UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM MATEMÁTICA APLICADA

Métodos Espectrais para Particionamento de Dados e Aplicações

por

Lucas Siviero Sibemberg

Dissertação submetida como requisito parcial
para a obtenção do grau de

Mestre em Matemática Aplicada

Prof. Dr. Luiz Emilio Allem
Orientador

Prof. Dr. Carlos Hoppen
Co-orientador

Porto Alegre, abril de 2022

CIP - CATALOGAÇÃO NA PUBLICAÇÃO

Siviero Sibemberg, Lucas

Métodos Espectrais para Particionamento de Dados e Aplicações / Lucas Siviero Sibemberg.—Porto Alegre: PPG-MAp da UFRGS, 2022.

138 p.: il.

Dissertação (mestrado)— Universidade Federal do Rio Grande do Sul, Programa de Pós-Graduação em Matemática Aplicada, Porto Alegre, 2022.
Orientador: Emilio Allem, Luiz;
Co-orientador: Hoppen, Carlos

Dissertação: Matemática Aplicada: Matemática Discreta e Combinatória,
Dissertação, Tese, Mestrado, Doutorado

Métodos Espectrais para Particionamento de Dados e Aplicações

por

Lucas Siviero Sibemberg

Dissertação submetida ao Programa de Pós-Graduação em Matemática Aplicada do Instituto de Matemática e Estatística da Universidade Federal do Rio Grande do Sul, como requisito parcial para a obtenção do grau de

Mestre em Matemática Aplicada

Linha de Pesquisa: Matemática Discreta e Combinatória

Orientador: Prof. Dr. Luiz Emilio Allem

Co-orientador: Prof. Dr. Carlos Hoppen

Banca examinadora:

Profa. Dra. Renata Raposo Del Vecchio
Universidade Federal Fluminense

Prof. Dr. Marcio Valk
Universidade Federal do Rio Grande do Sul

Prof. Dr. Vilmar Trevisan
Universidade Federal do Rio Grande do Sul

Prof. Dr. Rodrigo Orsini Braga
Universidade Federal do Rio Grande do Sul

Dissertação apresentada e aprovada em fevereiro de 2022.

Prof. Dr. Lucas Oliveira
Coordenador

SUMÁRIO

RESUMO

Atualmente temos uma grande quantidade de dados disponíveis e é uma tarefa muito difícil interpretá-los. Desta maneira, classificar esses dados em um pequeno número de grupos baseado em suas afinidades pode ajudar a obter informações valiosas sobre eles. Este é o objetivo dos algoritmos de clusterização (particionamento), que buscam dividir dados em um determinado número de clusters (grupos) de forma que dados que possuam mais afinidade fiquem no mesmo cluster e dados com menos afinidade fiquem em clusters diferentes. Nesta dissertação trabalhamos com métodos espectrais para particionamento de dados, que usam ingredientes de álgebra linear e teoria espectral de grafos.

Em nossa primeira contribuição apresentamos os resultados que obtivemos em duas aplicações das técnicas espectrais. A primeira aplicação está relacionada ao mercado financeiro, onde apresentamos uma estratégia em que clusterizamos um conjunto de ações e utilizamos critérios relacionados ao factor investing para montar portfólios. A segunda aplicação está relacionada à pandemia da COVID-19, onde obtivemos uma classificação do estado do Rio Grande do Sul em três clusters (regiões) de risco, alto risco, médio risco e baixo risco.

Terminamos apresentando um novo algoritmo de clusterização espectral, mais especificamente desenvolvemos uma nova medida de similaridade. A nossa medida apresenta uma série de vantagens: (1) o usuário não precisa definir nenhum parâmetro para utilizar a medida, tornando-a fácil de aplicar; (2) a medida é invariante sob translações e expansões; (3) a medida apresentou bom desempenho em conjuntos de dados sintéticos e, em situações reais, apresentou desempenho similar a outros métodos existentes, que precisam de pelo menos um parâmetro de escala definido pelo usuário para serem utilizados.

ABSTRACT

Nowadays we have a large amount of data available and it is a very difficult task to interpret it. In this way, classifying this data into a small number of groups based on their affinities can help to obtain valuable insight about them. This is the aim of clustering (partitioning) algorithms, which seek to split data into a certain number of clusters (groups) so that data with more affinity lie in the same cluster and data with less affinity lie in different clusters. In this dissertation we work with spectral methods for data partitioning, which use ingredients from linear algebra and spectral graph theory.

In our first contribution we present the results we obtained in two applications of spectral techniques. The first application is related to the financial market, where we present a strategy in which we cluster a set of stocks and use criteria related to the factor investing to build portfolios. The second application is related to the COVID-19 pandemic, where we obtained a classification of the state of Rio Grande do Sul in three clusters (regions) of risk, high risk, medium risk and low risk.

We finish presenting a new spectral clustering algorithm, more specifically, we developed a new similarity measure. Our measure has a number of advantages: (1) the user does not need to define any parameters to use the measure, making it easy to apply; (2) the measure is invariant under translations and expansions; (3) the measure performed well in synthetic data sets and, in real situations, it performed similarly to other existing methods, which need at least one user-defined scale parameter to be used.

1 INTRODUÇÃO

Atualmente temos uma grande quantidade de dados disponíveis e é uma tarefa muito difícil interpretá-los. Classificar esses dados em um pequeno número de grupos pode nos ajudar a obter informações valiosas sobre nossos dados. Este é o objetivo dos algoritmos de clusterização, também conhecidos como algoritmos de agrupamento ou particionamento, que buscam dividir dados em um determinado número de clusters de forma que dados com propriedades semelhantes fiquem no mesmo cluster e dados diferentes fiquem em clusters diferentes [74]. As técnicas de clusterização têm sido amplamente aplicadas em muitos campos. Por exemplo, elas desempenham um papel fundamental em problemas como segmentação de imagem [90], reconhecimento de padrões [77] e separação de fala [8]. Elas também aparecem naturalmente em problemas do mundo real em áreas como biologia [52], ciência da computação [83], economia [44], medicina [62] e outros.

Resolver problemas de classificação de dados é algo muito antigo. Afinal, uma das habilidades mais básicas dos seres vivos envolve o agrupamento de objetos. Por exemplo, o homem primitivo foi capaz de perceber que muitos objetos compartilhavam de certas propriedades, os alimentos poderiam ser venenosos, amargos, adocicados, entre outros, os animais poderiam ser ferozes, dóceis, domesticáveis, entre outros. Ao longo do desenvolvimento da humanidade, mais classificações foram feitas. Na comunicação, passamos a classificar palavras em grupos, por exemplo em relação à sua função gramatical, como substantivo, adjetivo, verbo, entre outros. Passamos a dividir os animais em grupos, por exemplo em espécies, como gatos, cachorros, cavalos, etc. Assim, ao longo da história diversos trabalhos relacionados a classificação de dados foram produzidos [37].

Durante muito tempo a classificação de dados foi feita de uma maneira subjetiva [35] e, por um lado, ainda é, visto que os problemas de classificação de dados são fortemente associados ao contexto que estão inseridos. O termo clusteri-

zação de dados apareceu pela primeira vez em 1954 no título de um trabalho sobre análise de dados antropológicos [84]. Acompanhando o desenvolvimento tecnológico mais dados passaram a ser gerados, ficando cada vez mais difícil clusterizar dados manualmente. Isso motivou o desenvolvimento dos primeiros métodos automáticos de clusterização.

As heurísticas de clusterização hierárquica foram apresentadas já em 1963 [34] e desenvolvidas ao longo do século XX, como por exemplo em [18, 68]. Esses métodos são conhecidos por serem simples e fornecerem boas interpretações sobre certas classes de dados, porém muitas vezes não podem lidar com estruturas mais complexas [15].

Outros métodos também foram introduzidos ao longo do mesmo século. O Algoritmo k-means é um dos algoritmos mais famosos desenvolvidos nessa época. O termo k-means foi apresentado pela primeira vez em 1967 por MacQueen em [43] em um artigo com mais de 33 mil citações, de acordo com o Google Acadêmico[1]. O primeiro algoritmo com a abordagem do Algoritmo k-means data de 1957 e foi proposto por Lloyd, porém só foi publicado em 1982 [41]. Muitas vezes é utilizado o nome de Algoritmo Lloyd-Forgy para o Algoritmo k-means, visto que em 1965 Forgy publicou um método essencialmente igual ao Algoritmo k-means [24]. O Algoritmo k-means é conhecido por lidar bem com conjuntos de dados que estejam distribuídos em regiões convexas, por isso o método não é recomendado em situações onde o conjunto não esteja dividido em regiões desse tipo.

Em meados dos anos 1970 começaram a surgir algoritmos de clusterização baseados em teoria de grafos e álgebra linear, hoje conhecidos como métodos de clusterização espectral. Para um método de clusterização espectral, é definida uma medida de similaridade que é tipicamente utilizada para mapear os elementos

[1]Disponível em: https://scholar.google.com.br/. Acesso em: 10 fev. 2022.

originais em um novo espaço euclidiano, realçando similaridades entre objetos - que vão além da distância entre dois elementos.

Luxburg [78] apresentou uma visão geral sobre a história da clusterização espectral. Segundo esse trabalho, o primeiro método espectral foi proposto por Donath e Hoffman em 1973 [19], onde os autores sugeriram que fosse construída uma partição de um grafo baseada nos autovetores da matriz de adjacência do grafo. No mesmo ano, Fiedler [22] descobriu que as bipartições do grafo estão fortemente relacionadas com o autovetor associado ao segundo menor autovalor da matriz laplaciana associada ao grafo. Mais precisamente, é possível bisectar um grafo baseado nas entradas positivas e negativas desse autovetor. Entre 1973 e 1999, diversos trabalhos sobre clusterização espectral foram publicados [27, 28, 51, 69].

A clusterização espectral se popularizou na comunidade de aprendizado de máquina, ou *machine learning*, mais especificamente na área de classificação de dados, a partir dos trabalhos de Shi e Malik [65] em 2000 e Ng *et al.* [49] em 2001, que, de acordo com o Google Acadêmico, possuem 20 mil e 10 mil citações, respectivamente. Durante o século XXI a clusterização espectral foi foco de pesquisa em muitos trabalhos, que vão desde melhorias em métodos conhecidos até a criação de métodos inteiramente novos. Como exemplo, citamos [89, 92, 33, 47].

Em seu trabalho, Ng *et al.* [49] afirmam que desconhecem métodos simples como o método espectral de [49] que atinjam resultados similares em conjuntos de dados desafiadores. E, de fato, esses novos algoritmos espectrais são conhecidos por possuir vantagens em relação aos algoritmos tradicionais apresentados no século XX, como o Algoritmo k-means [78]. Por exemplo, quando o Algoritmo k-means atinge um bom resultado em um conjunto de dados, provavelmente o algoritmo espectral obterá um resultado pelo menos tão bom, porém o contrário não é verdade. Vale mencionar que esse reconhecimento da qualidade das partições produzidas por métodos espectrais até o momento não se refletiu em muitos resultados teóricos sobre o seu desempenho.

Essa dissertação tem por objetivo geral estudar os métodos espectrais, buscando entender (1) como esses métodos se adequam em problemas de classificação reais e (2) como seus parâmetros e ingredientes afetam o seu desempenho. Esse trabalho possui três objetivos específicos.

(a) Apresentar dois métodos clássicos de classificação de dados, buscando entender as suas vantagens e desvantagens.

(b) Estudar a fundamentação teórica que motiva um método espectral e identificar possíveis generalizações.

(c) Realizar aplicações dos métodos espectrais em contextos reais, buscando avaliar o desempenho desses métodos em situações complexas.

Toda a programação realizada nesse trabalho foi desenvolvida na linguagem de programação python[2].

O trabalho está dividido em cinco capítulos, além da introdução. No Capítulo 2, formalizamos algumas noções de ciência de dados, como conjuntos de dados e problema de clusterização. Além disso, apresentamos duas heurísticas comumente utilizadas para encontrar soluções para o problema de classificação de dados, o Algoritmo single linkage e o Algoritmo k-means. Ainda apresentamos as demonstrações das propriedades fundamentais do Algoritmo k-means, que raramente estão presentes na literatura dessa área.

No Capítulo 3, apresentamos a família de algoritmo de clusterização espectral. Damos destaque para o algoritmo espectral de Ng *et al.* [49] e apresentamos a sua motivação teórica em teoria de grafos e álgebra linear. Também apresentamos a estrutura geral de um algoritmo de clusterização espectral, identificando os maiores focos de pesquisa na área de métodos espectrais. Por fim, apresentamos diversas aplicações de um método espectral em diferentes conjuntos de dados.

[2]Disponível em: https://www.python.org/. Acesso em: 10 fev. 2022.

O Capítulo 4 apresenta duas aplicações de nossa autoria, do algoritmo de clusterização espectral a problemas com dados reais. O objetivo dessas aplicações era avaliar o desempenho de algoritmos de clusterização em problemas complexos. A primeira aplicação está relacionada ao mercado financeiro, onde temos por objetivo formar um portfólio de ações que possua um bom desempenho sem assumir muito risco. A segunda aplicação está relacionada à pandemia da COVID-19, onde temos por objetivo obter uma classificação de risco do estado do Rio Grande do sul e avaliar a qualidade dessa classificação.

No Capítulo 5 apresentamos o nosso principal resultado desenvolvido no período do mestrado, que consiste em uma nova medida de similaridade para o algoritmo de clusterização espectral. Definir a medida de similaridade é muito importante, pois é a partir da similaridade entre cada par de objetos de um conjunto de dados que o método espectral realiza a classificação. A nossa medida de similaridade apresenta uma série de vantagens. O usuário não precisa definir nenhum parâmetro para utilizar a medida, tornando-a fácil de aplicar. Ela é invariante sob translações e expansões e é fácil de ser calculada. Além disso, ela apresentou bom desempenho em conjuntos de dados sintéticos e, em situações reais, apresentou desempenho similar a outros métodos existentes, que precisam de pelo menos um parâmetro definido pelo usuário para serem utilizados.

No Capítulo 6, serão apresentadas as considerações finais e ideias para trabalhos futuros.

2 CLUSTERIZAÇÃO

Esse capítulo apresenta uma introdução ao problema de clusterização e algumas heurísticas utilizadas para lidar com ele. A primeira seção apresenta ao leitor desde a definição de conjunto de dados até a formulação do problema de clusterização associado a uma função objetivo. A segunda seção inclui duas heurísticas que são comumente utilizadas em problemas de classificação de dados.

2.1 Formulação do Problema de Clusterização

Para formular o problema de clusterização, primeiramente é necessário definir o que é um conjunto de dados e uma partição dele. Seja $X = \{x_1, \ldots, x_n\}$ um conjunto finito de objetos, definimos a função de $\mathcal{O} : X \to \mathbb{R}^m$ que associa cada objeto a um vetor com coordenadas que representam suas características ou observações feitas sobre ele. Assim, dado $i \in [n] = \{1, \ldots, n\}$, $\mathcal{O}(x_i) = (x_1^{(i)}, \ldots, x_m^{(i)})$, onde $x_j^{(i)}$ é a j-ésima observação do objeto i de X. Citamos um exemplo de conjunto de dados em um contexto real.

Exemplo 2.1. Um conjunto de dados clássico, muito utilizado para fazer testes de classificação de dados, é conhecido como Iris [20]. O conjunto de dados $X = \{x_1, \ldots, x_{150}\}$ consiste em um conjunto de 150 plantas do gênero Iris. Neste caso, a função $\mathcal{O}$ é definida pelas medidas de estruturas da planta conhecidas como sépala e pétala, ilustradas na Figura 2.1. Assim, $\mathcal{O}(x_i) = (x_1^{(i)}, x_2^{(i)}, x_3^{(i)}, x_4^{(i)})$, onde $x_1^{(i)}$ é o comprimento da sépala da planta x_i, $x_2^{(i)}$ é a largura dessa sépala, $x_3^{(i)}$ é o comprimento da pétala de x_i e $x_4^{(i)}$ é a largura dessa pétala. No conjunto de dados essas medidas são fornecidas em centímetros.

No conjunto X, foram incluídos 50 exemplares de cada uma das seguintes espécies: *setosa*, *versicolor* e *virginica*. Um exemplo de cada uma delas aparece na Figura 2.1.

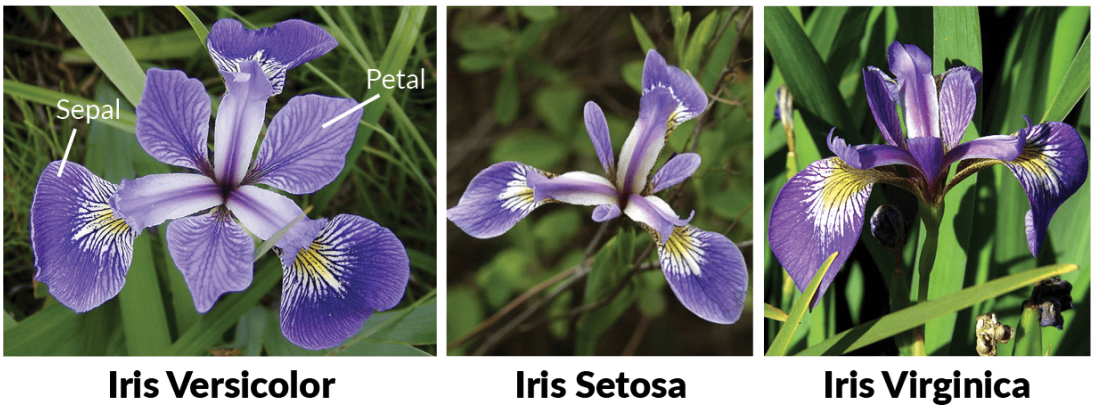

Figura 2.1: Exemplar de cada espécie de Iris.

Disponível em:

`https://www.datacamp.com/community/tutorials/machine-learning-in-r`.

Acesso em: 31 jan. 2022.

Um exemplo de problema de clusterização é justamente utilizar as observações de cada flor em X para classificar a sua espécie, buscando encontrar a partição original.

Nessa seção apresentamos o problema de clusterização em um conjunto de dados qualquer. Dessa forma, definimos o que é uma partição do conjunto X.

Definição 2.1. *Seja $k \in \mathbb{N}$. Dizemos que $\mathcal{C} = \{C_1, \ldots, C_k\}$ é uma partição de X se:*

(i) $C_\ell \neq \emptyset$, $\forall \ell \in [k]$,

(ii) $C_a \cap C_b = \emptyset$, $\forall a, b \in [k]$,

(iii) $\cup_{\ell=1}^{k} C_\ell = X$.

Cada $C_\ell \in \mathcal{C}$ é chamado de cluster.

Assim, particionar um conjunto de dados se resume a dividi-lo em subconjuntos disjuntos não-vazios complementares. Note que é bem simples encontrar

uma partição de um conjunto qualquer se não impusermos restrições sobre o tipo de divisão desejada. Assim, para definir o problema de clusterização buscamos uma partição do conjunto de dados que otimize alguma função objetivo. Mais formalmente, dados $X \subset \mathbb{R}^m$ finito, $k \in \mathbb{N}$ e $\mathcal{U}_k$ o conjunto que contém todas partições possíveis de X em k clusters, buscamos encontrar a partição $\mathcal{C}$ de X que minimize uma função objetivo $F : \mathcal{U}_k \to \mathbb{R}$, que representa o custo da partição. A função objetivo pode ser escolhida de diversas maneiras. Uma possibilidade é a função objetivo $F_1 : \mathcal{U}_k \to \mathbb{R}$, tal que

$$F_1(\mathcal{C}) = \sum_{\ell \in [k]} \frac{1}{|C_\ell|} \sum_{i,j \in C_\ell} ||x_i - x_j||^2, \tag{2.1}$$

onde $\mathcal{C} = \{C_1, \ldots, C_k\}$.

Nessa definição estão sendo levados em conta alguns fatores. Primeiramente, a função F_1 contabiliza somente dados que pertençam a um mesmo cluster, onde cada cluster contribui com $\frac{1}{|C_\ell|} \sum_{i,j \in C_\ell} ||x_i - x_j||^2$. Assim, podemos entender que a função F_1 é uma soma da média das distâncias entre pares de elementos de cada cluster. Note que se os pontos de um mesmo cluster estiverem próximos, a soma das distâncias desse cluster será um valor relativamente baixo. Em geral, gostaríamos que pontos próximos tendessem a estar em um mesmo cluster e pontos distantes, em clusters diferentes. Uma partição $\mathcal{C}$ que minimize a função F_1 é chamada de partição ótima em relação a F_1.

Exemplo 2.2. Para ilustrar essa situação consideramos o conjunto de dados X em $\mathbb{R}^2$ presente na Figura 2.2, onde $|X| = 25$. A função $\mathcal{O}$ simplesmente associa cada ponto às suas coordenadas em $\mathbb{R}^2$. O objetivo é encontrar a partição ótima de X em $k = 2$ clusters com respeito à função F_1, ou seja, queremos encontrar a partição $\mathcal{C}^{(0)}$ de X tal que $F_1(\mathcal{C}^{(0)}) \leq F_1(\mathcal{C})$, para todo $\mathcal{C} \in \mathcal{U}_k$.

Na Figura 2.3 apresentamos duas partições possíveis de X. Na esquerda, a partição ótima $\mathcal{C}^{(0)}$, na qual $F_1(\mathcal{C}^{(0)}) \approx 35$. À direita, uma partição $\mathcal{C}^{(1)}$ de X, na qual $F_1(\mathcal{C}^{(1)}) \approx 1648$. Uma maneira de verificar que $\mathcal{C}^{(0)}$ é a partição ótima é calcular

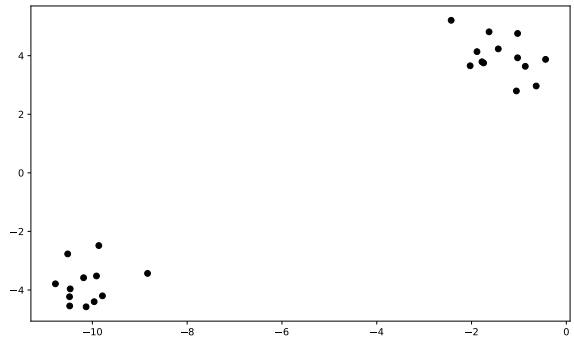

Figura 2.2: Conjunto de dados X.

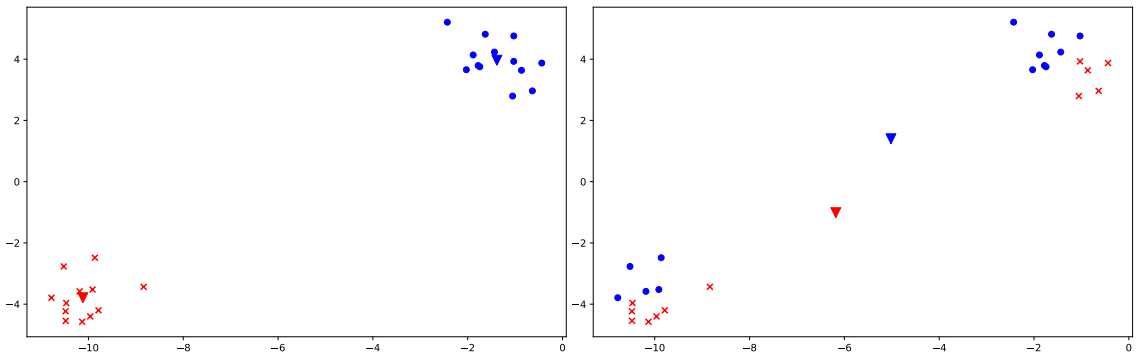

Figura 2.3: Duas partições possíveis do conjunto de dados X. Pontos com a mesma cor e o mesmo formato pertencem ao mesmo cluster. Cada ponto em formato de triângulo representa o ponto médio do cluster colorido com a sua cor.

o valor da função F_1 para todas partições em dois clusters de X, ou seja, avaliando as 16.777.215 partições diferentes de X verificamos que $\mathcal{C}^{(0)}$ é a partição que minimiza F_1. Mesmo sem fazer este cálculo é possível ter uma intuição que $\mathcal{C}^{(0)}$ é a partição ótima, pois qualquer outra partição teria um cluster com pelo menos um par de pontos mais distantes, como ilustrado na Figura 2.3 (direita), que faria o valor da função F_1 aumentar.

Durante esse trabalho, para simplificar, quando X for um conjunto de pontos em um espaço euclidiano, vamos considerar $X = \{x_1, \ldots, x_n\}$ tal que $\mathcal{O}(x_i) = x_i$.

No Exemplo 2.2 encontramos a partição ótima esgotando todas as partições do conjunto de dados. Mais geralmente, para um conjunto de dados $X \subset \mathbb{R}^m$, onde $|X| = n$, nessa estratégia precisaríamos verificar o número de partições possíveis de X em $k = 2$ clusters, que é igual a $2^{n-1} - 1$.

O problema de clusterização com a função objetivo F_1 é NP-difícil, veja [1]. Supondo que P$\neq$NP, isso significa que não existe algoritmo que responda à seguinte pergunta em tempo polinomial (com respeito a n): dados $n \in \mathbb{N}$, um conjunto de dados $X \subset \mathbb{R}^m$ com $|X| = n$ e $\delta > 0$, existe uma partição $\mathcal{C}$ de X em k clusters tal que $F_1(\mathcal{C}) \leq \delta$?

No Exemplo 2.2 é possível utilizar o ponto médio do cluster, conhecido como centroide, para representar todo o cluster. Note que os centroides da Figura 2.3 (esquerda) constituem uma representação mais próxima de cada cluster do que os centroides da Figura 2.3 (direita). Em conjuntos com muitos dados, é útil representar clusters por meio de um único ponto, visto que isso reduz a necessidade de armazenar conjuntos de dados muito grandes.

Seguindo a ideia de representar os clusters por centroides, também é possível utilizar uma função objetivo motivada pelos centroides. Dados um conjunto de dados X e um inteiro positivo k, seja $J_1 : \mathcal{U}_k \to \mathbb{R}$ definida por

$$J_1(\mathcal{C}) = \sum_{\ell=1}^{k} \sum_{x \in C_\ell} ||x - \mu_\ell||^2, \tag{2.2}$$

onde $\mathcal{C} = \{C_1, \ldots, C_k\}$ é uma partição de X e $\mu_\ell = \frac{1}{|C_\ell|} \sum_{x \in C_\ell} x$ é o centroide do ℓ-ésimo cluster.

A função J_1 definida pela equação (2.2) tem semelhanças com a função F_1 definida pela equação (2.1), em particular se pontos de um mesmo cluster C_ℓ estão muito próximos uns dos outros, o centroide estará próximo dos pontos do cluster e, por conseguinte, a contribuição do cluster C_ℓ no valor da função J_1 será

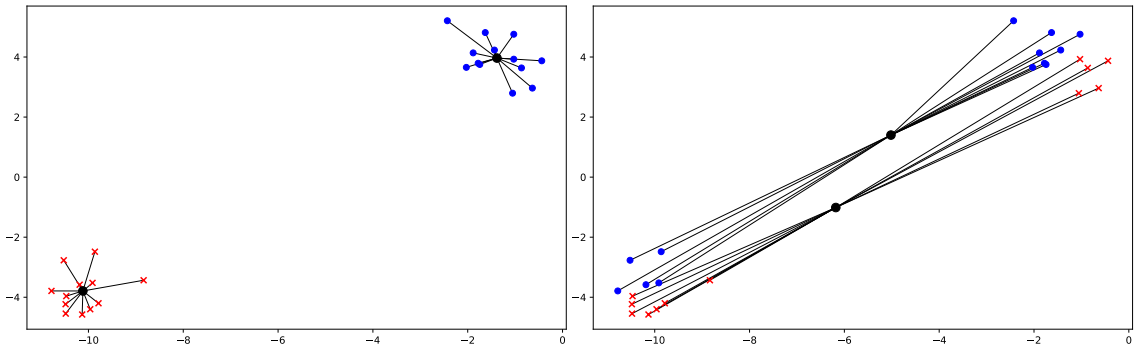

Figura 2.4: Duas partições possíveis do conjunto de dados, mostrando o centroide de cada cluster da partição. Cada aresta representa a distância do ponto do cluster ao centroide do mesmo.

baixo. Assim como no caso da função F_1, buscamos encontrar a partição $\mathcal{C}^{(0)}$ tal que $J_1(\mathcal{C}^{(0)}) \leq J_1(\mathcal{C})$, para toda partição $\mathcal{C}$ de um conjunto de dados X em k clusters.

Em relação aos dois casos ilustrados na Figura 2.3, temos que a partição $\mathcal{C}^{(0)}$ (esquerda) também é a partição ótima em relação a J_1, onde $J_1(\mathcal{C}^{(0)}) \approx 17.5$. Para a partição $\mathcal{C}^{(1)}$, apresentada na direita da Figura 2.3, $J_1(\mathcal{C}^{(1)}) \approx 824$. Visualmente podemos ver o porquê de $J_1(\mathcal{C}^{(0)})$ ser menor que $J_1(\mathcal{C}^{(1)})$ na Figura 2.4. A soma dos quadrados dos comprimentos das arestas de cada cluster da Figura 2.4 é a contribuição de cada cluster no valor da função J_1.

A seguir iremos mostrar que as funções J_1 e F_1 são equivalentes no seguinte sentido. Dados um conjunto de dados X e um inteiro positivo k, sejam $\mathcal{C}, \mathcal{D}$ partições quaisquer de X então $F_1(\mathcal{C}) < F_1(\mathcal{D})$ se, e somente se, $J_1(\mathcal{C}) < J_1(\mathcal{D})$. Logo, as partições ótimas em relação a F_1 e a J_1 são exatamente as mesmas. A equivalência de F_1 e J_1 é uma consequência imediata do Lema 2.1.

Lema 2.1. *Se $\mathcal{C}$ é uma partição de um conjunto de dados X, então $F_1(\mathcal{C}) = 2J_1(\mathcal{C})$.*

Demonstração. Vamos mostrar que

$$\frac{1}{|C_\ell|}\sum_{i\in C_\ell}\sum_{j\in C_\ell}||x_i - x_j||^2 = 2(\sum_{i\in C_\ell}||x_i||^2 - |C_\ell| \cdot ||\mu_\ell||^2) = 2\sum_{i\in C_\ell}||x_i - \mu_\ell||^2.$$

Por um lado,

$$\begin{aligned}
\sum_{i\in C_\ell}\sum_{j\in C_\ell}||x_i - x_j||^2 &= \sum_{i\in C_\ell}\sum_{j\in C_\ell}(||x_i||^2 + ||x_j||^2 - 2\langle x_i, x_j\rangle) \\
&= \sum_{i\in C_\ell}(\sum_{j\in C_\ell}||x_i||^2 + \sum_{j\in C_\ell}||x_j||^2 - 2\sum_{j\in C_\ell}\langle x_i, x_j\rangle) \\
&= |C_\ell|\sum_{i\in C_\ell}||x_i||^2 + \sum_{i\in C_\ell}\sum_{j\in C_\ell}||x_j||^2 - 2|C_\ell|\sum_{i\in C_\ell}\langle x_i, \mu_\ell\rangle \\
&= |C_\ell|\sum_{i\in C_\ell}||x_i||^2 + |C_\ell|\sum_{j\in C_\ell}||x_j||^2 - 2|C_\ell|\cdot|C_\ell|\langle \mu_\ell, \mu_\ell\rangle \\
&= 2|C_\ell|\sum_{i\in C_\ell}||x_i||^2 - 2|C_\ell|^2||\mu_\ell||^2 \\
&= 2|C_\ell|(\sum_{i\in C_\ell}||x_i||^2 - |C_\ell|\cdot||\mu_\ell||^2)
\end{aligned} \tag{2.3}$$

Por outro lado,

$$\begin{aligned}
\sum_{i\in C_\ell}||x_i - \mu_\ell||^2 &= \sum_{i\in C_\ell}(||x_i||^2 + ||\mu_\ell||^2 - 2\langle x_i, \mu_\ell\rangle) \\
&= \sum_{i\in C_\ell}||x_i||^2 + \sum_{i\in C_\ell}||\mu_\ell||^2 - \sum_{i\in C_\ell}2\langle x_i, \mu_\ell\rangle \\
&= \sum_{i\in C_\ell}||x_i||^2 + |C_\ell|\cdot||\mu_\ell||^2 - \sum_{i\in C_\ell}2\langle x_i, \mu_\ell\rangle \\
&= \sum_{i\in C_\ell}||x_i||^2 + |C_\ell|\cdot||\mu_\ell||^2 - 2|C_\ell|\cdot||\mu_\ell||^2 \\
&= \sum_{i\in C_\ell}||x_i||^2 - |C_\ell|\cdot||\mu_\ell||^2
\end{aligned} \tag{2.4}$$

Pelas equações (2.3) e (2.4), temos que

$$\frac{1}{|C_\ell|}\sum_{i\in C_\ell}\sum_{j\in C_\ell}||x_i - x_j||^2 = 2\sum_{i\in C_\ell}||x_i - \mu_\ell||^2.$$

Logo,

$$\sum_{\ell\in[k]}\frac{1}{|C_\ell|}\sum_{i,j\in C_\ell}||x_i - x_j||^2 = 2\sum_{\ell\in[k]}\sum_{i\in C_\ell}||x_i - \mu_\ell||^2,$$

concluindo a demonstração. □

Assim, não faz diferença buscar a partição ótima em relação a F_1 ou J_1, pois a partição é exatamente a mesma.

Até o momento formulamos o problema de clusterização e descrevemos versões do problema de clusterização para as quais é difícil encontrar uma partição ótima. Mostramos duas opções de funções objetivo e percebemos que resolver o problema de clusterização em relação a essas funções pode ser difícil, mas ainda não discutimos métodos para procurar soluções de boa qualidade para esse problema. O objetivo da próxima seção é apresentar duas heurísticas existentes, mostrando suas vantagens e desvantagens.

2.2 Heurísticas para o Problema de Clusterização

Nessa seção serão apresentados dois algoritmos de clusterização muito utilizados para classificações de dados. Podemos entender um algoritmo de clusterização como uma sequência finita de regras aplicadas a um conjunto de dados de entrada, tendo como saída uma partição desse conjunto. Nesses algoritmos, além do conjunto de dados, é comum o usuário fornecer o número de clusters k como entrada.

Pode passar despercebido, mas já citamos um algoritmo de clusterização nesse trabalho. Afinal calcular todas as partições possíveis de um conjunto de dados X e escolher aquela que obtenha melhor resultado com relação a alguma função objetivo é um algoritmo de clusterização. Mas é claro que ninguém considera utilizar esse algoritmo em alguma situação real com muitos dados, pois a quantidade de partições cresce exponencialmente à medida que o número de dados aumenta, sendo impraticável encontrar a partição ótima.

Antes de discutirmos algoritmos específicos é importante fazer um destaque. Comumente os algoritmos são representados por um pseudo-código, onde se

usa linguagem simples, sendo facilmente replicável em uma máquina por um usuário experiente. A seguir apresentamos os métodos de clusterização hierárquica.

2.2.1 A Clusterização Hierárquica

Com o objetivo de definir uma hierarquia de clusters de um conjunto de dados, existem os métodos de clusterização hierárquica.

Para definir um método hierárquico é necessária a noção de medida de dissimilaridade de cluster. Dado um conjunto de dados X, seja $\mathcal{C} = \{C_1, \ldots, C_k\}$ uma partição de X. Uma medida de dissimilaridade $D : \mathcal{C} \times \mathcal{C} \to \mathbb{R}$ atribui um valor real para cada par de clusters de C_ℓ, C_α. A interpretação dessa medida é que quanto maior é $D(C_\ell, C_\alpha)$, maior é a diferença entre os dois clusters. Por exemplo,

$$D_1(C_\ell, C_\alpha) = \min\{||x - y|| : x \in C_\ell, y \in C_\alpha\} \tag{2.5}$$

é uma medida de dissimilaridade, onde $||x - y||$ é a distância euclidiana entre x e y. Perceba que, em D_1, a dissimilaridade entre dois clusters é dada pela distância entre o par $x \in C_\ell$, $y \in C_\alpha$ mais próximo.

Dado um conjunto de dados X, os algoritmos de clusterização hierárquica utilizam duas entradas: um inteiro positivo k e uma medida de dissimilaridade de cluster $D : \mathcal{C} \times \mathcal{C} \to \mathbb{R}$. A partir dessas entradas, um método de clusterização hierárquica consiste em um, dentre dois tipos:

- Aglomerativo: Todos os pontos começam em clusters distintos e passo a passo combinamos dois clusters que otimizem uma medida de dissimilaridade D, até que todos pontos estejam em um mesmo cluster.
- Divisivo: Todos os pontos começam em um mesmo cluster e passo a passo dividimos em dois, o cluster que otimiza uma medida de dissimilaridade D.

Independentemente de ser aglomerativo ou divisivo, todo método de clusterização hierárquica forma uma hierarquia de clusters, pois o método constrói uma cadeia de partições de X, $\mathcal{C}^{(1)}, \mathcal{C}^{(2)}, \ldots, \mathcal{C}^{(n)}$, onde a i-ésima partição é obtida a partir de uma iteração do método na $(i-1)$-ésima partição. Se buscamos uma partição de X em k clusters, basta considerar a partição $\mathcal{C}^{(n-k)}$, no caso do método aglomerativo e $\mathcal{C}^{(k)}$, no caso do método divisivo. Essa sequência de partições é representada por meio de um dendrograma [18]. O dendrograma é um diagrama, em forma de árvore, que se ramifica conforme um novo passo do método é realizado.

A medida de dissimilaridade D_1 (equação (2.5)) motiva o algoritmo de clusterização hierárquica single linkage [68], apresentado no Algoritmo 1.

Algoritmo 1: Single Linkage

Entrada: $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^m$

Saída: Sequência de partições $\mathcal{C}^{(1)}, \mathcal{C}^{(2)}, \ldots, \mathcal{C}^{(n)}$

1 Inicialize com a partição $\mathcal{C}^{(1)} = \{C_1, \ldots, C_n\}$ tal que $C_i = \{x_i\}$ $\forall i \in \{1, \ldots, n\}$.

2 **para** $t \in \{1, \ldots, n-1\}$ **faça**

3 Seja $(C_\alpha^{(t)}, C_\beta^{(t)})$, $\alpha < \beta$, o par de clusters que atinge o valor $\min\{D_1(A, B) : A, B \in \mathcal{C}\}$. Se existirem $(\alpha_1, \beta_1), \ldots, (\alpha_q, \beta_q)$ tal que $(C_{\alpha_j}^{(t)}, C_{\beta_j}^{(t)})$ atinge o valor mínimo, seja α o índice tal que $\alpha \leq \alpha_j, \forall j \in \{1, \ldots, q\}$. Escolha β tal que $\beta \leq \beta_j$, para todo (α_j, β_j) em que $\alpha = \alpha_j$.

4 Defina $C_\alpha^{(t+1)} = C_\alpha^{(t)} \cup C_\beta^{(t)}$.

5 Defina $C_\ell^{(t+1)} = C_\ell^{(t)}$, para todo $C_\ell^{(t)} \in \mathcal{C}^{(t)}$, onde $\ell \neq \alpha, \beta$.

6 Defina $\mathcal{C}^{(t+1)} = \{C_\ell^{(t+1)}\}_\ell \cup \{C_\alpha^{(t+1)}\}$

7 **fim**

8 **Retorne** $\mathcal{C}^{(1)}, \mathcal{C}^{(2)}, \ldots, \mathcal{C}^{(n)}$

O exemplo 2.3 apresenta uma aplicação do Algoritmo single linkage.

Exemplo 2.3. Seja $X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\}$, onde $x_1 = (1, 1)$, $x_2 = (1, 1.5)$, $x_3 = (2, 1)$, $x_4 = (3, 1.5)$, $x_5 = (7.5, 3)$, $x_6 = (8.5, 3)$, $x_7 = (8, 5)$, $x_8 = (8, 6)$. A Figura 2.5 retrata esses pontos no plano $\mathbb{R}^2$.

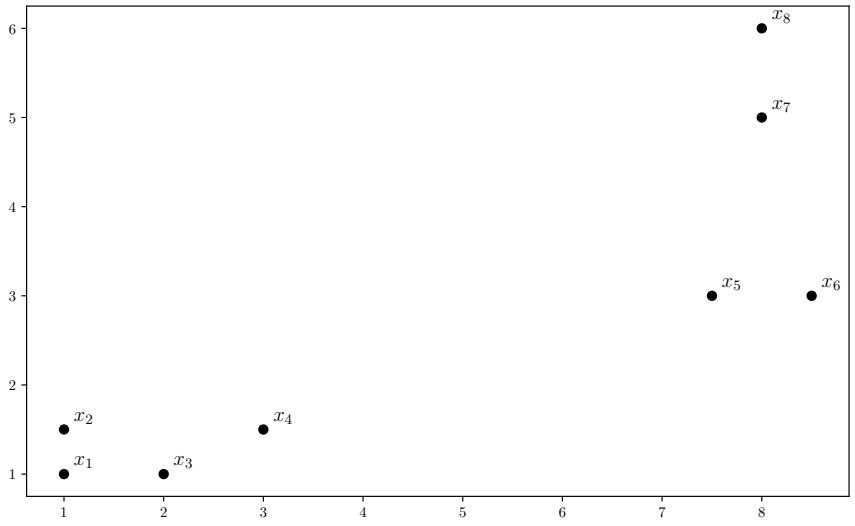

Figura 2.5: Conjunto de dados X.

