Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Estatística

Dissertação de Mestrado

Um algoritmo PSO especializado para o problema de detecção de clusters espaciais

por

Guilherme Dias Malvão

Brasília, 20 de março de 2020

Um algoritmo PSO especializado para o problema de detecção de clusters espaciais

por

Guilherme Dias Malvão

Dissertação apresentada ao Departamento de Estatística da Universidade de Brasília, como requisito parcial para obtenção do título de Mestre em Estatística.

Orientador: Prof. Dr. André Luiz Fernandes Cançado

Brasília, 20 de março de 2020

Dissertação submetida ao Programa de Pós-Graduação em Estatística do Departamento de Estatística da Universidade de Brasília como parte dos requisitos para a obtenção do grau de Mestre em Estatística.

Texto aprovado por:

Prof. Dr. André Luiz Fernandes Cançado
Orientador, EST/UnB

Prof. Dr. Guilherme Souza Rodrigues
EST/UnB

Prof. Dr. Alexandre Celestino Leite Almeida
UFSJ/CAP

Algumas batalhas são vencidas com espadas e lanças, outras com papel e caneta.

(Tywin Lannister)

Agradecimentos

Agradeço à minha namorada, noiva e esposa, Clarissa Cardoso Oesselmman, por todo carinho, apoio e compreenssão.

Ao meu orientador Professor Dr. André Luiz Fernandes Cançado, por toda paciência, ajuda e disponibilidade. E também por aceitado me orientar antes mesmo de ter sido aceito no mestrado.

À minha família e amigos que me fizeram chegar aqui e que me dão força sempe.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

Resumo

A detecção de *clusters* espaciais tem grande importância para tomada de decisão em diversas áreas como saúde e segurança pública. O método Scan Circular de Kulldorff (1997) é a maior referência quando se trata em detecção de *clusters* espaciais. O método não tem um bom desempenho quando o cluster verdadeiro tem um formato não circular. Neste trabalho propomos um algoritmo baseado no Particle Swarm Optimizaton (PSO), que é capaz de detectar e identificar *clusters* espaciais. Os dois métodos são comparados em 4 cenários, onde cada um tem um *cluster* com formato específico que são gerados através de simulações proposta por Kulldorff, Tango e Park (2003). Também é feita a comparação entre os dois métodos para dados de óbitos por doenças pulmonares obstrutivas crônicas no estado do Mato Grosso, Brasil, no ano de 2015. São encontrados resultados favoráveis para o método proposto, principalmente nos casos onde o *cluster* não tem formato circular.

Palavras-chaves: Detecção de clusters espaciais, estatística espacial, otimização.

Abstract

The detection of spatial clusters is of great importance for decision making in several areas such as health and public safety. The Scan Circular method of Kulldorff (1997) is the most important reference when it comes to detecting spatial clusters. The method does not perform well when the real cluster has a non-circular shape. We propose an algorithm inspired by the Particle Swarm Optimizaton (PSO), which is capable of detecting and identifying spatial clusters. The two methods are compared in 4 scenarios, where each one has a cluster with specific format that are generated through simulations proposed by Kulldorff, Tango e Park (2003). A comparison is also made between the two methods for chronic obstructive pulmonary diseases data in the state of Mato Grosso, Brazil, in 2015. Favorable results are found for the proposed method, especially in cases where the cluster has no circular shape.

Keywords: Spatial cluster detection, spatial statistics, optimization.

Sumário

Capítulo 1

Introdução

A Estatística espacial é a área da Estatística que estuda os fenômenos aleatórios ao longo do espaço e tem aplicações em diversas áreas como agronomia, demografia, criminologia epidemiologia e outras. Na epidemiologia, podemos estar interessados, por exemplo, em verificar se a proporção de casos de uma doença em bairros de uma cidade se dá de forma aleatória ou segue algum padrão. Se em um determinado grupo de bairros vizinhos observarmos uma incidência significativamente maior dessa doença, podemos caracterizar esses bairros como um conglomerado espacial (ou *cluster* espacial). Na agronomia Cuadros et al. (2017) identifica cluster em fazendas de plantações de batata que apresentaram um certo tipo de vírus, na demografia Collado Chaves (2003) detecta a presença de cluster de alta fecundidade de adolecentes na região metropolitana da Costa Rica, na criminologia Minamisava et al. (2009) encontra clusters espaciais de mortes violentas em uma região recém-urbanizada do Brasil.

Devido à grande relevância do tema, encontramos na literatura diversos trabalhos na área de clustering espacial. Choynowski (1959) afirma que embora a distribuição de algum fenômeno sobre uma área geográfica em termos de frequências absolutas ou porcentagens seja útil como uma descrição simples dos dados, ela pode ser falha quando se quer fazer inferências com base na distribuição espacial do fenômeno. Assim, ele sugere que se calcule uma probabilidade para cada região com base no número de casos em cada uma delas utilizando a distribuição de

Poisson.

Whittemore et al. (1987) propõem um método para testar a existêcia de um cluster ao longo do espaço. O método é baseado em um teste de hipótese onde a hipótese nula é de que não existe cluster naquele espaço. A estatística de teste é baseada na distância média entre todos os pares de casos. Whittemore et al. (1987) aplicam o teste para detectar a presença de cluster nos setores censitários de São Francisco, onde foram observados um total de 63 casos de carcinoma de células escamosas anal e retal no período de 1973 até 1981. O valor encontrado para a estatística de teste foi significativo apresentando um p-valor menor que 0,001, indicando que existe a presença de cluster espacial em São Francisco, ou seja, existe um grupo de setores censitários vizinhos em que a probabilidade de contrair a doença dentro dele é maior do que fora. O método, porém, não fornece a localização exata do cluster.

Já no artigo de Openshaw et al. (1988) é proposto um método chamado Geographical Analysis Machine (GAM). Esse método propõe identificar a localização espacial de um possível cluster. Para realizar o teste são examinados vários círculos de raio r no mapa, sendo que os valores de r variam dentro de um intervalo previamente definido pelo pesquisador. O centro de cada círculo varia ao longo dos vértices de uma malha quadriculada sobreposta ao mapa. Para cada círculo de raio r é calculada a quantidade de casos que ocorrem no seu interior, e se essa quantidade for significativamente maior que o esperado o círculo é desenhado no mapa. O teste foi aplicado em algumas regiões da Inglaterra entre 1968 e 1984, onde foram observados 853 casos de leucemia em crianças de até 15 anos. Foram examinados 812.993 círculos e 1792 foram considerados significativos. O procedimento GAM fornece um excelente método descritivo para encontrar áreas com altas incidências da doença. No entanto, não é fornecida uma medida final para responder sobre a existência de cluster de forma geral.

Turnbull et al. (1989) utilizam, assim como nos outros trabalhos citados anteriormente, a ausência de clusters espaciais como hipótese nula. Sob essa hipótese, espera-se que regiões com população de mesmo tamanho apresentem a mesma quantidade de ocorrências de casos. Em função disso, Turnbull et al. (1989) propõem criar um conjunto de janelas sobrepostas

ou áreas de tamanho populacional constante centradas em uma grade irregular formada pelos centroides das regiões. O fato de fixar previamente o tamanho populacional para as janelas, aumenta a comparabilidade entre as mesmas.

Considerando o número de casos em cada janela como sendo C_{iR}, $i = 1, 2, \ldots, J$, onde i representa i-ésima janela, J o número total de janelas, e R a população total pré-fixada, então C_{iR} deve ser diretamente proporcional às taxas da doença. Os valores das taxas resultantes podem ser considerados variáveis aleatórias identicamente distribuídas, mas não independentes, e portanto podem ser usadas na construção estatística do teste. Uma escolha natural para a estatística de teste é o máximo dos C_{iR}, $M_R = \max(C_{1R}, C_{2R}, \ldots, C_{JR})$. A hipótese nula é rejeitada se M_R for maior que algum valor de corte k, onde k é determinado pela distribuição de M_R sob a hipótese nula e pelo nível de significância α.

Besag e Newell (1991) propõem um método bem similar ao de Turnbull et al. (1989), uma diferença ocorre em usar um número pré estabelecido de casos no lugar de um número pré estabelecido da população. As informações necessárias para realizar o método são: os casos da doença na área do estudo, registrados por um período de vários anos, o local de residência no momento de diagnóstico da doença e uma população correspondente em risco. Cada caso da doença durante o período do estudo foi atribuído a um centroide de uma região particular, determinado pelo local de residência. Assim, os dados para cada região consistem nas coordenadas dos centroides, o número de casos e uma população em risco.

Besag e Newell (1991) realizam um teste de significância para decidir se a região associada forma o centro de um cluster com $k + 1$ casos. A hipótese nula do teste é que o número total de casos é distribuído aleatoriamente entre toda população, ou seja a probabilidade de ocorrer um caso na região i é t_i/T, onde t_i é a população da região i e T é toda a população da área em estudo. No entanto, eles enfatizam que cada teste individual examina apenas a estrutura local do padrão e não tentam compensar um aparente cluster uma vez detectado. Considerando um caso particular, seja R_0 a região de ocorrência do caso, e $R_1, R_2, \ldots$ as próximas regiões, determinadas pela distâncias de R_0, onde R_1 é a primeira região mais próxima de R_0, R_2 é a

segunda região mais próxima de R_0 e assim sucessivamente. Definindo $D_i = (\sum_{j=0}^{i} c_j) - 1$ e $u_i = (\sum_{j=0}^{i} t_j) - 1$, então $D_0 \leq D_1 \leq \ldots$ são os números de casos acumulados em $R_0, R_1, \ldots$, e $u_0 \leq u_1 \leq \ldots$ são as correspondentes populações acumuladas. Besag e Newell (1991) definem $M = \min\{i : D_i \geq k\}$, assim, se o valor observado de M for baixo, é um indicativo de cluster em torno da região R_0, formalmente se m é o valor observado de M, o nível de significância do teste pode ser calculado sob H_0 como sendo $P(M \leq m)$. Assim, o nível de significância para cada cluster em potencial pode ser calculado, é sugerido pelos autores que seja desenhado no mapa todos os clusters com nível de significância menor que 5%.

O método foi aplicado na mesma base de dados utilizada no artigo de Openshaw et al. (1988) para o período de 1975 até 1985. Utilizando um valor de $k = 4$, é comparado o número de clusters encontrados com o número esperado, a separação dos clusters encontrados em grupos sem sobreposição e a utilização da simulação de Monte Carlo para verificar o nível de significância de cada um dos clusters e para a área como um todo.

Como pode ser visto, ao longo dos anos, foram desenvolvidos diversos métodos para detecção de clusters espaciais, mas o método que se tornou mais utilizado foi a estatística Scan Espacial proposto por Kulldorff (1997) devido sua capacidade de realizar inferências e de ser de fácil implementação. Devido sua grande relevância iremos dedicar uma seção para explicar com detalhes esse métodos.

Duczmal et al. (2007), propõe uma nova abordagem para detecção e inferência de clusters espaciais de formas irregulares, usando um algoritmo genético. Dado um mapa dividido em regiões com populações correspondentes em risco e casos, as operações relacionadas a gráficos são minimizadas por meio de uma rápida geração de nascimentos e avaliação eficiente da estatística Scan Espacial de Kuldorff. Uma função de penalidade baseada no conceito de não compactação geométrica é empregado para evitar irregularidades excessivas na forma geométrica do cluster.

No artigo de Izakian e Pedrycz (2012) é citado que a estatística Scan Espacial de Kulldorff tem sido aplicada a vários problemas de detecção de cluster geográficos de doenças, e também

é citado a estatística Scan de Kulldorff para problemas espaço-temporal. É observado que como a forma da janela de varredura usada nesses métodos é circular ou elíptica, eles não conseguem encontrar aglomerados de formato irregular, como os aglomerados que ocorrem ao longo vales fluviais ou nos casos em que a transmissão de doenças está ligada à rede viária. Izakian e Pedrycz (2012) propõe uma estrutura geométrica mais flexível para ser usada como uma janela de varredura espacial ou espaço-temporal. O método *particle swarm optimization* (PSO) proposto por Kennedy e Eberhart (1995) é usado para otimizar a janela de varredura para determinar grupos de doenças. O método proposto é avaliado em vários conjuntos de dados espaciais e espaço-temporais incluindo dados de mortalidade de câncer de mama no nordeste dos EUA 1988-1992 e diferentes tipos de câncer no Novo México 1982-2007. Os resultados experimentais mostraram que a abordagem introduzida superou os resultados produzidos pela estatística Scan circular e elíptica em termos de eficiência, especialmente ao lidar com formatos de clusters irregulares.

A proposta desse trabalho é apresentar um algoritmo baseado no PSO, que seja capaz de identificar e detectar clusters espaciais. O algoritimo foi implementado em linguagem R e comparado com a Scan Espacial de Kulldorff por meio de dados simulados e dados reais.

Capítulo 2

Metodologia

2.1 Revisão Metodológica

Nesta seção iremos descrever em detalhes o método proposto por Kulldorff (1997), para realizar a detecção e inferência de clusters espaciais.

2.1.1 Estatística Scan Espacial de Kulldorff

Considere um mapa dividido em r regiões, cada região com população n_i e número de casos x_i, $i = 1, \ldots, r$. Assim, a população total do mapa será $N = \sum_{i=1}^{r} n_i$ e o número total casos será $C = \sum_{i=1}^{r} x_i$. Uma zona z é um subconjunto de regiões conexas no mapa e Z é o conjunto de todas zonas. Vamos denotar o número de casos em uma zona z por $c_z = \sum_{i \in z} x_i$, e a população da zona z por $n_z = \sum_{i \in z} n_i$. Analogamente, o número de casos e a população fora da zona z são dados, respectivamente, por $c_{\bar{z}} = \sum_{i \notin z} x_i$ e $n_{\bar{z}} = \sum_{i \notin z} n_i$.

O método de Kulldorff (1997) propõe detectar a presença de clusters em um mapa e identificá-los corretamente. Para tanto, define-se um cluster como uma zona z para qual a probabilidade de um indivíduo vir a ser um caso é maior do que fora de z.

