Noname manuscript No.
(will be inserted by the editor)

A software to Spatial "K"luster Analysis Through Edge Removal

Spatial and non-spatial application

Elias T. Krainski · Renato M. Assunção · Paulo J. Ribeiro Jr.

Received: date / Accepted: date

Resumo Neste artigo nós apresentamos a implementação em R da Spatial "K"luster Analisys Through Edge Removal ...

Keywords Regionalization · Cluster Analysis · Minimum Spanning Tree

1 Introdução

This work is motivated by the problem of grouping n points of a time series or n geografical areas by a clustering method which acknowledges the ordering and neighbouhood structure of such type of data.

Standard cluster analysis uses observations y_i, $i = 1, 2, ..., n$, from a set or sample of n individuais spliting them on $k \leq n$ groups according to some criteria to constitute the groups and assuming independence between the individuals and methods are implemented in software as, for instance, the **cluster** package (Maechler, Rousseeuw, Struyf & Hubert 2005) for the R software (CITAR O R). The possible number and composition of the clusters from n individuais is tipically large and the cluster methods are motivated by the ideia of having greater homogeneity within groups and heterogeneity and distinction between groups.

Agradecimentos: Grants or other notes about the article that should go on the front page should be placed here. General acknowledgments should be placed at the end of the article.

Elias T. Krainski
Universidade Federal de Minas Gerais - UFMG, Laboratório de Estatística Espacial - LESTE, Av. Pres. Antônio Carlos, 6627 Belo Horizonte - MG / Brasil CEP: 31270-901
Tel.: +55-31-3409-5917
Fax: +55-31-3409-5924
E-mail: eliaskrainski@yahoo.com.br *Present address:* of F. Author

Renato M. Assunção
Universidade Federal de Minas Gerais - UFMG, Departamento de Estatística, Av. Pres. Antônio Carlos, 6627 Belo Horizonte - MG / Brasil CEP: 31270-901
Tel.: +55-31-3409-5917
Fax: +55-31-3409-5924

Paulo J. Ribeiro Jr.
Universidade Federal do Paraná - UFPR, Departamento de Estatística, Setor de Ciências Exatas, Universidade Federal do Paraná, Curitiba, PR - Brasil 81.531 - 990
Tel.: +55-41-3361-3573
Fax: +55-41-3361-3141

MAIS REVISÃO NECESSÁRIA...

When grouping temporal or spatial data an additional contiguity restriction that the group members should be *neighboors* may be imposed for practical purposes of handling the groups, The restriction reduces quite dramatically options for the number and composition of the groups and is implemented by the SKATER (Spatial 'K'luster Analysis by Tree Edge Removal) method ((Assunção & Reis 2002), (Assunção, Neves, Câmara & Freitas 2006)).

A general computational implementation allowing for clustering with spatial or temporal data under contiguity restrictions is presented and possible extensions not confined to such data formats are discussed. The SKATER implementation is a two steps procedure: the first imposes restrictions defining a unique primary group through a *minimum spanning tree* conecting all the units by a graph which will be splited generating further groups in the the second step by assessing the homogeineity within and heterogeneity between groups.

The implementation expands the original SKATER (REFS???) proposal by (i) considering measures based on densities when obtaining the *minimum spanning tree*; (ii) using likelihood based measures to determine how groups are obtained by splitting original groups; (iii) applying the methods for point process, geostatistical data and other structures without temporal or spatial indexing.

The next Section presents the SKATER and illustrate the usage of the function `skater()`, contributed to the R package **spdep**. Density and likelihood based measures and presented in Section 3 with an application to count data considering the Poisson distribution. Section 4 presets applications to point process data, time series data and data without spatial indexing.

2 O método SKATER

SKATER - Spatial 'K'luster Analisys Through Edge Removal - is a proposal cluster analysis of spatial data (Assunção & Reis 2002, Assunção et al. 2006). The methods consists of two mais steps: the obtention of a *minumum spanning tree* given by a graph conecting the area and the posterior removal of "stems" of the tree, or edges of the graph, generating the groups.

Consider a spatial region $\boldsymbol{D}$ partitioned in areas $v_1, ..., v_n$, with $v_i \cup v_j = \emptyset$ for $i \neq j$. Consider also the data vector $\boldsymbol{y} = \{y_1, y_2, ..., y_n\}$, with y_i the data measured at the area v_i. We aim at define k groups, $g_1, ..., g_k$ with $\bigcup_{i=0}^{k} g_i = \boldsymbol{D}$ and $g_i = \bigcup_{j \in I_i} v_j$ and $I_i \in \{1, 2, ..., n\}$ is a set of indexes of the i^{th} group.

Consider an adjacency matrix $\boldsymbol{A}$ of dimension n *timesn* related to the neighbourhood structure of the areas within $\boldsymbol{D}$, e.g. each row of $\boldsymbol{A}$ is binary 0/1 vector related to an area the with non null elements indicating the columns of the neighbouring areas. In other words $\boldsymbol{A}$ defines a graph $\boldsymbol{G}_D$ of the neighboring structure within region the $\boldsymbol{D}$. The graph $\boldsymbol{G}_D$ is defined by the set of nodes $\boldsymbol{V}_D$ associated with the areas $v_1, v_2, ..., v_n$, and a set of edges $\boldsymbol{E}_D$ specified by the adjacency structure. An edge (i, j) exists if the node (area) i is neighbour of the node (area) j.

2.1 Minimum sppaning tree

The *minimum spanning tree*-(MST), of the n nodes is a connected graph Υ with n nodes and $n-1$ edges. With a connected graph Υ allows, starting from v_i, reaching any other v_j by the $n-1$ edges of Υ. Further, if an edge is removed Υ is divided in two unconnectd graphs which one being a MST. Form the set of observations $y_1, y_2, ..., y_n$ is is assigned for each possible edge (i, j) between a pair of nodes an associated "cost" $d(i, j) = d_{ij}$ given by a measure of dissimilarity between nodes v_i e v_j. The MST is the set of edges $e_1, e_2, ..., e_{n-1}$ which minimises an overall "cost" of the graph such as $\sum_{j=1}^{n-1} d(e_j)$.

The MST can be obtained by the Prim's algorithm (Prim 1957) which operates as follows. Start with an empty set of nodes $\boldsymbol{V}_{in} = \emptyset$ and an empty set of edges $\Upsilon = \emptyset$. Any i^{th} node from the

set $1, 2, \ldots, n$ can be selected and included in $\boldsymbol{V}_{in}$. Then iterate between the following steps until all nodes are included in $\boldsymbol{V}_{in}$:

1. compute the dissimilarity between nodes in $\boldsymbol{V}_{in}$ and neighbours not included in $\boldsymbol{V}_{in}$;
2. identify the lowest dissimilarity pair (v_i, v_j), $v_i \in \boldsymbol{V}_{in}$ and $v_j \notin \boldsymbol{V}_{in}$;
3. add v_j to $\boldsymbol{V}_{in}$ and (i, j) to Υ.

The set of edges in Υ is the MST and Υ is unique if and only if the dissimilarity measure is continuous and $d_{ij} \neq d_{rs}$ for any pairs (i, j) and (r, s), $(i, j) \neq (r, s)$. (REF HERE???)

2.2 The edge removal

The next step is the sequential removal from Υ. For each removed edge the current MST generate two groups with individuals also connected by a MST. For choice of edge the simply removal of the costly $(rmmax(d_{ij}))$ edge is not suitable since this is a local measure of dissimilarity. Assunção et al. (2006) favors a choice which maximises the homogeineity within elements within the resulting groups. For intance, consider the nodes v_1, v_2, v_3 e v_4 and edges $\{(1, 2), (2, 3), (3, 4)\}$, with associatd costs $d_{12} < d_{2,3} < d_{3,4}$. Suppose v_2 is very similar to v_4, and v_3 more similar to v_2. Assuming a measure of homogeneity within groups it is possible to find a smaller total between the groups given by the removal of lesser costly edge $(1, 2)$ than the total resulting of the removal of $(3, 4)$.

Define $H(g_i)$ the homogeneity within the group g_i, H_0 the primary homogeneity for all nodes within a single group, and $H_k = \sum_{i=1}^{k} H(g_i)$ the total homogeneity for the k groups at a particular partitioning step of the algorithm. Initially consider all the n nodes within a unique group, the MST Υ and compute H_0. Defina o vetor gv de tamanho n para identificação de grupos a que cada nó pertence, inicialmente todos são iguais a 1. Defina o vetor ge de tamanho $n-1$ para identificação de grupos a que cada aresta da MST pertence, inicialmente todos são iguais a 1. Defina um vetor dw de tamanho $n - 1$. Cada elemento deste vetor armazena a diferença de homogeneidade obtida se a aresta correspondente for removida. Essa diferença é entre a homogeneidade atual e a soma de homogeneidade dos grupos gerados pela remoção da aresta.

Inicialmente, preencha o vetor dw, removendo temporariamente cada aresta. Ordene a MST de acordo com dw em ordem decrescente. Então corte definitivamente a primeira aresta da MST ordenada. Atualize gv, por exemplo, colocando 2 nas posição de gv referente ao nó da primeira coordenada da aresta removida e nas posições referentes aos nós ainda ligados a este nó. Atualize ge, por exemplo, colocando 2 nas posições referentes 'as arestas que contém nós ainda conectados ao nó da primeira coordenada da aresta removida. Recalcule dw e ordene-o em ordem decrescente, colocando nesta mesma ordem as arestas existentes e ge.

Prossiga na remoção de arestas até que um critério de parada seja atingido, com os seguintes passos:

1. corte a primeira aresta
2. atualize o vetores gv e ge
3. recalcule dw nas posições correspondentes 'a arestas que fazem parte dos novos dois grupos gerados
4. ordene dw em ordem decrescente e também ge e o conjunto de arestas existentes nesta mesma ordem
5. se o criterio de parada não foi atingido volte ao passo 1.

É possível também estabelecer uma restrição aos grupos formados, por exemplo, um número mínimo de nós em cada grupo ou de população em cada grupo. Neste caso, na atualização de dw, deve-se verificar que ao remover a aresta, os grupos gerados satisfazem a restrição. Se não satisfazem, fazer dw igual a zero. Suponha que a restrição é que o número de nós em cada grupo seja no mínimo 3 e temos um grupo candidato com 10 nós. Se ao dividi-lo o resultado seja um grupo com 8 nós e outro com 2 nós. Neste caso, fazemos dw correspondente ser igual a zero, embora a diferença de homogeneidade não necessariamente seja igual a zero.

