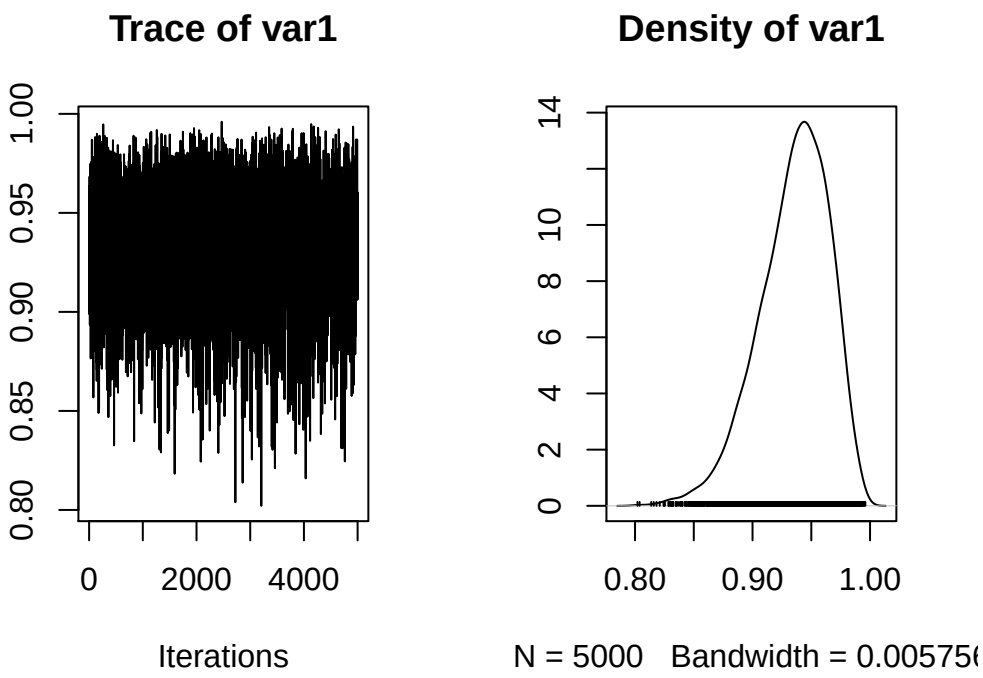
The convergence of the MCMC samples can be assessed by viewing traceplots of the samples for certain parameters. The set of samples for a given parameter have converged if they show no trend and random scatter above and below the average value. The samples are stored in the `samples` element of the `R list` object `model1`, which for this model has elements

```
summary(model1$samples)
```

```
##        Length  Class Mode
## beta      5000 mcmc  numeric
## phi    5805000 mcmc  numeric
## tau2      5000 mcmc  numeric
## rho       5000 mcmc  numeric
## fitted 5805000 mcmc  numeric
## Y            1 mcmc  logical
```

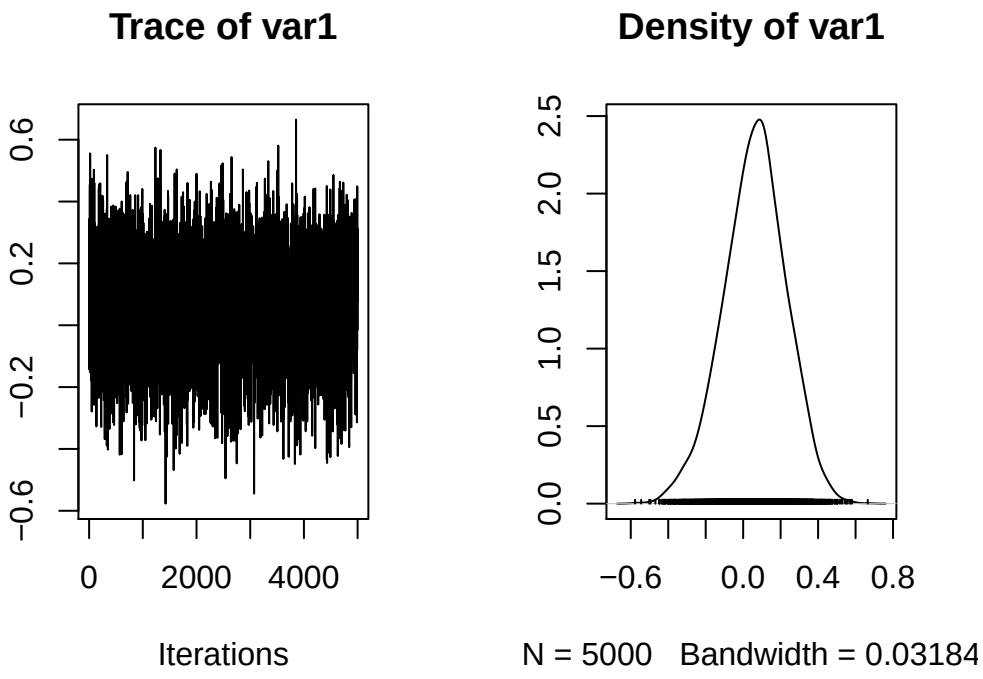
which correspond to the different parameters in the model. For example, to plot the traceplot for the spatial dependence parameter ρ use the following code.

```
plot(model1$samples$rho)
```



