

Problema de Agrupamento Capacitado via Estimador de Densidade Kernel

Dirceu Scaldelai, Solange R. dos Santos, Gislaine Ap. Peričaro, Juliano F. da Mota
Colegiado de Matemática, Universidade Estadual do Paraná, Campus de Campo Mourão
Av. Comendador Norberto Marcondes, 733, Campo Mourão - Paraná - Brasil, CEP 87.302-060
dirceu.scaldelai@ies.unespar.edu.br

Luiz C. Matioli, Lucas G. Pedroso
Departamento de Matemática, Universidade Federal do Paraná
Av. Cel. Francisco H. dos Santos, 100 - J. das Américas, Curitiba-Paraná- Brasil, CEP 81.530-000

RESUMO

Neste trabalho, apresentamos um novo algoritmo de clusterização com restrição de capacidade sobre os clusters, denominado WKDE-CCP. Ele é baseado na otimização do estimador de densidade kernel ponderado para a determinação de centroides, seguida da alocação de observações aos clusters de forma a maximizar a dissimilaridade entre eles, respeitando a restrição de capacidade de cada cluster. Além disso, propomos um algoritmo de refinamento da solução, que utiliza a comparação e troca simples de observações entre os clusters com o objetivo de minimizar a soma das distâncias das observações aos respectivos centroides, garantindo uma melhor distribuição dos elementos. Experimentos numéricos foram conduzidos para ilustrar o desempenho e a versatilidade do algoritmo proposto, indicando que o Algoritmo WKDE-CCP é eficiente e competitivo quando comparado a outros algoritmos da literatura que utilizam restrições de capacidade em suas formulações. Características como simplicidade e eficiência tornam o algoritmo proposto um campo de pesquisa promissor.

PALAVRAS CHAVE. Estimador de Densidade Kernel. Otimização. Problema de clusterização com capacidade.

ABSTRACT

In this work, we present a novel clustering algorithm with capacity constraints on the clusters, named WKDE-CCP. It is based on the optimization of the weighted kernel density estimation for centroid determination, followed by the allocation of observations to clusters in a manner that maximizes dissimilarity among them, while respecting the capacity constraint of each cluster. Additionally, we propose a solution refinement algorithm that employs a simple comparison and exchange of observations between clusters, aiming to minimize the sum of distances from observations to their respective centroids, thereby ensuring a better distribution of elements. Numerical experiments were conducted to illustrate the performance and versatility of the proposed algorithm, indicating that the WKDE-CCP algorithm is efficient and competitive when compared to other algorithms in the literature that incorporate capacity constraints in their formulations. Features such as simplicity and efficiency make the proposed algorithm a promising research direction.

KEYWORDS. Kernel Density Estimator. Optimization. Capacitated Clustering Problem.

1. Introdução

A análise de cluster, ou clustering, é uma técnica fundamental no campo da Ciência de Dados e do Aprendizado de Máquina, utilizada para agrupar objetos semelhantes em grupos distintos com base em medidas de dissimilaridade [Mount e Zemel, 2019; Said e Torra, 2019]. Essa abordagem é amplamente aplicada em diversas áreas, como segmentação de clientes, processamento de imagens, diagnósticos médicos e sistemas de recomendação, destacando-se por sua capacidade de identificar padrões e estruturas ocultas em conjuntos de dados complexos [Aljarah et al., 2021; Gan et al., 2020]. No entanto, em muitos cenários práticos, a simples identificação de grupos não é suficiente; é necessário impor restrições adicionais para garantir que os grupos gerados atendam a requisitos específicos, como limites de capacidade ou tamanho.

Diante desse cenário, o Problema de Clusterização com Capacidade (CCP, do inglês *Capacitated Clustering Problem*), ganha relevância [Gnägi e Baumann, 2021]. O CCP é uma variação do problema clássico de clusterização, no qual são incorporadas restrições de capacidade aos clusters, garantindo que a soma de determinados atributos (como peso, demanda ou tamanho) dentro de cada grupo não ultrapasse um limite pré-definido [França et al., 1999]. Essa abordagem é particularmente útil em aplicações do mundo real, como planejamento logístico, design de redes de distribuição, alocação de recursos e organização de territórios, onde a capacidade dos clusters é um fator crítico para a eficiência e viabilidade das soluções.