2.3 Application to multivariate continuous data

Nesta seção nós mostramos rapidamente como utilizar a função `skater()` para fazer uma análise de agrupamentos com restrição espacial dos dados de violent crime rates by US State. Estes dados estão disponíveis no R, (R Development Core Team 2008). Nós consideramos a análise de 48 estados continentais, excluindo Alaska e Hawaii, e o conjunto de três tipos de crimes: murder, assault and rape.

```
> dpad <- scale(dsel <- USArrests[-c(2, 11), c(1, 2, 4)])
```

Nós utilizamos o mapa dos EUA disponível no pacote **maps**, (code by Richard A. Becker & version by Ray Brownrigg Enhancements by Thomas P Minka <surname@stat.cmu.edu> 2009) e utilizamos a função `map2SpatialPolygons()` para converter o mapa para o formato do pacote **sp** e utilizar a função `map2nb()` para encontrar a lista de vizinhança por contiguidade.

```
> require(maps)
> usa.m <- map("state", fill = TRUE, col = "transparent", plot = FALSE)
> require(spdep)
> IDs <- sapply(strsplit(usa.m$names, ":"), function(x) x[1])
> usa.pl <- map2SpatialPolygons(usa.m, IDs = IDs)
> usa48.nb <- subset(poly2nb(usa.pl), c(rep(T, 7), F, rep(T, 41)))
```

A análise inicia com o cálculo de custos das arestas indicadoras de vizinhança entre os estados. O cálculo de custos é feito com a função `nbcosts()`. Há várias funções de distância implementadas e é possível utilizar uma função definida pelo usuário. Nós vamos utilizar a default, que é a distância Euclidiana. Escolhendo esta distância, é melhor considerar os dados padronizados.

```
> nbc.e <- nbcosts(usa48.nb, dpad)
```

A seguir, nós juntamos a lista de custos com a lista de vizinhança, criando um objeto da classe `listw`.

```
> nbw.e <- nb2listw(usa48.nb, nbc.e, style = "B")
```

A partir da lista de vizinhança ponderada, podemos encontrar a árvore geradora mínima (AGM) usando a função `mstree()`. Esta função implementa o algoritmo de Prin (Prim 1957). Esta função tem dois argumentos, o primeiro deve ser um objeto `listw` e o segundo um inteiro indicando o nó inicial. Este argumento pode ser `NULL` e neste caso será sorteado aleatóriamente um nó inicial.

```
> mst.e <- mstree(nbw.e)
```

O resultado da função `mstree()` é um objeto da classe `mst`. Este objeto é uma matriz de três colunas, cada linha contendo os nós do vértice e o custo.

A partir da AGM, nós utilizamos a função `skater()` para fazer podar as arestas e gerar os grupos de áreas homogêneas. Na função `skater()` podemos entar com a AGM e realizar $k-1$ cortes de arestas, gerando k grupos. Nesta função podemos utilizar várias funções de homogeneidade, inclusive alguma definida pelo usuário. Nós utilizamos a distância Euclidiana, a default. Gerando inicialmente 4 grupos

```
> res.e4 <- skater(mst.e[, 1:2], dpad, 3)
```

Essa função também pode receber como input um objeto da classe `skater` e fazer podas adicionais. Nós geramos 5, 6 e 7 grupos usando essa funcionalidade:

```
> res.e5 <- skater(res.e4, dpad, 1)
> res.e6 <- skater(res.e5, dpad, 1)
> res.e7 <- skater(res.e6, dpad, 1)
```

Na figura **??** nós plotamos o mapa dos EUA considerando 4, 5, 6 e 7 grupos. As cores são tons de cinza proporcionais a soma das médias em cada grupo das três taxas de crimes. O grafo sobreposto é a AGM gerada e dessa forma podemos observar quais arestas foram podadas.

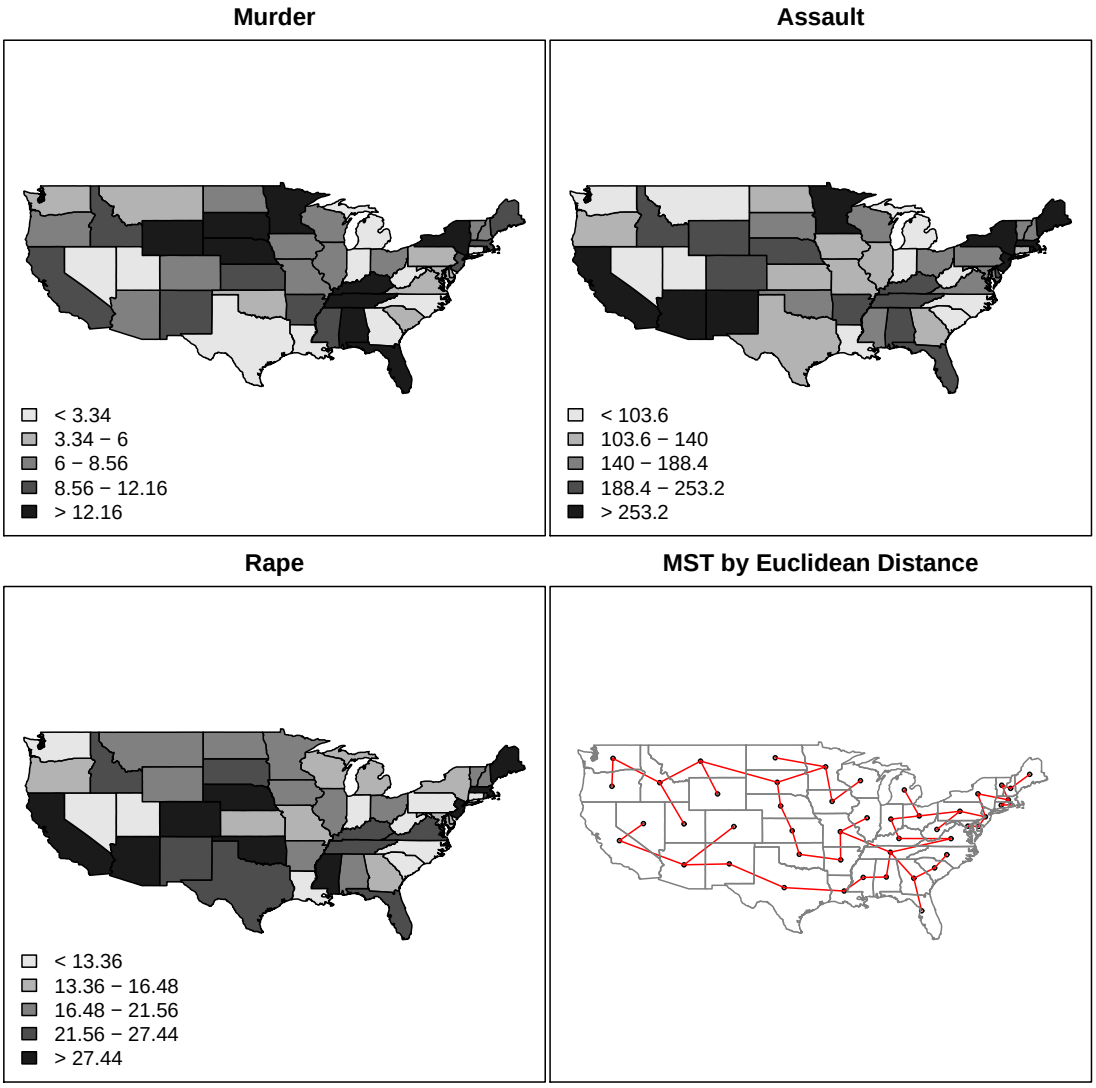

Figura 1 Nesta figura nós plotamos um mapa para cada taxa de crimes em tons de cinza proporcionais à taxa de crime e a AGM gerada considerando a distância Euclidiana

3 SKATER baseada em densidade

Nesta seção nós propomos o uso de medidas baseadas em modelos probabilisticos para ambos os passos do SKATER. Na construção da árvore geradora mínima propomos uma medida de distáncia baseada na função de densidade da variável aleatória, uma medida de plausibilidade. No passo de remoção de arestas, propomos o uso da verossimilhança como medida de homogeneidade dos grupos. Ambas as propostas fazem possível a extensão da metodologia para várias direções. Agora nós podemos considerar dados de contagem, por exemplo, considerando a distribuição de Poisson na medida de plausibilidade. Também podemos testar a cada passo da remoção de arestas, a significância da divisão de um grupo.

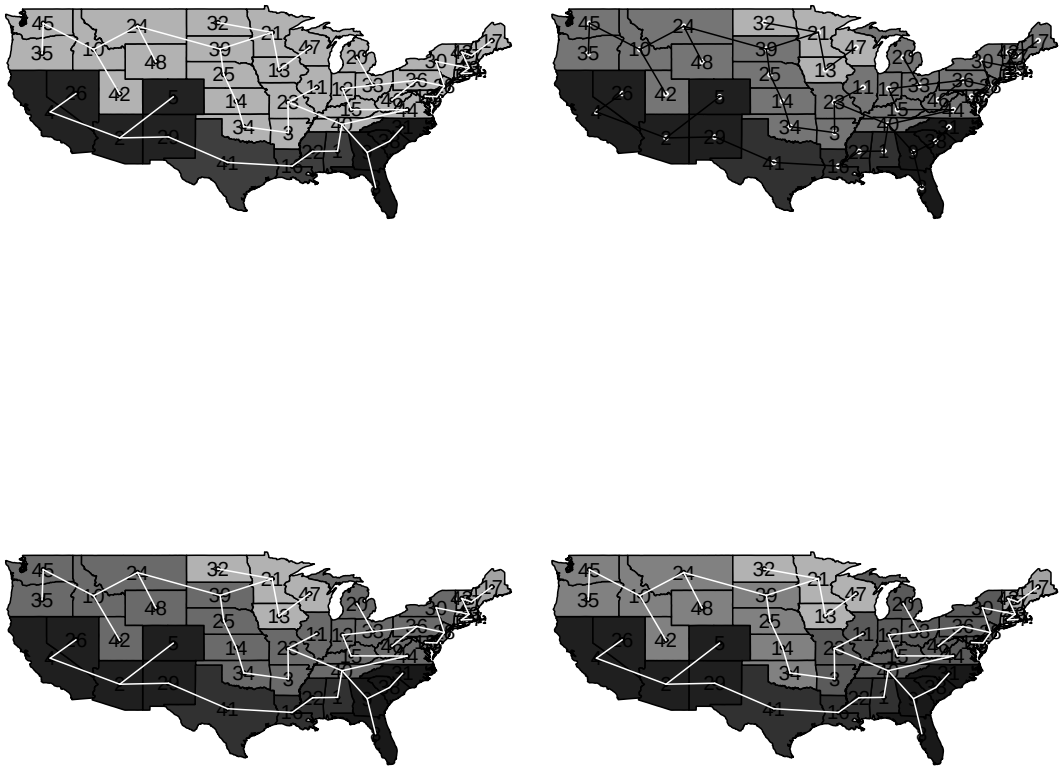

Figura 2 Results by SKATER to obtain four to seven groups using Euclidean distance.

