Augusto Cesar Ribeiro Nunes

Adaptação do algoritmo *Particle Swarm Optimization* (PSO) para identificação de *clusters* espaciais
Relatório Parcial

Brasília, Brasil

Novembro de 2016

Augusto Cesar Ribeiro Nunes

Adaptação do algoritmo *Particle Swarm Optimization* (PSO) para identificação de *clusters* espaciais
Relatório Parcial

Relatório Parcial do Trabalho de Conclusão de Curso a ser apresentado para obtenção do título de Bacharel em Estatística.

Orientador: Prof. Dr. **André Luiz Fernandes Cançado**

Universidade de Brasília

Departamento de Estatística

Graduação

Brasília, Brasil

Novembro de 2016

Introdução

Este relatório parcial retrata o estado atual do trabalho de conclusão de curso em questão. Segundo o **??** do trabalho, aprovado em proposta de projeto, ele compreende o que foi feito dentre os meses de agosto e outubro de 2016.

A Análise e Identificação de *clusters* é um problema da Estatística Espacial onde a heurística utilizada está intimamente ligada à natureza do fenômeno - sua dimensionalidade, a presença de uma estrutura de correlações relevante etc.

Temos em mãos um problema onde há a utilização de dados pontuais: registros das coordenadas (X_i, Y_i) do evento i; dados de área: onde não dispomos da localização precisa do evento mas sim de uma referência da área onde o mesmo ocorreu; e de dados de superfície: o elemento do conjunto de dados é formado por uma tripla (X_i, Y_i, Z_i) correspondente à coordenada geográfica do evento e a uma medição Z_i realizada. Usualmente, ao analisar dados pontuais e dados de área, nosso interesse é por questões relativas à distribuição geográfica do processo pontual, quando o interesse se volta para dados de superfície, o interesse é por processos inferenciais que descrevam um modelo de probabilidade para o processo em questão.

O problema de identificação de *clusters* a que este trabalho se dedica consiste em, ao analisar um espaço geográfico bidimensional G, particionado em regiões e no qual há a ocorrência de um processo pontual não-estocástico e dicotômico, avaliar se o mesmo ocorre de maneira meramente aleatória ou se há algum padrão definidor de um conglomerado, ou seja, tratamos de um processo pontual utilizando dados de área ao invés de dados pontuais. O motivador é o Algoritmo do *Scan* Circular de Kulldorff, brevemente descrito em **??**. Apresentada por (**??**), esta heurística simples e versátil tornou-se ubíqua graças a sua eficiência e interpretabilidade. O objetivo final do trabalho será obter um novo procedimento para a sua janela de análise (*scan window*), que não será mais circular mas sim irá considerar um critério de adjacência, baseada no algoritmo *Particle Swarm Optimization* (PSO).

O Algoritmo PSO por sua vez é muito brevemente descrito em **??**. Proposto por (**??**), provou-se ser uma heurística de otimização robusta e cujas aplicações perpassam várias áreas, como mostram (**??**).

O Capítulo **??** e suas respectivas seções contém descrições dos passos já concluídos e resultados obtidos atee então. Dada a natureza essencialmente computacional do trabalho, eles decorrem diretamente das implementações e do código fonte apresentado no Apêndice **??**, que são citados ao longo do texto à medida que se faz necessária a referência.

No Capítulo **??** é apresentada uma previsão tentada dos estágios imediatamente posteriores, sob uma ótica de planejamento.

Os conjuntos de dados utilizados no trabalho encontram-se no Anexo **??**.

1 O *Scan* Circular de Kulldorff

1.1 Descrição

(**??**) tinha como um de seus objetivos estender a metodologia de detecção de conglomerados em um processo pontual unidimensional sugerida por (**??**) a um contexto bidimension‘al. Abordagens anteriores com o mesmo intuito usando processos de Poisson não-homogêneos (**??**) e extensões do modelo Binomial (**??**) resultaram em modelos de flexibilidade e eficiência questionáveis. A proposta de Kulldorff é significativamente mais simples e versátil, e a única limitação da análise é que há um condicionamento perene ao número total de casos observados. Suas prescrições são poucas e simples:

- Conhece-se a população sob risco do evento no espaço geográfico G e a mesma mantém-se constante durante a análise[1];
- Conhece-se o número de casos do evento no espaço geográfico G.
- Bem como as coordenadas geográficas de ocorrência de todos estes eventos.

