Lattice data

Alexander J Ohrt, Mikel Majewski, Victor Duque

14 diciembre, 2021

This project deals with the data of the incidence of larynx cancer diagnosed in a period of 10 years in two districts in the north-east of England, Mersey and West Lancashire. Both regions are divided into 144 electoral districts with a total of 876 cases of canacer detected. The dataset also include the number of expected cases calculated via internal standarization using the population census of the 10th year of the studied period.

Load libraries

1. Exploratoy analysis in search of spatial correlation.

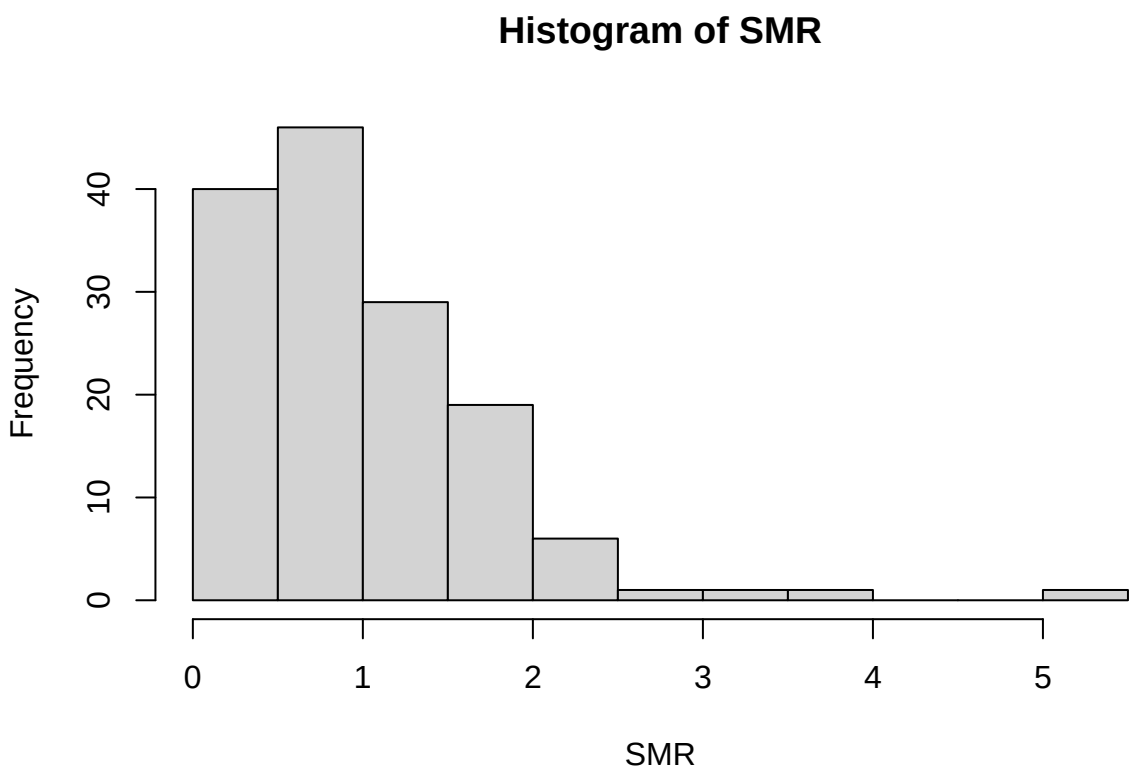
1.1. Build a spatial polygon data frame for the England larynx cancer data.

First of all we read the mapa, create the SMR variable and plot the Mersey and West Lancashire map and the histogram and map with data to see how it is distributed:

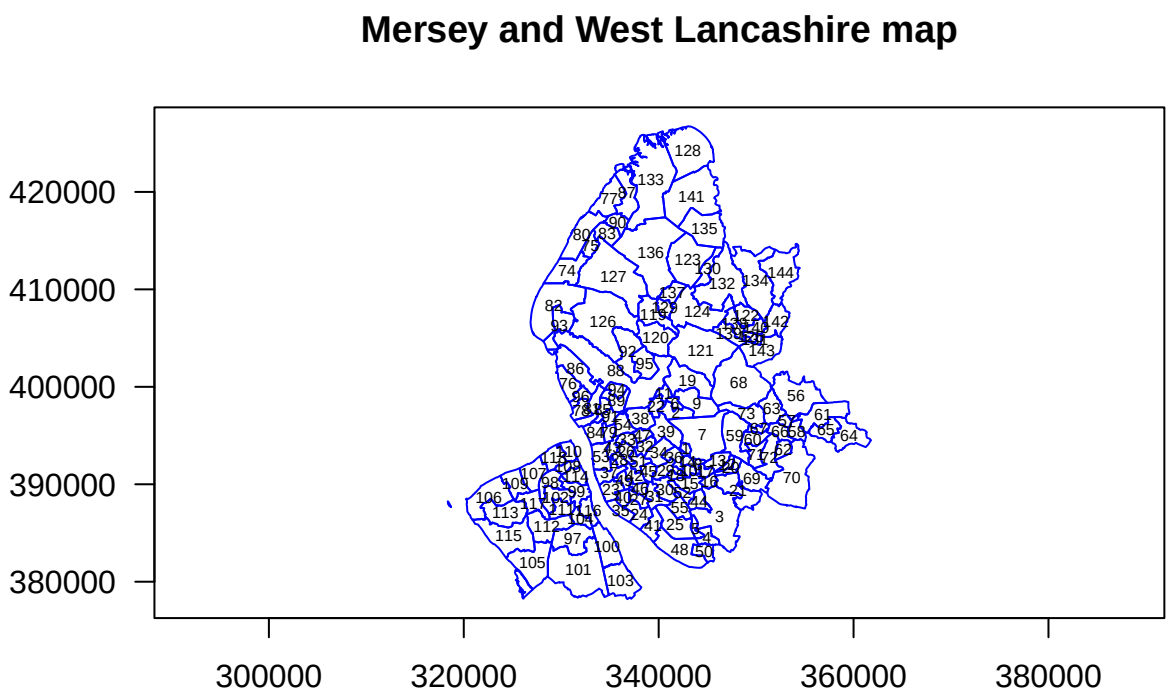
```
invisible(nwen <-sf::st_read("NWEngland.shp"))
```

```
#> Reading layer 'NWEngland' from data source
#>   'C:\Users\user\Desktop\Uni\Master\Epidemologia espacial\Practica2\NWEngland.shp'
#>   using driver 'ESRI Shapefile'
#> Simple feature collection with 144 features and 3 fields
#> Geometry type: MULTIPOLYGON
#> Dimension:     XY
#> Bounding box:  xmin: 318351 ymin: 378209 xmax: 361798 ymax: 426737
#> CRS:           NA
```

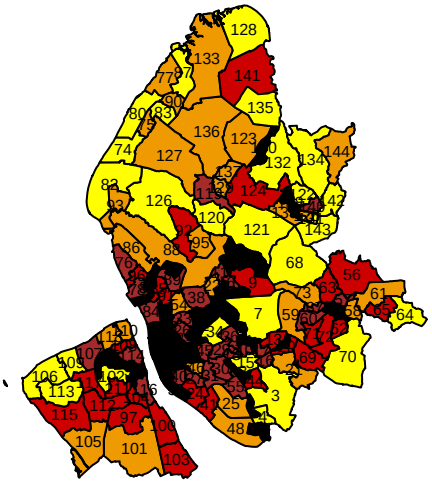
```r
invisible(lardata <- read.table("larynx_data.txt"))
nwen$O<-lardata$O
nwen$E<-lardata$E

#convert a sf to spatial*Dataframe
invisible(snwen<-sf::as_Spatial(nwen))

## grab the data from the sf object
df <- nwen
df$geometry <- NULL
df <- as.data.frame(df)

# create the SpatialPolygonsDataFrame
invisible(nwen.sids <- sp::SpatialPolygonsDataFrame(snwen, data = df ))
#slot(nwen.sids,"data")

#Compute larynx cancer incidence during 10 years (1982-1991) as the Standardized morbi
SMR<-nwen.sids$O/nwen.sids$E
hist(SMR)
```



```
# We can see this histogram is similar to a Poisson Distribution.

#Add this SMR to de data
nwen.sids$SMR <- SMR
#slot(nwen.sids,"data")

#Plot the map
plot(nwen.sids, border="blue", axes=TRUE, las=1)
text(coordinates(nwen.sids),label=nwen.sids$poly_id,cex=0.5)
title(main=paste("Mersey and West Lancashire map"))
```




```
#Plot SMR
breaks <- round(quantile(nwen.sids$SMR, probs=seq(0,1,0.2)), digits=2)
colours <- c("yellow", "orange2", "red3", "brown", "black")

plot(nwen.sids, col=colours[findInterval(nwen.sids$SMR, breaks,all.inside=TRUE)])
legend(x=c(-84, -80), y=c(33, 34.5), legend=leglabs(breaks),fill=colours, bty="n",cex=0.
title(main=paste("SMR in Mersey and West Lancashire"))
text(coordinates(nwen.sids),label=as.factor(nwen.sids$poly_id),cex=0.5)
```



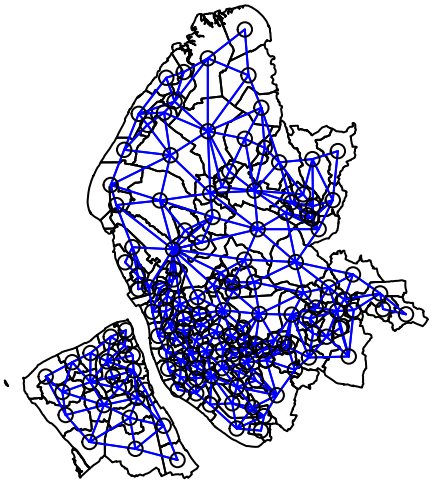
As we fisrt see on the histogram it seems that it follows a Poisson or Gamma distribution. Therefore we have hight frequencies at low SMR values which then decreases.

