Disease mapping with R

V. Gómez-Rubio

Departamento de Matemáticas
Escuela de Ingenieros Industriales de Albacete
U. de Castilla-La Mancha

Imperial College London, 16 May 2014

based on work by Roger S. Bivand, Edzer Pebesma and H. Rue

Disease mapping: Introduction

Motivation

- The number of cases of a disease at some aggregation level it is often used for disease mapping
- Cases usually represent mortality by a disease
- Hence, our response variable has integer values and a Poisson model seems reasonable
- It may be possible that other covariates are available as well
- The aim is to determine any existing relation between the number of cases and some risk factors

Statistical Model

$$O_i \sim Po(\mu_i);\ \mu_i = E_i\theta_i;\ \log(\mu_i) = \log(E_i) + \log(\theta_i)$$

$$\log(\theta_i) = \alpha + \beta x_i$$

Relative Risk

Description

- In this type of study incidence or risk are measured using a reference population (which is used to compute a global rate r)
- Hence, all the results are *relative* to that rate and reference population
- For this reason it is called *relative risk*, because it depends on the reference population
- A value of 1 means that risk is the same as in the reference population
- Values higher than 1 mean that the risk is higher than in the reference population
- For this reason, we are interesting in detecting those regions with significant $\theta_i > 1$

North Carolina SIDS data

Description

- This data sets records cases of Sudden Infant Death Syndrome (SIDS) in North CArolian (U.S.A.)
- Cases grouped in two periods: 1971-1974 and 1975-1979
- Risk population is the total number of births in the study period
- As covariate, the proportion of non-white births are available
- The expected number of cases is computed as

$$E_i = rN_i; r = \frac{\sum_i O_i}{\sum_i N_i}$$

- $\log(E_i)$ is introduces as an fixed *offset* in the model
- An estimate of the risk is the Standardised Mortality Rate (SMR) O_i/E_i

Model fitting

Variables of interest

- SID74: Number of cases in 1971-1974
- BIR74: Number of births 1971-1974
- NWBIR74: Number of non-white births in 1971-1974

```
> library(spdep)
> data(nc.sids)
> nc.sids$NWPROP74<-nc.sids$NWBIR74/nc.sids$BIR74
> nc.sids$NWPROP79<-nc.sids$NWBIR79/nc.sids$BIR79
> r74<-sum(nc.sids$SID74)/sum(nc.sids$BIR74)
> nc.sids$EXP74<-r74*nc.sids$BIR74
> nc.sids$SMR74<-nc.sids$SID74/nc.sids$EXP74
> r79<-sum(nc.sids$SID79)/sum(nc.sids$BIR79)
> nc.sids$EXP79<-r79*nc.sids$BIR79
> nc.sids$SMR79<-nc.sids$SID79/nc.sids$EXP79
> ncglm74<-glm(SID74~offset(log(EXP74))+NWPROP74, data=nc.sids, family="poisson")
> ncglm79<-glm(SID79~offset(log(EXP79))+NWPROP79, data=nc.sids, family="poisson")
>
```

Model fitting: 1971-1974

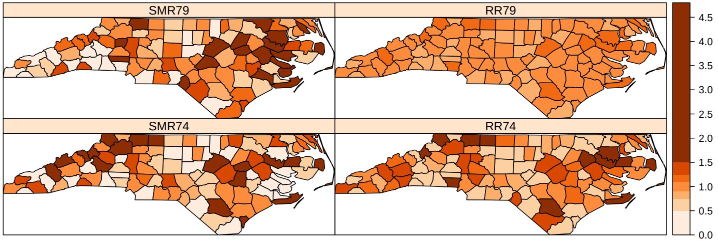
```
> summary(ncglm74)
Call:
glm(formula = SID74 ~ offset(log(EXP74)) + NWPROP74, family = "poisson",
    data = nc.sids)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.1101  -0.8745  -0.2231   0.5977   3.5100

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.64627    0.09007  -7.175 7.22e-13 ***
NWPROP74     1.86850    0.21720   8.603  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 203.34  on 99  degrees of freedom
Residual deviance: 132.21  on 98  degrees of freedom
AIC: 441.62

Number of Fisher Scoring iterations: 4
```

Model fitting: 1975-1979

```
> summary(ncglm79)
Call:
glm(formula = SID79 ~ offset(log(EXP79)) + NWPROP79, family = "poisson",
    data = nc.sids)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.7936  -0.8647   0.0773   0.6357   3.4748

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.16697    0.07725  -2.161   0.0307 *
NWPROP79     0.51020    0.20662   2.469   0.0135 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 167.85  on 99  degrees of freedom
Residual deviance: 161.83  on 98  degrees of freedom
AIC: 498.1

Number of Fisher Scoring iterations: 4
```