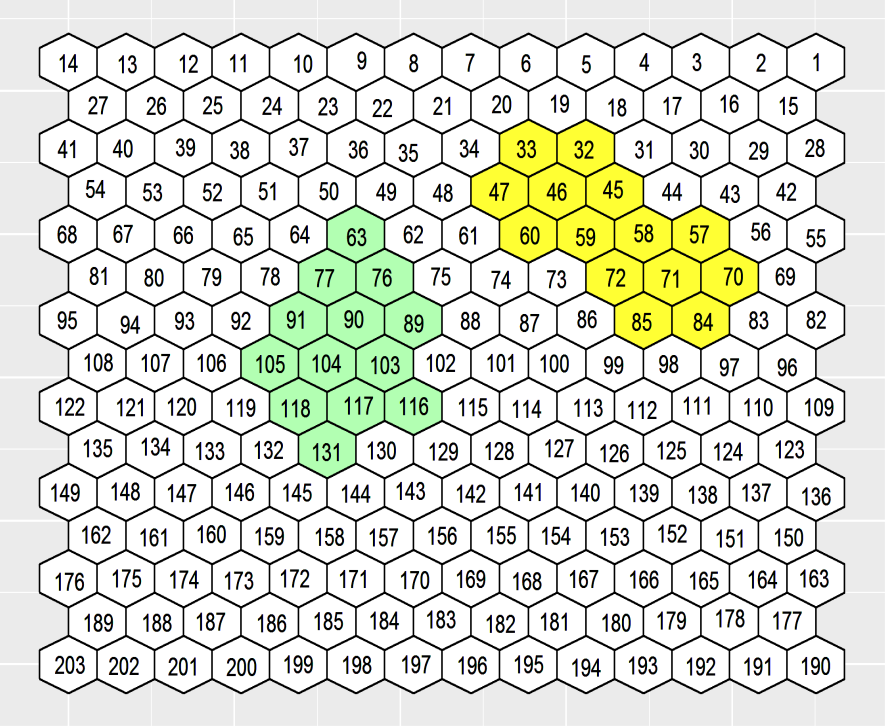

Figura 2.1: Mapa com identificador de cada região e duas zonas.

A estatística Scan Espacial de Kulldorff é definida a partir de um teste de razão de verossimilhanças, associado às seguintes hipóteses:

- H_0: p_z=$p_{\bar{z}}$=p_0. Para toda zona z
- H_1: Existe alguma zona z tal que $p_z > p_{\bar{z}}$,

em que p_z é a probabilidade de um indivíduo dentro da zona z ser um caso e $p_{\bar{z}}$ é a probabilidade de um indivíduo fora da zona z ser um caso. Uma zona z será considerada um cluster se para essa zona o teste rejeitar a hipótese nula.

O número de casos x_i em uma região i pode ser modelado através de diferentes distribuições. Para cada distribuição escolhida a forma da estatística do teste de razão de verossimilhança será diferente. A seguir iremos mostrar estatística de teste para as distribuições Binomial e Poisson.

Modelo Binomial

Suponha que $x_i \sim Binomial(n_i, p_i)$, em que $p_i = p_z$ se $i \in z$ e $p_i = p_{\bar{z}}$ se $i \notin z$.

Sob H_0 temos que p_z=$p_{\bar{z}}$=p_0. Assim a verossimilhança pode ser escrita como

$$L_0(p_0) = \left[\prod_{i=1}^{r} \begin{pmatrix} n_i \\ x_i \end{pmatrix}\right] p_0^C (1-p_0)^{N-C},$$

e a log-verossimilhança tem a forma

$$l_0(p_0) = \sum_{i=1}^{r} \log \begin{pmatrix} n_i \\ x_i \end{pmatrix} + C\log(p_0) + (N-C)\log(1-p_0).$$

Derivando a log-verossimilhança em relação a p_0 e igualando a zero, obtemos o estimador $\hat{p}_0$ de p_0 dado por $\hat{p}_0 = C/N$.

Sob H_1 temos que existe alguma zona z para a qual probabilidade de um indivíduo ser um caso é maior que fora dela, ou seja $p_z > p_{\bar{z}}$. Nesse caso, a verossimilhança assume a forma

$$L(p_z, p_{\bar{z}}) = \prod_{i\in z} \begin{pmatrix} n_i \\ x_i \end{pmatrix} p_z^{c_z}(1-p_z)^{n_z - c_z} \prod_{i\notin z} \begin{pmatrix} n_i \\ x_i \end{pmatrix} p_{\bar{z}}^{c_{\bar{z}}}(1-p_{\bar{z}})^{n_{\bar{z}} - c_{\bar{z}}}, \tag{2.1}$$

e a log-verossimilhança será dada por

$$\begin{aligned} l(p_z, p_{\bar{z}}) &= \sum_{i\in z} \log \begin{pmatrix} n_i \\ x_i \end{pmatrix} + c_z \log(p_z) + (n_z - c_z)\log(1-p_z) \\ &+ \sum_{i\notin z} \log \begin{pmatrix} n_i \\ x_i \end{pmatrix} + c_{\bar{z}} \log(p_{\bar{z}}) + (n_{\bar{z}} - c_{\bar{z}})\log(1-p_{\bar{z}}). \end{aligned} \tag{2.2}$$

Maximizando a função de log-verossimilhança obtemos os estimadores $\hat{p}_{\bar{z}} = c_{\bar{z}}/n_{\bar{z}}$ e $\hat{p}_z = c_z/n_z$.

Substituindo p_0 pelo seu estimador em $L_0(p_0)$ temos

$$\sup_{p_z=p_{\bar{z}}=p_0} L(p_0) = L_0 = \prod_{i=1}^{r} \left(\begin{array}{c} n_i \\ x_i \end{array} \right) \left(\frac{C}{N}\right)^{C} \left(1 - \frac{C}{N}\right)^{N-C}. \tag{2.3}$$

De forma análoga, substituindo os estimadores de p_z e $p_{\bar{z}}$ em $L(p_z, p_{\bar{z}})$

$$\begin{aligned} \sup_{p_z>p_{\bar{z}}} L(p_z, p_{\bar{z}}) = L(z) &= \prod_{i\in z} \left(\begin{array}{c} n_i \\ x_i \end{array} \right) \left(\frac{c_z}{n_z}\right)^{c_z} \left(1 - \left(\frac{c_z}{n_z}\right)\right)^{n_z-c_z} \\ &\times \prod_{i\notin z} \left(\begin{array}{c} n_i \\ x_i \end{array} \right) \left(\frac{c_{\bar{z}}}{n_{\bar{z}}}\right)^{c_{\bar{z}}} \left(1 - \left(\frac{c_{\bar{z}}}{n_{\bar{z}}}\right)\right)^{n_{\bar{z}}-c_{\bar{z}}}. \end{aligned} \tag{2.4}$$

A razão de verossimilhança é definida por

$$\lambda_z = \frac{\sup_{p_z>p_{\bar{z}}} L(p_z, p_{\bar{z}})}{\sup_{p_z=p_{\bar{z}}=p_0} L(p_0)}. \tag{2.5}$$

É de fácil verificação que o denominador na equação acima se reduz a L_0 visto na equação 2.3. L_0 depende somente do total do número de casos e não da sua distribuição espacial, e é uma constante desde que seja condicionada a C. O numerador pode ser encontrado em dois passos. Primeiro, para uma zona z fixa, maximizamos $L(p_z, p_{\bar{z}})$ em relação a p_z e $p_{\bar{z}}$ obtendo a forma descrita na equação 2.5.

Seja $L(z) = \sup_{p_z > p_0} L(p_z, p_0) =$

$$\begin{cases} \prod_{i \in z} \begin{pmatrix} n_i \\ x_i \end{pmatrix} (\frac{c_z}{n_z})^{c_z} (1 - (\frac{c_z}{n_z}))^{n_z - c_z} \prod_{i \notin z} \begin{pmatrix} n_i \\ x_i \end{pmatrix} (\frac{c_{\bar{z}}}{n_{\bar{z}}})^{c_{\bar{z}}} (1 - (\frac{c_{\bar{z}}}{n_{\bar{z}}}))^{n_{\bar{z}} - c_{\bar{z}}}, & \text{se } \frac{c_z}{n_z} > \frac{c_{\bar{z}}}{n_{\bar{z}}} \\ \prod_{i=1}^{r_z} \begin{pmatrix} n_i \\ x_i \end{pmatrix} (\frac{C}{N})^{C} (1 - \frac{C}{N})^{N - C}, & \text{caso contrário} \end{cases} \tag{2.6}$$

Utilizando as equações 2.3 e 2.6 podemos escrever a estatística de teste na forma

$$\lambda = \sup_z \lambda_z = \sup_z \frac{L(z)}{L_0}. \tag{2.7}$$

Isto é, a estatística de teste será dada pelo maior valor da razão de verossimilhança calculada para todas as zonas candidatas e a zona z que maximiza λ_z é chamada de cluster mais verossímil.

Modelo Poisson

Agora, consideremos que o número de casos na i-ésima região seja $x_i \sim Poisson(n_i p_i)$ em que $p_i = p_z$ se $i \in z$ e $p_i = p_{\bar{z}}$ se $i \notin z$. Sob H_0 temos que p_z=$p_{\bar{z}}$=p_0. A verossimilhança pode ser escrita como

$$L_0(p_0) = \prod_{i=1}^{r} \frac{e^{-n_i p_0} (-n_i p_0)^{x_i}}{x_i!} = \frac{e^{-N p_0} p_0^C \prod_{i=1}^{r} {n_i}^{x_i}}{\prod_{i=1}^{r} (x_i!)}$$

e a log-verossimilhança tem a forma

$$l_0(p_0) = -N p_0 + C \log(p_0) + \sum_{i=1}^{r} x_i \log(n_i) - \sum_{i=1}^{r} \log(x_i!).$$

Derivando a log-verossimilhança em relação a p_0 e igualando a zero, obtemos o estimador $\hat{p}_0$ de p_0 dado por $\hat{p}_0 = C/N$.

Sob H_1 temos que existe alguma zona z para qual a probabilidade de um indivíduo ser um caso é maior que fora dela, ou seja $p_z > p_{\hat{z}}$. Nesse caso, a verossimilhança assume a forma

$$\begin{aligned} L(p_z, p_{\bar{z}}) &= \prod_{i \in z} \frac{e^{-n_i p_z}(-n_i p_z)^{x_i}}{x_i!} \prod_{i \notin z} \frac{e^{-n_i p_{\bar{z}}}(-n_i p_{\bar{z}})^{x_i}}{x_i!} \\ &= \frac{e^{-n_z p_z} p_z^{c_z} \prod_{i \in z} {n_i}^{x_i}}{\prod_{i \in z}(x_i!)} \frac{e^{-n_{\bar{z}} p_{\bar{z}}} p_{\bar{z}}^{c_{\bar{z}}} \prod_{i \notin z} {n_i}^{x_i}}{\prod_{i \notin z}(x_i!)}, \end{aligned} \tag{2.8}$$

e a log-verossimilhança será dada por

$$\begin{aligned} l(p_z, p_{\bar{z}}) = &- n_z p_z + c_z \log(p_z) + \sum_{i \in z} x_i \log(n_i) - \sum_{i \in z} \log(x_i!) \\ &- n_{\bar{z}} p_{\bar{z}} + c_{\bar{z}} \log(p_{\bar{z}}) + \prod_{i \notin z} x_i \log(n_i) - \prod_{i \notin z} \log(x_i!). \end{aligned} \tag{2.9}$$

Maximizando a função de log-verossimilhança obtemos os estimadores $\hat{p}_{\bar{z}} = c_{\bar{z}}/n_{\bar{z}}$ e $\hat{p}_z = c_z/n_z$.

Substituindo p_0 pelo seu estimador em $L_0(p_0)$ temos

$$\sup_{p_z = p_{\bar{z}} = p_0} L(p_0) = L_0 = \prod_{i=1}^{r} \frac{e^{-n_i \frac{C}{N}}(-n_i \frac{C}{N})^{x_i}}{x_i!} = \frac{e^{-N \frac{C}{N}}(\frac{C}{N})^C \prod_{i=1}^{r} {n_i}^{x_i}}{\prod_{i=1}^{r}(x_i!)}.$$

De forma análoga, substituindo os estimadores p_z e $p_{\bar{z}}$ em $L(p_z, p_{\bar{z}})$ temos

$$\begin{aligned} \sup_{p_z > p_{\bar{z}}} L(p_z, p_{\bar{z}}) = L(z) &= \prod_{i \in z} \frac{e^{-n_i \frac{c_z}{n_z}}(-n_i \frac{c_z}{n_z})^{x_i}}{x_i!} \prod_{i \notin z} \frac{e^{-n_i \frac{c_{\bar{z}}}{n_{\bar{z}}}}(-n_i \frac{c_{\bar{z}}}{n_{\bar{z}}})^{x_i}}{x_i!} \\ &= \frac{e^{-n_z \frac{c_z}{n_z}} \frac{c_z}{n_z}^{c_z} \prod_{i \in z} {n_i}^{x_i}}{\prod_{i \in z}(x_i!)} \frac{e^{-n_{\bar{z}} \frac{c_{\bar{z}}}{n_{\bar{z}}}} \frac{c_{\bar{z}}}{n_{\bar{z}}}^{c_{\bar{z}}} \prod_{i \notin z} {n_i}^{x_i}}{\prod_{i \notin z}(x_i!)}. \end{aligned} \tag{2.10}$$

Utilizando a razão de verossimilhança definida em 2.5 temos que

$$\lambda_z = \begin{cases} \left(\frac{c_z/n_z}{C/N} \right)^{c_z} \left(\frac{c_{\bar{z}}/n_{\bar{z}}}{C/N} \right)^{c_{\bar{z}}}, & \text{se } \frac{c_z}{n_z} > \frac{c_{\bar{z}}}{n_{\bar{z}}} \\ 1, & \text{caso contrário.} \end{cases} \tag{2.11}$$

Detectando o Cluster

Como vimos, o cluster mais verossímil é a zona z para a qual a razão de verossimilhança λ_z assume seu valor máximo, dentre todas as zonas candidatas. Obviamente, para um mapa com algumas dezenas de regiões já seria computacionalmente inviável avaliar todos os subconjuntos conexos de regiões. Assim, primeiramente devemos definir quais zonas serão selecionadas como candidatas a cluster. Kulldorff (1997) utiliza o método das janelas circulares com diferentes centros e raios e, por esse motivo, é chamado de varredura circular.

Uma informação espacial necessária para construção das janelas circulares são as coordenadas do centroide de cada uma das r regiões do mapa. O centroide de uma região é um ponto qualquer em seu interior - usualmente é escolhido como o centro administrativo da região. A partir das coordenadas dos centroides é possível calcular a distância entre as regiões, definidas como as distâncias entre os centroides. Assim, partindo de uma região i, calculamos o vetor de distâncias $d = (d_{i(1)}, d_{i(2)}, \ldots, d_{i(r)})$, em que o índice (j) representa a j-ésima região mais próxima da região i, e portando $d_{i(1)} < d_{i(2)} < \ldots < d_{i(r)}$. Observe que (1)=$i$ e, portanto, $d_{i(1)} = 0$, pois é a distância entre a região i e ela mesma.

O conjunto de zonas candidatas Z é construído da seguinte forma. A primeira zona é formada apenas pela região 1. A segunda, pela região 1 e pela região mais próxima dela. A terceira, pela região 1 e pelas duas regiões mais próximas, e assim sucessivamente. As regiões são adicionadas, então, por ordem de distância, enquanto a soma de suas populações não ultrapassar um limite pré-estabelecido, p_{max}. Quando esse limite é atingido o processo de aglutinação cessa e o procedimento de construção de zonas candidatas é reiniciado, partindo-se de cada uma das demais regiões 2, 3, ..., r. O limite p_{max} é utilizado pois não faria sentido encontrar um cluster

que englobasse, digamos, mais da metade da população do mapa. Por esse motivo é comum definir p_{max} como metade da população, isto é, p_{max}=$N/2$. Nesse caso, ao invés de um grande cluster de alta incidência, o mapa deve apresentar um pequeno cluster de baixa incidência. Por esse motivo, é comum utilizar valores de p_{max} não superiores a 50% da população total do mapa. Porém, em algumas situações, a critério do analista, pode-se usar valores menores que $N/2$.