Nesse contexto, propomos um novo algoritmo de agrupamento com restrição de capacidade máxima sobre os clusters, denominado “*Weighted Kernel Density Estimator for Capacitated Clustering Problem*”, ou simplesmente WKDE-CCP. O Algoritmo WKDE-CCP é um algoritmo de duas fases: A primeira é alicerçada na múltipla otimização da função KDE ponderada, para determinação de centroides com posterior alocação de observações aos clusters de forma a maximizar a dissimilaridade entre eles, respeitando o número pré-definido de cluster assim com a capacidade máxima de cada cluster. Já na segunda, é realizado um processo de refinamento da solução, obtida na primeira fase, que utiliza a comparação e troca simples de observações entre os clusters, desde que atendida a restrição de capacidade, com intuito de minimizar a soma das distâncias das observações aos respectivos centroides.

A utilização da otimização da função KDE como metodologia para o agrupamento de dados já foi empregada em outros trabalhos, como em Scaldelai et al. [2020, 2024]; Matioli et al. [2017]. De modo geral, o uso do KDE visa identificar regiões com alta densidade de observações, sendo o ponto de máximo da função um candidato a centroide do agrupamento. Nos estudos mencionados, todos os algoritmos desenvolvidos tinham como objetivo identificar clusters como regiões densamente povoadas por pontos no espaço de dados, separadas por áreas de baixa densidade, sem nenhuma restrição sobre o conjunto de dados ou sobre o agrupamento gerado. No entanto, quando se dispõe de informações sobre a estrutura dos clusters, como a demanda das observações e a capacidade dos clusters, é possível associar a característica da otimização da função KDE, que favorece a determinação de centroides em regiões densas de pontos, à imposição de restrições de capacidade para criar clusters densos, compactos e que atendam a essas restrições.

Nesse sentido, o Algoritmo WKDE-CCP foi estruturado e implementado no *Software R*, [R Core Team, 2019]. Experimentos numéricos foram realizados para explorar seu desempenho individualmente e/ou compara-lo a conjuntos Benchmarks. Assim, a estrutura deste artigo está organizada da seguinte forma: na Seção 2, descrevemos os elementos essenciais para o desenvolvimento desta pesquisa. Na Seção 3, estabelecemos o Algoritmo WKDE-CCP. Os experimentos numéricos são relatados na Seção 4. Finalmente, na Seção 5, considerações finais concluem nosso texto.

2. Preliminar

Esta seção fornece material de apoio necessário para a apresentação do trabalho. Para aprofundamento, sugerimos as seguintes leituras [Gramacki, 2018; Wand e Jones, 1994; Silverman, 1986; Scaldelai et al., 2020, 2024; França et al., 1999; Mulvey e Beck, 1984; Negreiros e Palhano, 2006a; Geetha et al., 2009]

2.1. Problemas de agrupamento capacitado

Os problemas de agrupamento capacitado (CCP) foram inicialmente propostos por Mulvey e Beck, visando a formação de clusters centrados em medianas representativas, respeitando capacidades predefinidas, com o objetivo de minimizar a soma das dissimilaridades entre os indivíduos e seus respectivos centros [Mulvey e Beck, 1984; Negreiros e Palhano, 2006a]. O CCP é classificado como um problema combinatório de alta complexidade, pertencente à classe NP-completo, tornando algoritmos exatos inviáveis para instâncias de grande porte, o que motiva o desenvolvimento de abordagens heurísticas [França et al., 1999]. Formalmente, o CCP pode ser modelado como um problema de programação inteira mista, no qual cada ponto é alocado a um centro de maneira a minimizar a soma das distâncias, respeitando a capacidade máxima de cada cluster [Negreiros e Palhano, 2006a; França et al., 1999].

A fim de apresentar a formulação matemática do CCP, vamos considerar a seguinte notação:

- $X = \{x_j \in \mathbb{R}^n | j = 1, \dots, m\}$ o conjunto de observações, onde m é o número de observações, n o número de características/atributos e x_j a j -ésima observação do conjunto X ;
- $\mathcal{C} = \{\mathcal{C}_i \subseteq X | i = 1, \dots, k\}$ o conjunto de clusters, onde cada $\mathcal{C}_i$ representa um grupo de observações;
- $U = \{u_{ij} | u_{ij} = 1 \text{ se } x_j \in \mathcal{C}_i \text{ e } u_{ij} = 0, \text{ caso contrário}, i = 1, \dots, k, j = 1, \dots, m\}$ o conjunto de atribuições;
- $\bar{C} = \{c_i \in \mathbb{R}^n | i = 1, \dots, k\}$ o conjunto dos centroides.
- $Q = \{q_j \in \mathbb{N}, j = 1, \dots, m\}$ o conjunto de demandas;
- $w \in \mathbb{N}$ a capacidade máxima de cada cluster.