Geographical data in R

```
> library(maptools)
> #Read North Carolina county map
> nc.sidsmap <- readShapePoly(system.file("etc/shapes/sids.shp",
+     package="spdep")[1], ID="FIPSNO")
> #Compute SMR
> nc.sidsmap$SMR74<-nc.sids$SMR74
> nc.sidsmap$RR74<-exp(coefficients(ncglm74)[1]+coefficients(ncglm74)[2]*nc.sids$NWP
> nc.sidsmap$SMR79<-nc.sids$SMR79
> nc.sidsmap$RR79<-exp(coefficients(ncglm79)[1]+coefficients(ncglm79)[2]*nc.sids$NWP
>
```



Statistical analysis of lattice data

Tests for spatial dependence

- Moran's I
- Geary's c

Spatial smoothing

- Global smoothing
- Local smoothing

Accounting for spatial dependence

- Autocorrelated models (spatial dependence in the error term)
- Mixed-effects models (spatial dependence in a random term)

Disease Mapping

Introduction

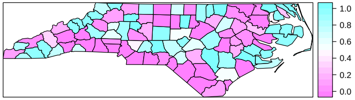
- Provide risk estimates in the region of study
- Data collected by Health Authorities
- Mortality and morbidity can be mapped

Further topics (Waller and Gotway, 2003;Elliott et al., 2000)

- Detection and assessment of risk factors: socio-economic background, environmental variables, etc.
- Detection of disease clusters
- Assessment of risk around putative pollution sources
- Analysis of case-control data as well as lattice data

Probability maps and smoothing

```
> nc.sidsmap$Observed<-nc.sidsmap$SID74
> nc.sidsmap$Expected<-nc.sids$EXP74
> nc.sidsmap$NWPROP<-nc.sids$NWBIR74/nc.sids$BIR74
> nc.sidsmap$SMR<-nc.sidsmap$Observed/nc.sidsmap$Expected
> nc.sidsmap$PPOIS<-ppois(nc.sidsmap$Observed,
+    nc.sidsmap$Expected, lower.tail =FALSE)
```



- Probability maps can be produced to show how likely the observed numbers of cases are
- Probability maps may account for the population size better than the SMR, which may show high extreme values in low populated areas
- In principle, probability maps can be computed for all types of models

Weaknesses of probability maps

If the underlying distribution of the data does not agree with our assumption, we may get several possible processes mixed up, overdispersion with spatial dependence:

```
> table(findInterval(nc.sidsmap$PPOIS, seq(0, 1, 1/10)))

 1  2  3  4  5  6  7  8  9 10
36  9  3  1  2  3  6  6  5 29

>
```

Bayesian Models

Main ideas

- All unknown *quantities* are treated as random variables
- Inference is based on the *posterior* distribution

$$f(\theta|y) \propto f(y|\theta)f(\theta)$$

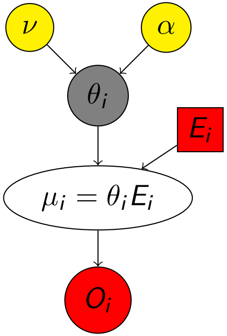
- $f(y|\theta)$ represents the likelihood of the model
- $f(\theta)$ is the *prior* distribution of θ and reflects the prior information about the parameters of the model
- $f(\theta)$ may be taken as vague as possible or according to some prior knowledge

Disease Mapping

$$\begin{array}{rcl} O_i|\theta_i & \sim & Po(E_i\theta_i) \\ \theta_i & \sim & f(\alpha) \end{array}$$

Poisson-Gamma model (Clayton and Kaldor, 1987)

$$\begin{array}{rcl} O_i|\theta_i & \sim & Po(E_i\theta_i) \\ \theta_i & \sim & Ga(\nu,\alpha) \end{array}$$



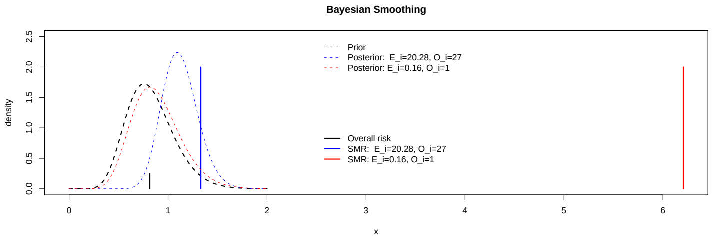
Derived distributions:

- $f(\theta_i|O_i) = Ga(O_i+\nu, E_i+\alpha)$
- $O_i|\nu,\alpha \sim \mathrm{NegBin}(\nu, p_i = \alpha/(\alpha+E_i))$
  This distribution allows for a higher variability of O_i

Bayesian learning

- The prior mean of θ_i is ν/α is a sort of *prior* knowledge
- The posterior means of the relative risks are

$$\hat{\theta}_i = \frac{O_i + \nu}{E_i + \alpha} = (1 - p_i) SMR_i + p_i \frac{\nu}{\alpha}; \qquad p_i = \alpha/(\alpha + E_i)$$



Empirical Bayes (Shrinkage) Estimators

Empirical Bayes estimation is based on computing the hyperparameters in the model from the data.

Poisson-Gamma model (Clayton and Kaldor, 1987)

- $\hat{\nu}$ and $\hat{\alpha}$ are computed using the Methods of Moments
- $\hat{\theta}_i = \frac{O_i+\hat{\nu}}{E_i+\hat{\alpha}} = (1-\hat{p}_i)SMR_i + \hat{p}_i\frac{\hat{\nu}}{\hat{\alpha}}$

Marshall's estimator (Marshall, 1991)

- Prior assumption: $E[\theta_i] = \mu,\ V[\theta_i] = \sigma^2$
- $\hat{\theta}_i = C_i\mu + (1-C_i)SMR_i$
- $\hat{\mu} = \frac{\sum_i O_i}{\sum_i E_i}$
- $\hat{C}_i = \frac{s^2-\hat{\mu}/\overline{E}}{s^2-\hat{\mu}/\overline{E}+\hat{\mu}/E_i}$

Empirical Bayes smoothing

The method of moments approach is implemented in spdep, while the maximum likelihood approach is implemented in DCluster:

```
> library(spdep)

> eb1 <- EBest(nc.sidsmap$Observed,nc.sidsmap$Expected)
> unlist(attr(eb1, "parameters"))

      a       b
3.46409 1.00000

> nc.sidsmap$EB_mm <-eb1$estmm

> library(DCluster)
> res <- empbaysmooth(nc.sidsmap$Observed,nc.sidsmap$Expected)
> unlist(res[2:3])

       nu     alpha
0.4764114 0.2185116

> nc.sidsmap$EB_ml <- res$smthrr
```

Accounting for spatial structure

Motivation

- Neighbours are likely to have similar risks
- PG and Marshall will produce the same results if the values are permuted at random
- Topology of the map needs to be taken into account in some way

Marshall's *local* estimator (Marshall, 1991)

- A spatial version was proposed considering that the neighbours have equal mean and variance instead of the global mean and variance
- The spatial smoothing is obtained because the shrinkage is done towards the local mean

Smoothing using spatial methods

How to define spatial structure?

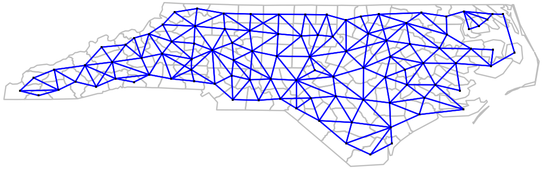
- A spatial structure (sometimes called *topology*) is usually defined by means of an area's neighbours
- Adjacency between areas (i.e., having a common border) is a common criterion to defined neighbours
- Other criteria can be used: road distance, geographical distance, etc.

Adjacency matrix

- `nb` objects are used to represent a list of neighbours
- Function `poly2nb()` can be used to extract adjancencies from a `SpatialPolygons` object
- Other similar functions can be used to create `nb` objects
- In addition to neighbourhood, a weight to measure the strength of the relationship can be defined (ans stored in a `listw` object)

Neighbours

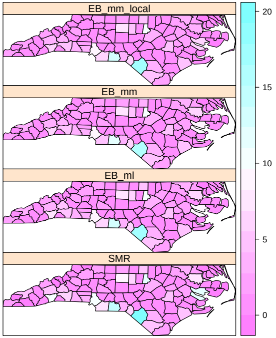
```
> neigh<-poly2nb(nc.sidsmap)
> plot(nc.sidsmap, border="gray")
> plot(neigh, coordinates(nc.sidsmap), pch=".", col="blue", add=TRUE)
>
```