Considerando o modelo escolhido para a contagem de casos, uma vez construído o conjunto de zonas candidatas Z, para cada zona $z \in Z$ computamos o valor de λ_z. Seja $z^* = \arg\max_z \lambda_z$ a zona mais verossímil, isto é, λ_{z*}=$\max_z \lambda_z$. Para testar se z^* é um cluster, precisamos comparar o valor de λ_{z^*} com sua distribuição sob H_0. Como a distribuição exata de λ_{z^*} é desconhecida, é necessário fazer uma simulação de Monte Carlo para obter uma amostra sob H_0. A simulação de Monte Carlo é feita utilizando réplicas do mapa original para as quais o número de casos C é distribuído aleatoriamente sob H_0. O método segue os seguintes passos:

1. Distribuir, sob H_0, C casos aleatoriamente ao longo das regiões do mapa. Isto pode ser feito segundo uma distribuição Multinomial(C, $\frac{n_1}{N}$, $\frac{n_2}{N}$, ..., $\frac{n_r}{N}$), isto é, com a probabilidade de cada região sendo a sua proporção de população em relação à população total do mapa.

2. Computar a zona mais verossímil z_1^* utilizando a distribuição de casos do passo anterior e armazenar o respectivo valor da estatística de teste

3. Repetir os passos 1 e 2 M vezes, obtendo uma amostra da estatística de teste sob H_0.

4. A hipótese H_0 de ausência de clusters é rejeitada, ao nível de significância α, se $\lambda_{z^*} > \lambda_c$, onde λ_c é o valor crítico que representa o percentil $(1-\alpha) \times 100\%$ da distribuição empírica sob H_0.

2.1.2 Problema a ser resolvido

Nesse trabalho estamos interessados em encontrar a zona ótima z que maximiza a razão de verossimilhança λ_z. Em uma mapa com r regiões existe um número muito alto de zonas candidatas que podem ser testadas. Como vimos na seção anterior, Kulldorff (1997) utiliza o método das janelas circulares para encontrar as zonas candidatas. A grande desvantagem dessa abordagem é que, por conta da imposição da forma das zonas candidatas, caso o cluster verdadeiro não tenha formato pelo menos aproximadamente circular, a solução apontada pode subestimar ou superestimar o cluster verdadeiro. No primeiro caso, apenas uma parte do cluster verdadeiro é detectada, deixando de incluir na solução regiões de alto risco relativo (falso negativo). Por outro lado, para incluir todas as regiões de alto risco pode ser necessário incluir outras regiões de baixo risco (falso positivo). Além disso, o poder de detecção da varredura circular pode ser severamente afetado quando o formato do cluster verdadeiro diverge substancialmente da forma circular.

Esse trabalho propõe, através do paradigma de otimização por população, encontrar a zona ótima z com um formato arbitrário, pois o método proposto por Kulldorff (1997) avalia apenas cluster com formato circular. Nas próximas seções iremos detalhar o que são algoritmos de otimização por população.

2.1.3 Otimização por Populações

A otimização, sob o ponto de vista prático, se trata do conjunto de métodos capazes de determinar as melhores configurações possíveis para a construção ou o funcionamento de sistemas de interesse para o ser humano. Entre os vários métodos de otimização existentes, estamos interessados nos métodos de Otimização por Populações. Ao contrário de métodos como o Método do Gradiente e o Método de Newton-Raphson, esses métodos trabalham, em cada iteração, com um conjunto de soluções, chamadas também de *indivíduos*. Embora exista uma grande quantidade de algoritmos de otimização por populações, aqui abordaremos, como exemplos, o

Simulated Annealing (SA) e os *Algoritmos Genéticos* (AG's), além do *Particle Swarm Optimization* (PSO) que é a ferramenta central deste trabalho.

De maneira geral podemos definir os métodos de populações conforme o Algoritmo 1.

Algoritmo 1: Algoritmo de populações

Output: P: População final

$i \leftarrow 1$

$P_i \leftarrow P_0$

while *não ocorrer critério de parada* **do**

  $A_i \leftarrow f(P_i)$;

  $P_{i+1} \leftarrow \text{G}(A_i, P_i, v_i)$;

  $i \leftarrow i + 1$;

end

P_0 é chamada de *população inicial*, ou seja, é o conjunto inicial de indivíduos (soluções). Esse conjunto é avaliado segundo a função-objetivo $f(\cdot)$, que é a função que temos interesse em otimizar, e o resultado dessa avaliação é armazenado em A_i. Note que A_i é um vetor, sendo que cada entrada desse vetor é a avaliação de um indivíduo de P_i. Após P_i ser avaliado através da função-objetivo $f(\cdot)$, obtém-se o próximo conjunto de indivíduos, ou seja, a próxima população. Isso é feito por meio da função $G(\cdot, \cdot, \cdot)$, que é um conjunto de regras que define a mudança da população corrente P_i para a próxima população P_{i+1}. Essas regras geralmente dependem da população corrente, P_i, e de sua avaliação, A_i, além de um conjunto de variáveis aleatórias, aqui representadas pelo vetor v_i. Conforme a população evolui no decorrer das iterações, espera-se que o conjunto de indivíduos que formam essa população convirja para uma região próxima do ótimo da função-objetivo $f(\cdot)$.

Nesse trabalho estamos interessados em encontrar a zona ótima z que maximiza a razão de verossimilhança λ_z. Logo, nesse contexto, cada indivíduo será uma zona e a população é um conjunto de zonas.

2.1.4 Simulated Annealing

O *Simulated Annealing* é um método de otimização voltado para resolver problemas de otimização combinatória. O método foi inspirado em conceitos da termodinâmica, especificamente no processo utilizado para fundir metais. Traduzido para o português, *annealing* significa recozimento. Nesse processo o metal é aquecido a uma temperatura elevada e depois é resfriado lentamente até que no final tenha uma massa homogênea típica dos metais.

Introduzido por Kirkpatrick, Gelatt e Vecchi (1983), o método consiste em otimizar uma função-objetivo $f(x)$, em que x pertence a um espaço de soluções enumerável. O método é realizado por níveis e em cada nível é simulada a temperatura no resfriamento. Escolhendo um ponto qualquer no espaço de soluções, vários pontos em sua vizinhança são gerados e são calculados os respectivos valores da função-objetivo. Pontos na vizinhança que apresentem melhora na função-objetivo são prontamente aceitos. Cada ponto que não apresente melhora na função-objetivo é aceito ou rejeitado por meio de uma probabilidade de aceitação que diminui a cada nível do processo.

O Algoritmo 2 apresenta os passos para esse método. Ele foi construído para minimizar uma função-objetivo f, em que T_i representa a temperatura no nível i e P_i representa a quantidade de pontos que serão gerados nesse nível. Esses pontos serão gerados na vizinhança de um ponto p que é escolhido aleatoriamente no espaço de soluções. Procurando evitar uma convergência precoce para um mínimo local, o algoritmo inicia com o valor de T_0 relativamente alto, de forma que mesmo os pontos que não melhoram a função-objetivo podem ser aceitos com mais facilidade. O valor da temperatura é diminuído a cada nível, de modo que na convergência o algoritmo se torna determinístico.

Algoritmo 2: Algoritmo Simulated Annealing

$p \leftarrow$ escolher um ponto aleatoriamente no espaço de soluções
$i \leftarrow 1$
$T_i \leftarrow T_0$
$P_i \leftarrow P_0$
while *não ocorrer critério de parada* **do**
    for $j \leftarrow 1$ **to** P_i **do**
        gerar w na vizinhança de p;
        if $f(w) \leq f(p)$ **then**
            $p \leftarrow w$;
        else if *Uniforme*$(0,1)$<$\exp\left(\frac{f(p)-f(w)}{T_i}\right)$ **then**
            $p \leftarrow w$;
        end
        $i \leftarrow i + 1$;
    end
    Calcular P_i e T_i;
end

2.1.5 Algoritmos Genéticos

Os *Algoritmos Genéticos* (AG's) são métodos computacionais de otimização cuja ideia central é inspirada na genética e na evolução natural dos seres vivos. Nos AG's o conjunto de indivíduos que formam a população são modificados a cada iteração seguindo regras probabilísticas idealizadas através de metáforas biológicas. Assim, espera-se que com o passar das iterações a população "evolua", resultando em uma população formada por soluções mais bem "adaptadas", isto é, soluções com melhores avaliações de função-objetivo. Existem três operações básicas para que um algoritmo de população seja caracterizado como um AG: cruzamento,

mutação e seleção.

- Cruzamento: Combina as características de dois ou mais indivíduos (pais) gerando novos indivíduos (filhos).
- Mutação: Utilizando as características de um indivíduo, gera outro indivíduo através de uma perturbação aleatória.
- Seleção: Utilizando a avaliação da função-objetivo dos indivíduos da população, indivíduos mais bem adaptados têm maior probabilidade de serem selecionados para realizar cruzamento, gerando filhos, e portanto, passarem suas características para a próxima geração. Por outro lado, indivíduos mal avaliados têm poucas chances de serem selecionados, levando-os à extinção, juntamente com suas características.

O Algoritmo 3 apresenta o esquema de um AG utilizando essas três operações.

Algoritmo 3: Algoritmo Genético

Output: P: População final

1 $i \leftarrow 1$

2 $P_i \leftarrow P_0$

3 **while** *não ocorrer critério de parada* **do**

4 selecionar P_{i+1} a partir de P_i;

5 cruzar e mutar P_{i+1};

6 avaliar P_{i+1};

7 $i \leftarrow i + 1$;

8 **end**

A Figura 2.2 apresenta as curvas de nível e um exemplo de cruzamento em um AG para um problema com duas variáveis contínuas. As soluções x_1 e x_2 são os pais. Uma das formas mais comuns de se operar cruzamento é gerar os filhos sobre o segmento de reta que une os pais (linha pontilhada). Usualmente a geração dos filhos sobre esse segmento é feita a partir de uma escolha uniforme sobre o segmento. Outras alternativas são a geração polarizada (em que a escolha é feita ao acaso mas não-uniforme, atribuindo maior probabilidade para soluções mais próximas do pai mais bem avaliado) ou a geração da solução ótima sobre o segmento.

Um AG pode conter outras operações além dessas três apresentadas. Um operador que é quase sempre utilizado é o elitismo, que garante que a melhor solução sempre é selecionada. Outro operador bastante utilizado é a operação de nicho. O operador de nicho é construído separando os indivíduos da população que, por alguma razão, não competem entre si. Sendo assim, a seleção da próxima população é feita dentro de cada nicho.

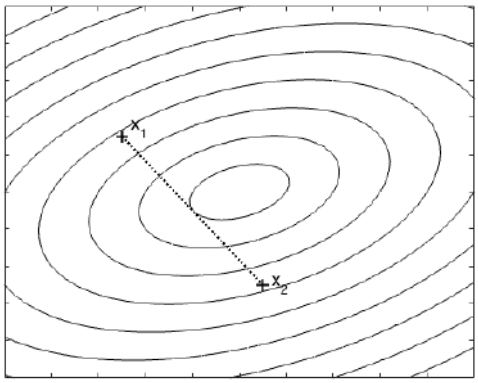

Figura 2.2: Curvas de nível da função-objetivo e cruzamento entre duas soluções x_1 e x_2. A partir dos pais, um filho pode ser gerado escolhendo-se um ponto sobre o segmento de reta que os une. A escolha pode ser ao acaso, polarizada ou otimizada.

2.1.6 Particle Swarm Optimization, PSO

O *Particle Swarm Optimizaton* (PSO), traduzido como otimização por enxame de partículas, é um algoritmo de otimização estocástico baseado em populações. Sua estratégia é inspirada no voo dos pássaros e movimentos de cardume de peixes e foi introduzido por Kennedy e Eberhart (1995). O PSO se inicia com a geração da população de partículas (indivíduos) no espaço de busca da função-objetivo. As posições da cada indivíduo são as possíveis soluções do problema.

A cada iteração, a busca por uma posição ideal é realizada atualizando as velocidades e posições das partículas. A velocidade de cada partícula é atualizada utilizando duas posições específicas, p_j e g, sendo p_j a melhor posição em que a j-ésima partícula já esteve até a iteração atual e g é a melhor posição já ocupada por qualquer partícula até a iteração atual. Considerando uma função-objetivo de n variáveis contínuas representadas por um vetor $X = (x_1, \ldots, x_n)$, a velocidade da $j-$ésima partícula na iteração $i + 1$ é dada por

$$\begin{aligned} V_j^{(i+1)} &= V_j^{(i)} + c_1 u_1^{(i)}(p_j^{(i)} - X_j^{(i)}) + c_2 u_2^{(i)}(g^{(i)} - X_j^{(i)}) \\ & \quad j = 1, \ldots, n. \end{aligned} \tag{2.12}$$

em que $V_j^{(i+1)}$ é a velocidade da $j-$ésima partícula na i-ésima iteração, $X_j^{(i)}$ é a posição $j-$ésima partícula na i-ésima iteração, $u_1^{(i)}$ e $u_2^{(i)}$ são valores gerados a partir de uma distribuição Uniforme(0,1). Além disso, c_1 e c_2 são constantes positivas, chamadas de coeficientes de aceleração, que controlam a influência de p_j e g no processo de busca.

A partir dessa velocidade, a posição da $j-$ésima partícula na iteração $i+1$ é dada por

$$X_j^{(i+1)} = X_j^{(i)} + V_j^{(i+1)}.$$

Notemos que, a nova posição da partícula é simplesmente a posição atual acrescida de sua velocidade, que nada mais é que o vetor resultante de três termos:

1. $V_j^{(i)}$, a velocidade na iteração anterior,

2. $c_1 u_1^{(i)}(p_j^{(i)} - X_j^{(i)})$, um vetor que aponta da posição atual para a melhor posição já ocupada pela partícula, e

3. $c_2 u_2^{(i)}(g^{(i)} - X_j^{(i)})$, um vetor que aponta da posição atual para a melhor posição já ocupada por qualquer partícula.

Assim, o movimento de cada partícula é regido por três componentes que, simultaneamente, tendem a fazer com que a partícula (1) continue voando na direção atual (*momentum*) (2) retorne à posição em que apresentou o melhor valor da função-objetivo (nostalgia) e (3) voe em direção à melhor posição já alcançada por qualquer partícula (gula).

A Figura 2.3 apresenta as posições ocupadas por uma partícula nas iterações $1-4$. Assim, é notável que a melhor posição ocupada por essa partícula (até a iteração 4) foi na iteração 2. Além disso, a melhor posição g ocupada pelo conjunto de todas as partículas é apresentada. A partir daí, a nova posição da partícula pode ser calculada como a soma da posição corrente à nova velocidade, que é a resultante dos três termos.