Once the map is buildt, We can see that it seems to be some clusters of hight incidence in the South-West on Lancashire and in the North-East in Mesey. While there is not a clear pattern nor a cluster of low incidence, which seems randomly distributed.

1.2. Construct the neighbors adjacency matrix.

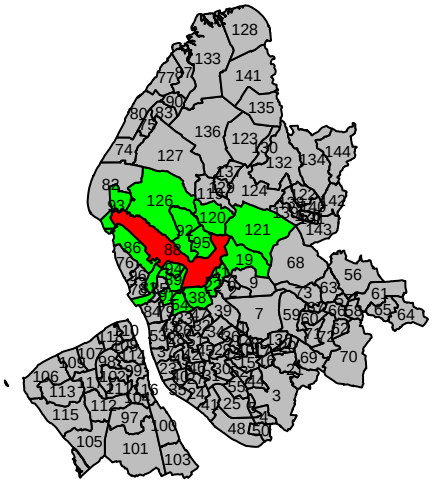
Here we show the adjecency map and the regions with more and less neighbours.

```
neig <- poly2nb(nwen.sids)
plot(nwen.sids)
plot(neig, coordinates(nwen.sids), add=TRUE, col="blue")
```

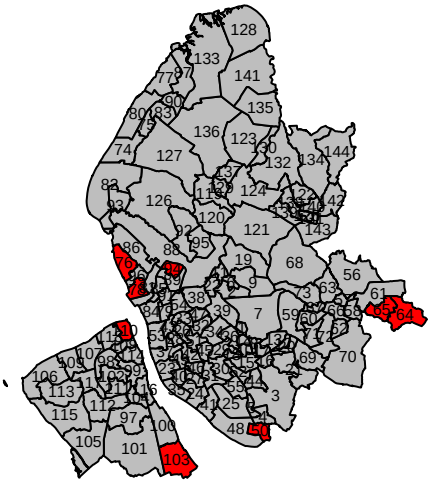


```r
#Show the region with more neighbors
cards <- card(neig) #Cardinalities for neighbours lists
invisible(sort(cards))
#which(cards == max(cards))
maxconts <- which(cards == max(cards))[1]
fg <- rep("grey", length(cards))
fg[maxconts] <- "red"
fg[neig[[maxconts]]] <- "green"
plot(nwen.sids, col=fg)
text(coordinates(nwen.sids), label=nwen.sids$poly_id,cex=0.5)
title(main="Region with largest number of contiguities")
```

Region with largest number of contiguities




```
#Show the region with less neighbors
cards <- card(neig)
invisible(sort(cards,decreasing = FALSE))
invisible(which(cards == min(cards)))
minconts <- which(cards == min(cards))[1:8]
fg <- rep("grey", length(cards))
fg[minconts] <- "red"
plot(nwen.sids, col=fg)
text(coordinates(nwen.sids), label=nwen.sids$poly_id,cex=0.5)
title(main="Region with lowest number of contiguities")
```



In order to see area effects it's important to take into account contiguity. In this case neightbouing districts. This way we will be able to see if the incidence of one region has some type of correlation with the inicidence of it's neightbours. Having more adjacent areasimplies that probably the region recieves influence from more laces than if the area is isolated where in case of having an influent neightbour will be more noticeable.

In these maps we can see that there is a cluser of zones with the larger number of neightbouring districts all of them being around the middle of the map and midelseized. On the other hand the regions with the least number of contiguities are mainly located at the corners of the map with some exceptions due to the shape of the sorrounding districts.

1.3. Calculate Moran's I and Geary's C.

We calculate this statistics to see the independence or the correlation of the data we have.

```
#ASSIGN WEIGHTS TO THE AREAS THAT ARE LINKED

#nb2listw:The function supplements a neighbours list with spatial weights
#for the chosen coding scheme.

#a.#B:Binary (1: neighbour, 0:otherwise)
```

```r
#b.#W: standardised to sum unity row.

w.sids<-nb2listw(neig, glist=NULL, style="W",  zero.policy=TRUE)   #Spatial weights for

summary(unlist(w.sids$weights))
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>  0.0625  0.1429  0.2000  0.1920  0.2000  0.5000
```

```r
summary(sapply(w.sids$weights,sum))
```

```
#>    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
#>       1       1       1       1       1       1
```

```r
#Morans test

sids.moran<-moran.test(nwen.sids$SMR ,w.sids)
sids.moran$estimate
```

```
#> Moran I statistic       Expectation          Variance
#>       0.313393219      -0.006993007       0.002649439
```

```r
#So we have a positive correlation in the data due we choose the alternative hypothesi

#Geary's test

sids.geary<-geary.test(nwen.sids$SMR, w.sids)
sids.geary$estimate
```

```
#> Geary C statistic       Expectation          Variance
#>       0.661678814       1.000000000       0.004979933
```

```r
#In the same way we choose the aternative hypothesis but this time correlation is nega
```

Applying Moran's and Geary's Tests to test the independence of the data considering the weights of each region.

In Moran's test we have that the statistic is greater than the expected value. The p-value is very little what indicates that we choose the alternative hypothesis concluding that we have a positive spatial autocorrelation and the nearest the regions are, the more similar data they have. To be concrete, the I statistic is 0.31 and the expected -0.006. The obtained p-value is 2,4e-10.

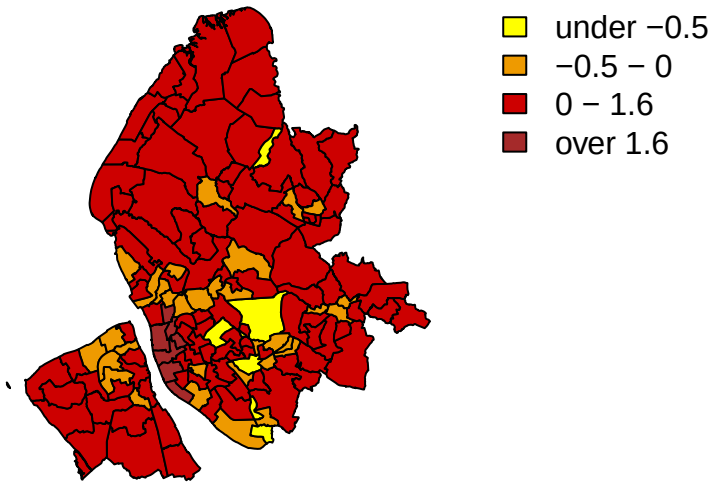
In Gaery's case, the C statistic is 0.66, and the expected value 1. Alse the p-value is little enough to accept the alternative hypothesis (p-value is 8.17 in this case) which means that Gaerys tests indicates a positive spatial autocorrelation as Moran's Test.

So we conclude that we have a positive spatial autocorrelation.

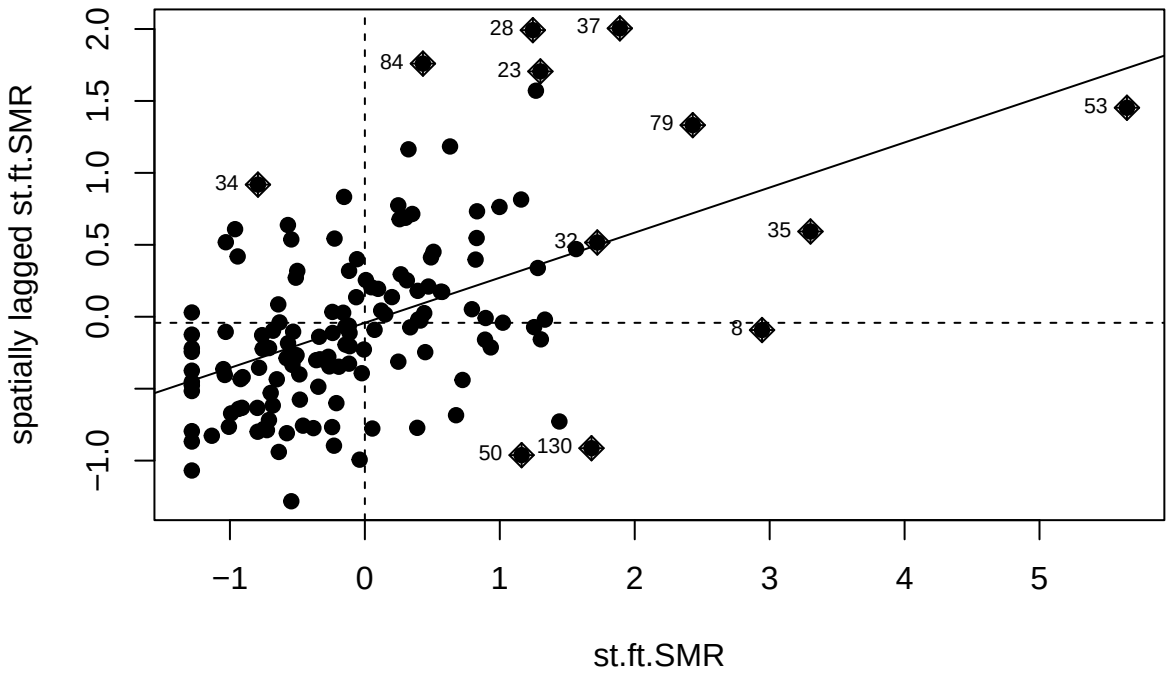
1.4. Local indicators.

Here we show first the map with the local indicators of each region and latter a scaterplot with the regresion of those indicators.