3.1 Distância baseada em distribuição de probabilidade

Nós consideramos que Y é uma variável aleatória com densidade de probabilidade ou função de probabilidade $p(y/\theta, \phi)$ qualquer, com $\int_{-\infty}^{\infty} p(y/\theta, \phi)dy = 1$, onde θ é o parâmetro de locação e ϕ é o parâmetro de dispersão. Valores próximos de y tem valores parecidos de $p(y/\theta, \phi)$ e valores de y muito distintos tem também valores distintos de $p(y/\theta, \phi)$, com θ fixo e igual para todo y.

Agora, vamos considerar a situação em que θ não é igual para todo y. Nesta situação, podemos ter variáveis preditoras X e $\theta = g(X)$. Logo a densidade de y passa a ser $p(y/X, \theta, \phi)$. Mesmo que X seja apenas um offset, um preditor com associação conhecida, θ pode variar para diferentes valores de y. Nosso interesse é utilizar $p(y/X, \theta, \phi)$ para medir distância entre duas observações y_i e y_j. Mas, para algumas distribuições de probabilidade, tais como Bernoulli e Poisson, a variância de y depende de θ, ou seja, depende de X. Portanto, não podemos usar apenas $p(y/X, \theta, \phi)$ para

comparar diferentes valores de y quando temos tais distribuições de probabilidade, precisamos de uma padronização de $p(y/X, \theta, \phi)$.

Na inferência estatística, com X e y conhecidos e fixados, a comparação de diferentes valores de θ e ϕ é feita via função de verossimilhança, $L(\theta, \phi/y, X) = p(y/X, \theta, phi) \in [0, +\infty)$. Mas a verossimilhança calculada em diferentes conjuntos de dados, $L(\theta, \phi/y, X)$ e $L\theta, \phi/y', X')$, não são comparáveis. Nesse contexto, uma função que poderia ser utilizada é a função de plausibilidade, proposta por Shafer (1976), que é uma função derivada da função de verossimilhança, . Derivando-a de $L(\theta, \phi/X, y)$, temos que

$$pl_{\theta,\phi}(\theta, \phi/X, y) = \frac{L(\theta, \phi/X, y)}{max_{\theta',\phi'}[L(\theta', \phi'/X, y)]} , \tag{1}$$

com $pl_{\theta,\phi}(\theta, \phi/X, y) \in [0, 1]$, $\forall y$, $\forall X$, $\forall \theta$, $\forall \phi$.

Usando a mesma idéia, nós vamos definir uma medida de plausibilidade para y fazendo

$$pl_y(y/X, \theta, \phi) = \frac{p(y/X, \theta, \phi)}{max_{y' \in \Re}[p(y/X, \theta, \phi]} , \tag{2}$$

com $pl_y(y/X, \theta) \in [0, 1]$, $\forall y$, $\forall X$, $\forall \theta$ e $\forall \phi$. Aqui nós avaliamos $p(y/X, \theta, \phi)$ considerando $y \in \Re$ para encontrar $max_{y' \in \Re}[p(y/X, \theta, \phi]$, embora y possa ser uma variável aleatória estritamente positiva ou discreta. Nós vamos tratar de casos particulares e mostrar que em alguns casos precisamos considerar $y \in \Re$ para encontrar $max_y p(y/X, \theta)$ mesmo quando y não é avaliado em todo o domínio de $\Re$.

Agora, vamos definir uma medida de distância entre duas observações baseada em $p(y/X, \theta, \phi)$. Seja y_i e y_j duas observações da variável aleatória Y e seja $p(y/X, \theta, \phi)$ sua função de densidade ou função de probabilidade. Se $Var(y)$ é independente de θ, então

$$d_{ij} = log\{[p(y_i/X_i, \theta, \phi)p(y_j/X_j, \theta, \phi)]^{-1}\} . \tag{3}$$

Se $Var(y)$ depende de θ, então

$$d_{ij} = log\{[pl_y(y_i/X_i, \theta, \phi)pl_y(y_j/X_j, \theta, \phi)]^{-1}\} . \tag{4}$$

A seguir, nós estudaremos $d(i, j)$ para alguns casos particulares de $p(y/X, \theta, \phi)$ e sobre a estimação de θ e ϕ.

3.1.1 Caso Gaussiano

Nesta seção vamos considerar que y tem distribuição Normal e que não temos covariáveis X. Primeiro, vamos considerar o caso em que temos grupos com médias diferentes. Consideramos que temos uma variável aleatória y com $y_i \sim N(\mu_i, \sigma^2)$, ou seja, a variância é constante mas a média não o é. Temos então que

$$p(y_i/\mu_i, \sigma^2) = (2\pi\sigma^2)^{1/2} exp\{-\frac{(y_i - \mu_i)^2}{2\sigma^2}\}$$

e

$$d(i, j) = log(2\pi\sigma^2) + \frac{1}{2\sigma^2}[(y_i - \mu_i)^2 + (y_j - \mu_j)^2] .$$

Vamos estimar μ_i e μ_j por $(y_i + y_j)/2$. Neste caso,

$$d(i, j) = log(2\pi) + (y_i - y_j)^2$$

é proporcional à distância euclidiana entre y_i e y_j.

Agora vamos considerar que a média e a variância não são iguais para todos os grupos. Neste caso, consideramos que $y_i \sim N(\mu_i, \sigma_i^2)$. Então

$$p(y_i/\mu_i, \sigma_i^2) = (2\pi\sigma_i^2)^{1/2} exp\{-\frac{(\mu_i\beta)^2}{2\sigma_i^2}\}$$

e

$$d(i,j) = log(2\pi\sigma_i^2) + \frac{1}{2\sigma_i^2}[(y_i - \mu_i)^2 + (y_j - \mu_j)^2] \, .$$

Agora, vamos estimar μ_i e μ_j por $\hat{y}_{ij} = (y_i + y_j)/2$ e σ_i^2 e σ_j^2 por $\hat{\sigma}_{ij}^2 = (y_i - \hat{y}_{ij})^2 + (y_j - \hat{y}_{ij})^2$. Obtemos que e $\hat{\sigma}_{ij}^2 = (y_i - y_j)^2/2$ e que

$$d(i,j) = log(\pi(y_i - y_j)^2)$$

, que é proporcional à distância euclidiana entre y_i e y_j.

Outra situação menos frequente é quando temos a situação onde a média é igual para todas as observações mas a variância é diferente entre os grupos. Neste caso, consideramos que $\mu = 0$ e temos $y_i \sim N(0, \sigma_i^2)$. Então

$$p(y_i/\sigma_i^2) = (2\pi\sigma_i^2)^{1/2} exp\{-\frac{(y_i)^2}{2\sigma_i^2}\}$$

e

$$d(i,j) = \frac{1}{2}[log(2\pi\sigma_i^2) + log(2\pi\sigma_j^2)] + \frac{y_i^2}{2\sigma_i^2} + \frac{y_j^2}{2\sigma_j^2}$$

Agora, vamos estimar σ_i^2 e σ_j^2 por $\hat{\sigma}_{ij}^2 = (y_i^2 + y_j^2)/2$. Obtemos que

$$d(i,j) = log(\pi(y_i - y_j)^2) + 1 \, ,$$

que é proporcional à distância euclidiana de y_i e y_j até a média de y.

3.1.2 Caso Poisson

Se y tem distribuição de Poisson, a média e a variância de y são iguais e são medidas por um único parâmetro, sendo um parâmetro tanto de locação quanto de escala. Neste caso, na análise de cluster nós temos grupos com média e variância diferentes. Portanto, para medir a distância entre y_i e y_j precisamos utilizar pl_y, pois se y_i e y_j pertencem a grupos diferentes, não podemos comparar essa distância com a distância entre y_i e y_k se k percente a um grupo diferente de j.

Outra situação é quando temos y_i e y_j no mesmo grupo, portanto taxa de ocorrências igual, mas i tem população (ou tempo, tamanho, etc.) diferente de j. Por exemplo, y_i é o número de casos de uma doença na área i e y_j é o número de casos da mesma doença na área j. y_i pode ser muito diferente que y_j apenas porque a população sob risco nessas áreas é muito diferente, mesmo que a taxa de ocorrências de ambas seja igual. Neste caso, a população, ou o número esperado de casos, é um termo que afeta o número de ocorrências de forma deterministica conhecida, sendo chamado de offset.

Em ambas as situações com y_i diferente de y_j, por serem de grupos diferentes ou por terem populações diferentes, nós precisamos considerar uma medida padronizada de distância. Neste caso, sugerimos pl_y. Nós vamos considerar que e_i é o número esperado de casos em i, podendo ser uma constante ou ser calculado em função de sua população (ou tempo, tamanho, etc.) e que $\lambda_i = \theta_i \times e_i$, fazendo $y_i \sim Poisson(\lambda_i)$, ou seja,

$$p(y_i/\theta_i, e_i) = p(y_i/\lambda_i) = \lambda_i^{y_i} e^{-\lambda_i} / \prod y_i \, .$$

Agora, vamos obter $max_{y'}\{p(y'/\lambda_i)\}$ e fazer a padronização de $p(y_i/\lambda_i)$, obtendo pl_y. Então precisamos encontrar

$$max_{y'}\{\lambda_i^{y'} e^{-\lambda_i}/(y'!)\} .$$

Derivando em relação a y' e igualando a zero não obtemos uma expressão fechada. Mas podemos ver na figura 4 que o máximo é zero se $\lambda_i < 0.5$ e se aproxima de $\lambda_i - 0.5$ se λ_i cresce.