Ao utilizar o Algoritmo 1 em X iniciamos com a partição $\mathcal{C}^{(1)} = \{C_1^{(1)}, \ldots, C_8^{(1)}\}$, onde $C_1^{(1)} = \{x_1\}$, $C_2^{(1)} = \{x_2\}$, $C_3^{(1)} = \{x_3\}$, $C_4^{(1)} = \{x_4\}$, $C_5^{(1)} = \{x_5\}$, $C_6^{(1)} = \{x_6\}$, $C_7^{(1)} = \{x_7\}$, $C_8^{(1)} = \{x_8\}$. No passo $t = 2$, percebemos que $\min(D_1)$ é atingido por $C_1^{(1)}$ e $C_2^{(1)}$, pois neste passo a distância entre os clusters é exatamente a distância entre cada par de pontos. Retornando os clusters $C_1^{(2)} = \{x_1, x_2\}$, $C_3^{(2)} = \{x_3\}$, $C_4^{(2)} = \{x_4\}$, $C_5^{(2)} = \{x_5\}$, $C_6^{(2)} = \{x_6\}$, $C_7^{(2)} = \{x_7\}$, $C_8^{(2)} = \{x_8\}$. Já no passo $t = 3$, há três pares de clusters que estão à mesma distância $\mathcal{C}_1^{(2)} = \{x_1, x_2\}$ e $\mathcal{C}_3^{(2)} = \{x_3\}$, $\mathcal{C}_5^{(2)} = \{x_5\}$ e $\mathcal{C}_6^{(2)} = \{x_6\}$, $\mathcal{C}_7^{(2)} = \{x_7\}$ e $\mathcal{C}_8^{(2)} = \{x_8\}$. De acordo com o Algoritmo single linkage combinamos $\mathcal{C}_1^{(2)}$ e $\mathcal{C}_3^{(2)}$, obtendo $\mathcal{C}_1^{(3)} = \{x_1, x_2, x_3\}$, $C_4^{(3)} = \{x_4\}$, $C_5^{(3)} = \{x_5\}$, $C_6^{(3)} = \{x_6\}$, $C_7^{(3)} = \{x_7\}$, $C_8^{(3)} = \{x_8\}$. Assim seguimos até que $\mathcal{C}^{(7)} = \{\mathcal{C}_1^{(7)}\} = \{\{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\}\}$. Após esse processo obtemos o dendrograma representado na Figura 2.6.

Observe que, se buscássemos uma partição de X em dois clusters, bastaria escolher a partição $\mathcal{C}^{(6)}$ na sequência de partições obtida pelo Algoritmo single linkage, resultando na partição $\{\{x_1, x_2, x_3, x_4\}, \{x_5, x_6, x_7, x_8\}\}$ de X.

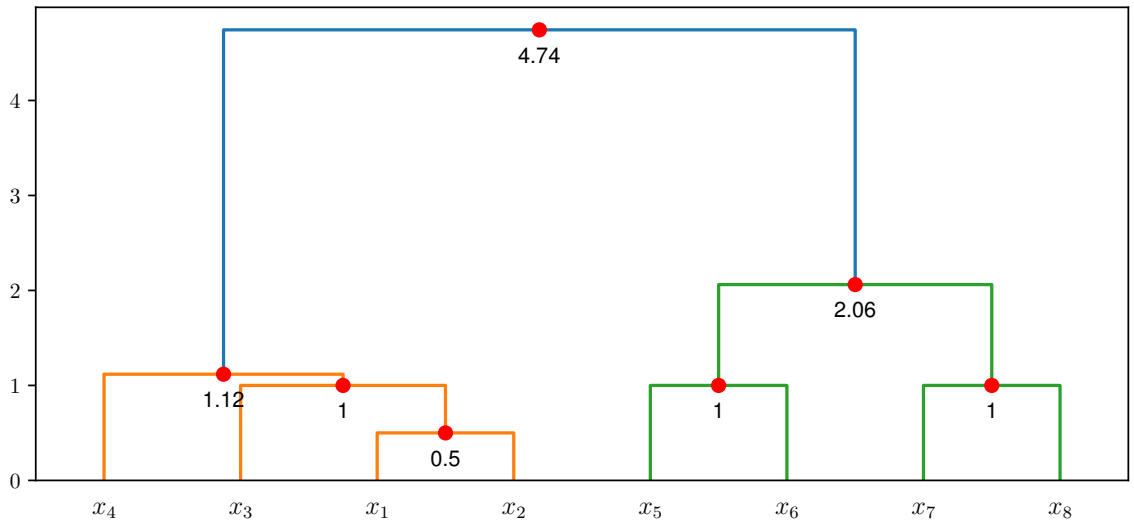

Figura 2.6: Dendrograma de X obtido utilizando o Algoritmo single linkage. Os valores associados aos pontos em vermelho representam a distância em que os dois clusters, que se juntaram, estavam.

Normalmente, a medida de dissimilaridade de cluster motiva o nome do método hierárquico, por exemplo o Average Linkage utiliza a medida $D_2(A, B) = \frac{1}{|A||B|} \sum_{x \in A, y \in B} ||x - y||$, onde $|| \cdot ||$ é uma métrica de distância. Algumas medidas de dissimilaridade definidas na literatura [68, 70, 18] são:

- $D_1(A, B) = \min\{||x - y|| : x \in A, x \in B\}$.
- $D_2(A, B) = \frac{1}{|A||B|} \sum_{x \in A, y \in B} ||x - y||$.
- $D_3(A, B) = \max\{||x - y|| : x \in A, x \in B\}$.
- $D_4(A, B) = ||\mu_A - \mu_B||$, onde μ_A, μ_B são os centroides de A e B, respectivamente.

Um benefício claro dos métodos de clusterização hierárquica é a garantia de que terminam em n passos.

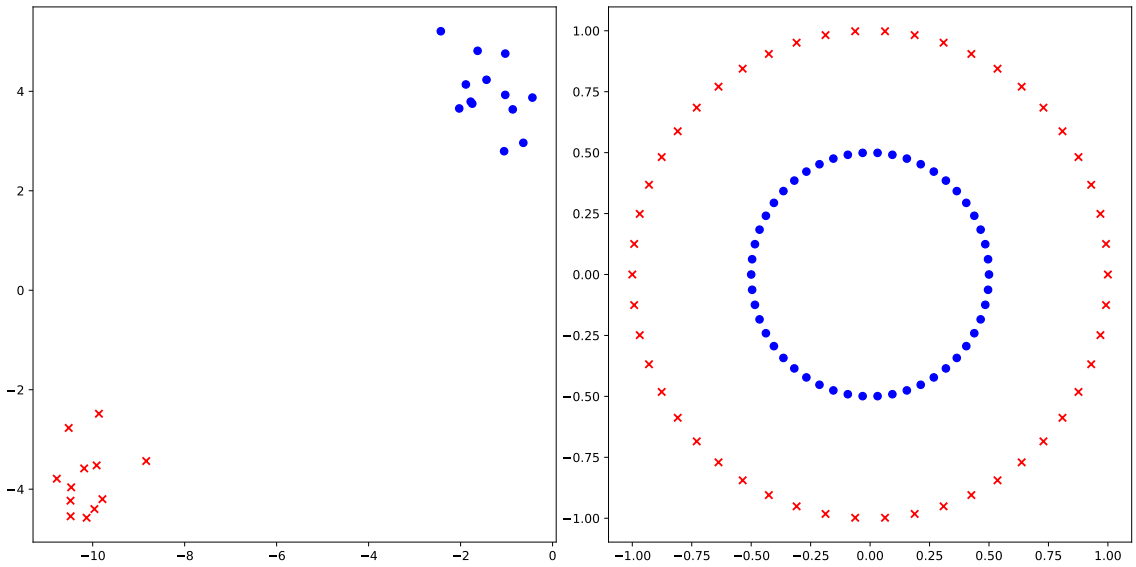

Figura 2.7: Na esquerda, resultado do Algoritmo 1 para o conjunto de dados X' (2.2). Na direita, resultado do Algoritmo 1 para um conjunto de dados com duas circunferências.

No exemplo 2.4, apresentamos o resultado do Algoritmo single linkage em dois conjuntos de dados distintos.

Exemplo 2.4. Na Figura 2.7 (direita) retratamos o mesmo conjunto de dados utilizado no exemplo 2.2. Na Figura 2.7 (esquerda) temos as circunferências C_1 e C_2 de raio 0.5 e 1, respectivamente. A partir delas definimos o conjunto de dados $X = C_1 \cup C_2$. É claro que, ao clusterizar X em $k = 2$ clusters, gostaríamos de obter cada circunferência em um cluster.

Na Figura 2.7 (esquerda) percebemos que o Algoritmo single linkage obtém a partição ótima em relação a J_1 para o conjunto de dados em questão. Na Figura 2.7 (direita) também obtemos o resultado que gostaríamos com o Algoritmo 1.

Apesar de bons resultados em alguns conjuntos de dados, como ilustrado no exemplo 2.4, os algoritmos de clusterização hierárquica são muito sensíveis a perturbações e *outliers* no conjunto de dados, onde *outliers* são elementos fora do

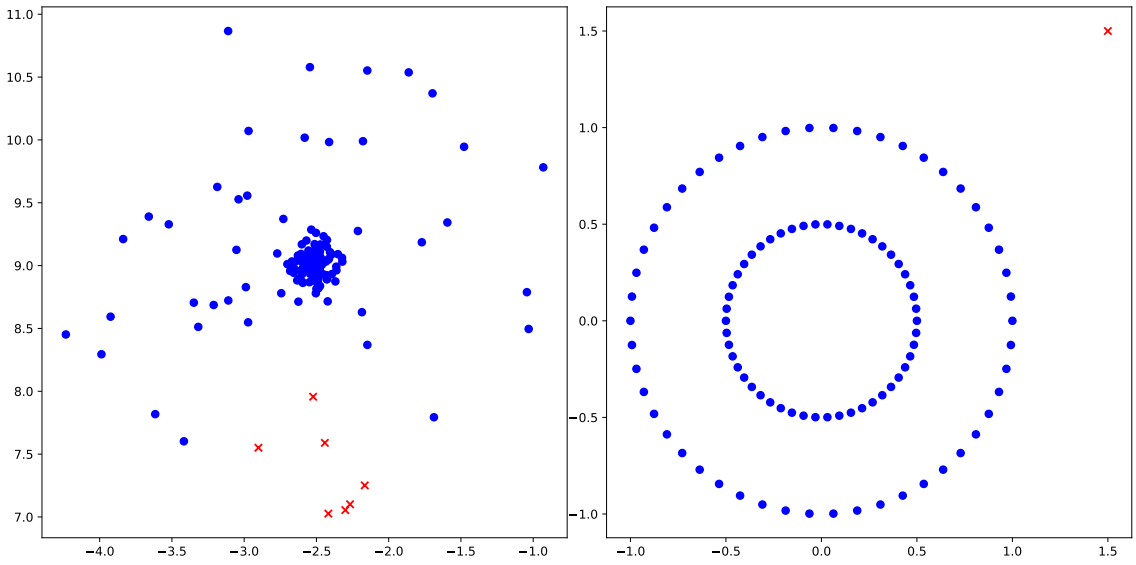

Figura 2.8: Na esquerda, o resultado do Algoritmo 1 para o conjunto de dados que contém duas nuvens. Na direita, o resultado do Algoritmo 1 para um conjunto de dados com duas circunferências e um *outlier*.

padrão do conjunto de dados. Para exemplificar isso, aplicamos o Algoritmo single linkage em dois novos conjuntos de dados.

Exemplo 2.5. Na Figura 2.8 (esquerda) mostramos um conjunto de dados que contêm duas nuvens, uma, no meio, com pontos muito próximos uns dos outros e outra, no entorno da primeira, onde os pontos estão distribuídos de forma esparsa no plano. O conjunto de dados é a união das duas nuvens. Dessa forma, a expectativa era que o algoritmo dividisse o conjunto em $k = 2$ clusters, um para cada nuvem. O Algoritmo 1 primeiro combina todos pontos da nuvem interna e depois combina os pontos da nuvem interna com os pontos externos mais próximos a ela. Se selecionarmos a partição da saída do algoritmo com exatamente $k = 2$ clusters obteremos a partição ilustrada na Figura 2.8 (esquerda).

Já na Figura 2.8 (direita), acrescentamos um ponto ao conjunto de dados da Figura 2.7 (direita), um *outlier* x', que pode ser visto no canto superior direito da Figura 2.7 (direita). Note que, após as duas circunferências C_1 e C_2 se

formarem, o algoritmo está numa etapa onde existem três clusters C_1, C_2 e $\{x'\}$. Nesse momento, o algoritmo enxerga que $d_1(C_1, C_2) < d_1(C_i, \{x'\}), i \in \{1, 2\}$ e portanto mescla as circunferências em um só cluster, resultando na partição ilustrada na Figura 2.8 (direita).

Apesar de suas fragilidades, métodos hierárquicos são muito utilizados pela sua simplicidade de implementação e resultados que podem ajudar a interpretar os conjuntos de dados, mas, usualmente, não fornecem bons resultados em situações mais complexas [15].

2.2.2 O Algoritmo k-means

O Algoritmo k-means é um algoritmo de clusterização que busca dividir um conjunto de dados X em k clusters em que cada ponto de X pertence ao cluster com o centroide mais próximo. Mais precisamente, o Algoritmo k-means é um algoritmo iterativo baseado em dois passos:

(1) Etapa de Atribuição: assinalar cada elemento ao cluster com centroide mais próximo.

(2) Etapa de Atualização: Recalcular novos centroides, a partir da última partição gerada.

Apesar de ser um método muito utilizado, as demonstrações das suas propriedades fundamentais são raramente apresentadas. Dessa forma, dedicamos essa subseção para realizar tais demonstrações. No Algoritmo 2, apresentamos o pseudo-código do Algoritmo k-means.

Algoritmo 2: k-means

Entrada: Conjunto de dados X, inteiro positivo k e partição inicial $\mathcal{C}^{(0)}$.

Saída: Partição $\mathcal{C} = \{C_1, \ldots, C_k\}$.

Define $t = 0$ e $s = 0$.
enquanto $t = 0$ **faça**
  $C_\ell^{(s+1)} = \{x : ||x - \mu_\ell^{(s)}|| < ||x_i - \mu_\theta^{(s)}||, \forall \theta \in [k]\}$,
  $\mathcal{C}^{(s+1)} = \{C_1^{(s+1)}, \ldots, C_k^{(s+1)}\}$
  para $x \in X$ **faça**
    se *x não está em nenhum cluster de* $\mathcal{C}^{(s+1)}$ **então**
      Calcule $C_\alpha^{(s+1)} = C_\alpha^{(s+1)} \cup \{x\}$, onde
      $\alpha = \min\{\ell : ||x_i - \mu_\ell^{(s)}|| \leq ||x_i - \mu_\theta^{(s)}||, \forall \theta \in [k]\}$.
    fim
  fim
  Calcule $\mu_\ell^{(s+1)} = \frac{1}{|C_\ell^{(s+1)}|} \sum_{x \in C_\ell^{(s+1)}} x$, $\forall \ell \in [k]$.
  $s = s + 1$
  se $\mathcal{C}^{(s)} = \mathcal{C}^{(s+1)}$ **então**
    $t = 1$
  fim
fim
Retorne $\mathcal{C}^{(s)}$

O Algoritmo 2 possui três ingredientes como entrada, um conjunto de dados, o número de clusters e uma partição inicial. A escolha do número de clusters k é uma dificuldade típica para os problemas de clusterização. Dessa forma, esse é um problema de pesquisa comum na área de classificação de dados, veja [73, 81, 21, 75]. A definição de uma partição inicial apropriada também é um problema amplamente estudado. Dois métodos simples e amplamente difundidos têm natureza aleatória.

(1) Defina a partição $\mathcal{C} = \{C_1, \ldots, C_k\}$, onde cada $x \in X$ é atribuído a um cluster C_ℓ aleatoriamente, com probabilidade uniforme. O procedimento é repetido caso um dos clusters fique vazio.

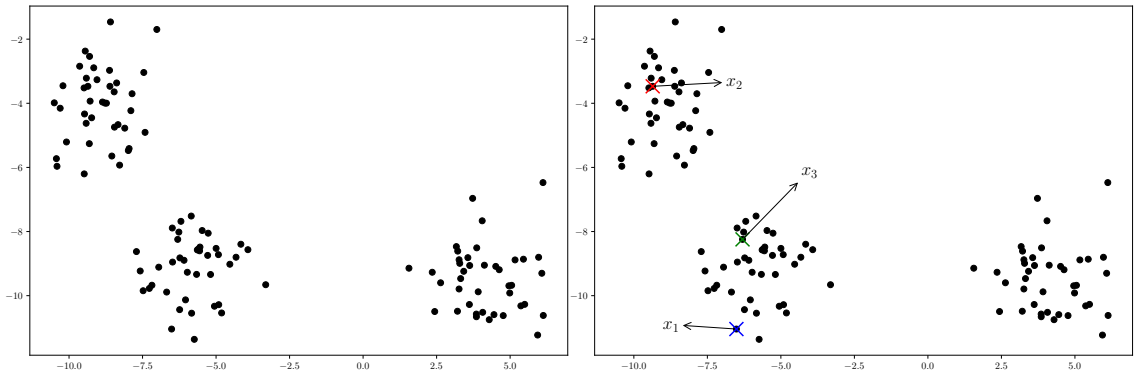

Figura 2.9: Na esquerda, conjunto de dados X'. Na direita, pontos iniciais escolhidos aleatoriamente x_1, x_2, x_3, azul, vermelho e verde, respectivamente.

(2) Escolha k pontos $x_1, \ldots, x_k \in X$ aleatórios, com probabilidade uniforme. Defina $\mathcal{C} = \{C_1, \ldots, C_k\}$ a partir do passo de atribuição do Algoritmo k-means, o passo (3) do Algoritmo 2.

No exemplo 2.6, iremos apresentar uma aplicação prática do Algoritmo k-means, apontando os detalhes em cada passo.

Exemplo 2.6. Consideramos o conjunto de dados X' definido na Figura 2.9 (esquerda). Será escolhido $k = 3$, visto que X' é construído juntando três conjuntos de dados distintos (as três nuvens presentes no plano). Ainda é necessária uma partição inicial para a entrada do algoritmo. Para definir a partição inicial iremos utilizar a inicialização aleatória (2) descrita acima. Dessa forma, escolhemos três pontos x_1, x_2, x_3 de X' aleatoriamente, com probabilidade uniforme e colocamos cada ponto em um cluster distinto, conforme a Figura 2.9 (direita).

Agora, definimos $\mathcal{C}^{(0)}$ a partir da regra: cada $x \in X'$ estará no mesmo cluster do ponto x_ℓ, $\ell \in \{1, 2, 3\}$, que está mais próximo de x, assim como no passo de construir a partição do Algoritmo k-means. Caso houvesse um ponto com mesma distância a dois elementos x_i, x_j, $i \neq j \in \{x_1, x_2, x_3\}$, escolheríamos o cluster que tivesse o menor valor no índice, porém nesse conjunto de dados não há um ponto a mesma distância dos x_i. A partição $\mathcal{C}^{(0)}$ está retratada na Figura 2.10 (esquerda). À

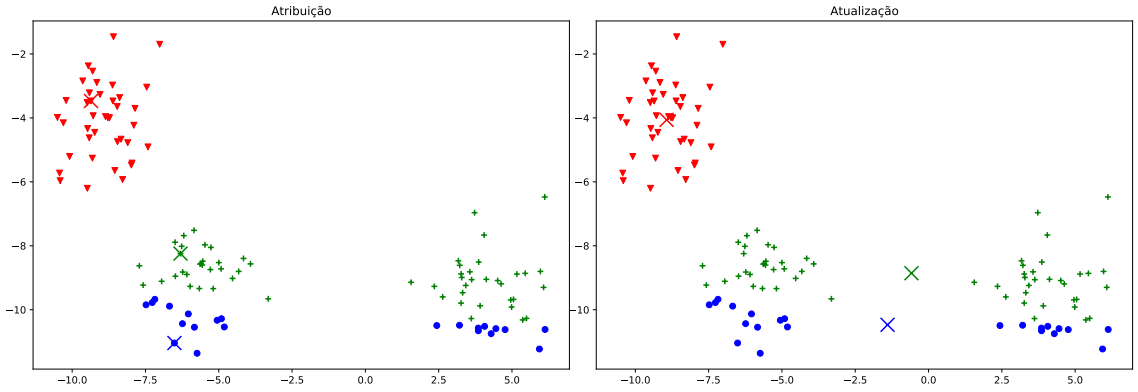

Figura 2.10: Na esquerda, partição $\mathcal{C}^{(0)}$ construída a partir dos pontos x_1, x_2 e x_3. Na direita, partição $\mathcal{C}^{(0)}$ e seus centroides $\mu_1^{(0)}, \mu_2^{(0)}, \mu_3^{(0)}$.

direita, na mesma figura, os centroides são calculados, por meio do ponto médio de cada novo cluster, sendo denotados por $\mu_1^{(0)}, \mu_2^{(0)}, \mu_3^{(0)}$.

Em suma, as entradas são $X = X'$, $k = 3$ e $\mathcal{C}^{(0)}$. Agora iremos executar a recursão do Algoritmo k-means. Iniciando com $t = 0$, aplicamos a regra (1) do Algoritmo 2, definindo a partição $\mathcal{C}^{(1)}$ (Figura 2.11, esquerda). Então, aplicamos a regra (2) do algoritmo para calcular os centroides $\mu_1^{(1)}, \mu_2^{(1)}, \mu_3^{(1)}$ da nova partição (Figura 2.11, direita).

Como $\mathcal{C}^{(0)} \neq \mathcal{C}^{(1)}$, t se mantém igual a 0, assim repetimos a recursão para obter $\mathcal{C}^{(2)}$ e $\mu_1^{(2)}, \mu_2^{(2)}, \mu_3^{(2)}$, ambos ilustrados na Figura 2.12.

Como $\mathcal{C}^{(1)} \neq \mathcal{C}^{(2)}$ repetimos a recursão. Ao calcular a nova partição, percebemos que todos os pontos já estão no mesmo cluster de seu centroide mais próximo, ou seja, a partição $\mathcal{C}^{(3)}$ será igual a $\mathcal{C}^{(2)}$. Isso ativa o critério de parada $t = 1$ e encerra o algoritmo. A saída é a partição $\mathcal{C}^{(2)}$, definida na Figura 2.13.

Percebe-se, no exemplo particular acima, que a partição final salta aos olhos do usuário antes de qualquer resultado, mas isso não é comum. Mesmo se tratando de dados em $\mathbb{R}^2$, cada coordenada de cada elemento do conjunto de dados representa uma observação de um objeto. Quando esses dados estão em um contexto

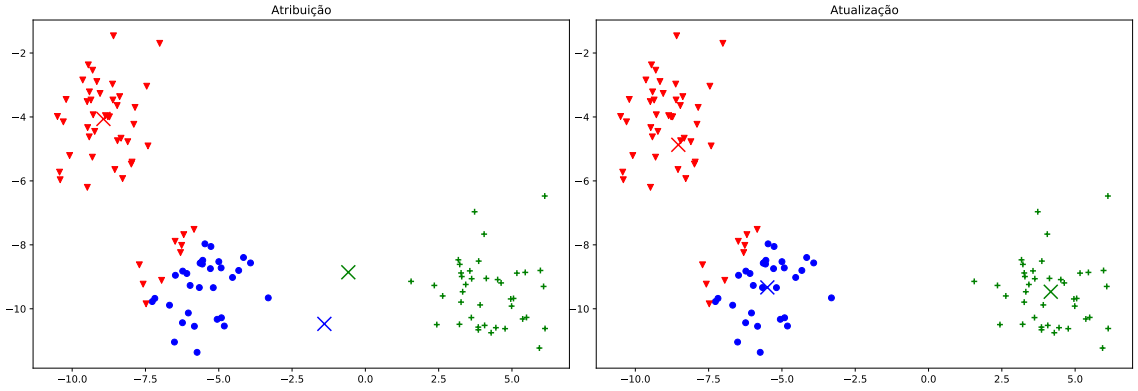

Figura 2.11: Na esquerda, partição $\mathcal{C}^{(1)}$ construída a partir dos centroides $\mu_1^{(0)}, \mu_2^{(0)}, \mu_3^{(0)}$, também presentes na figura (esquerda). Na direita, partição $\mathcal{C}^{(1)}$ e seus centroides $\mu_1^{(1)}, \mu_2^{(1)}, \mu_3^{(1)}$.

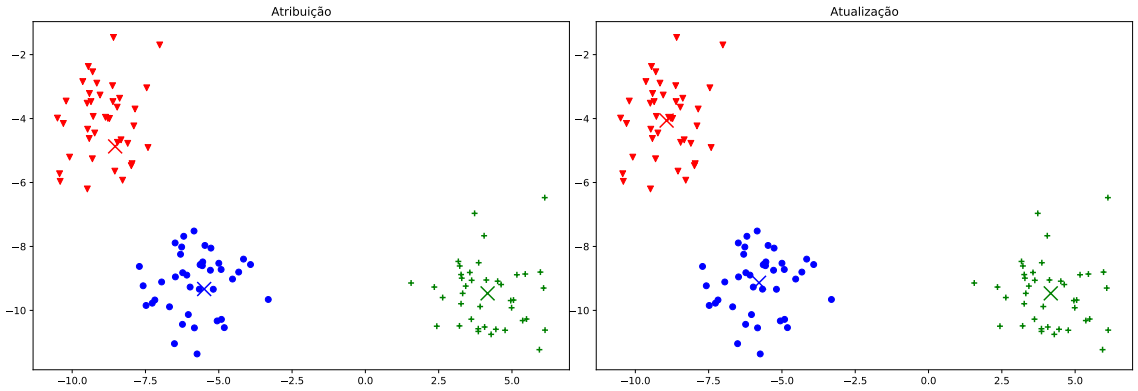

Figura 2.12: Na esquerda, partição $\mathcal{C}^{(2)}$ construída a partir dos centroides $\mu_1^{(1)}, \mu_2^{(1)}, \mu_3^{(1)}$. Na direita, partição $\mathcal{C}^{(2)}$ e seus centroides $\mu_1^{(2)}, \mu_2^{(2)}, \mu_3^{(2)}$.

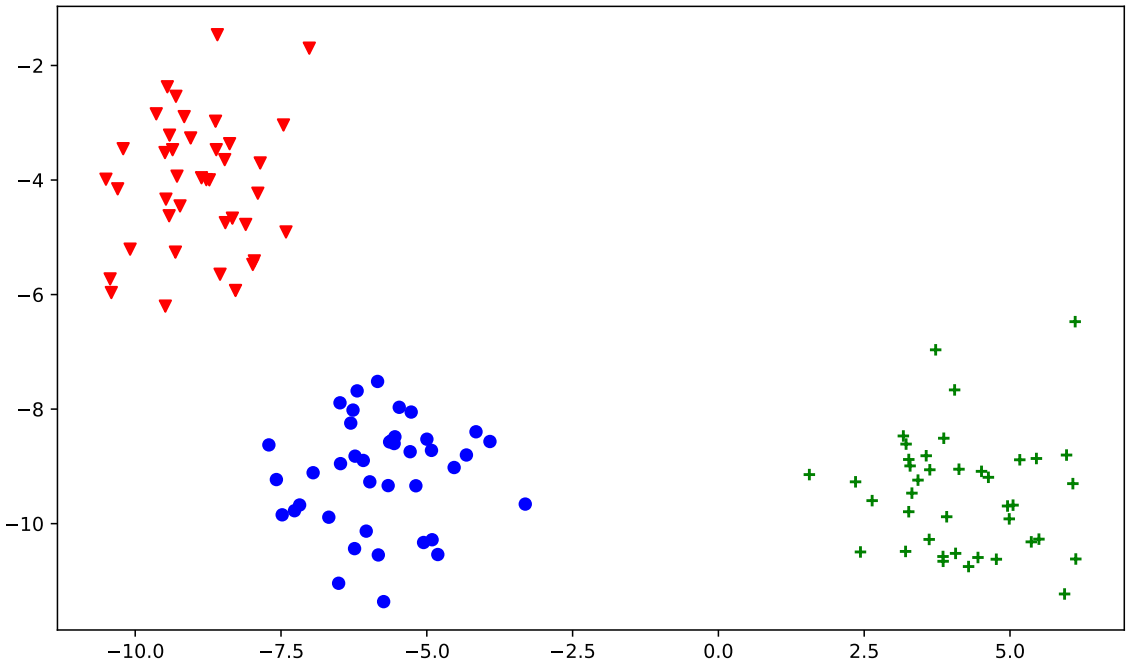

Figura 2.13: Partição $\mathcal{C}^{(2)}$, saída do Algoritmo k-means para o conjunto de dados X', com as condições iniciais definidas.

real, padrões com geometria simples são muito incomuns, trazendo a importância de utilizar um algoritmo de clusterização.

Apesar de dois passos simples, que claramente motivam o nome do algoritmo ser k-médias - na tradução literal -, o Algoritmo 2 apresenta diversas particularidades, relacionadas às respostas dos seguintes questionamentos:

- O algoritmo termina?
- A partição de saída do Algoritmo 2 possui alguma relação com alguma função objetivo?
- Como a partição inicial influencia no resultado?
- Os clusters formados pelo algoritmo possuem alguma propriedade topológica?

- O algoritmo é eficiente?

Na sequência dessa subseção respondemos a cada um dos itens acima. Iniciaremos apresentando um resultado que mostra a relação entre o Algoritmo k-means e a função objetivo J_1, definida na equação (2.2). O resultado deixa claro que, cada vez que mudamos a partição pela iteração do Algoritmo k-means, o valor da função J_1 dessa nova partição é menor do que o da anterior.

Proposição 2.1. *Dados um conjunto de dados X, $k \in \mathbb{N}$ e $\mathcal{C}^{(0)}$ partição de X, seja $\mathcal{C}^{(1)}$ a partição de X obtida após uma iteração do Algoritmo 2 a partir da partição $\mathcal{C}^{(0)}$.*

$$\textit{Se } \mathcal{C}^{(0)} \neq \mathcal{C}^{(1)}, \textit{ então } J_1(C^{(0)}) > J_1(C^{(1)}).$$

Essa proposição fornece a resposta a duas das perguntas realizadas. Primeiro, o Algoritmo k-means possui uma relação com a função objetivo J_1, pois esse resultado mostra que o Algoritmo k-means encerra caso ele encontre uma partição ótima em relação a J_1. Apesar disso, não há garantia que o Algoritmo k-means atinja o valor mínimo global desta função.

Segundo, uma consequência direta da proposição é que o Algoritmo k-means termina em uma quantidade finita de passos. Afinal, caso o algoritmo não terminasse, teríamos uma sequência infinita de partições com valor de J_1 diferente, o que é impossível pelo fato de X ser finito. Vale mencionar que, em aplicações práticas, além da garantia de terminar, o Algoritmo k-means converge rapidamente para sua saída [2]. Antes de abordarmos os outros questionamentos feitos anteriormente, vamos demonstrar em detalhes a Proposição 2.1.

Demonstração. Seja X conjunto de dados, $k \in \mathbb{N}$ e $\mathcal{C}^{(0)} = \{C_1^{(0)}, \ldots, C_k^{(0)}\}$ partição qualquer de X. Definimos a partição $\mathcal{C}^{(1)} = \{C_1^{(1)}, \ldots, C_k^{(1)}\}$ obtida após uma única iteração do Algoritmo k-means. Sejam $U^{(0)} = (\mu_1^{(0)}, \ldots, \mu_k^{(0)})$ e $U^{(1)} = (\mu_1^{(1)}, \ldots, \mu_k^{(1)})$ os centroides de $\mathcal{C}^{(0)}$ e $\mathcal{C}^{(1)}$, respectivamente. Supomos que $U^{(0)} \neq U^{(1)}$.

Para mostrar que $J_1(C^{(0)}) > J_1(C^{(1)})$ utilizaremos duas afirmações sobre duas funções auxiliares.

Definimos a função auxiliar $J_{U^{(0)}} : \mathcal{U}_k \to \mathbb{R}$ tal que $J_{U^{(0)}}(\mathcal{C}) = \sum_{\ell=1}^{k} \sum_{x \in C_\ell} ||x - \mu_\ell^{(0)}||$.
Afirmação 1: $J_{U^{(0)}}(\mathcal{C}^{(1)}) \leq J_{U^{(0)}}(\mathcal{C}^{(0)})$.

Demonstração. Por definição, $J_{U^{(0)}}(\mathcal{C}^{(1)}) = \sum_{\ell=1}^{k} \sum_{x \in C_\ell^{(1)}} ||x - \mu_\ell^{(0)}||$ e $J_{U^{(0)}}(\mathcal{C}^{(0)}) = \sum_{\ell=1}^{k} \sum_{x \in C_\ell^{(0)}} ||x - \mu_\ell^{(0)}||$.

Todo $x \in X$ está em algum cluster $C_\ell^{(1)}$. Pela definição de $C_\ell^{(1)}$ vale que $||x - \mu_\ell^{(0)}|| \leq ||x - \mu_\theta^{(0)}||$, para todo $\theta \in [k]$. Em particular, $||x - \mu_\ell^{(0)}|| \leq ||x - \mu_{\ell'}^{(0)}||$, onde $x \in C_{\ell'}^{(0)}$. Assim, percebe-se que a contribuição de cada x na função $J_{U^{(0)}}(\mathcal{C}^{(1)})$ é menor ou igual à contribuição do x na função $J_{U^{(0)}}(\mathcal{C}^{(0)})$. Logo, $J_{U^{(0)}}(\mathcal{C}^{(1)}) \leq J_{U^{(0)}}(\mathcal{C}^{(0)})$. □

Definimos a função $J_{C_\ell^{(1)}} : \mathbb{R}^n \to \mathbb{R}$, tal que $J_{C_\ell^{(1)}}(u) = \sum_{x \in C_\ell^{(1)}} ||x - u||$.

Afirmação 2: $\min_{u \in \mathbb{R}^n} J_{C_\ell^{(1)}}(u) = J_{C_\ell^{(1)}}(\mu_\ell^{(1)})$.

Demonstração. Note que $J_{C_\ell^{(1)}}$ é uma soma de funções diferenciáveis e, portanto, é diferenciável em todo ponto $u \in \mathbb{R}^m$. Dessa forma, o único candidato a extremo global da função é o ponto que possui derivadas parciais iguais a zero. Dessa forma,

$$\begin{aligned} \frac{\partial}{\partial u_t} J_{C_\ell^{(1)}}(u) &= \frac{\partial}{\partial u_t} \sum_{x \in C_\ell^{(1)}} ||x - u||^2 = \sum_{x \in C_\ell^{(1)}} \frac{\partial}{\partial u_t} ||x - u||^2 \\ &= \sum_{x \in C_\ell^{(1)}} \frac{\partial}{\partial u_t} (x_t - u_t)^2 = \sum_{x \in C_\ell^{(1)}} 2(u_t - x_t) \end{aligned} \tag{2.6}$$

Como queremos $\frac{\partial}{\partial u_{\ell_t}} J_{C_\ell^{(1)}}(u) = 0$, temos que $\sum_{x \in C_\ell^{(1)}} (u_t - x_t) = 0$, então $\sum_{x \in C_\ell^{(1)}} x_t = \sum_{x \in C_\ell^{(1)}} u_t$. Logo, $\sum_{x \in C_\ell^{(1)}} x_t = |C_\ell^{(1)}| u_t$ e, finalmente, $u_t = \frac{1}{|C_\ell^{(1)}|} \sum_{x \in C_\ell^{(1)}} x_t$. Note que esse é exatamente o valor da t-ésima coordenada de $\mu_\ell^{(1)}$, o centroide de $C_\ell^{(1)}$.

Para confirmar que $\mu_\ell^{(1)}$ é ponto mínimo global, basta verificar que a matriz Hessiana (veja [12]) é positiva semi-definida. Com efeito, $\frac{\partial^2}{\partial u_t \partial u_j} J_{C_\ell^{(1)}}(\mu_\ell^{(1)}) = 0$ se $t \neq j$ e $\frac{\partial^2}{\partial u_t^2} J_{C_\ell^{(1)}}(\mu_\ell) = 2|C_\ell|$, se $t = j$. Assim todos os autovalores da matriz Hessiana são positivos. Logo, $u = \mu_\ell^{(1)}$ é mínimo global de $\sum_{x \in C_\ell^{(1)}} ||x - u||$. □

A seguinte sequência conclui a demonstração da Proposição 2.1. Pela equação (2.2) temos que

$$J_1(\mathcal{C}^{(1)}) = \sum_{\ell=1}^{k} \sum_{x \in C_\ell^{(1)}} ||x - \mu_\ell^{(1)}||,$$

e pela Afirmação 2

$$\sum_{\ell=1}^{k} \sum_{x \in C_\ell^{(1)}} ||x - \mu_\ell^{(1)}|| < \sum_{\ell=1}^{k} \sum_{x \in C_\ell^{(1)}} ||x - \mu_\ell^{(0)}|| = J_{U^{(0)}}(\mathcal{C}^{(1)}).$$

Note que a desigualdade acima é estrita pois pelo menos um dos centroides é diferente. Portanto, pela Afirmação 1

$$J_{U^{(0)}}(\mathcal{C}^{(1)}) \leq J_{U^{(0)}}(\mathcal{C}^{(0)}) = \sum_{\ell=1}^{k} \sum_{x \in C_\ell^{(0)}} ||x - \mu_\ell^{(0)}|| = J_1(\mathcal{C}^{(0)}).$$

E, dessa maneira, podemos concluir que

$$J_1(\mathcal{C}^{(1)}) < J_1(\mathcal{C}^{(0)}).$$

□

Como destacado no pseudo-código, para a utilização do Algoritmo k-means é necessário definir uma partição inicial para ser utilizada na entrada do algoritmo. Testes práticos mostram que a saída do Algoritmo 2 é muito sensível à definição dessa partição inicial. Iniciamos mostrando isso em um exemplo teórico e, depois, em um prático.

Exemplo 2.7 (Exemplo Teórico)**.** Para $c > 1$, definimos o conjunto de dados $X = \{x_1, x_2, x_3, x_4\}$, onde $x_1 = (-1, -c)$, $x_2 = (-1, c)$, $x_3 = (1, c)$ e $x_4 = (1, -c)$. Esse conjunto está retratado na Figura 2.14.

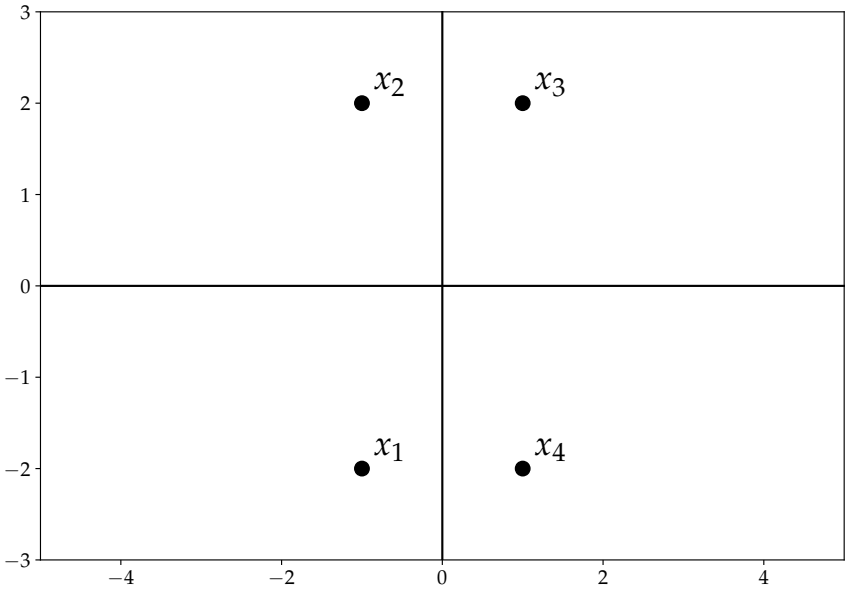

Figura 2.14: Conjunto de dados X representado no plano, para $c = 2$.

Vamos dividir esse conjunto em dois clusters pelo Algoritmo k-means. Vamos propor duas inicializações diferentes.

(1) $\mathcal{C} = \{C_1, C_2\}$, onde $C_1 = \{x_1, x_2\}$ e $C_2 = \{x_3, x_4\}$. Calculando os centroides de cada temos $\mu_1 = (-1, 0)$ e $\mu_2 = (1, 0)$. Pela Figura 2.15 é fácil ver que cada ponto já está no mesmo cluster do centroide mais próximo.

Então $\mathcal{C}$ já é a própria saída do Algoritmo k-means. Agora precisamos avaliar, o quão próxima essa partição está da ótima. Então vamos calcular $J_1(\mathcal{C})$.

$$J_1(\mathcal{C}) = ||x_1 - \mu_1||^2 + ||x_2 - \mu_1||^2 + ||x_3 - \mu_2||^2 + ||x_4 - \mu_2||^2 = 4c^2 \tag{2.7}$$

Como $c > 1$, fazendo $c \to \infty$ então $J_1(\mathcal{C}) \to \infty$. Agora vamos iniciar com outra partição e comparar os resultados.

(2) $\mathcal{C}' = \{C_1, C_2\}$, onde $C_1' = \{x_1, x_4\}$ e $C_2' = \{x_2, x_3\}$. Calculando os centroides de cada temos $\mu_1' = (0, -c)$ e $\mu_2' = (0, c)$. Pela Figura 2.16 é fácil ver que cada ponto já está no mesmo cluster do centroide mais próximo.