In this plot the left panel is the traceplot which shows no trend and hence convergence, while the right panel shows a density estimate of the samples. Convergence for the other parameters can be checked in similar ways. For example, the random effect ϕ_{457} in the 457th area can be checked using the code:

```
plot(model1$samples$phi[ ,457])
```



3.4. Assessing model adequacy

An important step in the modelling process is checking the fitted model is adequate, which for normal linear models includes things such as normality of the residuals, constant variance etc. Here, as the data are non-normal (binomial) one may not expect that the residuals are normally distributed with constant variance. However we can still check for outliers by plotting:

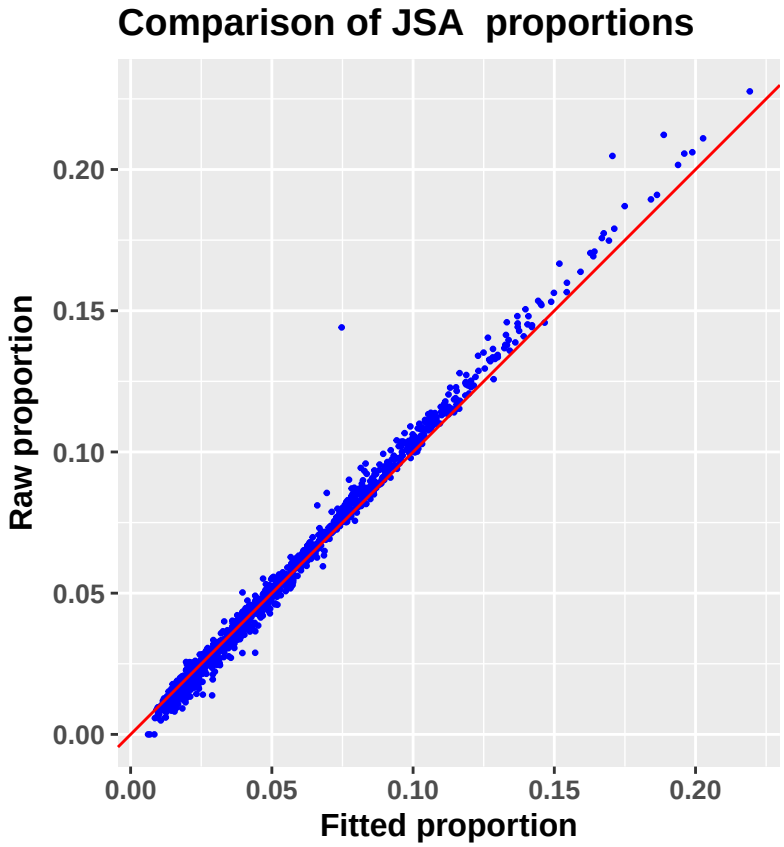
- the fitted proportions from the model against the simple raw proportions.

Outliers will show themselves by not following the pattern in the rest of the plot. Before drawing this plot we need to add the estimated proportions to the `sp.gla` spatial data set, which can be done using the following code.

```
sp.gla@data$fitted.prop <- model1$fitted.values / sp.gla@data$workpop2011
```

We now plot the fitted proportions from the model against the simple raw proportions using the code below.

```
library(ggplot2)
ggplot(sp.gla@data, aes(x=fitted.prop, y=propJSA2011)) +
    geom_point(colour="blue", size=0.5) +
    coord_equal() +
    xlab("Fitted proportion") +
    ylab("Raw proportion") +
    ggtitle("Comparison of JSA  proportions") +
    geom_abline(slope=1, intercept=0, color="red") +
    theme(text=element_text(face="bold", size=12))
```



With the exception of a small number of outliers, the model appears to fit the data well.