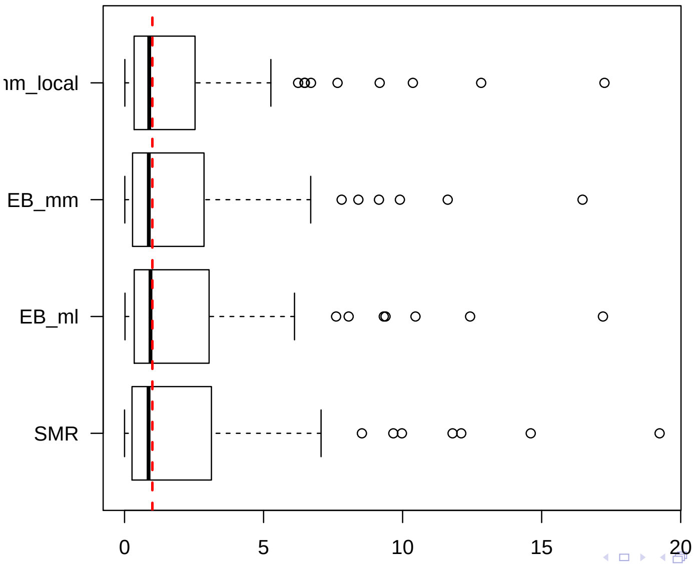
Local Empirical Bayes smoothing

If instead of shrinking to a global rate, we shrink to a local rate, we may be able to take unobserved heterogeneity into account; here we use the list of neighbours:

```
> eb2 <- EBlocal(nc.sidsmap$Observed, nc.sidsmap$Expected, neigh)
> nc.sidsmap$EB_mm_local <- eb2$est
```



Comparison of estimators



Measures of spatial dependence

Moran's I

Moran's I is a measure of spatial autocorrelation and it is defined as

$$I = \frac{n}{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} w_{ij}(y_i - \overline{y})(y_j - \overline{y})}{\sum_{i=1}^{n}(y_i - \overline{y})^2}$$

w_{ij} are spatial weights.

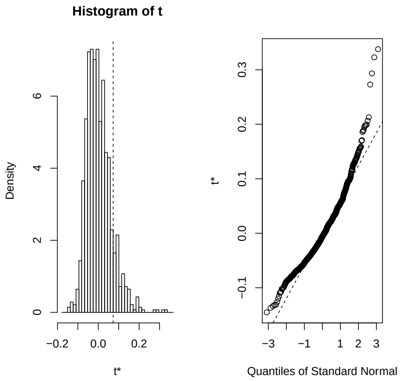
Tests for spatial dependence

- Tests based on a Normal approximation
- Permutation tests
- Parametric bootstrap

Moran's I

DCluster provides a permutation bootstrap test for spatial autocorrelation of the difference between observed and expected counts:

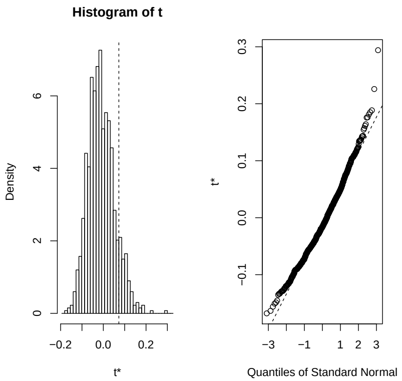
```
> lw <- nb2listw(neigh)
> set.seed(1)
> moran.boot <- boot(as(nc.sidsmap, "data.frame"), statistic = moranI.boot,
+ R = 999, listw = lw, n = length(neigh), S0 = Szero(lw))
```



Moran's I

It also provides parametric bootstraps for variants, including the Negative Binomial:

```
> moran.pgboot <- boot(as(nc.sidsmap, "data.frame"), statistic = moranI.pboot,
+    sim = "parametric", ran.gen = negbin.sim, R = 999, listw = lw, n =
+ length(neigh), S0 = Szero(lw))
```



Assunçao and Reis' correction

The Assunçao and Reis' correction to Moran's I (necessary when rates are used) is implemented in spdep:

```
> EBImoran.mc(nc.sidsmap$Observed, nc.sidsmap$Expected, lw, nsim = 999)

Monte-Carlo simulation of Empirical Bayes Index

data:  cases: nc.sidsmap$Observed, risk population: nc.sidsmap$Expected
weights: lw
number of simulations + 1: 1000

statistic = 0.2317, observed rank = 927, p-value = 0.073
alternative hypothesis: greater
```