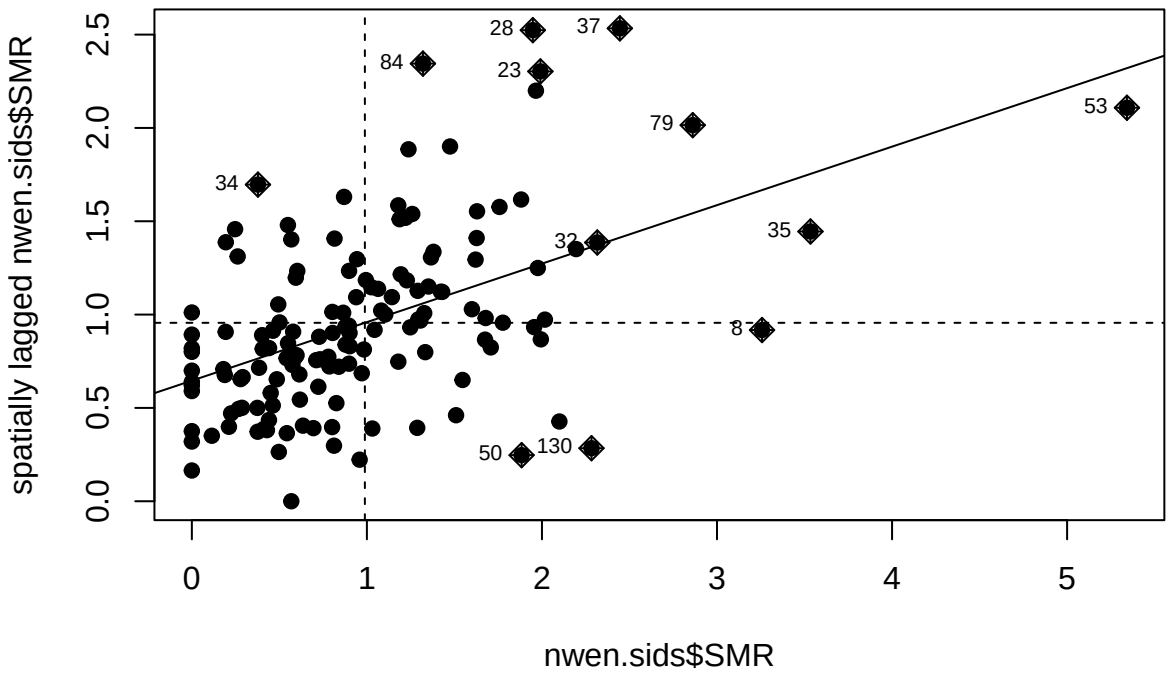
```
sids.local.moran<-localmoran(nwen.sids$SMR, w.sids)
#Northampton is the region 5
#Anson is the region 85
#Richmond is the region number 89

#head(sids.local.moran)
#Only to look the results
invisible(data.frame(nwen.sids$SMR,sids.local.moran))

#Add I Moran results to spatial polygon dataframe

invisible(slot(nwen.sids,"data")<-cbind(slot(nwen.sids,"data"),sids.local.moran))

#Plot  I local Moral
#Intervals

l.inf<-round(min(nwen.sids$Ii),digits=2)
l.sup<-round(max(nwen.sids$Ii),digits=2)
library(classInt)
I.q4<-findInterval(nwen.sids$Ii,c((l.inf-0.1),-0.5,0,1.6,(l.sup+0.1)))
breakI<-c((l.inf-0.1),-0.5,0,1.6,(l.sup+0.1))
pal.I<- c("yellow", "orange2", "red3", "brown")                    #Colours
plot(nwen.sids, col=pal.I[I.q4])                                   #Color assignme
legend("topright", legend=leglabs(breakI),fill=pal.I, bty="n")
```




```
#We can see some clusters in the South-West on Lancashire.

#Scatterplot moran Tests

st.ft.SMR<-(nwen.sids$SMR-mean(nwen.sids$SMR))/sd(nwen.sids$SMR)
xx<-moran.plot(st.ft.SMR,w.sids ,labels=as.factor(nwen.sids$poly_id), pch=19)
```




```
xx<-moran.plot(nwen.sids$SMR,w.sids ,labels=as.factor(nwen.sids$poly_id), pch=19)
```




```
#We see a positive correlation. The most extrem cases are region 28, 37, 50 and 130.
```

We can see in this case that for most of the regions the local I indicator is greater than 0. We can see the same cluster as in the first map (Ex 1.1) in the South-West on Lancashire. So we conclude there is in fact a spatial positive autocorrelation as we thought.

Also we can see in the graphic that regions number 28, 37, 50 and 130 are the most extreme cases.

2. Considering overdispersion (Poisson dostrobition assumed)

```
###Expected cases. SMR>1 means that there is a higher risk in the region of study that

# data used: nwen.sids$SMR

#Fit a Poisson regression
result.pois.lar<-glm(O~1+offset(log(E)),data=nwen.sids,family="poisson")
summary(result.pois.lar)
```

```
#>
```

```
#> Call:
#> glm(formula = O ~ 1 + offset(log(E)), family = "poisson", data = nwen.sids)
#>
#> Deviance Residuals:
#>     Min       1Q   Median       3Q      Max
#> -3.3436  -1.4977  -0.3710   0.7553   6.5680
#>
#> Coefficients:
#>              Estimate Std. Error z value Pr(>|z|)
#> (Intercept) 1.142e-05  3.379e-02       0        1
#>
#> (Dispersion parameter for poisson family taken to be 1)
#>
#>     Null deviance: 396  on 143  degrees of freedom
#> Residual deviance: 396  on 143  degrees of freedom
#> AIC: 862.15
#>
#> Number of Fisher Scoring iterations: 5
```

```
nwen.sids$residuals<-result.pois.lar$residuals

list(residual.deviance           = deviance(result.pois.lar),
     residual.degrees.of.freedom = df.residual(result.pois.lar),
     dispersion.parameter        = deviance(result.pois.lar)/df.residual(result.pois.lar
     chisq.p.value               = pchisq(deviance(result.pois.lar), df.residual(result.
)
```

```
#> $residual.deviance
#> [1] 395.9974
#>
#> $residual.degrees.of.freedom
#> [1] 143
#>
#> $dispersion.parameter
#> [1] 2.769212
#>
#> $chisq.p.value
#> [1] 1.291244e-25
```

```
# The dispertion parameter is 2.77, way bigger than 1, so we can consider that data is

# The quasipoisson family differ from the binomial and poisson families only in that t

result.qpois.lar <- glm(O~1+offset(log(E)), family = quasipoisson(link = log),data = nwe
summary(result.qpois.lar)
```

```
#>
#> Call:
#> glm(formula = O ~ 1 + offset(log(E)), family = quasipoisson(link = log),
#>     data = nwen.sids)
#>
#> Deviance Residuals:
#>     Min       1Q   Median       3Q      Max
#> -3.3436  -1.4977  -0.3710   0.7553   6.5680
#>
#> Coefficients:
#>              Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 1.142e-05  5.949e-02       0        1
#>
#> (Dispersion parameter for quasipoisson family taken to be 3.099842)
#>
#>     Null deviance: 396  on 143  degrees of freedom
#> Residual deviance: 396  on 143  degrees of freedom
#> AIC: NA
#>
#> Number of Fisher Scoring iterations: 5
```

```
summary.glm(result.qpois.lar)$dispersion
```

```
#> [1] 3.099842
```

```
#Fit Binomial Negative.
result.BN.lar <- glm.nb(O~1+offset(log(E)), data = nwen.sids)
summary(result.BN.lar)
```

```
#>
#> Call:
#> glm.nb(formula = O ~ 1 + offset(log(E)), data = nwen.sids, init.theta = 3.519599834,
#>     link = log)
#>
#> Deviance Residuals:
#>     Min       1Q   Median       3Q      Max
#> -2.1794  -0.9484  -0.2280   0.4190   3.5304
#>
#> Coefficients:
#>             Estimate Std. Error z value Pr(>|z|)
#> (Intercept)  0.01241    0.05744   0.216    0.829
```

```
#>
#> (Dispersion parameter for Negative Binomial(3.5196) family taken to be 1)
#>
#>     Null deviance: 153.97  on 143  degrees of freedom
#> Residual deviance: 153.97  on 143  degrees of freedom
#> AIC: 752.92
#>
#> Number of Fisher Scoring iterations: 1
#>
#>
#>               Theta:  3.520
#>           Std. Err.:  0.683
#>
#>  2 x log-likelihood:  -748.923
```

```
result.BN.lar$aic
```

```
#> [1] 752.9226
```

```
result.pois.lar$aic
```

```
#> [1] 862.1544
```

```
#Negative Binomial AIC is lower so it fits better
```

As the Poisson distribution for the number of cases is suposed, first we have to fit the data into the model. Doing this we will be able to if it fits well. After fitting the model we can see that the dispersion parameter is almost 2.77 hence it is way bigger than it should be. If the dispersion parameter was 1 means the is no dispersionat all, but in this case being more than twice the number it is clear that there is overdispersion.

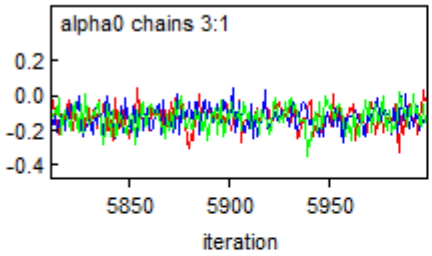
If the data is overdispersed the model won't be very explanatory, therefore, as the reason of overdispersion is unknown, we try to fit an quasi-Poisson distribution that doesn't fix the dispersion parameter at 1 so it can deal with overdispersion modeling it accordingly. Aside from the quasi-Poisson the negative Binomial can be also used. To see which of the two models fits better/ explains more can be used few parameters. In this case we look at the AIC, being the lower the better. Thus, the negative Binomial distribution is the one that fits better the data without carrying overdispersion.