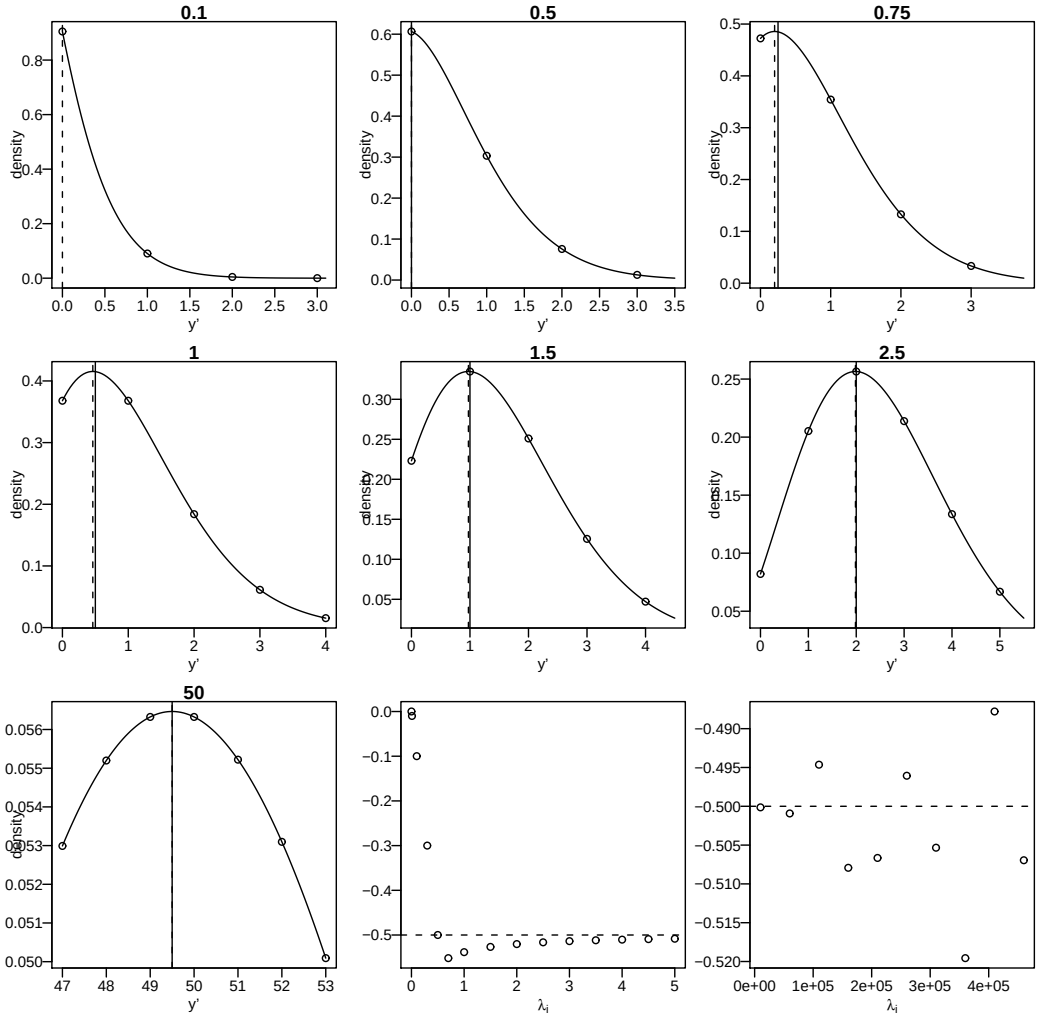

Figura 3 Nos 7 primeiros gráficos (três superiores, três ao meio e um abaixo), temos a densidade da função de probabilidade de Poisson para alguns valores de λ. A linha contínua vertical corresponde ao $\lambda - 0.5$ e a linha tracejada está no valor que tem a maior probabilidade. Os dois últimos gráficos mostram a diferença entre λ e o valor que maximiza a função de probabilidade de Poisson, para valores pequenos de λ (gráfico na coluna central) e para valores grandes de λ (gráfico na coluna à direita).

3.2 Homogeneidade baseada em verossimilhança

Agora, nós definimos uma medida de homogeneidade para os grupos baseada na verossimilhança. Vamos considerar o caso em que temos uma função de probabilidade $p(y/\theta, \phi)$ qualquer. Nós

definimos

$$H_i = -\sum_j log(p(y_j/\hat{\theta}_i, \hat{\phi}_i))$$

a homogeneidade do grupo i, com j os elementos do grupo i, $\hat{\theta}_i$ a estimativa de θ com as observações do grupo i e $\hat{\phi}_i$ a estimativa de ϕ com as observações do grupo i. Neste caso, temos que

$$\sum_{i=1}^{k} H_i$$

é a o negativo do logaritmo da verossimilhança considerando k grupos.

Com essa definição, podemos definir

$$D_k = H_k - H_{k-1}$$

a razão de verossimilhança. Sob hipótese de que não é necessário k grupos, mas apenas $k-1$ grupos, D_k tem distribuição assintótica χ^2_p, onde p é o número de parâmetros de $p(y/\theta, \phi)$. Esse fato pode ser utilizado como um critério de parada da divisão sucessiva de grupos no passo 2 do procedimento SKATER.

3.3 Exemplo com dados de Poisson

Muitos estudos tem por objetivo estudar e dados de ocorrência de doenças em distritos administrativos. A análise de agrupamentos é uma técnica que pode ser útil para definir políticas de controle de doenças. Nesta seção, nós exemplificamos a aplicação da função `skater()` para dados de Poisson utilizando uma medida de distância não implementada.

Para isso, nós definimos duas medidas baseadas em densidade de dissimilaridade para gerar a MST. Também definimos uma medida de homogeneidade para o segundo passo.

3.3.1 Duas medidas baseadas em densidade no caso poisson

Seja $f(\boldsymbol{y}_i|\hat{\theta_J}) = f(i|J)$ a densidade de $\boldsymbol{y}_i$ considerando $\hat{\theta_J}$ é a estimativa dos parâmetros da função de densidade usando as observações $\boldsymbol{y}_J$, em que J é um conjunto de índices. Nós consideramos duas funções de dissimilaridade baseadas em $f(.|.)$. Suponha que temos as áreas i e j, então temos $f(j|i)$ e $f(j|i,j)$. A primeira avalia a densidade de $\boldsymbol{y}_j$ usando $\hat{\theta}_i$ e a segunda usa $\hat{\theta_{i,j}}$. A primeira é uma densidade condicional e a segunda é conjunta. Porém, se nós temos observações nas áreas r e s, r vizinha de s, como comparar $f(j|i)$ com $f(r|s)$? Para que essas medidas sejam comparáveis, nós fazemos algumas considerações e definimos medidas padronizadas. Consideramos que $f(i|i)$ maximiza $f(.|i)$. Se nós temos a área i e esta área tem dois vizinhos, as áreas j e k. Suponha que y_i e y_j tem parâmetros iguais e y_k tem parâmetros diferentes. Nós esperamos que $f(j|i) > f(k|i)$ e $f(j|i,j) > f(j|i,k)$.

Seja

$$d_{i|j} = 1 - \frac{f(i|j)}{max(f(.|j))}) \,, \tag{5}$$

a densidade padronizada condicional ou medida *condicional dissimilarity* - (CD) e seja

$$d_{i,j} = 1 - \frac{f(i|i,j)f(j|i,j)}{2max(f(.|i,j))}) \,, \tag{6}$$

a densidade padronizada conjunta ou medida *joint dissimilarity* - (JD). Essa padronização faz com $d_{i,j} \in (0,1)$ e $d_{i|j} \in (0,1)$. $d_{i,j} = 0$ se $f(i|j) = f(j|j)$ e $d_{i,j} = 1$ se $f(i|j) = 0$. O mesmo para $d_{i|j}$.

Agora, vamos obter $max\{f(.|i,j)\}$ e $max\{f(.|j)\}$ e fazer a padronização em $d_{i|j}$. Então precisamos encontrar

$$max_{y'}\{\lambda^{y'}e^{-\lambda}/(y'!)\ .$$

Derivando em relação a y' e igualando a zero não obtemos uma expressão fechada. Mas podemos ver na figura 4 que o máximo é zero se $\lambda_i < 0.5$ e se aproxima de $\lambda_i - 0.5$ se λ_i cresce.

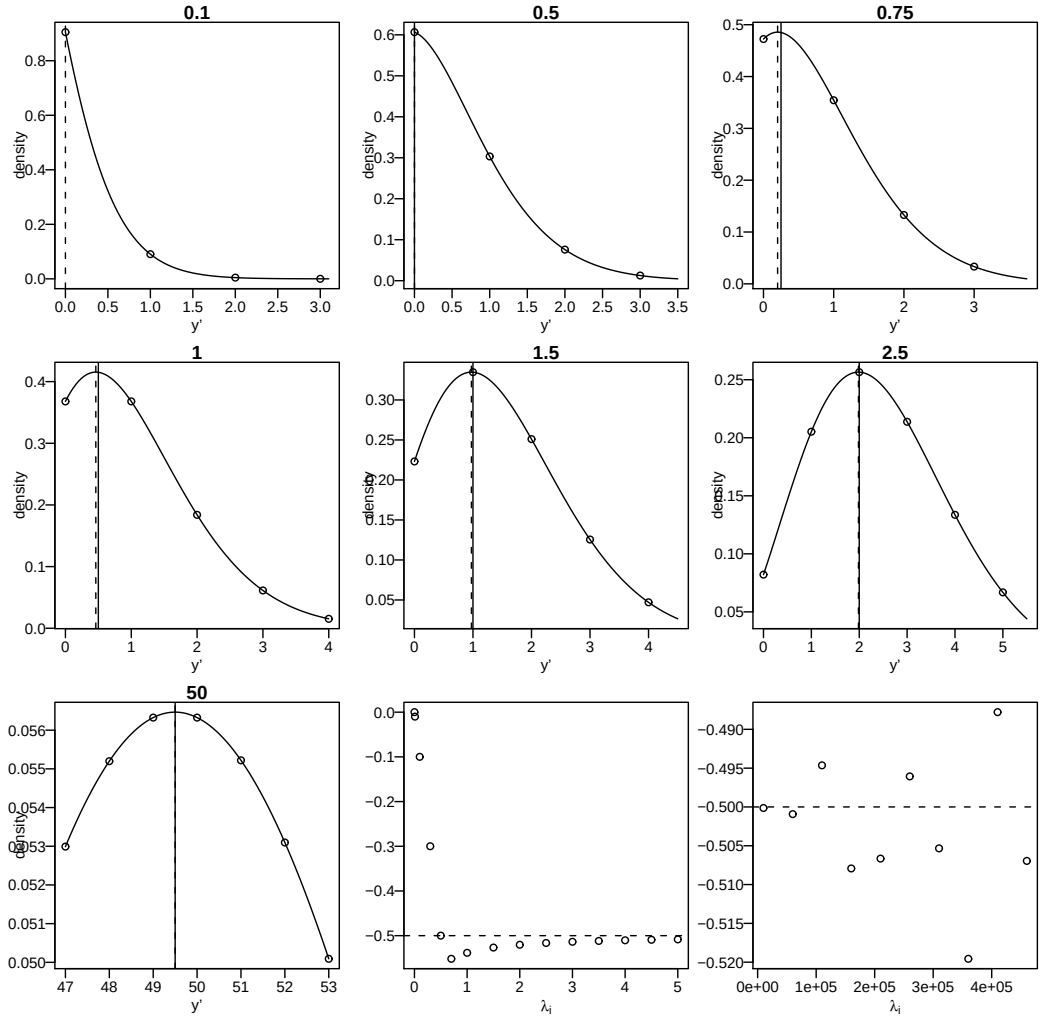

Figura 4 Nos 7 primeiros gráficos (três superiores, três ao meio e um abaixo), temos a densidade da função de probabilidade de Poisson para alguns valores de λ. A linha contínua vertical corresponde ao $\lambda - 0.5$ e a linha tracejada está no valor que tem a maior probabilidade. Os dois últimos gráficos mostram a diferença entre λ e o valor que maximiza a função de probabilidade de Poisson, para valores pequenos de λ (gráfico na coluna central) e para valores grandes de λ (gráfico na coluna à direita).