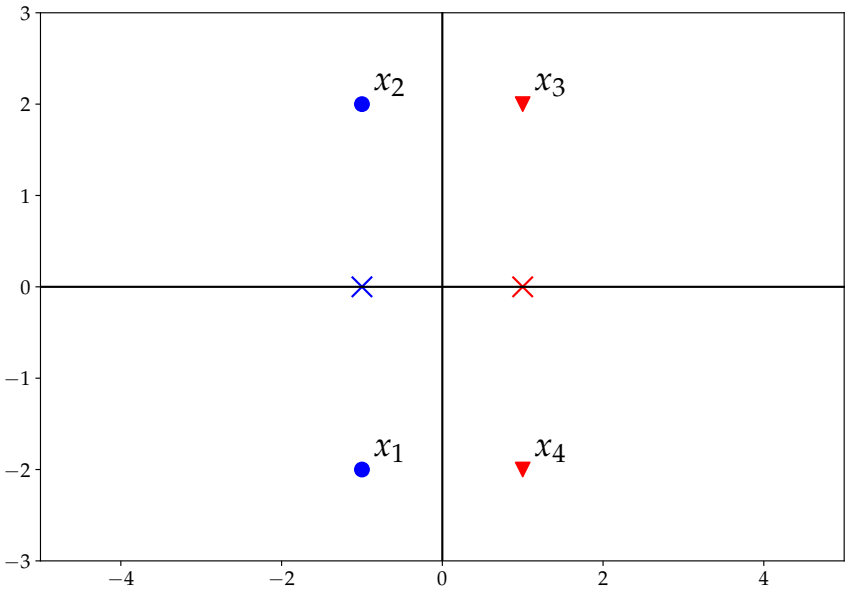

Figura 2.15: Partição $\mathcal{C} = \{C_1, C_2\}$, onde $C_1 = \{x_1, x_2\}$ e $C_2 = \{x_3, x_4\}$ de X.

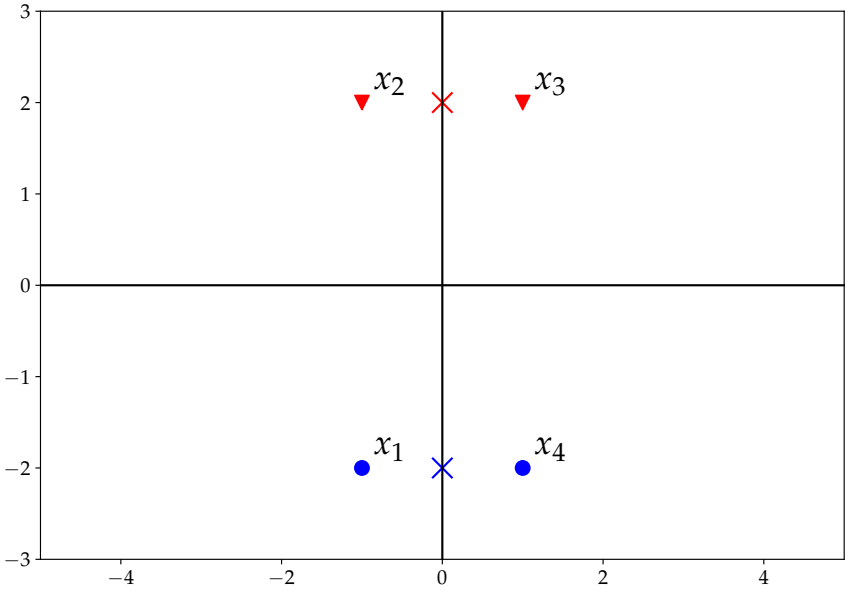

Figura 2.16: Partição $\mathcal{C}' = \{C_1, C_2\}$, onde $C_1' = \{x_1, x_4\}$ e $C_2' = \{x_2, x_3\}$ de X.

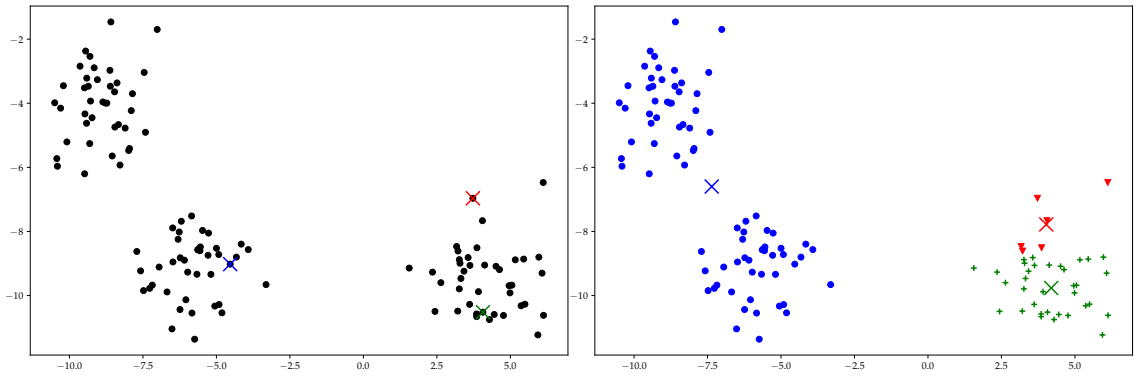

Figura 2.17: À esquerda, centroides que induzirão a partição inicial utilizada na entrada do Algoritmo k-means. À direita, partição de entrada utilizada no Algoritmo k-means.

Então, novamente, $\mathcal{C}'$ já é a própria saída do Algoritmo k-means. Além disso,

$$J_1(\mathcal{C}') = ||x_1 - \mu'_1||^2 + ||x_4 - \mu'_1||^2 + ||x_2 - \mu'_2||^2 + ||x_3 - \mu'_2||^2 = 4. \tag{2.8}$$

A partir dos casos descritos acima, fica claro que $\mathcal{C}'$ é a partição ótima nesse caso e que, dependendo da escolha da partição inicial, o valor da função J_1 pode tender a infinito, à medida que c cresce, ou pode ficar constante, independente de quão grande seja c.

Exemplo 2.8 (Exemplo Prático)**.** Retomando o exemplo em que aplicamos o Algoritmo k-means no conjunto de dados X', retratado na Figura 2.9. Nesse caso, inicializamos com uma partição que nos levou à partição que buscávamos no final. Mas isso nem sempre ocorre. A Figura 2.17 mostra outra inicialização que será utilizada na entrada do Algoritmo 2. Ao finalizarmos o Algoritmo k-means obtemos a partição apresentada na Figura 2.18 como saída. Claramente diferente da partição final que gostaríamos.

Acabamos de mostrar dois exemplos para para os quais obtivemos partições de saída diferentes, dependendo da partição inicial utilizada na entrada do Algoritmo k-means. Em ambos os exemplos, dependendo da partição inicial, foi

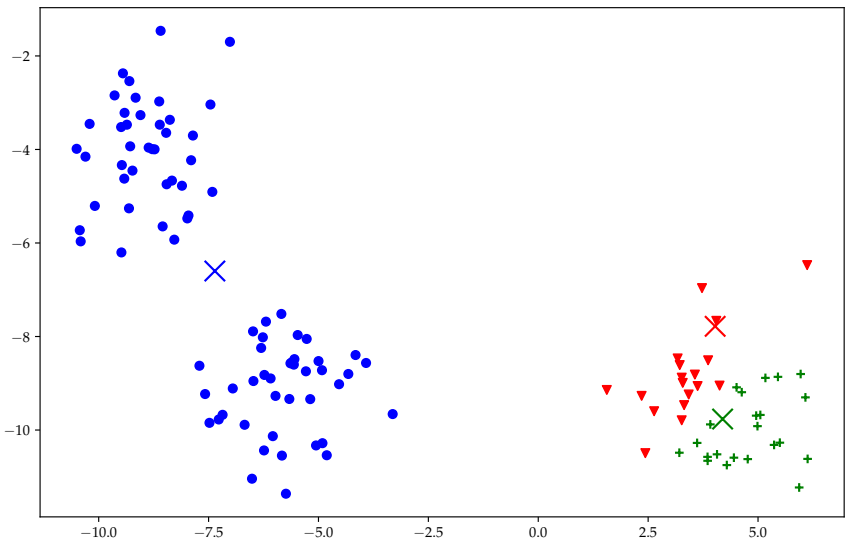

Figura 2.18: Partição final, utilizando a partição retratada na Figura 2.17 como entrada.

possível encontrar a partição ótima. Uma maneira de evitar inicializações ruins é executar múltiplas vezes o Algoritmo k-means para uma inicialização aleatória e escolher a partição que obtenha menor valor para a função objetivo J_1 (2.2).

Na literatura há algumas maneiras de definir a partição inicial para o Algoritmo k-means além das duas que já apresentamos no início dessa seção, veja [14]. Apresentamos uma delas, o Algoritmo k-means++ [3] que define a partição inicial de uma maneira particular.

Algoritmo 3: k-means++

Entrada: Conjunto de dados X e inteiro positivo k.

Saída: Partição $\mathcal{C} = \{C_1, \ldots, C_k\}$.

Define $t = 0$ e $s = 0$.

Escolhe aleatoriamente $x \in X$ e defina $\mu_1 = x$. Seja $U = \{\mu_1\}$.

para $\ell = 2$ *até* k **faça**

  Para cada $x \in X$, calcule $d(x, U) = \min\{||x - \mu|| : \mu \in U\}$.

  Escolhe $x \in X$ aleatoriamente, onde a probabilidade de cada x ser selecionado é

$$p(x) = \frac{d(x, U)^2}{\sum_{x' \in X} d(x', U)^2}.$$

  Define $\mu_\ell = x$.

  Calcule $U = U \cup \mu_\ell$.

fim

Define a partição $\mathcal{C}^{(0)}$ que atribui cada ponto ao cluster do centroide mais próximo.

Execute o Algoritmo k-means para X, k e $\mathcal{C}^{(0)}$.

Retorne *Partição* $\mathcal{C}$

O Algoritmo 3 escolhe os k pontos iniciais iterativamente. Quando ℓ pontos já foram escolhidos, o $(\ell + 1)$-ésimo ponto tende a ser escolhido distante de todos os primeiros. Além disso, esse método é mais rápido que o Algoritmo k-means utilizando como entrada a partição inicial construída a partir de um método de natureza aleatória com probabilidade uniforme [3].

Em relação ao questionamento sobre topologia, vamos mostrar que o Algoritmo k-means só produz clusters convexos.

Proposição 2.2. *Dados um conjunto de dados $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^m$ e k inteiro positivo, seja $\mathcal{C} = \{C_1, \ldots, C_k\}$ a saída do Algoritmo 2, tendo como entradas X, k e uma partição $\mathcal{C}^{(0)}$ de X. Então C_ℓ é convexo para todo $\ell \in [k]$, isto é, se $u, v \in C_\ell$ e $y = tu + (1 - t)v$ para algum $t \in [0, 1]$, então então $y \in C_\ell$.*

Demonstração. Seja $\mathcal{C} = \{C_1, \ldots, C_k\}$ a saída do Algoritmo 2 para X, iniciando com uma partição qualquer. Fixamos $\ell \in [k]$. Sejam $u, v \in C_\ell$.

Dado $y \in r = \{tu + (1-t)v : t \in (0,1)\}$, seja $\theta \neq \ell \in [k]$. Vamos supor por absurdo que $y \in C_\theta$ para algum $\theta \neq \ell$. Considere o hiperplano $H = \{x \in \mathbb{R}^m : ||x - \mu_\ell|| = ||x - \mu_\theta||\}$. Sabemos que existem apenas três possibilidades para a interseção entre um segmento r e o hiperplano H:

(1) $H \cap r = r$, o próprio segmento r;

(2) $H \cap r = \{z\}$, apenas um ponto do espaço;

(3) $H \cap r = \emptyset$, vazia.

Se vale (1), então para todo $x \in r$ vale que $||x - \mu_\ell|| = ||x - \mu_\theta||$. Em particular $||u - \mu_\ell|| = ||u - \mu_\theta||$. Como $u \in C_\ell$, vale que $\ell < \theta$. Pelo Algoritmo 2, isso atribui y ao cluster C_ℓ, o que é um absurdo. Se vale (2), então $H \cap r$ é um dos extremos de r, que são u e v. Logo, $r \setminus v \subset \{x \in \mathbb{R}^m : ||x - \mu_\ell|| < ||x - \mu_\theta||\}$. Isso é um absurdo pois $y \in r \setminus v$. Se vale (3), então todo $x \in r$ é tal que $||x - \mu_\ell|| > ||x - \mu_\theta||$, pois isso vale para y e se existisse x, onde $||x - \mu_\ell|| < ||x - \mu_\theta||$, r precisaria cruzar H. Isso é um absurdo pois $u \in C_\ell$.

Logo, os três casos são impossíveis. Portanto, $y \in C_\ell$.

□

Retomando o último questionamento desta subseção, o tempo de execução do Algoritmo k-means é muito estudado (veja [29]), principalmente pela incerteza de quantas iterações serão necessárias para finalizar o algoritmo. Dado o conjunto de dados $X \subset \mathbb{R}^m$, fixando k, m inteiros positivos, o algoritmo termina em tempo $O(nkmT)$, onde $n = |X|$ e T é o número de iterações realizadas. Em teoria T pode depender de n, mas em muitas aplicações práticas o tempo computacional é

Figura 2.19: Torre Eiffel. Disponível em: `https://turismo.eurodicas.com.br/torre-eiffel-paris/`. Acesso em: 31 jan. 2022.

linear no tamanho do conjunto de dados, veja [2] para uma discussão mais completa sobre a complexidade do algoritmo e o número de iterações necessárias.

A seguir apresentamos dois novos exemplos de aplicação do Algoritmo k-means.

Exemplo 2.9. É comum utilizar os algoritmos de clusterização para segmentar imagens. Nessa caso, cada pixel da imagem é um vetor, onde as entradas desse vetor representam a intensidade de determinadas cores. Vamos utilizar o Algoritmo 3 para seccionar a imagem da torre Eiffel retratada na Figura 2.19.

Essa é uma imagem 450×750, ou seja, possui um total de $n = 337.500$ pixels. Cada pixel está no sistema RGB, isto é, para $i \in \{1, \ldots, n\}$ $\mathcal{O}(x_i) = (x_1^{(i)}, x_2^{(i)}, x_3^{(i)})$, onde $x_1^{(i)} \in \{0, \ldots, 255\}$ codifica a intensidade de vermelho, sendo 0 desligado e 255 o mais intenso possível. $x_2^{(i)}, x_3^{(i)}$ codificam as cores verde e azul, respectivamente. Colocando como entrada $X = \{x_1, \ldots, x_n\}$ e $k \in \{2, 3, 4, 8\}$ no Algoritmo 3 obtemos os resultados apresentados na Figura 2.20.

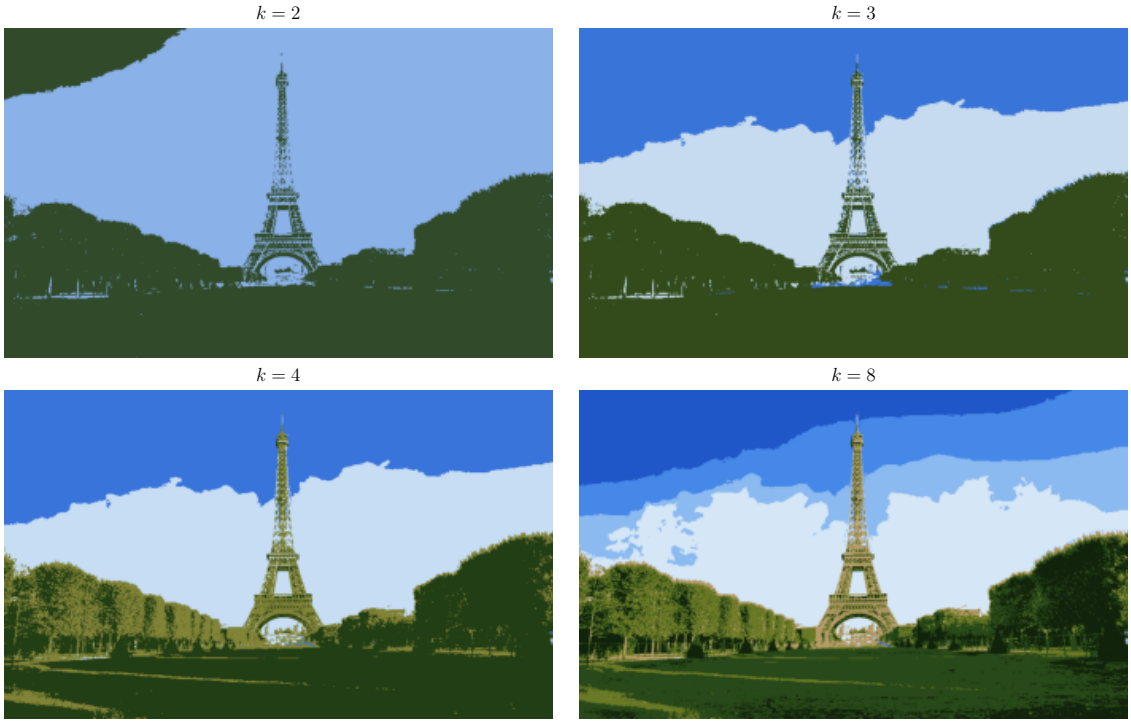

Figura 2.20: Resultados do Algoritmo k-means++ no conjunto de dados X para $k \in \{2, 3, 4, 8\}$. A cor escolhida para o cluster é a cor obtida pelo centroide do cluster.

Visualmente o resultado do Algoritmo k-means++ é bastante satisfatório na Figura 2.20. Note que, podemos perceber que aumentar o valor de k é o mesmo que aumentar o número de cores na imagem. A cor atribuída a cada pixel da imagem é dada pela combinação RGB das coordenadas do centroide do cluster a que esse ponto pertence.

Exemplo 2.10. Definimos X um conjunto de dados baseado na união dos pontos de duas circunferências, uma de raio 0.5 e outra de raio 1. Se formos clusterizar esse conjunto, pela sua construção, gostaríamos que cada circunferência ficasse em um único cluster. Pela limitação de produzir apenas clusters convexos, veja a Proposição 2.2, o Algoritmo k-means não consegue atingir tal partição. Uma saída do Algoritmo k-means para esse conjunto está retratada na Figura 2.21 (esquerda). Enquanto isso, métodos espectrais são conhecidos por lidar bem em situações mais complexas [78], como o exemplo em questão. Dessa forma, o resultado de um método espectral está retratado na Figura 2.21 (direita). Note que esse método consegue dividir o conjunto de dados nas duas circunferências que foram dadas.

No próximo capítulo apresentaremos a estrutura geral dos métodos de clusterização espectral e a fundamentação teórica por trás dessas técnicas.

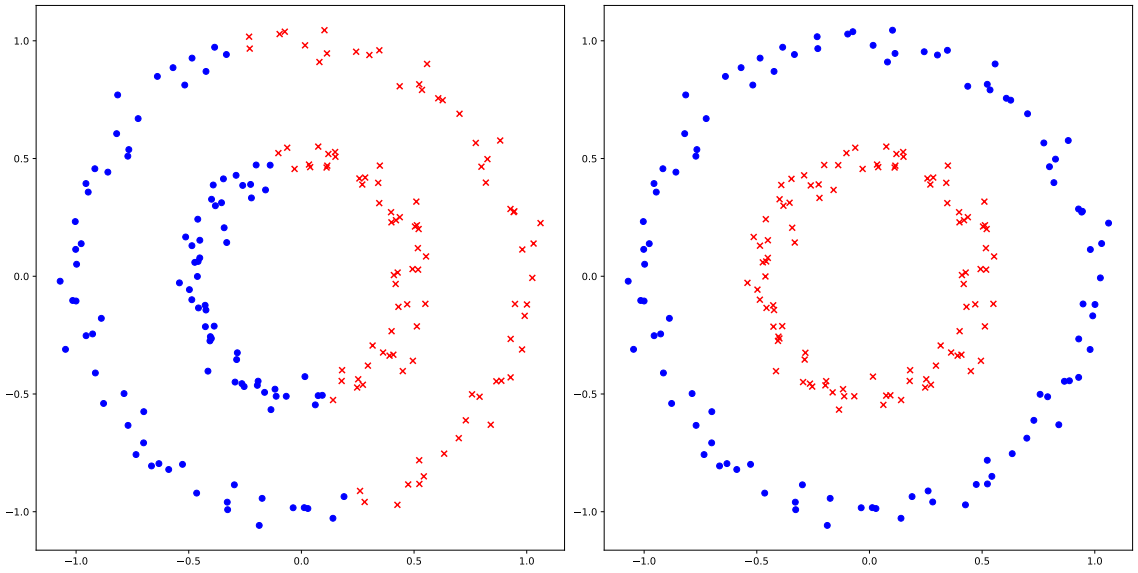

Figura 2.21: Na esquerda, o resultado de uma aplicação do Algoritmo k-means no conjunto de dados com pontos distribuídos em duas circunferências. Na direita, resultado de um método de clusterização espectral no conjunto de dados com pontos distribuídos em duas circunferências.

3 A CLUSTERIZAÇÃO ESPECTRAL

Nesse capítulo, apresentamos uma nova família de algoritmos de clusterização, os algoritmos de clusterização espectral. A fundamentação desses algoritmos está baseada na álgebra linear. Esses algoritmos possuem uma matriz de similaridade como entrada, tipicamente utilizada para mapear um conjunto de dados em um novo espaço euclidiano, realçando similaridades entre objetos - que vão além da distância entre dois elementos. Os métodos espectrais são conhecidos por lidar com situações mais complexas do que, por exemplo, as heurísticas já apresentadas na Seção 2.2.

Esse capítulo está estruturado da seguinte maneira. A Seção 3.1 apresenta um dos algoritmos espectrais mais famosos da literatura. Na Seção 3.2 embasamos o método apresentado na Seção 3.1 em teoria de grafos e álgebra linear. Por fim, na Seção 3.3 apresentamos uma estrutura geral dos métodos espectrais e algumas aplicações.

3.1 Algoritmo de clusterização espectral de Ng, Jordan e Weiss

Essa seção tem por objetivo apresentar o algoritmo espectral apresentado por Ng *et al.* [49]. A fundamentação teórica desse algoritmo será discutida na seção seguinte. Além disso iremos definir a noção de medida de similaridade, que é uma entrada dos métodos espectrais.

No Algoritmo k-means (e suas variantes), utilizamos os vetores que contêm as observações como entrada. Nos métodos espectrais utilizamos uma medida de similaridade como entrada, que definimos agora.

Dado um conjunto de dados X, definimos uma medida de similaridade como uma função $s : X \times X \to \mathbb{R}_{\geq 0}$, onde pares de objetos x_i, x_j com mais afinidade no conjunto X possuem maior medida de similaridade $s(x_i, x_j)$, enquanto que pares com menos afinidade possuem $s(x_i, x_j)$ menor. Além disso, deve valer que $s(x_i, x_j) = s(x_j, x_i)$. A medida de similaridade utilizada para um conjunto de dados depende do contexto do problema de classificação de dados. No Capítulo 4 e no Capítulo 5 discutiremos esses aspectos com mais profundidade. Definimos como matriz de similaridade a matriz construída diretamente da medida de similaridade, isto é, $S = (s_{ij})$ é tal que $s_{ij} = s(x_i, x_j)$. Também podemos definir uma medida de dissimilaridade d, que é definida da mesma maneira que uma medida de similaridade, mas com interpretação oposta: pares de pontos onde d é maior possuem menos similaridade, enquanto que pares de pontos onde d é menor possuem mais similaridade.

Exemplo 3.1. Dado um conjunto de dados $X \subset \mathbb{R}^m$, $d(x_i, x_j) = ||x_i - x_j||$ é uma medida de dissimilaridade, pois pontos mais distantes (menos similares) têm d maior. Enquanto que $s(x_i, x_j) = \frac{1}{d(x_i, x_j)}$, se $i \neq j$, é uma medida de similaridade, pois quanto mais próximos dois pontos estão, maior é a similaridade entre eles.

No algoritmo espectral que iremos definir abaixo, será necessário calcular os autovalores e autovetores de uma matriz obtida a partir da matriz de similaridade S. Esses autovalores precisam ser reais pois será necessário sequencia-los. A matriz de similaridade S é simétrica, por consequência essa matriz particular obtida por S também é simétrica. Um resultado muito tradicional sobre matrizes simétricas é que todos os seus autovalores são números reais [31]. Isso permite que o sequenciamento dos autovalores possa ser devidamente feito.

Apresentamos a seguir um dos algoritmos espectrais mais famosos, definido por Ng, Jordan e Weiss [49].

Algoritmo 4: Cluserização Espectral NJW [49]

Entrada: Conjunto de dados X, inteiro positivo k e $\sigma > 0$.

Saída: Partição $\mathcal{C} = \{C_1, \ldots, C_k\}$.

1 Construa a matriz de similaridade $S = (s_{ij})$ de ordem n, onde $s_{ij} = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right)$, se $i \neq j$ e $s_{ij} = 0$, caso contrário.

2 Calcule $\mathcal{L} = D^{-1/2}(D - S)D^{-1/2}$, onde $D = (d_{ij})$ é uma matriz diagonal cuja i-ésima entrada é dada por $d_{ii} = \sum_{j=1}^{n} s_{ij}$.

3 Encontre os autovetores ortonormais $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k \in \mathbb{R}^n$, associados aos k menores autovalores $\lambda_1 \leq \cdots \leq \lambda_k$ de $\mathcal{L}$, respectivamente. Considere a matriz $\mathbf{X} = [\mathbf{x}_1 \mathbf{x}_2 \cdots \mathbf{x}_k] \in \mathbb{R}^{n \times k}$, cujas colunas são dadas pelos vetores $\mathbf{x}_i$.

4 Defina a matriz $\mathbf{Y} = (\mathbf{Y}_{ij})$ a partir de $\mathbf{X}$, onde $\mathbf{Y}_{ij} = \mathbf{X}_{ij} / \sqrt{\sum_{j'=1}^{n} \mathbf{X}_{ij'}^2}$.

5 Tratando cada linha de $\mathbf{Y}$ como um vetor de $\mathbb{R}^k$, utilize o Algoritmo k-means, com inicialização aleatória, para obter uma partição desses vetores.

6 Atribua o objeto original $x_i \in X$ para o cluster ℓ se, e somente se, a i-ésima linha de $\mathbf{Y}$ foi atribuída para o cluster ℓ. Denote essa partição de X por $\mathcal{C}$.

7 **Retorne** $\mathcal{C}$

No Algoritmo 4, Ng *et al.* [49] utilizam uma medida de similaridade chamada de *kernel Gaussiano*. Nessa similaridade, à medida que x, y se aproximam, $s(x, y)$ fica próximo de 1 e, à medida que x, y ficam mais distantes, $s(x, y)$ se aproxima de 0.

Note que o passo (5) não é determinístico, uma vez que o Algoritmo k-means utiliza uma inicialização aleatória. Por isso, usualmente, em aplicações do Algoritmo 4, o passo (5) é realizado um número Q de vezes para evitar inicializações que levem a soluções ruins, como foi o caso das aplicações do Algoritmo 2 nas situações dos exemplos 2.7 e 2.8.

Chamamos a atenção de que, no passo (5) do Algoritmo 4, utilizamos o Algoritmo k-means para clusterizar os vetores da matriz Y. À primeira vista isso

pode parecer fazer pouco sentido, pois poderíamos utilizar o Algoritmo k-means diretamente na base de dados. Em relação a essa questão, nesse algoritmo espectral buscamos mapear nosso conjunto de dados em outro espaço, de dimensão k, que realce a similaridade entre os pontos. Para ilustrar isso retomamos o exemplo retratado na Figura 2.21.

Exemplo 3.2. Sabemos que o Algoritmo k-means não consegue encontrar clusters não convexos e, por isso, não consegue encontrar a partição retratada na Figura 3.1 (em cima) para o conjunto de dados X representado no plano da Figura 3.1 (acima). Vamos aplicar o Algoritmo 4 nesse conjunto de dados X. Utilizamos X, $\sigma = 0.1$ e $k = 2$ como entradas para o Algoritmo 4.

A partir de S, calculamos $\mathcal{L} = D^{-1/2}(D - S)D^{-1/2}$, onde $D = (d_{ij})$, tal que $d_{ii} = \sum_{j=1}^{n} s_{ij}$ e $d_{ij} = 0$, se $i \neq j$. Em seguida, consideramos a matriz $\mathbf{X} \in \mathbb{R}^{n \times k}$, colocando nas colunas de $\mathbf{X}$ os $k = 2$ autovetores de $\mathcal{L}$ associados aos 2 menores autovalores de $\mathcal{L}$. As linhas dessa matriz $\mathbf{X}$ podem representar o conjunto X, como ilustrado na Figura 3.1 (esquerda), onde o ponto $x_i \in X$ é representado pela i-ésima linha de $\mathbf{X}$. Note que as linhas da matriz $\mathbf{X}$ já realçam os clusters que buscamos. Isso é evidenciado quando calculamos Y a partir da normalização das linhas de $\mathbf{X}$, que coloca os pontos em lugares distantes na casca da esfera de raio unitário e dimensão 2. Nesse novo conjunto de dados, o Algoritmo k-means encontra perfeitamente a partição que buscávamos.

Em suma, mapeamos cada ponto x_i de X em um novo elemento $\tilde{x}_i$, isto é, $M : X \to \mathbb{R}^2$ tal que $M(x_i) = \tilde{x}_i$, onde $\mathbf{X}$ é a matriz definida no passo (2) do Algoritmo 4 e $\tilde{x}_i$ é a i-ésima linha de $\mathbf{X}$. As flechas da Figura 3.1 (de cima para esquerda inferior) ilustram que os pontos de uma mesma circunferência ficam acumulados em uma região. Além disso, evidenciamos o cluster de cada circunferência ao normalizar as linhas de $\mathbf{X}$.

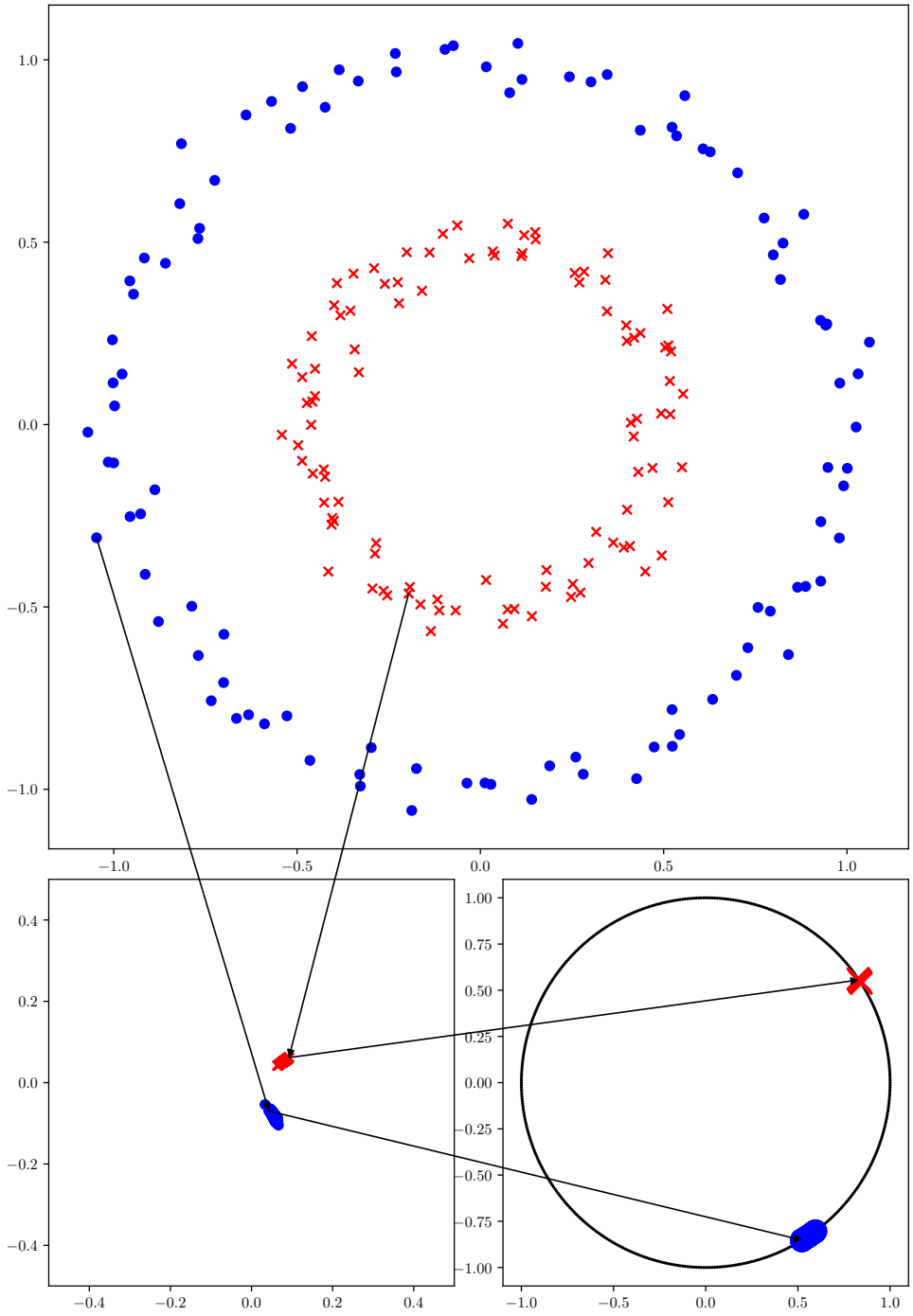

Figura 3.1: Em cima, um conjunto de dados X representado no plano. Na parte inferior esquerda, cada linha da matriz $\mathbf{X}$ é um ponto, onde a linha i representa o ponto $x_i \in X$. Na parte inferior direita, representação no plano da matriz Y obtida no passo (4) do Algoritmo 4, normalizando as linhas de $\mathbf{X}$. As cores/formas representam os clusters dos pontos no caso onde cada cluster é uma das circunferências.

Para o leitor mais familiarizado com teoria de grafos, há uma relação clara entre um conjunto de dados X, acompanhado de uma medida de similaridade, e um grafo ponderado.

Consideramos grafos ponderados $G = (V, E, \omega)$, onde $V = \{v_1, \ldots, v_n\}$ é o conjunto de vértices, E é o conjunto de arestas. Por fim, $\omega \colon E \to \mathbb{R}_{>0}$ atribui um peso positivo w_{ij} a cada aresta $ij \in E$.

Nesse sentido, podemos definir o grafo de similaridade de um conjunto de dados X, associado a uma medida de similaridade s, como o grafo ponderado $G = (V, E, \omega)$, onde V é o conjunto de dados X, $E = \{\{x_i, x_j\} : s(x_i, x_j) > 0\}$ e $w_{ij} = s(x_i, x_j)$.

Essa representação do conjunto de dados como um grafo ponderado é importante para que possamos entender a teoria por trás do Algoritmo 4, que apresentaremos a seguir.

3.2 A fundamentação teórica da clusterização espectral

Nessa seção consideraremos grafos ponderados, mas vale lembrar que qualquer conjunto de dados acompanhado de uma medida de similaridade pode ser transformado em um grafo ponderado. Então, sempre que considerarmos o conjunto de vértices V, este está associado a um conjunto de dados X e, quando considerarmos arestas e seus pesos, estes estão associados às similaridades entre os objetos do conjunto de dados.

A seguir apresentamos algumas noções importantes para essa seção e para o restante desse trabalho.

Novamente, estamos considerando apenas grafos ponderados $G = (V, E, \omega)$, onde $V = \{v_1, \ldots, v_n\}$ é o conjunto de vértices, E é o conjunto de arestas e

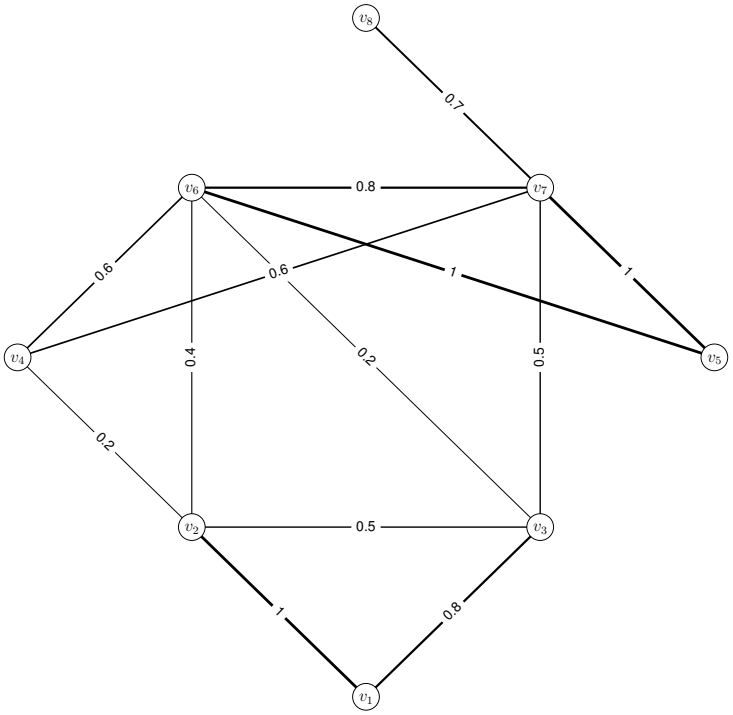

Figura 3.2: Exemplo de grafo ponderado.

$\omega\colon E \to \mathbb{R}_{>0}$ atribui um peso positivo w_{ij} a cada aresta $ij \in E$, em que dizemos que a aresta ij é incidente aos vértices i e j. Utilizaremos tanto grafo ponderado, quanto grafo, para nos referir aos grafos ponderados, nessa seção.

É comum representarmos os grafos no plano como pontos, que representam os vértices de V, com ligações entre esses pontos, que são definidas a partir das arestas em E. No plano, o número que acompanha a aresta é o peso atribuído a ela em G. Veja a Figura 3.2 para um exemplo de um grafo ponderado $G_1 = (V, E, \omega)$, com $n = 8$ vértices, representado no plano.

Um conceito importante é o grau de um vértice do grafo. Dado um grafo G de vértices $v_1, \ldots, v_n$, definimos o grau do i-ésimo vértice como $d_i = \sum_{j=1}^{n} w_{ij}$. No caso de G_1, $d_1 = 1.8$, $d_2 = 2.1$ e $d_8 = 0.7$. Se $d_i = 0$ dizemos que $v_i \in V$ é um vértice isolado.

Um passeio em um grafo G é uma sequência de vértices intercalada com uma sequência de arestas, onde cada aresta é precedida e sucedida por seus vértices incidentes. Um caminho entre $u, v \in V$ é um passeio que não contem vértices e arestas repetidas e seu primeiro e último elemento são u e v. Por exemplo, um caminho entre v_1 e v_8 em G_1 é a sequência $(v_1, v_2, v_6, v_7, v_8)$. Além disso, um grafo G é dito conexo se, para quaisquer $u, v \in V$ distintos, é possível obter um caminho entre u e v. Caso contrário G é desconexo.

A seguir serão definidas algumas matrizes associadas ao grafo, principalmente as matrizes associadas ao conjunto de dados utilizadas pelo Algoritmo 4. A partir de um grafo ponderado G é possível defini-se a matriz de graus de G, $D = (d_{ij})$ por $d_{ij} = d_i$, se $i = j$ e 0, caso contrário. A matriz de similaridade de G, $S = (s_{ij})$, é definida por $s_{ij} = w_{ij}$ se $i \neq j$ e 0, caso contrário. Para o grafo G_1 (Figura 3.2) a matriz de graus D e a matriz de similaridade S são as seguintes:

$$D = \begin{bmatrix} 1.8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2.1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3.6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.7 \end{bmatrix} \quad S = \begin{bmatrix} 0 & 1 & 0.8 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0.5 & 0.2 & 0 & 0.4 & 0 & 0 \\ 0.8 & 0.5 & 0 & 0 & 0 & 0.2 & 0.5 & 0 \\ 0 & 0.2 & 0 & 0 & 0 & 0.6 & 0.6 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0.4 & 0.2 & 0.6 & 1 & 0 & 0.8 & 0 \\ 0 & 0 & 0.5 & 0.6 & 1 & 0.8 & 0 & 0.7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.7 & 0 \end{bmatrix}$$

A partir dessas duas matrizes é possível definir a matriz Laplaciana normalizada de um grafo G. Assim, dado um grafo ponderado G, $\mathcal{L} = D^{-1/2}LD^{-1/2}$ é sua matriz Laplaciana normalizada, onde $L = D - S$ é a matriz Laplaciana de G. Perceba que todas essas matrizes possuem a propriedade de ser simétrica. Como esse trabalho está centrado no tema clusterização, estamos interessados em clusterizar o conjunto de vértices desse grafo. Nesse sentido, para dois subconjuntos disjuntos

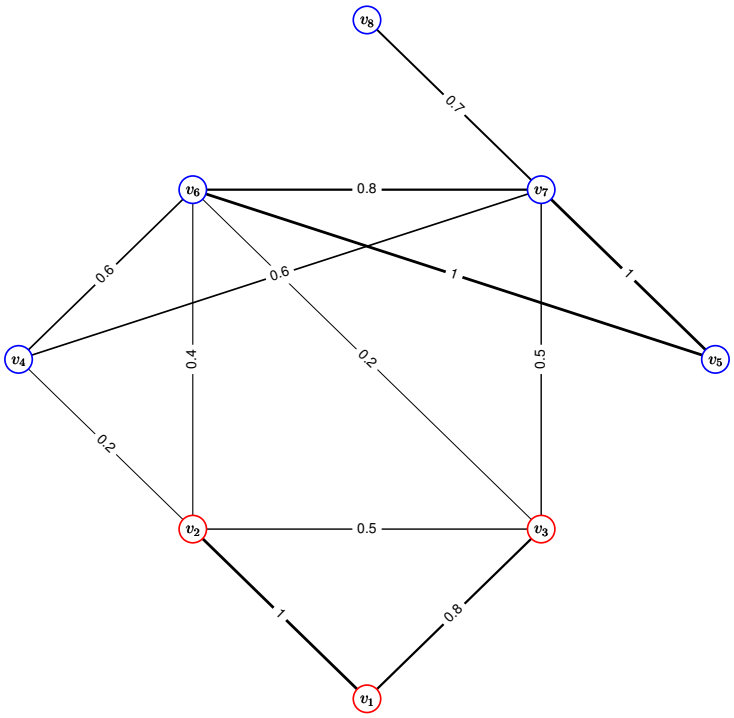

Figura 3.3: Exemplo de divisão de V em dois subconjuntos. A é o conjunto de vértices coloridos em vermelho, B em azul.