3. Fitting of the models with Gibbs sampling and three chains of initial values

3.1. Heterogenity model

The calculations are done using Winbugs directly. The results can be seen as follows:

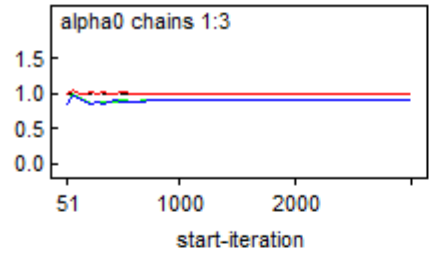
Dynamic trace



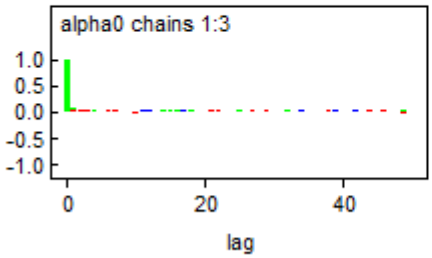
Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
alpha0	-0.1278	0.06193	4.91E-4	-0.2521	-0.1266	-0.01028	1	18000

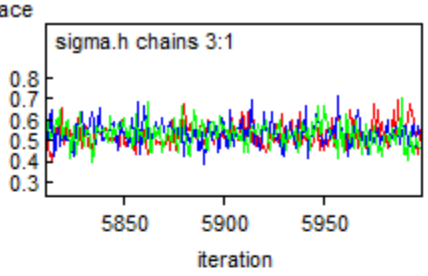
Gelman Rubin statistic



Autocorrelation function



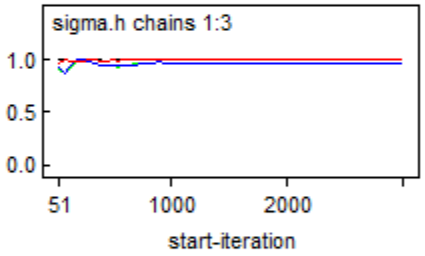
Dynamic trace



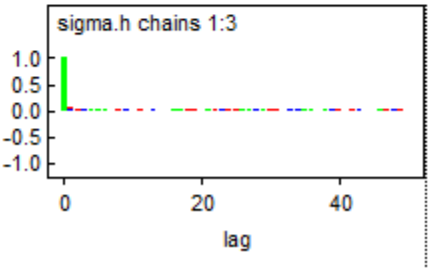
Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
sigma.h	0.5299	0.05438	4.346E-4	0.4299	0.5279	0.6408	1	18000

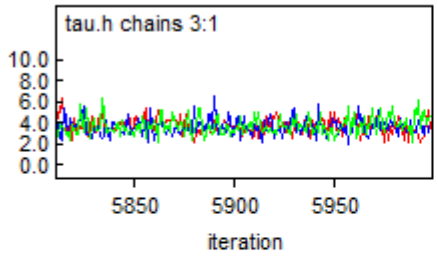
Gelman Rubin statistic



Autocorrelation function



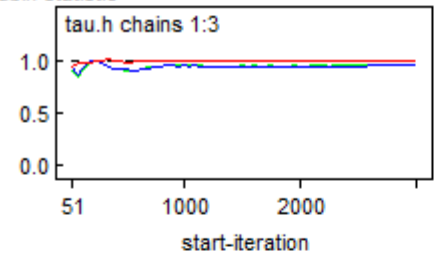
Dynamic trace



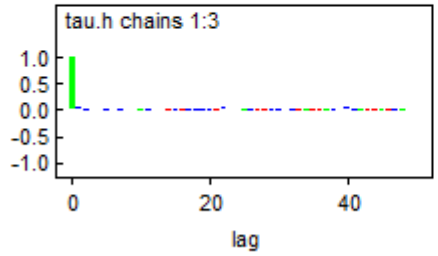
Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
tau.h	3.676	0.7712	0.006148	2.436	3.589	5.411	1	18000

Gelman Rubin statistic



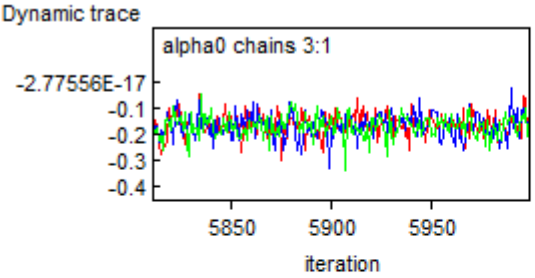
Autocorrelation function



3.2. Conditionaly autoregressive (spatial)

The calculations are done using Winbugs directly. The results can be seen as follows:

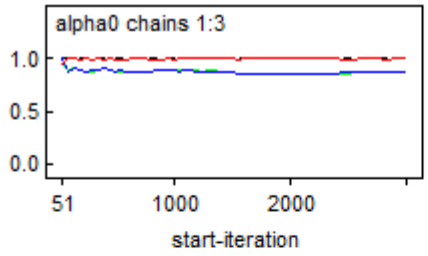
Dynamic trace



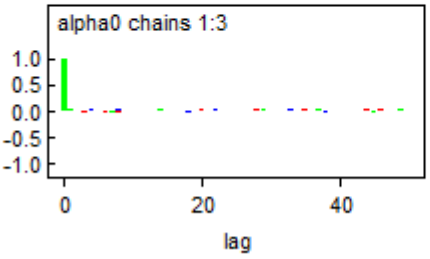
Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
alpha0	-0.165	0.04467	3.224E-4	-0.2543	-0.1647	-0.07863	1	18000

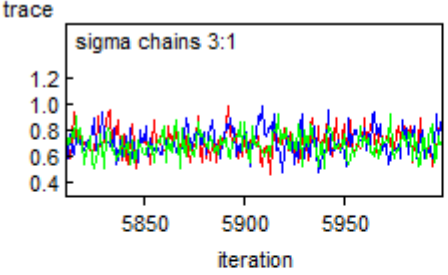
Gelman Rubin statistic



Autocorrelation function



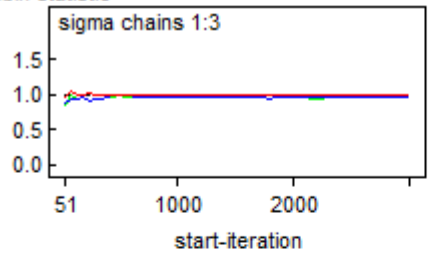
Dynamic trace



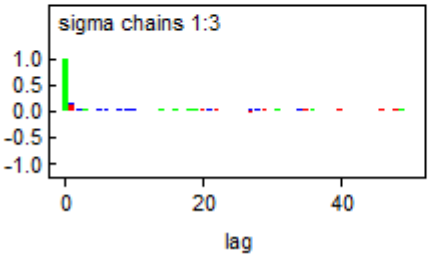
Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
sigma	0.7075	0.09352	7.918E-4	0.5382	0.7031	0.904	1	18000

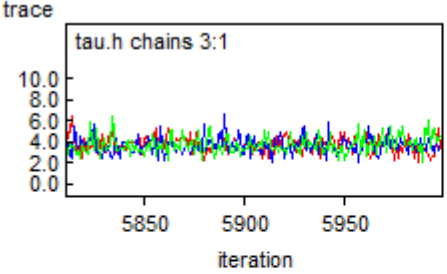
Gelman Rubin statistic



Autocorrelation function



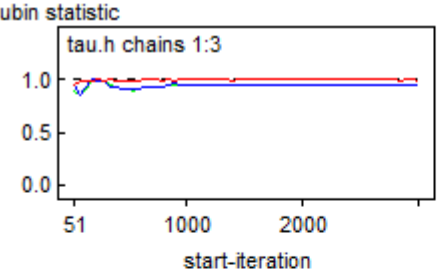
Dynamic trace



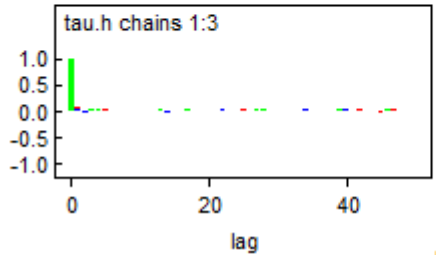
Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
tau.h	3.676	0.7712	0.006148	2.436	3.589	5.411	1	18000

