Journal of Statistical Software

MMMMMM YYYY, Volume VV, Issue II. *http://www.jstatsoft.org/*

DClusterm: Model-based Detection of Disease Clusters

Virgilio Gómez-Rubio
Universidad de Castilla-La Mancha

Paula Moraga
Queensland University of Technology

John Molitor
Oregon State University

Barry Rowlingson
Lancaster University

Abstract

The detection of regions with unusual high risk plays an important role in disease mapping and the analysis of public health data. In particular, the detection of groups of areas (i.e., clusters) where the risk is significantly high is often conducted by Public Health authorities.

Many methods have been proposed for the detection of disease clusters, most of them based on moving windows, such as Kulldorff's Spatial Scan Statistics (SSS). Here we describe a model-based approach for the detection of disease clusters implemented in the **DClusterm** package. Our model-based approach is based on representing a large number of possible clusters by dummy variables and then fitting many generalized linear models to the data where these covariates are included one at a time. Cluster detection is done by performing a variable or model selection among all fitted models using different criteria.

Because of our model-based approach, cluster detection can be performed using different types of likelihoods and latent effects. We cover the detection of spatial and spatio-temporal clusters, as well as how to account for covariates, zero-inflated datasets and overdispersion in the data.

Keywords: disease cluster, spatial statistics, R.

1. Introduction

The analysis of epidemiological data at small area level often involves accounting for possible risk factors and other important covariates using different types of regression models. However, it is not uncommon that after a number of covariates have been accounted for, residuals

still show a spatial distribution that defines some groups of areas with unusual high epidemiological risk. Hence, in many occasions it is not clear whether all spatial risk factors have been included in our model.

Public health data are often aggregated over small administrative areas because of confidentiality issues, but it is not uncommon that individual data are available. Generalized Linear Models (GLM, McCullagh and Nelder 1989) are a common framework for disease mapping to model aggregated and individual data. GLMs not only model Poisson or Binomial responses, but they can also link the outcome to a linear predictor on the covariates (and, possibly, other effects). However, until recently, it was not clear how to use GLMs to detect clusters of disease, i.e., a group of contiguous areas with significant high risk.

In order to detect disease clusters, the most widely used method is probably the Spatial Scan Statistic (SSS) proposed by Kulldorff (1997). Given a possible cluster z, the SSS will compare the relative risk in the cluster θ_z to the relative risk outside the cluster $\theta_{\overline{z}}$ using the following test:

$$\begin{aligned} H_0: &\quad \theta_z = \theta_{\overline{z}} \\ H_1: &\quad \theta_z > \theta_{\overline{z}} \end{aligned}$$

Here, a likelihood ratio statistic is used. Many different possible clusters are tested in turn by changing the areas in z and the most likely cluster (i.e., the one with the highest value of the test statistic) is selected. Significance of this cluster is assessed with a Monte Carlo test that also provides the p-value.

In this paper we will describe a novel approach to disease cluster detection that provides a link between GLMs and SSS. We have implemented this approach via the freely available **DClusterm** package for the R statistical software. This approach involves fitting many different GLMs for which dummy variables that represent possible clusters are included one at a time. Cluster detection is based on selecting a number of dummy cluster variables using variable selection methods.

In this paper we will describe how some methods that provide a link between GLMs and SSS which have been implemented in the **DClusterm** package for the R software, so that the detection of disease clusters is approached from a regression point of view. As described later, this will involve fitting many different GLMs for which dummy variables that represent possible clusters are included one at a time. Cluster detection is based on selecting a number of dummy cluster variables using variable selection methods.

This paper is organized as follows. Section 2 will introduce the link between GLM and SSS. Next, in Section 3 we show how to include random effects in the detection of disease clusters to account for over-dispersion. The detection of disease clusters for zero-inflated data is discussed in Section 4. Section 5 describes how to extend these ideas to detect clusters in space and time. Finally, a discussion and some final remarks are provided in Section 6.

2. Generalized linear models for cluster detection

2.1. General description

Jung (2009); Zhang and Lin (2009) provide an explicit link between GLMs and the SSS, and

show that the test statistic for a given cluster is equivalent to fitting a GLM using a cluster variable as a predictor. This cluster variable is a dummy variable which is 1 for the areas in the cluster and 0 for the areas outside the cluster. By including cluster covariates in the model we obtain an estimate of the increased risk (as measured by its associated coefficient) and its significance (by means of the associated p-value, for example).

For example, for a Poisson model with expected counts E_i we could have the following distribution for the observed cases O_i:

$$O_i \sim Po(E_i\theta_i)$$

$$\log(\mu_i) = \log(E_i) + \alpha + \beta x_i$$

We let $\log(E_i)$ denote an offset to account for the population structure (age, gender, etc.). θ_i is the relative risk and it measures any deviation (increase or decrease) in the incidence of the disease from the expected number of cases. Finally, x_i represents a covariate with possible risk factors associated with the disease.

Fitting this model will provide estimates $\hat{\alpha}$ and $\hat{\beta}$. This will account for the (spatial) effects of the covariates. In order to include the cluster variable the effects of the covariates will be kept fixed. Hence, the cluster covariates will be used in a model with fixed coefficients for the covariates:

$$\log(\mu_i) = \log(E_i) + \hat{\alpha} + \hat{\beta} x_i + \gamma_j c_i^{(j)}$$

This means that the offset now is $\log(E_i)+\hat{\alpha}+\hat{\beta}x_i$. $c_i^{(j)}$ is a dummy cluster variable associated to a cluster j, with j an index over the list of all possible clusters being tested, and defined as

$$c_i^{(j)} = \begin{cases} 1 & \text{if area } i \text{ belongs to cluster } j \\ 0 & \text{otherwise} \end{cases}$$

Here, γ_j is a measure of the risk in the cluster. We are only interested in clusters whose coefficient is significantly higher than 0 (i.e., increased risk), hence those with a significant negative coefficient will be ignored. Testing different clusters will produce many different cluster covariates. We can use model selection techniques to select the most important cluster in the study region. In particular, the log-likelihood can be used to compare the model with the cluster variable to the null model (i.e., the one with the covariates only).

Regarding the effect of the covariates, it is possible to perform cluster detection without considering covariates in the model. Then a cluster detection accounting for the covariates will likely provide a different number of clusters. By comparing the clusters detected in both cases we will be able to find which clusters are linked to underlying risk factors included in the model and which clusters remain unexplained by the covariates. In the examples that we include in this paper we will always consider both scenarios to better understand how cluster detection works with these methods.

Bilancia and Demarinis (2014); Gómez-Rubio, Moraga, and Molitor (2015) describe a similar approach to the detection of disease clusters using Bayesian hierarchical models. The Integrated Nested Laplace Approximation (INLA) is used in both cases for model fitting as

it provides computational benefits over other computationally expensive methods, such as Markov Chain Monte Carlo.

2.2. Leukemia in upstate New York

We first consider an example where we model counts of leukemia cases in upstate New York. These data are provided in the `NY8` dataset, available in package `DClusterm`. It provides cases of leukemia in different census tracts in upstate New York. This data set has been analyzed by several authors (Waller, Turnbull, Clark, and Nasca 1992; Waller and Gotway 2004). The location of leukemia is thought to be linked to the use of Trichloroethene (TCE) by several companies in the area. Figure 1 shows the Standardized Mortality Ratios of the census tracts and the locations of the industries using TCE.