3.5. Mapping the estimated proportions

The next stage in this session is to map the estimated proportions in poverty, which uses similar code to that illustrated in the previous practical session. First, we need to turn the `sp.gla` object into a `data.frame` object to enable plotting via `ggplot()` as shown below:

```
library(rgeos)
```

```
## rgeos version: 0.3-23, (SVN revision 546)
##  GEOS runtime version: 3.6.1-CAPI-1.10.1 r0
##  Linking to sp version: 1.2-4
##  Polygon checking: TRUE
```

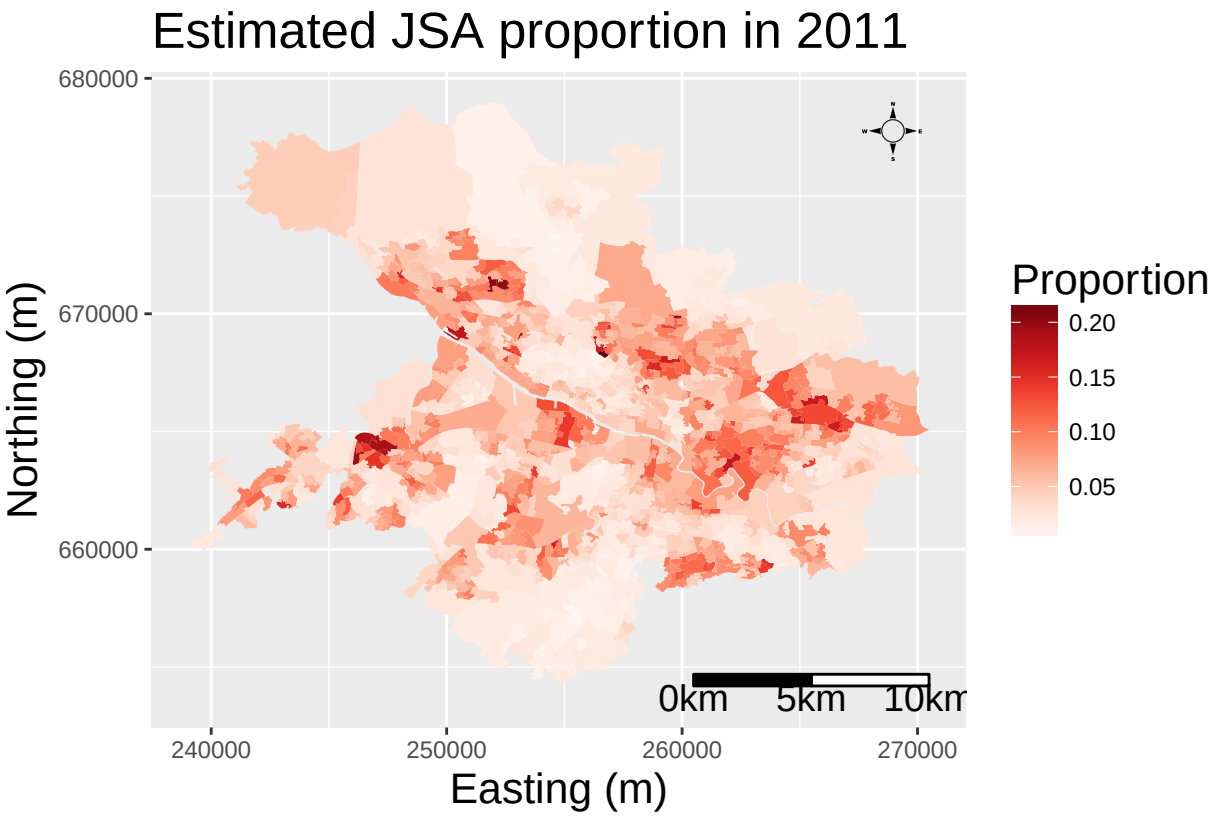
```
library(maptools)
```

```
## Checking rgeos availability: TRUE
```

```
sp.gla@data$id <- rownames(sp.gla@data)
temp1 <- fortify(sp.gla, region = "id")
sp.gla2 <- merge(temp1, sp.gla@data, by = "id")
```

The first three lines load the libraries required, while the last three create the `data.frame` object `sp.gla2` that can be plotted by `ggplot()`. Then the plot can be created using the following code, which is similar to that in the previous practical session.

```
library(ggsn)
library(RColorBrewer)
ggplot(data = sp.gla2, aes(x=long, y=lat, goup=group, fill = c(fitted.prop))) +
    geom_polygon() +
    coord_equal() +
    xlab("Easting (m)") +
    ylab("Northing (m)") +
    labs(title = "Estimated JSA proportion in 2011", fill = "Proportion") +
    theme(title = element_text(size=16)) +
    scale_fill_gradientn(colors=brewer.pal(n=9, name="Reds")) +
    north(sp.gla2, location="topright", symbol=16) +
    scalebar(sp.gla2, dist=5)
```



3.6. Estimating the index of Dissimilarity

A large number of segregation indices have been proposed in the literature, and a seminal critique is given by Massey and Denton (1988). They identify five different dimensions to measuring the degree of segregation: Evenness, Exposure, Concentration, Centralisation and

Clustering. Here we focus on the Dissimilarity index, which is a measure of evenness. Recall that we have K datazones (areas), for which we have the following data:

- N_k - the total number of people in the kth datazone, which here is the total number of working age people ($\texttt{workpop2011}_k$).
- θ_k - an estimate of the proportion of people who are in poverty, which here is the estimated proportion of working age people claiming JSA.