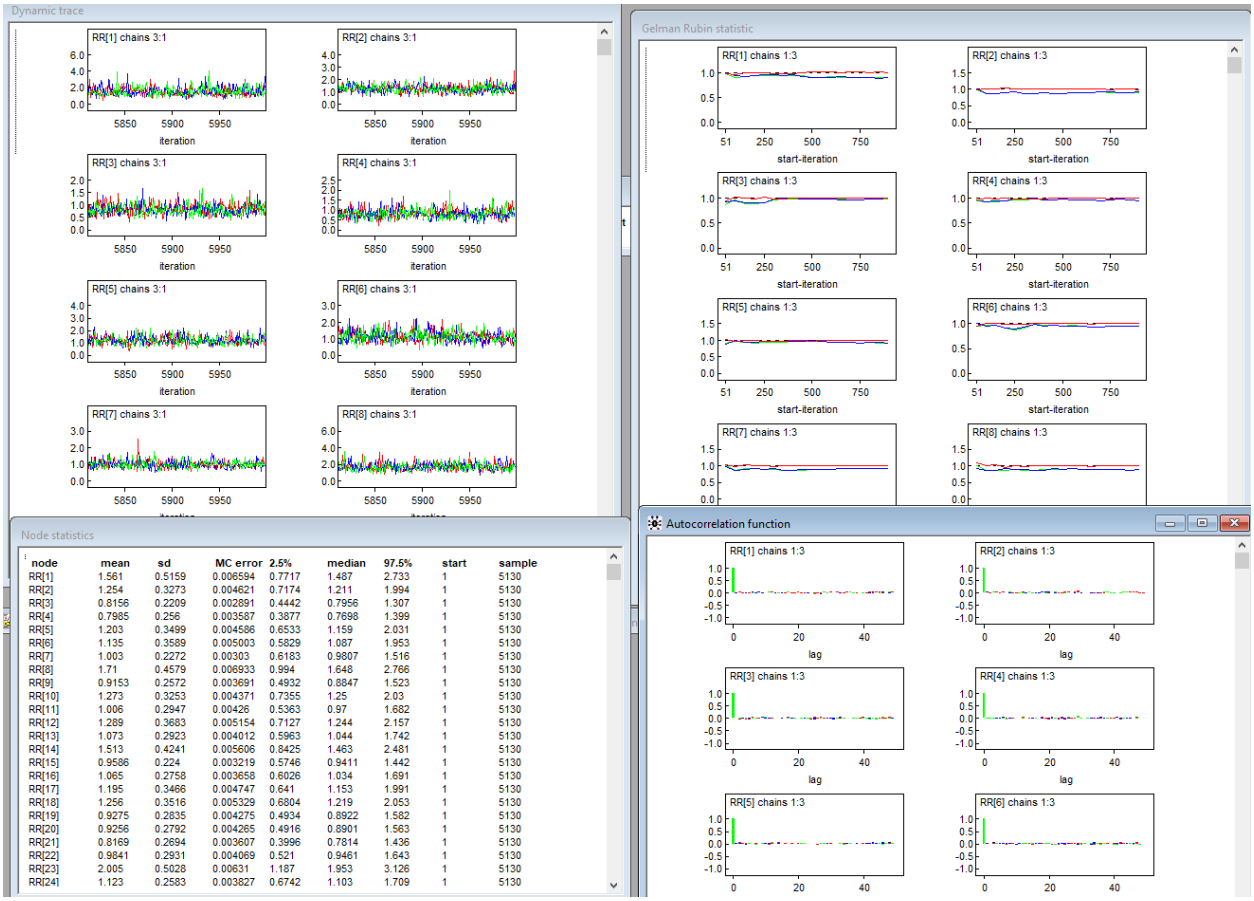
Gelman Rubin statistic



Autocorrelation function



model is syntactically correct

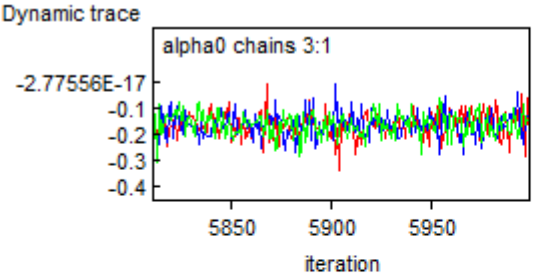


node	mean	sd	MC error	2.5%	median	97.5%	start	sample
RR[1]	1.561	0.5159	0.006594	0.7717	1.487	2.733	1	5130
RR[2]	1.254	0.3273	0.004621	0.7174	1.211	1.994	1	5130
RR[3]	0.8156	0.2209	0.002891	0.4442	0.7956	1.307	1	5130
RR[4]	0.7985	0.256	0.003587	0.3877	0.7698	1.399	1	5130
RR[5]	1.203	0.3499	0.004586	0.6533	1.159	2.031	1	5130
RR[6]	1.135	0.3589	0.005003	0.5829	1.087	1.953	1	5130
RR[7]	1.003	0.2272	0.00303	0.6183	0.9807	1.516	1	5130
RR[8]	1.71	0.4579	0.006933	0.994	1.648	2.766	1	5130
RR[9]	0.9153	0.2572	0.003691	0.4932	0.8847	1.523	1	5130
RR[10]	1.273	0.3253	0.004371	0.7355	1.25	2.03	1	5130
RR[11]	1.006	0.2947	0.00426	0.5363	0.97	1.682	1	5130
RR[12]	1.289	0.3683	0.005154	0.7127	1.244	2.157	1	5130
RR[13]	1.073	0.2923	0.004012	0.5963	1.044	1.742	1	5130
RR[14]	1.513	0.4241	0.005606	0.8425	1.463	2.481	1	5130
RR[15]	0.9586	0.224	0.003219	0.5746	0.9411	1.442	1	5130
RR[16]	1.065	0.2758	0.003658	0.6026	1.034	1.691	1	5130
RR[17]	1.195	0.3466	0.004747	0.641	1.153	1.991	1	5130
RR[18]	1.256	0.3516	0.005329	0.6804	1.219	2.053	1	5130
RR[19]	0.9275	0.2835	0.004275	0.4934	0.8922	1.582	1	5130
RR[20]	0.9256	0.2792	0.004265	0.4916	0.8901	1.563	1	5130
RR[21]	0.8169	0.2694	0.003607	0.3996	0.7814	1.436	1	5130
RR[22]	0.9841	0.2931	0.004069	0.521	0.9461	1.643	1	5130
RR[23]	2.005	0.5028	0.00631	1.187	1.953	3.126	1	5130
RR[24]	1.123	0.2583	0.003827	0.6742	1.103	1.709	1	5130

3.3. Convolution

The calculations are done using Winbugs directly. The results can be seen as follows:

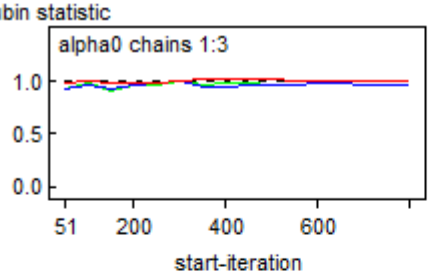
Dynamic trace



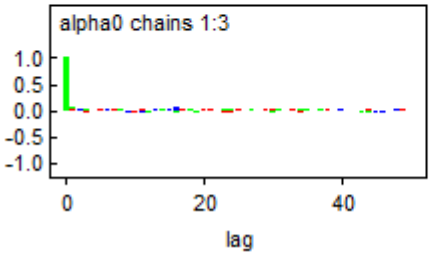
Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
alpha0	-0.167	0.04694	0.001552	-0.2637	-0.1666	-0.08118	1	1050

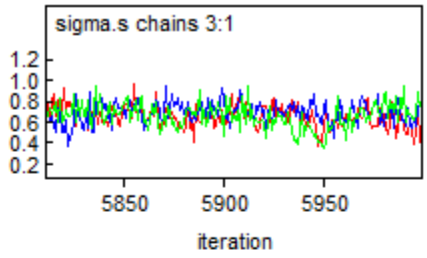
Gelman Rubin statistic



Autocorrelation function



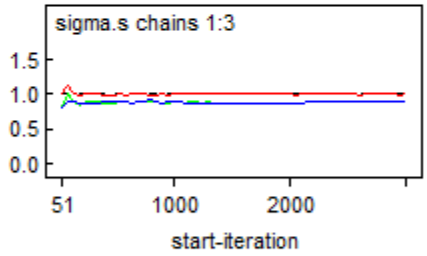
Dynamic trace



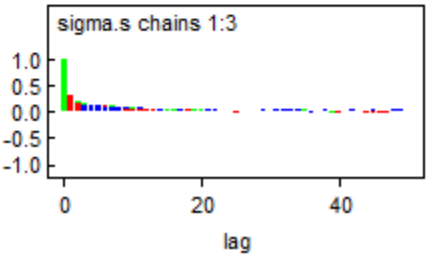
Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
sigma.s	0.6658	0.1031	0.00172	0.4675	0.6644	0.8751	1	18000

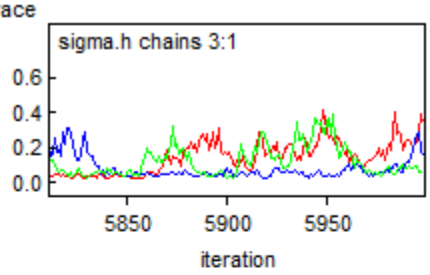
Gelman Rubin statistic



Autocorrelation function



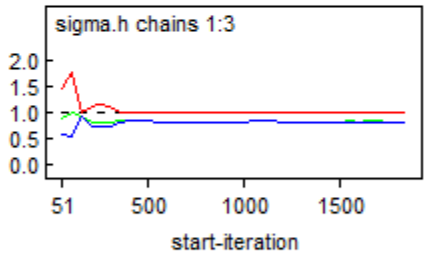
Dynamic trace



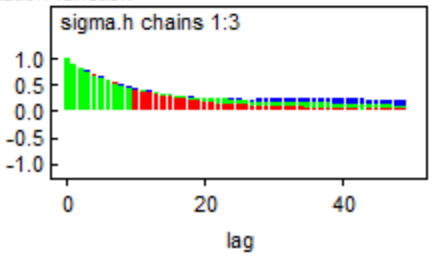
Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
sigma.h	0.126	0.07908	0.00353	0.02608	0.108	0.3081	1	11070

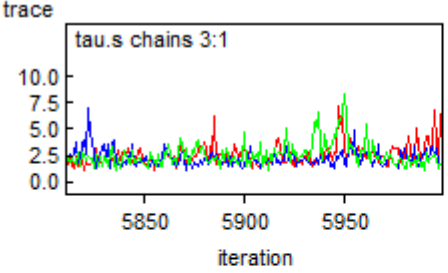
Gelman Rubin statistic



Autocorrelation function



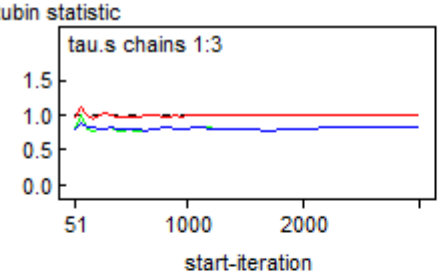
Dynamic trace



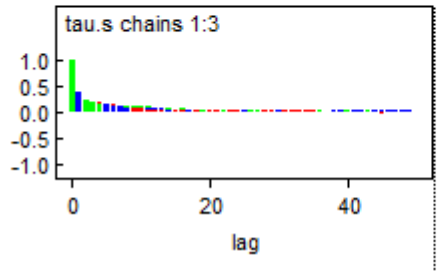
Node statistics

node	mean	sd	MC error	2.5%	median	97.5%	start	sample
tau.s	2.437	0.8574	0.01525	1.306	2.266	4.576	1	18000

