

# SISMID 2021: R Notes Clustering and Cluster Detection

Jon Wakefield  
University of Washington

2021-07-09

## North Carolina SIDS Data

The `nc.sids` data frame has 100 rows and 21 columns and can be found in the `spdep` library.

It contains data given in Cressie (1991, pp. 386-9), Cressie and Read (1985) and Cressie and Chan (1989) on sudden infant deaths in North Carolina for 1974–78 and 1979–84.

The data set also contains the neighbour list given by Cressie and Chan (1989) omitting self-neighbours (`ncCC89.nb`), and the neighbour list given by Cressie and Read (1985) for contiguities (`ncCR85.nb`).

Data are available on the numbers of cases and on the number of births, both dichotomized by a binary indicator of race.

The data are ordered by county ID number, not alphabetically as in the source tables.

# North Carolina SIDS Data

The code below plots the county boundaries along with the observed SMRs for 1974.

The expected numbers are based on internal standardization with a single stratum. So the single reference probability is the incidence of SIDS in 1974.

```
library(maptools)
library(spdep)
nc.sids <- readShapeSpatial(system.file("shapes/sids.shp",
  package = "maptools")[1], IDvar = "FIPSNO",
  proj4string = CRS("+proj=longlat +ellps=clrk66"))
nc.sids2 <- nc.sids # Create a copy, to add to
Y <- nc.sids$SID74
E <- nc.sids$BIR74 * sum(Y)/sum(nc.sids$BIR74)
nc.sids2$SMR74 <- Y/E
nc.sids2$EXP74 <- E
brks <- seq(0, 5, 1)
rm(nc.sids) # We load another version of this later, so tidy up here
```

## SMR Plot

The map of the SMRs shows a number of counties with high relative risks (the risk relative to the state wide risk).

```
spplot(nc.sids2, "SMR74", at = brks,  
       col.regions = grey.colors(5, start = 0.9,  
       end = 0.1))
```

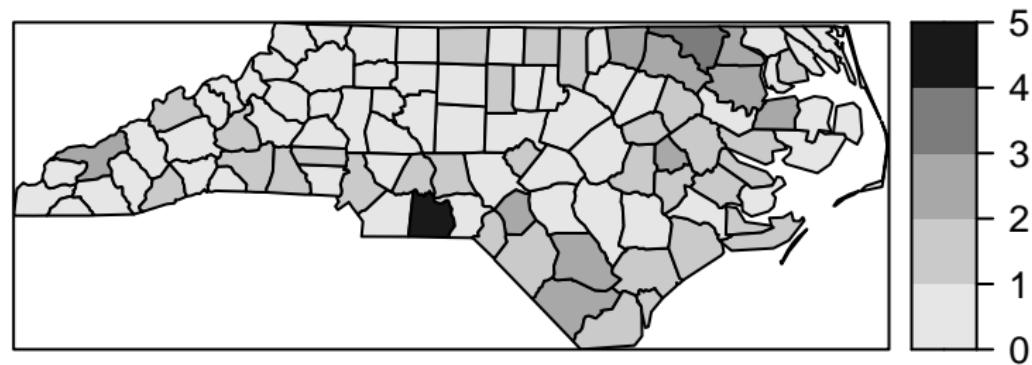


Figure 1: Map of SMRs for SIDS in 1974 in North Carolina

# Disease Mapping

We first fit a non-spatial random effects model:

$$\begin{aligned} Y_i | \beta_0, e_i &\sim_{iid} \text{Poisson}(E_i e^{\beta_0 + e_i}), \\ e_i | \sigma_e^2 &\sim_{iid} N(0, \sigma_e^2) \end{aligned}$$

with the default priors on  $\beta_0$  and  $\sigma_e^2$ .

```
library(INLA)
nc.sids2$ID <- 1:100
m0 <- inla(SID74 ~ f(ID, model = "iid"),
            family = "poisson", E = EXP74, data = as.data.frame(nc.sids2),
            control.predictor = list(compute = TRUE))
```

The `control.predictor` argument indicates we want fitted values.

# Disease Mapping

Examine the first few “fitted values”, summaries of the posterior distribution of  $\exp(\beta_0 + e_i)$ ,  $i = 1, \dots, n$ .

```
head(m0$summary.fitted.values)
##               mean        sd 0.025quant 0.5quant 0.975quant
## fitted.Predictor.001 1.2471225 0.2903238 0.7558722 1.2204607 1.890457
## fitted.Predictor.002 0.7716701 0.2691840 0.3463012 0.7378817 1.393246
## fitted.Predictor.003 0.9172848 0.3458915 0.3983970 0.8654076 1.740145
## fitted.Predictor.004 2.6881922 0.7827812 1.4514039 2.5896503 4.484594
## fitted.Predictor.005 0.9050712 0.3143757 0.4171151 0.8625052 1.640895
## fitted.Predictor.006 0.8579050 0.3125256 0.3778924 0.8143494 1.592539
##               mode
## fitted.Predictor.001 1.1688218
## fitted.Predictor.002 0.6768216
## fitted.Predictor.003 0.7775835
## fitted.Predictor.004 2.3903259
## fitted.Predictor.005 0.7875240
## fitted.Predictor.006 0.7384506
```