Agora, nós definimos uma medida de homogeneidade para os grupos. Vamos considerar o caso em que temos uma função de densidade $f(i|I)$ a ser avaliada em $\boldsymbol{y}_i$ o vetor de observações da área i com parâmetros estimados a partir das observações das áreas I, $\boldsymbol{y}_I$, e I é um conjunto de índices. Considereamos também que $l(i|I) = log(f(i|I))$. Suponha que o grupo j contém as áreas I. Seja $\hat{\theta_J}$

os parâmetros estimados com as observações $\boldsymbol{y}_J$. Seja

$$H(j) = \sum_{i \in I} -l(i|I) \; . \tag{7}$$

Essa medida é uma função de deviance. Lembramos que se nós dividimos o grupo j em dois grupos, j' e j'', então $H(j') + H(j'') < H(j)$. Além disso, essa diferença de deviance tem distribuição assintótica χ^2_p, onde p é o número de variáveis em $\boldsymbol{y}$. Esse fato pode ser utilizado como um critério de parada da divisão sucessiva de grupos no passo 2 do procedimento SKATER.

3.4 Exemplo

Nesta seção mostramos como utilizar a função `skater()` para fazer a análise de agrupamentos quando tempos uma variável de contagem. Nós vamos utilizar neste exemplo, a distância função de distância DJ definida. Incialmente vamos carregar o mapa da meso-região de Belo Horizonte:

```
> require(spdep)
> bh <- readShapePoly(system.file("etc/shapes/bhicv.shp", package = "spdep")[1])
> (n <- nrow(coordinates(bh)))
```

```
[1] 98
```

O objeto `bh` contém o mapa das 98 cidades que fazem parte da meso região de Belo Horizonte e 7 variáveis, uma delas sendo a população. Nós vamos dividir o mapa em cinco regiões, mostradas na figura 5 e atribuir uma taxa diferente para cada grupo. Nós simulamos um conjunto de dados a taxa definida para fazer a análise de agrupamentos. Na figura 5 observamos a SMR dos dados simulados.

Nós precisamos definir uma função que calcule $d_{i,j}$ e utiliza-la no argumento `otherfun` da função `skater()`. Essa função deve ter três argumentos, um `data.frame`, um inteiro indicando a área da na qual a função será avaliada e o terceiro, um vetor de inteiros para indicar as áreas vizinhas.

Nós criamos um `data.frame` com a população e os casos simulados obtemos lista de vizinhança:

```
> dat <- data.frame(pop = bh$Population, obs = obs)
> nb.bh <- poly2nb(bh)
```

Com isso, temos que a primeira coluna do `data.frame` contém a população e a segunda contém o número de casos. Então, nós definimos a função para cálculo de DJ considerando essa estrutura de dados. Nós trabalhamos na escala logaritma para melhor precisão nos resultados. Além disso, nós consideramos duas correções para evitar o efeito de estar-mos trabalhando com taxas pequenas. 1) se a taxa estimada é zero, fazemos com que seja igual a 1e-10. 2) se o número esperado de casos é menor que 1 e o número observado é zero, fazemos com que o valor máximo que a função de densidade de probabilidade pode assumir seja 1. Inicialmente, nós vamos escrever uma função para o cálculo de DJ que retorne os valores intermediários das quantidades envolvidas nos cálculos e aplicar essa função as dados do município 6:

```
> dj0.pois <- function(data, i, j) {
+     tx <- c(data[i, 2] + data[j, 2])/c(data[i, 1] + data[j, 1])
+     tx[tx == 0] <- 1e-10
+     ei <- tx * dat[i, 1]
+     ej <- tx * dat[j, 1]
+     fi <- dpois(dat[i, 2], ei, log = TRUE)
+     fj <- dpois(dat[j, 2], ej, log = TRUE)
+     maxi <- ifelse(ei < 0.5, ei/10, ei - 0.5) * log(ei) - ei -
```

```
> grs <- list()
> grs$g1 <- which(coordinates(bh)[, 2] > -19)
> grs$g2 <- which(coordinates(bh)[, 2] < (-20.2))
> grs$g3 <- setdiff(which(coordinates(bh)[, 1] > -43.5), unlist(grs))
> grs$g4 <- setdiff(which(coordinates(bh)[, 1] < (-44.3)), unlist(grs))
> grs$g5 <- setdiff(1:n, unlist(grs))
> gr <- integer(n)
> for (i in 1:length(grs)) gr[grs[[i]]] <- i
> lambda <- c(0.001, 0.001, 5e-04, 5e-04, 1e-04)[gr]
> esp0 <- lambda * bh$Population
> set.seed(123)
> obs <- rpois(n, esp0)
> sum(obs == 0)

[1] 16

> tx.obs <- sum(obs)/sum(bh$Population)
> esp.obs <- tx.obs * bh$Population
> summary(smr <- obs/esp.obs)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  0.4212  2.1660  2.7000  4.5950  9.7570

> brea.smr <- c(0, 0.2, 0.5, 2, 5, Inf)
> table(cl.smr <- findInterval(smr, brea.smr))

 1  2  3  4  5
17 13 15 35 18

> par(mfrow = c(1, 2), mar = c(0, 0, 0, 0))
> plot(bh, col = gray(1 - lambda * 100))
> text(-43.5, c(-18.2, -18.1), c("0.5%", "G1"))
> text(-43.3, c(-20.9, -20.8), c("0.5%", "G2"))
> text(-42.7, c(-19.6, -19.5), c("0.2%", "G3"))
> text(-45, c(-19.3, -19.2), c("0.2%", "G4"))
> text(-44.4, c(-18.9, -18.8), c("0.05%", "G5"))
> plot(bh, col = gray(1 - 1:5/7)[cl.smr])
> legend("topleft", leglabs(brea.smr, "<", ">"), fill = gray(1 -
+     1:5/7), bty = "n")
```

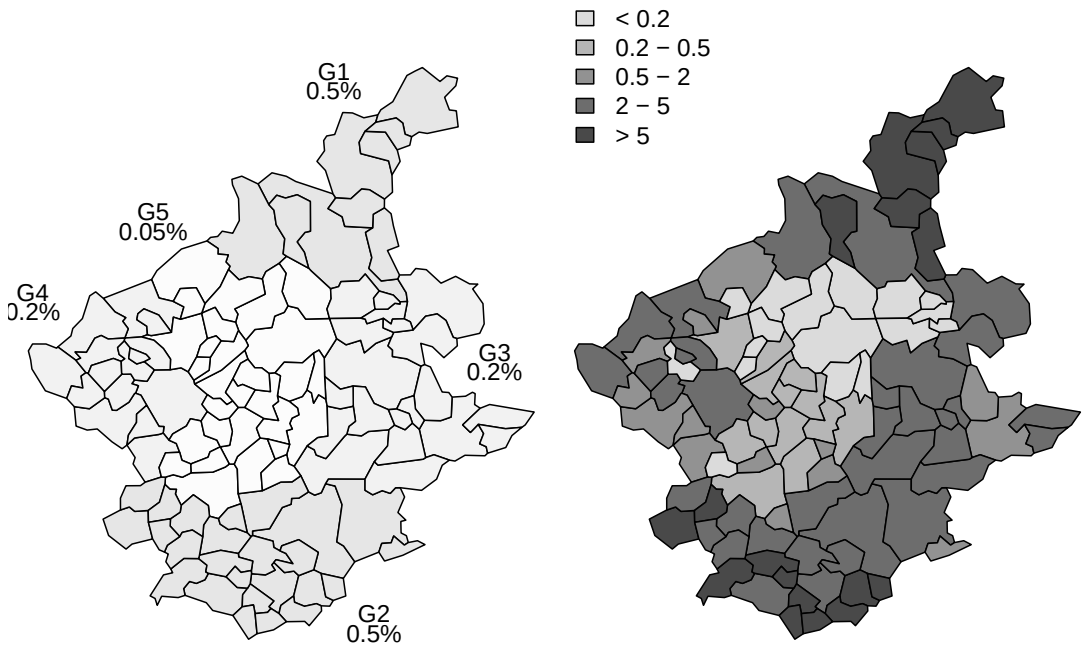

Figura 5 Map of Belo Hozionte macro-region with five groups defined for example (left) and the map of SMR obtained with one simulated data

```
+         lgamma(ifelse(ei < 0.5, 1 + ei/10, ei + 0.5))
+     maxj <- ifelse(ej < 0.5, ej/10, ej - 0.5) * log(ej) - ej -
+         lgamma(ifelse(ej < 0.5, 1 + ej/10, ej + 0.5))
+     maxi[dat[i, 2] == 0 & ei < 1] <- 0
+     maxj[dat[j, 2] == 0 & ej < 1] <- 0
+     data.frame(pop = dat[j, 1], obs = dat[j, 2], tx0 = 1000 *
+         dat[j, 2]/dat[j, 1], tx.j = 1000 * tx, ei, ej, fi = exp(fi),
+         fj = exp(fj), mi = exp(maxi), mj = exp(maxj), dij = 1 -
+             exp(fi - maxi + fj - maxj), row.names = paste(j))
+ }
> data.frame(dat[6, ], tx = 1000 * dat[6, 2]/dat[6, 1])

   pop obs       tx
6 4320   4 0.925926

> round(dj0.pois(dat, 6, nb.bh[[6]]), 3)

    pop obs   tx0  tx.j    ei    ej    fi    fj    mi    mj   dij
7  6797  10 1.471 1.259 5.440 8.560 0.158 0.112 0.172 0.137 0.252
9  5389  10 1.856 1.442 6.229 7.771 0.124 0.093 0.161 0.144 0.501
10 4393   4 0.911 0.918 3.966 4.034 0.195 0.195 0.202 0.201 0.061
12 7760   8 1.031 0.993 4.291 7.709 0.193 0.139 0.194 0.144 0.044
13 6593   3 0.455 0.641 2.771 4.229 0.154 0.184 0.243 0.196 0.407

> 1000 * dat[6, 2]/dat[6, 1] - 1000 * dat[nb.bh[[6]], 2]/dat[nb.bh[[6]],
+     1]

[1] -0.54531138 -0.92970592  0.01538643 -0.10500191  0.47089787
```

Observamos que o municípios 9 e 13 são os mais distantes do município 6. Esse resultado é razoável pois a diferença entre a taxa bruta observada em ambos e a taxa bruta do município 6 são as maiores entre os 5 vizinhos. Nós observamos que o município 10 é muito parecido com o município 6, pois tem população com tamanho muito próximo e ocorreu o mesmo número de casos. Porém, $d_{6,12} < d_{6,10}$, embora a taxa brutra do município 12 seja 1.03 por mil e do município 10 seja 0,91 por mil, esta sendo mais próxima da taxa bruta do município 6, 0,92 por mil. Esse caso é curioso porque observamos aqui que a medida de $d_{i,j}$ não é simétrica. Podemos observar também que o município 12 tem população quase o dobro maior que a população dos municípios 6 e 10, e ocorreu o dobro do número de casos, 8 casos. Neste caso, a variância da taxa bruta do município 12 é menor que dos municípios 6 e 10 e esta pode ser uma justificativa plausível para o fato de que $d_{6,12} < d_{6,10}$.