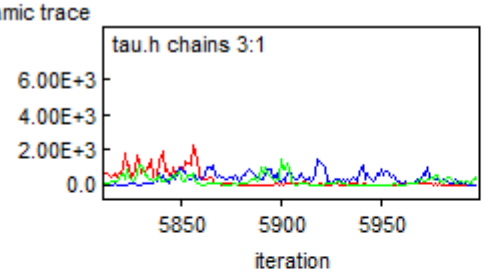
Gelman Rubin statistic



Autocorrelation function



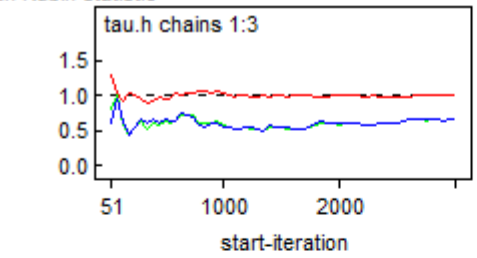
Dynamic trace



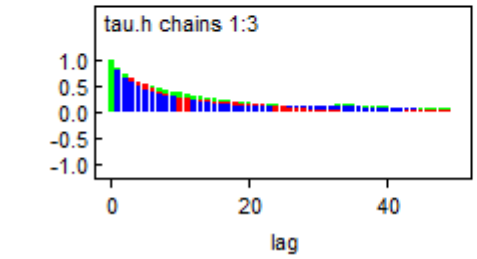
Node statistics

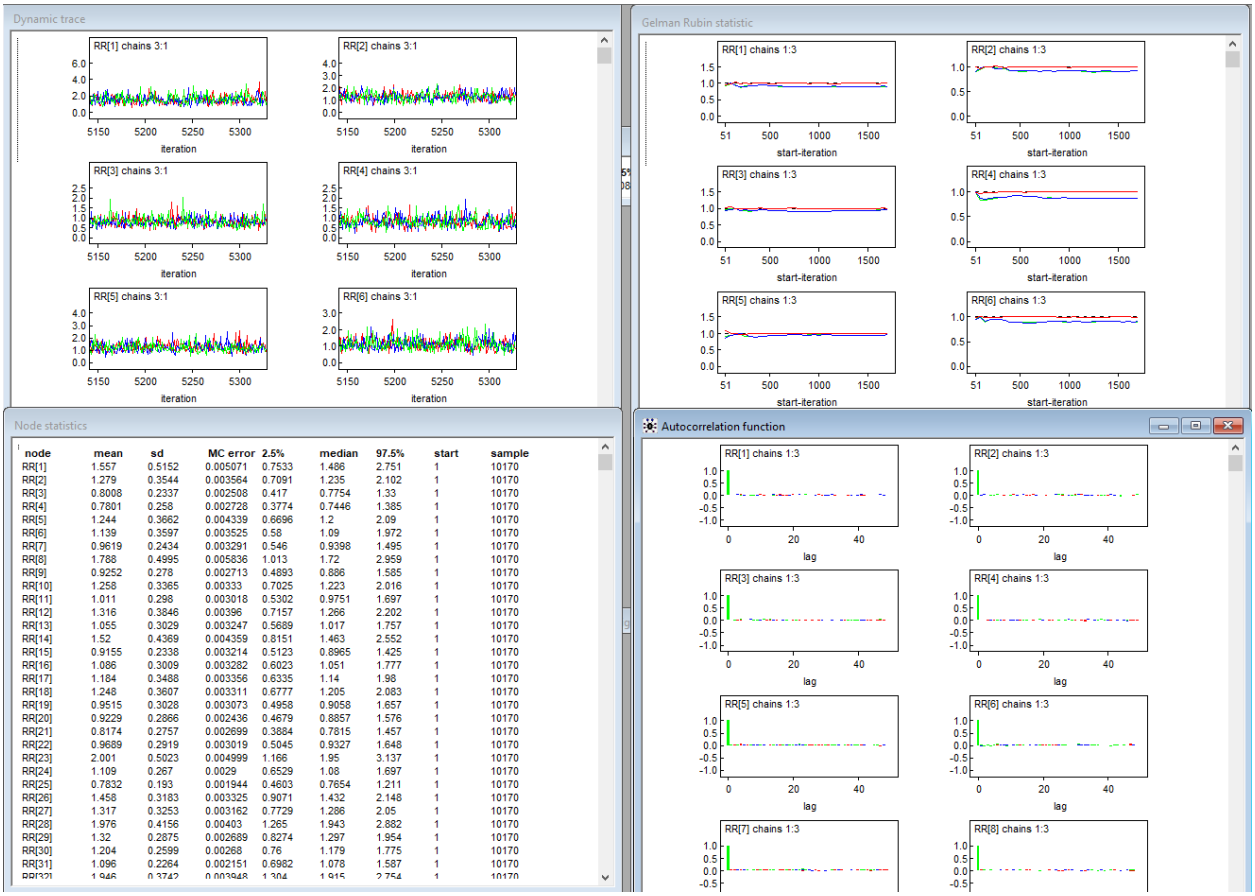
node	mean	sd	MC error	2.5%	median	97.5%	start	sample
tau.h	256.2	443.4	14.06	10.49	89.53	1498.0	1	18000