In order to measure exposure, the inverse of the distance to the nearest TCE site has been used (variable `PEXPOSURE` in the dataset). In addition, two other socioeconomic covariates have been used: the percentage of people aged 65 or more (variable `PCTAGE65P`) and the percentage of people who own their home (variable `PCTOWNHOME`).

Hence, our first action is to load some required packages and the dataset itself.

```
R> library(DClusterm)
R> library(xts)
R> data(NY8)
```

A number of cases could not be linked to their actual location and they were distributed uniformly over the study area, making the counts real numbers instead of integers. We have rounded these values as we intend to use a Poisson likelihood for the analysis. Furthermore, expected counts are computed using the overall incidence ratio (i.e., total number of cases divided by the total population). Age-sex standardization is not possible in this case as this information is not available in our dataset.

```
R> NY8$Observed <- round(NY8$Cases)
R> NY8$Expected <- NY8$POP8 * sum(NY8$Observed)/sum(NY8$POP8)
R> NY8$SMR <- NY8$Observed/NY8$Expected
```

The centres of the areas will be stored in columns named `x` and `y`. This will be used later when ordering the areas by increasing distance to the putative cluster centre. If the location of the main populated cities are available these could be used but here we will use function `coordinates()` instead:

```
R> NY8$x <- coordinates(NY8)[, 1]
R> NY8$y <- coordinates(NY8)[, 2]
```

As **DClusterm** is designed to detect clusters in space and time, it will always expect data to be from one of the classes in package **spacetime** to store all the data, even if they are available for a single period of time. In this case, a `STFDF` object is created to store all the data but note that in this case we do not have a truly space-time dataset. Hence, the time series in the data is made of a single date entry (`"1972-01-01"`) and that it is just a convenient way of having our data in a `STFDF` object.

```
R> NY8st <- STFDF(as(NY8, "SpatialPolygons"), xts(1, as.Date("1972-01-01")),
+       NY8@data, endTime = as.POSIXct(strptime(c("1972-01-01"),
+           "%Y-%m-%d"), tz = "GMT"))
```

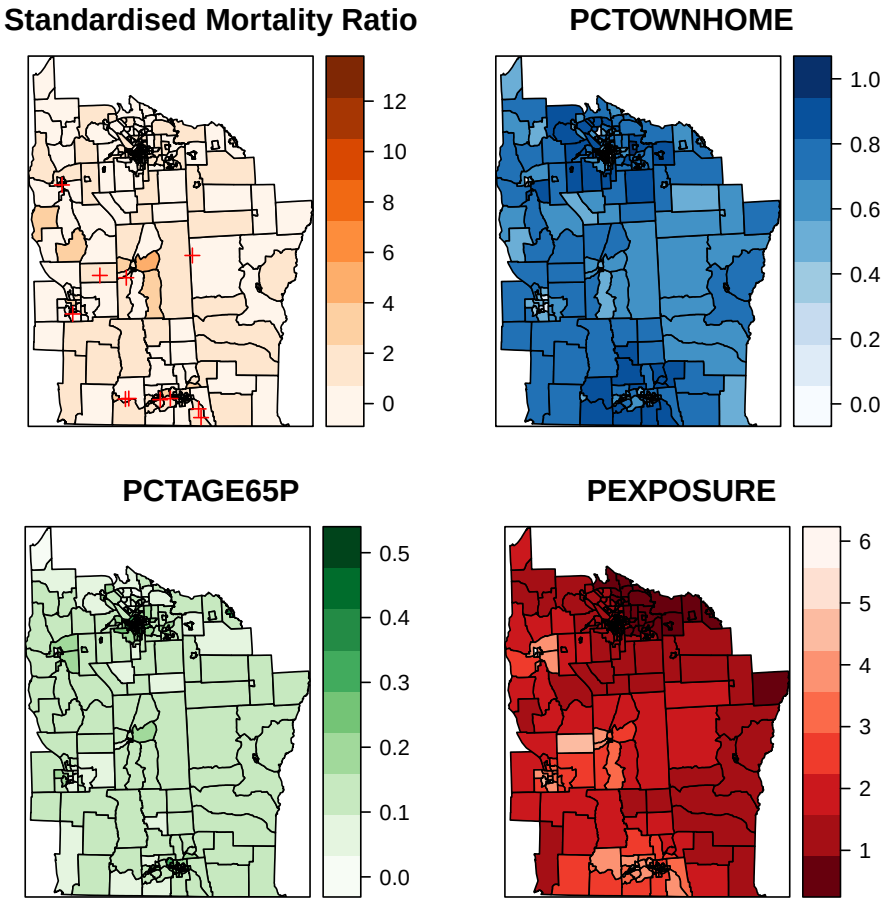

Figure 1: Standardised Mortality Ratios and covariates of the incidence of Leukemia in upstate New York dataset. PEXPOSURE is the inverse of the distance to the nearest TCE site, PCTAGE65P is the percentage people aged 65 or more, and PCTOWNHOME is the percentage of people who own their home. The red crosses in the top-left map mark the locations of the TCE sites.

2.3. Cluster detection

Cluster detection with no covariates

First of all, a model with no covariates will be fitted and used as a baseline, so that other models can be compared to this one (for example, using the AIC or the log–likelihood) to assess whether they provide a better fit.

```
R> ny.m0 <- glm(Observed ~ offset(log(Expected)) + 1, family = "poisson",
```

```
+      data = NY8)
```

Function `DetectClustersModel()` will take this baseline model (using argument `model0`), create the cluster dummy variables and test them in turn. Then, those clusters with a highest significance will be reported.

Argument `thegrid` will take a 2-column `data.frame` (with names `x` and y) with the centres of possible clusters. If the grid of cluster centres is not defined, then a rectangular grid is used with a distance between adjacent points defined by argument `step`. Dummy cluster variables are created around these points by adding areas to the cluster until a certain percentage of the population (defined by argument `fractpop`) or until a certain distance about the centre (defined by argument `radius`) has been reached. When testing for significant cluster variables, argument `alpha` defines the significance level.

`DetectClustersModel()` can detect spatial and spatio-temporal clusters, which is why its first argument is a space-time object. The type of clusters that are investigated is defined by argument `typeCluster`. In the example we have used `typeCluster = "S"`.

Other options include the number of replicates for Monte Carlo tests (argument `R`) if cluster assessment is done by simulation. By default, Monte Carlo tests are not used. **DClusterm** allows for parallel computing using several cores as implemented in package **parallel**. The number of cores to use is defined in option `mc.cores` (now 4 cores are used):

```
R> options(mc.cores = 4)
```

In the following example, to reduce the computational burden, we have only looked for clusters around 5 areas (whose rows in `NY8` are defined in variable `idxcl`). In a real application we advise the use of all locations (area centroids or actual locations of individual data).

```
R> idxcl <- c(120, 12, 89, 139, 146)
R> ny.cl0 <- DetectClustersModel(NY8st, thegrid = as.data.frame(NY8)[idxcl,
+      c("x", "y")], fractpop = 0.15, alpha = 0.05, radius = Inf,
+      step = NULL, typeCluster = "S", R = NULL, model0 = ny.m0)
```