A seguir, nós redefinimos a função do cálculo de distância para que retorne apenas $d_{i,j}$, calculamos a lista de custos e criamos o objeto da classe `listw` que contém a lista de vizinhança e a lista de custos.

```
> dj.pois <- function(data, i, j) {
+     tx <- c(data[i, 2] + data[j, 2])/c(data[i, 1] + data[j, 1])
+     tx[tx == 0] <- 1e-10
+     ei <- tx * dat[i, 1]
+     ej <- tx * dat[j, 1]
+     fi <- dpois(dat[i, 2], ei, log = TRUE)
+     fj <- dpois(dat[j, 2], ej, log = TRUE)
+     maxi <- ifelse(ei < 0.5, ei/10, ei - 0.5) * log(ei) - ei -
+         lgamma(ifelse(ei < 0.5, 1 + ei/10, ei + 0.5))
+     maxj <- ifelse(ej < 0.5, ej/10, ej - 0.5) * log(ej) - ej -
+         lgamma(ifelse(ej < 0.5, 1 + ej/10, ej + 0.5))
```

```
+     maxi[dat[i, 2] == 0 & ei < 1] <- 0
+     maxj[dat[j, 2] == 0 & ej < 1] <- 0
+     1 - exp(fi - maxi + fj - maxj)
+ }
> nb.dj <- nbcosts(nb.bh, dat, meth = "oth", oth = dj.pois)
> nb.wj <- nb2listw(nb.bh, nb.dj)
```

A partir do objeto `listw` nós obtemos a árvore geradora mínima:

```
> mstj.bh <- mstree(nb.wj)
```

Agora, nós podamos alguns galhos da AGM utilizando o algoritmo SKATER implementado na função `skater`. Para isso, precisamos definir a função para o cálculo da medida de homogeneidade. Essa função deve possuir dois argumentos, o primeiro deve ser um `data.frame` e o segundo um vetor inteiro com os índices dos elementos do grupo. Nos definimos uma função para o cálculo da verossimilhança avaliada no ponto da estimativa de máxima verossimilhança da taxa estimada com os dados do grupo.

```
> nllpois <- function(data, id) {
+     tx <- sum(data[id, 2])/sum(data[id, 1])
+     -sum(dpois(data[id, 2], data[id, 1] * tx, log = TRUE))
+ }
```

Inicialmente vamos obter 2 grupos, fazendo uma poda na árvore. Atribuimos o resultado a um objeto e em seguida obtemos 3, 4 e 5 grupos sequencialmente, atribuindo cada resultado a um objeto diferente. Ao final, fazemos mais 5 podas para gerar 10 grupos, com objetivo de estudar o decaimento da função de homogeneidade.

```
> sk2 <- skater(mstj.bh[, 1:2], dat, 1, method = "other", other = nllpois)
> sk3 <- skater(sk2, dat, 1, method = "other", other = nllpois)
> sk4 <- skater(sk3, dat, 1, method = "other", other = nllpois)
```

```
> par(mfrow = c(2, 2), mar = c(0, 0, 0, 0))
> plot(bh, col = gray(1 - sqrt(lambda) * 10))
> plot(mstj.bh, coordinates(bh), label.areas = "", cex.cir = 0.01,
+     cex.lab = 0.001, add = T, col = "yellow")
> plot(sk2, coordinates(bh), cex.cir = 0.01, cex.lab = 0.001, add = T,
+     lwd = 2)
> plot(bh, col = gray(1 - sqrt(lambda) * 10))
> plot(sk3, coordinates(bh), cex.cir = 0.01, cex.lab = 0.001, add = T,
+     lwd = 2)
> plot(bh, col = gray(1 - sqrt(lambda) * 10))
> plot(sk4, coordinates(bh), cex.cir = 0.01, cex.lab = 0.001, add = T,
+     lwd = 2)
> plot(bh, col = gray(1 - sqrt(lambda) * 10))
> plot(sk5, coordinates(bh), cex.cir = 0.01, cex.lab = 0.001, add = T,
+     lwd = 2)
```

Figura 6 The sub-trees obtained by SKATER in results for two groups (top left), three groups (top right), four groups (bottom left) and five groups (bottom right).

```
> sk5 <- skater(sk4, dat, 1, method = "other", other = nllpois)
> sk10 <- skater(sk5, dat, 5, method = "other", other = nllpois)
```

Podemos observar o decaimento da soma de verossimilhanças e utilizar a diferença como critério de parada. Podemos decidir parar quando essa diferença for menor que q tal que $P(\chi < q) = 1 - \alpha$ com α o nível de significância e $P(\chi)$ a distribuição qui-quadrado com um grau de liberdade. A distribuição qui-quadrado justifica-se pelo fato de que sob-hipótese núla a diferença da soma sequencial de homogeneidades é uma diferença de verossimihanças, com diferença de um parâmetro.

```
> sk10$ssw

 [1] 981.6896 563.4223 320.9714 224.4314 207.8563 201.6048 198.3827 196.0463
 [9] 193.9180 191.7909

> diff(sk10$ssw)

[1] -418.267315 -242.450828  -96.540053  -16.575047   -6.251574   -3.222115
[7]   -2.336325   -2.128342   -2.127078
```

Observamos que para esse conjunto de dados simulados, o cinco é o número adequado de grupos, se consideramos $\alpha = 0.05$ e $q = 3.84$

Na figura 3.4 temos os mapas dos grupos e as sub-arvores obtidas a cada passo, desde dois grupos até cinco grupos. No gráfico superior esquerdo, a aresta amarela é o a aresta cortada no primeiro passo do algoritmo e as arestas vermelhas e azuis definem os dois grupos gerados. Olhando para cada gráfico, podemos seguir os passos do SKATER. O resultado obtido para cinco grupos, gráfico inferior direito, é bastante satisfatório, pois um grupo foi perfeitamente identificado e os demais não são muito diferentes dos verdadeiros grupos.

3.5 Avaliação empírica do SKATER no caso Poisson univariado

Suponha que y_i é a realização de uma variável de Poisson na área i com população P_i. Então nós adotamos que $\boldsymbol{y}_i \sim Poisson(\lambda_i P_i)$. Para avaliar o procedimento SKATER para a análise de cluster espacial nesse caso, nós tomamos os 107 municípios da macroregião de Belo Horizonte, a capital do estado de Minas Gerais, no Brasil.

Nós definimos cinco grupos com os seguintes valores de λ. O grupo 1 ao norte com $\lambda_1 = 0.003$ e nove municípios, o grupo 2 ao sul com $\lambda_2 = 0.003$ e 25 municípios, o grupo 3 ao leste com $\lambda_3 = 0.002$ e 21 municípios, o grupo 4 ao oeste com $\lambda_3 = 0.002$ e 17 municípios e o grupo 5 ao centro da região com $\lambda_5 = 0.001$ e 33 municípios. Esses grupos são mostrados no mapa da esquerda da figura 7.

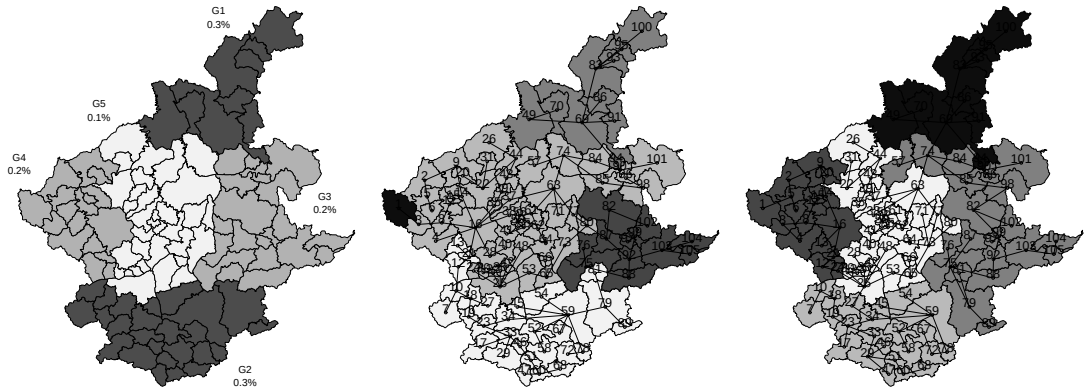

Figura 7 Groups defined for simulation study (left), the groups defined by SKATER with CD measure (center) and the groups defined by SKATER with JD measure (right)

Na figura 7 nós mostramos os grupos definidos (esquerda) e os resultados do SKATER com um conjunto de dados simulados baseado na MST construida com as medidas CD (centro) e JD (direita). Adicionalmente, nós mostramos a MST gerada com as respectivas medidas de dissimilaridade. As cores nas figuras são os grupos definidos (esquerda) e encontrados pelo SKATER com as MST encontradas por ambas as medidas (centro e direita). O critério de parada adotado foi até que cinco grupos foram obtidos. Então é necessário ter uma boa medida de dissimilaridade e gerar uma MST adequada. Olhando para esses resultados, notamos que os grupos encontrados estão fortemente associados à MST gerada. Será que um resultado melhor teria sido obtido com seis grupos?

É fácil avaliar a significância de gerar mais um grupo. Seja SH_i a soma de $H(.)$ obtida no passo $i-1$ onde temos i grupos, SH_1 é $H(.)$ obtida considerando que todas as áreas pertencem ao mesmo grupo. Também seja $DSH_i = SH_{i+1} - SH_i$ a diferença de soma de homogeneidades. Neste caso $DSH_i \sim \chi^2(1)$ sob hipótese de que a redução de soma de homogeneidade não é significativa.

Nós realizamos um pequeno e simples estudo de simulação para avaliar as medidas CD e JC. Com os cinco grupos definidos anteriormente, nós geramos 1000 conjuntos de dados e executamos o SKATER com critério de parada até que 10 grupos foram encontrados. A cada número de grupos encontrados $(2, ..., 10)$, nós calculamos SH, avaliamos a significância de DSH_i e comparamos os grupos encontrados com os grupos verdadeiros. Para essa comparação, nós fizemos uma tabela cruzada do vetor indicador de grupo gerado como o vetor indicador dos grupos verdadeiros. Uma medida adequada nesse caso é a estatística κ, que mede a concordância entre dois classificações. A tabela de freqüências geradas entre os dois vetores de identificadores precisa ser ajustada. Primeiro porque o rótulo definido do grupo verdadeiro pode não ser o mesmo que o rotulado pelo SKATER. Neste caso nós trocamos os rótulos de forma que o grupo obtido pelo SKATER que teve a maior freqüência relativa de municípios do primeiro grupo verdadeiro seja rotulado também como número 1, e assim também para os demais grupos. No caso em que o número de grupos obtidos é diferente de 5, nós completamos a tabela de contingência com colunas ou linhas de zeros.