Seja G um Espaço Geográfico bidimensional onde há a presença de um processo pontual espacial N, onde N(A) é o número de pontos no conjunto $A \in G$. Conforme a janela de análise se movimenta sobre o Espaço Geográfico G, ela define uma coleção (uma zona) de regiões $\mathcal{Z} \in G$. Considerando um modelo Bernoulli para o processo pontual, temos que indivíduos que apresentam certa característica de interesse são pontos, e a localização destes indivíduos representa o processo pontual. No modelo, há exatamente uma zona $Z \in G$ tal que cada indivíduo dentro dela tem probabilidade p de ser um ponto (i.e. de possuir a característica de interesse), enquanto a probabilidade de indivíduos fora da zona em questão é q. A Figura **??** abaixo ilustra o modelo neste estágio.

A escolha do formato dos subconjuntos $A \in G$ varia de acordo com o contexto: bairros, ruas, distritos, quadrantes etc. Para facilitar a ilustração aqui utilizada, consideramos divisões em quadrados regulares, como na Figura **??**.

O passo a seguir pode ser considerado uma abstração útil: podemos definir um centroide[2] onde estão *condensados* todos os dados da região. A definição deste centroide pode partir de critérios práticos ou meramente geométricos: o caso inicial na região, uma unidade de saúde de referência, a prefeitura de cada cidade, uma instituição financeira etc. Essa medida mitiga o problema combinatório para a avaliação de conglomerados que

[1] Kulldorff também considerou modelos estocásticos em extensões de seu trabalho original

[2] Aqui a definição de *centroide* é *ad hoc*: não corresponde necessariamente ao centro geométrico da região

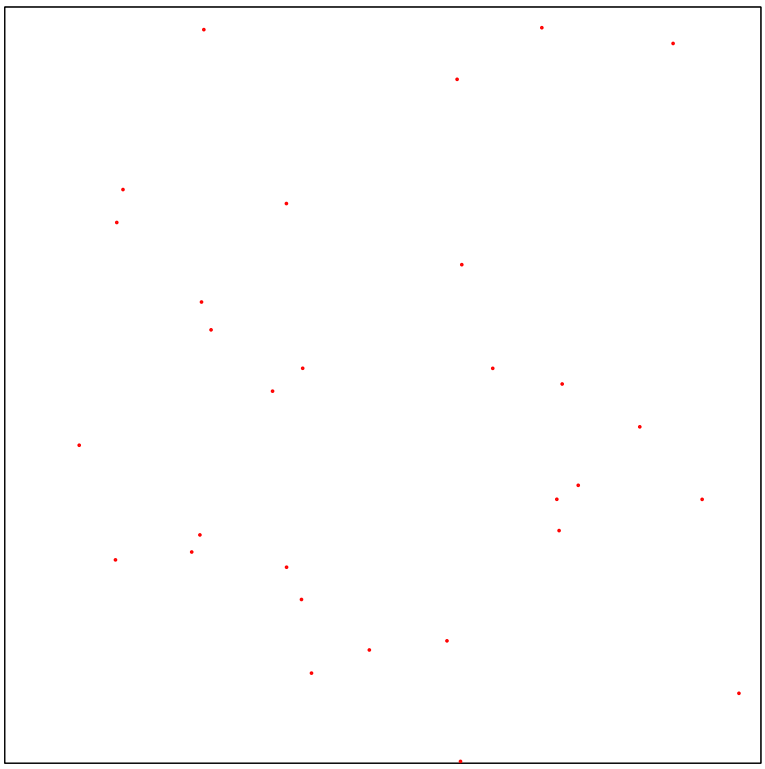

Figura 1 – Modelo inicial do *Scan* Circular de Kulldorff em um espaço geográfico com 30 casos (pontos vermelhos)