# Disease Mapping

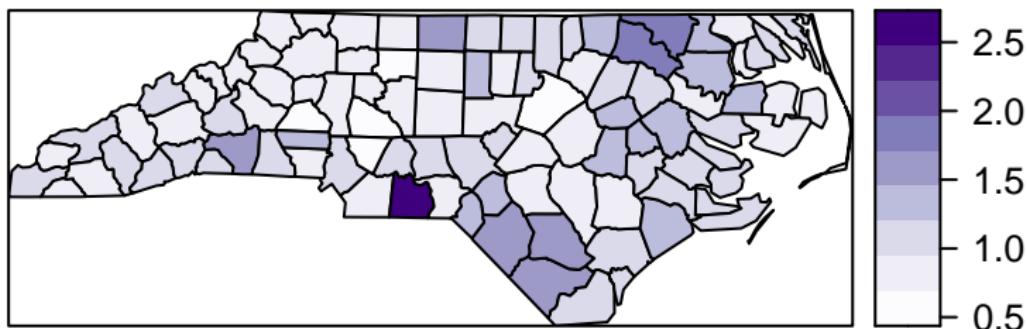
Create two interesting inferential summaries:

- the posterior median of the relative risk
- the posterior probability that the relative risk is greater than an epidemiologically significant value. We look at 1.5 and 2.5.

```
nc.sids2$m0RRmedian <- m0$summary.fitted.values[, 4]
nc.sids2$m0prob1.5 <- 1 - sapply(m0$marginals.fitted.values,
  function(x) inla.pmarginal(1.5, x))
nc.sids2$m0prob2.5 <- 1 - sapply(m0$marginals.fitted.values,
  function(x) inla.pmarginal(2.5, x))
```

# Disease Mapping

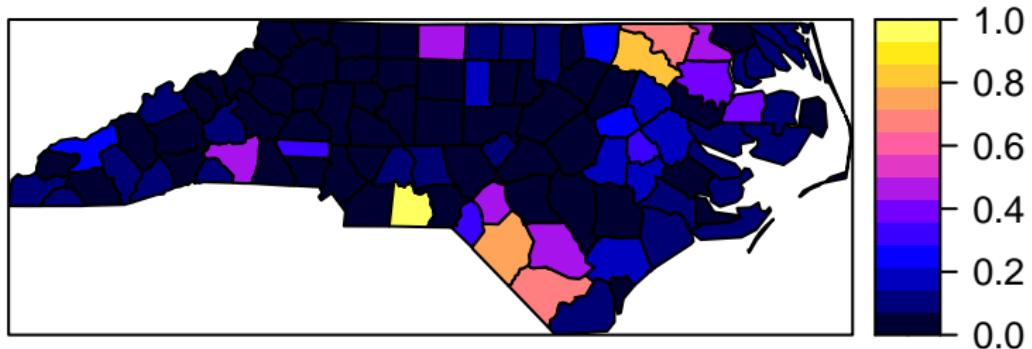
```
# Display posterior means of  
# relative risks  
library(RColorBrewer)  
spplot(nc.sids2, "mORRpmedian", col.regions = brewer.pal(9,  
    "Purples"), cuts = 8)
```



A number of counties have high median values.

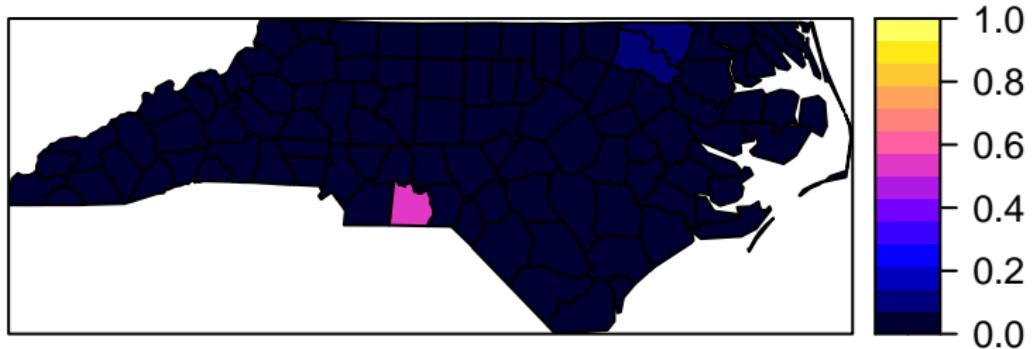
# Disease Mapping

```
spplot(nc.sids2, "m0prob1.5", at = seq(0,  
1, length.out = 15))
```



# Disease Mapping

```
spplot(nc.sids2, "m0prob2.5", at = seq(0,  
1, length.out = 15))
```



# Disease Mapping with IID and ICAR random effects

We now fit a model with non-spatial and ICAR spatial random effects.