Below is a summary of the clusters detected. Dates can be ignored as this is a purely spatial cluster analysis. In the case of spatio-temporal clusters, dates shown define the temporal range of the cluster. Values `x` and `y` defined the cluster centre, `size` is the number of areas in the cluster, `statistic` represents the point estimate of the associated cluster coefficient. Also, note that only clusters with a lower `pvalue` than argument `alpha` are returned. `cluster` indicates whether the cluster is a significant one. Finally, note how detected clusters are ordered by increasing value of `pvalue`, so that the most significant clusters are reported first.

```
R> ny.cl0
```

```
           x       y size      minDateCluster      maxDateCluster
11  424728.9 4661404   39 1972-01-01 01:00:00 1972-01-01 01:00:00
88  409430.4 4720092    9 1972-01-01 01:00:00 1972-01-01 01:00:00
119 404710.7 4768346   24 1972-01-01 01:00:00 1972-01-01 01:00:00
```

```
    statistic       pvalue      risk cluster
11   8.044846 0.0000604120 0.3916904    TRUE
88   6.967107 0.0001893208 0.6455613    TRUE
119  3.254824 0.0107290781 0.4445236    TRUE
```

Function `get.knclusters()` can be used to extract the indices of the areas in the different clusters detected. In the next lines, we show how to write a new function called `get.allknclusters()` to obtain all the clusters detected together in a single variable:

```
R> get.allknclusters <- function(spdf, knresults) {
+      clusters <- rep("", nrow(spdf))
+      knclusters <- get.knclusters(spdf, knresults)
+      if (length(knclusters) > 0) {
+          clusters[unique(unlist(knclusters))] <- "CLUSTER"
+          clusters <- as.factor(clusters)
+      }
+      return(clusters)
+  }
R> NY8$CLUSTER0 <- get.allknclusters(NY8, ny.cl0)
```

The areas and centers of the clusters detected are shown in Figure 2. Because of the lack of adjustment for covariates these clusters show regions of high risk based on the raw data (observed and expected counts) alone.

Cluster detection after adjusting for covariates

Similarly, clusters can be detected after adjusting for significant risk factors. First of all, we will fit a Poisson regression with the 3 covariates mentioned earlier. As it can be seen, all three are significant:

```
R> form.m1 <- ny.m1 <- glm(Observed ~ offset(log(Expected)) +
+      PCTOWNHOME + PCTAGE65P + PEXPOSURE, family = "poisson",
+      data = NY8)
R> summary(ny.m1)

Call:
glm(formula = Observed ~ offset(log(Expected)) + PCTOWNHOME +
    PCTAGE65P + PEXPOSURE, family = "poisson", data = NY8)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.9099  -1.1294  -0.1768   0.6385   3.2426

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.65507    0.18550  -3.531 0.000413 ***
PCTOWNHOME  -0.36472    0.19316  -1.888 0.058998 .
```

```
PCTAGE65P    4.05031    0.60559   6.688 2.26e-11 ***
PEXPOSURE    0.15141    0.03165   4.784 1.72e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 459.05  on 280  degrees of freedom
Residual deviance: 384.01  on 277  degrees of freedom
AIC: 958.97

Number of Fisher Scoring iterations: 5
```

As the three covariates are significant, the expected number of cases will be different now and the detected clusters may be different in this case. Cluster detection is performed as in the previous example, but now we use the model that adjusts for covariates instead:

```
R> ny.cl1 <- DetectClustersModel(NY8st, thegrid = as.data.frame(NY8)[idxcl,
+      c("x", "y")], fractpop = 0.15, alpha = 0.05, typeCluster = "S",
+      R = NULL, model0 = ny.m1)

R> ny.cl1

           x       y size      minDateCluster      maxDateCluster
88  409430.4 4720092    9 1972-01-01 01:00:00 1972-01-01 01:00:00
119 404710.7 4768346   20 1972-01-01 01:00:00 1972-01-01 01:00:00
    statistic       pvalue      risk cluster
88   5.861204 0.0006175202 0.5869176    TRUE
119  3.160591 0.0119304026 0.4882633    TRUE
```

Similarly as in the previous example, we can use function `get.allkncluster()` to get the indices of the areas in the clusters and add a new variable to the `SpatialPolygonsDataFrame`:

```
R> NY8$CLUSTER1 <- get.allknclusters(NY8, ny.cl1)
```

Figure 2 shows the clusters detected after adjusting for covariates. Compared to the example with no covariate adjustment, one cluster has disappeared. Hence, that cluster has been explained by the effect of the covariates. Another cluster is a bit smaller in size, which means that covariates only explain a small part of it. The most significant cluster remains the same. In all cases, cluster significance has been reduced by the effect of the covariates.

3. Mixed-effects models for cluster detection

Random effects can be incorporated into our models to account for unmeasured risk factors. Cluster detection will be performed as usual, but we should keep in mind that by including random effects and dummy cluster covariates there may be a clash between the two. By using

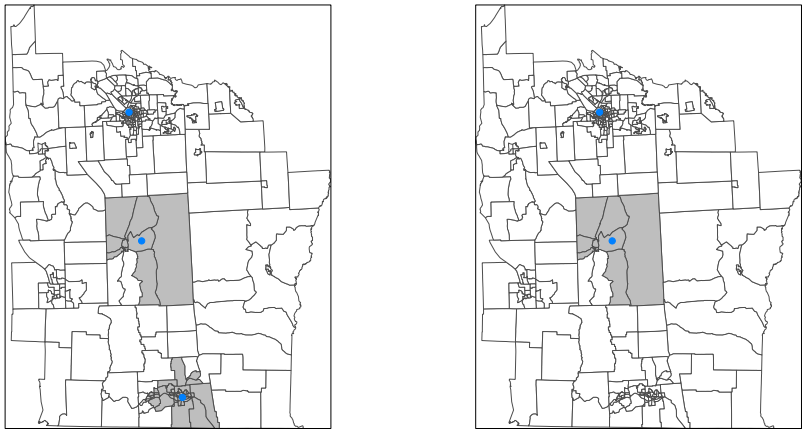

Figure 2: Clusters detected with no covariate adjustment (left) and after adjusting for covariates (right). Areas in clusters are shaded in gray and cluster centres are represented by blue dots.

dummy variables we are intentionally looking for unexplained spatial variation in the data. Hence, random effects should aim at modelling a different structure in the data.

Random effects are particularly useful to model over-dispersion in count data. For the Poisson case, this will mean that the relative risk can be modelled as:

$$\log(\mu_i) = \log(E_i) + \alpha + \beta x_i + \gamma_j c_i^{(j)} + u_i \quad (1)$$

$$u_i \sim N(0, \sigma_u^2) \quad (2)$$

where u_i represents a random effect Normally distributed with zero mean and variance σ_u^2. Note that random effects can be defined to be spatially correlated, as suggested by Bilancia and Demarinis (2014). However, this can produce a clash between the dummy cluster variables and the random effects.

3.1. Leukemia in upstate New York

We go back to the example on the leukemia incidence in upstate New York to show how models can include random effects and, at the same time, detect disease clusters. In this particular example, random effects will be important in order to reflect any over-dispersion present in the data. For this reason, our first step here is to test the data for over-dispersion using Dean's P_B and P'_B score tests (see, Dean 1992, for details). These two tests have been implemented in functions `DeanB()` and `DeanB2()` in the **DCluster** package. They both take a `glm` object and perform the score tests:

```
R> DeanB(ny.m0)

        Dean's P_B test for overdispersion

data:  ny.m0
```

```
P_B = 5.5755, p-value = 1.234e-08
alternative hypothesis: greater
```

```
R> DeanB2(ny.m0)
```

```
        Dean's P'_B test for overdispersion

data:  ny.m0
P'_B = 5.6233, p-value = 9.368e-09
alternative hypothesis: greater
```

From the results, it is clear that when no covariates are included data are clearly over-dispersed. Hence, a Poisson distribution will not be appropriate to model the observed counts in each tract.