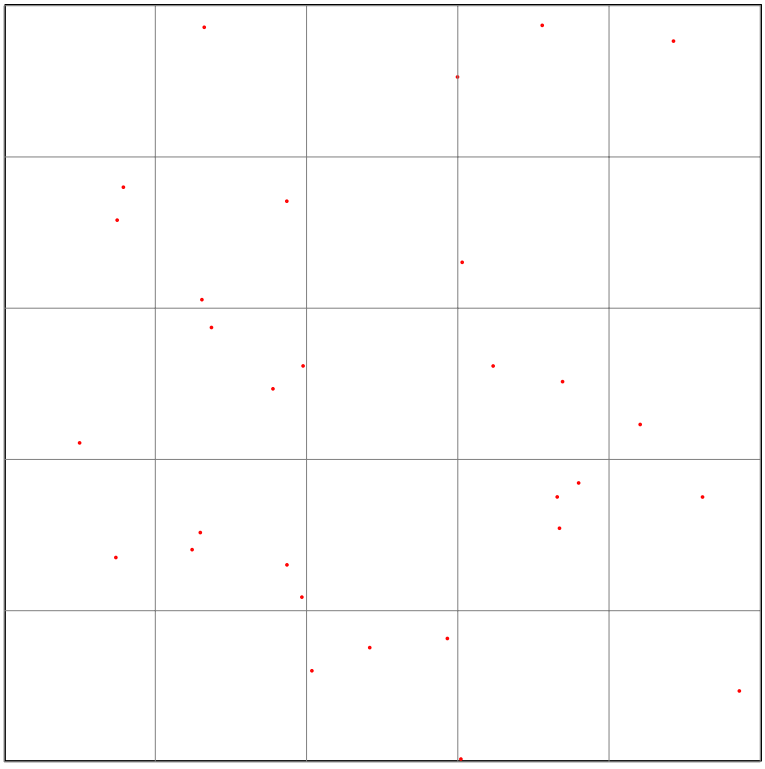

Figura 2 – Modelo do *Scan* Circular de Kulldorff em um espaço geográfico com 30 casos e dividido em 25 regiões regulares. Cada uma dessas regiões é numerada.

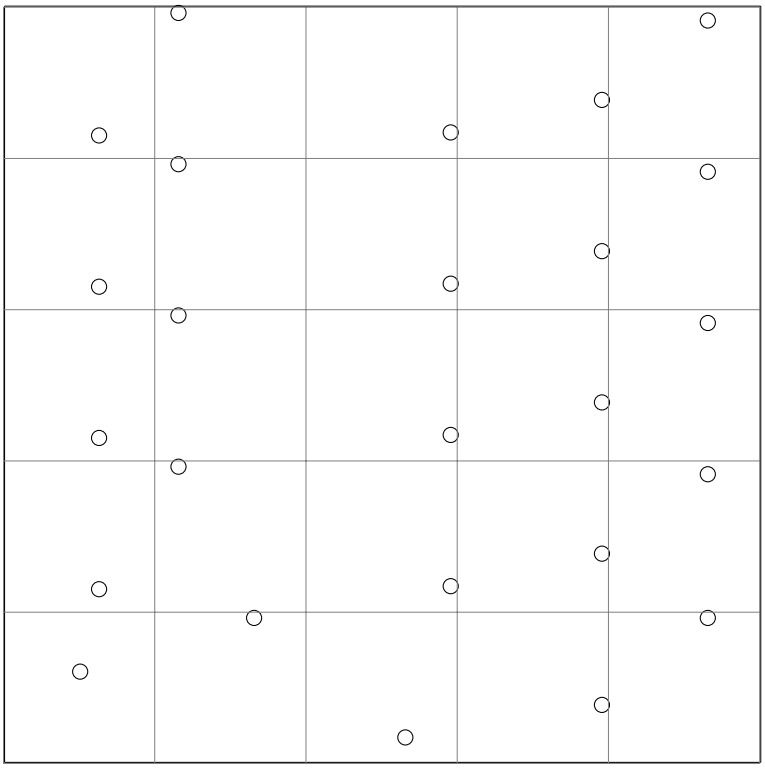

Figura 3 – Modelo do *Scan* Circular de Kulldorff obtido considerando os *centroides* (circulos pretos) das regiões definidas anteriormente. Cada um dos 25 *centroides* concentra todos os casos e a população da respectiva região

surge quando n se torna suficientemente grande, mas ainda assim há prejuízo à eficiência do método quando a escolha das regiões e de seus centroides não é realizada de maneira cuidadosa. Em particular, o conjunto de dados utilizado no trabalho é artificial, e os centroides foram definidos com a introdução de um ruído aleatório às suas coordenadas x e y. Uma ilustração deste procedimento é feita na Figura **??**