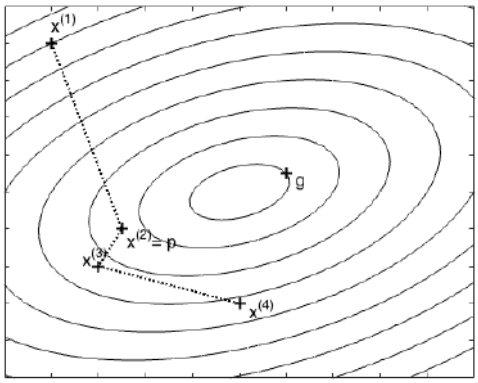

Figura 2.3: Curvas de nível da função-objetivo e movimento de uma partícula ao longo de quatro iterações. A melhor posição ocupada por ela foi na iteração 2 (ponto p) e a melhor posição ocupada por qualquer das partículas é representada por g.

2.2 Algoritmo proposto

O presente trabalho propõe identificar e detectar clusters através de um algoritmo baseado no PSO. Fazendo um paralelo com o algoritmo PSO, cada zona z é uma partícula que deve se movimentar sobre o espaço de busca até que se atinja o máximo da estatística de teste λ_z. Obviamente, nesse caso os conceitos de velocidade e posição devem ser adaptados, já que o espaço de busca é discreto e, em geral, possui alta dimensionalidade. De fato, agora a codificação de uma partícula deixa de ser a de um vetor $(x_1, \ldots, x_n)$ composto por n variáveis contínuas e passa a ser o vetor $(x_1, \ldots, x_r)$, em que $x_i = 1$ se a $i-$ésima região pertence à zona z, e zero caso contrário, $i = 1, \ldots, r$.

Para inicializarmos o algoritmo, serão necessárias as seguintes informações.

- População: Vetor contendo o tamanho populacional de cada região.
- Casos: Vetor contendo o número de casos em cada região.
- Centroide: as coordenadas do centroide de cada região.
- Proporção populacional: Proporção máxima do tamanho populacional p_{max} que uma zona

z poderá atingir para iniciar o algoritmo.

- Matriz de vizinhança: matriz W em que $w_{ij} = 1$ se a região i faz fronteira com a região j, e $w_{ij} = 0$ caso contrário.
- Proporção de voo (cruzamento): Proporção máxima que uma zona z irá voar em direção ao seu pivô na iteração corrente.
- Probabilidade de mutação: Probabilidade de uma zona sofrer mutação após o "voo" (cruzamento).
- Número de iterações: Número máximo de iterações do algoritmo.
- Penalização α: intensidade da penalização espacial.

A seguir iremos detalhar como o algoritmo é inicializado e como foram implementadas as demais operações.

2.2.1 Inicialização das partículas

O conjunto P_0 de partículas (zonas) utilizado para iniciar o algoritmo é escolhido a partir das zonas circulares geradas pelo método descrito por Kulldorff (1997), isto é, as zonas obtidas pelas janelas circulares. De cada conjunto de zonas circulares obtidas a partir de janelas centradas em uma região, escolhemos a de maior valor de λ_z. Em outras palavras, de todas as zonas com centro na região i, escolhemos a de maior valor de razão de verossimilhança, $i = 1, \ldots, r$. Assim, a população inicial P_0 de partículas é composta por r zonas, o mesmo número de regiões no mapa, e todas as regiões pertencem a pelo menos uma partícula.

2.2.2 Movimentação de partículas

Enquanto a movimentação de partículas em um espaço contínuo é perfeitamente compreensível, essa movimentação em um espaço discreto não é trivial. Por exemplo, a simples troca

de *bits* entre duas soluções espacialmente separadas deve gerar, na grande maioria das vezes, soluções espúrias, formadas por conjuntos desconexos de regiões espalhadas sobre o mapa. Para evitar essas soluções inviáveis, é necessário implementar uma maneira de fazer com que uma solução caminhe em direção a outra gerando outra solução factível, isto é, espacialmente conexa.

Para tanto, introduzimos o conceito de *nicho*. Seja P_i a população corrente na i-ésima iteração do algoritmo e considere a zona $z \in P_i$ que corresponde ao maior valor de λ_z. A essa zona chamaremos pivô. Agora, identificamos todas as zonas que possuem alguma interseção com a zona pivô. Chamaremos esse conjunto de zonas formado pela zona pivô e pelas demais zonas com as quais ela possui interseção de nicho, o qual será denotado por $\mathbf{n_1}$. Agora seja $P_i' = P_i - \mathbf{n_1}$ o conjunto das zonas que não estão presentes no nicho $\mathbf{n_1}$. A partir de P_i' construímos um novo nicho $\mathbf{n_2}$ identificando a zona pivô (isto é, a zona mais verossímil de P_i') e as zonas que fazem interseção com ela. O processo é repetido até que todas as zonas façam parte de algum nicho. Note que é possível que um nicho contenha apenas uma zona.

Com a intenção de emular o voo das partículas, para cada nicho faremos as zonas pertencentes a ele "caminharem" na direção do respectivo pivô. A forma com que esse "voo" é feito garante que apenas soluções conexas são geradas após o voo. Além disso, espera-se que dessas zonas híbridas (que possuem características tando da partícula que está voando quanto do pivô) possam eventualmente surgir algumas com maior valor de λ_z e também com formato espacial arbitrário e não apenas circular.

Consideremos z_1 e z_2 duas zonas que possuem alguma interseção. Sejam n_1 o número de regiões de z_1 que não estão na interseção com z_2, n_2 o número de regiões de z_2 que não estão na interseção com z_1 e $n = n_1 + n_2$. Suponha que desejamos fazer z_1 voar em direção a z_2. Esse procedimento é feito removendo regiões de z_1 e acrescentando regiões de z_2. Então, cada região removida de z_1 ou acrescentada de z_2 representa $(1/n) \times 100\%$ do total de "passos" possíveis para se completar o caminho de z_1 para z_2. Assim, se por exemplo quiséssemos fazer z_1 voar 100% em direção a z_2, teríamos que remover todas as n_1 regiões de z_1 e acrescentar todas as

n_2 regiões de z_2, obtendo ao fim uma cópia exata da zona z_2. Porém, para que o processo de remoção e adição de regiões não gere zonas desconexas, essas operações devem ser feitas numa ordem que respeite a conectividade da zona.

Para ilustrar esse procedimento, vamos analisar a Figura 2.4 começando da esquerda para direita e de cima para baixo. No Gráfico A temos a zona z_1, em amarelo, composta por 7 regiões, e no Gráfico B temos a zona z_2, em azul, composta por 9 regiões. No Gráfico C temos z_1 e z_2 sobrepostos e as regiões que representam a interseção estão na cor verde. Ainda em relação ao Gráfico C, é possível perceber que n_1=4, n_2=6 e, consequentemente, n=10. Assim cada região removida de z_1 ou acrescentada de z_2 representa 10% de caminhada de z_1 em direção a z_2.

Para fazermos z_1 voar em direção a z_2, a primeira coisa que devemos fazer é atribuir níveis para as regiões que fazem parte da interseção, para as regiões que fazem parte apenas de z_1 e para as regiões que fazem parte apenas de z_2. Esses níveis serão atribuídos da seguinte maneira: as regiões da interseção recebem nível 0; as regiões que fazem parte apenas de z_1 recebem os níveis de forma crescente de 1 até n_1; e as regiões que fazem parte apenas de z_2 recebem os níveis de forma crescente de -n_2 até -1.

A região de z_1 que irá receber o nível 1 deve ser alguma região que faz fronteira com (isto é, é vizinha a) alguma região que recebeu nível 0. Percebemos que três regiões de z_1 são vizinhas das regiões com nível 0, então cada uma delas tem probabilidade 1/3 de receber o nível 1. Para uma região receber o nível 2, iremos verificar quais são as regiões vizinhas das regiões com nível 0 ou 1, e cada uma dessas regiões poderá receber o nível 2 com mesma probabilidade. Esse processo é repetido até que todas as regiões de z_1 tenham recebido um nível. Os níveis de z_2 são atribuídos analogamente aos níveis de z_1. Utilizamos níveis negativos apenas para evitar a confunsão com os níveis de z_1. As regiões com seus respectivos níveis atribuídos estão no Gráfico D.

Uma vez que todas regiões receberam os níveis adequadamente, iremos adicionar e retirar regiões de z_1 até que se tenha atingido a proporção p desejada. Como vimos n=10, então cada região removida de z_1 ou acrescentada de z_2 representa 10% do caminho. O caminho é per-

corrido alternando-se entre a remoção da região de maior nível presente na solução corrente (regiões amarelas e verdes) e a adição da região de menor (mais negativo) nível que ainda não está presente na solução corrente (regiões azuis), enquanto houver níveis positivos e negativos. Caso os níveis positivos acabem, os passos seguintes são dados somente acrescentando as regiões com níveis negativos, e caso os níveis negativos acabem, os passos seguintes são dados apenas pela remoção das regiões com nível positivo.

Observando o Gráfico E da Figura 2.4, retiramos a região de maior nível (4) de z_1, o que representa 10% do caminho. No Gráfico F acrescentamos a região de menor nível (-6) de z_2, o que representa mais 10%. Depois repetimos esse processo de retirar de z_1 e acrescentar de z_2 até que se tenha atingido a proporção desejada. Por exemplo, se $p = 80\%$ a zona resultante seria a observada no Gráfico L, formada pelas regiões de cor verde. A remoção de regiões de z_1 do maior para o menor nível, e a adição de regiões de z_2 do nível mais negativo para o menos negativo garantem a conectividade das soluções geradas, uma vez que a atribuição dos níveis é feita considerando-se as vizinhanças.

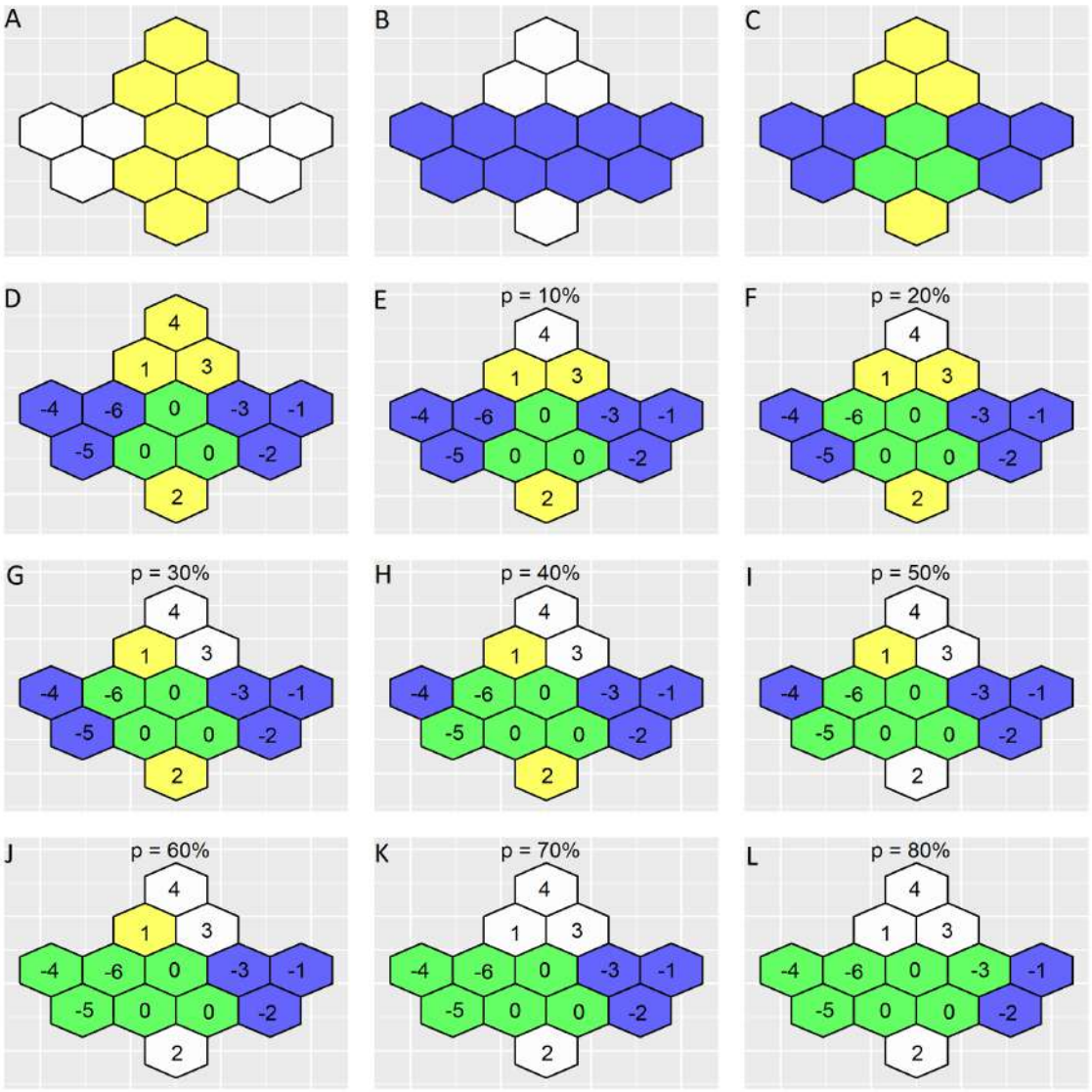

Figura 2.4: Zona voando em direção a outra.

Após todas as zonas caminharem em direção aos seus pivôs, algumas delas podem sofrer mutação. Isso é feito com o intuito de introduzir maior aleatoriedade ao processo, o que deve evitar a convergência precoce do algoritmo para máximos locais.

2.2.3 Mutação

A mutação consiste em adicionar uma região que é vizinha da zona a ser mutada, ou retirar uma região dessa zona. Para cada partícula da população corrente, a mutação irá ocorrer com uma probabilidade definida previamente. Caso ocorra a mutação, a probabilidade de adicionar ou retirar uma zona será de 0,5. Considere o exemplo da Figura 2.5, começando da esquerda para direita e de cima para baixo.

No Gráfico A temos uma zona z, em amarelo, formada por 12 regiões. No Gráfico B temos, em roxo, todas as regiões que são candidatas para fazer parte da zona z, caso o sorteio tenha decidido pela mutação que adiciona uma região. Dentre elas, uma é escolhida aleatoriamente e é acrescentada à zona z.

Caso a mutação seja do tipo que exclui uma região da zona z é preciso tomar um cuidado, pois só poderá ser excluída alguma região que não comprometa a conectividade da zona. Nos Gráficos C e D podemos observar que a remoção da região 102 ou da região 115 levaria a soluções desconexas. Com exceção dessas duas regiões, cada uma das outras tem a mesma probabilidade de ser escolhida para ser retirada da zona z.