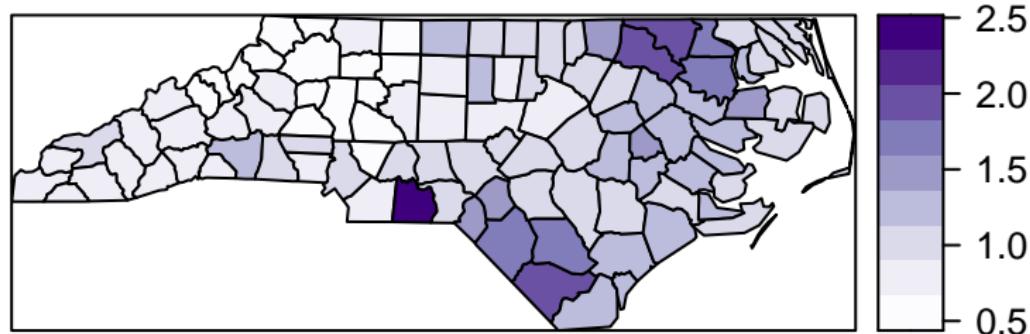
```
nc.sids2$ID2 <- 1:100
download.file("http://faculty.washington.edu/jonno/SISMIDmaterial/NC.graph",
  destfile = "R-examples/NC.graph")
formula <- SID74 ~ 1 + f(ID, model = "bym2", graph = "R-examples/NC.graph",
  scale.model = T, constr = T, hyper = list(phi = list(prior = "pc",
    param = c(0.5, 0.5), initial = 1), prec = list(prior = "pc.prec",
    param = c(0.5/0.31, 0.01), initial = 5)))
m1 <- inla(formula, family = "poisson", data = as.data.frame(nc.sids2),
  E = EXP74, control.predictor = list(compute = TRUE),
  control.compute = list(config = TRUE))
```

# Disease Mapping with IID and ICAR random effects

```
# Define summary quantities of interest as with iid model
nc.sids2$m1RRmedian <- m1$summary.fitted.values[, 4]
nc.sids2$m1prob1.5 <- 1 - sapply(m1$marginals.fitted.values,
  function(x) inla.pmarginal(1.5, x))
nc.sids2$m1prob2.5 <- 1 - sapply(m1$marginals.fitted.values,
  function(x) inla.pmarginal(2.5, x))
```

## Display posterior means of relative risks

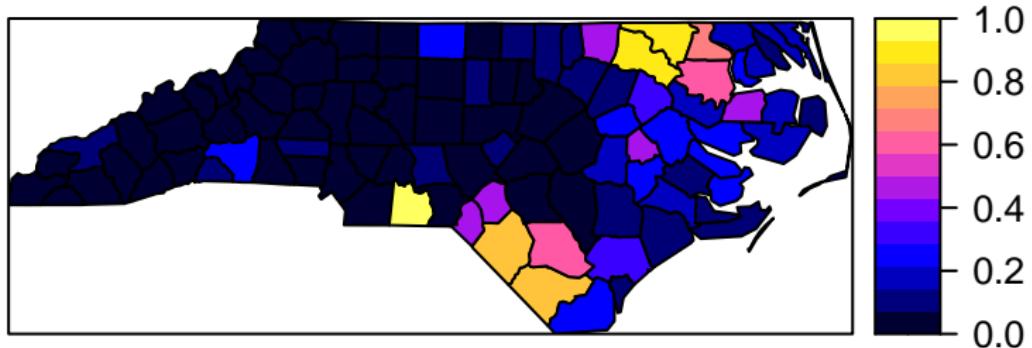
```
spplot(nc.sids2, "m1RRmedian", col.regions = brewer.pal(9,  
"Purples"), cuts = 8)
```



A number of counties have high mean values.

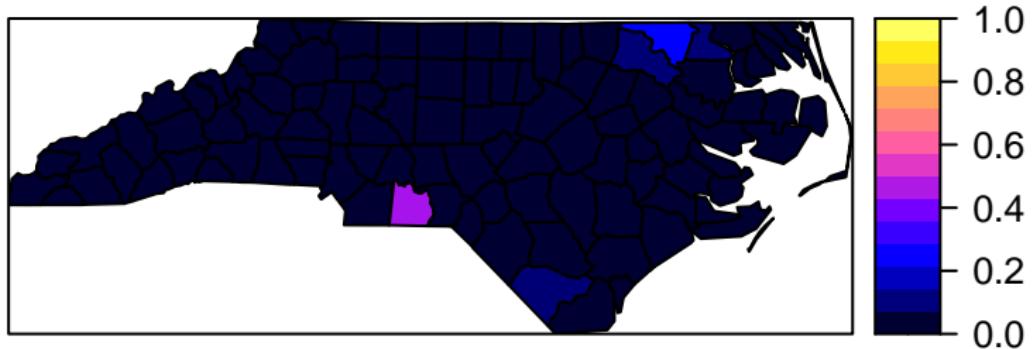
# Disease Mapping

```
spplot(nc.sids2, "m1prob1.5", at = seq(0,  
1, length.out = 15))
```



# Disease Mapping

```
spplot(nc.sids2, "m1prob2.5", at = seq(0,  
1, length.out = 15))
```



# Disease Mapping with IID and ICAR random effects

Summarize the IID+ICAR model

```
m1$summary.fixed[1:5]
##                   mean          sd 0.025quant 0.5quant 0.975quant
## (Intercept) -0.05047494 0.0581825 -0.167713 -0.04947954 0.0611524
m1$summary.hyperpar[1:5]
##                   mean          sd 0.025quant 0.5quant 0.975quant
## Precision for ID 5.5024192 1.9348593 2.5563478 5.228198 10.0908458
## Phi for ID      0.6885186 0.2248202 0.1932743 0.733535 0.9818016
```

## Lognormal spatial model with covariates

Now we provide maps of the non-spatial and spatial random effects.