Nessas condições, temos o modelo CCP dado por:

$$\begin{aligned}
 \min_{u, c} \quad & \sum_{i=1}^k \sum_{j=1}^m u_{ij} \|x_j - c_i\|^2 \\
 \text{s.a.} \quad & \sum_{i=1}^k u_{ij} = 1, \quad j = 1, \dots, m. \\
 & \sum_{j=1}^m q_j u_{ij} \leq w \quad i = 1, \dots, k. \\
 & u_{ij} \in \{0, 1\}, \quad i = 1, \dots, k, \quad j = 1, \dots, m.
 \end{aligned} \tag{1}$$

No modelo (1), a restrição $\sum_{i=1}^k u_{ij} = 1$ garante que cada observação esteja alocada em um, e apenas um, grupo. Por outro lado, a restrição $\sum_{j=1}^m q_j u_{ij} \leq w$ garante que cada cluster tenha no máximo a capacidade w . Essa última restrição, nos leva as restrições implícitas $q_j \leq w, \forall j$ e $\sum_{j=1}^m q_j \leq kw$.

2.2. Estimador de Densidade Kernel

O Estimador de Densidade Kernel (KDE, do inglês *Kernel Density Estimation*) é um método não paramétrico utilizado para suavizar dados a partir de amostras finitas. Essa técnica oferece uma abordagem simples para identificar padrões em conjuntos de dados sem a necessidade de assumir um modelo paramétrico pré-definido. De acordo com [Gramacki, 2018; Wand e Jones, 1994], o KDE é uma das ferramentas mais relevantes e amplamente aplicadas para suavização de dados, sendo frequentemente empregado na obtenção de soluções aproximadas.

De acordo com [Gramacki, 2018; Wand e Jones, 1994; Silverman, 1986], seja $x \in \mathbb{R}^n$ um vetor do espaço vetorial n -dimensional; $X = \{x_j \in \mathbb{R}^n | j = 1, \dots, m\}$ o conjunto de observações e $H \in \mathbb{R}^{n \times n}$ a matriz de largura de banda ou matriz de suavização, não aleatória, simétrica e positiva definida, o estimador de densidade kernel multivariado $\hat{f}$, com kernel Gaussiano, é dada por

$$\hat{f}(x, H) = m^{-1} (2\pi)^{-\frac{n}{2}} |H|^{-\frac{1}{2}} \sum_{j=1}^m \exp \left(-\frac{1}{2} (x - x_j)^T H^{-1} (x - x_j) \right). \quad (2)$$

Conforme Gramacki [2018], o KDE multivariado, dado por (2), pode ser interpretado como uma soma ponderada de “colisões” de densidade centradas em cada ponto x_j do conjunto de dados. Embora o termo soma ponderada seja utilizado, cada observação x_j possui o mesmo peso na construção do KDE, sendo a proximidade entre observações e a utilização do kernel Gaussiano os principais fatores que influenciam a ponderação da função. No entanto, em conjuntos de dados provenientes de um CCP, nos quais cada observação apresenta uma demanda específica, é possível adaptar a expressão da equação (2) para obter, de fato, uma soma ponderada.

Considerando o conjunto das demandas Q , para definir a função KDE ponderada, basta multiplicar cada kernel Gaussiano centrado em x_j pela respectiva demanda q_j , além de dividir a soma pelo inverso multiplicativo da soma das demandas, ou seja,

$$\hat{f}(x, H) = \left(\sum_{j=1}^m q_j \right)^{-1} (2\pi)^{-\frac{n}{2}} |H|^{-\frac{1}{2}} \sum_{j=1}^m q_j \cdot \exp \left(-\frac{1}{2} (x - x_j)^T H^{-1} (x - x_j) \right). \quad (3)$$

A função KDE dada em (3) depende apenas da variável $x \in \mathbb{R}^n$, uma vez que a matriz H estará fixada. Porém, escolher a matriz largura de banda ideal é um problema extremamente delicado, uma vez que, pequenas variações em seus elementos afetam significativamente a forma e a orientação axial do KDE.