(clusters) $A, B \subset V$, é possível definir o peso das arestas entre A e B como

$$W(A, B) = \sum_{v_i \in A, v_j \in B} w_{ij}. \tag{3.1}$$

Também é possível definir o tamanho desses subconjuntos a partir do número de vértices ou pelo peso das arestas que incidem nos vértices contidos no subconjunto. Assim, definimos $|A|$ como o número de vértices em A e $\text{vol}(A) = \sum_{v_i \in A} d_i$ a soma do peso das arestas incidentes nos vértices do subconjunto A. Na Figura 3.3 dividimos o conjunto de vértices de G_1 (Figura 3.2) em dois subconjuntos, $A = \{v_1, v_2, v_3\}$ e $B = \{v_4, v_5, v_6, v_7, v_8\}$.

Para a divisão retratada na Figura 3.3, $W(A, B) = w_{24} + w_{26} + w_{36} + w_{37} = 1.3$. Além disso, $|A| = 3$, $\text{vol}(A) = 5.9$, $|B| = 5$ e $\text{vol}(B) = 10.7$. A seguir,

utilizaremos essas noções para trabalhar com o problema de clusterização em grafos ponderados.

Podemos resumir o problema de clusterização em grafos como encontrar uma partição do conjunto de vértices V desse grafo que otimize uma função objetivo. Em geral, os problemas de clusterização em grafos buscam partições onde vértices mais similares (pesos maiores nas arestas) tendam a estar em um mesmo cluster e vértices menos similares (até mesmo sem arestas) tendam a estar em clusters distintos. Nesse sentido, uma possível definição de função é o *cut* [78]. Dado um grafo $G = (V, E, \omega)$ e $\mathcal{C} = \{C_1, \ldots, C_k\}$ uma partição de V, definimos o $cut : \mathcal{U} \to \mathbb{R}$

$$cut(\mathcal{C}) = cut(C_1, \ldots, C_k) = \frac{1}{2} \sum_{\ell=1}^{k} W(C_\ell, \overline{C_\ell}), \tag{3.2}$$

onde W é a expressão definida em 3.1. O fator $1/2$ não permite uma contagem dupla dos pesos das arestas, afinal se $i \in A$ e $j \in B$, então w_{ij} é contada em $W(A, \overline{A})$ e em $W(B, \overline{B})$. Note que quanto menor é o valor $cut(\mathcal{C})$ menor é o peso das arestas entre clusters. Portanto, podemos definir o problema de clusterização com função objetivo *cut*. A boa notícia é que encontrar a partição ótima em relação ao *cut* é relativamente fácil e o problema pode ser resolvido eficientemente (veja [71]). No entanto, na prática a partição que minimiza o *cut* não é muito boa, pois frequentemente a solução do problema separa um único vértice de todo o resto do grafo e, normalmente, queremos clusterizar conjuntos grandes de dados (vértices) em poucos clusters, onde todos tenham tamanho muito menor que o conjunto original. Isso acontece pois o *cut* só analisa o que acontece entre pares de pontos de clusters distintos e não o que acontece entre pares de pontos do mesmo cluster.

Para corrigir isso, existem funções que consideram os pesos das arestas dentro de cada cluster. As duas funções objetivo mais comuns são o *Ratiocut* (equação (3.3)) e o *Ncut* (equação (3.4)). Mais exemplos de funções objetivo podem ser encontradas em [33]. Vale mencionar que o Ratiocut foi apresentado pela primeira vez por Hagen e Kahng [28]. O Ncut, conhecido como *normalized cut*, foi apresen-

tado pela primeira vez por Shi e Malik [65]. Foi a partir de [65] e [49] que os métodos espectrais se popularizaram na comunidade de *machine learning* [78].

$$\text{Ratiocut}(\mathcal{C}) = \frac{1}{2}\sum_{\ell=1}^{k}\frac{cut(C_\ell, \overline{C_\ell})}{|C_\ell|} = \sum_{\ell=1}^{k}\frac{W(C_\ell, \overline{C_\ell})}{|C_\ell|} \tag{3.3}$$

$$\text{Ncut}(\mathcal{C}) = \frac{1}{2}\sum_{\ell=1}^{k}\frac{cut(C_\ell, \overline{C_\ell})}{\text{vol}(C_\ell)} = \sum_{\ell=1}^{k}\frac{W(C_\ell, \overline{C_\ell})}{\text{vol}(C_\ell)} \tag{3.4}$$

Assim como para o *cut*, uma partição ótima nesse contexto, é a partição $\mathcal{C}$ de V que minimiza o valor de $\text{Ncut}(\mathcal{C})$ ou $\text{Ratiocut}(\mathcal{C})$. Observe que essas funções objetivo levam em consideração o que acontece entre clusters distintos e o que acontece no dentro do próprio cluster. Por um lado, os únicos pesos que aparecem no numerador dos termos em (3.4) e (3.3) são pesos de arestas cujos extremos estão em clusters distintas, de modo que minimizar a função favorece partições tais que vértices em clusters diferentes tenham peso pequeno. No denominador há uma pequena diferença.

Primeiro, perceba que, o mínimo da função $\sum_{\ell=1}^{k} 1/vol(C_\ell)$ é atingido quando os $vol(C_\ell)$ coincidem[1] (análogo para $\sum_{\ell=1}^{k} 1/|C_\ell|$). Então ao minimizar qualquer uma das funções objetivo Ncut ou Ratiocut beneficiamos partições balanceadas. Em relação às diferenças, em (3.4) o denominador do termo associado a C_ℓ conta o peso de cada aresta com ambas as extremidades em C_ℓ duas vezes, enquanto as outras arestas incidentes com C_ℓ são contadas apenas uma vez. Assim, aumentar o peso das arestas internas diminui o valor do corte. Enquanto que em (3.3) é contado apenas o número de vértices em cada cluster. Infelizmente, a mudança do *cut* para o Ncut ou Ratiocut torna o problema NP-difícil [79].

[1]Para ver isso, considere $f : \mathbb{R}^k_{>0} \to \mathbb{R}$, onde $f(x) = \sum_{i=1}^{k} \frac{1}{x_i}$, com a restrição de que $\sum_{i=1}^{k} x_i = N$ para algum $N > 0$. Pela restrição, $x_k = N - \sum_{i=1}^{k-1} x_i$. Logo $\sum_{i=1}^{k} \frac{1}{x_i} = \sum_{i=1}^{k-1} \frac{1}{x_i} + \frac{1}{N-\sum_{i=1}^{k-1} x_i}$. Diferenciando essa função percebe-se que as derivadas parciais são iguais a zero quando $x_i = x_j$.

Os cálculos que vêm a seguir são endereçados a resolver uma relaxação[2] do problema de clusterização para o Ncut e são facilmente replicáveis para o caso do Ratiocut. Assim, formalizamos o problema de clusterização relativo ao Ncut no Problema 3.1.

Problema 3.1. *Sejam $G = (V, E, \omega)$ um grafo ponderado e $k \in \mathbb{N}$. Queremos encontrar uma partição $\mathcal{C}^{(0)}$ tal que*

$$Ncut(\mathcal{C}^{(0)}) = \min_{\mathcal{C}} Ncut(\mathcal{C}). \tag{3.5}$$

Iremos modelar nosso problema de clusterização como um problema de matrizes de álgebra linear. Para isso, precisamos obter uma representação da partição no formato de uma matriz, o que pode ser feito através da matriz de incidência de cluster.

Definição 3.1. *$H \in \mathbb{R}^{n \times k}$ é uma matriz de incidência de cluster se ela satisfaz:*

(1) Toda linha de H tem uma, e somente uma, entrada diferente de zero.

(2) Toda coluna de H tem pelo menos uma entrada diferente de zero.

Perceba que, dado o grafo ponderado G, a partir de uma partição $\mathcal{C}$ qualquer, obtemos uma matriz de incidência de cluster, basta considerar $H^{(\mathcal{C})} = (h_{i\ell}^{(\mathcal{C})})$, onde $h_{i\ell}^{(\mathcal{C})} = 1$, se $x_i \in C_\ell$ e 0, caso contrário. Por outro lado, pela Definição 3.1, é possível construir uma partição a partir de qualquer matriz de incidência de cluster. Se $H^{(\mathcal{C})} \in \mathbb{R}^{n \times k}$ satisfaz a Definição 3.1, seja $\mathcal{C} = \{C_1, \ldots, C_k\}$ tal que $x_i \in C_\ell$ se, e somente se, $h_{i\ell}^{(\mathcal{C})} \neq 0$. É fácil ver que $\mathcal{C}$ é uma partição de X. Voltando ao nosso problema, $H^{(\mathcal{C})} = (h_{i\ell}^{(\mathcal{C})})$ terá uma forma especial.

[2]Relaxação é uma estratégia de modelagem, onde aproximamos um problema difícil por um problema próximo mais fácil de resolver.

Dados um grafo ponderado $G = (V, E, \omega)$, sem vértices isolados, e $k \in \mathbb{N}$, seja $\mathcal{C}$ partição de V em k clusters. Definimos $H^{(\mathcal{C})} = (h_{i\ell}^{(\mathcal{C})}) \in \mathbb{R}^{n\times k}$ por

$$h_{i\ell}^{(\mathcal{C})} = \begin{cases} \frac{1}{\sqrt{vol(C_\ell)}}, & \text{se } x_i \in C_\ell; \\ 0, & \text{caso contrário.} \end{cases} \tag{3.6}$$

Para facilitar a notação utilizaremos $H = H^{\mathcal{C}}$. Para esse grafo G e número de clusters k definimos o conjunto que contém todas matrizes H no formato da equação (3.6) por $\mathcal{H}$, isto é, $\mathcal{H} = \{H \in \mathbb{R}^{n\times k} : H \text{ no formato da equação (3.6)}\}$. Uma matriz $H \in \mathcal{H}$ possui duas propriedades:

(1) $H^T DH = I_k$, onde I_k é a matriz identidade de ordem k.

(2) $tr(H^T LH) = Ncut(\mathcal{C})$, onde L é a matriz Laplaciana de G com respeito a matriz de similaridade S.

Mostraremos o item (2). A demonstração do item (1) é análoga à do item (2) para calcular as entradas da matriz

Demonstração. Vamos mostrar que $(H^T LH)_{\ell\ell} = \frac{cut(C_\ell, \overline{C_\ell})}{vol(C_\ell)}$.
Para cada $\ell = 1, \ldots, k$ observe que $(H^T L)_{\ell j} = \sum_{i\in C_\ell} \frac{1}{\sqrt{vol(C_\ell)}} L_{ij}$. Assim,

$$\begin{aligned}(H^T LH)_{\ell\ell} &= \sum_{j\in C_\ell} \frac{1}{\sqrt{vol(C_\ell)}} \sum_{i\in C_\ell} \frac{1}{\sqrt{vol(C_\ell)}} L_{ij} \\ &= \sum_{j\in C_\ell} \sum_{i\in C_\ell} \frac{1}{vol(C_\ell)} L_{ij} \\ &= \frac{1}{vol(C_\ell)} \Big(\sum_{i=j\in C_\ell} L_{ij} + \sum_{i\neq j\in C_\ell} L_{ij} \Big)\end{aligned}$$

Pela definição de L, $L_{ij} = d_i$ se $i = j$ e, $L_{ij} = -w_{ij}$, caso contrário. Então segue que,

$$vol(C_\ell) \cdot (H^T LH)_{\ell\ell} = \sum_{i\in C_\ell} d_i - \sum_{i\neq j\in C_\ell} w_{ij}$$

$$
\begin{aligned}
&= \sum_{i\in C_\ell}(\sum_{j\in V} w_{ij}) - \sum_{i\in C_\ell}(\sum_{j\neq i\in C_\ell} w_{ij}) \\
&= \sum_{i\in C_\ell}(\sum_{j\in V} w_{ij} - \sum_{j\neq i\in C_\ell} w_{ij}) \\
&= \sum_{i\in C_\ell}(\sum_{j\notin C_\ell} w_{ij}) \\
&= W(C_\ell, \overline{C_\ell}) \\
&= \frac{1}{2}\left(W(C_\ell, \overline{C_\ell}) + W(\overline{C_\ell}, C_\ell)\right) = cut(C_\ell, \overline{C_\ell})
\end{aligned}
$$

Logo, $(H^T LH)_{\ell\ell} = \frac{cut(C_\ell, \overline{C_\ell})}{vol(C_\ell)}$. Portanto, $tr(H^T LH) = \sum_{\ell=1}^{k} \frac{cut(C_\ell, \overline{C_\ell})}{vol(C_\ell)} = \text{Ncut}(\mathcal{C})$. □

Com essas propriedades, podemos reescrever o problema de clusterização como um problema de matrizes, observando que antes o conjunto universo era formado por todas as partições de um conjunto X em k clusters, enquanto que agora o conjunto universo consiste em todas matrizes $H \in \mathcal{H}$.

Problema 3.2. *Dados um grafo ponderado $G = (V, E, \omega)$ e $k \in \mathbb{N}$, o objetivo é encontrar $H^{(0)}$ tal que*

$$
tr(H^{(0)T} LH^{(0)}) = \min_{H\in\mathcal{H}} tr(H^T LH), \text{ sujeito a } H^T DH = I. \tag{3.7}
$$

Observe que o Problema 3.1 é equivalente ao Problema 3.2. Assim, modelamos nosso problema de clusterização como um problema de matrizes de álgebra linear. No Problema 3.3 apresentamos um problema de otimização similar ao Problema 3.2.

Problema 3.3. *Dados um grafo ponderado $G = (V, E, \omega)$ e $k \in \mathbb{N}$, o objetivo é encontrar $Y^{(0)}$ tal que*

$$
tr(Y^{(0)T} \mathcal{L} Y^{(0)}) = \min_{Y\in\mathbb{R}^{n\times k}} tr(Y^T \mathcal{L} Y), \text{ sujeito a } Y^T Y = I
$$

É sabido como encontrar uma solução para o Problema 3.3. Nesse sentido, apresentamos o Teorema 3.1.

Teorema 3.1. *[Rayleigh-Ritz [31]] Seja M matriz real de ordem n e simétrica com autovalores $\lambda_1 \leq \ldots \leq \lambda_n$ correspondentes à base de autovetores ortonormais $u_1, \ldots, u_n$, respectivamente. Para todo $k \leq n$, vale:*

$$\min_{Y \in \mathbb{R}^{n \times k}, Y^T Y = I} tr(Y^T M Y) = \lambda_1 + \cdots + \lambda_k. \tag{3.8}$$

Se Y for tal que suas colunas geram o $Span\{u_1, \ldots, u_k\}$, então Y é uma solução de 3.8.

A demonstração do Teorema 3.1 pode ser encontrada a partir da página 246 em [31], no Teorema 4.3.28 e Corolário 4.3.39. Então, pelo Teorema 3.1, para encontrar a solução do Problema 3.3 basta calcular os k autovetores associados aos k menores autovalores da matriz Laplaciana normalizada $\mathcal{L}$ de G. É claro que gostaríamos de reduzir nosso problema de clusterização NP-difícil para um problema de encontrar autovetores, mas a solução que encontramos para o Problema 3.3 não serve para o Problema3.2. Apesar do Problema 3.3 possuir similaridades com o Problema 3.2, há um aspecto diferente. No contexto do Problema 3.3 a única restrição sobre a matriz $Y \in \mathbb{R}^{n \times k}$ é que $Y^T Y = I$, enquanto que no caso que queremos resolver $H \in \mathbb{R}^{n \times k}$ deve pertencer a $\mathcal{H}$ e $H^T D H = I$. Este último é possível adequar ao Problema 3.3, basta fazer a substituição $H = D^{-1/2} Y$. Dessa forma, $H^T L H$ passa a ser $(D^{-1/2}Y)^T L D^{-1/2} Y = Y^T (D^{-1/2} L D^{-1/2}) Y = Y^T \mathcal{L} Y$, onde $\mathcal{L} = D^{-1/2} L D^{-1/2}$ é a matriz Laplaciana normalizada de S, a mesma utilizada no Algoritmo 4. Nesse caso, substituímos o conjunto $\mathcal{H}$ por $\mathcal{Y} = \{Y : Y = H D^{1/2},\ H \in \mathcal{H}\}$. A partir dessa modelagem, nosso problema de álgebra linear se resume em:

Problema 3.4. *Dados um grafo ponderado $G = (V, E, \omega)$ e $k \in \mathbb{N}$, o objetivo é encontrar $Y^{(0)}$ tal que*

$$tr(Y^{(0)T} \mathcal{L} Y^{(0)}) = \min_{Y \in \mathcal{Y}} tr(Y^T \mathcal{L} Y),\ \textit{sujeito a}\ Y^T Y = I.$$

Assim, a única diferença do Problema 3.4 para o Problema 3.3 é o formato da matriz Y. Em ambos ela precisa respeitar a restrição $Y^T Y = I$, mas, no

primeiro, o conjunto universo é $U_1 = \{Y \in \mathbb{R}^{n\times k} : Y \in \mathcal{Y} \text{ e } Y^TY = I\}$, enquanto que no segundo, o conjunto onde buscamos uma solução é $U_2 = \{Y \in \mathbb{R}^{n\times k} : Y^TY = I\}$. É claro que $U_1 \subset U_2$, logo, sendo $\mathcal{L}$ a matriz Laplaciana normalizada de um grafo ponderado $G = (V, E, \omega)$, temos que

$$f(\mathcal{L}) := \min_{Y\in U_2} tr(Y^T\mathcal{L}Y) \leq \min_{Y\in U_1} tr(Y^T\mathcal{L}Y) = \min_{\mathcal{C}\in\mathcal{U}_k} \text{Ncut}(\mathcal{C}), \tag{3.9}$$

onde $\mathcal{U}_k$ é o conjunto de todas partições de V em k clusters.

Como é comum em relaxações, o lado esquerdo da desigualdade (3.9) pode ser calculado eficientemente e fornece um limite inferior para o valor do Ncut da partição ótima. Por outro lado, não há conexão óbvia entre uma matriz Y que atinge o valor $f(\mathcal{L})$ (ou seja, uma matriz construída a partir de autovetores associados aos k menores autovalores de $\mathcal{L}$) e uma partição em k clusters $\mathcal{C}^{(0)}$ tal que $\text{Ncut}(\mathcal{C}^{(0)})$ seja próximo do lado direito da desigualdade (3.9). Na literatura há alguns métodos que transformam uma matriz Y, que atinge o valor $f(\mathcal{L})$, em uma partição de G [65, 49]. Motivados pela teoria da perturbação de matrizes, Ng *et al.* [49] sugerem que se construa uma matriz $\tilde{Y}$, normalizando as linhas de Y. Para então, clusterizar as linhas da matriz $\tilde{Y}$ utilizando Algoritmo k-means e atribuir o vértice $v_i \in V$ ao mesmo cluster da i-ésima linha de Y.

Uma maneira de avaliar a qualidade da partição de saída $\mathcal{C}$ do Algoritmo k-means é observar a proporção $\text{Ncut}(\mathcal{C}^{(0)})/f(\mathcal{L}) \geq 1$. Se esta razão for exatamente 1, a partição $\mathcal{C}^{(0)}$ é uma partição ótima. Caso contrário, ele fornece um limite superior no valor da razão $\rho(\mathcal{C}^{(0)}) = \text{Ncut}(\mathcal{C}^{(0)})/\min_{\mathcal{C}\in\mathcal{U}_k}\text{Ncut}(\mathcal{C})$. Entretanto, mencionamos que a diferença entre $f(\mathcal{L})$ e $\min_{\mathcal{C}\in\mathcal{U}_k}\text{Ncut}(\mathcal{C})$ pode ser arbitrariamente grande. Mesmo assim, os métodos espectrais se mostraram muito eficientes em muitas aplicações práticas, referenciamos [49, 33, 78] para maiores explicações sobre os resultados empíricos.

Em suma, essa discussão motiva o Algoritmo 5.

Algoritmo 5: Clusterização Espectral com base no Ncut

Entrada: Conjunto de dados X e inteiro positivo k.

Saída: Partição $\mathcal{C} = \{C_1, \ldots, C_k\}$.

1 Construa a matriz de similaridade $S = (s_{ij})$.

2 Calcule $\mathcal{L} = D^{-1/2}(D - S)D^{-1/2}$, onde $D = (d_{ij})$ é uma matriz diagonal cuja i-ésima entrada é dada por $d_{ii} = \sum_{j=1}^{n} s_{ij}$.

3 Encontre os autovetores ortonormais $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k \in \mathbb{R}^n$, associados aos k menores autovalores $\lambda_1 \leq \cdots \leq \lambda_k$ de $\mathcal{L}$, respectivamente. Considere a matriz $\mathbf{X} = [\mathbf{x}_1 \mathbf{x}_2 \cdots \mathbf{x}_k] \in \mathbb{R}^{n \times k}$, cujas colunas são dadas pelos vetores $\mathbf{x}_i$

4 Defina a matriz $\mathbf{Y} = (\mathbf{Y}_{ij})$ a partir de $\mathbf{X}$, onde $\mathbf{Y}_{ij} = \mathbf{X}_{ij} / \sqrt{\sum_{j'=1}^{n} \mathbf{X}_{ij'}^2}$.

5 Faça o seguinte para $j \in \{1, \ldots, Q\}$. Tratando cada linha de $\mathbf{Y}$ como um vetor de $\mathbb{R}^k$, utilize o Algoritmo k-means, com inicialização aleatória, para obter uma partição desses vetores. Atribua o objeto original $x_i \in X$ para o cluster ℓ se, e somente se, a i-ésima linha de $\mathbf{Y}$ foi atribuída para o cluster ℓ. Denote essa partição de X por $\mathcal{P}_j$.

6 Seja $\mathcal{C}$, a partição com menor Ncut entre todas partições obtidas no passo (5).

7 **Retorne** $\mathcal{C}$

O Algoritmo 5 é basicamente o Algoritmo 4 iterado Q vezes dando liberdade ao usuário para escolher uma medida de similaridade mais adequada. Abaixo sumarizamos como essa seção motiva os passos do Algoritmo 5. Dados um conjunto de dados $X = \{x_1, \ldots, x_n\}$ e $k \in \mathbb{N}$, seja S a matriz de similaridade de X.

(1) Relaxamos o Problema 3.1 para o Problema 3.4.

(2) Resolvemos o Problema 3.4, considerando a matriz $\mathbf{X} = [\mathbf{x}_1 \mathbf{x}_2 \cdots \mathbf{x}_k] \in \mathbb{R}^{m \times k}$, cujas colunas são dadas pelos autovetores ortonormais associados aos k menores autovalores de $\mathcal{L} = D^{-1/2}(D - S)D^{-1/2}$.

(3) Para gerar uma partição de X a partir de $\mathbf{X}$, calculamos a matriz Y normalizando as linhas de $\mathbf{X}$ e aplicamos o Algoritmo k-means nas

linhas de Y. A partição gerada é tal que um ponto $x_i \in X$ é atribuído ao mesmo cluster da i-ésima linha de Y.

3.3 A estrutura dos métodos espectrais e aplicações

Na Seção 3.1 apresentamos um dos primeiros métodos espectrais associado ao Ncut e na Seção 3.2 expomos a fundamentação teórica em grafos e álgebra linear que subsidia o método. Nessa seção buscamos generalizar o Algoritmo 5. Dados um conjunto de objetos X e $k \in \mathbb{N}$, um método espectral tem a seguinte estrutura [33]:

(1) Definir uma matriz de similaridade S.

(2) Calcular uma matriz Laplaciana[3] a partir de S, dependendo da função objetivo utilizada.

(3) Selecionar os autovetores da matriz Laplaciana e como será feita a partição de X em k clusters a partir deles.

No caso do Algoritmo 4, a matriz de similaridade utilizada é baseada no kernel Gaussiano. O problema relaxado do Ncut induz a utilização da matriz Laplaciana normalizada $\mathcal{L}$. São selecionados os k autovetores associados aos k menores autovalores, tal que a partição dos objetos de X é obtida após clusterizar uma matriz associada aos autovetores de $\mathcal{L}$. A medida de similaridade do kernel Gaussiano é sugerida pelos autores de [49].

A partir disso, a pesquisa em clusterização espectral é concentrada em cinco aspectos [33]:

[3]Algumas opções são a matriz Laplaciana não normalizada, Laplaciana normalizada ou a p-Laplaciana [42].

(1) Construir uma matriz de similaridade S [92, 89, 47, 38].

(2) Escolher a matriz Laplaciana [65, 42, 25, 17].

(3) Definir o número de clusters k [78, 81, 21, 75].

(4) Seleção de autovetores [65, 49, 87, 93, 36, 53].

(5) Aplicações da clusterização espectral [67, 66, 40, 91, 16, 8].

Nesse trabalho atuamos em duas vertentes da pesquisa em clusterização espectral. No capítulo 4 realizamos duas aplicações do Algoritmo 4 em situações reais. E, no capítulo 5, nos aprofundamos na problemática por trás de definir uma matriz de similaridade e as limitação da medida de similaridade utilizada pelo Algoritmo 4. Além disso, iremos definir uma nova medida de similaridade.

A respeito do item (3) dos problemas de pesquisa em clusterização espectral, existem critérios derivados dos métodos espectrais que ajudam o usuário a escolher o número de clusters k. Nesse trabalho vamos utilizar um deles: o *eigengap* [78]. Nesse critério, o objetivo é escolher um número de classes k tal que os autovalores $\lambda_1, \ldots, \lambda_k$ são bem pequenos e λ_{k+1} é relativamente grande, onde $\lambda_1 \leq \cdots \leq \lambda_n$ são os autovalores da matriz Laplaciana normalizada $\mathcal{L}$. A principal justificativa para esse processo é uma propriedade da matriz Laplacina normalizada: o grafo ponderado G possui exatamente k componentes conexas se, e somente se, $\lambda_1 = \cdots = \lambda_k = 0$ e $\lambda_{k+1} > 0$, onde $\lambda_1 \leq \cdots \leq \lambda_n$ são os autovalores da matriz Laplaciana normalizada $\mathcal{L}$ de G[4] [13]. Perceba que se um grafo ponderado (o grafo que representa a similaridade entre os objetos do conjunto de dados) estiver dividido em exatamente k componentes, essas seriam exatamente os k clusters que buscaríamos. É claro que, na prática, raramente o grafo obtido pelo conjunto de dados é dividido em componentes conexas.

[4] $\mathcal{L} = D^{-1/2}(D - S)D^{-1/2}$, onde S é a matriz de similaridade de G e D é a matriz diagonal dos graus dos vértices de G.

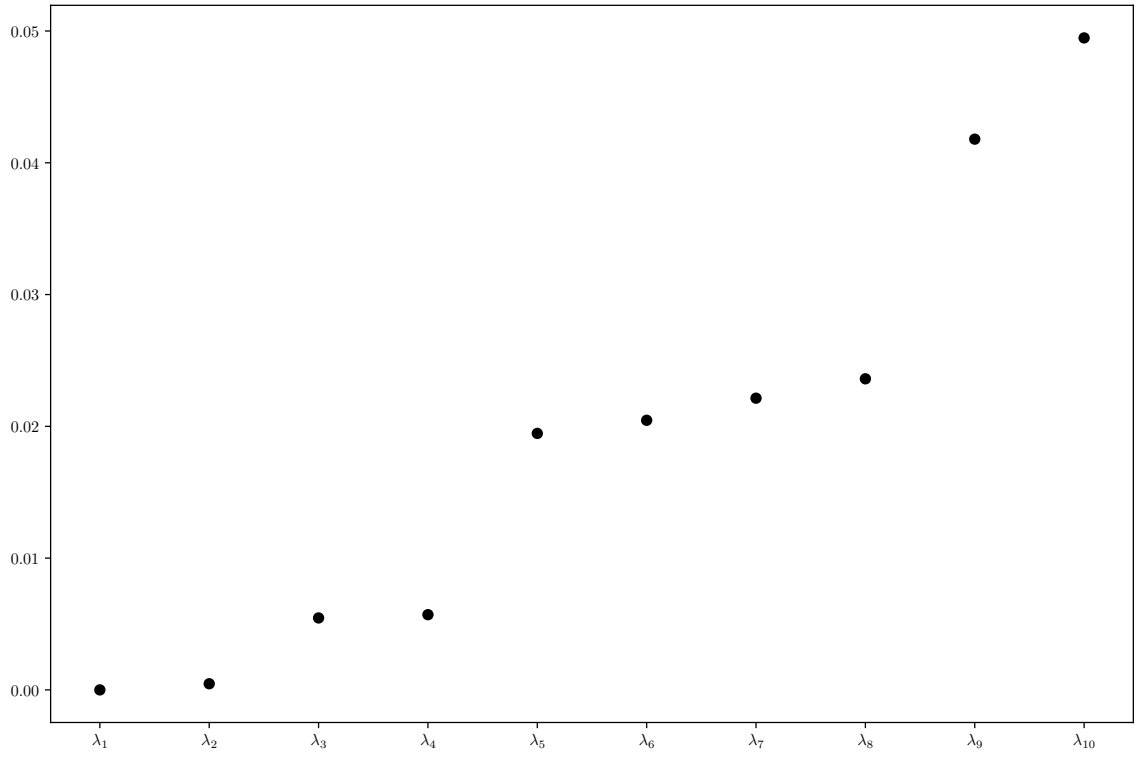

Figura 3.4: Sequência dos dez primeiros autovalores da matriz Laplaciana normalizada obtida a partir de S.

O *eigengap* pode ser medido pela primeira diferença considerável ou pela maior razão entre autovalores consecutivos, ou seja, considerar a maior das diferenças $|\lambda_{k+1} - \lambda_k|$ ou das razões λ_{k+1}/λ_k, quando λ_k é um valor pequeno. A ideia é que *eigengap* seja grande e que λ_k esteja próximo de zero.

Exemplo 3.3. Nós iremos ilustrar o *eigengap* no conjunto de dados X do exemplo 3.1 (superior). Consideramos a mesma matriz de similaridade S utilizada no exemplo. A Figura 3.4 apresenta os primeiros 10 autovalores da matriz Laplaciana normalizada $\mathcal{L}$. Os valores aproximados foram $\lambda_1 = 0, \lambda_2 = 0.0004, \lambda_3 = 0.0054, \lambda_4 = 0.0057, \lambda_5 = 0.019, \lambda_6 = 0.020, \lambda_7 = 0.022, \lambda_8 = 0.023, \lambda_9 = 0.41, \lambda_{10} = 0.049$. Note que o primeiro salto acontece entre λ_2 e λ_3, onde $|\lambda_3 - \lambda_2| \approx \lambda_3$ e $\lambda_3/\lambda_2 \approx 11.70$, o que sugere $k = 2$ para o número de clusters nesse problema de clusterização.

Contudo, é importante notar que a matriz Laplaciana normalizada é construída a partir da matriz de similaridade. Dessa forma, a qualidade do *eigengap* depende diretamente da escolha de uma boa matriz de similaridade.

Para finalizar esse capítulo, vamos mostrar os resultados do Algoritmo 4 em nove conjuntos de dados contidos em $\mathbb{R}^2$ (Figura 3.5) e em dois conjuntos de dados contidos em $\mathbb{R}^3$ (Figura 3.6). O valor de σ utilizado em cada conjunto de dados estará destacado no título da figura que contém o conjunto. Executamos o passo (4) do Algoritmo 4 $Q = 100$ vezes e escolhemos a solução que apresentou o menor Ncut.

O método espectral utilizado consegue lidar bem com todos os conjuntos de dados presentes na Figura 3.5 e na Figura 3.6. Vale mencionar que tanto o Algoritmo k-means, quanto o Algoritmo single linkage, podem lidar bem com um subconjunto desses conjuntos de dados, mas nenhum deles consegue atingir os mesmos resultados do algoritmo de clusterização espectral. Por exemplo, o Algoritmo k-means não consegue dividir as duas circunferências em dois clusters distintos na Figura 3.5 (diagonal inferior) ou as duas luas em clusters distintos na Figura 3.5 (diagonal superior), pois o Algoritmo k-means só produz clusters convexos. O Algoritmo single linkage não consegue dividir as duas circunferências em dois clusters distintos na Figura 3.5 (superior), pois ele identifica o *outlier* como um cluster solitário.

No próximo capítulo abordaremos aplicações em situações reais onde o contexto do problema é mais complexo que apenas clusterizar pontos do espaço.

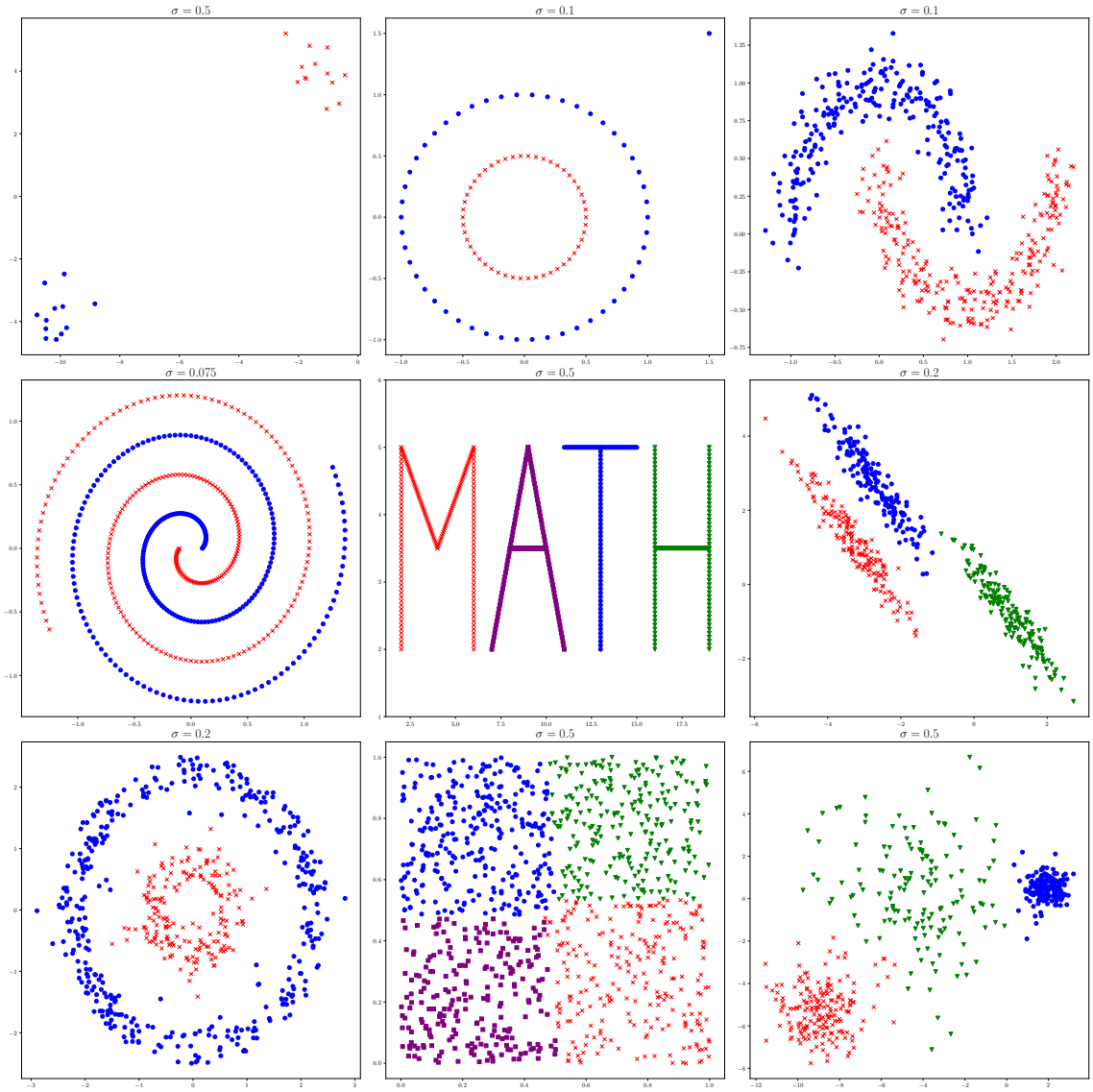

Figura 3.5: Resultados do Algoritmo 4 em nove conjuntos de dados de $\mathbb{R}^2$.

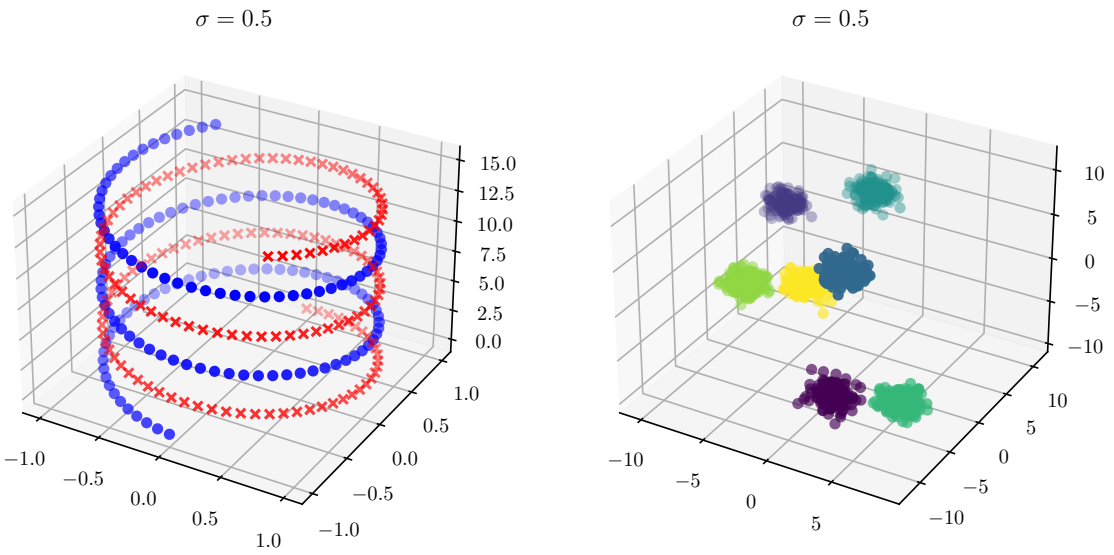

Figura 3.6: Resultados do Algoritmo 4 em dois conjuntos de dados de $\mathbb{R}^3$.

4 APLICAÇÕES

Esse capítulo apresenta aplicações do algoritmo de clusterização espectral a problemas com dados reais. Realizamos essas aplicações para avaliar o desempenho do algoritmo em situações desse tipo. A primeira aplicação está relacionada ao mercado financeiro, enquanto que a segunda está relacionada à pandemia da COVID-19. Cada seção inicia contextualizando o leitor sobre o problema e contempla três aspectos: (1) dados utilizados, (2) metodologia e (3) resultados.

4.1 Portfólios baseados em Clusterização Espectral e Factor Investing

Nessa seção apresentamos uma aplicação do algoritmo espectral em um problema sobre investimentos na bolsa de valores, a saber, o problema de montar um portfólio de ações com bom desempenho. Iniciamos com algumas noções do mercado financeiro brasileiro.

Uma sociedade anônima é uma forma jurídica de constituição de empresas na qual o capital da empresa está dividido em unidades de capital indivisíveis, onde cada unidade é chamada de ação [10]. Os proprietários das ações são chamados de acionistas. As empresas que são sociedades anônimas podem ser de capital fechado ou capital aberto. No primeiro caso, há poucos acionistas e para uma nova pessoa adquirir ações, deve ser realizada uma escrituração da transferência da propriedade [54]. Por outro lado, se a empresa é de capital aberto, então parte de suas ações são livremente negociadas sem necessidade de escrituração pública de propriedade [55].

A bolsa de valores é justamente o local onde são negociados valores mobiliários, entre eles as ações de empresas de capital aberto [32]. Uma empresa

ingressa na bolsa de valores através de uma oferta pública inicial (IPO[1]) realizando uma oferta de ações ao mercado. Isso significa que a empresa deixará de ser uma empresa de capital fechado para se tornar uma empresa de capital aberto. Nessa oferta pública, é estabelecido um preço inicial para cada ação que deverá ser pago pelos novos investidores [56]. Após o IPO, as ações oferecidas ao mercado podem ser comercializadas livremente pelos investidores, fazendo com que os preços oscilem de acordo com a oferta e demanda.

Algumas noções de contabilidade também são importantes. O lucro líquido de uma empresa em um período T é a diferença entre a receita e o custo total no período T, onde a receita é toda renda gerada pela empresa e o custo é toda saída monetária da empresa [58]. Dividendos são uma parcela do lucro líquido que é distribuída aos acionistas [59]. Os ativos de uma empresa consistem em tudo que pode ser convertido em valores monetários, como produtos em estoque, maquinário, imóveis ou dinheiro em caixa [80]. Já passivos de uma empresa são as obrigações que ela tem com terceiros, como dívidas ou contas [80]. O patrimônio líquido é um indicador contábil que representa a diferença entre ativo e passivo de uma empresa [80].