No gráfico à esquerda da figura 8, nós mostramos os resultados obtidos para SH_i, $i = 1, ..., 10$. Nós plotamos a mediana e os quantis 0.025 e 0.975 de SH_i em função de i. Notamos que SH tem decaimento exponencial e notamos que SH estabiliza após terem sido obtidos 5 grupos, o que é um resultado esperado. Notamos também que SH considerando a MST obtida com CD tem um decaimento mais lento. Comparando DSH_i com o valor 3.84, nós avaliamos se o DSH_i é significativo ao nível de 5% de significância. Para a medida CD, em todas as simulações, DSH_i foi significativo para $i < 6$, significando que pelo menos 6 grupos foi mais adequado para todos os conjuntos de

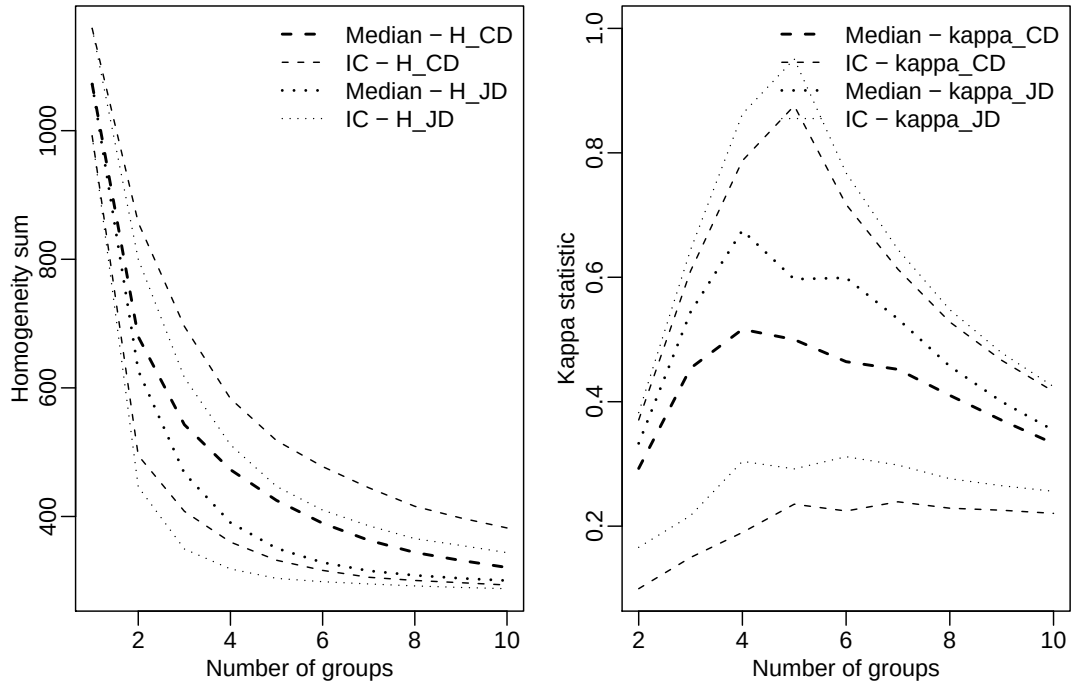

Figura 8 Homogeneity function sum by number of groups (left) and kappa statistic by number of groups (right). The dashed lines are results obtained by CD measure and the dotted lines for JD measure. The more larges lines are for median and minus larges for percentiles of 2.5% and 97.5%.

dados simulados. Para i = 6, 7, 8 e 9, DSH_i foi significativo em 991, 961, 902 e 812 simulações respectivamente. Considerando JD, DSH_i foi significativo para todas as simulações para $i < 4$, ou seja, quatro grupos ou mais foi significativo em todas as simulações. Neste caso, para i = 4, 5, 6, 7, 8 e 9, DSH_i foi significativo em 995, 962, 877, 738, 543 e 377 simulações respectivamente. Esses resultados apontam que a medida JD é mais adequada que CD.

Nós mostramos os resultados obtidos para a estatística κ no gráfico da direita da figura 8. Notamos que os valores mais altos geralmente foram obtidos com 5 grupos, o que era esperado. Notamos aqui também a superioridade da medida JD. Com essa medida, em mais da metade das simulações a concordância é maior que 60% se forem obtidos 3, 5 ou 6 grupos.

4 Aplicação a dados que não são de área

Nesta seção nós fazemos aplicação a dados de processos pontuais, dados de séries temporais e dados independentes de referência espacia e temporal.

4.1 Exemplo com dados de processos pontuais

Nesta seção nós utilizamos a SKATER para a análise agrupamentos em processos pontuais. Um processo pontual são as localizações da ocorrência de eventos em uma região, por exemplo, ocorrência de uma espécie de planta em uma região, casos de doença em uma cidade, etc.

Nosso objetivo é dividir a região em sub-regiões com incidência de casos similares. É trivial notar que quanto menor a incidência de casos numa região, maior a distância entre os casos.

Portanto, se fazemos uma tesselagem de Voronoi, (Okabe, Boots, Sugihara & Chiu 2000), a partir das localizações das ocorrências, temos que quanto menor a incidência de casos numa sub-região, maior a área de influência, dada pela tesselagem, dos pontos dentro dessa sub-região.

Suponha que y_i é a área de influência obtida pela tesselagem de Voronoi. Se em toda a região a incidência é a mesma, então é fácil estimar $E(y_i)$, por exemplo, dividindo a área total da região pelo número de casos. Mas se isso não ocorre, podemos utilizar a análise de agrupamentos para agrupar pontos com área de influência similar e estimar $E(y_i/i \in g_j)$, onde g_j, $j = 1, ..., k$, são as k sub-regiões homogêneas ou grupos.

Nós podemos utilizar a SKATER para dividir uma região em sub-regiões homogêneas quanto à ocorrência de casos. Podemos considerar cada ponto como um nó do grafo e as arestas a adjacência entre as áreas de influência dos pontos obtidas pela tesselagem de Voronoi. Os dados será a área de influência de cada localização.

Nós vamos considerar os dados de padrão pontual de 65 Pinheiros Japoneses, (Numata 1961), disponíveis no pacote **spatstat**.

Inicialmente carregamos os dados, fazemos a tesselagem e obtemos a lista de vizinhança.

```
> require(spatstat)
> data(japanesepines)
> del <- deldir(japanesepines, rw = c(0, 1, 0, 1))
> d.area <- data.frame(del$summary)$dir.area
> nb.del <- tapply(c(del$dirs[, 5], del$dirs[, 6]), c(del$dirs[,
+     6], del$dirs[, 5]), as.integer)
> class(nb.del) <- "nb"
```

A seguir, carregamos o pacote **spdep**, geramos a lista de custos, juntamos a lista de vizinhança com a lista de custos e obtemos a MST. No cálculo dos custos, nós utilizamos a distância euclidiana, que é proporcional a distância baseada na distribuição normal univariada.

```
> require(spdep)
> costs.a <- nbcosts(nb.del, d.area)
> nbw.a <- nb2listw(nb.del, costs.a, style = "B")
> mst.a <- mstree(nbw.a)
```

A medida de homogeneidade que utilizamos foi o negativo do logaritmo da verossimilhança, considerando distribuição normal. Nós consideramos que a variância é a mesma em toda a área e estimada a partir das áreas de todo o conjunto de 65 pontos.

```
> ldnorm <- function(x, id) -sum(dnorm(x[id], mean(x[id]), sd(d.area),
+     log = TRUE))
```

Inicialmente nós dividimos a área em 5 sub-áreas. Com a soma da homogeneidade obtida a cada passo, nós calculamos o ganho em homogeneidade a cada passo, que são diferenças de log-verossimilhanças. Essas diferenças tem distribuição qui-quadrado, como vimos na Seção 2. Com a área dividida 3 grupos, o ganho na verossimilhança é aproximadamente 2.4 ao subdividi-la em 4 grupos. Considerando a distribuição qui-quadrado com 1 grau de liberdade, três grupos não serem suficientes (valor-p) é 0.22. Neste caso nós decidimos que três é o número de grupos adequado.

```
> sk5.a <- skater(mst.a[, 1:2], d.area, 4, method = ldnorm)
> diff(sk5.a$ssw)

[1] -11.112478  -6.058635  -2.424631  -1.624751

> sk3.a <- skater(mst.a[, 1:2], d.area, 2, method = ldnorm)
```

Observando a Figura 4.1, nós observamos que há duas sub-regiões pequenas com áreas de influência maiores. Uma ao sul e oeste da área com seis pontos e outra ao leste com três pontos. Ou seja, nessas duas sub-areas a incidência de ocorrências é significativamente menor que no restante da área, pelo teste de número de grupos.

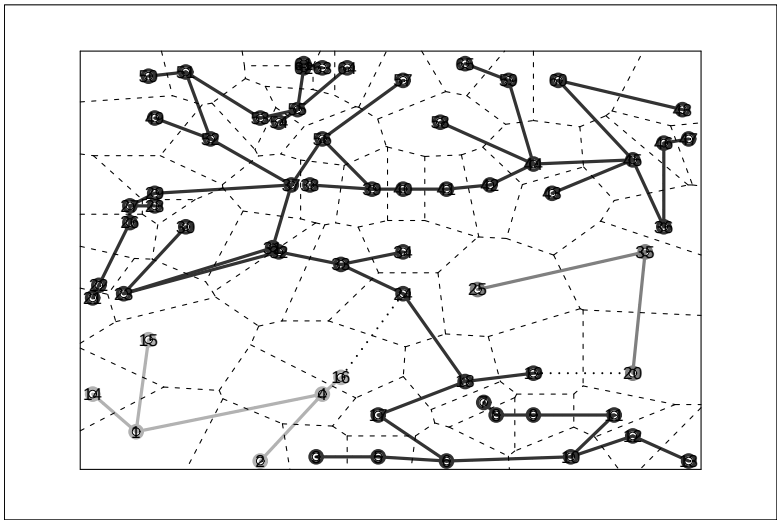

Figura 9 The points is the locations of 65 japanese pines at square. The dashed lines are the Voronoi tesselation. The continuous lines are the MST pruned and the two dotted lines are the pruned edges of MST.