Conforme mencionado por [Gramacki, 2018], existem três tipos principais de seletores de largura de banda: regras de ouro (ROT), métodos baseados na noção de validação cruzada (CV) e métodos baseados em *plugging*. Para esse algoritmo, optamos por utilizar a matriz H como sendo uma matriz diagonal com cada elemento definido pelo tipo ROT, visto que de acordo com [Wand e Jones, 1994] ela consiste em uma fórmula simples e facilmente computável fornecendo uma largura de banda “razoável” para uma ampla gama de situações. Em [Silverman, 1986; Gramacki, 2018], os autores apresentam uma forma otimizada para a escolha das componentes da matriz diagonal H . Em [Matioli et al., 2017; Scaldelai et al., 2020, 2024], é considerado um acréscimo de um escalar ao cálculo dessas componentes, com o objetivo de mitigar o excesso de suavização decorrente da determinação das larguras de banda sob a premissa de normalidade multivariada. Em nosso trabalho, considerando, ainda, que o problema em estudo envolve observações ponderadas, cada componente da matriz diagonal H é definida como

$$h_i = \alpha \left(\frac{4}{n+2} \right)^{\frac{1}{n+4}} \sigma_{w_i} \left(\sum_{j=1}^m q_j \right)^{-\frac{1}{n+4}}, \quad i = 1, \dots, n. \quad (4)$$

sendo σ_{w_i} o desvio padrão ponderado das componentes e $\sum_{j=1}^m q_j$ a soma das demandas.

A matriz H , cujas componentes são definidas em (4) e a função KDE, definida em (3), são a base para o desenvolvimento do algoritmo para clusterização com restrição de capacidade sobre os clusters e testes apresentados a seguir.

3. Algoritmo WKDE-CCP

Esta seção apresenta o Algoritmo WKDE-CCP (Weighted Kernel Density Estimator for Capacitated Clustering Problem), estruturado em duas fases complementares.

Na primeira fase (Algoritmo 1), realizamos a otimização de uma função KDE ponderada, selecionando centroides estratégicos e alocando as observações a clusters, de forma a maximizar a dissimilaridade entre grupos, respeitando a capacidade máxima permitida.

Na segunda fase (Algoritmo 2), a solução inicial é refinada por meio da realocação sistemática de observações, respeitando a restrição de capacidade, com o objetivo de minimizar a soma das distâncias entre observações e seus respectivos centroides, melhorando a qualidade da partição.

Algoritmo 1: WKDE-CCP (Fase 1)

Input: $X = \{x_j \in \mathbb{R}^n, j = 1, \dots, m\}$, $\alpha > 0$, $Q = \{q_j \in \mathbb{N}, j = 1, \dots, m\}$, $w \in \mathbb{N}$, $k \in \mathbb{N}$

1 **Set:** $\bar{X} = X$, $\bar{Q} = Q$, $\bar{k} = k$, $\bar{m} = m$

2 **for** $i = 1 : (k - 1)$ **do**

3 Defina: H e $\hat{f}$ como em (4) e (3);

4 Determine: $c_i \in \arg \max_{x \in \mathbb{R}^n} \{\hat{f}(x, H)\}$;

5 Calcule: $w' = \left\lfloor \frac{\sum_{j=1}^{\bar{m}} q_j}{\bar{k}} \right\rfloor$

6 Calcule: $d_j = \|x_j - c_i\|$, $j = 1, \dots, m$;

7 Determine a permutação σ de modo que os elementos de d sejam classificados em ordem crescente: $d' = (d'_1, d'_2, \dots, d'_{\bar{m}})$, onde $d'_1 \leq d'_2 \leq \dots \leq d'_{\bar{m}}$, e $d'_j = d_{\sigma(j)}$;

8 Reordene Q de acordo com σ : $Q' = (q'_1, q'_2, \dots, q'_{\bar{m}})$, where $q'_j = q_{\sigma(j)}$;

9 Determine: $t^* = \min \left\{ t \in \{1, 2, \dots, \bar{m}\} \mid w' \leq \sum_{j=1}^t q'_j \leq w \right\}$;

10 Defina:

11 $r^* = d'_{t^*}$;

12 $J = \{j \mid x_j \in \bar{X} \text{ e } \|x_j - c_i\| \leq r^*\}$;

13 $\mathcal{C}_i = \{x_j \mid j \in J \text{ e } x_j \in \bar{X}\}$;

14 $\bar{X} = \bar{X} \setminus \mathcal{C}_i$;

15 $\bar{Q} = \bar{Q} \setminus \{q_j \mid j \in J \text{ e } q_j \in \bar{Q}\}$;

16 $\bar{m} = \bar{m} - |J|$;

17 **end**

18 Define: $\mathcal{C}_k = \bar{X}$;

Output: $U = \left\{ u_{ij} \mid u_{ij} = \begin{cases} 1, & \text{se } x_j \in \mathcal{C}_i \\ 0, & \text{se } x_j \notin \mathcal{C}_i \end{cases}, i = 1, \dots, k, j = 1, \dots, m \right\}$