The same tests applied to the model with covariates produce a similar result:

```
R> DeanB(ny.m1)
```

```
        Dean's P_B test for overdispersion

data:  ny.m1
P_B = 2.0145, p-value = 0.02198
alternative hypothesis: greater
```

```
R> DeanB2(ny.m1)
```

```
        Dean's P'_B test for overdispersion

data:  ny.m1
P'_B = 2.2391, p-value = 0.01257
alternative hypothesis: greater
```

Although p-values have increased, they are both small and we may still consider that data are over-dispersed. Hence, we will aim at detecting clusters using a Poisson regression with independent random effects to account for census tract-level heterogeneity.

3.2. Cluster detection

Cluster detection with no covariates

The baseline model with no covariate will now be fitted with function `glmer()` from package **lme4** (Bates, Mächler, Bolker, and Walker 2015). This function is similar to `glm()` for GLMs but it will allow us to include random effects in the model as part of the `formula` argument.

```
R> ny.mm0 <- glmer(Observed ~ offset(log(Expected)) + (1 |
+      AREANAME), data = as(NY8, "data.frame"), family = "poisson")
R> summary(ny.mm0)
```

```
Generalized linear mixed model fit by maximum likelihood (Laplace
  Approximation) [glmerMod]
 Family: poisson  ( log )
Formula: Observed ~ offset(log(Expected)) + (1 | AREANAME)
   Data: as(NY8, "data.frame")

     AIC      BIC   logLik deviance df.resid
  1010.8   1018.1   -503.4   1006.8      279

Scaled residuals:
    Min      1Q  Median      3Q     Max
-2.1185 -0.8799 -0.2617  0.7784  5.0263

Random effects:
 Groups   Name        Variance Std.Dev.
 AREANAME (Intercept) 0.2111   0.4594
Number of obs: 281, groups:  AREANAME, 64

Fixed effects:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -0.2410     0.1051  -2.293   0.0219 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
R> ny.clmm0 <- DetectClustersModel(NY8st, thegrid = as.data.frame(NY8)[idxcl,
+      c("x", "y")], fractpop = 0.15, alpha = 0.05, typeCluster = "S",
+      R = NULL, model0 = ny.mm0)

[1] 1 1

R> ny.clmm0

           x       y size      minDateCluster      maxDateCluster
88  409430.4 4720092    9 1972-01-01 01:00:00 1972-01-01 01:00:00
119 404710.7 4768346   24 1972-01-01 01:00:00 1972-01-01 01:00:00
    statistic       pvalue      risk cluster
88   7.577766 9.900781e-05 0.7880056    TRUE
119  2.415960 2.793750e-02 0.7862091    TRUE
```

After accounting for overdispersion, the number of clusters detected is 2. These are shown in Figure 3. The largest cluster detected before has now disappeared and only the two smallest clusters remain. This may be due to the fact that the first cluster has the smallest SMR and risk. Hence, allowing for extra-variation will make the discrepancy between observed and expected less extreme and this cluster will be the first one to be declared as non-significant. The two remaining clusters have the same size and a very similar risk than in the previous case.

Cluster detection with covariates

```
R> ny.mm1 <- glmer(Observed ~ offset(log(Expected)) + PCTOWNHOME +
+      PCTAGE65P + PEXPOSURE + (1 | AREANAME), data = as(NY8,
+      "data.frame"), family = "poisson")
R> summary(ny.mm1)

Generalized linear mixed model fit by maximum likelihood (Laplace
  Approximation) [glmerMod]
 Family: poisson  ( log )
Formula:
Observed ~ offset(log(Expected)) + PCTOWNHOME + PCTAGE65P + PEXPOSURE +
    (1 | AREANAME)
   Data: as(NY8, "data.frame")

     AIC      BIC   logLik deviance df.resid
   959.8    978.0   -474.9    949.8      276

Scaled residuals:
    Min      1Q  Median      3Q     Max
-2.1451 -0.8697 -0.1932  0.6963  4.0848

Random effects:
 Groups   Name        Variance Std.Dev.
 AREANAME (Intercept) 0.01532  0.1238
Number of obs: 281, groups:  AREANAME, 64

Fixed effects:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.73898    0.21947  -3.367  0.00076 ***
PCTOWNHOME  -0.38390    0.21622  -1.776  0.07582 .
PCTAGE65P    4.04222    0.62575   6.460 1.05e-10 ***
PEXPOSURE    0.16159    0.03662   4.413 1.02e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
           (Intr) PCTOWN PCTAGE
PCTOWNHOME -0.734
PCTAGE65P  -0.465  0.145
PEXPOSURE  -0.567  0.211 -0.023

R> ny.clmm1 <- DetectClustersModel(NY8st, thegrid = as.data.frame(NY8)[idxcl,
+      c("x", "y")], fractpop = 0.15, alpha = 0.05, typeCluster = "S",
+      R = NULL, model0 = ny.mm1)

[1] 1 1

R> ny.clmm1
```

```
          x       y size      minDateCluster      maxDateCluster
88 409430.4 4720092    1 1972-01-01 01:00:00 1972-01-01 01:00:00
   statistic      pvalue       risk cluster
88  4.051264 0.004420356 0.03289885    TRUE
```

When overdispersion and covariates are included in the model, only one of the clusters remains with size 1 and a slightly reduced estimate of the cluster coefficient. This should not come as a surprise given that we have already seen how including covariates explains some of the extra-variation and that by including random effects the significance of the clusters is also reduced. A summary of the clusters detected can be found in Figure 3.

As in the example in Section 2.2, we have created two cluster variables to display the clusters obtained:

```
R> NY8$CLUSTERMM0 <- get.allknclusters(NY8, ny.clmm0)
R> NY8$CLUSTERMM1 <- get.allknclusters(NY8, ny.clmm1)
```

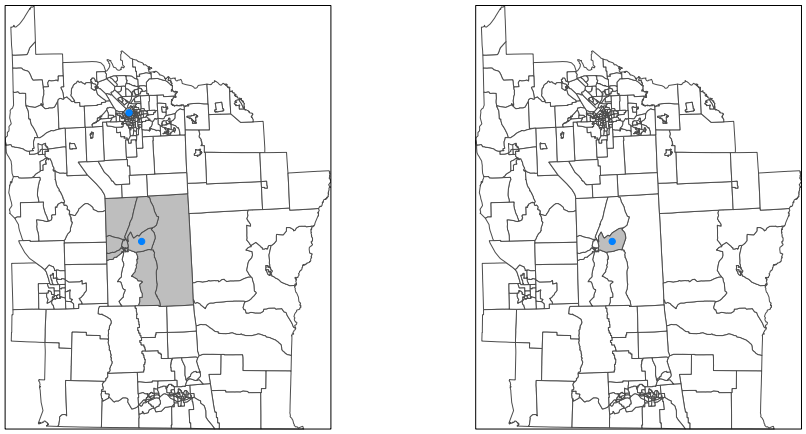

Figure 3: Clusters detected with no covariate adjustment (left) and after adjusting for covariates (right) using a mixed-effects model to account for overdispersion of the data. Areas in clusters are shaded in gray and cluster centers are represented by blue dots.