Spatial Models for Lattice Data

- We will focus on the use of Generalized Linear Models
- In particular, Poisson models

$$O_i \sim Po(\mu_i) \quad \log(\mu_i) = \alpha + \beta x_i + u_i + v_i$$

- $u_i \sim N(0, \sigma^2_u)$ is a random effect that accounts for non-spatial variation
- $v_i \sim N(0, G)$ is a random effects that accounts for spatial variation, encoded in variance-covariance matrix G:
  - Spatially Autoregressive Specification (SAR models)

$$G = \sigma^2_v[(I - \rho W)^T (I - \rho W)]^{-1}$$

  - Conditionally Autoregressive Specification (CAR models)

$$G = \sigma^2_v[I - \rho W]^{-1}$$

Fitting base GLM models

We can fit GLMs for the base model with only the intercept, for the Poisson, quasi-Poisson, and Negative Binomial, to give a starting point with respect to overdispersion:

```
> base.glm <- glm(Observed ~ 1 + offset(log(Expected)), data = nc.sidsmap,
+    family = poisson())
> base.glmQ <- glm(Observed ~ 1 + offset(log(Expected)), data = nc.sidsmap,
+    family = quasipoisson())
> library(MASS)
> base.nb <- glm.nb(Observed ~ 1 + offset(log(Expected)), data = nc.sidsmap)
```

Overdispersion

Motivation

The Poisson assumption may be too strict in some cases

- It imposes $E[O_i] = Var[O_i]$
- Usually, $E[O_i] < Var[O_i]$
- E_i and θ_i may have not been estimated with accuracy: important covariates missing, spatial structure ignored, etc.
- Overdispersion may *appear* if the wrong model is used

Solutions

- Propose a better model
- Incorporate significant covariates
- Use random effects to account for spatial and non-spatial patterns

Tests for overdispersion

Tests for overdispersion, based in part on work by Dean, are provided in DCluster:

```
> test.nb.pois(base.nb, base.glm)

Likelihood ratio test for overdispersion

data:  base.nb : base.glm
LR = 904.0209, = 1, p-value < 2.2e-16
sample estimates:
      zscore p.mayor.modZ
3.6175325718 0.0002974249

> DeanB(base.glm)

Dean's P_B test for overdispersion

data:  base.glm
P_B = 72.1199, p-value < 2.2e-16
alternative hypothesis: greater
```

Fitting GLMs

We can augment the base model with the proportion of non-white births and other socio-economic risk factors (if available):

```
> nc.sidsmap.glm <- glm(Observed ~ offset(log(Expected))+NWPROP,
+     data = nc.sidsmap, family = poisson())
> nc.sidsmap.glmQ <- glm(Observed ~ offset(log(Expected))+NWPROP,
+     data = nc.sidsmap, family = quasipoisson())
> nc.sidsmap.nb <- glm.nb(Observed ~ offset(log(Expected))+NWPROP,
+     data = nc.sidsmap)
> #unlist(summary(nc.sidsmap.nb)[20:21])
>
> anova(base.nb, nc.sidsmap.nb)

Likelihood ratio tests of Negative Binomial Models

Response: Observed
                           Model     theta Resid. df    2 x log-lik.   Test
1       1 + offset(log(Expected)) 0.5920218        99       -642.0286
2 offset(log(Expected)) + NWPROP 0.6032657        98       -640.2443 1 vs 2
     df LR stat.   Pr(Chi)
1
2     1 1.784342 0.1816171
```

Fitting GLMs

```
> summary(nc.sidsmap.nb)
Call:
glm.nb(formula = Observed ~ offset(log(Expected)) + NWPROP, data = nc.sidsmap,
    init.theta = 0.6032657263, link = log)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.3681  -1.1713  -0.5126   0.2832   2.1313

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   1.0151     0.2440    4.16 3.19e-05 ***
NWPROP       -0.8531     0.6561   -1.30    0.193
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(0.6033) family taken to be 1)

    Null deviance: 118.54  on 99  degrees of freedom
Residual deviance: 116.74  on 98  degrees of freedom
AIC: 646.24

Number of Fisher Scoring iterations: 1


              Theta:  0.6033
          Std. Err.:  0.0878

 2 x log-likelihood:  -640.2440
>
```

Tests for overdispersion (after adjusting for covariates)

Overdispersion may be partly explained when important factors are accounted for. In the end, it may not be spatial correlation but model misspecification:

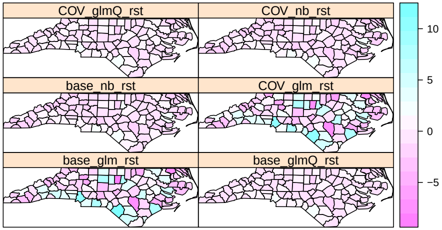
```
> DeanB(base.glm)

Dean's P_B test for overdispersion

data:  base.glm
P_B = 72.1199, p-value < 2.2e-16
alternative hypothesis: greater

>

> DeanB(nc.sidsmap.glm)

Dean's P_B test for overdispersion

data:  nc.sidsmap.glm
P_B = 71.6409, p-value < 2.2e-16
alternative hypothesis: greater
```