The Dissimilarity index D is computed as

$$D = \sum_{k=1}^{K} \frac{N_k|\theta_k - \theta|}{2N\theta(1-\theta)} \tag{3}$$

Here:

- $N = \sum_{k=1}^{K} N_k$ - is the total number of people in all $K = 1161$ datazones in Glasgow.
- θ - is the overall estimated proportion of working age people in Glasgow in poverty.

The value of D lies in the interval $[0, 1]$, where 0 represents complete evenness (i.e. $\theta_k = \theta$ in all datazones k) and 1 represents complete segregation where θ_k equals zero or one in each datazone.

Therefore we take the following steps to compute an estimate and a 95% uncertainty interval for D.

1. Fit the model that accounts for the spatial correlation in the data. The result of fitting this model produces M (here $M = 1,000$) samples of each of the parameters (β_0, ϕ_k), etc. Each of the M samples of (β_0, ϕ_k) is used to compute a sample of the estimated proportions θ_k for each area k.

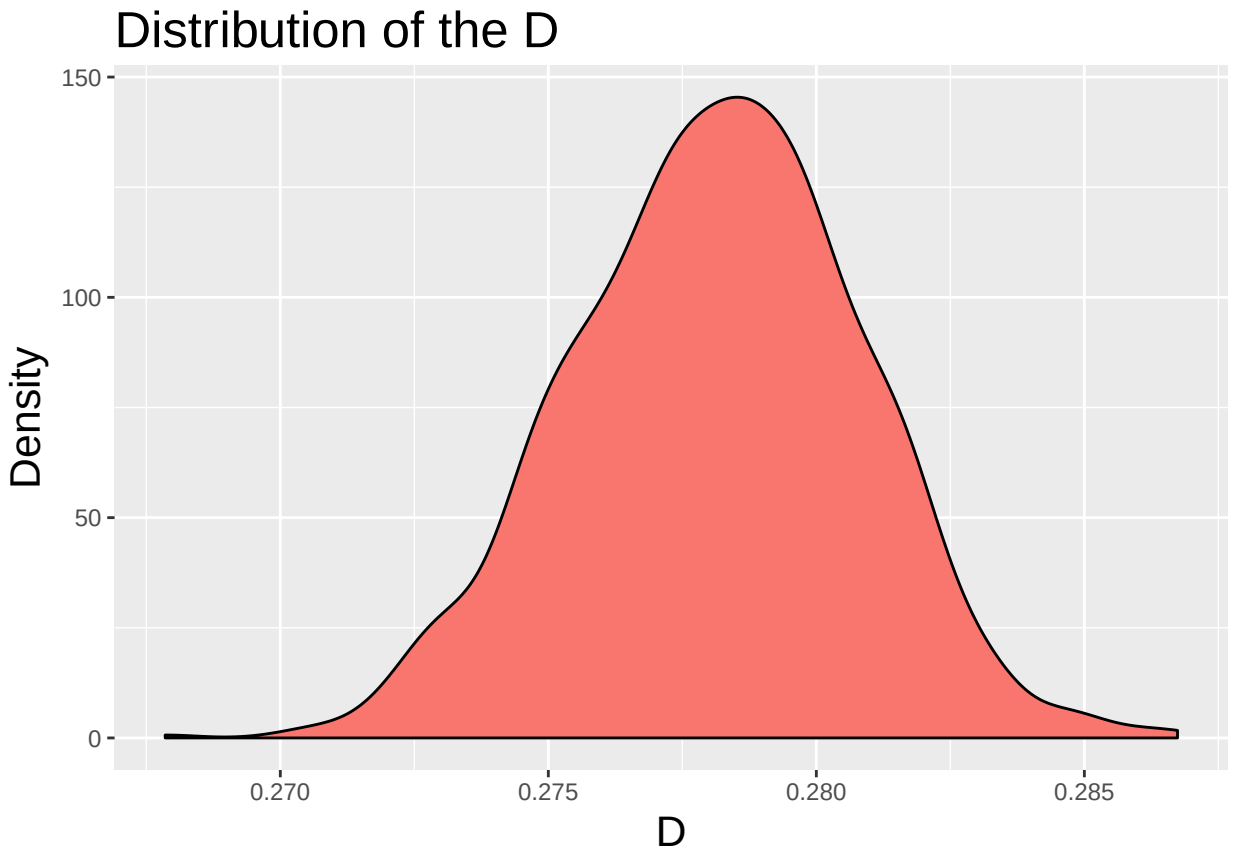
2. For each of the M sets of estimated proportions $(\theta_1^{(r)}, \ldots, \theta_K^{(r)})$ as r ranges between $1, \ldots, M$ compute D .