4. Zero-inflated models for cluster detection

The analysis of rare diseases often involves datasets where there are many areas with zero counts, leading to zero-inflated data. In this situation the Poisson or Binomial likelihoods may not be suitable to fit a model and other distributions for the data should be used. Gómez-Rubio and López-Quílez (2010) discuss this issue and they have extended model-based cluster detection methods to account for zero-inflation.

For count data, a zero-inflated Poisson model could be used. In this case, observed number of cases come from a mixture distribution of a Poisson and a distribution with all mass at zero. Probabilities are as follows:

$$Pr(O_i = n_i) = \begin{cases} \pi_i + (1-\pi_i)Po(0|\theta_i E_i) & n_i = 0 \\ (1-\pi_i)Po(n_i|\theta_i E_i) & n_i = 1, 2, \ldots \end{cases}$$

Here $Po(O_i|\theta_i E_i)$ represents the probability of observing O_i cases using a Poisson distribution with mean $\theta_i E_i$. π_i represents the proportion of zeroes observed that do not come from the Poisson distribution.

Relative risks θ_i can be modeled using a log-linear model to depend on some relevant risk factors. Also, it is common that all π_i's are taken equal to a single value π.

4.1. Brain cancer in Navarre (Spain)

Ugarte, Ibáñez, and Militino (2006) analyze the incidence of brain cancer in Navarre (Spain) at the health district level. Figure 4 shows the Standardized Mortality Ratios. As it can be seen there are many areas where the SMR is zero because there are no cases in those areas. Ugarte, Ibáñez, and Militino (2004) have also reported a significant zero-inflation of these data compared to a Poisson distribution. For cluster detection, the method implemented in `DClusterm` is similar to the one used in Gómez-Rubio and López-Quílez (2010) for the detection of disease clusters of rare diseases.

4.2. Cluster detection

Before starting our cluster detection methods, we will check the appropriateness of a Poisson distribution for this data. Fitting a log-linear model (with no covariates) gives the following model:

```
R> nav.m0 <- glm(OBSERVED ~ offset(log(EXPECTED)) + 1, family = "poisson",
+      data = brainnav)
R> summary(nav.m0)

Call:
glm(formula = OBSERVED ~ offset(log(EXPECTED)) + 1, family = "poisson",
    data = brainnav)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5227  -1.4783  -0.3203   0.7042   1.6393

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -7.752e-06  8.805e-02       0        1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 63.733  on 39  degrees of freedom
Residual deviance: 63.733  on 39  degrees of freedom
AIC: 145.02
```

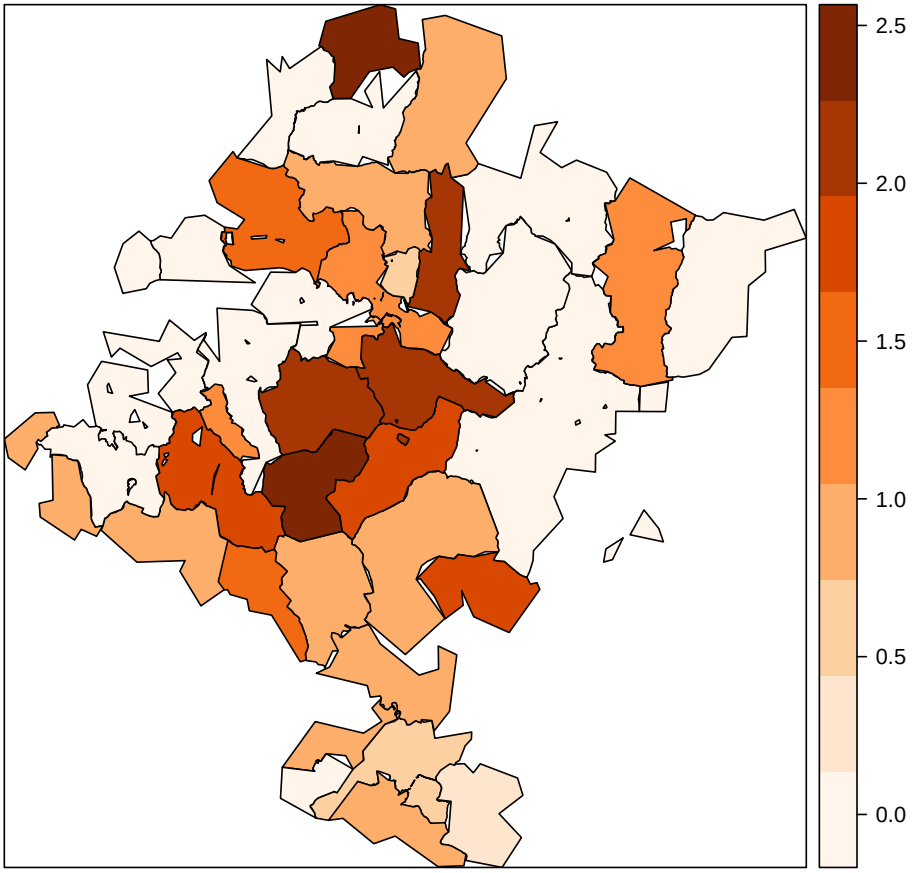

Figure 4: SMR of brain cancer in Navarre (Spain).

```
Number of Fisher Scoring iterations: 5
```

Furthermore, a quasipoisson model has been fitted in order to assess any extra-variation in the data:

```
R> nav.m0q <- glm(OBSERVED ~ offset(log(EXPECTED)) + 1,
+       data = brainnav, family = "quasipoisson")
R> summary(nav.m0q)

Call:
glm(formula = OBSERVED ~ offset(log(EXPECTED)) + 1, family = "quasipoisson",
    data = brainnav)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.5227  -1.4783  -0.3203   0.7042   1.6393
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -7.752e-06  9.703e-02       0        1

(Dispersion parameter for quasipoisson family taken to be 1.214555)

    Null deviance: 63.733  on 39  degrees of freedom
Residual deviance: 63.733  on 39  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 5
```

The dispersion parameter in the previous model seems to be higher than 1, which may mean that the Poisson distribution is not appropriate.

For this reason, and following Ugarte *et al.* (2004), a zero-inflated Poisson model has been fitted using function `zeroinfl()` from package **pscl**. Here is the resulting model:

```
R> nav.m0zip <- zeroinfl(OBSERVED ~ offset(log(EXPECTED)) +
+      1 | 1, data = brainnav, dist = "poisson", x = TRUE)
R> summary(nav.m0zip)

Call:
zeroinfl(formula = OBSERVED ~ offset(log(EXPECTED)) + 1 | 1,
    data = brainnav, dist = "poisson", x = TRUE)

Pearson residuals:
    Min      1Q  Median      3Q     Max
-1.3585 -0.9137 -0.1378  0.7137  1.8091

Count model coefficients (poisson with log link):
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  0.09347    0.09459   0.988    0.323

Zero-inflation model coefficients (binomial with logit link):
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -1.6158     0.6435  -2.511    0.012 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Number of iterations in BFGS optimization: 9
Log-likelihood: -69.08 on 2 Df
```

Hence, the zero-inflated Poisson model will be used now to detect clusters of disease. As in the example on the New York leukemia dataset, a **spacetime** object will store all the information. The column for the expected counts must be named `Expected`, and this is our first step. Note

also that, because only one time period is considered, data will have a single value and it is the 1st of January of 1990.

```
R> brainnav$Expected <- brainnav$EXPECTED
R> brainnavst <- STFDF(as(brainnav, "SpatialPolygons"),
+      xts(1, as.Date("1990-01-01")), as(brainnav, "data.frame"),
+      endTime = as.POSIXct(strptime(c("1990-01-01"), "%Y-%m-%d"),
+          tz = "GMT"))
```