O procedimento de identificação dos *clusters* utiliza o Modelo Bernoulli e a razão de verossimilhança é obtida a partir das seguintes hipóteses:

$$H_0 : p = q, \qquad \text{para todo } \mathcal{Z} \in G \tag{1.1}$$

$$H_1 : p > q, \qquad \text{para algum } \mathcal{Z} \in G. \tag{1.2}$$

$$L(Z, p, q) = p^{n_Z}(1-p)^{\mu(Z)-n_Z} q^{n_G-n_Z}(1-q)^{(\mu(G)-\mu(Z))-(n_G-n_Z)} \tag{1.3}$$

Onde n_Z e n_G são o número de caso observados na zona Z e em todo o espaço geográfico G, respectivamente; e $\mu(Z)$ e $\mu(G)$ são os casos esperados na zona Z e no espaço geográfico G, respectivamente.

A próxima etapa, crucial, é a janela de análise do algoritmo propriamente dita:

1. Inicializar uma região[3] qualquer;
2. Obter **??** para a região caso $n_Z > n_G$;
3. **Compor uma zona com a região inicial e a região cujo centroide seja o mais próximo do centroide inicial;**
4. Obter **??** para a região caso $n_Z > n_G$;
5. Repetir a composição em 3. e o cálculo de **??**;
6. Parar quando o tamanho populacional da zona for maior que a metade do tamanho populacional total.

O Passo 3. é problemático: o critério de distância considera uma definição arbitrária - o centroide da região - e a janela de análise claramente cresce de maneira, como um **círculo** segundo o próprio nome do algoritmo sugere, o que afeta gravemente o poder do algoritmo na detecção de *clusters* de formatos irregulares. Em (**??**) o algoritmo da janela de análise foi estendido para considerar também formatos elípticos para o conglomerado. Ainda assim, a falta de flexibilidade do algoritmo, seja ele em formato circular ou elíptico, motiva fortemente a heurística considerada neste trabalho, descrita no capítulo a seguir.

Finalmente, a decisão para o teste das hipóteses 1.1 e 1.2 baseia-se em Simulação de Monte Carlo, procedimento descrito em **??**.

[3] Poderia-se falar em *centroide* e *região* como sinônimos, a escolha pelo segundo termo é em nome da clareza.

2 O Algoritmo *Particle Swarm Optimization* (PSO)

O Algoritmo PSO tem como objetivo a otimização iterativa de funções não-lineares, e traz inspiração de comportamentos sociais biológicos como a revoada de aves e cardumes de peixes.

Técnicas de *Swarming*, termo que pode ser traduzido como *enxameamento* e que foi sugerido inicialmente por (**??**), são amplamente utilizadas em contextos de Inteligência Artificial: a partir de um número considerável de agentes que interagem de maneira local e global, estes procedimentos buscam descrições de Estados Ótimos para os modelos. Estes *Estados Ótimos* geralmente são os máximos globais de funções objetivo não-lineares.

O uso neste trabalho de um algoritmo *baseado* no PSO se resumirá a encontrar uma maneira, ainda indefinida, para que a partir de uma série de zonas candidatas avalie-se interseções entre estas zonas e obtenha-se, assim, uma nova zona composta cuja forma não seja necessariamente regular, ao contrário do imposto pelo *Scan* Circular de Kulldorff original. As prescrições e suposições do Modelo Bernoulli descritas no Capítulo **??** serão as mesmas: as únicas diferença serão a forma da janela de análise e a definição de um critério de decisão para identificar o *cluster*.

3 Resultados Parciais

3.1 *Scan* Circular de Kulldorff

Este algoritmo encontra-se implementado no *software* SaTScan (**??**), mantido pelo próprio Martin Kulldorff. Apesar de gratuito (i.e. distribuído sem ônus para o usuário), sua licença não é *livre* na maioria das acepções utilizadas (**??**), mais gravemente a não disponibilização de seu código-fonte. Isso impede, por exemplo, que utilizemos como base a implementação do SaTScan, feita em Java, para o R, o *software* escolhido para implementação neste trabalho. Uma pesquisa no serviço *crantastic* (**??**), que disponibiliza uma ferramenta de busca nos pacotes disponíveis livremente para utilização no R, não retornou resultados para palavras-chave como "kulldorff circular scan", "circular scan"e similares. Uma pesquisa similar no *github* também não retornou dentre os resultados implementações utilizáveis do *Scan* Circular de Kulldorff.