Estimates of residual relative risk (posterior medians), of the non-spatial  $e^{e_i}$  and the spatial contributions  $e^{S_i}$ .

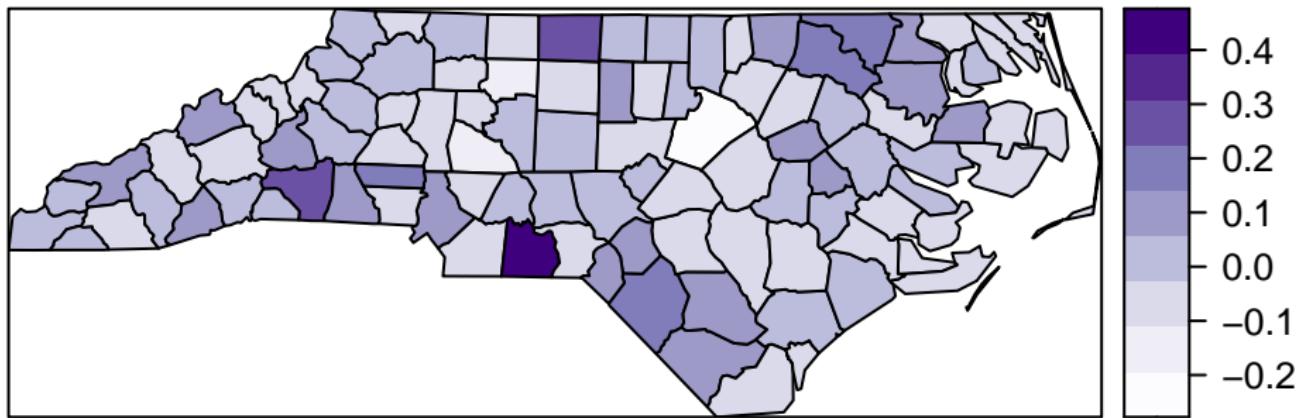
The BYM2 formulation for the random effect is  $b_i = S_i + e_i$  where  $S_i$  is spatial and  $e_i$  is IID. INLA stores  $b_i$  (the first 56 rows) and  $S_i$  (the next 56 rows) and so we find the non-spatial via  $e_i = b_i - S_i$ .

Note the differences in the scales: the spatial random effects dominate here.

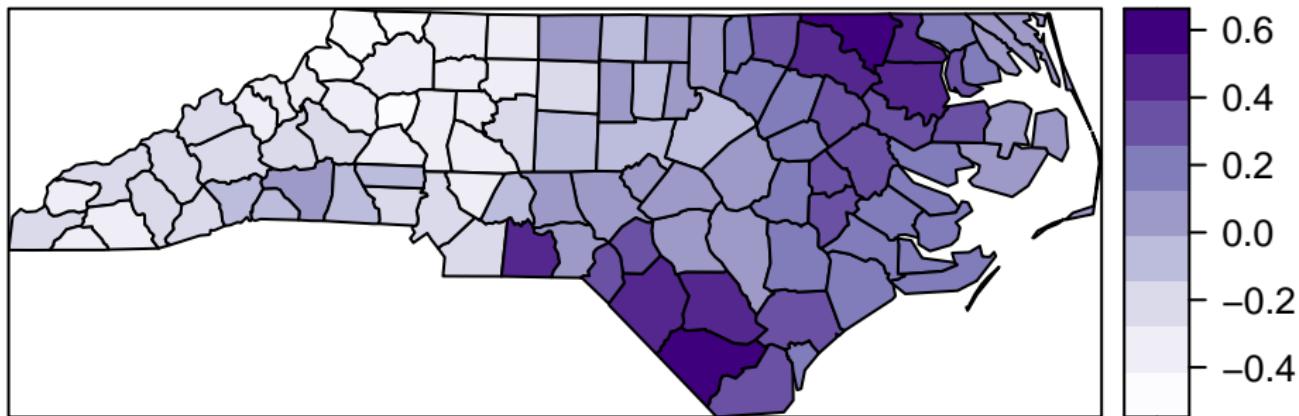
## Extracting the random effects

```
m1samp <- inla.posterior.sample(n = 1000, m1)
m1samp_mat <- matrix(0, nrow = 1000, ncol = 2)
for (i in 1:1000) {
    m1samp_mat[i, ] <- m1samp[[i]]$hyperpar[1:2]
}
m1scale_region <- mean(sqrt(m1samp_mat[, 2])/sqrt(m1samp_mat[, 1]))
# obtain RE estimates
N <- 100
m1struct <- m1$summary.random[[1]]$mean[(N + 1):(N *
    2)]
m1combined <- m1$summary.random[[1]]$mean[1:N]
m1struct <- m1struct * m1scale_region
m1iid <- m1combined - m1struct
#
nc.sids2$REsnonspat <- m1iid
nc.sids2$REsspat <- m1struct
```