Residuals of the GLMs

In the same way that we ”stacked up”smoothed rates, we can add the standardised residulats of the GLM fits to our Spatial object, to examine them visually for patterning:

```
> nc.sidsmap$base_glm_rst <- rstandard(base.glm)
> nc.sidsmap$base_glmQ_rst <- rstandard(base.glmQ)
> nc.sidsmap$base_nb_rst <- rstandard(base.nb)
> nc.sidsmap$COV_glm_rst <- rstandard(nc.sidsmap.glm)
> nc.sidsmap$COV_glmQ_rst <- rstandard(nc.sidsmap.glmQ)
> nc.sidsmap$COV_nb_rst <- rstandard(nc.sidsmap.nb)
```

Residuals of the GLMs



Detection of disease clusters

- Main question: do areas with high risk tend to appear together?
- Spatial autocorrelation may indicate the presence of clusters
- Kulldorf(2006) makes an extensive review of these methods
- Types of methods:
  - General methods; assessment of clustering
  - Focused methods; assessment of risk around pollution sources
  - Scan methods; detect actual location of clusters

Geographic Analysis Machine

- This is the paradigm of scan method
- The method works as follows:
  - A (regular) grid that covers the study area is defined
  - A circular window is placed in time at each point of the grid
  - At each point, a test for clustering is performed on the areas included in the window
  - The default is to compute the p-value using a Poisson model
  - If the test is significant, then the circle is drawn
- We will end up with some (lots?) of overlapping circles highlighting the locations of the clusters

Kulldorff's Spatial Scan Statistic

- In order to avoid the problem of multiple clustering, only the most likely cluster in the area is considered
- the LRT is based on comparing the risk inside the window to the risk outside the window; significance is obtained with a MC test

Detection of disease clusters

DCluster implements several methods for the detection of disease clusters. We may use the spatial scan statistic on this data:

```
> nc.sidsmap$x<-coordinates(nc.sidsmap)[,1]
> nc.sidsmap$y<-coordinates(nc.sidsmap)[,2]
> mle<-calculate.mle(as(nc.sidsmap, "data.frame"), model="poisson")
> knresults<-opgam(data=as(nc.sidsmap, "data.frame"),
+    thegrid=as(nc.sidsmap, "data.frame")[,c("x","y")],
+    alpha=.05, iscluster=kn.iscluster, fractpop=.5, R=100, model="poisson",
+    mle=mle)
> knresults[order(knresults$statistic, decreasing =TRUE), ][1:10, ]

              x        y    statistic cluster     pvalue size
37179 -80.53134 34.98631 1.064096e+32       1 0.00990099    4
37155 -79.10111 34.63874 2.158523e+27       1 0.00990099    1
37119 -80.82937 35.24492 5.475084e+26       1 0.00990099    1
37047 -78.65492 34.26323 1.521457e+26       1 0.00990099    5
37071 -81.17521 35.29235 1.596468e+24       1 0.00990099    3
37109 -81.22072 35.48059 6.380103e+19       1 0.00990099    6
37045 -81.54949 35.32960 4.286854e+18       1 0.00990099   11
37165 -79.47720 34.83949 4.075319e+15       1 0.00990099    6
37153 -79.74231 35.00146 2.189350e+15       1 0.00990099   20
37025 -80.55087 35.38489 4.916856e+13       1 0.00990099    5
```

Bayesian Spatial Models

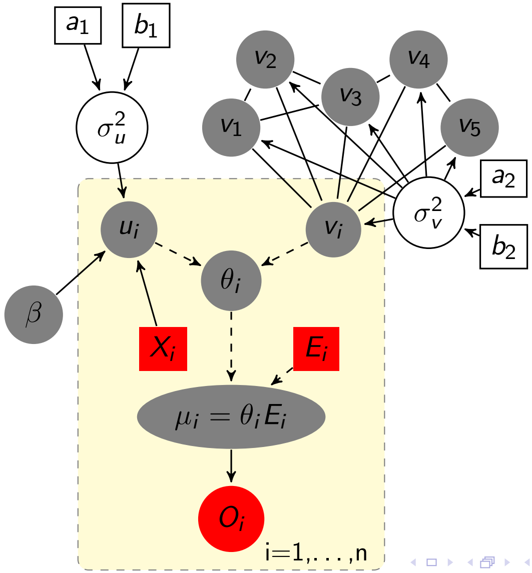
- We are pretty badly misspecified, but can the spatial dependence be separated from the distributional assumptions?
- There is a paper in Geographical Analysis using permutation bootstrap on Moran's I of the deviance residuals of GLM fits, but this doesn't help with overdispersion
- Bayesian Models provide a framework to propose and fit models which specify different sources of variation as random effects
- There is a function to export neighbour lists to Brugs/Openbugs and/or WinBUGS. A similar function exists for INLA
- We can already export SpatialPolygons to WinBugs, but here this would require further manual intervention to set links to islands