Não obstante estas limitações, uma implementação *inédita* do algoritmo serviu para facilitar grandemente a compreensão do mesmo, e de sua limitação que levou à proposição da nova heurística que serve como Objetivo deste trabalho. Ainda que o método sugerido por Kulldorff seja facilmente explicável em algumas palavras, colocá-lo em prática a partir do zero possibilitou uma familiarização importante.

O código-fonte em **??** é, apesar de correto, problemático em termos de eficiência do código. Uma análise de perfil (*profiling*) do código utilizando a ferramenta `Rprofvis` do R, disponível em (**??**), mostra o culpado: o incremento iterativo do *data frame* `resultado`. Este tipo de erro é comum, sendo inclusive citado no texto "The R Inferno"(**??**), do consultor Patrick Burns. Na verdade, sua solução é, em teoria, simples: via de regra a alocação de uma estrutura de dados qualquer previamente considerando seu tamanho final ou máximo sempre será mais eficiente que sua alocação iterativa. Como o objetivo maior do trabalho não envolve uma implementação eficiente do *Scan* Circular, não há inicialmente a preocupação em mitigar o efeito da técnica flagrantemente ineficaz de alocação de dados.

Ainda sobre o código-fonte, sua leitura deve ser razoavelmente intuitiva mas ele essencialmente especifica uma função que realiza duas coisas, a saber:

- Obtém a matriz de distâncias dos centroides das regiões utilizando distância euclidiana.
- Calcula o logaritmo da razão de verossimilhança para cada uma das zonas candidatas.

Estas zonas são nada mais do que composições de regiões, e como o nome sugere

e deve ficar claro à leitura do código-fonte a heurística de composição destas zonas é a distância a partir de um centroide inicial até os centroides das outras regiões no espaço.

3.2 Simulação de Monte-Carlo

Na etapa seguinte, foi feita a verificação de significância do *cluster* mais verossímil, i.e. cujo logaritmo da razão de verossimilhança seja maior. A hipótese nula a ser testada é de que a probabilidade de ocorrência do evento *dentro* da zona candidata que apresenta o maior valor da estatística em questão é igual à probabilidade de ocorrência do evento fora desta zona em particular.

Para tanto, podemos gerar um número *considerável* de realizações de uma distribuição Multinomial sob hipótese nula, ou seja, cujas componentes do seu vetor de probabilidade são proporcionais ao produto entre a razão do total de casos sobre a população total e a população na região. Em notação usual, foram geradas realização de uma variável aleatória $X \sim (\sum_i P_i, \mu_0 P_1, \mu_0 P_2, \ldots, \mu_n P_n)$, onde μ_0 é a média geral sob a hipótese nula, e $P_i, i = 1, \ldots, n$ é a população na região i.

De acordo com o conjunto de dados aqui utilizado, disponibilizado pelo orientador, chegou-se à conclusão de que a zona formada pelas regiões 130, 131, 143, 144, 145, 157, 158 têm probabilidade de ocorrência do evento significativamente superior à observada nas outras zonas.

O baixo número de realizações da simulação dá-se em parte por conta da falta de eficiência do *Scan* Circular retratada na seção anterior. Ainda assim, o resultado observado no trabalho condiz com o resultado obtido previamente pelo orientador.

4 Próximos Passos

Há a tentativa de implementação de uma heurística para composição das zonas que não leve em consideração a distância entre os centroides, mas sim quais apresentam a propriedade de adjacência entre si. Este algoritmo estaria descrito, grosso modo, nos seguintes termos.

1. Inicializa-se uma região i qualquer.
2. A partir de i, e tendo acesso a quais regiões são adjacentes a i, obter a zona onde o logaritmo da razão de verossimilhança é maior quando há a composição da região i com suas adjacentes.
3. A partir da zona composta pelas duas regiões, i e aquela que mais aumenta a razão de verossimilhança, compor um vetor de vizinhanças para a zona a partir das regiões que a compõem.
4. Pare quando o tamanho populacional da zona for maior que metade da população total.

Eventualmente, o procedimento descrito acima encontrará um certo número de *zonas candidatas*. A partir destas zonas candidatas, a intenção é trabalhar com zonas candidatas que apresentem interseção entre si e compô-las de forma que possam apresentar formatos não-regulares.