As ações de uma empresa podem ser classificadas em dois tipos: ON (Ordinárias nominativas), com direito a voto ou PN (Preferenciais nominativas), com direito preferencial aos dividendos [57]. Na bolsa de valores, as ações de uma empresa recebem um código de 4 letras (em geral uma abreviação do nome da empresa) e um número $c \in \{3, 4, 5, 6, 7, 8, 11\}$. Quando $c = 3$ o código representa uma ação ON, quando $c \in \{4, 5, 6, 7, 8\}$ o código representa uma ação PN e, quando $c = 11$ o código representa uma ação UNIT, que é uma ação que é simultaneamente ON e PN. Seja $p_i : \mathbb{Q} \to \mathbb{Q}$ tal que $p_i(t)$ é o preço de fechamento da ação x_i no dia t, onde preço de fechamento é o último preço negociado daquela ação no dia que t representa. $A_{i_1}(t)$, $A_{i_2}(t)$ e $A_{i_3}(t)$ são o número de ações ON, PN e UNIT de uma empresa, onde x_{i_1}, x_{i_2} e x_{i_3} são as ações ON, PN e UNIT dessa empresa, respectivamente. O valor de

[1]Initial Public Offering

mercado de uma empresa no dia t é dado por $A_{i_1}(t)\cdot p_{i_1}(t)+A_{i_2}(t)\cdot p_{i_2}(t)+A_{i_3}(t)\cdot p_{i_3}(t)$. O desempenho de uma ação x_i no período $T = (t_i, t_f)$ é dado por

$$d_i(T) = \frac{p_i(t_f)}{p_i(t_i)}, \tag{4.1}$$

onde t_i e t_f representam uma data inicial e final, respectivamente.

Por exemplo, no dia 4 de fev. de 2022 a empresa brasileira Petrobras possuía um total de $N = 13.044.496.930$, divididas em 7.442.454.142 ações ON (PETR3) e 5.602.042.788 ações PN (PETR4). Na bolsa de valores, a empresa negocia um total de $49,5\%$ de suas ações ON e $81,51\%$ de suas ações PN. Se t representasse o dia 4 de fev. de 2022 e x_{i_1} e x_{i_2} representassem as ações PETR3 e PETR4, respectivamente, então $p_{i_1}(t) = R\$35,91$ e $p_{i_2}(t) = R\$32,63$. Juntando essas informações, o valor de mercado da Petrobras é $A_{i_1}(t) \cdot p_{i_1}(t) + A_{i_2}(t) \cdot p_{i_2}(t) = R\$450.053.184.412$. No período $T = (t_i, t_f)$, onde t_i representa o dia 1 de jan. de 2021 e t_f o dia 31 de dez. 2021, x_{i_1} =PETR3 obteve um desempenho de $6,41\%$, visto que $p_{i_1}(t_i) = R\$28,85$ e $p_{i_1}(t_f) = R\$30,70$. No mesmo período T, $d_{i_2}(T) = 0,11\%$[2].

No Brasil, existe apenas uma bolsa de valores, a B3 - Brasil, Bolsa, Balcão [5]. A negociação de ações ocorre em dias úteis, entre 10 e 17 horas (ou 18 horas, dependendo da época do ano), no que é conhecido como pregão. No dia 4 de fevereiro de 2022 havia 397 empresas com ações listadas na bolsa de valores, totalizando um valor de mercado de R$ 4.733.264.568.900 [4]. O Ibovespa (Índice bovespa) é o principal indicador de desempenho das ações negociadas na B3, conforme [7]. O desempenho do Ibovespa é uma média ponderada do desempenho de um subconjunto de ações da B3. O Ibovespa é atualizado a cada quadrimestre e para uma ação participar no índice ela deve satisfazer alguns critérios, entre eles ser negociada a um valor acima de R$1, estar entre as ações mais negociadas da bolsa e estar presente em mais de 95% dos pregões [7]. Note que mais de um tipo de ação da mesma empresa pode participar do índice. Além disso, o peso de cada ação no

[2]Dados retirados de `https://statusinvest.com.br/acoes/petr3`. Acesso em: 4 fev. 2022.

índice é proporcional ao valor de mercado da empresa[3]. Em suma, o Ibovespa é um índice que acompanha o desempenho das maiores empresas brasileiras. Por isso, em geral, os investidores tem o Ibovespa como uma referência para o desempenho de seus portfólios.

A bolsa de valores vem se popularizando entre as pessoas físicas brasileiras. Do início de 2018 até o fim do primeiro semestre de 2021 houve cerca de 3 milhões de novos CPF's registrados em operações da B3, onde em 2018 haviam cerca de 800 mil pessoas físicas [6]. Em geral, essas pessoas buscam comprar ações com dois objetivos principais: (1) vender a ação por um preço maior do que pagaram e (2) receber dividendos da empresa. Não é obvio qual ação deve ser comprada para atingir esses objetivos. Além disso, como as oscilações nos preços podem ser negativas, o retorno financeiro do investidor pode ser negativo. Dessa forma, é comum o investidor formar um portfólio $\mathcal{P}$ de ações, onde $\mathcal{P}$ é um subconjunto do conjunto de ações X. O desempenho $d_{\mathcal{P}}(T)$ do portfólio no período T é dado por $\frac{1}{|\mathcal{P}|}\sum_{x_i \in \mathcal{P}} d_i(T)$. Portanto temos por objetivo construir $\mathcal{P}$, tal que

$$d_{\mathcal{P}}(T) = \max_{\mathcal{P}'} d_{\mathcal{P}'}(T). \tag{4.2}$$

É claro que o portfólio que representa a solução ótima é composto por uma única ação: a ação que mais irá valorizar na bolsa de valores. Infelizmente, tentar prever qual será essa ação é muito arriscado, visto que essa ação pode ter um desempenho muito ruim por causa algum evento particular que ocorra na empresa ou no seu setor. Uma alternativa menos arriscada é investir no código BOVA11[4] que é lastreado no Ibovespa, isto é, possui desempenho muito vinculado ao do Ibovespa. O desempenho do Ibovespa é muito sensível à variação de empresas de poucos setores, que representam em mais de 40% do índice: Vale e Petrobras (commodities)

[3]A composição do Ibovespa pode ser vista em `https://www.b3.com.br/pt_br/market-data-e-indices/indices/indices-amplos/ibovespa.htm`. Acesso em: 4 fev. 2022.

[4]Mais informações em `https://www.blackrock.com/br/products/251816/ishares-ibovespa-fundo-de-ndice-fund`. Acesso em: 4 fev. 2022.

e Banco do Brasil, Itaú e Bradesco (setor bancário). Ou seja, um problema no mercado de commodities ou bancário afeta muito o índice. Visto essa concentração do Ibovespa em poucos setores, normalmente o investidor busca um retorno maior que o índice. Assim, o investidor deve montar seu portfólio ou delegar para algum fundo de investimentos a seleção do portfólio. Infelizmente essa segunda opção não tem se mostrado muito vantajosa, tendo em vista que, em um período de três anos, 44% dos fundos tiveram desempenho menor que o Ibovespa [76]. Além disso, esses fundos possuem taxas altas que consomem o desempenho do investidor. Devido a isso, as pessoas físicas estão mais motivadas a montar seu próprio portfólio. O problema é definir quantas ações e, principalmente, quais ações vão ser selecionadas para compor o portfólio.

Retomando o primeiro questionamento do problema, economistas defendem que um portfólio deve ser diversificado, contendo várias ações descorrelacionadas [48]. Dessa forma, o desempenho de uma ação por um evento particular impacta pouco no desempenho do portfólio como um todo. Entretanto, não há consenso entre os economistas sobre o número ideal de ações em um portfólio [48]. O problema é que, à medida que são colocadas mais ações, uma ação particular com bom desempenho impacta pouco positivamente no desempenho do portfólio. Diversos autores sugerem que o tamanho do portfólio deve ser de 8 a 20 ações, dependendo da correlação entre as ações selecionadas, veja [26, 46, 85]. Ainda assim, escolher as ações do portfólio é um desafio para o investidor. A seguir apresentamos uma estratégia bem difundida no mercado financeiro.

Existem critérios bem estabelecidos entre economistas para fazer essa seleção, conhecidos como critérios de *factor investing* (investimento em fatores) [9]. A premissa é que os riscos e retornos de uma ação podem ser explicados através de fatores. Um fator que vamos utilizar é o fator valor que visa identificar ações cujos preços estejam subvalorizados pelo mercado [9]. Uma forma de quantificar esse fator é pelo critério preço sobre lucro (p/l). Dada uma ação x_i e um período $T = (t_i, t_f)$,

o p/l de x_i é medido por $\frac{p_i(t_f)}{e_i(T)}$, onde $p_i(t_f)$ é o preço de x_i em $t = t_f$ e $e_i(T)$ é o lucro líquido por ação de x_i no período T. Além disso, podemos entender $\frac{p_i(t_f)}{e_i(T)} > 0$ como a quantidade de anos necessários para a empresa retornar ao acionista o valor investido em lucro líquido, assim, quanto menor é $\frac{p_i(t_f)}{e_i(T)}$, mais rápido é o retorno ao acionista e mais desvalorizada está a ação.

Outro fator bem estabelecido que iremos utilizar nesse trabalho é o fator qualidade, que visa capturar qualidade de uma empresa [9]. Um critério que quantifica esse fator é o Retorno sobre Patrimônio Líquido (ROE[5]). O ROE da a ação x_i no período $T = (t_i, t_f)$ é dado por $\frac{l_i(T)}{PL_i(t_f)}$, onde $PL_i(t_f)$ é o patrimônio líquido de x_i no dia $t = t_f$ e $l_i(T)$ é o lucro líquido da empresa no período T. Ou seja, o ROE pode ser entendido como o lucro líquido da empresa por cada real de patrimônio líquido, portanto é razoável que quanto maior, mais eficiente é a empresa.

Portfólios podem ser montados diretamente dos fatores de *factor investing* [9]. Uma possibilidade é compor o portfólio $\mathcal{P}$ com as k ações do Ibovespa com menor p/l positivo. Outra possibilidade é compor o portfólio $\mathcal{P}$ com as k ações com maior ROE positivo [48].

Nesse trabalho propomos uma nova estratégia de montagem de portfólio de ações que utiliza clusterização espectral combinada com *factor investing*. Essa estratégia consiste em dois passos: (1) utilizar o algoritmo espectral para dividir ações em um número pré-estabelecido k de clusters, onde ações mais correlacionadas tendam a estar em um mesmo cluster e (2) escolher uma ação de cada cluster, a partir de um critério pré-definido de *factor investing*.

[5]Return on Equity.

4.1.1 Estratégia

Seja $X = \{x_1, \ldots, x_n\}$ um conjunto universo de ações, fixe $k \in \mathbb{N}$ e um período $T = (t_i, t_f)$.

Para cada ação $x_i \in X$, defina o vetor $\mathcal{O}^{(T)}(x_i) = p_i = (p_i(t_i), \ldots, p_i(t_f))$. Defina a correlação [61] entre $\mathcal{O}^{(T)}(x_i)$ e $\mathcal{O}^{(T)}(x_j)$ como

$$w_{ij}^{(T)} = \frac{\sum\limits_{t_i \leq t \leq t_f} (p_i(t) - \overline{p_i})(p_j(t) - \overline{p_j})}{\sqrt{\sum\limits_{t_i \leq t \leq t_f} (p_i(t) - \overline{p_i})^2 (p_j(t) - \overline{p_j})^2}}, \tag{4.3}$$

onde $\overline{p_i}$ é a média dos valores do vetor p_i. Calcule a matriz de similaridade $S^{(T)} = (s_{ij}^{(T)})$ tal que $s_{ij}^{(T)} = \frac{1+w_{ij}^{(T)}}{2}$. Então divida X em k clusters $C_1, \ldots, C_k$ utilizando o Algoritmo 5 com a matriz $S^{(T)}$ como entrada.

Perceba que utilizamos $s_{ij}^{(T)} = \frac{1+w_{ij}^{(T)}}{2}$, pois $w_{ij}^{(T)} \in [-1, 1]$, enquanto que na matriz de similaridade devemos considerar apenas valores não negativos, veja o Capítulo 3. Além disso, quanto maior é $s_{ij}^{(T)}$, maior é a correlação entre as ações x_i e x_j e existe uma maior a chance de eles estarem em um mesmo cluster. Isso é importante pois qualquer par de ações que ficar em um mesmo cluster não poderá estar simultaneamente no portfólio construído, já que iremos escolher uma única ação de cada cluster.

Uma vez construída a partição $\mathcal{C} = \{C_1, \ldots, C_k\}$ de X, escolha uma ação de cada cluster para compor o portfólio com base em um critério de *factor investing*. Propomos duas maneiras de fazer isso, que definem duas estratégias distintas. Na primeira selecione a ação de cada cluster com menor p/l positivo, mais exatamente, defina o portfólio $\mathcal{P}$ por

$$\mathcal{P} = \{x_j : \frac{p_j(t_f)}{e_j(T)} = \min_i \frac{p_i(t_f)}{e_i(T)}, \text{ onde } e_i(T) > 0, x_i \in C_\ell, \text{ para } \ell \in [k]\}.$$

A segunda maneira é fazer a seleção da ação de cada cluster a partir do ROE. Assim selecione a ação de cada cluster com maior ROE, mais exatamente,

defina o portfólio $\mathcal{P}$ por

$$\mathcal{P} = \{x_j : \frac{l_j(T)}{PL_j(t_f)} = \min_i \frac{l_i(T)}{PL_i(t_f)}, x_i \in C_\ell, \text{ para } \ell \in [k]\}.$$

Os critérios de *factor investing* utilizados nesse trabalho são considerados em períodos de doze meses. Por isso, cada portfólio montado por essa estratégia utiliza sempre um período de um ano e tem um prazo de validade de um ano. Se o objetivo for montar um portfólio mais longevo, basta ao final de cada período re-selecionar as ações do portfólio a partir da nossa estratégia. Portanto, vende-se as ações do portfólio antigo e utiliza-se o valor monetário para comprar o novo portfólio sugerido pela estratégia.

Na próxima subseção apresentamos os resultados da nossa estratégia em três anos consecutivos. Todos dados contábeis das empresas estão contidos em relatórios gerenciais disponibilizados por elas trimestralmente. Dados sobre o Ibovespa são disponibilizados publicamente. O site *yahoo finance*[6] reúne dados do mercado financeiro, onde qualquer pessoa pode acessá-los. Para esse trabalho realizamos a coleta de dados no yahoo finance. Os dados contábeis utilizados são: lucro líquido, patrimônio líquido, número de ações, no último dia de cada ano $m \in \{2017, 2018, 2019\}$. O preço diário de fechamento, de cada ação ou do Ibovespa, utilizado foi entre os anos 2017 e 2020, inclusive.

4.1.2 Resultados

Iremos definir os parâmetros de entrada da nossa estratégia. Consideramos X o conjunto de ações presentes no Ibovespa. Utilizamos $k = 10$ clusters e tomamos $T \in \{T_1, T_2, T_3\}$ três períodos diferentes. T_1 é o período entre 1 de jan. de 2017 e 31 de dez. de 2017, T_2 é o período entre 1 de jan. de 2018 e 31 de dez. de 2018 e T_3 é o período entre 1 de jan. de 2019 e 31 de dez. de 2019. $\mathcal{P}_1^{(T)}$ denota o port-

[6] `https://finance.yahoo.com/`. Acesso em: 4 fev. 2022.

fólio formado pela nossa estratégia utilizando o critério p/l. $\mathcal{P}_2^{(T)}$ denota o portfólio formado pela nossa estratégia utilizando o critério ROE. A Tabela 4.1 apresenta os clusters obtidos em $T = T_1$. A Tabela 4.2 e a Tabela 4.3 apresentam os clusters obtidos em $T = T_2$ e $T = T_3$, respectivamente.

Tabela 4.1: Clusters obtidos utilizando a estratégia da Subseção 4.1.1, a partir de dados dos preços das ações do ano de 2017.

T_1		Ações
C_1	=	{AZUL4, CVCB3, GOLL4, QUAL3, RAIL3, SBSP3}
C_2	=	{BBAS3, BBDC3, BBDC4, ITSA4, ITUB4, SANB11}
C_3	=	{BRAP4, CSNA3, GGBR4, GOAU4, USIM5, VALE3}
C_4	=	{BBSE3, CCRO3, CIEL3, ECOR3, EGIE3, ENBR3, EQTL3, TAEE11}
C_5	=	{BRML3, BTOW3, CYRE3, HYPE3, IGTA3, LAME4, LREN3, MRVE3, MULT3, NATU3, PCAR3}
C_6	=	{ABEV3, BRFS3, KLBN11, MRFG3, RADL3, SMLS3, TIMP3, UGPA3, WEGE3}
C_7	=	{CMIG4, ELET3, ELET6}
C_8	=	{BPAC11, BRDT3, EMBR3, IRBR3, RENT3, SUZB3}
C_9	=	{BRKM5, CSAN3, PETR3, PETR4}
C_{10}	=	{FLRY3, JBSS3, MGLU3, VIVT3, VVAR3}

Os portfólios $\mathcal{P}_1^{(T)}$ para $T \in \{T_1, T_2, T_3\}$ estão apresentados na Tabela 4.4. Os portfólios $\mathcal{P}_2^{(T)}$ para $T \in \{T_1, T_2, T_3\}$ estão apresentados na Tabela 4.5.

Para avaliar nossa estratégia comparamos o seu desempenho com o Ibovespa e três outras três estratégias. $\mathcal{P}_3$ é o portfólio que no início de cada ano escolhe as 10 ações com menor $p/l > 0$. $\mathcal{P}_4$ é o portfólio que no início de cada ano escolhe as 10 ações com maior ROE. $\mathcal{P}_5$ contém as mesmas ações do Ibovespa, porém todas as ações são consideradas com o mesmo peso. Como sempre utilizamos dados

Tabela 4.2: Clusters obtidos utilizando a estratégia da Subseção 4.1.1, a partir de dados dos preços das ações do ano de 2018.

T_2		Ações
C_1	=	{ABEV3, EQTL3, SBSP3, TIMS3, VIVT3}
C_2	=	{B3SA3,BRML3,FLRY3,HYPE3,IGTA3,LREN3,MULT3, PCAR3,RENT3}
C_3	=	{AZUL4,BTOW3,CMIG4,CVCB3,GOLL4,LAME4,MGLU3, RAIL3,SMLS3}
C_4	=	{BRAP4, CSNA3, GGBR4, GOAU4, USIM5, VALE3}
C_5	=	{BRKM5, EGIE3, IRBR3, KLBN11, SUZB3, WEGE3}
C_6	=	{BBDC3, BBDC4, BBSE3, ITSA4, ITUB4, MRVE3,SANB11}
C_7	=	{BRFS3, CCRO3, ECOR3, EMBR3, JBSS3, MRFG3, NATU3, UGPA3}
C_8	=	{BRDT3, PETR3, PETR4}
C_9	=	{BBAS3, CYRE3, ELET3, ELET6, QUAL3, VVAR3}
C_{10}	=	{BPAC11, CIEL3, CSAN3, ENBR3, GNDI3, RADL3, TAEE11}

de doze meses, o período considerado para calcular o desempenho será os doze meses seguintes. Assim, para $q \in \{1, 2, 3, 4, 5\}$ o desempenho de $\mathcal{P}_q^{(T_1)}$ é dado no intervalo de tempo 1 de jan. de 2018 até 31 de dez. de 2018. O mesmo vale para T_2 e T_3. Para medir o desempenho de uma estratégia ao longo dos três anos basta calcular $d_{\mathcal{P}_q} = d_{\mathcal{P}_q^{(T_1)}} \cdot d_{\mathcal{P}_q^{(T_2)}} \cdot d_{\mathcal{P}_q^{(T_3)}}$, para $q \in \{1, 2, 3, 4, 5\}$.

Na Tabela 4.6 realizamos uma comparação das rentabilidades dos portfólios. Na linha q, coluna j apresentamos o valor exato de $d_{\mathcal{P}_q^{(T_j)}}$. Para a última coluna imagine seis investidores distintos. Cada um deles possui um capital de R$ 10.000 no primeiro dia do ano de 2018. Cada um desses investidores comprará um dos seis portfólios, onde cada investidor possui um portfólio distinto. O valor da linha q da última coluna é exatamente o capital final do investidor que investiu no

Tabela 4.3: Clusters obtidos utilizando a estratégia da Subseção 4.1.1, a partir de dados dos preços das ações do ano de 2019.

T_3		Ações
C_1	=	{BRFS3, IRBR3, JBSS3, MRFG3, NTCO3, QUAL3}
C_2	=	{BRML3,BTOW3,CVCB3,IGTA3,LAME4,LREN3, MULT3,RAIL3,RENT3}
C_3	=	{BPAC11, BRDT3, CMIG4, CYRE3, ELET3, ELET6, ENBR3, GNDI3, MRVE3, SBSP3, TAEE11}
C_4	=	{BRKM5, EQTL3, KLBN11, MGLU3, SUZB3}
C_5	=	{AZUL4, COGN3, GOLL4, SMLS3, VVAR3, YDUQ3}
C_6	=	{PETR3, PETR4}
C_7	=	{BBAS3, BBDC3, BBDC4, ITSA4, ITUB4, SANB11}
C_8	=	{BRAP4, CSNA3, GGBR4, GOAU4, USIM5, VALE3}
C_9	=	{ABEV3, BBSE3, CCRO3, CIEL3, ECOR3, FLRY3, HYPE3, UGPA3}
C_{10}	=	{B3SA3, CSAN3, EGIE3, EMBR3, PCAR3, RADL3, TIMS3, VIVT3, WEGE3}

portfólio $\mathcal{P}_q$. Também apresentamos a Figura 4.1 com o desempenho diário de cada um desses investires.

No período analisado, a nossa estratégia gerou portfólios com desempenhos melhores que os demais. Com os resultados parciais que obtivemos, acreditamos que a nossa estratégia é muito promissora e que é uma boa alternativa para o investidor. Além de seu bom desempenho comparado a outros portfólios, essa estratégia ocupa pouco tempo de análise do investidor, visto que utiliza apenas dados quantitativos em sua análise. Ainda planejamos estender nossa estratégia para outros anos e outros critérios possíveis, com o objetivo de obter um melhor nível de segurança.

Tabela 4.4: Portfólios construídos a partir da primeira estratégia apresentada na Subseção 4.1.1.

Portfólio		Ações
$\mathcal{P}_1^{(T_1)}$	=	{SBSP3, BBAS3, BRAP4, EGIE3, MRVE3, SMLS3, CMIG4, IRBR3, BRKM5, VIVT3}
$\mathcal{P}_1^{(T_2)}$	=	{VIVT3, BRML3, SMLS3, CSNA3, EGIE3, MRVE3, MRFG3, BRDT3, ELET3, ENBR3}
$\mathcal{P}_1^{(T_3)}$	=	{JBSS3, BRML3, ELET3, EQTL3, SMLS3, PETR4, BBAS3, CSNA3, BBSE3, TIMS3}

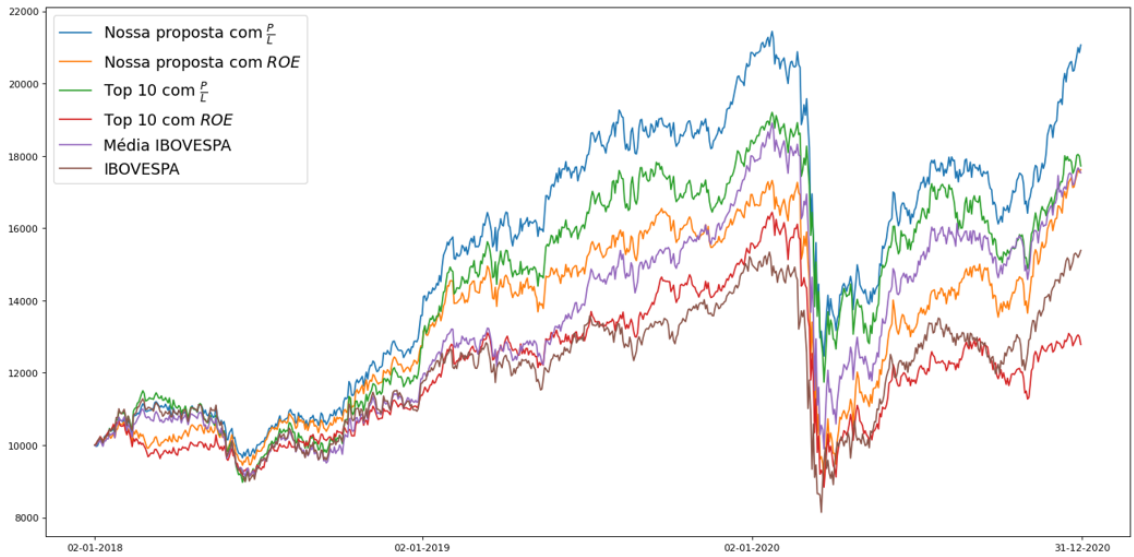

Figura 4.1: Gráfico diário contendo o desempenho dos portfólios definidos na Subseção 4.1.2, ao longo dos três anos, iniciando com um capital de R$10.000,00.

4.2 Classificação de risco em redes complexas: o caso da COVID-19 no Rio Grande do Sul

Nessa seção buscamos aplicar uma metodologia desenvolvida por Peixoto *et al.* [50] no estado do Rio Grande do Sul, com o objetivo de obter uma classificação de risco do estado, avaliando a qualidade dessa classificação.

Tabela 4.5: Portfólios construídos a partir da segunda estratégia apresentada na Subseção 4.1.1.

Portfólio		Ações
$\mathcal{P}_2^{(T_1)}$	=	{CVCB3, ITUB4, BRAP4, ECOR3, NTCO3, SMLS3, CMIG4, IRBR3, BRKM5, FLRY3}
$\mathcal{P}_2^{(T_2)}$	=	{ABEV3, LREN3, SMLS3, CSNA3, BRKM5, BBSE3, MRFG3, BRDT3, ELET3, CIEL3}
$\mathcal{P}_2^{(T_3)}$	=	{MRFG3, LREN3, BRDT3, EQTL3, AZUL4, PETR3, ITUB4, CSNA3, BBSE3, EGIE3}

Tabela 4.6: Comparação das Estratégias.

Estratégia	T_1	T_2	T_3	Capital Final
$d_{\mathcal{P}_1}$	35,82%	53,53%	1%	R$21.064,37
$d_{\mathcal{P}_2}$	26,98%	32,83%	4,06%	R$17.554,15
Ibovespa	17,04%	29,96%	-0,14%	R$15.191,74
$d_{\mathcal{P}_3}$	18,61%	50,12%	0%	R$17.822,99
$d_{\mathcal{P}_4}$	26,89%	45,18%	-0,33%	R$17.827,25
$d_{\mathcal{P}_5}$	13,32%	36,93%	-17,96%	R$12.732,06

O ano de 2020 foi marcado pelo surto global do vírus SARS-CoV-2, que causa a doença COVID-19 em humanos. Em março daquele ano, a Organização Mundial da Saúde (OMS) declarou a COVID-19 como pandemia mundial. A característica principal, e mais preocupante, dessa doença é o fato de ser altamente contagiosa, o que torna difícil barrar a sua propagação e levou governos mundo afora a tomar diversas medidas, desde o fechamento obrigatório de escolas e atividades econômicas até a proibição de viagens entre países. A COVID-19 não demorou muito para chegar, e se espalhar, no Brasil e logo em março de 2020 já havia transmissão

comunitária no país [45]. Reconhecendo a gravidade da doença e o desconhecimento sobre seu prognóstico e virulência, muitos cientistas e autoridades de saúde empregaram métodos epidemiológicos para entender o seu comportamento. Os modelos epidemiológicos são amplamente utilizados e há uma vasta literatura relacionada a eles, como por exemplo modelos compartimentais [11]. No enfrentamento da COVID-19 os modelos compartimentais também foram de suma importância para compreender a transmissão da doença. como em [39], [64] e [88]. Nessas abordagens, um dos objetivos é entender como a doença se propagaria em uma região no momento em que os primeiros casos são identificados com base em diversos parâmetros hipotéticos de transmissão e em diversos cenários de redução de mobilidade.

Nessa linha, Peixoto *et al.* [50] investigaram como dados de mobilidade populacional podem ser empregados para construir uma classificação de risco relacionada à COVID-19. Eles propuseram uma metodologia que foi aplicada para os municípios dos estados de São Paulo e Rio Janeiro, obtendo uma classificação de risco para esses municípios em três categorias, denominadas risco baixo, médio ou alto. No contexto dessa metodologia, a palavra risco está associada ao tempo esperado até que os primeiros casos sejam identificados em um município, isto é, quanto maior o risco, mais rápido a doença tenderá a atingir o município.

Essa seção busca aplicar essa abordagem a mais um estado, além de avaliar a qualidade da predição nesse caso específico e investigar aspectos que fundamentam a metodologia [50].

4.2.1 Metodologia

A seguir apresentamos essa metodologia, que se baseia em um modelo epidemiológico Suscetível - Infectado (SI), ao qual são incorporados dados de mobilidade entre municípios. O modelo SI é uma forma simplificada de descrever a transmissão inicial de doenças em populações e considera que todos os indivíduos

podem ser classificados em duas categorias, ou compartimentos: aqueles que ainda não contraíram a doença e permanecem vulneráveis (suscetíveis) e aqueles que contraíram a doença e têm a capacidade de transmiti-la (infectados). Modelos SI não preveem que haja recuperação ou mortalidade, não sendo adequados para previsões de longo prazo. Nesse sentido, tais modelos são indicados para capturar a evolução inicial de uma doença, particularmente quando poucos dados sobre ela são conhecidos, o que é compatível com os objetivos desse trabalho.

A propagação da doença será modelada através de relações de recorrência com duas componentes, uma em que o contágio acontece entre indivíduos de um mesmo município i e outro que acontece entre indivíduos de municípios i e j diferentes. Dentro de cada município i os novos casos são obtidos por um modelo SI tradicional, enquanto que entre os municípios i e j são incorporados os dados de mobilidade. No modelo, o número de infectados no dia $t+1$ no município i é dado por

$$I_i(t+1) = I_i(t) + (1+r)I_i(t)\left(\frac{N_i - I_i(t)}{N_i}\right) + s\Bigg[\sum_{i\neq j}\omega_{ji}(t)I_j(t) - \sum_{i\neq j}\omega_{ij}(t)I_i(t)\Bigg], \tag{4.4}$$

onde r é a taxa de transmissão da doença, s é um parâmetro de correção para os dados de mobilidade, $I_i(t)$ é a quantidade de infectados no dia t e N_i é a população do município. O modelo considera que a taxa de transmissão da doença e a população de cada município independem do tempo. O termo $\omega_{ij}(t)$ denota a proporção da população de i que se desloca para j no dia t, isto é, a razão $\omega_{ij}(t) = \frac{W_{ij}(t)}{N_i}$ onde $W(t) = (W_{ij}(t))$ é a matriz cuja entrada ij representa a quantidade de pessoas que se deslocaram do município i para o município j no dia t. No contexto de [50], os dados que compõem essa matriz são dados anônimos de mobilidade baseados na geolocalização de telefone celulares, disponibilizados pela empresa In Loco, hoje

incorporada pela Incognia[7]. O papel de s é corrigir eventuais imprecisões nos dados de mobilidade utilizados. Por um lado, como os dados são anônimos, se uma pessoa faz várias viagens em um mesmo dia, essas viagens são contadas separadamente, e neste caso valores de $s < 1$ suporiam que a estimativa está acima do número verdadeiro de viagens. Por outro lado, existem viagens que não são capturadas pelo sistema da In Loco, e assim valores de $s > 1$ suporiam que a estimativa está abaixo do número verdadeiro de viagens. Finalmente, $s = 1$ significaria que os dados retratam o que de fato aconteceu. Ainda vale notar que, nesse modelo, as variáveis $I_i(t)$ podem assumir valores não inteiros.

Para dividir os municípios no conjunto $M = \{m_1, m_2, \ldots, m_n\}$ em três grupos de risco, consideram-se diferentes valores de s e, para cada um desses valores, determina-se a evolução da doença aplicando a recorrência (4.4) com um caso inicial na capital do estado. Cada município i é associado a um vetor v_i que, para cada valor de s considerado, identifica o primeiro dia em que o município i contabiliza pelo menos um caso, isto é, $\min\{t : I_i(t) \geq 1\}$. Esses vetores são divididos em três grupos de risco a partir de um algoritmo de clusterização. Ao final, o risco atribuído a cada município vem do grupo que contém o vetor associado a ele.

O algoritmo para a obter a classificação de risco é apresentado a seguir.

[7]Disponível em: `https://www.incognia.com/pt/`. Acesso em: 24 jan. 2022.

Algoritmo 6: Classificação de Risco

Entrada: Conjuntos $M = \{m_1, m_2, \ldots, m_n\}$ e $N = \{N_{m_1}, \ldots, N_{m_n}\}$. Vetor $\mathbf{s} = (s_1, s_2, \ldots, s_K)$ de números positivos. Inteiros positivos T e Q, constante real $r \geq 0$. Matrizes $W(t)$ de dimensão $n \times n$, para $1 \leq t \leq T$.

Saída: Partição $\mathcal{P} = \{A, B, C\}$ de M, onde os municípios em A são classificados como risco alto; em B, risco médio; e em C, risco baixo.

1 Defina um vetor

$$v_i = (v_i^1, \ldots, v_i^K) \in \mathbb{R}^K \tag{4.5}$$

com o valor $T + 1$ em todas entradas, para cada $i \in M$.

2 **para** $\ell \in (1, \ldots, K)$ **faça**

3 Atribua $I_{m_1}^{s_\ell}(0) = 1$ e $I_{m_i}^{s_\ell}(0) = 0, \forall i \in \{2, \ldots, n\}$

4 **para** $0 \leq t \leq T - 1$ **faça**

5 Para cada $i \in M$, defina $I_i^{s_\ell}(t + 1)$, por meio da equação de recorrência (4.4).

6 Para cada $i \in M$, se $I_i^{s_\ell}(t + 1) \geq 1$ e $v_i^\ell = T + 1$, redefina $v_i^\ell = t + 1$.

7 **fim**

8 **fim**

9 Separe o conjunto $V = \{v_1, v_2, \ldots, v_n\}$ em três conjuntos de risco C_A, C_B, C_C (alto, médio e baixo, respectivamente) via Algoritmo k-means.

10 Seja $\mathcal{C}$ a partição tal que $i \in M$ é atribuído ao cluster γ se, e somente se, v_i encontra-se em C_γ.

11 **Retorne** $\mathcal{C}$

No trabalho de [50] foram utilizados os seguintes parâmetros. Os conjuntos M e N foram constituídos pelos municípios de cada estado e suas populações, respectivamente. As matrizes $W(t)$ continham os dados de mobilidade. Para o valor da taxa de transmissão foi utilizado $r = 0,4$, que é aproximadamente $\frac{R_0}{6}$, onde $R_0 = 2,68$ é o número efetivo de reprodução da COVID-19, calculado em meados de 2020 com base nos dados sobre a doença em Wuhan, China, e 6 é o tempo médio,

em dias, de incubação da doença, conforme [86]. Além disso, foi utilizado o vetor

$$S = (0.001, 0.005, 0.1, 0.2, 0.4, 0.5, 0.6, 0.8, 1, 1.2, 1.4, 1.6, 1.8, 2, 2.5, 3). \quad (4.6)$$

4.2.2 Dados

Nessa aplicação foram considerados os 167 municípios do Rio Grande do Sul, onde a população estimada de 2019 é maior que 10.000 de acordo com o Instituto Brasileiro de Geografia e Estatística (IBGE) [8]. Os municípios do estado que satisfazem esse critério estão marcados no mapa da Figura 4.2. Em nossa aplicação, o valor de N_i é exatamente essa população estimada para o município i.

Assim como em [50] os dados de mobilidade social utilizados nessa aplicação foram de deslocamento baseados na posição geográfica de telefones celulares em todo o Brasil. Esses dados de mobilidade foram disponibilizados pela empresa In Loco, que coletou dados anônimos de localização de usuários de telefones celulares que compõem uma base de dados. No sistema da empresa, é considerado que uma pessoa se deslocou dentro de seu município caso haja pelo menos pelo menos dois registros dela em pontos distantes de pelo menos 450 metros e que uma pessoa se deslocou do município i para o município j ($i \neq j$) se há registros da mesma em i e j, nesta ordem. Esses dados são utilizados para representar o número de deslocamentos de um município i para um outro município j no dia t, com $i, j \in \{m_1, \ldots, m_n\}$. Precisamente, construímos a matriz $W(t) = (W_{ij}(t))$ de ordem n, onde $W_{ij}(t)$ é o número de pessoas que se deslocaram do município i para o município j no dia t segundo esses dados.

Como o objetivo do modelo é prever uma classificação de risco para a dispersão inicial da doença, foram utilizados dados de mobilidade em uma época de

[8] Disponível em: `https://www.ibge.gov.br/estatisticas/sociais/populacao`. Acesso em: 23 jan. 2022.

normalidade, isto é, em que não havia redução de mobilidade por conta de medidas voluntárias ou obrigatórias de isolamento social. Portanto, para aplicar esse modelo, utilizamos dados do período anterior a quaisquer medidas de isolamento. Além disso, como o critério de classificação está baseado na ocorrência do primeiro caso, foi necessário escolher um período suficientemente longo para que o modelo atribuísse um primeiro caso a cada município. No nosso caso, foram necessários $T = 38$ dias para que isso ocorresse. Em virtude disso, utilizamos dados dos 38 dias entre 1º de fevereiro e 14 de março de 2020, que foi o dia em que o isolamento voluntário foi encorajado e os dados de mobilidade mostraram alterações por conta do isolamento social.

No período mencionado acima, a média dos registros do número de deslocamentos diários, no Rio Grande do Sul, foi de aproximadamente 577 mil, que corresponde a aproximadamente $5,8\%$ da população do estado. A partir desses dados, pode-se identificar que ocorreram deslocamentos em todos os n municípios, com cobertura relativamente maior na região metropolitana, no litoral norte e na serra. Se considerarmos a proporção do total de registros de pessoas que se deslocaram em cada município pela população do município, a proporção das regiões metropolitana, litoral e serra, em conjunto, ficou aproximadamente $7,1\%$, enquanto que nos demais municípios foi de $3,7\%$. Com exceção de três municípios, todos tiveram pelo menos a proporção de $1,0\%$ de registros de deslocamentos diários em relação à população. Para efeitos de comparação, os dados utilizados por [50] resultaram em uma média diária de registros de deslocamento de $4,3$ milhões para o estado de São Paulo e de 800 mil para o estado do Rio de Janeiro, o que corresponde a aproximadamente $9,5\%$ e $4,8\%$ da população de cada estado, respectivamente.

Para realizar a comparação entre o risco atribuído pelo modelo e o momento em que o primeiro caso foi registrado em cada município, utilizamos os registros oficiais de casos mantidos pela Secretaria da Saúde do Estado do Rio Grande do Sul [63].

4.2.3 Resultados

Essa subseção tem como objetivo apresentar a classificação de risco obtida para o estado do Rio Grande do Sul e compará-la com as classificações obtidas para outros estados, assim como com a evolução da COVID-19 segundo os dados oficiais. Por fim, temos o objetivo de comparar a técnica de particionamento descrita na Subseção 4.2.1 com uma abordagem espectral.

Para realizar o primeiro objetivo, que é definir a classificação de risco para os municípios do Rio Grande do Sul com base na metodologia da Subseção 4.2.1, aplicamos o Algoritmo 6 para os parâmetros epidemiológicos utilizados em [50] e os dados de população e deslocamento descritos na Subseção 4.2.2. Obtivemos a classificação de risco ilustrada na Figura 4.2, utilizando $Q = 500$.