Function `DetectClustersModel()` will perform the cluster detection using a `zeroinfl` model. This provides a very flexible way of handling different types of models in R for cluster detection.

```
R> nav.cl0 <- DetectClustersModel(brainnavst, coordinates(brainnav),
+      fractpop = 0.25, alpha = 0.05, typeCluster = "S",
+      R = NULL, model0 = nav.m0zip)
```

The output will show the following clusters:

```
R> nav.cl0

          x       y size      minDateCluster      maxDateCluster
31 596886.8 4710520    4 1990-01-01 01:00:00 1990-01-01 01:00:00
30 611795.5 4713762    3 1990-01-01 01:00:00 1990-01-01 01:00:00
   statistic     pvalue      risk cluster
31  2.520092 0.02476587 0.5987255    TRUE
30  2.016942 0.04459518 0.6139100    TRUE
```

As it can be seen, two clusters (with a p-value lower than 0.05) are detected. However, they overlap and we will just consider the one with the lowest p-value, which is shown in Figure 5.

An index for the areas in each of the detected cluster can be obtained with function `knbinary()`. This function will return a `data.frame` with all the dummy cluster variables, i.e., the returned `data.frame` will have as many columns as clusters and a number of rows equal to the number of areas. Entries will be 1 if the corresponding areas are in a cluster and 0 otherwise. These indices can be used for a number of analyses, such as checking whether two clusters overlap or computing the number of times an area is included in a cluster. In the following example we obtain the representation of all the clusters detected and the first one, the most significant, is added as a new column to the original `SpatialPolygonsDataFrame` to be displayed in Figure 5.

```
R> nav.clusters <- get.knclusters(brainnav, nav.cl0)
R> brainnav$CLUSTER <- ""
R> brainnav$CLUSTER[nav.clusters[[1]]] <- "CLUSTER"
R> brainnav$CLUSTER <- as.factor(brainnav$CLUSTER)
```

5. Spatio-temporal clusters

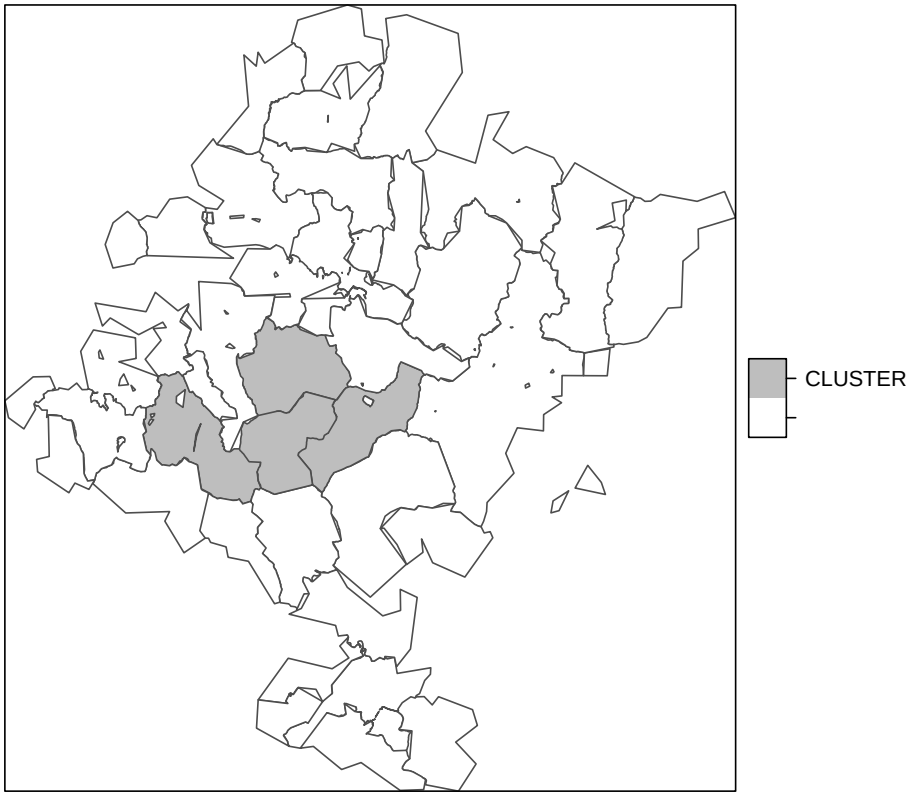

Figure 5: Cluster of brain cancer detected in Navarre (Spain).

Jung (2009) discusses how to extend model-based approaches for the detection of spatial disease clusters to space and time. Gómez-Rubio *et al.* (2015) propose the following model:

$$\log(\mu_{i,t}) = \log(E_{i,t}) + \gamma_j c_{i,t}^{(j)} \tag{3}$$

where $\mu_{i,t}$ is the mean of area i at time t and $c_{i,t}^{(j)}$ a cluster dummy variable for spatio-temporal cluster j. Random effects can also be included in equation (3) as described in Section 3 and zero-inflated distributions can also be considered as in Section 4.

Note how now data are indexed according to space and time. Dummy cluster variables are defined as in the spatial case, by considering areas in the cluster according to their distance to the cluster center, for data within a particular time period. When defining a temporal cluster, areas are aggregated using all possible temporal windows up to a predefined temporal range.

5.1. Brain cancer in New Mexico

The `brainNM` dataset (included in **DClusterm**) contains yearly cases of brain cancer in New

Mexico from 1973 to 1991 (inclusive) in a **spacetime** object. The data set has been taken from the SaTScan website and the area boundaries from the U.S. Census Bureau. In addition, the location of Los Alamos National Laboratory (LANL) has been included (from Wikipedia). Inverse distance to this site can be used to test for increased risk in the areas around the Laboratory as no other covariates are available.

```
R> data(brainNM)
```

Expected counts have been obtained using age and sex standardization over the whole period of time. Hence, yearly differences are likely to be seen when plotting the data. Standardized Mortality Ratios have been plotted in Figure 6.