5 Cronograma

As atividades a serem desenvolvidas são as seguintes:

1. Escolha do tema a ser abordado.
2. Desenvolvimento da proposta de projeto.
3. Entrega da proposta de projeto.
4. Revisão de literatura.
5. Elaboração da apresentação da proposta.
6. Apresentação oral da proposta.
7. Implementação
8. Verificação dos Modelos
9. Elaboração do relatório parcial.
10. Entrega do relatório parcial ao Professor Orientador.
11. Correção do relatório parcial.
12. Entrega do relatório parcial para a banca.
13. Desenvolvimento do modelo.
14. Elaboração do relatório final.
15. Entrega do relatório final ao Professor Orientador.
16. Correção do do relatório final.
17. Entrega do relatório final para a banca.

Tabela 1 – Cronograma

	2016					2017						
	Agosto	Setembro	Outubro	Novembro	Dezembro	Janeiro	Fevereiro	Março	Abril	Maio	Junho	Julho
??												
??												
??												
??												
??												
??												
??												
??												
??												
??												
??												
??												
??												
??												
??												
??												

Apêndices

APÊNDICE A – Códigos-fonte

Os códigos-fonte aqui disponibilizados encontram-se hospedados no repositório <https://github.com/august-o/tcc>.

A.1 *Scan* Circular de Kulldorff

```
arquivos <- list.files("dados_hex", full.names = TRUE)
dados <- lapply(arquivos, read.table, header = FALSE)
dados <- data.frame(dados)
names(dados) <- c("Casos", "X_c", "Y_c", "Pop")

#' Title llr.ksc
#'
#' @param dados: data-frame com quatro colunas: casos, X_c, Y_c e populacao
#'
#' @return Data Frame com relacao de zonas e respectivos logs. da razao de veros.
#' @export
#'
#' @examples
llr.ksc <- function(dados){
  pop_total <- sum(dados$Pop)
  casos_total <- sum(dados$Casos)

  # Matriz de distâncias

  mat_dist <- dist(dados[, 2:3])
  mat_dist <- as.matrix(mat_dist)

  diag(mat_dist) <- NA
  mat_dist_ind <- matrix(ncol = nrow(dados), nrow = nrow(dados) - 1)

  # Gerando a matriz de distancias ordenada por indices
  for (i in 1:nrow(dados)) {
    mat_dist_ind[, i] <- head(order(mat_dist[, i]), -1)
  }


  resultado <- data.frame()

  for(i in 1:ncol(mat_dist_ind)){
    zona <- i
    for(j in mat_dist_ind[,i]){
```

```
37        zona <- append(zona,j)
38        n_z <- sum(dados[zona, 4])
39        if(n_z > pop_total/2){
40          zona <- zona[1:length(zona)-1]
41          next()
42        }
43        c_z <- sum(dados[zona, 1])
44        mu_z <- casos_total * (n_z / pop_total)
45        ifelse(c_z > mu_z,
46               llr_z <-
47                 c_z * log(c_z / mu_z) +
48                 (casos_total - c_z) * log((casos_total - c_z) /
(casos_total - mu_z)),
49               llr_z <- 0)
50        resultado <- rbind(resultado, cbind(list(zona), llr_z))
51
52      }
53    }
54    return(resultado)
55  }
```

A.2 Simulação de Monte-Carlo

```
1  cluster.max <- data.frame()
2  llr.max.vet <- data.frame()
3
4  for(i in 1:100){
5    dados$Casos <- rmultinom(1, pop_total, (casos_total*dados$Pop)/pop_total)
6    resultado <- llr.func(dados)
7    llr.max.vet <- rbind(llr.max.vet, resultado[which.max(resultado$llr_z),])
8  }
```

ANEXO A – Conjuntos de Dados

Os conjuntos de dados utilizados neste trabalho encontram-se hospedados no repositório <https://github.com/august-o/tcc/tree/master/dados_hex>.

A.1 dados.csv

Arquivo tabulado que descreve o número de casos observados e a população de 203 regiões, bem como as coordenadas de seus *centroides*.

A.2 hex.adj

Matriz de adjacências para as 203 regiões. A componente (i,j) nesta matriz é igual a 1 se as regiões i e j são adjacentes entre si e 0 caso contrário.