## Lognormal spatial model with covariates: non-spatial random effects



# Lognormal spatial model with covariates: spatial random effects



## Disease Mapping with IID and ICAR random effects

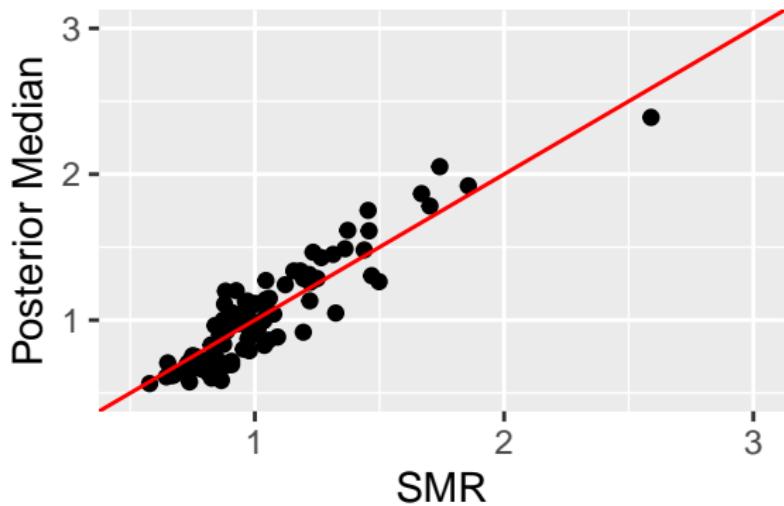
Both summaries show differences with the iid only model, with changes in the obvious direction.

The spatial smoothing model gives a larger collection in the north-east, for example, and a single area in the west is not highlighted.

The BYM2 model strongly suggests that there is considerable spatial variation here ( $\phi$ ).

# Disease Mapping: Comparison of posterior median RRs

```
library(ggplot2)
t0 <- m0$summary.fitted.values[, 4]
t1 <- m1$summary.fitted.values[, 4]
ggplot(data.frame(t0, t1), aes(x = t0, y = t1)) + geom_point() +
  labs(y = "Posterior Median", x = "SMR") + geom_abline(intercept = 0,
  slope = 1, color = "red") + xlim(0.5, 3) + ylim(0.5, 3)
```



## Disease mapping with IID and ICAR random effects

We now examine the variances of the spatial and non-spatial random effects.

Recall that the ICAR model variance has a conditional interpretation.

To obtain a rough estimate of the marginal variance we obtain the posterior median of the  $S_i$ 's and evaluate their variance.

From the output below, we conclude that the spatial random effects dominate for the SIDS data so that we conclude there is clustering of cases in neighboring areas.

## Clustering via Moran's I

We evaluate Moran's test for spatial autocorrelation using the "W" style weight function: this standardizes the weights so that for each area the weights sum to 1. Also define the "B" style for later.

To obtain a variable with approximately constant variance we form residuals from an intercept only model.

```
library(spdep)
# Note the nc.sids loaded from the data() command
# is in a different order to that obtained from the
# shapefile
data(nc.sids)
col.W <- nb2listw(ncCR85.nb, style = "W", zero.policy = TRUE)
col.B <- nb2listw(ncCR85.nb, style = "B", zero.policy = TRUE)
rm(nc.sids)
quasipmod <- glm(SID74 ~ 1, offset = log(EXP74), data = nc.sids2,
                  family = quasipoisson())
sidsres <- residuals(quasipmod, type = "pearson")
```

# Clustering via Moran's I

```
moran.test(sidsres, col.W)
##
##  Moran I test under randomisation
##
## data: sidsres
## weights: col.W
##
## Moran I statistic standard deviate = 2.4351, p-value = 0.007444
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic      Expectation      Variance
##          0.147531140     -0.010101010     0.004190361
```

This analysis suggests significant clustering.

## Clustering via Moran's $I$

Moran's test may suggest spatial autocorrelation if there exists a non-constant mean function.

Hence, we should endeavor to remove the large-scale trends.

Below we fit a model with Eastings and Northings (of the County seat) as covariates – both show some at least some association.