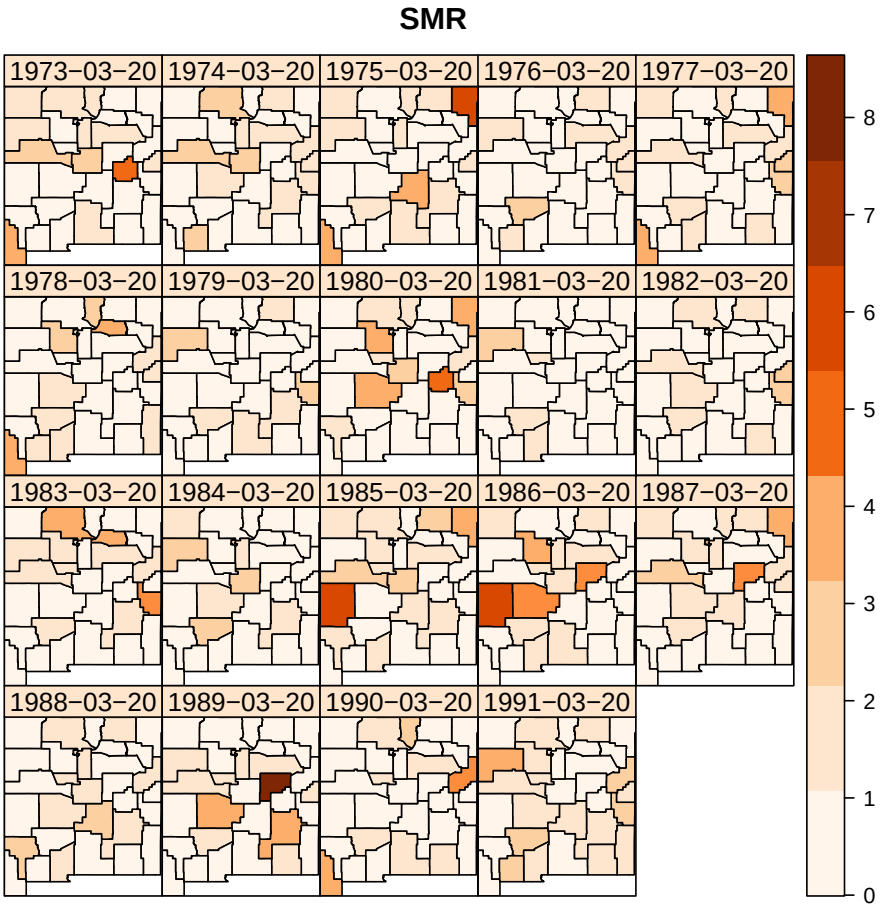

Figure 6: Standardized Mortality Ratios of brain cancer in New Mexico.

5.2. Cluster detection

Cluster detection with no covariates

Similarly as in the purely spatial case, a Poisson model with no covariates will be fitted first:

```
R> nm.m0 <- glm(Observed ~ offset(log(Expected)) + 1, family = "poisson",
+      data = brainst)
R> summary(nm.m0)

Call:
glm(formula = Observed ~ offset(log(Expected)) + 1, family = "poisson",
    data = brainst)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4874  -0.9998  -0.4339   0.3773   3.1321

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) 2.834e-16  2.917e-02       0        1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 631.64  on 607  degrees of freedom
Residual deviance: 631.64  on 607  degrees of freedom
AIC: 1585.6

Number of Fisher Scoring iterations: 5
```

Before proceeding to disease cluster detection, we have extracted the centroids of the counties in New Mexico by using function `coordinates()` from the `sp` slot in the `STIDF` object that stores the data.

```
R> NM.coords <- coordinates(brainst@sp)
```

Cluster detection with function `DetectClustersModel()` takes arguments `minDateUser` and `maxDateUser` to define the minimum and maximum times that are considered when looking for clusters. In this example the time period has been constrained to 1985–1989. Furthermore, `typeCluster = "ST"` is used to look for spatio-temporal clusters.

```
R> nm.cl0 <- DetectClustersModel(brainst, NM.coords, minDateUser = "1985-01-01",
+      maxDateUser = "1989-01-01", fractpop = 0.15, alpha = 0.05,
+      typeCluster = "ST", R = NULL, model0 = nm.m0)
```

180 possible clusters have been found this time. However, note that most of them overlap and may be different configurations of the same cluster. The first 5 are summarized below:

```
R> nm.cl0[1:5, ]

             x        y size      minDateCluster      maxDateCluster
0286 -106.3073 35.86930    3 1986-03-20 01:00:00 1988-03-20 01:00:00
```

```
0496 -105.9761 35.50684    2 1986-03-20 01:00:00 1988-03-20 01:00:00
0531 -106.9303 34.00725    9 1985-03-20 01:00:00 1986-03-20 01:00:00
0498 -105.9761 35.50684    2 1987-03-20 01:00:00 1988-03-20 01:00:00
0288 -106.3073 35.86930    2 1987-03-20 01:00:00 1988-03-20 01:00:00
     statistic       pvalue      risk cluster
0286  7.493492 0.0001082553 0.6814588    TRUE
0496  6.438221 0.0003327442 0.6970405    TRUE
0531  6.378992 0.0003544929 0.3838756    TRUE
0498  6.331113 0.0003731179 0.8070901    TRUE
0288  6.331113 0.0003731179 0.8070901    TRUE
```

Cluster detection after adjusting for covariates

In this case, we will use the inverse of the distance to LANL as a covariate as no other information about the areas is available. Distances have been computed using function `spDistsN1()` from package **sp**. Given that coordinates are expressed in longitude and latitude great circle distances are used.

```
R> dst <- spDistsN1(pts = NM.coords, pt = losalamos, longlat = TRUE)
```

Distances need to be put together in a way that values are available for all time periods. In this case, given that distances do not change over time, a vector is created by repeating the vector of distances as many times as time slots (years) we have in the dataset.

```
R> nyears <- length(unique(brainst$Year))
R> brainst$IDLANL <- rep(1/dst, nyears)
```

With all these data we are now able to fit a baseline model.

```
R> nm.m1 <- glm(Observed ~ offset(log(Expected)) + IDLANL,
+      family = "poisson", data = brainst)
R> summary(nm.m1)

Call:
glm(formula = Observed ~ offset(log(Expected)) + IDLANL, family = "poisson",
    data = brainst)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.4832  -0.9982  -0.4280   0.3775   3.1424

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.005721   0.029897  -0.191    0.848
IDLANL       0.338194   0.364900   0.927    0.354
```

```
(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 631.64  on 607  degrees of freedom
Residual deviance: 630.84  on 606  degrees of freedom
AIC: 1586.8

Number of Fisher Scoring iterations: 5
```

Note how now the included covariate is not significant. For illustrative purposes, we will still keep the covariate in our model for the cluster detection. However, non-significant covariates will have a tiny impact on the clusters detected as they will not produce a change in the expected number of cases.

```
R> nm.cl1 <- DetectClustersModel(brainst, NM.coords, fractpop = 0.15,
+      alpha = 0.05, minDateUser = "1985-01-01", maxDateUser = "1989-01-01",
+      typeCluster = "ST", R = NULL, model0 = nm.m1)
```

The number of clusters detected in this case is 179, very similar to the 180 found when no covariates where included in the model. This was expected as the included covariate was not significant. By inspecting the five most significant clusters we can observe that they are very similar to the ones detected before:

```
R> nm.cl1[1:5, ]
```

```
             x        y size      minDateCluster      maxDateCluster
0286 -106.3073 35.86930    3 1986-03-20 01:00:00 1988-03-20 01:00:00
0531 -106.9303 34.00725    9 1985-03-20 01:00:00 1986-03-20 01:00:00
0533 -106.9303 34.00725   10 1985-03-20 01:00:00 1988-03-20 01:00:00
0498 -105.9761 35.50684    2 1987-03-20 01:00:00 1988-03-20 01:00:00
0288 -106.3073 35.86930    2 1987-03-20 01:00:00 1988-03-20 01:00:00
     statistic       pvalue      risk cluster
0286  6.857043 0.0002128519 0.6487025    TRUE
0531  6.468793 0.0003220500 0.3867413    TRUE
0533  6.127863 0.0004638328 0.2581938    TRUE
0498  5.789489 0.0006670157 0.7673274    TRUE
0288  5.789489 0.0006670157 0.7673274    TRUE
```

Note how these clusters only cover a few areas, but over several years.