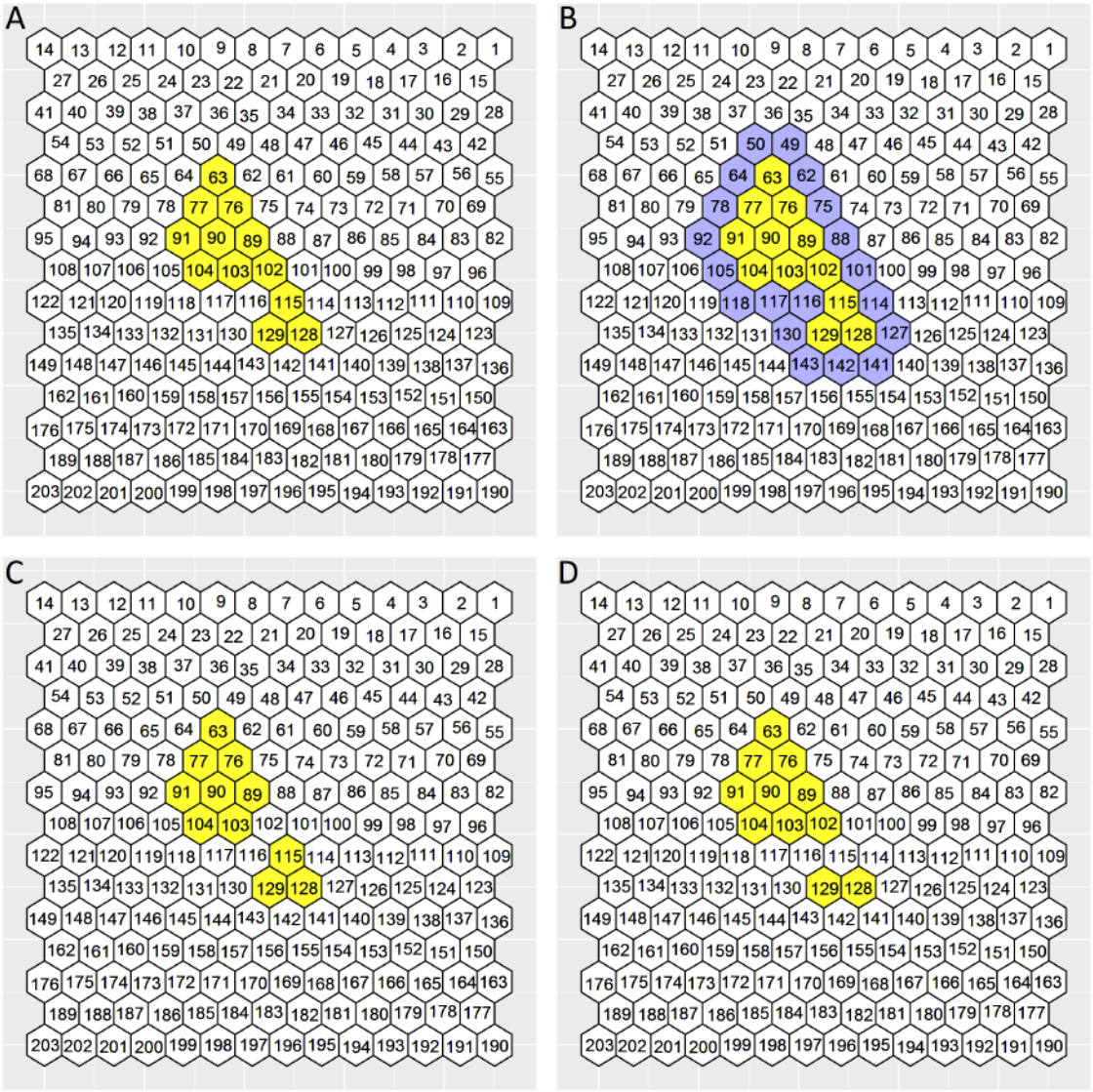

Figura 2.5: Mutação.

2.2.4 Penalização

Além de avaliar cada solução em relação à razão de verossimilhança, é necessário introduzir uma correção (penalização) espacial para evitar que zonas demasiadamente irregulares sejam consideradas. Essa penalização é necessária pois, sem ela, é possível que o algoritmo acabe apontando como solução uma zona em "forma de polvo" cujos tentáculos se espalham por todo o mapa, unindo regiões de alta incidência, mas cuja informação geográfica é inútil. Um cluster,

embora possa apresentar formato irregular, deve ser capaz de apontar uma região razoavelmente compacta do espaço para que o pesquisador possa identificar as causas do aumento da incidência naquela área do mapa. Assim, a penalização é mais forte quanto mais irregular é a zona, de forma que zonas muito irregulares devem ter seus valores de λ_z mais penalizados. Por outro lado, uma zona com formato irregular deve persistir mesmo com a penalização caso seu valor de razão de verossimilhança seja suficientemente grande.

Várias penalizações espaciais foram propostas na literatura. A compactação geométrica Cg_z proposta por Duczmal, Kulldorff e Huang (2006) é definida como a área da zona z dividida pela área do círculo com o mesmo perímetro que o fecho convexo da zona z. Essa função penaliza candidatos a cluster com formato altamente irregular e é definida como

$$Cg_z = \frac{4\pi A_z}{3H_z^2},$$

onde A_z e H_z são, respectivamente, a área e o perímetro do fecho convexo da zona z. Outra penalização é a de dispersão d_z, proposta por Oliveira et al. (2019), definida como

$$d_z = \frac{2(d_1 d_2)}{(d_1 + d_2)},$$

onde d_1 é a diferença entre máximo e o mínimo de x_i, e y_i é a diferença entre o máximo e o mínimo de y_i, sendo (x_i, y_i) as coordenadas dos centroides de cada região da zona z.

Pela simplicidade de cálculo e por sua eficiência, escolhemos trabalhar com a penalização proposta por Yiannakoulias, Rosychuk e Hodgson (2007), definida como

$$k_z = \frac{e_z}{3 * (r_z - 2)},$$

em que e_z é o número de conexões entre as regiões da zona z e r_z é o número de regiões da zona z. Esse método penaliza a zona z proporcionalmente na razão entre número de conexões da zona z e o número total de possíveis de conexões que uma zona z pode ter. Com isso, k_z

será um valor entre 0 e 1, e quanto mais próximo de 1 menor será o impacto da penalização. Então quanto maior o número de conexões existentes na zona z mais compacta será essa zona e consequentemente resultará em uma menor penalização.

Assim, ao invés de trabalhar diretamente com o valor da razão de verossimilhança λ_z, trabalharemos com a quantidade $\lambda_z^{k_z^\alpha}$, com $\alpha \geq 0$. O parâmetro α representa a força da penalização, de modo que se $\alpha = 0$, então não há penalização alguma e a penalização torna-se mais forte à medida que o valor de α aumenta. Assim, quanto maior o valor de α maior é a tendência de que soluções irregulares sejam severamente penalizadas e que apenas soluções compactas permaneçam.

2.2.5 População interna e externa

A movimentação de cada solução é feita sempre em direção ao respectivo pivô. Essa movimentação pode resultar em uma solução melhor ou pior, mas como o pivô não se movimenta, não há o risco de que essa solução - que é a melhor daquele nicho - seja perdida. Por outro lado, caso o pivô sofra mutação, é possível que a nova solução seja pior. Nesse caso, aquele nicho passaria a ter um pivô pior que o anterior e, mais grave, seria possível que o algoritmo eventualmente perdesse a sua melhor solução já encontrada.

Para que não corramos o risco de perder a melhor solução já encontrada pelo algoritmo, ele trabalha com duas populações, chamadas população interna e população externa. Seja P_{int_i} a população interna e P_{ext_i} a população externa na i-ésima iteração. Todas as operações são realizadas sobre a população interna, cujos pivôs podem ser perdidos. Em seguida, a população externa é atualizada com as novas soluções, mais os antigos pivôs. A partir daí são obtidos os novos nichos, que podem ter os pivôs alterados em função do surgimento de eventuais soluções melhores.

2.2.6 Resumo do Algoritmo

O algoritmo pode ser resumido nos seguintes passos:

1. Gerar o conjunto inicial Z de zonas z, composto por r zonas, $Z = \{z_1, z_2, \ldots, z_r\}$. Armazene na população interna e na população externa.

2. Avaliar cada solução, isto é, calcular o valor de λ_z para cada zona z.

3. Na população interna, escolher a zona com maior valor de λ_z (pivô) e verificar quais zonas fazem interseção com ela na população externa. Armazene essas regiões.

4. Na população interna, escolher, dentre as zonas que não foram selecionadas anteriormente, a zona com maior valor de λ_z (pivô) e verificar quais zonas fazem interseção com ela na população externa. Armazene essas regiões.

5. Repita o passo 4 até que todas as zonas tenham sido escolhidas. Os conjuntos de zonas resultantes dos passos $3 - 5$ serão denominados nichos.

6. Em cada nicho fazer as zonas "voarem", com uma proporção p, em direção à zona com maior valor de λ_z.

7. Operar mutação nas novas zonas.

8. Atualize a população externa como sendo a população corrente mais os pivôs anteriores.

9. Repita os passos $2 - 8$ n vezes.

Capítulo 3

Simulação e análise de desempenho

Nesse capítulo é descrito como foram feitas as simulações e a análise de desempenho do algoritmo. Para testar o método e compará-lo ao Scan Circular, clusters artificiais com formas variadas foram gerados em um mapa formado por 203 regiões hexagonais. Como o mapa é formado por regiões regularmente espaçadas, o centroide de cada uma delas foi escolhido ao acaso de modo a evitar empates ao se calcular as distâncias. Sendo conhecida a localização exata de cada cluster artificial, podemos verificar, para cada método, a capacidade de detecção e identificação correta do cluster. A análise de desempenho foi feita utilizando os conceitos de sensibilidade, PPV (*positive predictive value*) e poder do teste. Nas simulações e análises são considerados 4 cenários diferentes, cada um com um formato específico, e para cada cenário são comparados os resultados utilizando o scan Circular e o algoritmo proposto com os seguintes valores de força de penalização α: 0, $0,25$, $0,5$ e 1.

3.1 Cenários

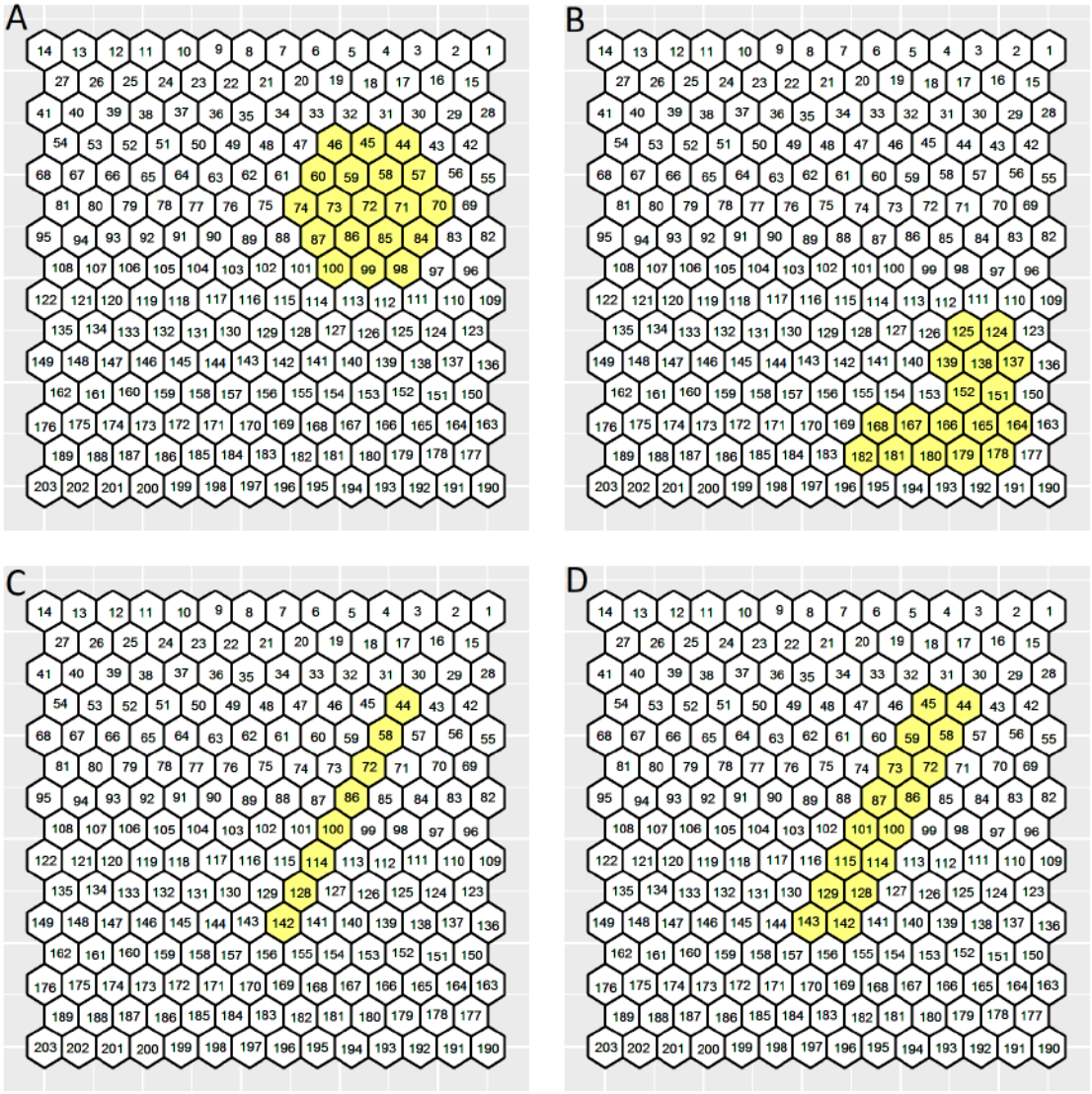

Figura 3.1: Figuras com 4 cluters reais e identificador de cada região.

Na Figura 3.1 temos 4 mapas, cada um com um cluster diferente, destacado na cor amarela. O primeiro cenário (Gráfico A) é composto por um cluster de formato circular com 19 regiões (46, 45, 44, 60, 59, 58, 57, 74, 73, 72, 71, 70, 87, 86, 85, 84, 99, 98, 97). O segundo cenário (Gráfico B) é composto por um cluster em forma de "L" deitado, que faz com que ele ainda seja relativamente compacto, mas não tanto quanto o primeiro. É formado por 17 regiões (125, 124, 139, 138, 137, 152, 151, 168, 167, 166, 165, 164, 182, 181, 180, 179, 178). O terceiro cenário (Gráfico C) é composto por um cluster que tem o formato menos compacto dentre os 4 cenários. Sua forma linear poderia emular, por exemplo, um cluster de uma doença que se espalha ao longo de um rio. É formado por 8 regiões (44, 58, 72, 86, 100, 114, 128, 142). O quarto e último cenário (Gráfico D) é composto por um cluster que tem o formato parecido com o cluster do terceiro cenário, porém um pouco mais largo. Suas 16 regiões são 44, 58, 72, 86, 100, 114, 128, 142, 45, 59, 73, 87, 101, 115, 129, 143.