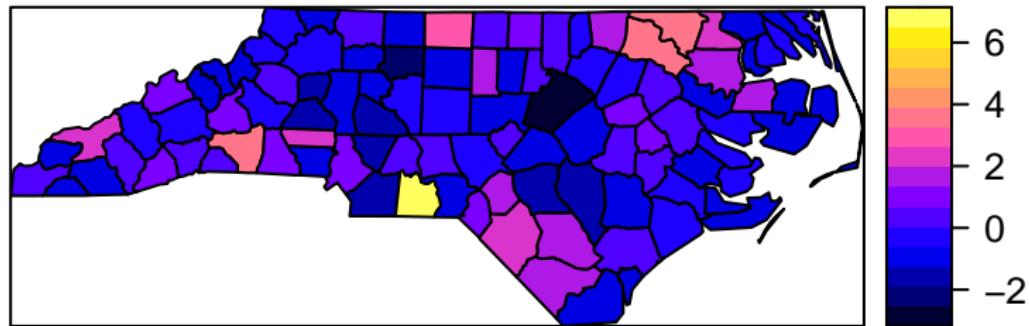
# Clustering via Moran's I

```
data(nc.sids)
nc.sids2 <- merge(nc.sids2, nc.sids[, c("CNTY.ID", "east", "north")], by.x = "CNTY_ID", by.y = "CNTY.ID")
rm(nc.sids)
quasipmod2 <- glm(SID74 ~ east + north, offset = log(EXP74), data = nc.sids2, family = quasipoisson())
summary(quasipmod2)
##
## Call:
## glm(formula = SID74 ~ east + north, family = quasipoisson(),
##      data = nc.sids2, offset = log(EXP74))
##
## Deviance Residuals:
##       Min      1Q   Median      3Q     Max
## -2.7961 -1.0249 -0.3475  0.6043  4.7261
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.2465437  0.2680159 -0.920  0.35992
## east         0.0020105  0.0006469  3.108  0.00247 **
## north        -0.0028032  0.0014545 -1.927  0.05687 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Dispersion parameter for quasipoisson family taken to be 2.039456
##
## Null deviance: 203.34 on 99 degrees of freedom
## Residual deviance: 171.80 on 97 degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 4
```

# Clustering via Moran's $I$

We map the residuals to get a visual on the clustering.

```
sidsres2 <- residuals(quasipmod2, type = "pearson")
nc.sids2$res <- sidsres2
par(mar = c(0.1, 0.1, 0.1, 0.1))
spplot(nc.sids2, "res")
```



## Clustering via Moran's I

The significance of the Moran statistic is reduced, though still significant if judged by conventional levels.

```
moran.test(sidsres2, col.W)
##
##  Moran I test under randomisation
##
## data: sidsres2
## weights: col.W
##
## Moran I statistic standard deviate = 2.1328, p-value = 0.01647
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic      Expectation      Variance
##          0.127428361     -0.010101010     0.004157993
```

## Neighborhood options

There are various coding schemes for the weights.

$B$  has 0/1 corresponding to non-neighbor/neighbor – this means areas with many neighbors are more influential.

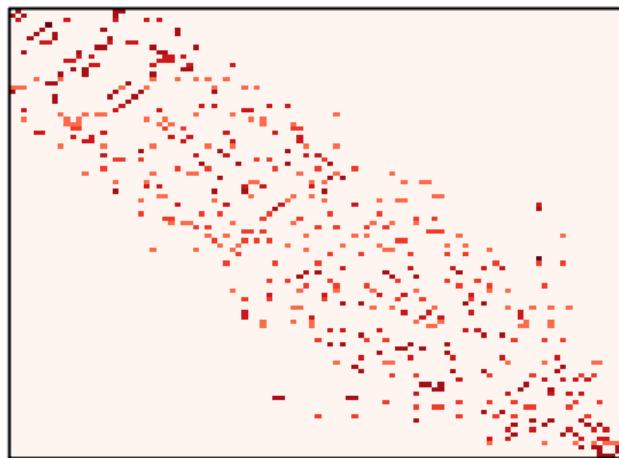
$W$  has rows standardized by the number of neighbors so that the sum for each row (area) is unity.

Weights can be more complex, depending on inverse distance, for example.  
See Bivand et al. (2013, Section 9.2).

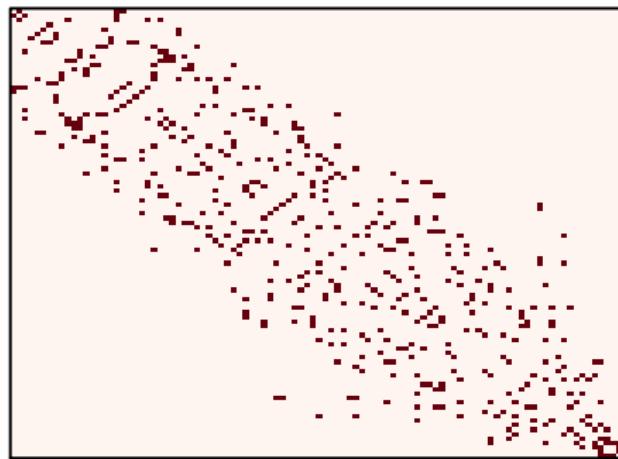
# Neighborhood options

```
library(RColorBrewer)
pal <- brewer.pal(9, "Reds")
z <- t(listw2mat(col.W))
brks <- c(0, 0.1, 0.143, 0.167, 0.2, 0.5, 1)
nbr3 <- length(brks) - 3
image(1:100, 1:100, z[, ncol(z):1], breaks = brks,
      col = pal[c(1, (9 - nbr3):9)], main = "W style",
      axes = FALSE)
box()
z <- t(listw2mat(col.B))
brks <- c(0, 0.1, 0.143, 0.167, 0.2, 0.5, 1)
nbr3 <- length(brks) - 3
image(1:100, 1:100, z[, ncol(z):1], breaks = brks,
      col = pal[c(1, (9 - nbr3):9)], main = "B style",
      axes = FALSE)
box()
```

## W style



## B style



Note the asymmetry in the "W" weights option.