Besag, York and Mollié model (Besag et al., 1991)

BYM split the risk into 3 main effects: covariates, unstructured random effects and spatial random effects

$$
\begin{array}{ccc}
O_i & \sim & Po(E_i\theta_i) \\
\log(\theta_i) & = & \alpha + \beta X_i + u_i + v_i \\
 & & \\
u_i & \sim & N(0, \sigma^2_u) \\
 & & \\
v_i & \sim & N(\frac{\sum_{j\sim i} v_j}{n_i}, \frac{\sigma^2_v}{n_i}) \\
 & & \\
f(\alpha) & \propto & 1 \\
f(\beta) & \propto & 1 \\
 & & \\
\sigma^2_u & \sim & Ga^{-1}(a_1, b_1) \\
\sigma^2_v & \sim & Ga^{-1}(a_2, b_2)
\end{array}
$$

BYM model: graphical representation



Bayesian inference for spatial models

- In general, $f(\theta|y)$ cannot be obtained in closed form
- Markov Chain Monte Carlo (MCMC) methods provide algorithms to obtain a sample from $f(\theta|y)$
- Different sampling algorithms (Gibbs sampling, Metropolis-Hastings, ...)
- These methods require:
  - Model
  - Data
  - Starting values of the parameter
- MCMC provide a sample of the joint distribution of the full ensemble of parameters (i.e., a multivariate distribution)

Bayesian inference with **WinBUGS**

- **WinBUGS** implements the Gibbs Sampler and other computationally efficient methods to handle a large number of models
- Available from `http://www.mrc-bsu.cam.ac.uk/software/bugs/`
- **OpenBUGS** provides a completely open source alternative to **WinBUGS**
- **R2WinBUGS** can be used to call **WinBUGS** from R
- Implements some specific models for spatial and spatio-temporal data analysis
- This can be problematic on some operating systems
- Call is done using `bugs()`
- Model is supplied in an external file

An example with WinBUGS

```
> WBneigh<-nb2WB(neigh)
> WBdata<-list(observed=nc.sidsmap$Observed, expected=nc.sidsmap$Expected,
+    N=nrow(nc.sidsmap), NWPROP=nc.sidsmap$NWPROP)
> WBdata<-c(WBdata, WBneigh)
> WBinits<-list(alpha=0, beta=c(0),
+    u=rep(0, nrow(nc.sidsmap)), v=rep(0, nrow(nc.sidsmap)), precu=1, precv=1
+ )
> #Set model file and working directory
> modelf<-paste(getwd(), "models/BYM-model.txt", sep="/")
> wdbym<-paste(getwd(), "results/BYM-dismap", sep="/")
> library(R2WinBUGS)
> BYM<-bugs(data=WBdata, inits=list(WBinits), parameters.to.save=c("theta"),
+ n.chains=1, n.thin=5, DIC=TRUE,
+    working.directory = wdbym,
+    n.iter=15000, n.burnin=10000, debug=TRUE,
+    model.file=modelf
+ )
>
>

> nc.sidsmap$BYM<-BYM$mean$theta
```

Approximate Bayesian Inference with INLA

- INLA stands for *Integrated Nested Laplace Approximation*
- Methodological approach described in Rue et al. (2009)
- Implemented in the **INLA** (sometimes called **R-INLA**) package
- INLA computes an approximation to the marginal distribution of the model parameters (i.e., $f(\theta_i|y)$) instead of the full joint posterior $f(\theta_i|y)$
- Uses computationally efficient algorithms for the computations
- VERY fast
- Flexible model building using a `formula`
- Call is done through `inla()`

Approximate Bayesian Inference with INLA

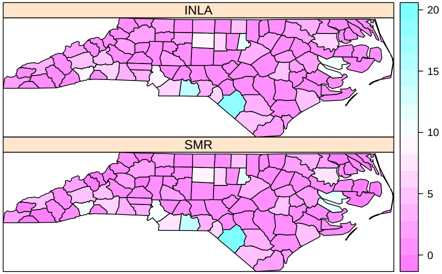
- Spatial effects are included in the model `formula` using the `f()` function
- Some interesting models are shown in the table below
- Check `http://www.r-inla.org` for more details

Name in `f()`	Model	Regular grid
`besag`	Intrinsic CAR	No
`besagproper`	Proper CAR	No
`bym`	Convolution model	No
`generic0`	$\Sigma = \frac{1}{\tau} Q^{-1}$	No
`generic1`	$\Sigma = \frac{1}{\tau}(I_n - \frac{\rho}{\lambda_{max}} C)^{-1}$	No
`rw2d`	2-D random walk	Yes
`matern2d`	Matérn correlation	Yes

Table : Summary of some latent models implemented in **R-INLA** for spatial statistics (Bivand et al., 2014, submitted to JSS).