Em todos os 4 cenários a população de cada região do mapa é homogênea, de tamanho n_i =1000, $i = 1, \ldots, 203$. Com isso a população total do mapa é $N = 203.000$. O número total de casos é $C = 406$ e o número x_i de casos em cada região é gerado artificialmente de tal forma que as regiões dentro do cluster têm um risco relativo maior que as regiões fora do dele. Os riscos relativos foram estabelecidos para que a hipótese nula fosse rejeitada com probabilidade 0,999 quando usado um teste binomial padrão, caso soubéssemos a exata localização do cluster a priori. Para melhor entendimento de como é feita a simulação para a geração dos cenários, mais informações podem ser obtidas no trabalho de Kulldorff, Tango e Park (2003).

3.2 Simulação

A simulação para avaliação dos algoritmos seguiu os seguintes passos:

1. Distribuir os $C = 406$ casos aleatoriamente sob H_0 ao longo das regiões do mapa. Isso pode ser feito gerando uma amostra da distribuição Multinomial ($C, \frac{n_1=1000}{N=206.000}, \frac{n_2=1000}{N=206.000}, \ldots, \frac{n_{203}=1000}{N=206.000}$).

2. Encontrar a zona mais verossímil z^*, utilizando a distribuição de casos do passo anterior e armazenar o respectivo valor da estatística de teste.

3. Repetir os passos 1 e 2 $M = 1000$ vezes, obtendo uma amostra da estatística de teste sob H_0

4. Computar o valor crítico dado pelo percentil 95% da distribuição da estatística de teste sob H_0.

5. Distribuir os $C = 406$ casos aleatoriamente sob H_a, utilizando os riscos relativos calculados conforme mencionado na seção anterior.

6. Encontrar a zona mais verossímil z^*, utilizando a distribuição de casos do passo anterior e armazenar o respectivo valor da estatística de teste.

7. Repetir os passos 5 e 6 $S = 1000$ vezes.

Para cada simulação sob H_a podemos verificar se o algoritmo detectou um cluster significativo comparando o valor da estatística de teste da zona mais verossímil para aquela simulação com o valor crítico obtido sob H_0. Como sabemos, os casos sob H_a foram gerados artificialmente para cada um dos 4 cenários contendo, de fato, em cada geração aleatória, um cluster. Isso é feito 1000 vezes para cada cenário, de modo que será possível verificar nessas 1000 simulações, quantas vezes o algoritmo detectou um cluster significativo no mapa. A análise do desempenho de cada método será avaliada segundo as métricas apresentadas na próxima seção.

3.3 Análise de desempenho

Para uma determinada simulação sob H_a, seja CE o cluster estimado (isto é, o cluster mais verossímil para aquela simulação) e seja CV o cluster verdadeiro. A sensibilidade é calculada utilizando a razão entre o tamanho da população das regiões na interseção entre CE e CV e o tamanho da população nas regiões de CV. Isto é,

$$\text{Sensibilidade} = \frac{\text{Pop}(CE \cap CV)}{\text{Pop}(CV)}, \tag{3.1}$$

em que Pop representa a população. Observando a equação 3.1 percebemos que ela consegue medir o quanto o algoritmo estimou corretamente as regiões que estavam no cluster real. Se o valor da sensibilidade for igual a 1, significa que todas as regiões do cluster real estão no cluster estimado. Porém ela não leva em consideração regiões que não estão no cluster real, então não podemos apenas analisar a sensibilidade, pois no cluster estimado pode conter várias regiões que não pertencem ao cluster real. Em função disso utilizamos também o PPV.

O PPV é calculado utilizando a razão entre o tamanho da população nas regiões da interseção entre CE e CV e o tamanho da população do CE. Ou seja,

$$\text{PPV} = \frac{\text{Pop}(CE \cap CV)}{\text{Pop}(CE)}. \tag{3.2}$$

Observando a equação 3.2 percebemos que ela mede o quanto do cluster estimado realmente pertence ao cluster verdadeiro. Se o valor do PPV for muito abaixo de 1, significa que o cluster estimado está detectando muitas regiões que não pertencem ao cluster verdadeiro. Podemos concluir que se tanto a sensibilidade quanto o PPV forem 1 isso significa que o algoritmo detectou de forma exata o cluster verdadeiro.

Para cada cenário foram executadas, portanto, 1000 simulações sob H_a. Para as simulações em que foi detectado um cluster (isto é, a zona mais verossímil foi significativa) computamos os valores da sensibilidade e do PPV. Ao fim das simulações, calculamos os valores médios

da sensibilidade e do PPV. Além disso, calculamos o poder do teste, que é a probabilidade de rejeitar se H_0 quando H_0 é falsa, ou seja, é a probabilidade do algoritmo detectar um cluster em um mapa que sabemos, a priori, que existe um cluster. Logo, o poder do teste pode ser estimado pela razão entre o número de vezes que H_0 foi rejeitada e o número de simulações.

3.4 Resultados

Para cada um dos quatro cenários executamos cinco algoritmos: Scan Circular e o algoritmo proposto para diferentes valores de α (0, 0.25, 0.5 e 1).

Os parâmetros escolhidos para o PSO foram os seguintes:

1. Proporção máxima da população dentro de uma zona: $0,15$
2. Proporção de voo (cruzamento): $0,50$
3. Probabilidade de mutação: $0,20$
4. Número de iterações: 100
5. α: 0, 0.25, 0.5 e 1.

A Tabela 3.1 apresenta os valores relativos a poder, sensibilidade e PPV para cada um dos métodos em cada um dos quatro cenários.

Analisando os dados que são observados na Tabela 3.1 percebemos que para o cenário 1, que tem um formato circular, o melhor desempenho foi do Scan Circular, tanto em relação ao poder quanto à sensibilidade e PPV. Quanto ao algoritmo proposto é possível perceber que quando o valor da penalização α aumenta a sensibilidade e PPV também aumentam. Isso é de se esperar pois a penalização é aplicada de tal forma que os clusters sejam mais compactos para maiores valores de α. Para o algoritmo proposto o melhor caso é quando α = 1, apresentando uma sensibilidade maior que o PPV. Isso significa que ele está encontrando regiões que pertencem ao cluster mas também está considerando outras regiões que não pertencem ao cluster verdadeiro.

Tabela 3.1: Resultados com PPV sensibilidade e poder do teste para os 4 cenários.

	Cenário 1			Cenário 2		
Penalização	PPV	Sens	Poder	PPV	Sens	Poder
circular	0.8975	0.8615	0.9510	0.6942	0.6915	0.8970
$\alpha = 1$	0.7663	0.8240	0.9250	0.7418	0.7655	0.8640
$\alpha = 0.5$	0.7412	0.7737	0.8970	0.7314	0.7717	0.7990
$\alpha = 0.25$	0.7132	0.7176	0.9460	0.7207	0.7236	0.8790
$\alpha = 0$	0.6881	0.6910	0.8740	0.7054	0.7012	0.7770
	Cenário 3			Cenário 4		
Penalização	PPV	Sens	Poder	PPV	Sens	Poder
circular	0.4926	0.3829	0.6630	0.6492	0.4033	0.7610
$\alpha = 1$	0.5648	0.6125	0.6320	0.6698	0.6676	0.7680
$\alpha = 0.5$	0.4708	0.7622	0.5820	0.6370	0.7093	0.7270
$\alpha = 0.25$	0.4974	0.7713	0.7350	0.6265	0.6773	0.8340
$\alpha = 0$	0.5204	0.8271	0.6260	0.6231	0.6741	0.7520

Para o cenário 2, o algoritmo com o maior poder de detecção foi o Scan Circular, porém a qualidade dos clusters detectados foi muito baixa tendo sensibilidade e PPV abaixo de 0,7, mesmo esse cenário sendo apenas um pouco menos circular que o cenário 1. Em todos os casos o algoritmo proposto teve uma qualidade dos clusters detectados mais eficiente. Aparentemente, o melhor desempenho considerando um compromisso entre poder, sensibilidade e PPV foi para $\alpha = 1$, e o pior para $\alpha = 0$.

No cenário 3, onde o cluster é formado apenas por uma fileira de regiões, percebemos que em todos os casos o PPV é abaixo de 0.57. Isso significa que são detectadas muitas regiões que não pertencem ao cluster verdadeiro, o que já era esperado, pois como o cluster é apenas uma fileira de regiões é aceitável que o algoritmo busque regiões vizinhas que não pertençam ao cluster verdadeiro. Em todos o casos a sensibilidade tem um desempenho melhor que o PPV exceto no Scan circular que tem uma sensibilidade muito baixa. De forma geral, nesse cenário, o algoritmo proposto apresentou um melhor desempenho comparado ao Scan Circular.

No quarto e último cenário, onde o cluster é formado por duas fileiras de regiões, houve uma melhora no desempenho dos algoritimos em relação ao cenário 3. O Scan Circular teve uma melhora no PPV, porém a sensibilidade continuou muito baixa, o que significa que o Scan

circular está detectando poucas regiões que pertencem ao cluster verdadeiro.

As figuras 3.2, 3.3, 3.4, 3.5 e 3.6, foram criadas a partir da frequência com que cada região apareceu na zona detectada, quando essa zona foi considerado um cluster, ou seja, foram consideradas apenas as frequências nos casos que foi rejeitada a hipótese nula. No mapa é representada a frequência de duas formas: uma é com o valor dela dentro da região, quando a frequência é maior que 0,05, e outra com a cor, quanto mais escura é a cor da região maior foi a frequência com que ela apareceu. Observando cada uma das figuras separadamente é possível ver em qual cenário cada algoritmo tem um melhor desempenho.

Na Figura 3.2 temos as frequências para o método Scan Circular. Percebemos que o melhor desempenho foi obtido para o cenário 1, o que era de se esperar, pois ele tem um formato circular. Vemos que as regiões que estão no centro do cluster foram detectadas por volta de 95% das vezes e as regiões da borda do cluster por volta de 80%. Para o cenário 2, percebemos que o Scan circular tem uma dificuldade em estimar o cluster verdadeiro devido ao seu formato irregular. Embora o cluster verdadeiro não tenha um formato circular, em média as regiões dos clusters estimados formam uma mancha com formato aproximadamente circular. Mesmo assim o método conseguiu detectar grande parte das regiões do cluster por volta de 80% das vezes, mas teve muita dificuldade em detectar a "cauda" de regiões na porção inferior esquerda do cluster. No terceiro cenário percebemos a dificuldade que o Scan circular tem para detectar corretamente o cluster. As regiões mais internas do cluster foram detectadas por volta de apenas 40% das vezes, o que corrobora o que foi visto na Tabela 3.1. No cenário 4 ele apresentou a mesma dificuldade com uma leve melhora em relação ao cenário anterior.

Nas figuras 3.3, 3.4, 3.5 e 3.6 temos as frequências para o algorítimo proposto com diferentes valores de α. Para α = 0 percebemos que o melhor desempenho foi para o cenário 3. Regiões que estão no centro do cluster foram detectadas por volta de 94% e esse valor vai diminuindo conforme se aproxima das bordas do cluster. Para os outros 3 cenários o desempenho foi similar detectando as regiões por volta de 70% das vezes. Para α = 0,25 percebemos que também se detectou melhor as regiões do centro do cluster no cenário 3, por volta de 86%, porém houve

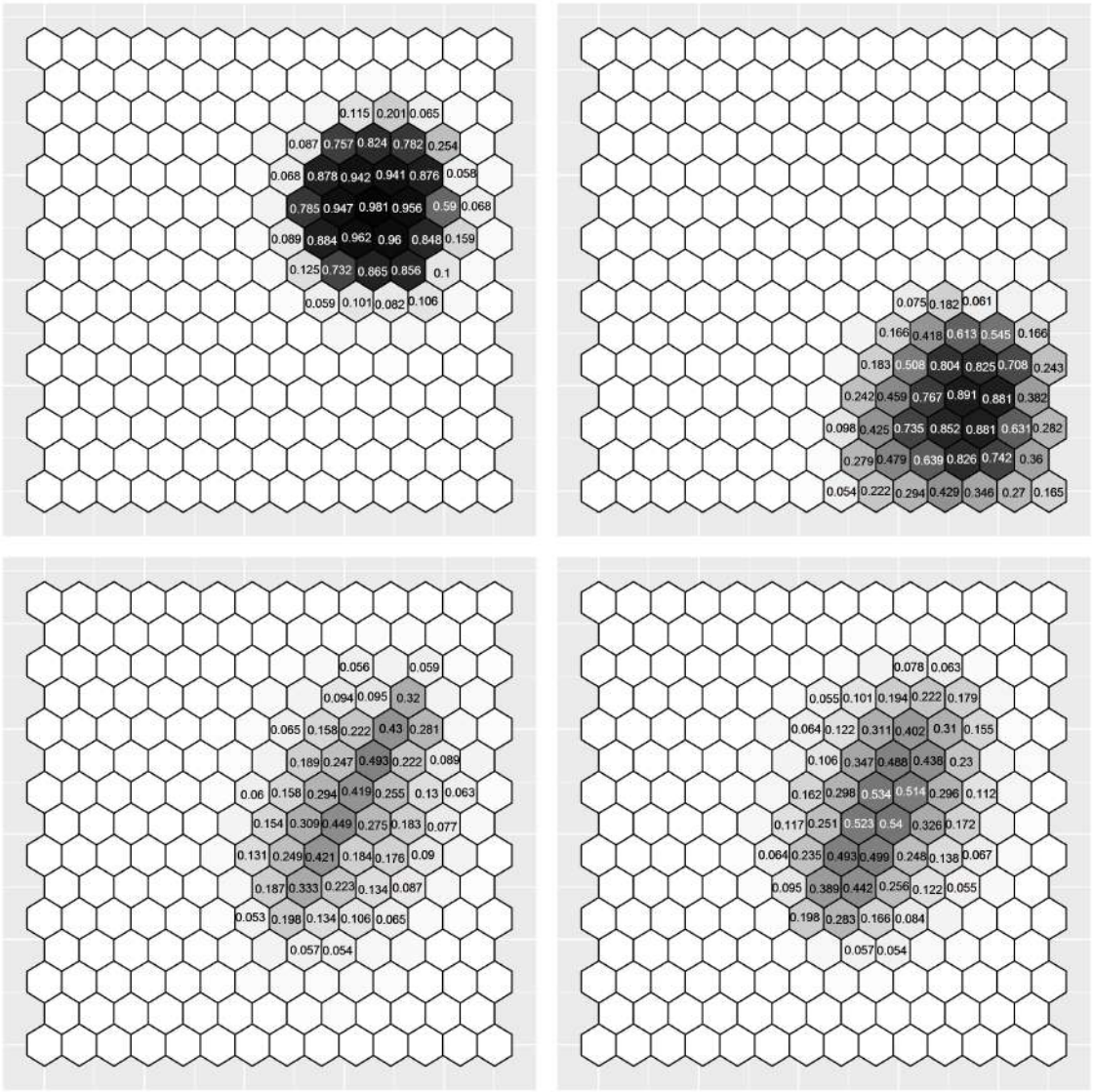

Figura 3.2: Frequência Scan Circular.

uma alta porcentagem de detecção de regiões que não pertencem ao cluster. Nos outros 3 cenários há um desempenho similar, assim como para α = 0, mas com a detecção das regiões um pouco superior, por volta de 74% das vezes. Para α = 0,5 os melhores desempenhos foram no cenário 1 e 2, que têm formatos mais compactos em comparação com os cenários 3 e 4. Para α = 1, os desempenhos nos cenários 1 e 2 foram muito superiores em relação aos cenários 2 e 3.