Para avaliar como a classificação de risco obtida se relacionou com os dados oficiais sobre a pandemia, definimos um índice que é dado pela proporção de pares (i, j) de municípios, onde o município i, para o qual o modelo atribuiu risco maior, registrou o seu primeiro caso oficial em um tempo inferior ou igual do município j. Primeiro, seja $d = (d_1, \ldots, d_n)$ vetor, tal que d_i é o número de dias entre a data do primeiro caso oficial no estado e o dia que i registrou seu primeiro caso oficial. Definimos a função $f : M \to \{1, 2, 3\}$, onde

$$f(i) = \begin{cases} 3, & \text{se } i \in A \\ 2, & \text{se } i \in B \\ 1, & \text{se } i \in C \end{cases}$$

e A, B, C são os clusters de risco alto, médio e baixo, respectivamente. Sejam $C^{RS} = \{(i, j) \in M \times M | f(i) > f(j) \text{ e } d_i \leq d_j\}$ e $T^{RS} = \{(i, j) \in M \times M | f(i) > f(j)\}$. Definimos P^{RS} da seguinte forma:

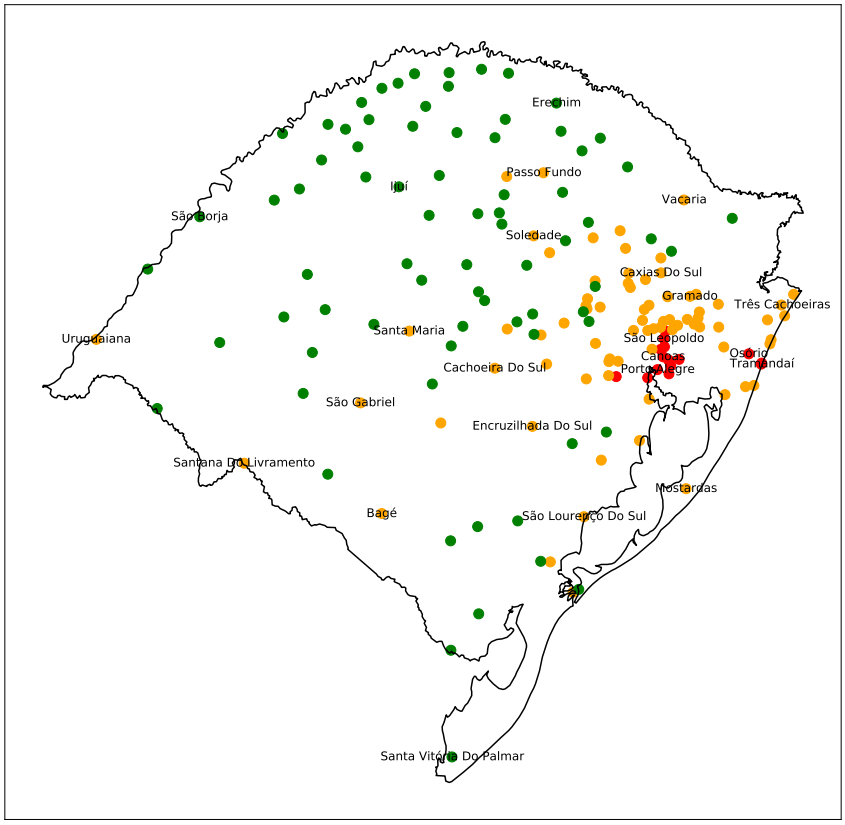

Figura 4.2: Divisão dos $n = 167$ municípios do Rio Grande do Sul em três regiões: risco alto (vermelho), risco médio (laranja) e risco baixo (verde), aplicando a metodologia da Subseção 4.2.1 aos dados da Subseção 4.2.2.

$$P^{RS} = \frac{|C^{RS}|}{|T^{RS}|}.$$

Note que, quando P^{RS} está mais próximo de 1, maior é a quantidade de pares de municípios em que o modelo atribuiu risco maior aos municípios que registraram os primeiros casos da doença. Realizando os cálculos para a classificação de risco apresentada na Figura 4.2 foi obtido $P^{RS} = 0,7521$. Também é possível definir o índice restrito a cada duas classificações de risco, dessa forma dados $X \neq Y \in \{A, B, C\}$, sejam $C^{RS}_{X,Y} = \{(i,j) \in X \times Y | f(i) > f(j) \text{ e } d_i \leq d_j\}$ e $T^{RS}_{X,Y} = \{(i,j) \in X \times Y | f(i) > f(j)\}$. Definimos $P^{RS}_{X,Y}$ como a razão

$$P_{X,Y}^{RS} = \frac{|C_{X,Y}^{RS}|}{|T_{X,Y}^{RS}|}.$$

Assim, $P_{A,B}^{RS}$ é a proporção de pares (i, j) de municípios onde além do modelo atribuir risco alto a i e médio a j, i registrou o primeiro caso oficial antes de j. Nós obtivemos $P_{A,B}^{RS} = 0,7662$, $P_{A,C}^{RS} = 0,9248$ e $P_{B,C}^{RS} = 0,7182$. Esses resultados revelam uma correlação positiva entre o risco atribuído ao município e o tempo que ele levou para se contaminar.

Uma segunda comparação que fizemos foi relacionar o valor de v_i^ℓ (definido na equação (4.5)) para municípios com mais de 50 mil habitantes para alguns valores do vetor S, definido na equação (4.6), e o tempo que levaram para confirmar um caso pelos dados oficiais, dado pelo vetor d. Note que como não havia consenso científico sobre o valor de R_0 e devido às limitações do modelo utilizado, estamos novamente interessados na ordem relativa entre os municípios com relação ao risco atribuído e a data de registro do primeiro caso, e não aos valores específicos de v_i^ℓ. Essa comparação pode ser vista no gráfico na Figura 4.3. Nós utilizamos o método dos mínimos quadrados (veja [30]) para investigar se os valores v_i^ℓ ficaram proporcionais aos valores d_i.

Mais precisamente, considerando $M_0 = \{i \in M | N_i \geq 50.000\}$ seja $u = (u_1, \ldots, u_{|M_0|})$ o vetor tal que $u_j = \frac{1}{K}\sum_{\ell=1}^{K} v_{m_j}^\ell$, para cada $m_j \in M_0$, e o ordenamento $(r_1, \ldots, r_{|M_0|})$ definido recursivamente da seguinte forma:

i) $r_1 = 1$. Redefine $u_{r_1} = T + 1$.

ii) $r_\ell = argmin\{u_i | m_i \in M_0\}$. Se existe $j \neq i$ tal que $u_i = u_j$ é escolhido i para o qual N_{m_i} é máximo. Redefine $u_{r_\ell} = T + 1$.

Considerando o conjunto de dados $\mathbb{X} = \{(r_i, d_{m_i}) | m_i \in M_0\}$, a reta que melhor aproxima os dados de $\mathbb{X}$ pelo método dos mínimos quadrados, em verde na

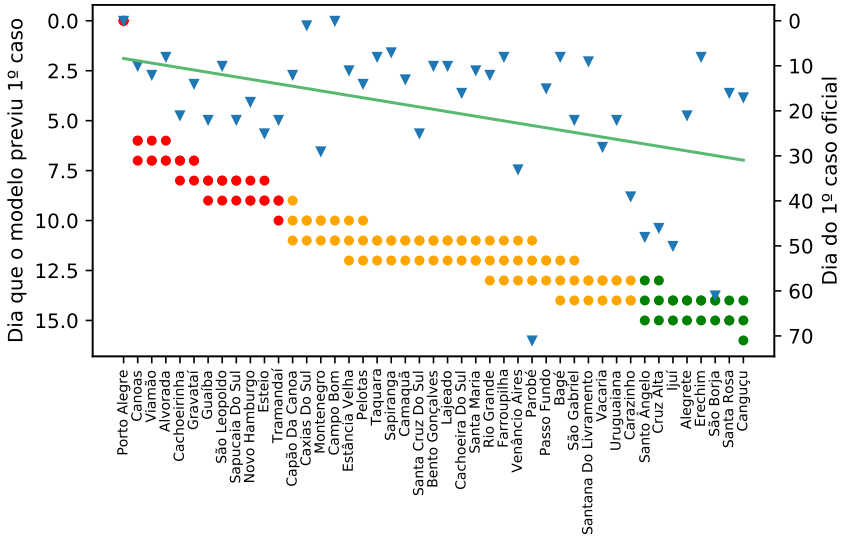

Figura 4.3: Os pontos se referem aos valores de v_i^ℓ para ℓ tal que $s_\ell \in \{0.5, 1, 1.6\}$, isto é, o número de dias até que o modelo previsse o primeiro caso no município i para os três valores de s_ℓ, os triângulos se referem ao dia d_i em que o município i registrou o primeiro caso oficial de COVID-19. A reta é uma aproximação dos triângulos por meio do método dos mínimos quadrados. As cores se referem ao nível de risco do município de acordo com a modelagem. Consideramos apenas os municípios com mais de 50 mil habitantes.

Figura 4.3. Isso reforça a hipótese de que a evolução da doença está relacionada com o nível de risco atribuído pelo modelo.

Para efeito de comparação, calculamos os índices correspondentes para as classificações de risco obtidas para os estados de São Paulo e Rio de Janeiro por [50], que podem ser vistas na Figura 4.4 e na Figura 4.5, respectivamente.

No cálculo dos índices P e $P_{\alpha,\beta}$ para São Paulo, o resultado obtido foi de $P^{SP} = 0,8562$, $P^{SP}_{A,B} = 0,8535$, $P^{SP}_{A,C} = 0,9750$ e $P^{SP}_{B,C} = 0,8032$. Já para o Rio de Janeiro, os valores foram de $P^{RJ} = 0,8536$, $P^{RJ}_{A,B} = 0,7764$, $P^{RJ}_{A,C} = 0,9898$ e $P^{RJ}_{B,C} = 0,8359$. Primeiramente, notamos que, assim como para o Rio Grande do Sul, os índices sugerem que a evolução inicial da doença está relacionada com a classificação de risco obtida pelo modelo.

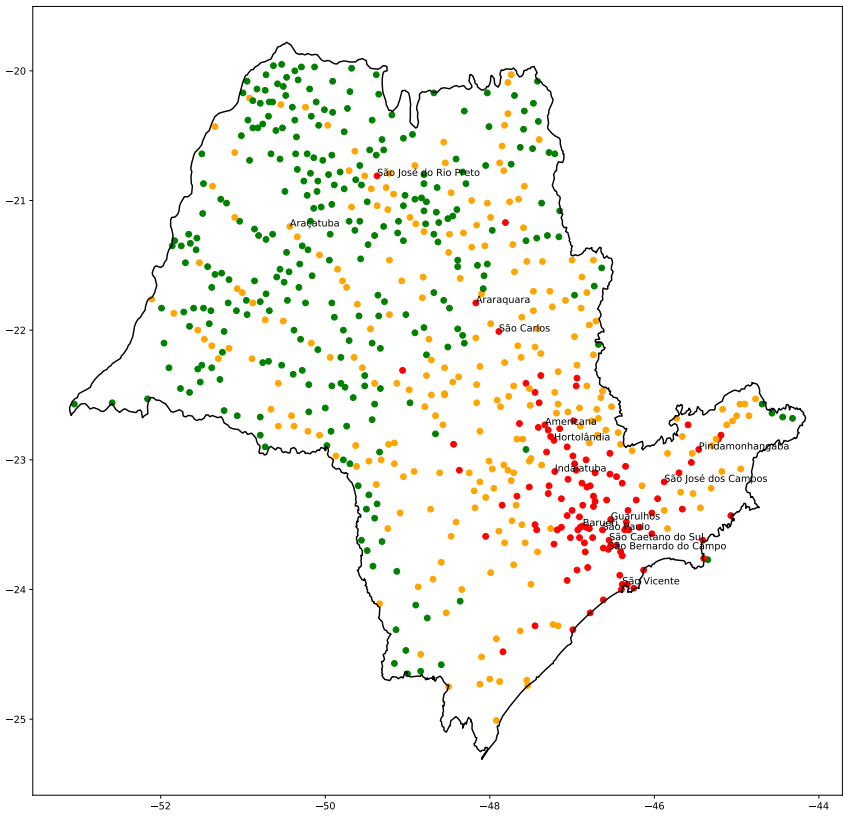

Figura 4.4: Divisão dos municípios de São Paulo em três regiões: risco alto (vermelho), risco médio (laranja) e risco baixo (verde), obtida em [50].

Quando se compara os valores de SP e RJ com os do RS, nota-se que os números deste último foram relativamente mais baixos. Levantamos dois fatores que podem ter influenciado nisso. O primeiro ponto é que, em nossa abordagem, consideramos apenas os municípios com mais de 10 mil habitantes, enquanto que as avaliações dos outros dois estados foram feitas para todos os municípios. A presença de municípios pequenos, tipicamente classificados com baixo risco, pode ter influenciado positivamente no valor dos índices. O segundo ponto a se destacar é que apenas quatro municípios do Rio Grande do Sul (Porto Alegre, Caxias do Sul, Campo Bom e Sapiranga) tiveram o primeiro registro de caso oficial antes do dia 19 de março, quando o governo do estado instituiu medidas obrigatórias de isolamento social, como o fechamento de escolas e estabelecimentos comerciais ([60]). Isso ocasi-

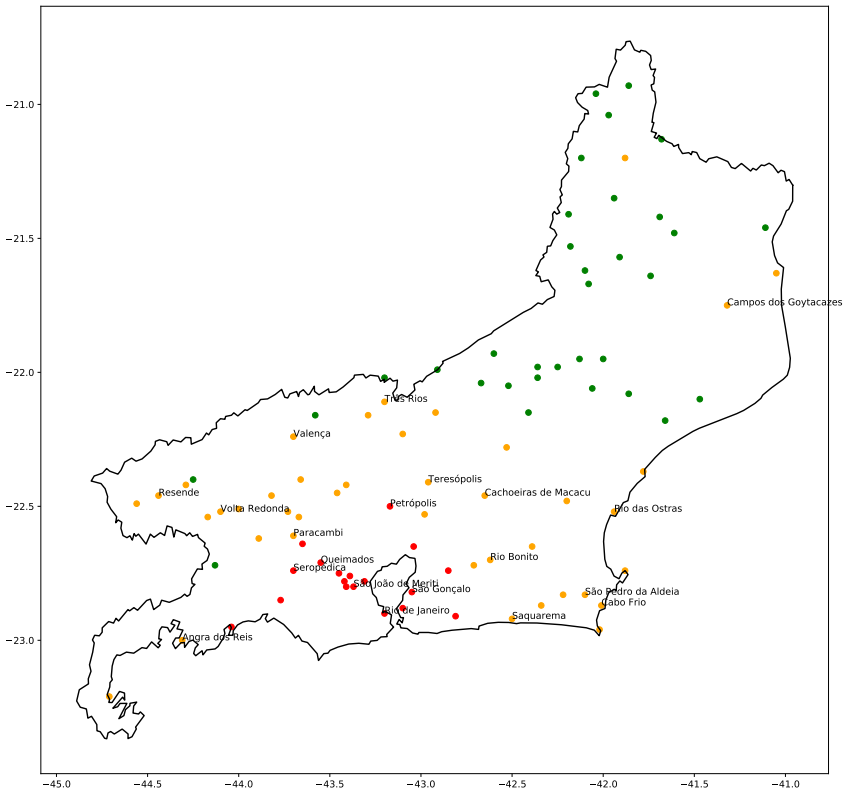

Figura 4.5: Divisão dos municípios do Rio de Janeiro em três regiões: risco alto (vermelho), risco médio (laranja) e risco baixo (verde), obtida em [50].

onou uma mudança de mobilidade, que não foi considerada no modelo. Nos estados de São Paulo e Rio de Janeiro, a COVID-19 se espalhou de forma mais ampla antes que medidas sanitárias fossem colocadas em prática, de forma que os dados de mobilidade podem ter sido mais condizentes com o padrão de deslocamentos até a identificação dos primeiros casos.

Quanto a comparação com a reta obtida pelo método dos mínimos quadrados para São Paulo e Rio de Janeiro, relacionada aos valores v_i^ℓ, percebemos comportamentos semelhantes ao observado para o Rio Grande do Sul. O caso de São Paulo, para os municípios com mais de 150 mil habitantes, está retratado na Figura 4.6 e o caso do Rio de Janeiro, para os municípios com mais de 50 mil habitantes, está ilustrado na Figura 4.7.

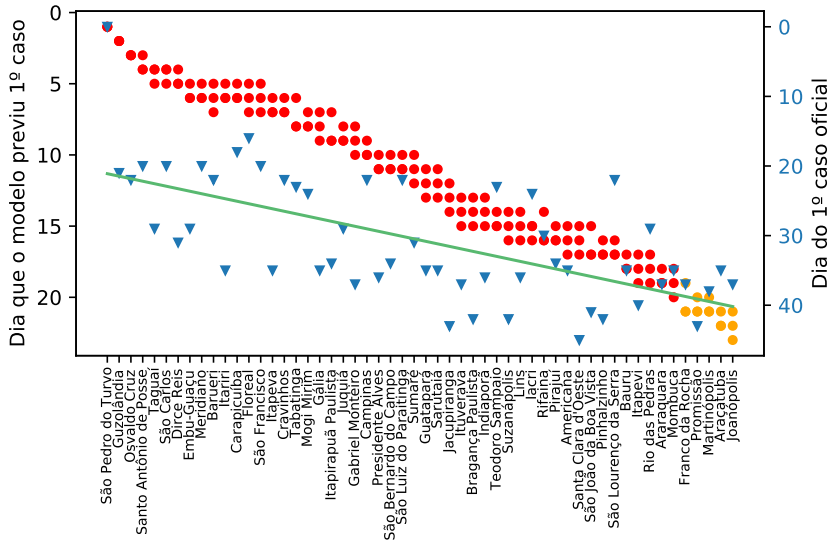

Figura 4.6: Os pontos se referem aos valores de v_i^ℓ para ℓ tal que $s_\ell \in \{0.5, 1, 1.6\}$, isto é, o número de dias até que o modelo previsse o primeiro caso no município i para os três valores de s_ℓ, os triângulos se referem ao dia d_i em que o município i registrou o primeiro caso oficial de COVID-19. A reta é uma aproximação dos triângulos por meio do método dos mínimos quadrados. As cores se referem ao nível de risco do município de acordo com a modelagem. Consideramos apenas os municípios com mais de 150 mil habitantes.

De um ponto de vista qualitativo, podemos perceber que no Rio Grande do Sul a região de maior risco foi a região metropolitana de Porto Alegre (região que reúne 34 municípios do estado) junto com dois municípios mais distantes: Osório e Tramandaí. Vale notar que Osório não é um município muito populoso (é apenas o 46º mais populoso do estado), mas possui um fluxo elevado de pessoas, visto que é um ponto de ligação entre o litoral gaúcho e as outras regiões do estado, além de ser um ponto de passagem da maior estrada que liga os municípios da região metropolitana com os outros estados do Brasil. Tramandaí é um município litorâneo mais distante de Porto Alegre, porém apresentou um alto fluxo de pessoas com Porto Alegre e Osório. Apesar de ser o menos povoado dentre os municípios de alto risco, Eldorado do Sul (52º mais populoso do estado), que está no limite da região metropolitana de Porto Alegre, apresentou um alto fluxo com a região metropolitana de Porto Alegre,

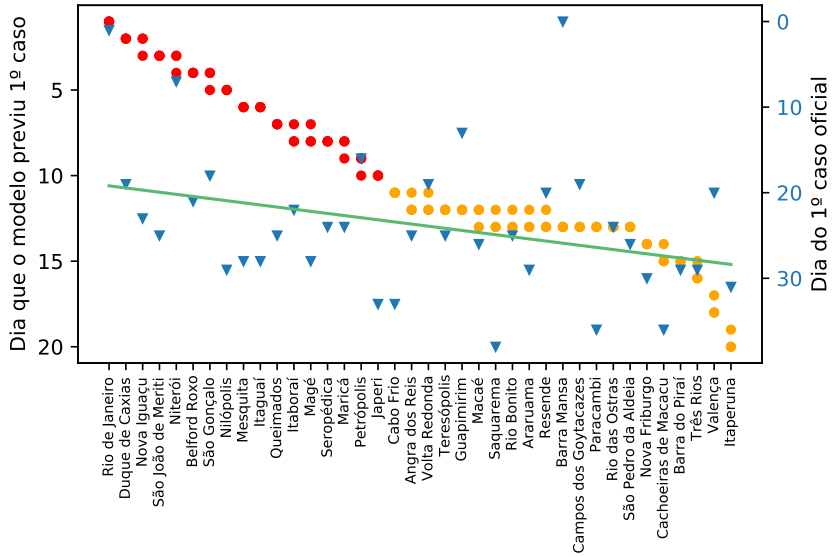

Figura 4.7: Os pontos se referem aos valores de v_i^ℓ para ℓ tal que $s_\ell \in \{0.5, 1, 1.6\}$, isto é, o número de dias até que o modelo previsse o primeiro caso no município i para os três valores de s_ℓ, os triângulos se referem ao dia d_i em que o município i registrou o primeiro caso oficial de COVID-19. A reta é uma aproximação dos triângulos por meio do método dos mínimos quadrados. As cores se referem ao nível de risco do município de acordo com a modelagem. Consideramos apenas os municípios com mais de 50 mil habitantes.

o que justifica o município ter ficado em uma região de alto risco. Ainda de um ponto de vista qualitativo percebe-se que alguns municípios importantes do Rio Grande do Sul, por exemplo Caxias do Sul, Pelotas e Santa Maria, não ficaram em uma região de alto risco, diferentemente do que aconteceu no estado de São Paulo retratado na Figura 4.4, onde municípios importantes, mas distantes da capital, também ficaram em um grupo de alto risco.

O último passo do Algoritmo 6 é um exemplo de problema de clusterização, onde queremos agrupar n vetores em $k = 3$ clusters, que foi resolvido via Algoritmo k-means. Como já mencionamos na Subseção **??**, o Algoritmo k-means possui algumas limitações, como por exemplo a dependência das condições iniciais

e a convexidade dos clusters produzidos. Assim é natural investigar alternativas que possam levar a clusterizações melhores, como utilizar a abordagem espectral.

Em nossa abordagem utilizamos o conjunto de dados $X = \{v_1, \ldots, v_n\}$, onde v_i são os vetores obtidos pelo Algoritmo 6, para $i \in \{1, \ldots n\}$, $k = 3$ e $\sigma = 1/\sqrt{2}$ como entrada do Algoritmo 4.

Utilizando a abordagem espectral para os nossos dados, obtivemos a classificação de risco retratada na Figura 4.8. Essa classificação é muito similar à classificação obtida na Figura 4.2, apenas cinco municípios tiveram sua classificação alterada. Os municípios de Osório e Tramandaí foram classificados com risco médio na abordagem espectral, enquanto que foram classificados com risco alto pelo Algoritmo 6, que utiliza Algoritmo k-means. Já os municípios de Guaporé, Veranópolis e Candelária, que anteriormente foram classificados com risco médio, foram classificados com risco baixo nessa nova classificação.

Com base na Seção 3.3, uma escolha possível para o número de clusters é aquela que maximiza o *eigengap*. O *eigengap* máximo encontrado em nossa aplicação foi entre os autovalores λ_4 e λ_3, onde a razão entre eles foi consideravelmente maior que a razão entre quaisquer outros dois autovalores consecutivos. Isso sugere que $k = 3$ é a melhor escolha para a quantidade de clusters que dividem o estado. O gráfico presente na Figura 4.9 ilustra a sequência das dez primeiras razões entre autovalores consecutivos, evidenciando que $k = 3$ é a melhor escolha. Os valores dos dez primeiros autovalores foram $\lambda_1 = 0, \lambda_2 = 0,0133, \lambda_3 = 0,0201, \lambda_4 = 0,0473, \lambda_5 = 0,0581, \lambda_6 = 0,0875, \lambda_7 = 0,1353, \lambda_8 = 0,1674, \lambda_9 = 0,1916, \lambda_{10} = 0,2989$.

Vale mencionar que, utilizando $Q = 500$, todas partições originadas pelo algoritmo espectral foram idênticas. Ainda nesse aspecto, na utilização do Algoritmo k-means diretamente no conjunto de vetores $\{v_1, \ldots, v_n\}$, com $Q = 500$, a partição de saída mais frequente foi obtida em apenas 180 vezes. Dessa forma, na nossa aplicação, a clusterização espectral se revelou mais estável que o Algoritmo k-means.

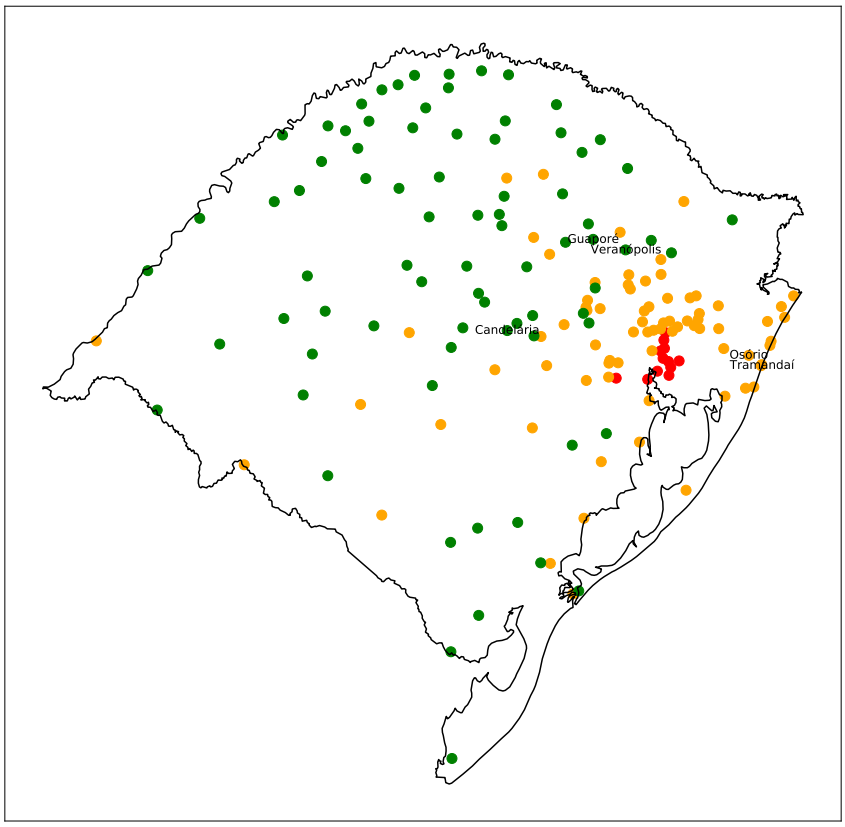

Figura 4.8: Divisão dos $n = 167$ municípios do Rio Grande do Sul em três regiões: risco alto (vermelho), risco médio (laranja) e risco baixo (verde), de acordo com a metodologia apresentada nessa subseção. Os municípios destacados na figura são aqueles nos quais houve alguma mudança frente a abordagem com utilização do Algoritmo k-means apresentada na Subseção 4.2.1.

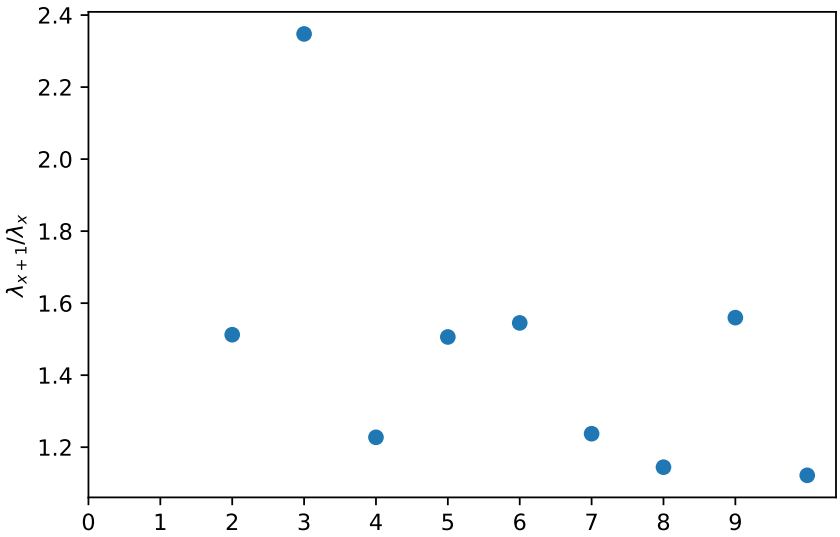

Figura 4.9: Gráfico da sequência das nove primeiras razões entre os autovalores.

5 MEDIDA DE SIMILARIDADE HIERÁRQUICA PARA CLUSTERIZAÇÃO ESPECTRAL

A medida de similaridade possui um papel fundamental no desempenho dos métodos de clusterização espectral. É a partir da similaridade entre cada par de objetos de um conjunto de dados que o método realiza a classificação.

Na Seção 3.3 apontamos que um dos problemas de pesquisa em clusterização espectral é justamente como definir a medida de similaridade. Esse capítulo se dedica a aprofundar esse aspecto. Nosso objetivo principal é apresentar uma nova medida de similaridade derivada do kernel Gaussiano, que por sua vez define um novo algoritmo espectral. Iremos comparar o desempenho desse algoritmo espectral com outros algoritmos cuja medida de similaridade também é baseada no kernel Gaussiano.

5.1 A Medida de Similaridade

O desempenho dos algoritmos espectrais é bem dependente da definição da medida de similaridade [15]. Lembramos que o Algoritmo 4 proposto por Ng *et al.* utiliza uma medida de similaridade baseada no kernel Gaussiano. Nela, dado o conjunto de dados X,

$$s_1(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right) \text{ se } i \neq j, \tag{5.1}$$

mede a similaridade entre $x_i, x_j \in X$, onde $||x_i - x_j||$ é a distância euclidiana entre x_i e x_j e $\sigma > 0$ é um parâmetro de escala. Se $i = j$, define-se $s_1(x_i, x_j) = 0$.

Observe que a função do kernel Gaussiano depende de um parâmetro de escala σ. É sabido que o resultado do algoritmo de clusterização espectral é bastante

sensível a este parâmetro [15]. Para ilustrar isso, perceba que quando σ é um valor muito alto, a similaridade entre qualquer par de pontos fica muito próxima de 1, enquanto que se σ é um valor muito baixo, a similaridade entre qualquer par de pontos fica muito próxima de 0.

Em muitas aplicações do algoritmo, esse parâmetro é estabelecido manualmente pelo usuário [92]. É comum que o algoritmo seja executado várias vezes para diferentes valores de σ e o melhor, em relação à alguma função objetivo, é escolhido. Uma clara desvantagem em executar o algoritmo para diversos valores de σ é que o torna muito mais lento. Em muitas aplicações, os valores considerados bons para σ formam um intervalo limitado, e um gargalo da utilização do método é encontrar tal σ, especialmente para estruturas mais complexas [92].

Exemplo 5.1. Para ilustrar o problema, aplicamos o Algoritmo 4 ao conjunto de dados da Figura 5.1 com $k = 3$ para dois valores de σ, 0.075 e 1. A partir desses valores o algoritmo produz classificações muito distintas, veja a Figura 5.1. Nesse caso, quando o parâmetro foi bem escolhido foi possível obter a partição que divide o conjunto de dados nas três circunferências.

Vamos tentar entender melhor por que os dois valores de σ apresentaram um comportamento tão diferente no conjunto de dados X da Figura 5.1. Vamos analisar os três pontos x_{119}, x_{123} e x_{283} em particular, que estão destacados na Figura 5.2. Note que, se buscarmos clusterizar o conjunto de dados nos três círculos que o compõem, os pontos x_{119} e x_{123} devem ser atribuídos a um mesmo cluster, enquanto que x_{283} deve ser atribuídos a outro. Logo, x_{119} e x_{123} devem ter similaridade grande entre si e x_{283} deve ter similaridade baixa com x_{119} e x_{123}. Vamos analisar os valores de $s_1(x_{119}, x_{123})$, $s_1(x_{119}, x_{283})$ e $s_1(x_{123}, x_{283})$ para $\sigma = 0.075$ e $\sigma = 1$. Quando $\sigma = 0.075$, $s_1(x_{119}, x_{123}) \approx 0.51$, $s_1(x_{119}, x_{283}) \approx 10^{-11}$ e $s_1(x_{123}, x_{283}) \approx 10^{-12}$. Quando $\sigma = 1$, $s_1(x_{119}, x_{123}) \approx 0.99$, $s_1(x_{119}, x_{283}) \approx 0.87$ e $s_1(x_{123}, x_{283}) \approx 0.86$. Percebemos que, se $\sigma = 1$, os três pares apresentam alta similaridade entre si, e mesmo que $s_1(x_{119}, x_{123})$ seja maior do que o valor com $\sigma = 0.075$, é difícil para o

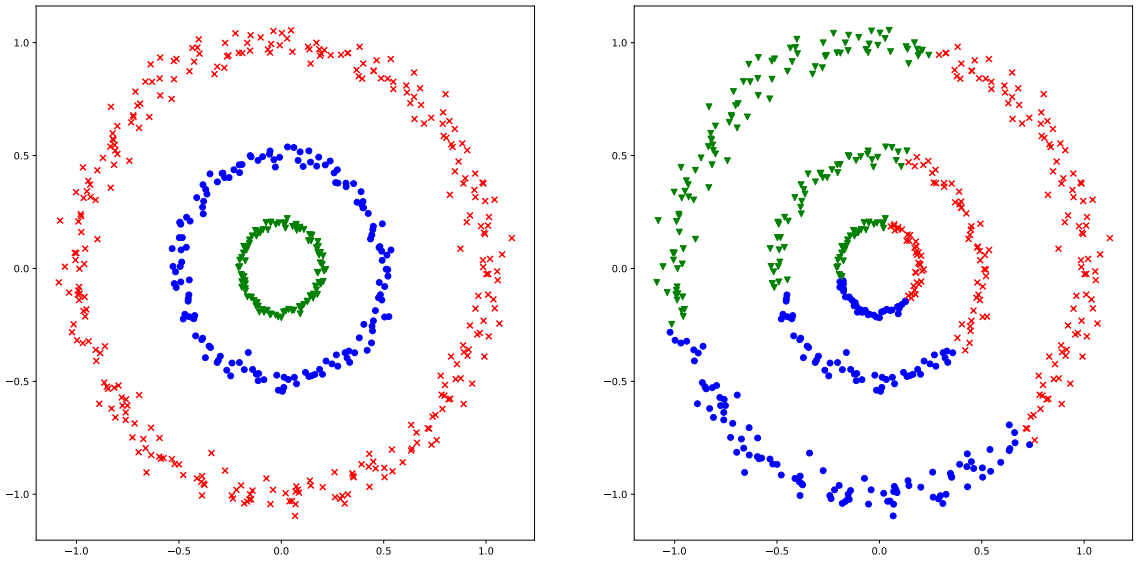

Figura 5.1: Resultados do Algoritmo 4 para dois parâmetros de escala diferentes em um conjunto de dados com três círculos. À esquerda, $\sigma = 0.075$ e à direita, $\sigma = 1$.

algoritmo distinguir se ele deve dividir x_{119} e x_{123} em um cluster sem x_{283}. Afinal, os métodos espectrais analisam o conjunto de dados como se este fosse um grafo e esses valores obtidos dizem que os três pares estão fortemente conectados.

É claro que isso gera um questionamento: como devemos escolher o σ? Já se sabe que não existe um número real σ que funcione para todo conjunto de dados. A Figura 5.3 ilustra esse fato. Ela mostra um conjunto de dados muito similar ao da Figura 5.1, onde a única diferença é uma mudança de escala. Nessa nova figura clusterizamos utilizando o valor de σ que foi adequado para o exemplo da Figura 5.1, obtendo uma partição incorreta neste caso.

Por conta disso, o valor do parâmetro σ deve depender do conjunto de dados. Algumas definições naturais de σ, como o desvio padrão e a variância, são conhecidas por terem um desempenho ruim em várias aplicações. Por exemplo, a saída do algoritmo espectral de Ng *et al.* utilizando o desvio padrão ou variância dos dados da Figura 5.1 para definir o σ, gera-se uma partição similar à apresentada na Figura 5.1 (direita).

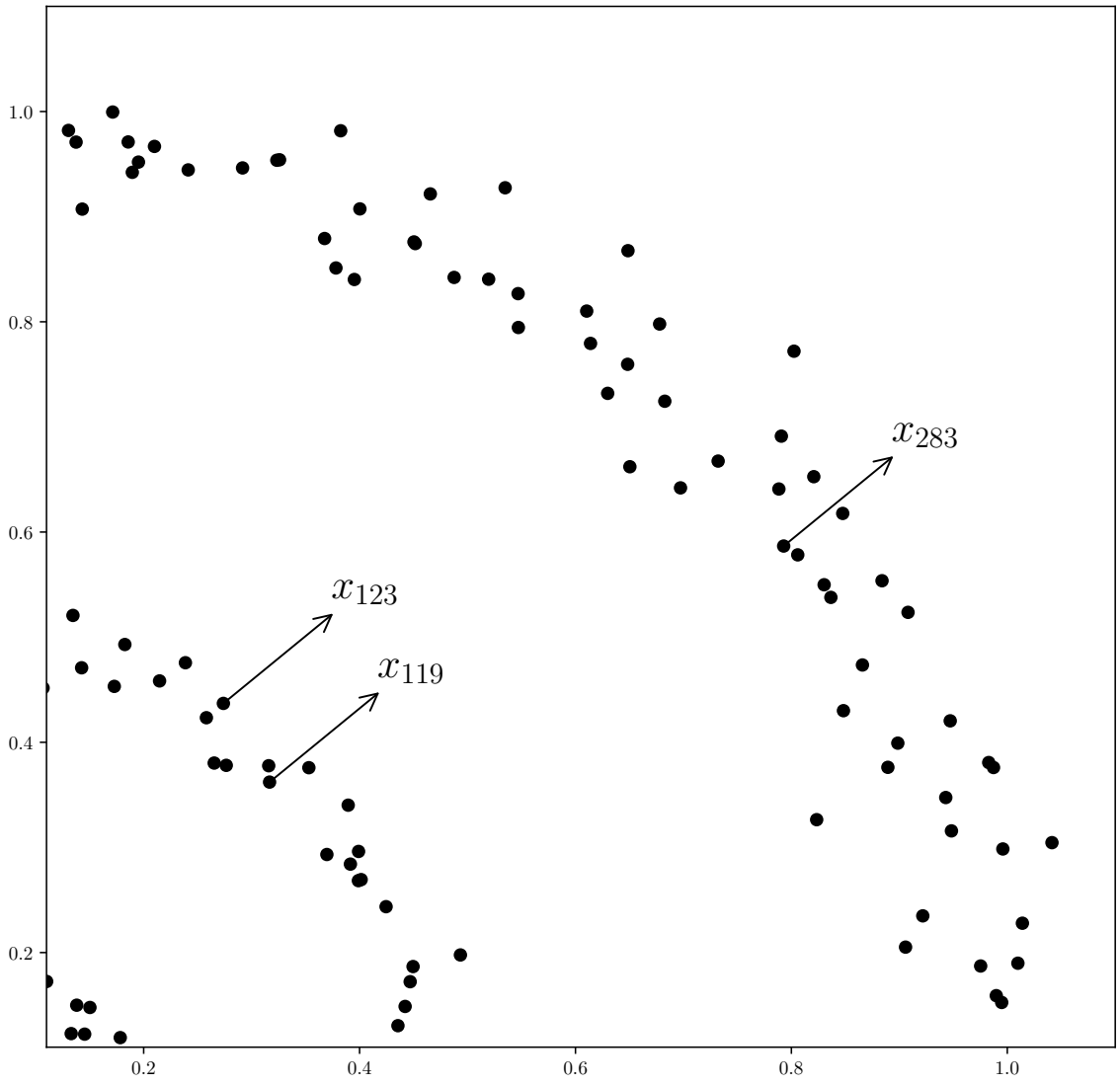

Figura 5.2: Recorte do conjunto de dados retratado na Figura 5.1, destacando três pontos em particular.

Em seu artigo, Ng *et al* [49] sugerem escolher o valor de σ que fornece os clusters mais compactos. Uma maneira de definir essa noção de compacidade é através da função J_1, definida na equação (2.2). Mais exatamente, dados um conjunto de dados, $k \in \mathbb{N}$ e $\sigma > 0$, seja y_i a i-ésima linha da matriz Y obtida no passo (4) do Algoritmo 4 e $\mathcal{C}$ a partição obtida pelo passo (5) do Algoritmo 4, então a função $J_1(\mathcal{C}) = \sum_{\ell=1}^{k} \sum_{y_i \in C_\ell} ||y_i - \mu_\ell||^2$ mede o quão compacto é $\mathcal{C}$. Quanto mais compacto

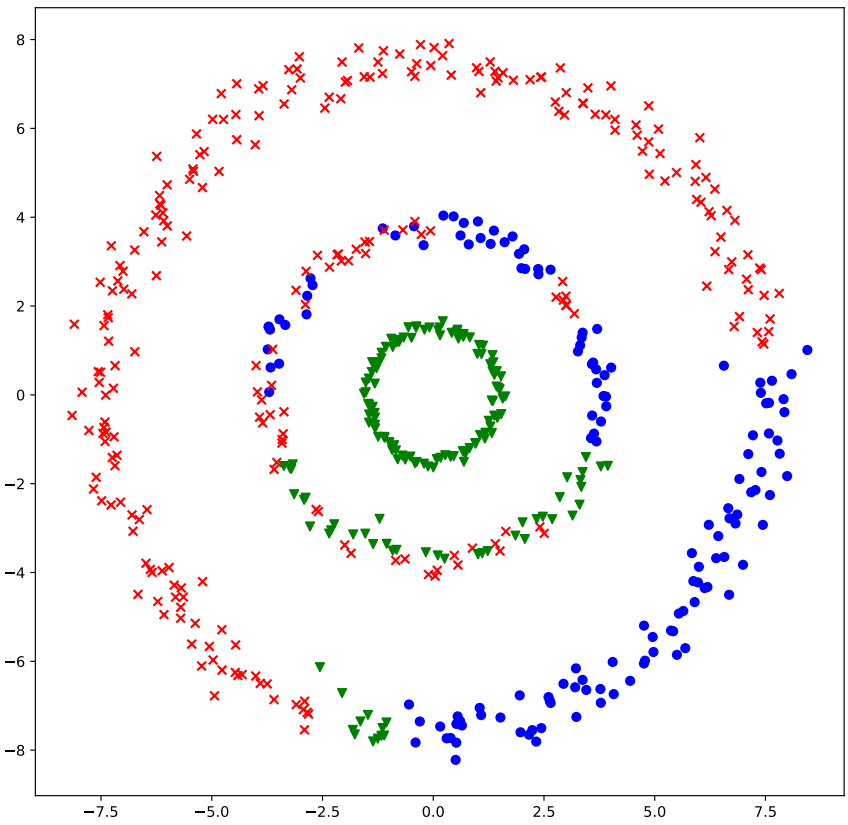

Figura 5.3: Resultado do Algoritmo 4 para $\sigma = 0.075$ em um conjunto de dados com três círculos.