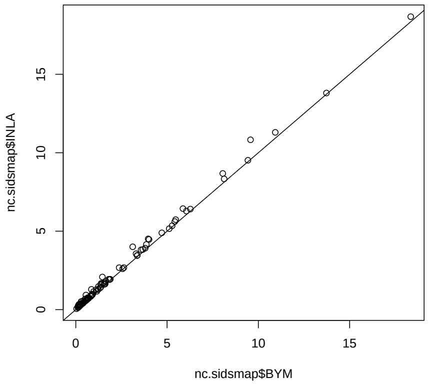
Approximate Bayesian Inference with INLA

```
> library(INLA)
> library(spdep)
> nb2INLA(file="nc.sidsmap.graph", neigh)
> nc.sidsmap$ID<-1:100#as.character(1:100)
> formula  <-  Observed~1+NWPROP+f(ID,model="bym",graph="nc.sidsmap.graph",param=c(1,0.00005),initial=2.8)#+ f(F
> mod  <-  inla(formula,family="poisson",data=as(nc.sidsmap, "data.frame"),
+    E=nc.sidsmap$Expected, control.inla=list(h=0.01),verbose=TRUE,
+    control.compute=list(dic=TRUE),
+    control.predictor=list(compute=TRUE))
> nc.sidsmap$INLA<-mod$summary.fitted.values$mean
```



MCMC versus INLA

```
DIC computed with WinBUGS: 491.904
DIC computed with INLA: 502.4917
```



Disease mapping with BayesX

- **BayesX** is a similar software for Bayesian inference
- Different estimation methods: MCMC, REML, PLS
- Flexible model building using a `formula`
- Spatial effects are included in the `formula` using a `sx()` function
- Provides a number of geo-additive effects, such as, splines that are *difficult* to implement in **WinBUGS** and **R-INLA**
- Call is done through `bayesx()`

Disease mapping with BayesX

- Spatial effects are included in the model `formula` using the `sx()` function
- Some interesting models are shown in the table below
- Check `http://www.bayesx.org` for more details or the manual page of `sx`

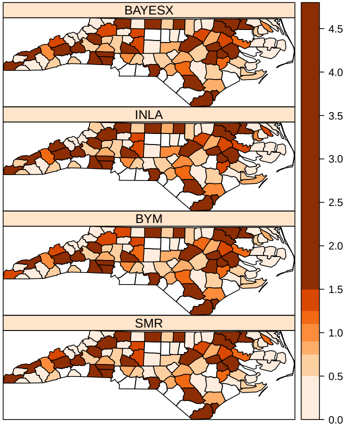
Name in `f()`	Model
`te`	Two dimensional P-spline
`kr`	Kriging with stationary Gaussian random fields
`gk`	Geokriging with stationary Gaussian random fields
`gs`	Geosplines based on two-dimensional P-splines
`mrf`	Markov random fields

Table : Summary of some latent models implemented in **BayesX** for spatial statistics.

Disease mapping with BayesX

```
> library(R2BayesX)
> ncgra <- nb2gra(neigh)
> nc.sidsmap$IDXSP <- as.numeric(rownames(ncgra))#Index for spatial effect
> if(!file.exists("results/bayesx-dismap.RData")) {
+
+ sidsbayesx <- bayesx(Observed ~ NWPROP+ sx(ID, bs = "re") +
+    sx(IDXSP, bs = "spatial", map = ncgra),
+    offset = log(nc.sidsmap$Expected), family = "poisson",
+    data = as(nc.sidsmap, "data.frame") )
+
+
+ save(file="results/bayesx-dismap.RData", list=c("sidsbayesx"))
+
+ } else{
+    load("results/bayesx-dismap.RData")
+ }
> nc.sidsmap$BAYESX <- sidsbayesx$fitted.values[order(sidsbayesx$bayesx.setup$order),2]/nc.sidsmap$Expected
```

Comparing different estimates


BAYESX
INLA
BYM
SMR
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0