In order to exploit the output from `DetectClustersModel()`, function `get.stclusters()` will take the data and this output to return a list with the indices of the areas in the clusters. The next example shows how to add a new variable to `brainst` with the space-time regions in the most significant cluster, which is displayed in Figure 7.

```
R> stcl <- get.stclusters(brainst, nm.cl0)
R> brainst$CLUSTER <- ""
R> brainst$CLUSTER[stcl[[1]]] <- "CLUSTER"
```

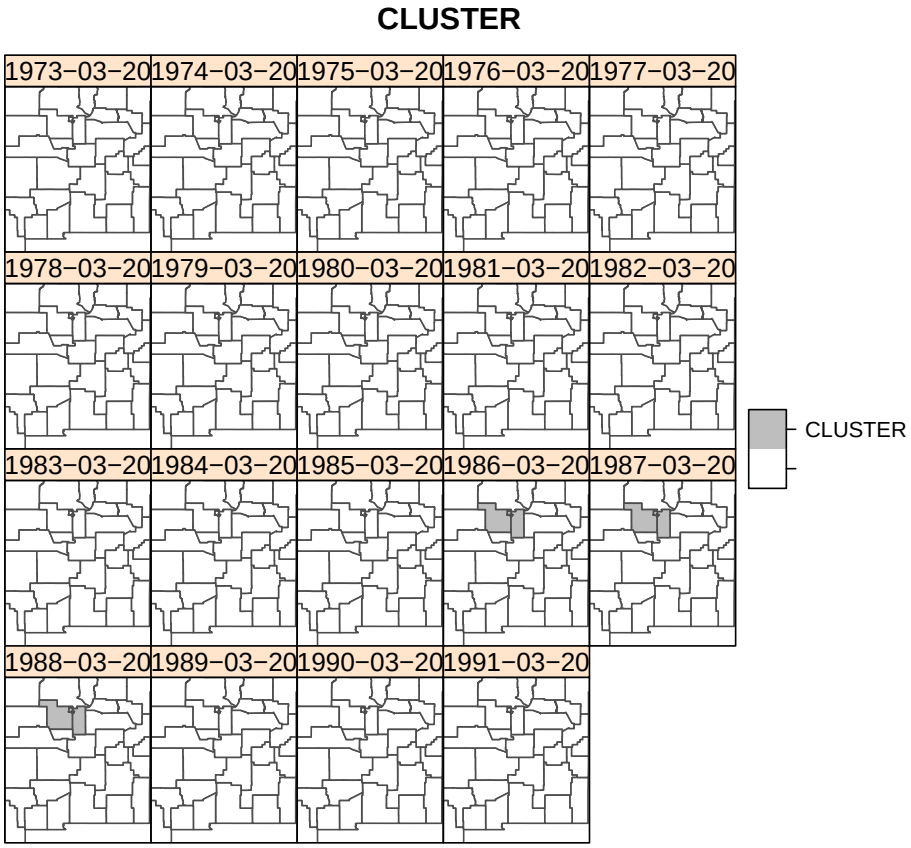

Figure 7: Most significant spatio-temporal cluster of brain cancer detected in New Mexico.

6. Discussion

In this paper we have introduced **DClusterm**, a new package for the R statistical computing software for the detection of disease clusters using a model-based approach, following recent developments by several authors. Clusters are represented by dummy variables that are introduced into a generalized linear model and different likelihoods can be used to account for different types of data. Because of this model-based approach, fixed effects (to consider relevant risk factors) and random effects (to account for other non-spatial unmeasured risk factors) can be put in the linear predictor as well.

In our examples we have considered well-known datasets to show how the functions in **DClusterm** tackle the problem of cluster detection. The results are similar to those found in relevant papers where the same datasets have been analyzed using a similar methodology. In particular, we have considered the case of the detection of clusters in space and space-time, zero-inflated data and over-dispersed data.

7. Acknowledgments

This package was initially developed by Paula Moraga as a Google Summer of Code Project. Virgilio Gómez-Rubio has been supported by grant PPIC-2014-001, funded by Consejería de Educación, Cultura y Deportes (JCCM) and FEDER, and grant MTM2016-77501-P, funded by Ministerio de Economía y Competitividad.

References

Bates D, Mächler M, Bolker B, Walker S (2015). "Fitting Linear Mixed-Effects Models Using lme4." *Journal of Statistical Software*, **67**(1), 1–48. doi:10.18637/jss.v067.i01.

Bilancia M, Demarinis G (2014). "Bayesian scanning of spatial disease rates with integrated nested Laplace approximation (INLA)." *Statistical Methods & Applications*, **23**(1), 71–94. ISSN 1618-2510. doi:10.1007/s10260-013-0241-8. URL http://dx.doi.org/10.1007/s10260-013-0241-8.

Dean CB (1992). "Testing for Overdispersion in Poisson and Binomial Regression Models." *Journal of the American Statistical Association*, **87**(418), 451–457.

Gómez-Rubio V, López-Quílez A (2010). "Statistical methods for the geographical analysis of rare diseases." *Advances in experimental medicine and biology*, **686**, 151–171.

Gómez-Rubio V, Moraga P, Molitor J (2015). "Fast Bayesian classification for disease mapping and the detection of disease clusters." *Submitted for publication*.

Jung I (2009). "A generalized linear models approach to spatial scan statistics for covariate adjustment." *Statistics in Medicine*, **28**(7), 1131–1143.

Kulldorff M (1997). "A Spatial Scan Statistic." *Communications in Statistics — Theory and Methods*, **26**(6), 1481–1496.

McCullagh P, Nelder J (1989). *Generalized linear models*. 2nd ed. edition. Chapman and Hall, London.

Ugarte MD, Ibáñez B, Militino AF (2004). "Testing for Poisson Zero Inflation in Disease Mapping." *Biometrical Journal*, **46**(5), 526–539.

Ugarte MD, Ibáñez B, Militino AF (2006). "Modelling risks in disease mapping." *Statistical Methods in Medical Research*, **15**, 21–35.

Waller L, Turnbull B, Clark L, Nasca P (1992). "Chronic disease surveillance and testing of clustering of disease and exposure: application to leukemia incidence in TCE-contamined dumpsites in upstate New York." *Environmetrics*, **3**, 281–300.

Waller LA, Gotway CA (2004). *Applied Spatial Statistics for Public Health Data*. John Wiley & Sons, Hoboken, New Jersey.

Zhang T, Lin G (2009). "Spatial scan statistics in loglinear models." *Computational Statistics and Data Analysis*, **53**(8), 2851–2858.

Affiliation:

Virgilio Gómez-Rubio
Department of Mathematics
School of Industrial Engineering
University of Castilla-La Mancha
02071 Albacete, Spain
E-mail: virgilio.gomez@uclm.es
URL: http://www.uclm.es/profesorado/vgomez

Paula Moraga
School of Mathematical Sciences
Queensland University of Technology
GPO Box 2434, Brisbane, QLD 4001, Australia
E-mail: paula.moragaserrano@qut.edu.au
URL:http://staff.qut.edu.au/staff/moragase/

John Molitor
College of Public Health and Human Sciences
Oregon State University
Corvallis, Oregon 97331, United States
E-mail: John.Molitor@oregonstate.edu
URL: http://health.oregonstate.edu/people/molitor-john

Barry Rowlingson
Lancaster Medical School
Furness Building
Lancaster University
Bailrigg, Lancaster LA1 4YG, United Kingdom
E-mail: b.rowlingson@lancaster.ac.uk
URL: http://www.lancaster.ac.uk/fhm/about-us/people/barry-rowlingson

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume VV, Issue II *Submitted:* yyyy-mm-dd
MMMMMM YYYY *Accepted:* yyyy-mm-dd