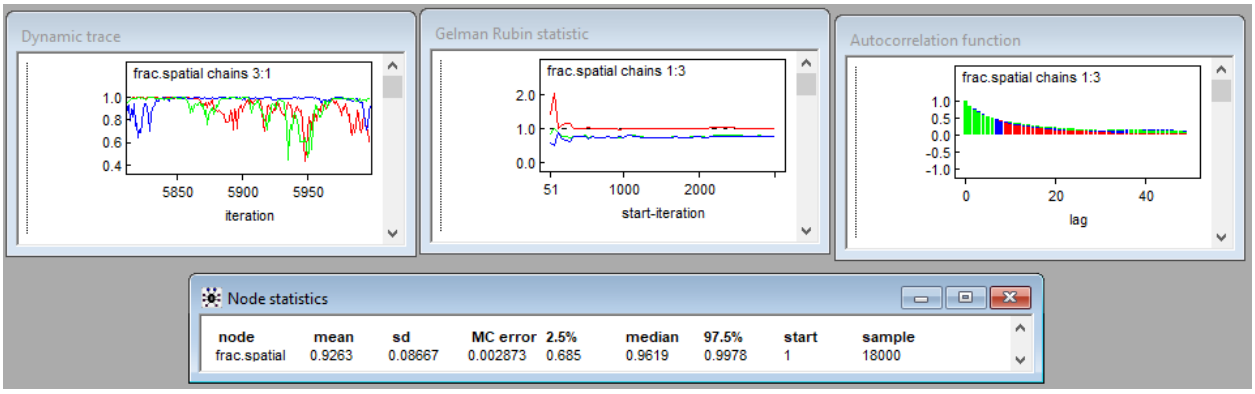
Gelman Rubin statistic

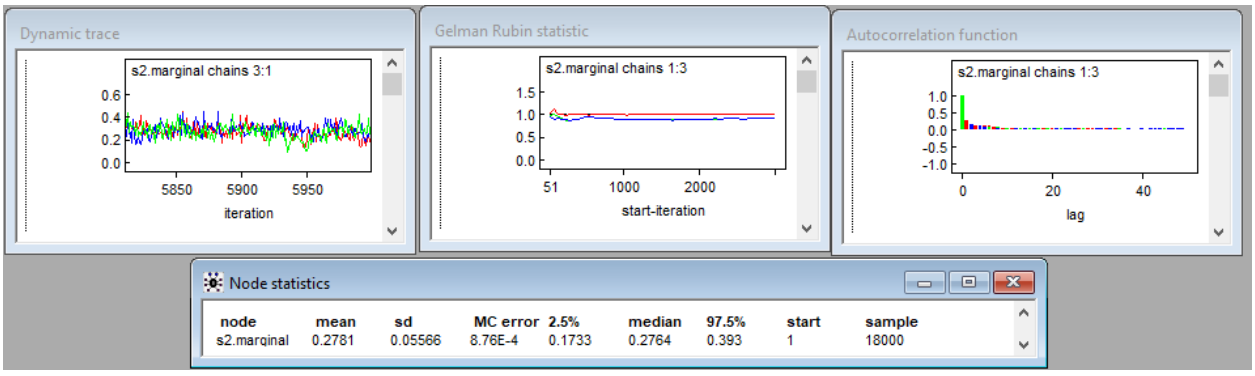


Autocorrelation function




Dynamic trace
RR[1] chains 3:1
RR[2] chains 3:1
RR[3] chains 3:1
RR[4] chains 3:1
RR[5] chains 3:1
RR[6] chains 3:1
iteration
Gelman Rubin statistic
RR[1] chains 1:3
RR[2] chains 1:3
RR[3] chains 1:3
RR[4] chains 1:3
RR[5] chains 1:3
RR[6] chains 1:3
start-iteration
Node statistics
node mean sd MC error 2.5% median 97.5% start sample
RR[1] 1.557 0.5152 0.005071 0.7533 1.486 2.751 1 10170
RR[2] 1.279 0.3544 0.003564 0.7091 1.235 2.102 1 10170
RR[3] 0.8008 0.2337 0.002508 0.417 0.7754 1.33 1 10170
RR[4] 0.7801 0.258 0.002728 0.3774 0.7446 1.385 1 10170
RR[5] 1.244 0.3662 0.004339 0.6696 1.2 2.09 1 10170
RR[6] 1.139 0.3597 0.003525 0.58 1.09 1.972 1 10170
RR[7] 0.9619 0.2434 0.003291 0.546 0.9398 1.495 1 10170
RR[8] 1.788 0.4995 0.005836 1.013 1.72 2.959 1 10170
RR[9] 0.9252 0.278 0.002713 0.4893 0.886 1.585 1 10170
RR[10] 1.258 0.3365 0.00333 0.7025 1.223 2.016 1 10170
RR[11] 1.011 0.298 0.003018 0.5302 0.9751 1.697 1 10170
RR[12] 1.316 0.3846 0.00396 0.7157 1.266 2.202 1 10170
RR[13] 1.055 0.3029 0.003247 0.5689 1.017 1.757 1 10170
RR[14] 1.52 0.4369 0.004359 0.8151 1.463 2.552 1 10170
RR[15] 0.9155 0.2338 0.003214 0.5123 0.8965 1.425 1 10170
RR[16] 1.086 0.3009 0.003282 0.6023 1.051 1.777 1 10170
RR[17] 1.184 0.3488 0.003356 0.6335 1.14 1.98 1 10170
RR[18] 1.248 0.3607 0.003311 0.6777 1.205 2.083 1 10170
RR[19] 0.9515 0.3028 0.003073 0.4958 0.9058 1.657 1 10170
RR[20] 0.9229 0.2866 0.002436 0.4679 0.8857 1.576 1 10170
RR[21] 0.8174 0.2757 0.002699 0.3884 0.7815 1.457 1 10170
RR[22] 0.9689 0.2919 0.003019 0.5045 0.9327 1.648 1 10170
RR[23] 2.001 0.5023 0.004999 1.166 1.95 3.137 1 10170
RR[24] 1.109 0.267 0.0029 0.6529 1.08 1.697 1 10170
RR[25] 0.7832 0.193 0.001944 0.4603 0.7654 1.211 1 10170
RR[26] 1.458 0.3183 0.003325 0.9071 1.432 2.148 1 10170
RR[27] 1.317 0.3253 0.003162 0.7729 1.286 2.05 1 10170
RR[28] 1.976 0.4156 0.00403 1.265 1.943 2.882 1 10170
RR[29] 1.32 0.2875 0.002689 0.8274 1.297 1.954 1 10170
RR[30] 1.204 0.2599 0.00268 0.76 1.179 1.775 1 10170
RR[31] 1.096 0.2264 0.002151 0.6982 1.078 1.587 1 10170
Autocorrelation function
RR[1] chains 1:3
RR[2] chains 1:3
RR[3] chains 1:3
RR[4] chains 1:3
RR[5] chains 1:3
RR[6] chains 1:3
RR[7] chains 1:3
RR[8] chains 1:3
lag


Dynamic trace
frac.spatial chains 3:1
iteration
Gelman Rubin statistic
frac.spatial chains 1:3
start-iteration
Autocorrelation function
frac.spatial chains 1:3
lag
Node statistics
node mean sd MC error 2.5% median 97.5% start sample
frac.spatial 0.9263 0.08667 0.002873 0.685 0.9619 0.9978 1 18000


Dynamic trace
s2.marginal chains 3:1
iteration
Gelman Rubin statistic
s2.marginal chains 1:3
start-iteration
Autocorrelation function
s2.marginal chains 1:3
lag
Node statistics
node mean sd MC error 2.5% median 97.5% start sample
s2.marginal 0.2781 0.05566 8.76E-4 0.1733 0.2764 0.393 1 18000

4. Selecction of the best model

As shown in the previouses parts all the chains for each of the parametres converge. Therefore all three models are able to fit well the data, however we should choose one of them, the one that fits the data the best. For this we are going to check the deviance information criterion (DIC) of each model.The DIC is a measure of goodness of fit which ponderate it with the complexity of the model. Therefore the best fit with the simplest model would be the optimum and lowest DIC.

Dbar = post.mean of -2logL; Dhat = -2LogL at post.mean of stochastic nodes

	Dbar	Dhat	pD	DIC
O	615.852	534.999	80.853	696.705
total	615.852	534.999	80.853	696.705

Dbar = post.mean of -2logL; Dhat = -2LogL at post.mean of stochastic nodes

	Dbar	Dhat	pD	DIC
O	618.104	561.525	56.579	674.683
total	618.104	561.525	56.579	674.683

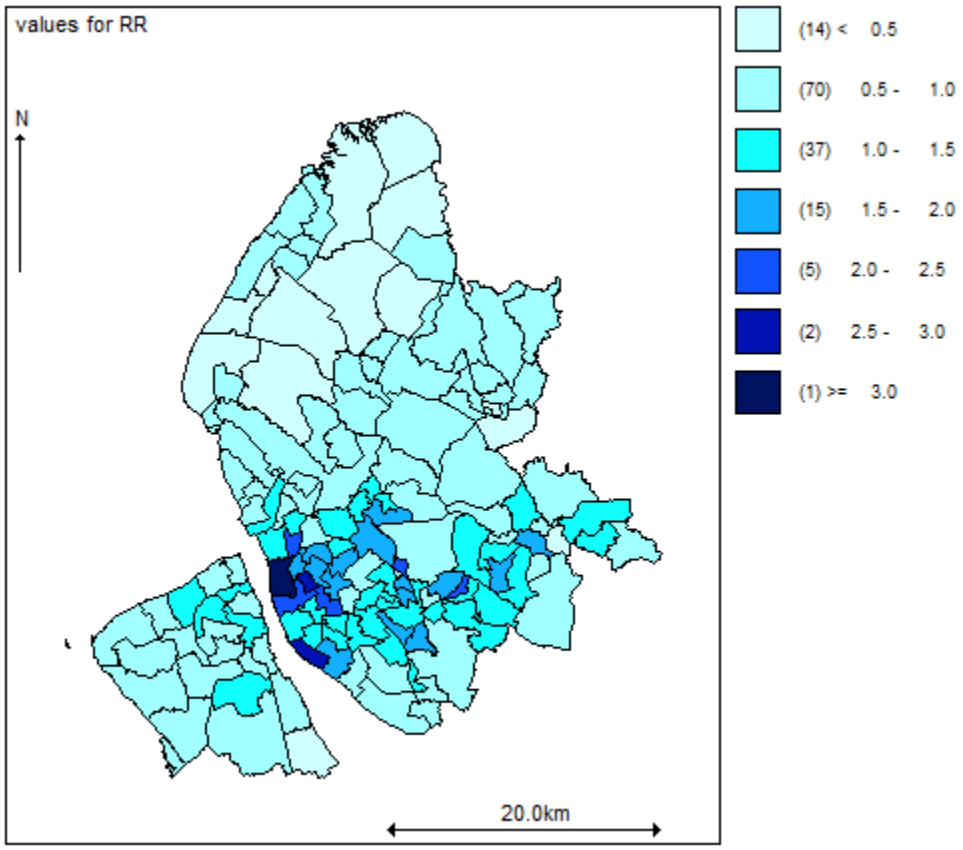
Dbar = post.mean of -2logL; Dhat = -2LogL at post.mean of stochastic nodes

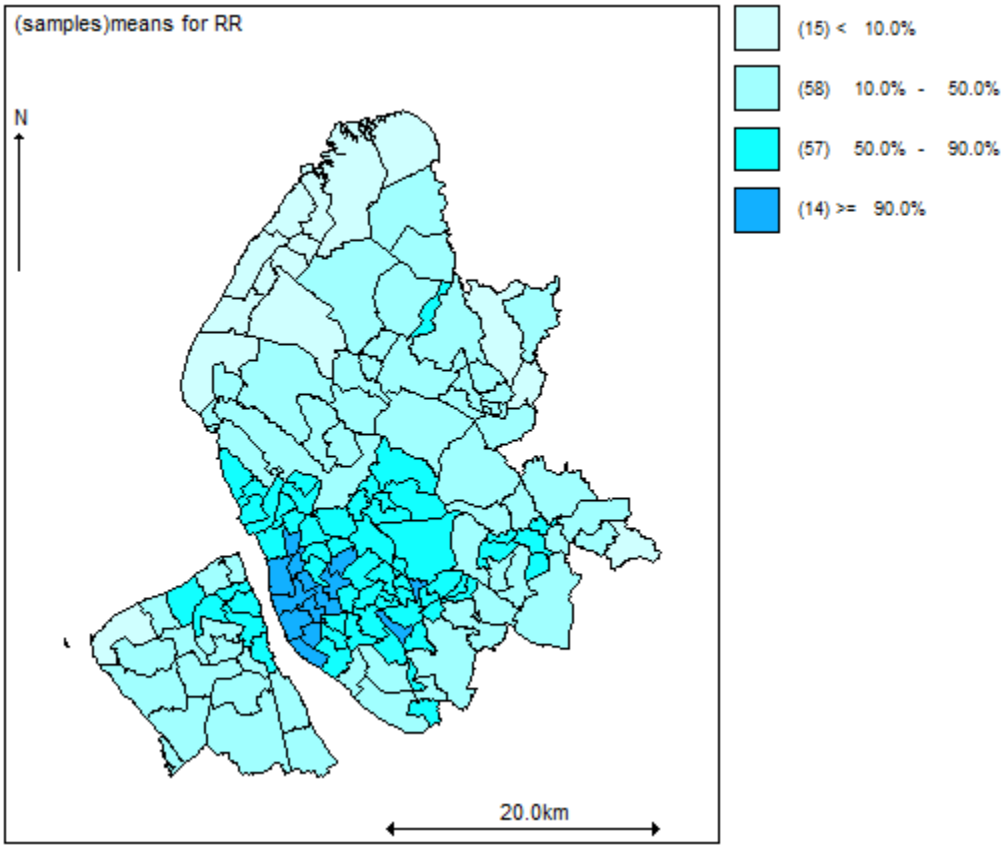
	Dbar	Dhat	pD	DIC
O	614.648	556.125	58.523	673.171
total	614.648	556.125	58.523	673.171

After comparing the scores, we see that the model 3, the convolution model is the one with the lowest DIC, hence the best model of the three, although the spatial one is very close to it so it can be a good alternative.