## Moran's I with a different neighborhood structure

We now use Moran's statistic on the detrended residuals, but with the binary "B" weight option. This option has unstandardized weights.

The conclusion, evidence of spatial autocorrelation, is the same as with the standardized weights option.

```
moran.test(sidsres2, col.B)
##
## Moran I test under randomisation
##
## data: sidsres2
## weights: col.B
##
## Moran I statistic standard deviate = 2.2357, p-value = 0.01269
## alternative hypothesis: greater
## sample estimates:
## Moran I statistic      Expectation      Variance
##          0.125344196     -0.010101010     0.003670354
```

## Clustering via Geary's C

We now use Geary's statistic on the detrended residuals, and come to the same conclusion

```
geary.test(sidsres2, col.W)
##
##  Geary C test under randomisation
##
## data: sidsres2
## weights: col.W
##
## Geary C statistic standard deviate = 2.3479, p-value = 0.009439
## alternative hypothesis: Expectation greater than statistic
## sample estimates:
## Geary C statistic      Expectation      Variance
##          0.8195420        1.0000000       0.0059072
```

## North Carolina SIDS Data: Clustering Conclusions

The disease mapping model shows that almost all of the residual variation is spatial.

Both of the Moran's  $I$  and Geary's  $c$  methods suggest that there is evidence of clustering in these data.

So all methods in agreement!

## Cluster detection

We now turn to cluster detection: detecting areas or contiguous collections of areas that appear to be at elevated risk.

We concentrate on the SatScan method but for illustration show the methods of Openshaw and Besag and Newell in action.

The results from these two methods are difficult to interpret formally (Openshaw's in particular) because of the multiple testing problem.

# Cluster detection

```
library(maptools)
library(maps)
library(ggplot2)
library(sp)
nc.sids <- readShapeSpatial(system.file("shapes/sids.shp",
    package = "maptools")[1], IDvar = "FIPSNO",
    proj4string = CRS("+proj=longlat +ellps=clrk66"))
population <- nc.sids$BIR74
cases <- nc.sids$SID74
```

# Cluster detection: Besag and Newell

We need to form a matrix containing the centroids.

```
getLabelPoint <- function(county) {  
  Polygon(county[c("long", "lat")])@labpt  
}  
df <- map_data("county", "north carolina") # NC region county data  
centNC <- by(df, df$subregion, getLabelPoint) # Returns list  
centNC <- do.call("rbind.data.frame", centNC) # Convert to Data Frame  
names(centNC) <- c("long", "lat") # Appropriate Header  
centroids <- matrix(0, nrow = length(cases), ncol = 2)  
for (i in 1:length(cases)) {  
  centroids[i, ] <- c(centNC$lat[i], centNC$long[i])  
}  
colnames(centroids) <- c("x", "y")  
rownames(centroids) <- 1:length(cases)  
geo <- centroids
```

## Cluster detection: SatScan method

We set 20% as the upper bound on the proportion of the population to be contained in any one potential cluster.

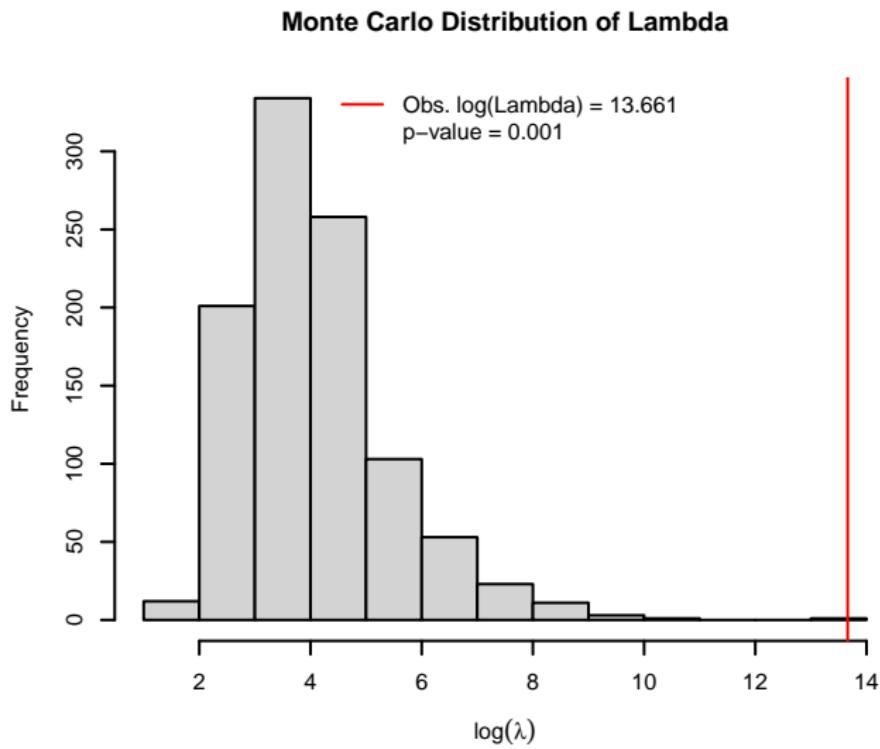
```
# SpatialEpi implementation of SatScan
library(SpatialEpi)
pop.upper.bound <- 0.2
n.simulations <- 999
alpha.level <- 0.05
Kpoisson <- kulldorff(geo, cases, population, expected.cases = NULL,
                       pop.upper.bound, n.simulations, alpha.level, plot = T)
names(Kpoisson)
## [1] "most.likely.cluster" "secondary.clusters"   "type"
## [4] "log.lkhd"           "simulated.log.lkhd"
names(Kpoisson$most.likely.cluster)
## [1] "location.IDs.included" "population"          "number.of.cases"
## [4] "expected.cases"        "SMR"                "log.likelihood.ratio"
## [7] "monte.carlo.rank"      "p.value"
Kcluster <- Kpoisson$most.likely.cluster$location.IDs.included
```