$\mathcal{C}$ for, menor é o valor $J_1(\mathcal{C})$. Note que esse critério pode ser aplicado para selecionar uma partição entre duas opções concorrentes, mas não restringe o intervalo onde bons valores de σ podem ser encontrados. Em geral, o que se faz é fixar uma janela de valores e procurar um valor adequado nessa janela [92].

Alguns autores, como por exemplo [33], salientaram que, por utilizar o mesmo parâmetro de escala para todo par de pontos, a medida de similaridade do kernel Gaussiano pode não refletir a distribuição dos dados de forma precisa, particularmente quando o conjunto de dados possui múltiplas escalas, como na Figura 5.4, onde não há σ utilizado na equação (5.1) tal que o método espectral identifique a nuvem densa central. Em geral, as partições encontradas pelo Algoritmo 4 se asse-

melham à Figura 5.4 (direita), mas a partição correta seria a retratada na Figura 5.4 (esquerda).

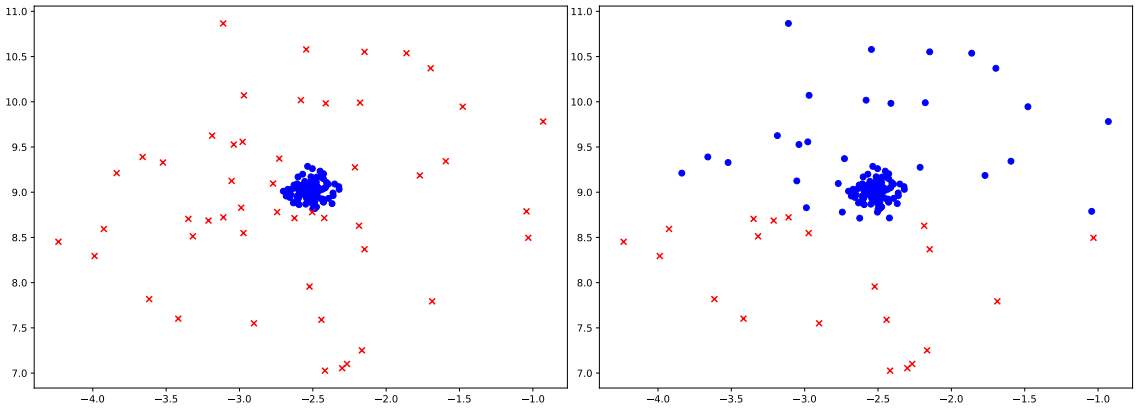

Figura 5.4: Partição correta fornecida pelo criador do conjunto de dados (esquerda). Resultado do Algoritmo 4 para $\sigma = 1$ (direita).

Visto as limitações apresentadas pela medida de similaridade definida pela equação (5.1), muitos autores propuseram novas noções de similaridade [38, 92, 89, 23, 15]. Essas medidas de similaridade vão de aumentar a faixa de valores bons para σ a substituir σ por um novo parâmetro mais fácil de definir. A seguir apresentamos algumas noções de medida de similaridade que serão exploradas na Seção 5.3. Inicialmente, descrevemos métodos que substituem o parâmetro σ na medida de similaridade definida em (5.1) por outros parâmetros.

Zelnik-Manor e Perona [89] substituem σ por um produto $\sigma_i\sigma_j$ que depende do par de pontos que está sendo considerado. O parâmetro σ_i é definido como $\sigma_i = ||x_i - x_{\ell_i}||$ onde x_{ℓ_i} denota o ℓ-ésimo elemento mais próximo de x_i. O valor $\ell \in \mathbb{N}$ é um parâmetro escolhido pelo usuário, e novamente desempenha um papel importante no controle do tamanho da vizinhança. Formalmente, os autores definiram uma medida de similaridade

$$s_2(x_i, x_j) = \exp\left(-||x_i - x_j||^2/\sigma_i\sigma_j\right), \ i \neq j.$$

O Algoritmo 5 utilizando essa medida de similaridade é conhecido como *Self-Tuning Spectral Clustering* (SC-ST). Uma das vantagens deste método é que, em aplicações típicas, mesmo valores razoavelmente pequenos de ℓ dão bons resultados, reduzindo o número de possibilidades para o parâmetro.

Zhang *et al.* [92] propuseram outra variação da medida de similaridade do kernel Gaussiano, chamada de *Density Adaptive Similarity Measure* (DA). Além do parâmetro σ, essa medida utiliza um parâmetro $\epsilon > 0$ que define se dois elementos estão próximos ou não. O parâmetro $\epsilon > 0$ deve ser definido pelo usuário. Para cada par de pontos, o número de vizinhos mais próximos comuns de x_i e x_j é definido como

$$\text{CNN}(x_i, x_j) = |\{x \in X : ||x_i - x|| < \epsilon \text{ e } ||x_j - x|| < \epsilon\}|.$$

A medida de similaridade é definida como

$$s_3(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{(2\sigma^2(\text{CNN}(x_i, x_j) + 1))}\right),\ i \neq j.$$

Notamos que, quando o desempenho do algoritmo depende de um parâmetro, como esses definidos acima, é natural executarmos o algoritmo várias vezes para parâmetros distintos, buscando um que melhor se adeque ao contexto do problema de classificação que está sendo trabalhado.

Em uma direção diferente, existem métodos cujo objetivo é fortalecer medidas de similaridade existentes, por exemplo, Fischer e Buhmann [23] propuseram a medida de similaridade *Path-Based* (PB). Com respeito à medida espectral padrão, ela é definida como

$$s_4(x_i, x_j) = \max_{p \in \phi_{ij}} \min_{1 \leq h < |p|} s_1(x_{p[h]}, x_{p[h+1]}),\ i \neq j$$

onde ϕ_{ij} é o conjunto de todos os caminhos p conectando x_i a x_j, $|p|$ denota o comprimento do caminho e $x_{p[h]}$ denota o h-ésimo vértice ao longo do caminho p.

Chang e Yeung [15] propuseram uma medida de similaridade que eles chamaram de medida de similaridade *Robust Path-Based* (RPB). Para definir essa

medida, os autores definiram inicialmente $\epsilon = \max_i \min_j ||x_i - x_j||$. Para cada elemento, eles consideram $w_i' = \sum_{x_j \in N_i} s_1(x_i, x_j)$, onde $N_i = \{x_j \in X : ||x_i - x_j|| \leq \epsilon\}$ e $w_i = w_i' / \max_j(w_j')$. A medida de similaridade para $i \neq j$ é definida como

$$s_5(x_i, x_j) = \max_{p \in \phi_{ij}} \min_{1 \leq h < |p|} w_{p[h]} s_1(x_{p[h]}, x_{p[h+1]}) w_{p[h+1]}.$$

Li e Guo [38] propuseram uma medida de similaridade chamada *Neighbor Propagation* (NP). Eles definem $\epsilon = \max_i \min_j ||x_i - x_j||$ e estabelecem o conjunto de relações vizinhas $R = \{(x_i, x_j) : ||x_i - x_j|| \leq \epsilon\}$. A partir disso, eles definem o neighbor propagation principle, que estende R a um conjunto R' de forma transitiva. Ou seja, se $(x_i, x_j) \in R$ e $(x_j, x_l) \in R$, então $(x_i, x_l) \in R'$. A medida de similaridade $s_6(x_i, x_j)$ é inicialmente definida para coincidir com $s_1(x_i, x_j)$. Depois, para cada par $(x_i, x_l) \in R' \setminus R$ tal que $(x_i, x_j), (x_j, x_l) \in R$, é redefinido como $s_6(x_i, x_l) = \min\{s_6(x_i, x_j), s_6(x_j, x_l)\}$.

Observamos que as medidas de similaridade s_4, s_5 e s_6 foram definidas utilizando o kernel Gaussiano como medida básica de similaridade. É claro que outras medidas de similaridade podem ser utilizadas como ponto de partida.

5.2 Medida de Similaridade Hierárquica

As medidas de similaridade descritas na seção anterior foram projetadas para aumentar a faixa de valores bons para o parâmetro de escala σ, ou para substituir σ por outro parâmetro de escala mais fácil de definir. Nesta seção, propomos uma medida de similaridade que modifica o kernel Gaussiano de uma forma que não requer nenhum parâmetro de escala. Em vez disso, ele incorpora informações de uma árvore hierárquica com pesos nos vértices, como explicaremos agora.

Iniciamos com terminologia. Considere um grafo H com pesos em seus vértices, ou seja, uma tripla $H = (V, \omega_V, E)$ com conjunto de vértices V, conjunto

de arestas E e uma função de peso $\omega_V : V \to \mathbb{R}_{\geq 0}$ que atribui um peso não negativo ω_i a cada vértice v_i em V.

Na Subseção 2.2.1, foi apresentado o Algoritmo single linkage. Esse algoritmo realiza, passo a passo, a união de clusters, até que todos os pontos estejam em um mesmo cluster. Esse algoritmo produz um dendrograma, que representa uma hierarquia de clusters. Definimos a árvore hierárquica a partir desse dendrograma. Inicialmente, um vértice de peso zero é criado para cada um dos pontos no conjunto de dados. Em outras palavras, existe um vértice para cada cluster no início do procedimento aglomerativo. Quando dois clusters C_ℓ e C_θ são combinados, um novo vértice correspondente a $C_\ell \cup C_\theta$ é criado, com peso referente a distância entre os dois clusters que é dada por $D_1(C_\ell, C_\theta) = \min\{||x-y|| : x \in C_\ell, y \in C_\theta\}$. Duas arestas são adicionadas para que esse novo vértice se torne o pai dos vértices correspondentes a C_ℓ e C_θ. Apresentamos o pseudo-código de um algoritmo que produz a árvore hierárquica a seguir.

Cada par de pontos x_i e x_j no conjunto de dados é conectado por um caminho p_{ij} na árvore hierárquica construída pelo Algoritmo 7. Seja V_{ij} o conjunto de vértices no caminho p_{ij}. Nós o utilizamos para definir um parâmetro para o kernel Gaussiano γ_{ij}:

$$\gamma_{ij} = \frac{|p_{ij}| - 2}{\sum_{x \in V_{ij}} \omega_x + ||x_i - x_j||}, \tag{5.2}$$

onde $|p_{ij}|$ é o número de arestas no caminho p_{ij}.

A intuição por trás da equação (5.2) é que pares de elementos que foram combinados em poucos passos na árvore hierárquica terão γ_{ij} baixo, enquanto que pares de elementos em que foi necessário muitos passos para estarem em um mesmo cluster pela abordagem hierárquica terão γ_{ij} alto. Afinal, γ_{ij} cresce à medida que $|p_{ij}|$ cresce. O denominador da equação (5.2) é definido dessa forma para que γ_{ij}

Algoritmo 7: Árvore Hierárquica

Entrada: Conjunto de dados $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^m$

Saída: Árvore Hierárquica $H = (V, \omega_V, E)$

1 Defina $\mathcal{C} = \{C_1, \ldots, C_n\}$ com clusters $C_i = \{x_i\}$ $\forall i \in \{1, \ldots, n\}$ e defina um grafo com peso nos vértices H, com n vértices isolados, um para cada cluster em $\mathcal{C}$, com peso $\omega_{\{x_i\}} = 0$.

2 **para** $\ell \in \{1, \ldots, n-1\}$ **faça**

3 Seja (C_α, C_β), $\alpha < \beta$, o par de clusters que atinge o valor $d^* = \min\{D_1(A, B) : A, B \in \mathcal{C}\}$. Se existirem $(\alpha_1, \beta_1), \ldots, (\alpha_q, \beta_q)$ tal que $(C_{\alpha_j}, C_{\beta_j})$ atinge o valor mínimo, seja α o índice tal que $\alpha \leq \alpha_j, \forall j \in \{1, \ldots, q\}$. Escolha β tal que $\beta \leq \beta_j$, para todo (α_j, β_j) em que $\alpha = \alpha_j$.

4 Adicione um vértice correspondente a $C_\alpha \cup C_\beta$ ao conjunto de vértices de H, conecte-o a C_α e C_β e atribua-o peso d^*.

5 Defina $C_\alpha = C_\alpha \cup C_\beta$.

6 Defina $\mathcal{C} = \mathcal{C} \setminus \{C_\beta\}$

7 **fim**

8 **Retorne** H

não assuma valores grandes rapidamente, pois temos por objetivo utilizá-lo como ingrediente de uma função exponencial.

Notamos que calcular a árvore hierárquica com base no conjunto de dados e calcular os parâmetros γ_{ij} é relativamente rápido em comparação com o custo computacional dos métodos de clusterização espectral. De fato, é computacionalmente mais barato do que calcular os autovetores associados aos menores autovalores da matriz Laplaciana normalizada diversas vezes (veja o passo (3) do Algoritmo 4).

A medida de similaridade hierárquica entre um par de pontos é então definida como:

$$s_h(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2}\gamma_{ij}^2\right), \text{ se } i \neq j. \tag{5.3}$$

Esta medida de similaridade pode ser vista como a substituição de $1/\sigma^2$ por γ_{ij}^2 em (5.1). Perceba que, quanto maior é γ_{ij} menor é a similaridade entre x_i e x_j, enquanto que quanto menor é γ_{ij} maior é a similaridade entre x_i e x_j. O Exemplo 5.2 ilustra como essa medida funciona na prática.

Exemplo 5.2. Consideramos $X = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\} = \{(1,1), (1,1.5), (2,1), (3,1.5), (7.5,3), (8.5,3), (8,5), (8,6)\}$. Vamos construir a árvore hierárquica de X. A sequência de oito figuras a seguir, ilustra todos os passos do Algoritmo 7. A Figura 5.5 mostra a inicialização do algoritmo, colocando cada ponto em um cluster distinto.

Depois da inicialização, começamos a recursão em $\ell = 1$, combinamos os dois clusters mais próximos, que no caso são $\{v_1\}$ e $\{v_2\}$. À Figura 5.6 ilustra essa situação.

A Figura 5.7 apresenta o resultado após juntarmos os dois clusters mais próximos, para $\ell = 2$, que são $\{v_1, v_2\}$ e $\{v_3\}$.

Para $\ell = 3$, juntamos $\{v_7\}$ e $\{v_8\}$ (Figura 5.8).

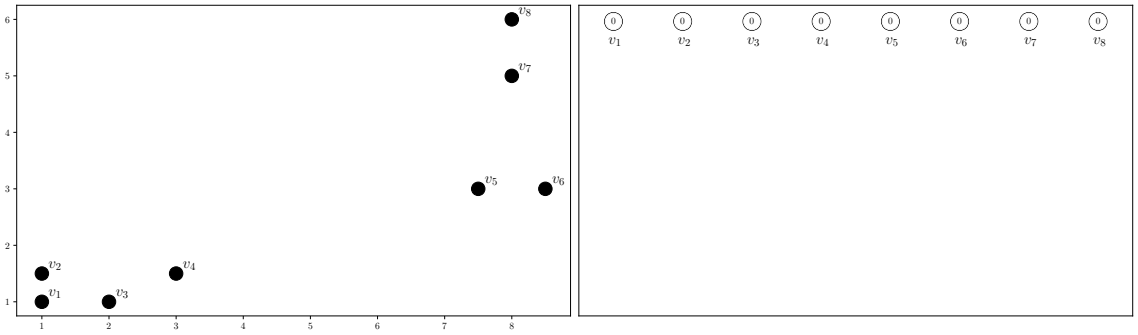

Figura 5.5: À esquerda, o conjunto de dados X, sem nenhum ponto em um mesmo cluster. A direita, o primeiro passo da construção da árvore hierárquica.

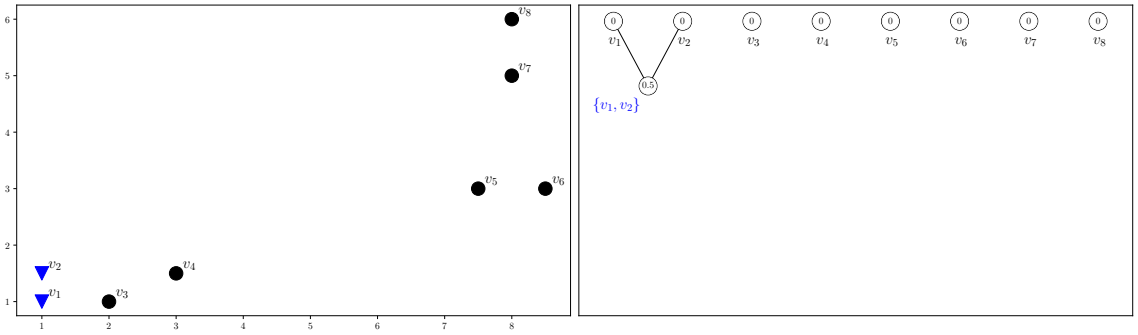

Figura 5.6: À esquerda, o conjunto de dados X e sua partição $\mathcal{C} = \{\{v_1, v_2\}, \{v_3\}, \{v_4\}, \{v_5\}, \{v_6\}, \{v_7\}, \{v_8\}\}$. À direita, a recursão para $\ell = 1$.

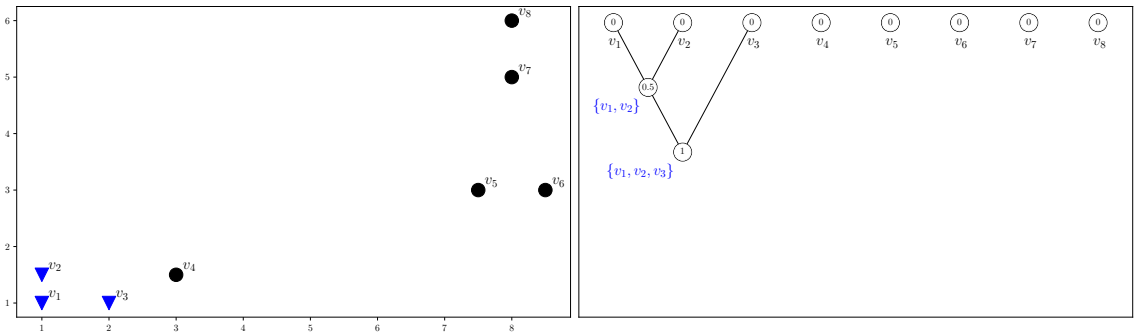

Figura 5.7: À esquerda, o conjunto de dados X e sua partição $\mathcal{C} = \{\{v_1, v_2, v_3\}, \{v_4\}, \{v_5\}, \{v_6\}, \{v_7\}, \{v_8\}\}$. À direita, a recursão para $\ell = 2$.

Para $\ell = 4$, juntamos $\{v_5\}$ e $\{v_6\}$ (Figura 5.9).

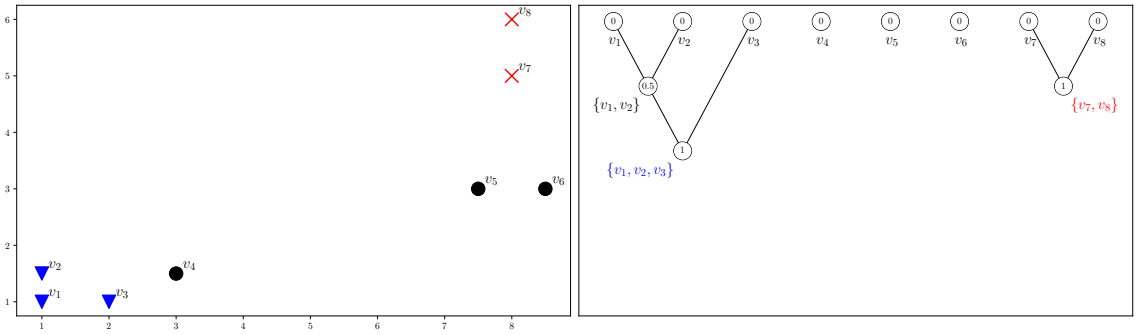

Figura 5.8: À esquerda, o conjunto de dados X e sua partição $\mathcal{C} = \{\{v_1, v_2, v_3\}, \{v_4\}, \{v_5\}, \{v_6\}, \{v_7, v_8\}\}$. À direita, a recursão para $\ell = 3$.

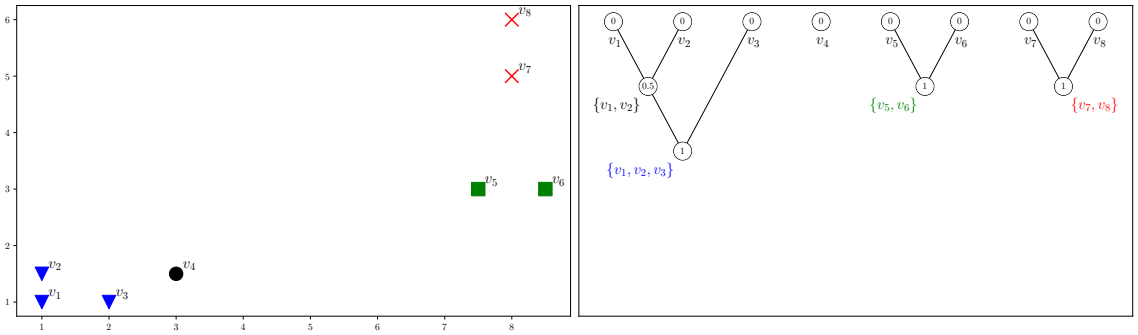

Figura 5.9: À esquerda, o conjunto de dados X e sua partição $\mathcal{C} = \{\{v_1, v_2, v_3\}, \{v_4\}, \{v_5, v_6\}, \{v_7, v_8\}\}$. À direita, a recursão para $\ell = 4$.

Para $\ell = 5$, juntamos $\{v_1, v_2, v_3\}$ e $\{v_4\}$ (Figura 5.10).

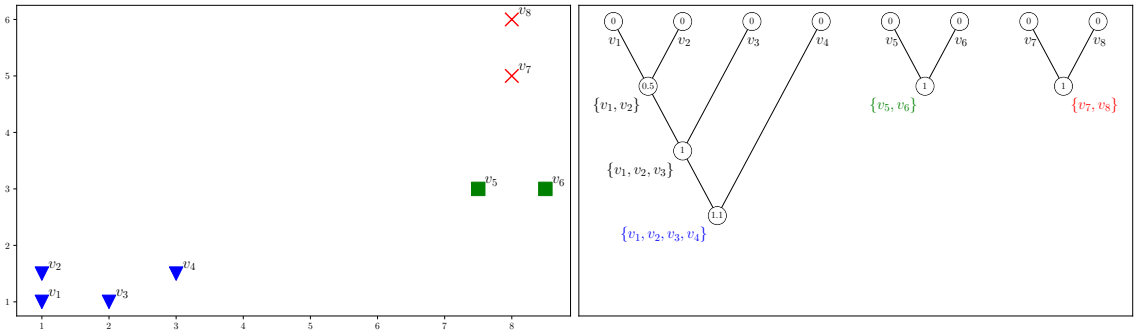

Figura 5.10: À esquerda, o conjunto de dados X e sua partição $\mathcal{C} = \{\{v_1, v_2, v_3, v_4\}, \{v_5, v_6\}, \{v_7, v_8\}\}$. À direita, a recursão para $\ell = 5$.

Para $\ell = 6$, juntamos $\{v_4, v_5\}$ e $\{v_7, v_8\}$ (Figura 5.11) e, para $\ell = 7$, juntamos $\{v_1, v_2, v_3, v_4\}$ e $\{v_5, v_6, v_7, v_8\}$ (Figura 5.12).

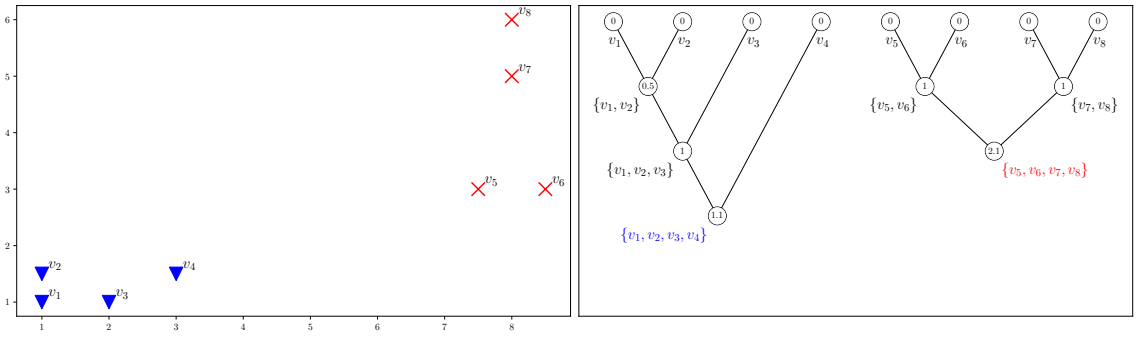

Figura 5.11: À esquerda, o conjunto de dados X e sua partição $\mathcal{C} = \{\{v_1, v_2, v_3, v_4\}, \{v_5, v_6, v_7, v_8\}\}$. À direita, a recursão para $\ell = 6$.

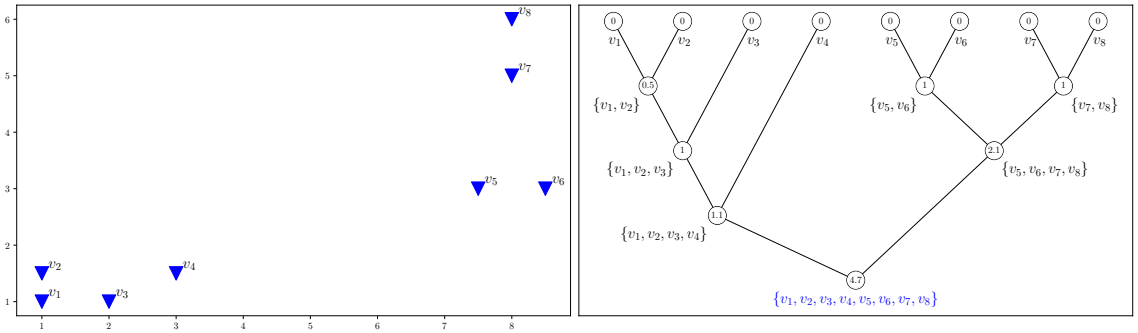

Figura 5.12: À esquerda, o conjunto de dados X e sua partição $\mathcal{C} = \{\{v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8\}\}$. À direita, a recursão para $\ell = 7$.

O algoritmo termina quando todos pontos estão em um mesmo cluster, tendo como saída a árvore hierárquica de X (Figura 5.12, direita).

Uma vez estabelecida a árvore hierárquica de X, o cálculo de cada γ_{ij} é simples, basta olhar para o caminho entre v_i e v_j. Exemplificamos com v_3 e v_5. $|p_{35}| = 6$ e $\sum_{v \in V_{ij}} \omega_x \approx 1 + 1.1 + 4.7 + 2.1 + 1 = 9.9$, esse caminho está ilustrado na Figura 5.13. Assim, $\gamma_{35} \approx 0.25$.

Essa nova medida de similaridade tem as seguintes características:

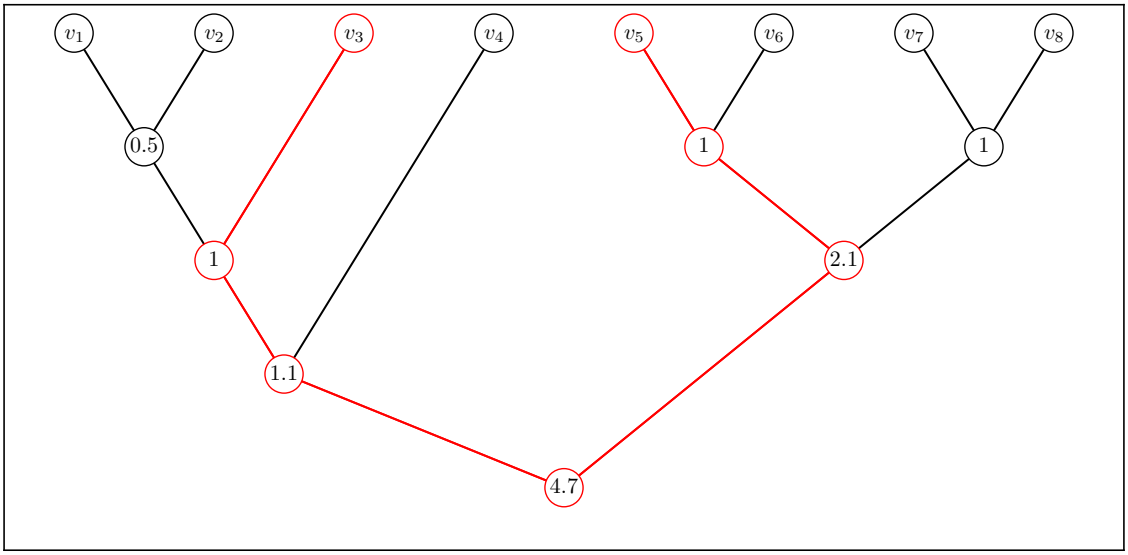

Figura 5.13: Árvore hierárquica do conjunto de dados X destacando o caminho entre v_3 e v_5, em vermelho.

(1) Nenhum parâmetro de escala precisa ser definido manualmente, pois γ_{ij} é calculado diretamente do conjunto de dados. Isso o torna mais fácil de ser aplicado.

(2) O algoritmo de clusterização espectral que utiliza essa similaridade é mais rápido do que algoritmos que são executados para vários valores do parâmetro de escala, pois os autovetores precisam ser calculados apenas uma vez.

(3) A similaridade hierárquica é invariante sob translações e expansões de um conjunto de dados. Afinal, dados os conjuntos de dados $X = \{x_1, \ldots, x_n\}$ e $\mathcal{X}' = \{x'_1, \ldots, x'_n\}$, onde $x'_i = c_1 x_i + c_2$, para $c_1, c_2, \neq 0$ fixados, é trivial ver que $s_h(x_i, x_j) = s_h(x'_i, x'_j)$ para todo i, j.

(4) Suponha que dois elementos x_i e x_j estejam mais próximos um do outro, no sentido de que $||x_i - x_j|| < \min\{||x_i - x_u||, ||x_j - x_u||\}$ para todo $u \in [n] \setminus \{i, j\}$. Seria natural supor que x_i e x_j pertencem ao mesmo cluster em uma partição ótima. Como qualquer par x_i, x_j com esta

propriedade está conectado por um caminho p_{ij} de comprimento dois na árvore hierárquica, a medida de similaridade hierárquica sempre atribui o valor $s_h(x_i, x_j) = 1$ (o valor máximo possível) para este par.

5.3 Experimentos

Para avaliar o desempenho de um algoritmo de clusterização é natural realizar experimentos em conjuntos de dados tradicionais, com o intuito de comparar seus resultados com os algoritmos já propostos na literatura, como o Algoritmo 4. Essa seção mostra nossos experimentos em conjuntos de dados sintéticos e em conjuntos de dados reais. Iremos nos referir a cada algoritmo por SC-NM (*Spectral Clustering - Nome da Medida*). Utilizamos as abreviaturas das medidas definidas na Seção 5.1. Dessa forma, definimos os algoritmos, SC-ST, SC-PB, SC-RPB, SC-NP e SC-DA, onde todos utilizam a estrutura do Algoritmo 5 com uma medida de similaridade em particular. SC-GK se referirá ao Algoritmo 4. SC-HA é o Algoritmo 5 utilizando a medida de similaridade hierárquica definida na Seção 5.2.

5.3.1 Escolha de Parâmetros

Como vimos na Seção 5.1, os algoritmos espectrais SC-GK, SC-ST, SC-PB, SC-RPB, SC-NP e SC-DA necessitam de parâmetros pré-definidos pelo usuário para poderem ser executados. Dedicamos essa subseção para apresentar quais serão os parâmetros utilizados em cada algoritmo.

Para escolher o valor do parâmetro σ no SC-GK, vários autores sugeriram procurar σ em um intervalo entre 10% e 20% do alcance total das distâncias euclidianas, veja [92]. Esse intervalo pode ser calculado considerando as distâncias entre todos os pares de pontos do conjunto de dados ou considerando apenas os valores de mínimo e máximo das distâncias entre todos pares de pontos do conjunto

de dados. Como a primeira opção é menos sensível à presença de *outliers*, nós a utilizamos neste trabalho. Devemos mencionar, no entanto, que ambas as opções produzem resultados muito semelhantes para nossos conjuntos de dados.

Mais precisamente, dado um conjunto de dados $X = \{x_1, \ldots, x_n\}$, definimos o vetor $d = (d_1, \ldots, d_N)$ contendo as distâncias entre cada par de pontos em X, em ordem crescente. Os valores de σ utilizados em nossas aplicações dos algoritmos SC-GK, SC-PB, SC-RPB, SC-DA e SC-NP são $\sigma = d_u$ para $u = 1, \ldots, \lceil \frac{N}{5} \rceil$. Ou seja, testamos todos os valores no primeiro quintil do conjunto de distâncias dos elementos no conjunto.

Com relação a outros parâmetros, ao executar o algoritmo SC-ST, o parâmetro ℓ é testado para todos os inteiros entre 2 e 20. Para os algoritmos SC-NP e SC-DA, o parâmetro ϵ é definido como $\epsilon = \max_i \min_j ||x_i - x_j||$. Finalmente, o número de vezes que executamos o passo (5) do Algoritmo 5 é $Q = 100$.

5.3.2 Experimentos em Conjuntos de Dados Sintéticos

Conjuntos de dados sintéticos são aqueles que são criados com o objetivo de desafiar os algoritmos de classificação a encontrar algum padrão definido pelo usuário. Nós aplicamos os algoritmos descritos nas Seções 5.1 e 5.2 a seis conjuntos de dados sintéticos que são frequentemente utilizados para avaliar o desempenho de métodos espectrais.

Os resultados para SC-GK, SC-ST, SC-PB, SC-DA e SC-HA estão descritos nas Figuras 5.14 a 5.18, respectivamente. Omitimos os resultados para SC-NP porque eles se mostraram muito semelhantes aos resultados para SC-GK, mesmo nos conjuntos de dados com pertubações e com a presença de *outliers*. O mesmo se aplica ao SC-RPB, cujos resultados foram muito semelhantes aos do SC-PB.

Cada figura contém os resultados de um dos métodos para todos os seis conjuntos de dados. Para simplificar, nos referimos a $F_{ij}^{(k)}$ para denotar o conjunto de dados na linha i e coluna j na Figura k. Na linha superior, os conjuntos de dados são, da esquerda para a direita, um círculo com duas nuvens, dois círculos com pertubação e duas luas. Na linha de baixo temos duas espirais, três círculos e duas nuvens com escalas diferentes. Como de costume, os objetos individuais que formam cada um dos conjuntos de dados são gerados separadamente e os algoritmos recebem a tarefa de identificar cada objeto individual. Eles recebem o conjunto de dados e o número correto de clusters como entrada.

Os conjuntos de dados $F_{13}^{(k)}, F_{21}^{(k)}$ e $F_{22}^{(k)}$ são conjuntos de dados em que o SC-GK funciona perfeitamente. O conjunto de dados $F_{23}^{(k)}$ contém dados em duas escalas diferentes, o que tende a ser muito desafiador para algoritmos de clusterização espectral. O conjunto de dados $F_{11}^{(k)}$ torna-se mais difícil à medida que os pontos contidos nas nuvens estão mais próximos dos pontos no círculo. O conjunto de dados $F_{12}^{(k)}$ contém pertubação, o que é um desafio para qualquer algoritmo de clusterização.

O algoritmo SC-GK encontrou a partição correta em 4 das 6 situações (veja a Figura 5.14). Para os conjuntos de dados $F_{11}^{(5.14)}$ e $F_{23}^{(5.14)}$, nenhum σ dentre os que utilizamos levou ao resultado correto. Em relação ao primeiro conjunto de dados, escolher um único valor de σ fixo para todo par de pontos dificulta a distinção entre clusters que estão próximos uns dos outros. Em relação ao último conjunto, é sabido que o SC-GK possui dificuldade em identificar clusters onde os tamanhos e densidades locais variam muito [92].

O algoritmo SC-DA obteve a partição correta em cinco dos conjuntos de dados, conforme ilustrado na Figura 5.15. Vale salientar que, exceto para $F_{23}^{(5.15)}$, a faixa de valores de σ onde a partição esperada apareceu aumentou, conforme esperado.

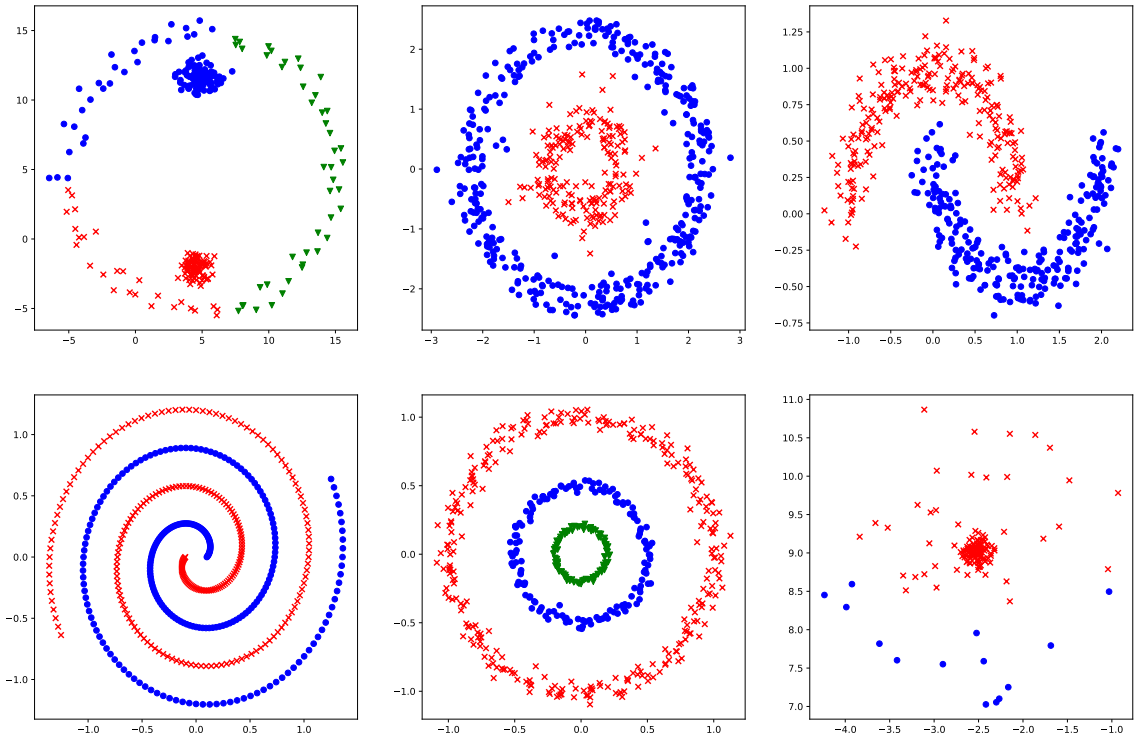

Figura 5.14: Resultados para cada conjunto de dados usando o algoritmo SC-GK [49]. No topo, da esquerda para a direita, os melhores resultados foram obtidos para $\sigma = 0,9$, $\sigma = 0,1$ e $\sigma = 0,05$, respectivamente. Na parte inferior, da esquerda para a direita, os melhores resultados foram obtidos para $\sigma = 0,1$, $\sigma = 0,075$ e $\sigma = 0,5$.

O algoritmo SC-ST encontrou a partição correta em quatro dos conjuntos de dados, conforme apresentado na Figura 5.16. O algoritmo se mostrou mais sensível ao pertubação, pois falhou para $F_{12}^{(5.16)}$ e $F_{13}^{(5.16)}$.

O SC-PB teve um bom desempenho geral, conseguindo obter o resultado correto em cinco dos seis conjuntos de dados (Figura 5.17). Além disso, teve um desempenho melhor que o SC-GK no conjunto de dados $F_{23}^{(5.17)}$.