3. From 2. we now have a distribution of M D values $(D^{(1)}, \ldots, D^{(M)})$, which is called the **posterior distribution**. Then we compute the median of this distribution for a point estimate (or the mean), and a 95% credible interval to quantify uncertainty is computed as the (2.5, 97.5) percentiles of this distribution. To see this in action we compute the posterior distribution for D using the code below.

```
D.dist <- rep(NA, 1000)
    for(i in 1:1000)
    {
    #### Compute the dissimilarity index
    theta.all <- sum(model1$samples$fitted[i, ]) / sum(sp.gla@data$workpop2011)
    theta <- model1$samples$fitted[i, ] / sp.gla@data$workpop2011
    D <- sum(sp.gla@data$workpop2011 * abs(theta - theta.all)) /
    (2 * sum(sp.gla@data$workpop2011) * theta.all * (1-theta.all))
    D.dist[i] <- D
    }
```

Here the lines within the `for()` loop compute D for a single MCMC sample, and then the `for()` loop does this computation for the 1,000 samples. This distribution can be visualised using the code below.

```
indices.dist <- data.frame(D=D.dist)
ggplot(indices.dist, aes(x=D)) +
    geom_density(aes(fill="red")) +
    xlab("D") +
    ylab("Density") +
    labs(title = "Distribution of the D") +
    theme(title = element_text(size=16), legend.position = "none")
```



To obtain the estimate (median) and a 95% credible interval use the code below.

```
quantile(indices.dist$D, c(0.5, 0.025, 0.975))
```

```
##       50%      2.5%     97.5%
## 0.2782594 0.2728469 0.2830972
```

4. Further example

The second example focuses on the spatial pattern in average property prices across Greater Glasgow, Scotland, in 2008. This is an ecological regression analysis, whose aim is to identify the factors that affect property prices and quantify their effects. The data come from the Scottish Statistics database (*http://statistics.gov.scot*), and the study region is the Greater Glasgow and Clyde health board (GGHB), which is split into 271 intermediate geographies (IG). These IGs are small areas that have a median area of 124 hectares and a median population of 4,239. The data are stored in `housedata.csv`, and contain the following columns.

- `IG` - A unique code identifying each intermediate geography.
- `price` - The average property price.
- `crime` - The crime rate in terms of the number of crimes per 10,000 people.
- `rooms` - The average number of rooms in a property.
- `sales` - The number of property sales in 2008.
- `driveshop` - The average time taken to drive to the nearest shopping centre.
- `type` - The most prevalent property type.

Modelling here will be done on the natural log of the `price` variable, because the data on the original scale are skewed and not appropriate for modelling with a Gaussian distribution. The shapefile elements for the IGs are stored in `IG.shp` and `IG.dbf`.

Task - Model the log of the `price` variable in terms of the covariates and the spatial random effects, and quantify the effects of each covariate on log property price.

References

Geweke, John. 1992. "Evaluating the Accuracy of Sampling-Based Approaches to the Calculation of Posterior Moments." In *Bayesian Statistics*, 169–93. University Press.

Kavanagh, L, D Lee, and G Pryce. 2016. "Is Poverty Decentralizing? Quantifying Uncertainty in the Decentralization of Urban Poverty." *Annals of the American Association of Geographers* 106: 1286–98.

Leroux, Brian G., Xingye Lei, and Norman Breslow. 2000. "Statistical Models in Epidemiology, the Environment, and Clinical Trials." In, 179–91. Springer-Verlag, New York. http://dx.doi.org/10.1007/978-1-4612-1284-3_4.

Massey, D, and N Denton. 1988. "The Dimensions of Residential Segregation." *Social Forces* 67: 281–315.

Spiegelhalter, D, N Best, B Carlin, and A Van der Linde. 2002. "Bayesian Measures of Model Complexity and Fit." *Journal of the Royal Statistical Society B* 64: 583–639.