## Cluster detection: SatScan method

We now turn to SatScan and set 20% as the upper bound on the proportion of the population to be contained in any one potential cluster.

```
head(Kpoisson$most.likely.cluster)
## $location.IDs.included
## [1] 74 40 58 33 54 7 25 98 52 8 96 64 42 69 66 94 31 46 67
##
## $population
## [1] 60452
##
## $number.of.cases
## [1] 177
##
## $expected.cases
## [1] 122.2004
##
## $SMR
## [1] 1.448441
##
## $log.likelihood.ratio
## [1] 13.66057
```

# Cluster detection: SatScan method



## Cluster detection: Set up polygons

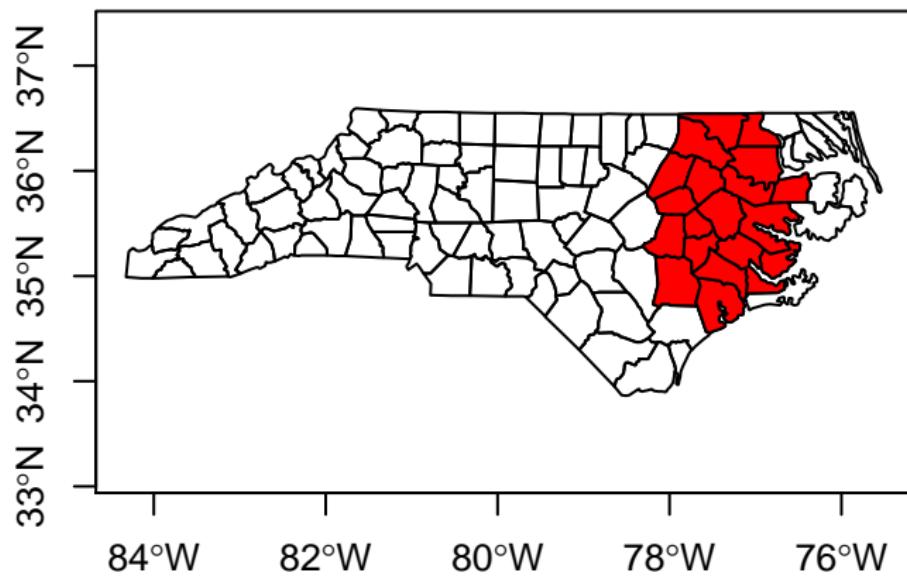
```
NCTemp <- map("county", "north carolina",
  fill = TRUE, plot = FALSE)
NCIDs <- substr(NCTemp$names, 1 + nchar("north carolina,"),
  nchar(NCTemp$names))
NC <- map2SpatialPolygons(NCTemp, IDs = NCIDs,
  proj4string = CRS("+proj=longlat"))
# Fix currituck county which is 3 islands
index <- match(c("currituck:knotts", "currituck:main",
  "currituck:spit"), NCIDs)
currituck <- list()
for (i in c(27:29)) currituck <- c(currituck,
  list(Polygon(NC@polygons[[i]]@Polygons[[1]]@coords)))
currituck <- Polygons(currituck, ID = "currituck")
```

## Cluster detection: make new spatial polygons object

```
NC.new <- NC@polygons[1:(index[1] - 1)]
NC.new <- c(NC.new, currituck)
NC.new <- c(NC.new, NC@polygons[(index[3] + 1):length(NC@polygons)])
NC.new <- SpatialPolygons(NC.new, proj4string = CRS("+proj=longlat"))
NCIDs <- c(NCIDs[1:(index[1] - 1)], "currituck", NCIDs[(index[3] +
    1):length(NC@polygons)])
NC <- NC.new
```

## SatScan method: Most likely cluster

```
op <- par(cex = 0.8)
plot(NC.new, axes = TRUE)
plot(NC.new[Kcluster], add = TRUE, col = "red")
```



## SatScan method: Secondary clusters

```
K2cluster <- Kpoisson$secondary.clusters[[1]]$location.IDs.included  
K2cluster  
## [1] 83 47 77 78 63 4  
plot(NC.new, axes = TRUE)  
plot(NC.new[K2cluster], add = TRUE, col = "red")
```

## SatScan method: Secondary clusters