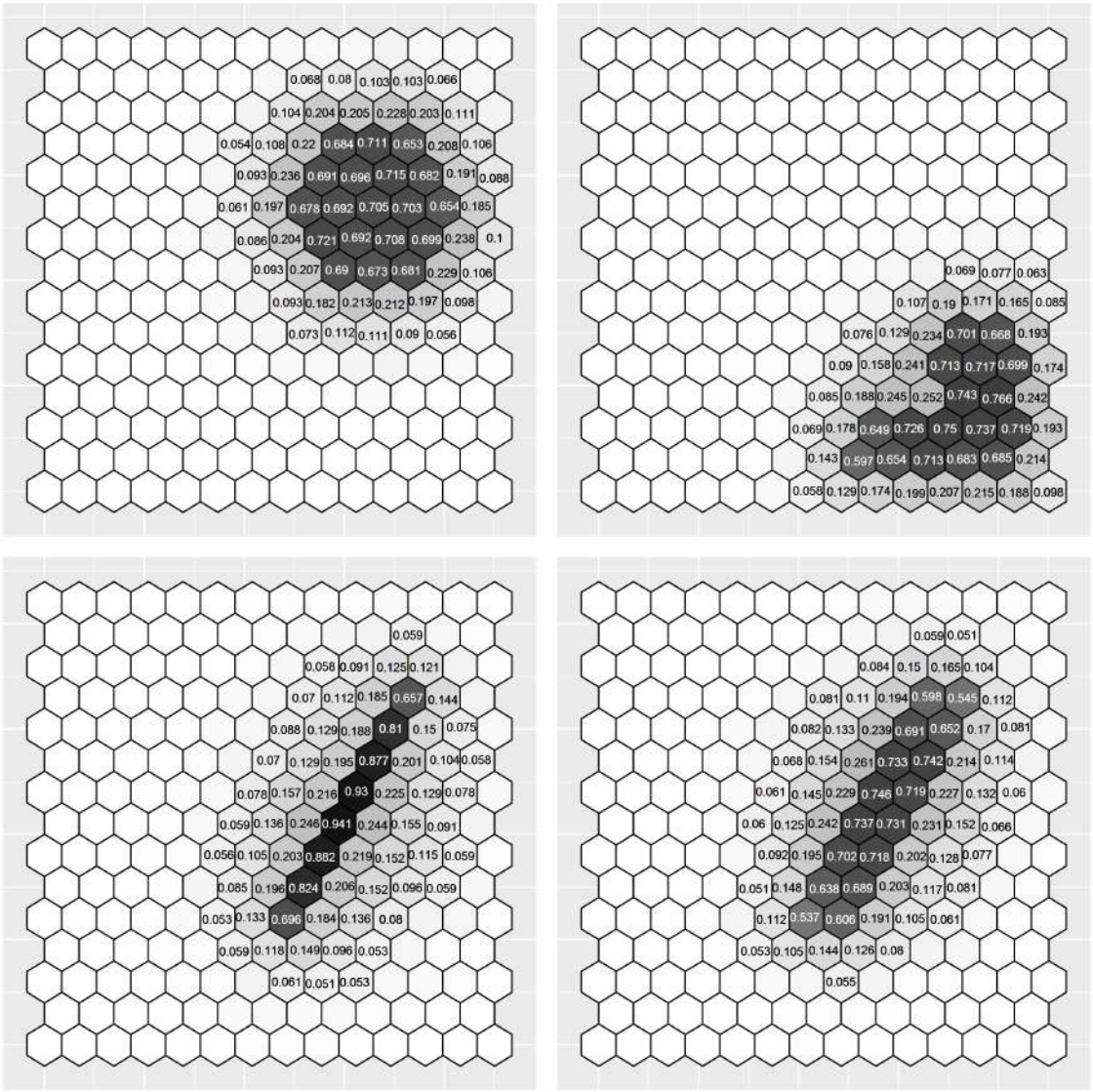

Figura 3.3: Frequência com penalização α=0.

De forma geral podemos perceber que quanto maior o valor da penalização α mais preciso será o cluster estimado quando o cluster verdadeiro tem um formato mais compacto e, analogamente, quanto mais próximo de 0 mais factível será o cluster estimado quando o cluster verdadeiro tem um formato irregular.

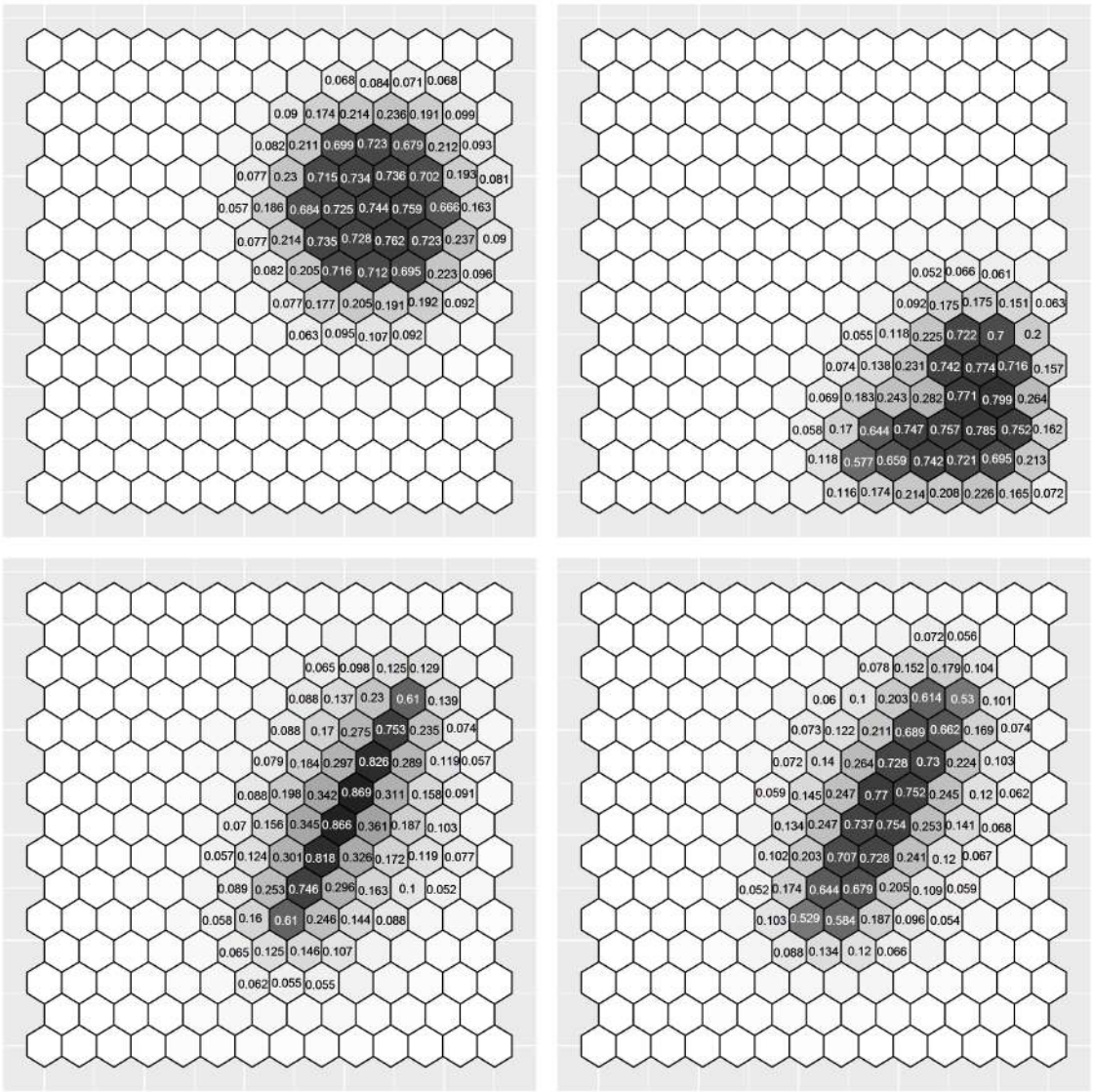

Figura 3.4: Frequência com penalização α=0,25.

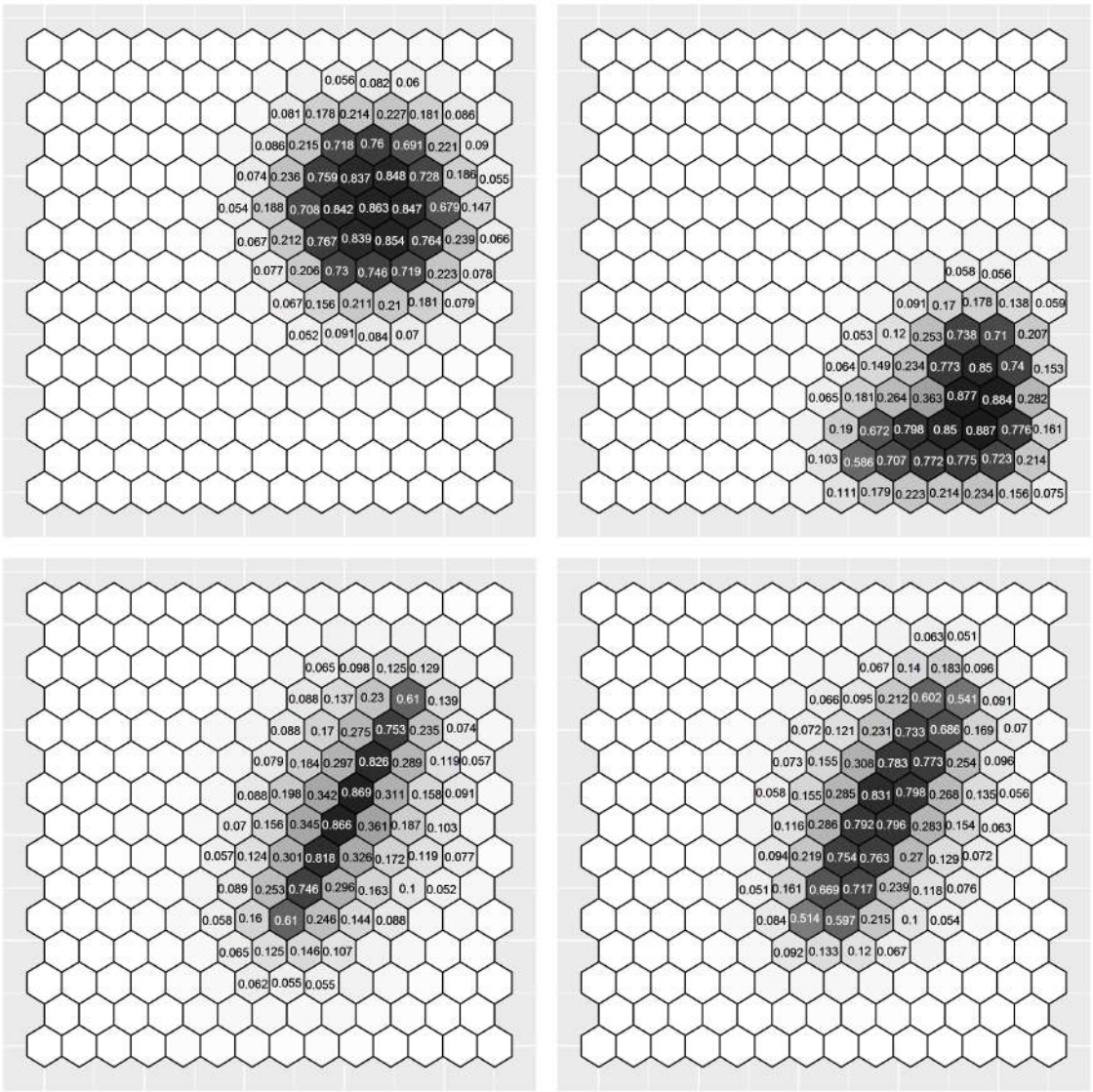

Figura 3.5: Frequência com penalização α=0,5.

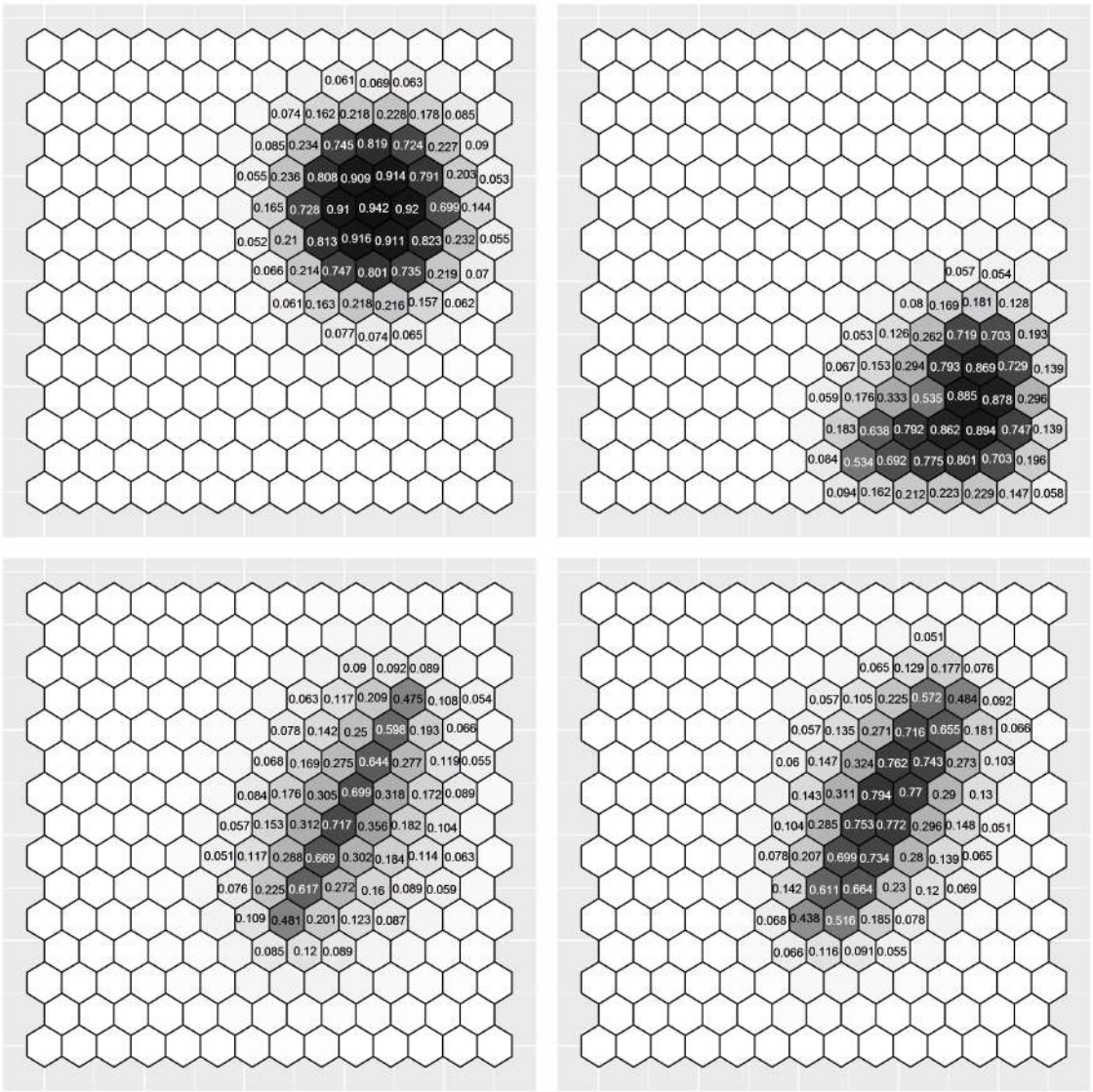

Figura 3.6: Frequência com penalização α=1.