5. Maps

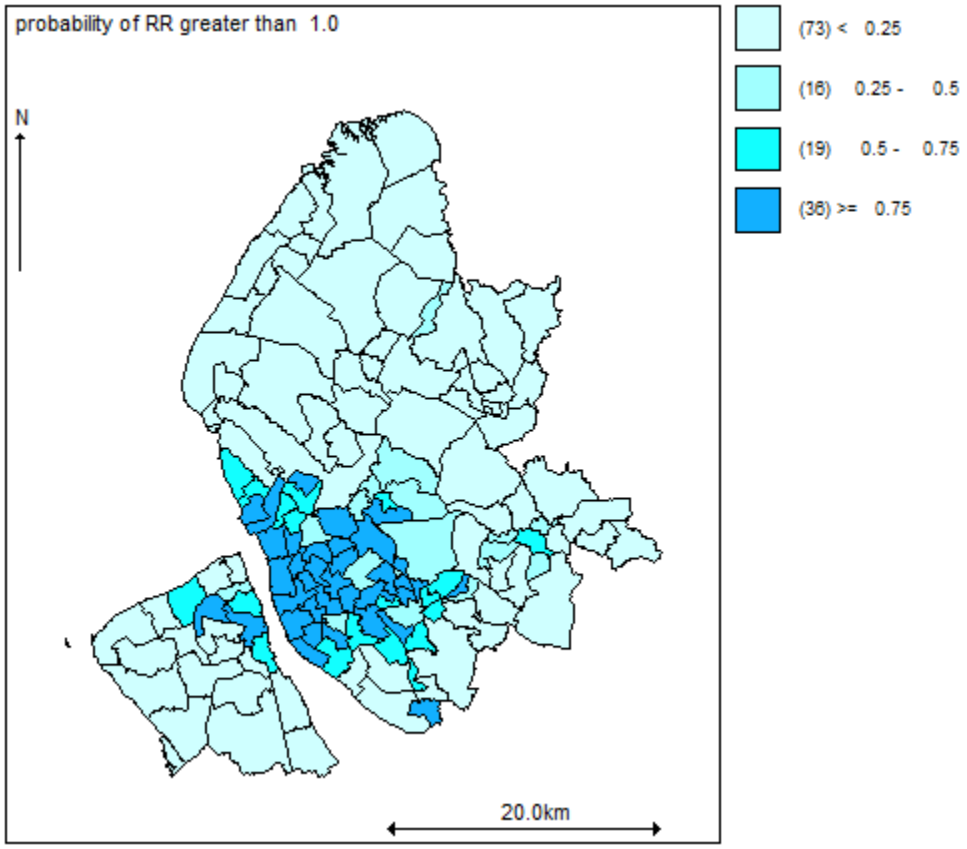
5.1. Observed SMR and Relative risk

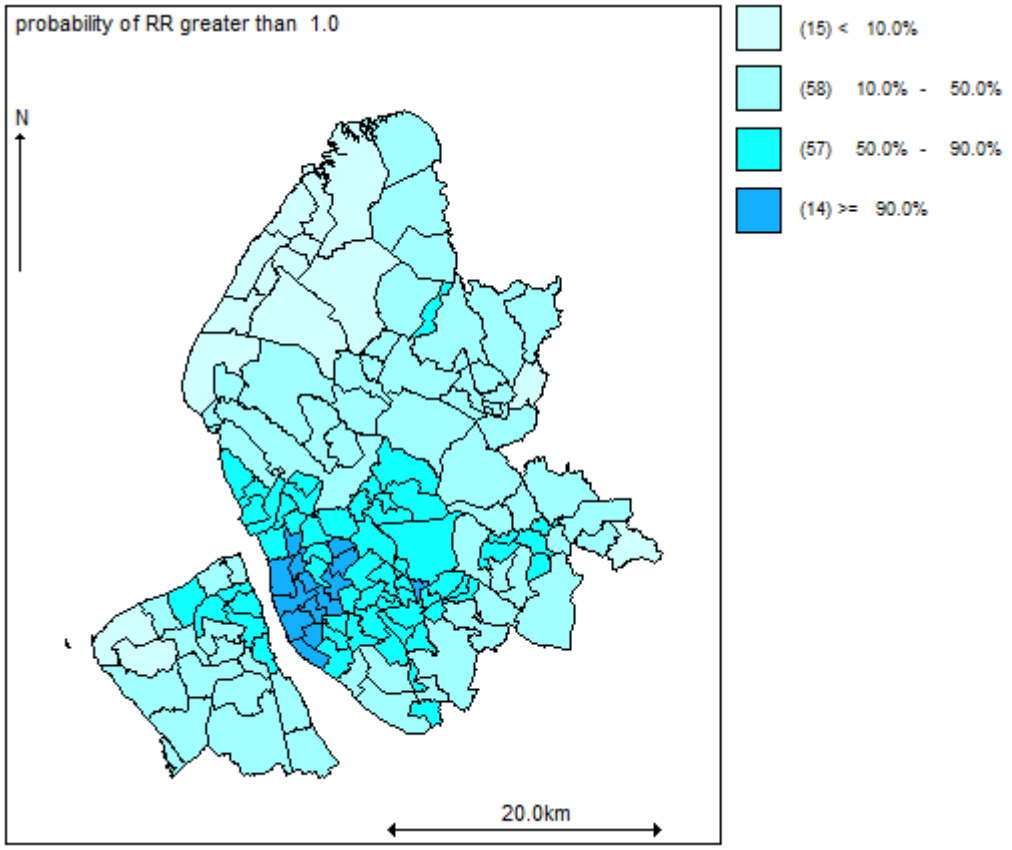




As we observed in the map of the begining, we can see that there is a cluster with higher risk in West-Lancashire.

5.2. Posterior probability with certain relative risk

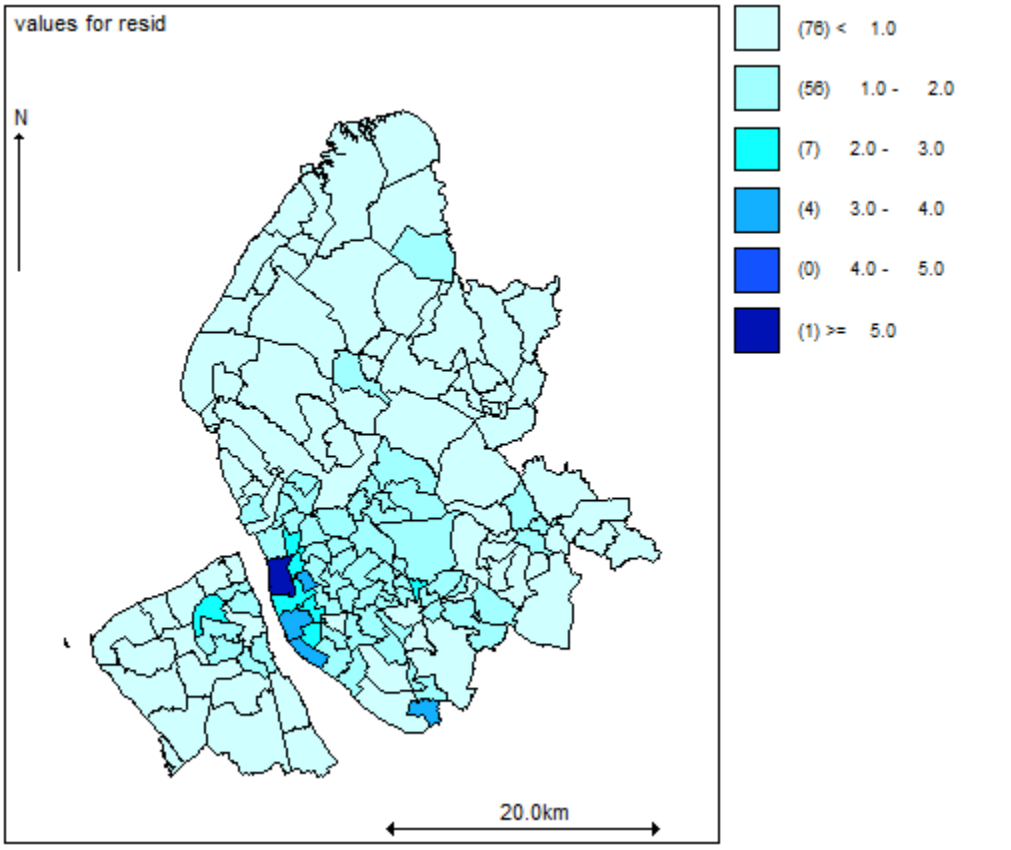




This map shows the districts where the relative risk is greater than 1. And as we can see, as we could see form the original map, the most "centric" region, is the one with greater risk. Therefore following our model the relationship between the cases obverved and the expected are quite congruent.

5.3. Random effects

The random effects, also called residuals are the part of the data that can not be explained by the model. A perfect model (one that includes all the variables and relationships would have no random effects). But since our model includes just some parameters and relationships between them there is a part that can be explained.



In this case we see that the random effects are bigger in the centre zone cluster. Being a districts with considerable RE. However besides this almost all of the map has low residuals meaning that the model is quite good specially having in mind that only 5 out of 144 districts have some consideralbe random effects.