O algoritmo proposto neste trabalho, SC-HA, obteve um bom desempenho para todos os seis conjuntos de dados, conforme ilustrado na Figura 5.18. Ele lidou com sucesso no conjunto de dados com pertubação ($F_{12}^{(5.18)}$), com várias escalas ($F_{23}^{(5.18)}$) e estruturas mais complexas ($F_{11}^{(5.18)}$). Em quatro dos conjuntos de

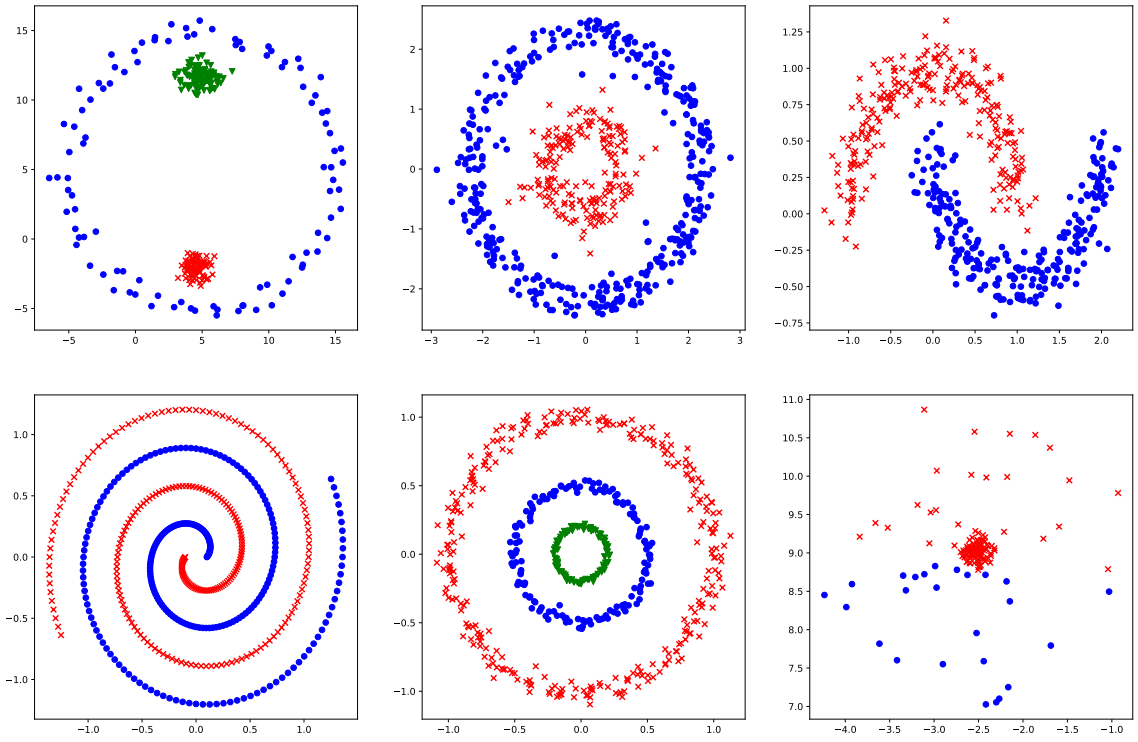

Figura 5.15: Resultados para cada conjunto de dados usando o algoritmo de clusterização espectral SC-DA [92]. Na parte superior, da esquerda para a direita, utilizamos $\sigma = 0.45$, $\sigma = 0.1$ e $\sigma = 0.1$, respectivamente. Na parte inferior, da esquerda para a direita, utilizamos $\sigma = 0,05$, $\sigma = 0,075$ e $\sigma = 1,5$.

dados, todos os pontos do conjunto de dados foram classificados corretamente, em $F_{11}^{(5.18)}$ exatamente dois pontos no conjunto de dados (dentre 300 no total) foram atribuídos ao cluster incorreto e em $F_{13}^{(5.18)}$ um único ponto (de 500) foi classificado incorretamente.

5.3.3 Experimentos em Conjuntos de Dados Reais

Também comparamos o desempenho dos diferentes algoritmos espectrais em dez conjuntos de dados reais extraídos do repositório UCI Machine Learning Repository [20]. Quantificamos o desempenho de cada método usando dois índices

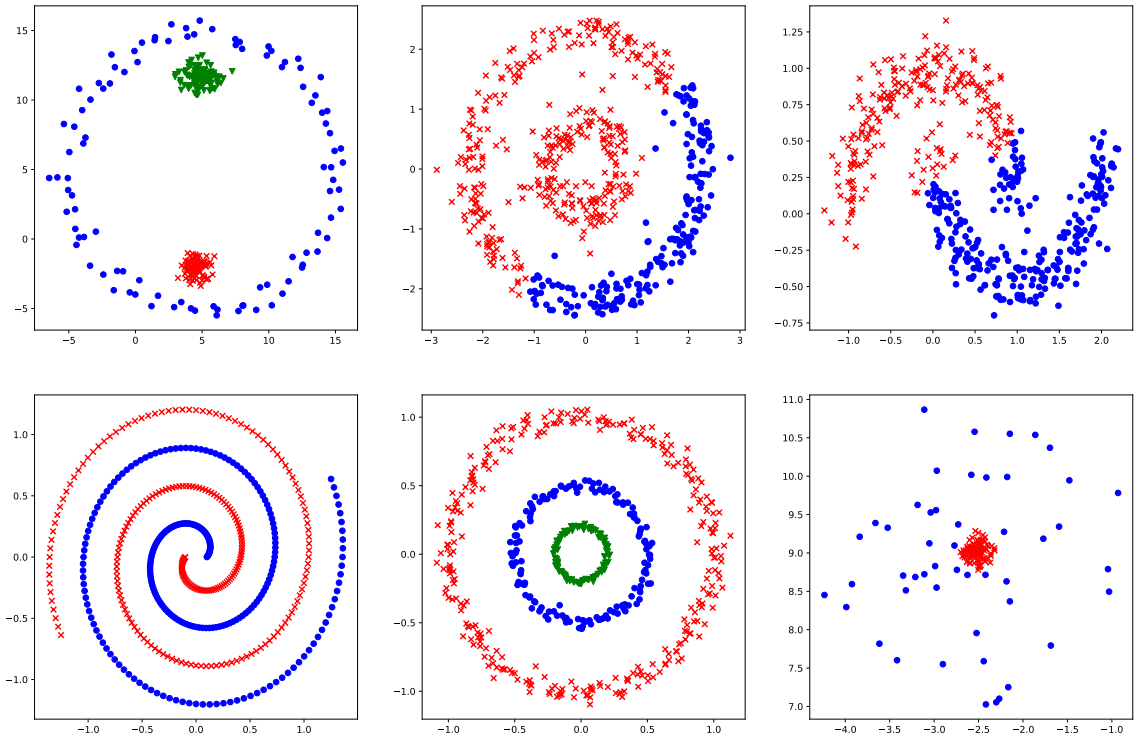

Figura 5.16: Resultados em cada conjunto de dados usando o algoritmo de clusterização espectral SC-ST [89]. Na parte superior, da esquerda para a direita, utilizamos $\ell = 7$, $\ell = 7$ e $\ell = 7$, respectivamente. Na parte inferior, da esquerda para a direita, utilizamos $\ell = 2$, $\ell = 7$ e $\ell = 3$.

de performance, a precisão e a informação mútua normalizada, que comparam a saída dos algoritmos com uma partição de referência, considerada correta, fornecida pelo repositório. Ressaltamos que esses são dois índices de performance comumente utilizadas pela comunidade científica [92].

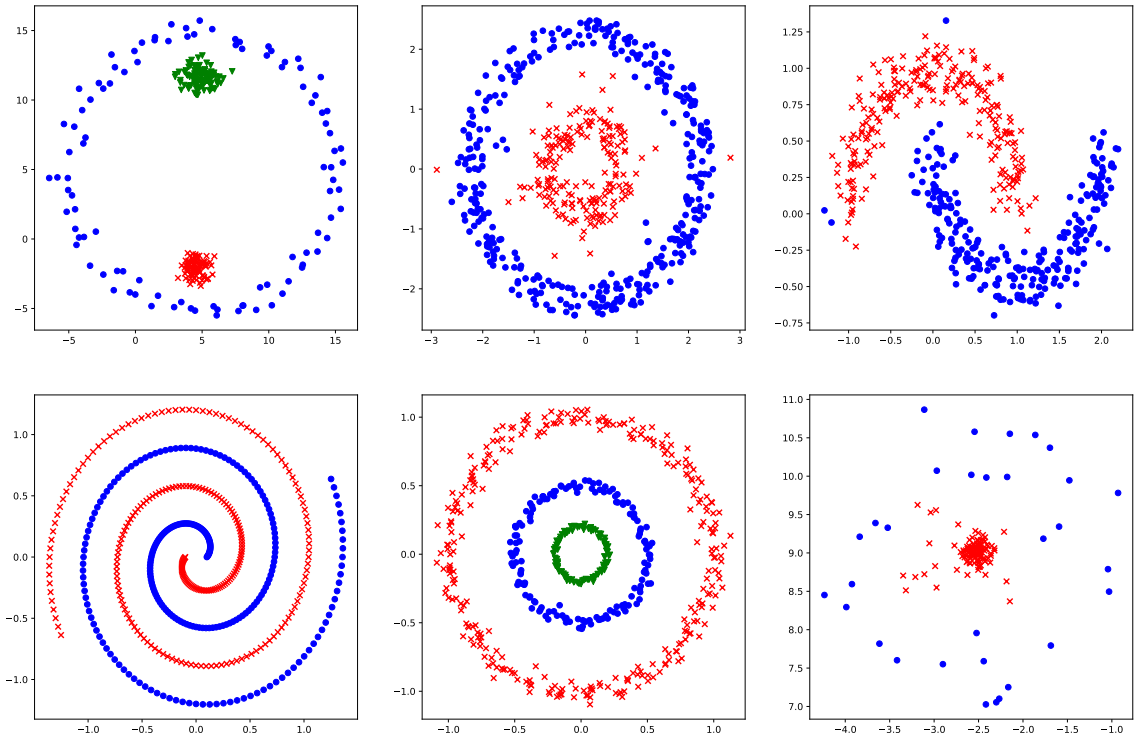

Figura 5.17: Resultados em cada conjunto de dados usando o algoritmo de clusterização espectral SC-PB [23]. Na parte superior, da esquerda para a direita, utilizamos $\sigma = 0.9$, $\sigma = 0.25$ e $\sigma = 0.05$, respectivamente. Na parte inferior, da esquerda para a direita, utilizamos $\sigma = 0,05$, $\sigma = 0,075$ e $\sigma = 0,5$.

5.3.4 Índices de Performance

Dado um conjunto de dados $X = \{x_1, \ldots, x_n\}$, o objetivo é comparar a partição correta $Z = \{Z_1, \ldots, Z_k\}$[1] e a saída $C = \{C_1, \ldots, C_k\}$ de um algoritmo de clusterização.

A precisão (ACC[2]) mede a proporção de pontos que foram classificados corretamente. Por sua simplicidade, é a medida de desempenho de cluster mais utili-

[1]Essa é a solução fornecida no repositório, em aplicações típicas do mundo real, a partição correta não é conhecida.

[2]Accuracy.

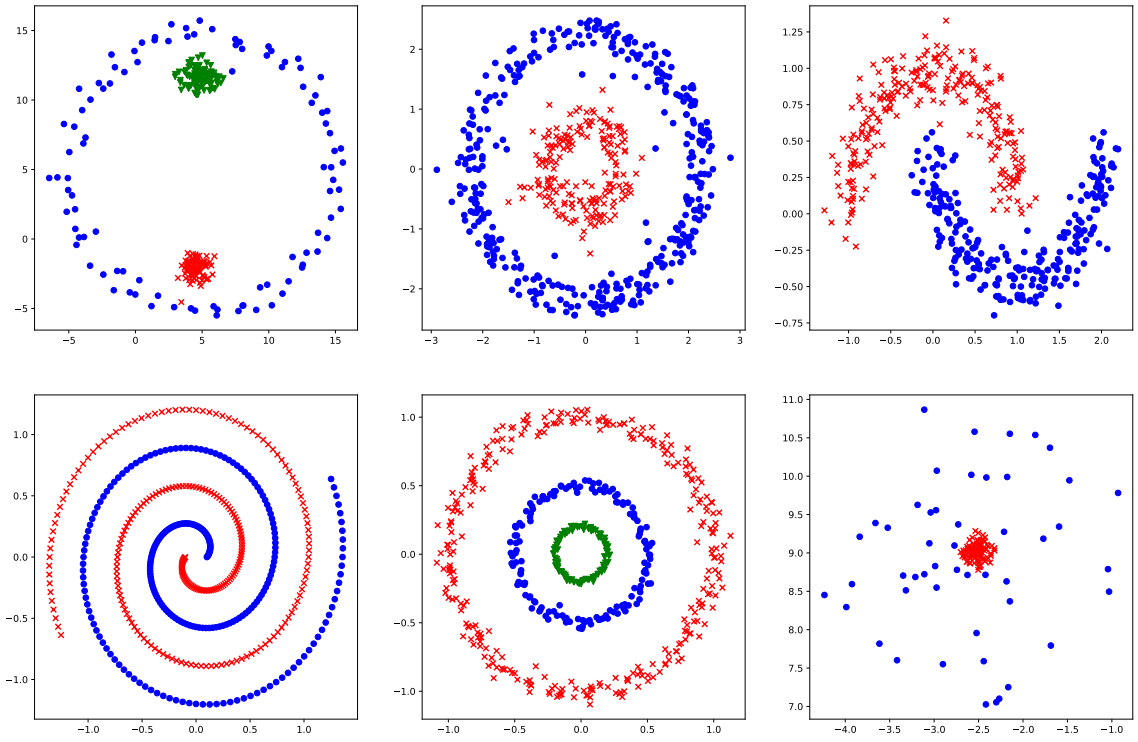

Figura 5.18: Resultados em cada conjunto de dados usando o algoritmo de clusterização espectral SC-HA.

zada. Definimos o ACC da seguinte forma, seja P o conjunto de todas as permutações $\pi = (\pi_1, \ldots, \pi_k)$ de $[k]$, e defina $f : P \to [0, 1]$ como $f(\pi) = \frac{1}{n} \sum_{\ell=1}^{k} |C_{\pi_\ell} \cap Z_\ell|$.

O valor da precisão é definido tomando a permutação para a qual f é máxima:

$$\mathrm{ACC}(C, Z) = \max_{\pi \in P} f(\pi). \tag{5.4}$$

Observe que, quanto maior a precisão, melhor é o desempenho.

Outro índice comum para medir a qualidade do agrupamento é a informação mútua normalizada (NMI[3]) [72], que é motivada pela definição de entropia. A entropia de uma partição $C = \{C_1, \ldots, C_k\}$ é definida como

[3]Normalized mutual information.

$$E_C = \sum_{\ell=1}^{k} |C_\ell| \log\left(\frac{|C_\ell|}{n}\right) \tag{5.5}$$

e a informação mútua normalizada entre dois agrupamentos C e Z é definida como

$$\mathrm{NMI}(C, Z) = \frac{1}{\sqrt{E_C E_Z}} \sum_{\ell,h=1}^{k} |C_\ell \cap Z_h| \log\left(\frac{n|C_\ell \cap Z_h|}{|C_\ell||Z_h|}\right). \tag{5.6}$$

Assim como a precisão, quanto maior o NMI, melhor o desempenho do algoritmo. Vale mencionar que, normalmente, o NMI é mais exigente que o ACC. Por exemplo, um pequeno subconjunto de elementos que foi classificado pode baixar muito o valor do NMI, enquanto que só diminui o valor ACC proporcionalmente ao tamanho desse subconjunto.

5.3.5 Resultados nos conjuntos de dados do UCI Machine Learning Repository

Essa subseção apresenta os resultados dos algoritmos de clusterização espectral sobre 10 conjuntos de dados reais disponíveis no Repositório UCI Machine Learning Repository [20]. Esses conjuntos de dados estão listados na Tabela 5.1 com as seguintes informações: #Instâncias é o número de pontos no conjunto de dados, #Atributos é a dimensão do conjunto de dados e #Clusters é o número correto de clusters do conjunto de dados.

Conforme descrito na Subseção 5.3.1, é definido alguns valores para os quais os parâmetros σ e ℓ podem ser escolhidos. Cada algoritmo espectral é executado utilizando cada vez um valor diferente para seu parâmetro de entrada. Ao final a melhor partição obtida é escolhida. Apresentamos os resultados em termos de duas noções diferentes de melhor partição. Uma é a partição obtida pelo algoritmo que mais se aproxima da partição de referência fornecida no repositório. Mesmo que isso esteja bem definido teoricamente, isso não faz sentido na prática, pois não haveria necessidade de utilizar um algoritmo de clusterização se a melhor partição já fosse

Tabela 5.1: Conjuntos de dados da UCI.

Data set	#Instâncias	#Atributos	#Clusters
Iris	150	4	3
Wine	178	13	3
Glass	214	9	6
Ionosphere	351	34	2
Sonar	208	60	2
Breast Cancer	683	9	2
WDBC	568	30	2
Ecoli	336	7	8
Seeds	210	7	3
Soybean	47	35	4

conhecida. Por isso, também comparamos a partição de referência com a partição mais compacta entre todos as partições obtidas pelo algoritmo, pois este pode ser calculado pelo usuário sem conhecimento prévio de uma partição correta.

Os resultados são apresentados nas Tabelas 5.2, 5.3, 5.4 e 5.5. As duas primeiras com relação ao ACC e as duas últimas com relação ao NMI. Cada entrada das tabelas tem dois valores, um que compara a partição de referência com a partição mais compacta obtida pelo algoritmo, e outro que compara a partição de referência com a partição obtida pelo algoritmo mais próxima da partição de referência (medido em termos de NMI).

De acordo com os resultados desses experimentos, o nosso algoritmo SC-HA obteve, em média, um desempenho melhor do que os outros algoritmos. Ele alcançou o melhor desempenho geral em quatro dos dez conjuntos de dados e, mesmo nos casos em que outros algoritmos tiveram melhor desempenho, os resultados não ficaram longe dos melhores valores.

Em suma, nesse trabalho propomos uma nova medida de similaridade, derivada do kernel Gaussiano. Essa medida de similaridade possui algumas de vantagens:

(1) Não é necessário definir nenhum parâmetro para utilizar a medida, a fazendo ser mais fácil de aplicar.

(2) É invariante sob translações e expansões.

(3) Ela pode ser calculada facilmente.

(4) De acordo com nossos experimentos, o desempenho do Algoritmo 5 utilizando essa medida de similaridade é semelhante ao desempenho alcançado pelos outros métodos (que são apresentados na Seção 5.1), mesmo que os outros métodos sejam executados para muitos valores de seu parâmetro de escala e a melhor solução seja selecionada.

Tabela 5.2: Resultados obtidos para cada algoritmo em função do índice ACC. Em cada entrada, o valor é o ACC entre a partição mais compacta obtida pelo algoritmo e a partição de referência do UCI.

Data set	SC-GK	SC-ST	SC-PB	SC-RPB	SC-DA	SC-NP	SC-HA
Iris	0.746	0.9	0.693	0.84	0.733	0.746	**0.96**
Wine	0.612	0.713	0.612	0.471	0.629	0.612	**0.730**
Glass	0.504	0.471	0.411	0.392	**0.630**	0.476	0.5
Ionosphere	0.509	**0.746**	0.717	0.723	0.538	0.541	0.740
Sonar	0.543	0.533	0.533	0.557	0.533	0.543	**0.586**
Breast Cancer	**0.967**	0.956	0.671	0.961	0.958	0.964	0.941
WDBC	0.640	0.827	0.633	0.845	0.676	0.640	**0.880**
Ecoli	0.583	**0.598**	0.467	0.380	0.407	0.571	0.541
Seeds	0.885	**0.914**	0.619	0.785	0.876	0.880	0.895
Soybean	0.723	0.765	0.723	0.702	0.723	**0.851**	0.808
Average	0.671	0.742	0.608	0.666	0.670	0.682	**0.758**

Tabela 5.3: Resultados obtidos para cada algoritmo em função do índice ACC. Em cada entrada, o valor é o ACC entre a partição obtida pelo algoritmo mais próxima da partição de referência (em termos de NMI) e a própria partição de referência.

Data set	SC-GK	SC-ST	SC-PB	SC-RPB	SC-DA	SC-NP	SC-HA
Iris	0.913	0.94	0.673	0.84	0.9	0.9	**0.96**
Wine	0.567	0.724	0.612	0.471	0.623	0.567	**0.730**
Glass	0.514	0.467	0.462	0.392	**0.630**	0.509	0.5
Ionosphere	0.709	0.746	**0.911**	0.723	0.663	0.740	0.740
Sonar	0.567	0.552	0.533	0.557	0.567	0.572	**0.586**
Breast Cancer	**0.970**	0.964	0.961	0.961	0.967	**0.970**	0.941
WDBC	0.640	**0.890**	0.637	0.845	0.676	0.640	0.880
Ecoli	**0.788**	0.592	0.470	0.380	0.547	0.693	0.538
Seeds	0.895	**0.914**	0.614	0.785	0.895	0.890	0.895
Soybean	0.765	0.765	0.723	0.702	0.723	**0.893**	0.808
Average	0.733	0.755	0.660	0.666	0.719	0.737	**0.758**

Tabela 5.4: Resultados obtidos para cada algoritmo em função do índice NMI. Em cada entrada, o valor é o NMI entre a partição mais compacta obtida pelo algoritmo e a partição de referência do UCI.

Data set	SC-GK	SC-ST	SC-PB	SC-RPB	SC-DA	SC-NP	SC-HA
Iris	0.704	0.777	0.624	0.678	0.703	0.704	**0.870**
Wine	0.404	**0.419**	0.404	0.167	0.418	0.404	0.414
Glass	0.419	0.391	0.332	0.300	**0.425**	0.359	0.407
Ionosphere	0.028	0.234	0.218	0.224	0.028	0.004	**0.286**
Sonar	0.005	0.021	0.004	0.015	0.003	0.005	**0.028**
Breast Cancer	**0.779**	0.744	0.027	0.767	0.736	0.764	0.722
WDBC	0.094	0.404	0.080	0.435	0.159	0.094	**0.487**
Ecoli	0.608	**0.624**	0.391	0.350	0.375	0.588	0.550
Seeds	0.679	**0.732**	0.459	0.527	0.635	0.646	0.676
Soybean	0.715	0.722	0.715	0.748	0.715	**0.771**	0.763
Average	0.444	0.507	0.326	0.421	0.417	0.434	**0.520**

Tabela 5.5: Resultados obtidos para cada algoritmo em função do índice NMI. Em cada entrada, o valor é o NMI entre a partição obtida pelo algoritmo mais próxima da partição de referência (em termos de NMI) e a própria partição de referência.

Data set	SC-GK	SC-ST	SC-PB	SC-RPB	SC-DA	SC-NP	SC-HA
Iris	0.794	0.824	0.669	0.678	0.777	0.777	**0.870**
Wine	**0.453**	0.437	0.404	0.167	0.431	**0.453**	0.414
Glass	0.448	0.420	**0.449**	0.300	0.425	**0.449**	0.407
Ionosphere	0.129	0.234	**0.590**	0.224	0.072	0.199	0.286
Sonar	0.014	**0.037**	0.004	0.015	0.010	0.014	0.028
Breast Cancer	0.796	0.779	0.767	0.767	0.779	**0.800**	0.722
WDBC	0.094	**0.500**	0.087	0.435	0.159	0.094	0.487
Ecoli	**0.678**	0.627	0.394	0.350	0.464	0.662	0.557
Seeds	0.694	**0.732**	0.472	0.527	0.694	0.700	0.676
Soybean	0.722	0.722	0.715	0.748	0.715	**0.847**	0.763
Average	0.482	**0.531**	0.455	0.421	0.453	0.499	0.521

6 CONSIDERAÇÕES FINAIS

Nesta dissertação, foi possível estudar métodos que podem ser utilizadas para o problema de otimização de classificação de dados. Nos aprofundamos em uma delas, a família dos métodos espectrais. Realizamos duas aplicações dos métodos espectrais em contextos reais, onde foi possível avaliar o desempenho desses métodos em situações complexas. Também desenvolvemos um novo método espectral, que utiliza uma medida de similaridade que não precisa de parâmetros de entrada. A seguir, listamos as contribuições dessa dissertação.

No Capítulo 2 apresentamos dois métodos utilizados para classificação de dados. Na Subseção 2.2.2 deste capítulo expomos o Algoritmo k-means que, provavelmente, é um dos algoritmos mais utilizados em problemas de classificação de dados. Apesar de muito conhecido, as demonstrações das suas propriedades fundamentais são raramente apresentadas na literatura. Nessa seção apresentamos essas demonstrações.

No Capítulo 3 trouxemos um apanhado da literatura em que explicamos a fundamentação teórica da clusterização espectral baseada em teoria de grafos e álgebra linear.

No Capítulo 4 apresentamos dois resultados que obtivemos ao longo dos dois anos de mestrado. Na Seção 4.1 utilizamos métodos espectrais para propor uma estratégia para montar um portfólio de ações com bom desempenho. Avaliamos essa estratégia durante três anos subsequentes e concluímos que ela fornece bons portfólios e é uma boa alternativa ao investidor. Os resultados obtidos nessa seção foram apresentados no CNMAC[1] (2021) e publicados no *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics* [66]. Na Seção 4.2

[1]Disponível em: http://www.cnmac.org.br/novo/index.php/CNMAC/conteudo/2021/40/85. Acesso em: 12 fev. 2022.

buscamos aplicar uma metodologia desenvolvida por Peixoto *et al.* [50] no estado do Rio Grande do Sul, com o objetivo de obter uma classificação de risco do estado e avaliar a qualidade dessa classificação. Esse trabalho foi desenvolvido em parceria com o prof. Pedro Peixoto, da USP. Os resultados desse trabalho foram apresentados no X ERMAC-RS[2] e estão disponíveis nos anais do evento. Foram selecionados 20 trabalhos do X ERMAC-RS para publicação em uma edição especial da revista Ciência e Natura, em que o nosso trabalho foi selecionado. Assim, uma versão estendida do nosso trabalho foi publicada na revista Ciência e Natura [67]. Ainda realizamos mais uma aplicação nesse período de dois anos, tendo por objetivo analisar a evolução da pandemia de COVID-19 no Rio Grande do Sul no ano de 2020, utilizando métodos espectrais. Esse trabalho está atualmente submetido para uma revista. Um recorte desse artigo foi apresentado no I ERMAC-MS[3].

No Capítulo 5 apresentamos um apanhado dos métodos espectrais relativos ao Ncut. Nossa contribuição nesse capítulo é a de um novo algoritmo espectral, a partir de uma medida de similaridade hierárquica. Um trabalho completo sobre o nosso algoritmo espectral ainda está em desenvolvimento.

Para trabalhos futuros, enxergamos pelo menos três direções possíveis:

1. Ampliar o universo de aplicações dos métodos espectrais em contextos reais. Em particular, planejamos estender nossa estratégia desenvolvida no mercado financeiro para outros ativos financeiros, como criptomoedas. Além disso, pretendemos avaliar o nosso algoritmo espectral, apresentado no Capítulo 5, em contextos reais.

2. Desenvolver novos métodos espectrais, explorando variações nos três passos gerais da estrutura desses métodos, apresentados na Seção 3.3.

[2]Disponível em: https://www.ufrgs.br/ermacrs2020/. Acesso em: 12 fev. 2022.
[3]Disponível em: https://www.sbmac.org.br/2020/07/i-ermac-ms/. Acesso em: 12 fev. 2022.

3. Explorar as maneiras existentes que auxiliam o usuário na definição do número de clusters k. Em particular, analisar critérios baseados em clusterização espectral que forneçam informações sobre o número de clusters.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Aloise, D., Deshpande, A., and Hansen, P. NP-hardness of Euclidean sum-of-squares clustering. *Mach Learn 75* (2009), 245–248.

[2] Arthur, D., and Vassilvitskii, S. How slow is the k-means method? *Association for Computing Machinery* (2006), 144–153.

[3] Arthur, D., and Vassilvitskii, S. k-means++: The advantages of careful seeding. *Society for Industrial and Applied Mathematics* (2007), 1027–1035.

[4] B3. Empresas listadas, 2022. Disponível em: `https://www.b3.com.br/pt_br/produtos-e-servicos/negociacao/renda-variavel/empresas-listadas.htm`. Acesso em: 4 fev. 2022.

[5] B3. Institucional, 2022. Disponível em: `https://www.b3.com.br/pt_br/b3/institucional/quem-somos/`. Acesso em: 4 fev. 2022.

[6] B3. Perfil pessoas físicas, 2022. Disponível em: `https://www.b3.com.br/pt_br/market-data-e-indices/servicos-de-dados/market-data/consultas/mercado-a-vista/perfil-pessoas-fisicas/perfil-pessoa-fisica/`. Acesso em: 4 fev. 2022.

[7] B3. Índice bovespa (Ibovespa B3), 2022. Disponível em: `http://www.b3.com.br/pt_br/market-data-e-indices/indices/indices-amplos/ibovespa.htm`. Acesso em: 4 fev. 2022.

[8] Bach, F. R., and Jordan, M. I. Learning spectral clustering, with application to speech separation. *The Journal of Machine Learning Research 7* (2006), 1963–2001.

[9] Bender, J., Briand, R., Melas, D., and Subramanian, R. A. Foundations of factor investing, 2013.

[10] Brasil. Lei No 6.404, de 15 de dezembro de 1976. *Diário Oficial [da] República Federativa do Brasil* (1976). Disponível em: `http://www.planalto.gov.br/ccivil_03/leis/l6404consol.htm`. Acesso em: 4 fev. 2022.

[11] Brauer, F. Compartmental models in epidemiology. *Mathematical Epidemiology* (2008), 19–79.

[12] Callahan, J. J. *Advanced calculus: a geometric view*, vol. 1. Springer, 2010.

[13] Cavers, M. S. *The normalized Laplacian matrix and general Randić index of graphs.* Faculty of Graduate Studies and Research, University of Regina, 2010.

[14] Celebi, M. E., Kingravi, H. A., and Vela, P. A. A comparative study of efficient initialization methods for the k-means clustering algorithm. *Expert Systems with Applications 40*, 1 (2013), 200–210.

[15] Chang, H., and Yeung, D. Robust Path-Based spectral clustering. *Pattern Recognition 41*, 1 (2008), 191–203.

[16] Chasanis, V. T., Likas, A. C., and Galatsanos, N. P. Scene detection in videos using shot clustering and sequence alignment. *IEEE transactions on multimedia 11*, 1 (2008), 89–100.

[17] Chen, W., and Feng, G. Spectral clustering with discriminant cuts. *Knowledge-Based Systems 28* (2012), 27–37.

[18] Defays, D. An efficient algorithm for a complete link method. *The Computer Journal 20*, 4 (1977), 364–366.

[19] Donath, W. E., and Hoffman, A. J. Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development 17*, 5 (1973), 420–425.

[20] Dua, D., and Graff, C. UCI Machine Learning Repository, 2019. Disponível em: `http://archive.ics.uci.edu/ml`. Acesso em: 4 fev. 2022.

[21] Fang, Y., and Wang, J. Selection of the number of clusters via the bootstrap method. *Computational Statistics & Data Analysis 56*, 3 (2012), 468–477.

[22] Fiedler, M. A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. *Czechoslovak mathematical journal 25*, 4 (1975), 619–633.

[23] Fischer, B., and Buhmann, J. Path-Based Clustering for grouping of smooth curves and texture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 25* (2003), 513–518.

[24] Forgy, E. W. Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *biometrics 21* (1965), 768–769.

[25] Frederix, K., and Van Barel, M. Sparse spectral clustering method based on the incomplete Cholesky decomposition. *Journal of Computational and Applied Mathematics 237*, 1 (2013), 145–161.

[26] Gitman, L. J., and Joehnk, M. D. *Fundamentals of Investing*. 1990.

[27] Guattery, S., and Miller, G. L. On the quality of spectral separators. *SIAM Journal on Matrix Analysis and Applications 19*, 3 (1998), 701–719.

[28] Hagen, L., and Kahng, A. B. New spectral methods for ratio cut partitioning and clustering. *IEEE transactions on computer-aided design of integrated circuits and systems 11*, 9 (1992), 1074–1085.

[29] Har-Peled, S., and Sadr, B. How fast is the k-means method. *Algorithmica 43*, 3 (2005), 877–885.

[30] Hastie, T., Tibshirani, R., and Friedman, J. *The elements of statistical learning*, 2nd ed. Springer, 2001.

[31] Horn, R. A., and Johnson, C. R. *Matrix analysis*. Cambridge university press, 2012.

[32] InfoMoney. Entenda como funciona o mercado de ações e a bolsa de valores, 2021. Disponível em: https://www.infomoney.com.br/guias/mercado-de-acoes/. Acesso em: 4 fev. 2022.

[33] Jia, H., Ding, S., Xu, X., and Nie, R. The latest research progress on spectral clustering. *Neural Computing and Applications 24*, 7 (2014), 1477–1486.

[34] Jr, W., and H, J. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association 58*, 301 (1963), 236–244.

[35] Kaufman, L., and Rousseeuw, P. J. *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009.

[36] Kriegel, H., Kröger, P., Sander, J., and Zimek, A. Density-based clustering. *WIREs Data Mining and Knowledge Discovery 1* (2011), 231–240.

[37] Landau, S., Leese, M., Stahl, D., and Everitt, B. S. *Cluster analysis*. John Wiley & Sons, 2011.

[38] Li, X., and Guo, L. Constructing affinity matrix in spectral clustering based on neighbor propagation. *Neurocomputing 97* (2012), 125–130.

[39] Linka, K., Peirlinck, M., Costabal, F. S., and Kuhl, E. Outbreak dynamics of COVID-19 in europe and the effect of travel restrictions. *Computer Methods in Biomechanics and Biomedical Engineering* (2020), 1–8.

[40] Liu, H., Zhao, F., and Jiao, L. Fuzzy spectral clustering with robust spatial information for image segmentation. *Applied Soft Computing 12*, 11 (2012), 3636–3647.

[41] Lloyd, S. Least squares quantization in PCM. *IEEE transactions on information theory 28*, 2 (1982), 129–137.

[42] Luo, D., Huang, H., Ding, C., and Nie, F. On the eigenvectors of p-Laplacian. *Machine Learning 81*, 1 (2010), 37–51.

[43] Macqueen, J. Some methods for classification and analysis of multivariate observations. *In 5-th Berkeley Symposium on Mathematical Statistics and Probability* (1967), 281–297.

[44] Mansano, R. E., Allem, L. E., Del-Vecchio, R. R., and Hoppen, C. Balanced portfolio via signed graphs and spectral clustering in the Brazilian stock market. *Quality & Quantity* (2021), 1–16.

[45] Ministério da Saúde. *Portaria 454, 20 de março de 2020.* Diário Oficial da União, 2020.

[46] Moses, E. A., and Cheney, J. M. *Investments: Analysis, Selection and Management.* 1989.

[47] Nataliani, Y., and Yang, M. Powered Gaussian kernel spectral clustering. *Neural Computing and Applications 31*, 1 (2019), 557–572.

[48] Newbould, G., and Poon, P. The minimum number of stocks needed for diversification, 1994.

[49] Ng, A. Y., Jordan, M. I., and Weiss, Y. On Spectral Clustering: Analysis and an algorithm. *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic* (2001), 849–856.

[50] Peixoto, P. S., Marcondes, D., Peixoto, C., and Oliva, S. M. Modeling future spread of infections via mobile geolocation data and population dynamics. An application to COVID-19 in Brazil. *PLoS ONE 15*, 7 (2020), 1–23.

[51] Pothen, A., Simon, H. D., and Liou, K.-P. Partitioning sparse matrices with eigenvectors of graphs. *SIAM journal on matrix analysis and applications 11*, 3 (1990), 430–452.

[52] Qin, G., and Gao, L. Spectral clustering for detecting protein complexes in protein–protein interaction (PPI) networks. *Mathematical and Computer Modelling 52* (2010), 2066–2074.

[53] Rebagliati, N., and Verri, A. Spectral clustering with more than k eigenvectors. *Neurocomputing 74*, 9 (2011), 1391–1401.

[54] Reis, T. Empresa de capital fechado tem características próprias. Saiba quais são, 2020. Disponível em: `https://www.suno.com.br/artigos/empresa-capital-fechado/`. Acesso em: 4 fev. 2022.

[55] Reis, T. Você sabe o que é uma empresa de capital aberto?, 2020. Disponível em: `https://www.suno.com.br/artigos/empresa-capital-aberto/`. Acesso em: 4 fev. 2022.

[56] Reis, T. IPO: como é feita a abertura de capital de uma empresa na bolsa?, 2021. Disponível em: `https://www.suno.com.br/artigos/ipo/`. Acesso em: 4 fev. 2022.

[57] Reis, T. O que são ações ordinárias, preferenciais e units?, 2021. Disponível em: `https://www.suno.com.br/artigos/o-que-sao-acoes-ordinarias-preferenciais-e-units/`. Acesso em: 4 fev. 2022.

[58] Reis, T. Lucro líquido: saiba o que é e como calculá-lo, 2022. Disponível em: `https://www.suno.com.br/artigos/lucro-liquido/`. Acesso em: 4 fev. 2022.

[59] Research, S. Guia completo sobre dividendos: o que são e como funcionam?, 2022. Disponível em: `https://www.suno.com.br/guias/dividendos/`. Acesso em: 4 fev. 2022.

[60] Rio Grande do Sul. *Decreto nº 55.128, de 19 de março de 2020.* Diário Oficial do Estado do Rio Grande do Sul, 2020.

[61] Sahoo, P. *Probability and Mathematical Statistics*. 01 2015.

[62] Sato, J. R., Takahashi, D. Y., Hoexter, M. Q., Massirer, K. B., and Fujita, A. Measuring network's entropy in ADHD: a new approach to investigate neuropsychiatric disorders. *Neuroimage 77* (2013), 44–51.

[63] Secretaria Estadual de Saúde (RS). *Painel Coronavírus RS*. Secretaria Estadual de Saúde, 2020.

[64] Senajith, E. D., Renuka, S. N., Neelika, M. G., and Janaka, S. H. An epidemiological model to aid decision-making for COVID-19 control in Sri Lanka. *PLoS ONE 15*, 8 (2020), 1–10.

[65] Shi, J., and Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 8 (2000), 888–905.

[66] Sibemberg, L. S., Allem, L. E., and Hoppen, C. Portfólios baseados em clusterização espectral e factor investing. *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics 8*, 1 (2021).

[67] Sibemberg, L. S., Allem, L. E., Hoppen, C., and da Silva Peixoto, P. Classificação de risco em redes complexas: o caso da COVID-19 no Rio Grande do Sul. *Ciência e Natura 43* (2021), 1.

[68] Sibson, R. SLINK: An optimally efficient algorithm for the single-link cluster method. *The Computer Journal 16*, 1 (1973), 30–34.

[69] Simon, H. D. Partitioning of unstructured problems for parallel processing. *Computing systems in engineering 2*, 2-3 (1991), 135–148.

[70] Sokal, R. R. A statistical method for evaluating systematic relationships. *Univ. Kansas, Sci. Bull. 38* (1958), 1409–1438.

[71] Stoer, M., and Wagner, F. A simple Min-Cut algorithm. *Journal of the ACM 44*, 4 (1997), 585–591.

[72] Strehl, A., and Ghosh, J. Cluster Ensembles – a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research 3* (2002), 583–617.

[73] Sugar, C. A., and James, G. M. Finding the number of clusters in a dataset: An information-theoretic approach. *Journal of the American Statistical Association 98*, 463 (2003), 750–763.

[74] Sun, J., Liu, J., and Zhao, L. Clustering algorithms research. *Journal of Software 19* (2008).

[75] Tepper, M., Musé, P., Almansa, A., and Mejail, M. Automatically finding clusters in normalized cuts. *Pattern Recognition 44*, 7 (2011), 1372–1386.

[76] VALOR. Só 44% dos fundos de ações ativos batem Ibovespa em 3 anos, 2019. Disponível em: `https://valor.globo.com/financas/noticia/2019/09/09/so-44-dos-fundos-de-acoes-ativos-batem-ibovespa-em-3-anos.ghtml`. Acesso em: 4 fev. 2022.

[77] van den Heuvel, M., Mandl, R., and Pol, H. Normalized cut group clustering of resting-state fMRI data. *PLoS one 3*, 4 (2008).

[78] von Luxburg, U. A tutorial on Spectral Clustering. *Statistics and computing 17*, 4 (2007), 395–416.

[79] Wagner, D., and Wagner, F. Between min cut and graph bisection. *The Computer Journal 711* (1993), 744–750.

[80] Wainberg, R. Patrimônio líquido: entenda o que é e como analisar esse indicador, 2022. Disponível em: `https://www.suno.com.br/artigos/patrimonio-liquido`. Acesso em: 4 fev. 2022.

[81] Wang, J. Consistent selection of the number of clusters via crossvalidation. *Biometrika 97*, 4 (2010), 893–904.

[82] Warnatz, J., Maas, U., and Dibble, R. W. *Combustion: Physical and Chemical Fundamentals, Modeling and Simulation, Experiments, Pollutant Formation*, 4th ed. Springer-Verlag Berlin Heidelberg, 2006.

[83] Wei, C., Yao, X., Gong, D., and Liu, H. Spectral clustering based mutant reduction for mutation testing. *Information and Software Technology 132* (2021).

[84] Wierzchoń, S. T., and Kłopotek, M. A. *Modern algorithms of cluster analysis*, vol. 34. Springer, 2018.

[85] Winger, B. J., and Frasca, R. R. *Investments: Introduction to Analysis and Plannig*. 1991.

[86] Wu, J. T., Leung, K., and Leung, G. M. Nowcasting and forecasting the potential domestic and international spread of the 2019-nCoV outbreak originating in Wuhan, China: a modelling study. *The Lancet 395* (2020), 689–697.

[87] Xiang, T., and Gong, S. Spectral Clustering with eigenvector selection. *Pattern Recognition 41*, 3 (2008), 1012–1029.

[88] Yoav, T., and Granek, R. Epidemiological model for the inhomogeneous spatial spreading of COVID-19 and other diseases. *PLoS ONE 16*, 2 (2021), 1–25.

[89] Zelnik-Manor, L., and Perona, P. Self-Tuning Spectral Clustering. *Advances in Neural Information Processing Systems (NIPS) 17* (2004).

[90] Zeng, S., Huang, R., Kang, Z., and Sang, N. Image segmentation using spectral clustering of Gaussian mixture models. *Neurocomputing 144* (2014), 346–356.

[91] Zeng, S., Sang, N., and Tong, X. Hand-written numeral recognition based on spectrum clustering. In *MIPPR 2011: Pattern Recognition and Computer Vision* (2011), vol. 8004, International Society for Optics and Photonics, p. 80040X.

[92] Zhang, X., Li, J., and Yu, H. Local density adaptive similarity measurement for spectral clustering. *Pattern Recognition Letters 32* (2011), 352–358.

[93] Zhao, F., Jiao, L., Liu, H., Gao, X., and Gong, M. Spectral clustering with eigenvector selection based on entropy ranking. *Neurocomputing 73*, 10-12 (2010), 1704–1717.