4.2 Aplicação a dados de séries temporais

Nesta seção nós aplicamos a análise a dados de séries temporais. Nós tomamos como exemplo uma série temporal simulada de um modelo auto regressivo de ordem 1. Nós simulamos uma série de tamanho 150 com parâmetro autoregressivo igual a 0.5 e com média igual a zero nos primeiros 50 valores, média igual a 1 nos próximos 50 valores e média igual a 2 nos últimos 50 valores.

```
> set.seed(123)
> n <- 150
> serie <- arima.sim(list(ar = 0.5), n) + rep(0:2, each = n/3)

> require(spdep)
```

A estrutura de uma série temporal já define uma MST. Neste caso, a análise parte da definição da MST:

```
> mst.s <- cbind(1:(n - 1), 2:n)
```

No cálculo da homogeneidade de um grupo, a verossimilhança, precisamos estimar os parâmetros do modelo ajustado à série nesse grupo. Nós vamos estimar os parâmetros utilizando optimização numérica. Para isso, definimos a função de verossimilhança com o primeiro argumento da função sendo os parâmetros do modelo.

```
> f <- function(pars, x) -sum(dnorm(x[-1], cbind(1, x[-length(x)] -
+     pars[1]) %*% pars[1:2], pars[3], log = TRUE))
```

A seguir definimos a função para o cálculo da verossimilhança, restrigindo que o cálculo seja feito apenas se houver mais de três observações no grupo.

```
> llnorm <- function(x, id) {
+     id <- sort(id)
+     if (length(id) < 5)
+         return(Inf)
+     else f(optim(c(mean(x[id]), 0, sd(x[id])), f, x = x[id])$par,
```

```
+         x[id])
+ }
> optim(c(mean(serie), 0, sd(serie)), f, x = serie)

$par
[1] 0.9669542 0.6173814 0.9752087

$value
[1] 207.6700

$counts
function gradient
     108       NA

$convergence
[1] 0

$message
NULL
```

Inicialmente aplicamos o SKATER para gerar cinco grupos:

```
> sk5 <- skater(mst.s, serie, 4, method = llnorm)
```

Após gerar cinco grupos, observamos a homogeneidade total a cada passo e as diferenças sucessivas. Considerando que o modelo ajustado tem três parâmetros, a queda de desvios sob hipótese de que não é necessário fazer uma divisão a mais tem distribuição qui-quadrado com três graus de liberdade.

```
> sk5$ssw

[1] 207.6700 198.6266 190.9931 185.0074 178.2188

> diff(sk5$ssw)

[1] -9.043347 -7.633564 -5.985644 -6.788636

> pchisq(-diff(sk5$ssw), 3, lower = FALSE)

[1] 0.02872009 0.05422412 0.11231079 0.07894865
```

Observamos que a queda de desvios obtida de dois para três grupos não é tão significativa. Isso deve ser devido ao fato de que na série simulada, apenas o parâmetro de média é que é diferente nos três grupos. Poderíamos fixar o parâmetro de correlação e de variância e estimar apenas a média em cada passo. Mas poderia levar a um má desempenho no início do processo. A variância, por exemplo, poderia ser subestimada no início, fazendo com que a verossimilhança não fosse tão grande.

A seguir, obtemos três grupos e plotamos o resultado, que pode ser visualizado na figura 4.2. Nessa figura, utilizamos os valores da série como coordenadas do grafo.

4.3 Aplicação a dados independentes

Nesta secao nos aplicamos o SKATER a dados independentes de referencia espacial e temporal. Nos aplicamos aos dados “iris”, “Edgar Anderson’s Iris Data”.

```
> sk3 <- skater(mst.s, serie, 2, method = llnorm)
> plot(sk3, cbind(1:n, serie))
```

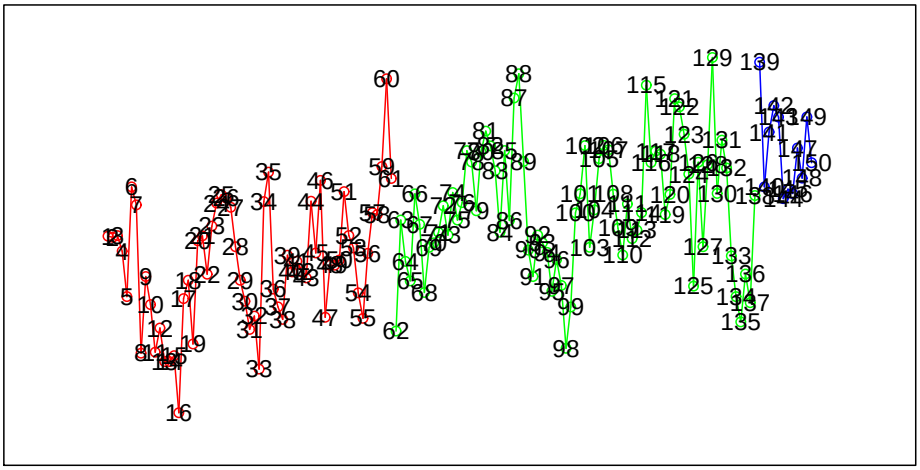

Figura 10 Resultado do SKATER aplicado a série simulada

This famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris. The species are: Iris setosa, versicolor, and virginica.

Nos comparamos o resultado do skater com 3 metodos hierarquicos e 3 metodos partitioning de analise de cluster, disponiveis no pacote **cluster** do **R**.

```
> data(iris)
> n <- nrow(dpad <- scale(iris[, 1:4]))
```

each observation is neighboor of all other

```
> nb.ir <- lapply(1:n, function(i) as.integer((1:n)[-i]))
> class(nb.ir) <- "nb"
```

```
> require(spdep)
```

calculing costs with Euclidean distance between observations

```
> e.costs.ir <- nbcosts(nb.ir, dpad)
```

```
> e.nbw.ir <- nb2listw(nb.ir, e.costs.ir, style = "B")
```

find a minimum spanning tree

```
> e.mst.ir <- mstree(e.nbw.ir)
```

generating 3 groups

```
> system.time(res1 <- skater(e.mst.ir[, 1:2], dpad, 2))

   user  system elapsed
  2.616   0.012   2.632

> table(res1$gr, iris[, 5])

    setosa versicolor virginica
  1      0          8        31
  2     50          0         0
  3      0         42        19
```

Hierarquical cluster analysis

```
> e.dist <- dist(dpad)
> h3 <- cutree(hclust(e.dist), 3)
```

other algorithims

```
> require(cluster)
```

Hierarquical
AGNES: Computes agglomerative hierarchical clustering of the dataset.

```
> ag3 <- cutree(agnes(e.dist), 3)
```

DIANA: Computes a divisive hierarchical clustering of the dataset returning an object of class 'diana'.

```
> di3 <- cutree(diana(e.dist), 3)
```

Partitioning Methods PAM: Partitioning Around Medoids

```
> pam3 <- pam(e.dist, 3)$clust
```

CLARA: Clustering Large Applications

```
> cla3 <- clara(e.dist, 3)$clust
```

FANNY: Computes a fuzzy clustering of the data into 'k' clusters.

```
> fan3 <- fanny(e.dist, 3)$clust
```

Funcao para reagrupar, para fazer comparacao

```
> reagr <- function(gr, ref) {
+     tab <- table(gr, ref)
+     mx <- apply(tab, 2, function(x) max(x/sum(x)))
+     if (any(duplicated(mx)))
+         tab <- t(tab)
+     id <- apply(tab, 2, function(x) which.max(x/sum(x)))
+     tab <- rbind(tab[id, ], tab[-id, ])
+     tab <- cbind(tab, matrix(0, nrow(tab), nrow(tab) - ncol(tab)))
+     if (any(duplicated(mx)))
+         tab <- t(tab)
+     class(tab) <- "table"
+     tab
+ }
```

fazendo as tabelas

```
> lapply(data.frame(res1$gr, ag3, di3, pam3, cla3, fan3), function(x) reagr(x,
+     iris[, 5]))

$res1.gr
  setosa versicolor virginica
2     50          0         0
3      0         42        19
1      0          8        31

$ag3
  setosa versicolor virginica
1     50          0         0
2      0         50        47
3      0          0         3

$di3
  setosa versicolor virginica
1     50          0         0
3      0         39        17
2      0         11        33

$pam3
  setosa versicolor virginica
1     50          0         0
3      0         41        14
2      0          9        36

$cla3
  setosa versicolor virginica
1     50          0         0
3      0         45        13
2      0          5        37

$fan3
  setosa versicolor virginica
1     50          0         0
3      0         38        11
2      0         12        39
```

calculando a proporco de elementos na diagonal

```
> apply(cbind(ska = res1$gr, ag3, di3, pam3, cla3, fan3), 2, function(x) sum(diag(reagr(x,
+     iris[, 5])))/n)

      ska       ag3       di3      pam3      cla3      fan3
0.8200000 0.6866667 0.8133333 0.8466667 0.8800000 0.8466667

> sum(diag(table(h3, iris[, 5])))/n

[1] 0.7866667
```

Observamos que o desempenho do skater compara-se ao depempenho dos Partitioning Methods e ganha dos metodos hierarquicos

Referências

Assunção, R. M.; Lage, J. P. & Reis, E. A. (2002). Análise de conglomerados espaciais via árvore geradora mínima, *Revista Brasileira de Estatística* **62**: 1–23.

Assunção, R. M., Neves, M. C., Câmara, G. & Freitas, C. d. (2006). Efficient regionalization techniques for socio-economic geographical units using minimum spanning trees, *Journal of Geographical Information Science* **20**(7): 797–811.

code by Richard A. Becker, O. S. & version by Ray Brownrigg Enhancements by Thomas P Minka <surname@stat.cmu.edu>, A. R. W. R. (2009). *maps: Draw Geographical Maps.* R package version 2.1-0.

Maechler, M., Rousseeuw, P., Struyf, A. & Hubert, M. (2005). Cluster analysis basics and extensions. Rousseeuw et al provided the S original which has been ported to R by Kurt Hornik and has since been enhanced by Martin Maechler: speed improvements, silhouette() functionality, bug fixes, etc. See the 'Changelog' file (in the package source).

Numata, M. (1961). Forest vegetation in the vicinity of choshi. coastal flora and vegetation at choshi, chiba prefecture, *IV Bulletin of Choshi Marine Laboratory, Chiba University* **3**: 28–48. in Japanese.

Okabe, A., Boots, B., Sugihara, K. & Chiu, S. N. (2000). *Spatial Tessellations - Concepts and Applications of Voronoi Diagrams*, 2nd edition edn, John Wiley. 671 pages. ISBN 0-471-98635-6.

Prim, R. C. (1957). Shortest connection networks and some generalisations, *Bell System Technical Journal* **36**: 1389–1401.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.
URL: *http://www.R-project.org*

Shafer, G. (1976). *A mathematical theory of evidence*, Princeton. 297p.