Capítulo 4

Aplicação em dados reais

O algoritmo proposto foi aplicado para estudo de cluster em óbitos ocorridos por doenças pulmonares obstrutivas crônicas no estado do Mato Grosso no ano de 2015. A população em risco é de 615.426 formada por pessoas de ambos os sexos com idade acima de 50 anos. Nesse ano foram observados um total de 580 casos de óbitos por doenças pulmonares obstrutivas crônicas no estado do Mato Grosso. O mapa do Mato Grosso é dividido em 141 municípios e os dados de óbitos foram obtitos através das Estatísticas Vitais de mortalidade pela CID-10 (Classificação internacional de doenças) do DATASUS.

Para a análise foram aplicados 6 algoritmos: Scan Circular e o PSO para diferentes valores de α (0, 0.25, 0.5, 1, 1,5). O Scan Circular foi aplicado utilizando a proporção máxima da população de 50% e para o algoritmo PSO foram utilizados os seguintes parâmetros.

1. Proporção máxima da população dentro de uma zona: $0,50$
2. Proporção de voo (cruzamento): $0,50$
3. Probabilidade de mutação: $0,20$
4. Número de iterações: 100
5. α: 0, 0.25, 0.5, 1 e 1,5.

Tabela 4.1: Resumo dos clusters detectados.

Penalização	# reg	casos	casos esp	pop	mortali	$\log(\lambda)$	$k^{\alpha}\log(\lambda)$	p_valor
circular	25	106	67.70	71830	1.476	10.697		0.003
α = 1,5	23	115	70.19	74482	1.544	13.995	9.600	0.049
α = 1	20	103	56.25	59689	1.726	17.706	11.476	0.060
α = 0,5	19	103	55.02	58383	1.764	18.865	14.469	0.056
α = 0,25	20	108	58.49	62065	1.740	19.150	16.533	0.056
α = 0	18	106	56.58	60031	1.766	19.540	19.540	0.043

Na Tabela 4.1 são apresentados os resultados gerados após a aplicação de cada algoritmo. Cada linha da Tabela 4.1 se refere ao algoritmo utilizado e nas colunas temos o número de regiões, o total de óbitos, o número esperado de óbitos, o tamanho da população em risco, a taxa de mortalidade por mil habitantes, o valor de $\log(\lambda)$, o valor de $\log(\lambda)$ penalizado e o $p - valor$ do cluster mais verossímil em cada caso.

Analisando os dados que são apresentados na Tabela 4.1 percebemos que todas as zonas encontradas foram consideradas um cluster ao nível de significância de 10%. O cluster formado pelo Scan Circular apresentou um total de 25 regiões e 23 regiões para o PSO com α = 1,5. Para os outros PSO foram encontrados clusters com um menor número de regiões: 20 para α = 0,25 e 1, 19 regiões para α = 0,5 e 18 regiões quando não temos penalização. O total de óbitos foi bem similar para todos os métodos, por volta de 105 óbitos, com exceção quando α = 1,5. Os tamanhos da populações em risco para o PSO foram bem próximos considerando os valores de α0, 0.25, 0.5 e 1, em torno de 60 mil. Já para o Scan Circular e para o PSO quando α = 1,5 a população em risco foi de acima de 71 mil, isso faz com que a taxa de mortalidade seja superior nos clusters encontrados pelo algoritmo PSO que ficaram com uma mortalidade por volta de 1,76 a cada mil habitantes. No Scan Circular a mortalidade foi de 1,476 e para o PSO quando α = 1,5 foi de 1,544.

O número esperado de óbitos em todos os algoritmos é bem inferior ao número de óbitos observado, o que corrobora a refutação da hipótese nula fazendo com que indívios dentro do cluster estejam mais propensos a terem um óbito por doenças pulmonares obstrutivas crônicas

do que fora dele. Percebemos também que a diferença entre os óbitos observados e esperados são maiores quando é usado o algorítmo PSO.

Observando os mapas da Figura 4.1 percebemos que os clusters encontrados pelo algoritmo PSO tiveram um formato arbitrário, bem diferente do formato circular encontrado pelo Scan Circular, com exceção quando α = 1,5, nesse caso, o cluster obtido tem um formato mais circular, se aproximando mais do Scan Circular . Também é possível perceber que um pequeno aumento da intensidade da penalização α faz com que o cluster seja mais compacto e tenha menos "buracos", fazendo com que as regiões dentro dele tenham mais vizinhos, e quando esse valor é mais alto o cluster vai tomando a forma circular.

Como o valor de α afeta diretamente o valor de λ, o algoritmo que apresentou o maior $\log(\lambda)$ foi quando não houve penalização, apresentando o valor de 19,54. Esse valor vai diminuindo com o aumento da intensidade da penalização, e quando α=1,5 o valor de $\log(\lambda)$ é 13,99 que ainda é superior ao valor do Scan circular que foi de 10,69.

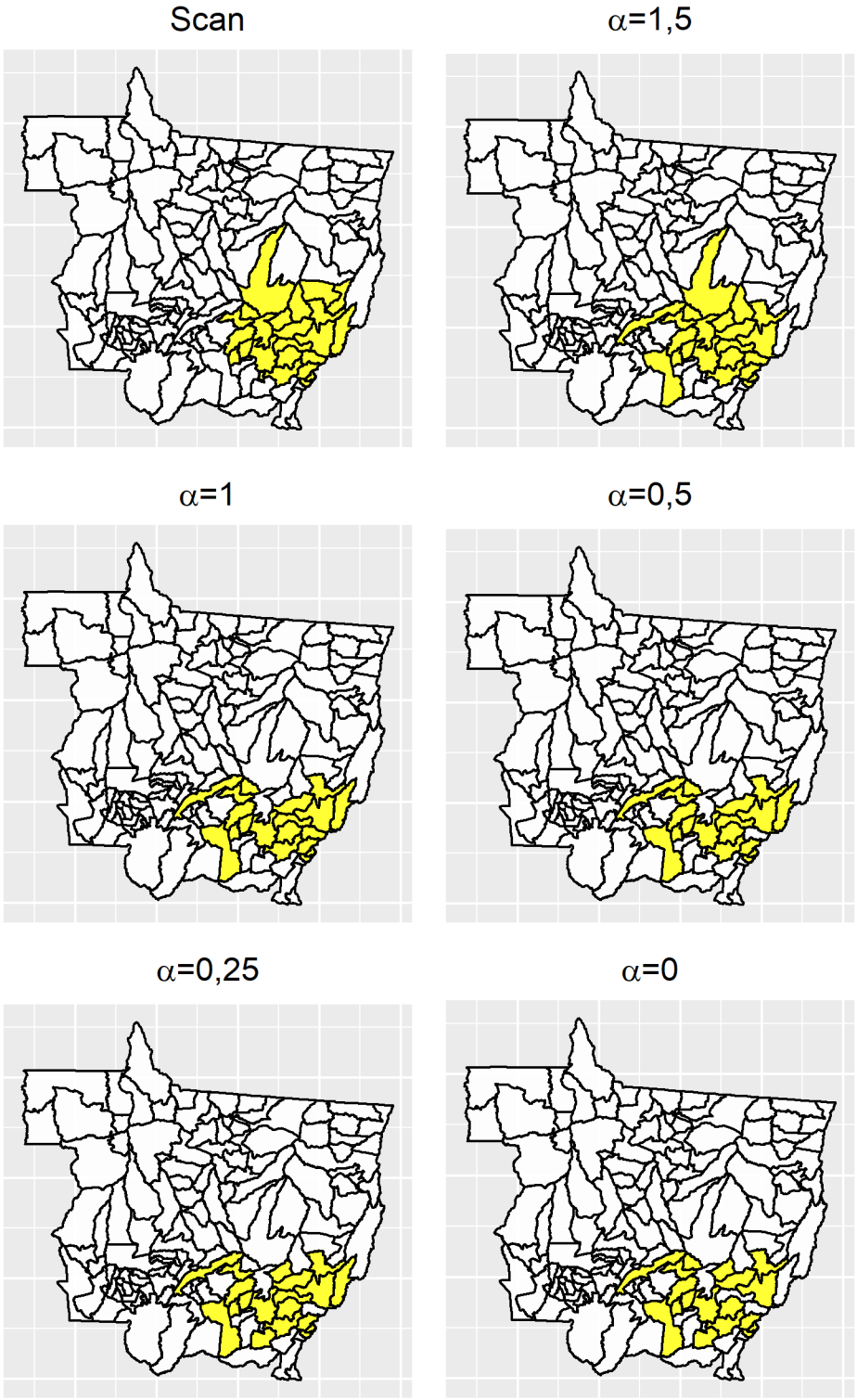

Figura 4.1: Clusters detectados.

Capítulo 5

Considerações finais

Nesse trabalho foi proposto um algoritmo PSO especializado para o problema de detecção de clusters espaciais. O algoritmo é baseado no PSO e também utiliza conceitos de algoritmos genéticos para solução do problema. A motivação se deve ao fato de que o método Scan Circular é altamente utilizado na literatura para problemas de detecção de cluster espaciais, porém caso o cluster verdadeiro não tenha formato pelo menos aproximadamente circular, a solução apontada pode subestimar ou superestimar o cluster verdadeiro.

Nas seções precedentes foram apresentadas simulações onde em alguns cenários em que o cluster verdadeiro não tinha um formato circular o algoritmo PSO apresentou resultados superiores comparado ao Scan Circular, e quando o cluster verdadeiro tinha formato circular apresentava resultados satisfatórios, quando utilizada a penalização espacial adequada.

5.1 Trabalhos Futuros

Em todas a simulações, com exceção de α, todos os parâmetros foram fixos. Apesar dos bons resultados obtidos, um aprofundamento no estudo desses parâmetros pode trazer melhoria da detecção dos cluster.

Umas das principais dificuldades no desenvolvimento desse trabalho foram o tempo com-

putacional gasto para fazer as simulações sob H_0 e H_A, e a implementação computacional que é feita utilizando a matriz de vizinhança.

Pode-se propor um critério de parada para o número de iterações do algoritmo, o que diminuiria bastante o tempo computacional. Seria interessante também fazer um estudo sobre a convergência do algoritmo em relação ao parâmetro de proporção de voo, o que também poderia diminuir o tempo computacional.

Um dos motivos pelo qual o Scan Circular é amplamente utilizado na literatura é por sua fácil forma de implementação e utilização. Como os métodos apresentados nesse trabalho estão implementados em linguagem R, um próximo passo seria melhorar sua performance computacional e criar um pacote para a utilização por outros pesquisadores.

Bibliografia

Besag, Julian e Newell, James (1991). “The detection of clusters in rare diseases”. *Journal of the Royal Statistical Society: Series A (Statistics in Society)* 154.1, pp. 143–155.

Choynowski, Mieczyslaw (1959). “Maps based on probabilities”. *Journal of the American Statistical Association* 54.286, pp. 385–388.

Collado Chaves, Andrea (2003). “Fecundidad adolescente en el gran área metropolitana de Costa Rica”.

Cuadros, Diego F et al. (2017). “Vector transmission alone fails to explain the potato yellow vein virus epidemic among potato crops in Colombia”. *Frontiers in plant science* 8, p. 1654.

Duczmal, Luiz, Kulldorff, Martin e Huang, Lan (2006). “Evaluation of spatial scan statistics for irregularly shaped clusters”. *Journal of Computational and Graphical Statistics* 15.2, pp. 428–442.

Duczmal, Luiz et al. (2007). “A genetic algorithm for irregularly shaped spatial scan statistics”. *Computational Statistics & Data Analysis* 52.1, pp. 43–52.

Izakian, Hesam e Pedrycz, Witold (2012). “A new PSO-optimized geometry of spatial and spatio-temporal scan statistics for disease outbreak detection”. *Swarm and Evolutionary Computation* 4, pp. 1–11.

Kennedy, J e Eberhart, R (1995). “Particle swarm optimization (PSO)”. *Proc. IEEE International Conference on Neural Networks, Perth, Australia*, pp. 1942–1948.

Kirkpatrick, Scott, Gelatt, C Daniel e Vecchi, Mario P (1983). “Optimization by simulated annealing”. *science* 220.4598, pp. 671–680.

Kulldorff, Martin (1997). “A spatial scan statistic”. *Communications in Statistics-Theory and methods* 26.6, pp. 1481–1496.

Kulldorff, Martin, Tango, Toshiro e Park, Peter J (2003). “Power comparisons for disease clustering tests”. *Computational Statistics & Data Analysis* 42.4, pp. 665–684.

Minamisava, Ruth et al. (2009). “Spatial clusters of violent deaths in a newly urbanized region of Brazil: highlighting the social disparities”. *International journal of health geographics* 8.1, p. 66.

Oliveira, Dênis Ricardo Xavier de et al. (2019). “Spatial cluster analysis using particle swarm optimization and dispersion function”. *Communications in Statistics-Simulation and Computation*, pp. 1–18.

Openshaw, Stan et al. (1988). “Investigation of leukaemia clusters by use of a geographical analysis machine”. *The Lancet* 331.8580, pp. 272–273.

Turnbull, Bruce W et al. (1989). *Monitoring for clusters of disease; Application to leukemia incidence in upstate New York*. Rel. técn. Cornell University Operations Research e Industrial Engineering.

Whittemore, Alice S et al. (1987). “A test to detect clusters of disease”. *Biometrika* 74.3, pp. 631–635.

Yiannakoulias, Nikolaos, Rosychuk, Rhonda J e Hodgson, John (2007). “Adaptations for finding irregularly shaped disease clusters”. *International Journal of Health Geographics* 6.1, p. 28.