A fim de proporcionar uma compreensão mais clara do funcionamento do *Algoritmo WKDE-CCP*, apresentamos um exemplo, utilizando um número reduzido de observações e clusters. Para tanto, selecionamos uma amostra do conjunto de dados *SJC_100_10*¹, o qual também será empregado nos experimentos numéricos descritos na seção seguinte. Essa amostra é composta por 30 observações, cuja demanda totaliza 1358 unidades, e será particionada em três grupos, cada

¹Disponível em <http://www.lac.inpe.br/~lorena/instancias.html>

Algoritmo 2: WKDE-CAP (Fase 2)

Input: $U = \{u_{ij} \mid u_{ij} = 1 \text{ se } x_j \in \mathcal{C}_i, \text{ senão } u_{ij} = 0, \quad i = 1, \dots, k, j = 1, \dots, m\};$
 $X = \{x_j \in \mathbb{R}^n, j = 1, \dots, m\}; Q = \{q_j \in \mathbb{N}, j = 1, \dots, m\}; w \in \mathbb{N}.$

1 **Defina:** $c_i = \frac{\sum_{j=1}^m u_{ij} x_j}{\sum_{j=1}^m u_{ij}}; d_{ij} = \|x_j - c_i\|^2; f = \sum_{i=1}^k \sum_{j=1}^m u_{ij} d_{ij}; \Delta_f = \infty$

2 **while** $\Delta_f > 0$ **do**

3 $reduc = -1$

4 **while** $reduc < 0$ **do**

5 **Defina:** $\Delta = [\delta_{ij}]$, onde $\delta_{ij} = d_{ij} - \sum_{i'=1}^k u_{i'j} d_{i'j};$

6 $(i^*, j^*) = \arg \min_{i,j} \{\delta_{ij}\}; \text{reduc} = \min \{\delta_{ij}\}; \text{cap}_{i^*} = \sum_{j=1}^m u_{i^*j} q_j + q_{j^*}$

7 **if** $\text{cap}_{i^*} \leq w$ **e** $reduc < 0$ **then**

8 $u_{i^*j^*} = 1$ **e** $u_{ij^*} = 0, \forall i \neq i^*$

9 $\delta_{ij^*} = 0, \forall i = 1 : k$

10 **else**

11 $\delta_{i^*j^*} = 0;$

12 **end**

13 **end**

14 **for** $i = 1 : k - 1$ **do**

15 **for** $i' = i + 1 : k$ **do**

16 $reduc = -1$

17 **while** $reduc < 0$ **do**

18 **Defina:**

19 $j_1 = \arg \min_{j \in \{1:m\}} \{d_{i'j} - d_{ij} \mid u_{ij} = 1\}$ **e** $j_2 = \arg \min_{j \in \{1:m\}} \{d_{ij} - d_{i'j} \mid u_{i'j} = 1\}$

20 $\text{cap}_i = \sum_{j=1}^m u_{i^*j} q_j + q_{j_2} - q_{j_1}$ **e** $\text{cap}_{i'} = \sum_{j=1}^m u_{i'j} q_j + q_{j_1} - q_{j_2}$

21 $reduc = (d_{i'j_1} - d_{ij_1}) + (d_{ij_2} - d_{i'j_2})$

22 **if** $reduc < 0$ **then**

23 **if** $\text{cap}_i \leq w$ **e** $\text{cap}_{i'} \leq w$ **then**

24 $u_{ij_1} = 0$ **e** $u_{ij_2} = 1$

25 $u_{i'j_1} = 1$ **e** $u_{i'j_2} = 0$

26 **else**

27 $d_{ij_1} = d_{ij_2} = d_{i'j_1} = d_{i'j_2} = 0$

28 **end**

29 **end**

30 **end**

31 **end**

32 **end**

33 **Defina:** $f_{old} = f; c_i = \frac{\sum_{j=1}^m u_{ij} x_j}{\sum_{j=1}^m u_{ij}}; d_{ij} = \|x_j - c_i\|^2; f = \sum_{i=1}^k \sum_{j=1}^m u_{ij} d_{ij}; \Delta_f = f_{old} - f$

34 **end**

Output: $f, U = \{u_{ij} \mid u_{ij} = 1 \text{ se } x_j \in \mathcal{C}_i, \text{ senão } u_{ij} = 0, \quad i = 1, \dots, k, j = 1, \dots, m\}$

um com capacidade máxima de 475 unidades. A disposição espacial das observações, bem como a respectiva demanda associada a cada uma, pode ser visualizada na Figura 1a.

Como mencionado, o Algoritmo *WKDE-CCP* é composto por duas fases, sendo a pri-

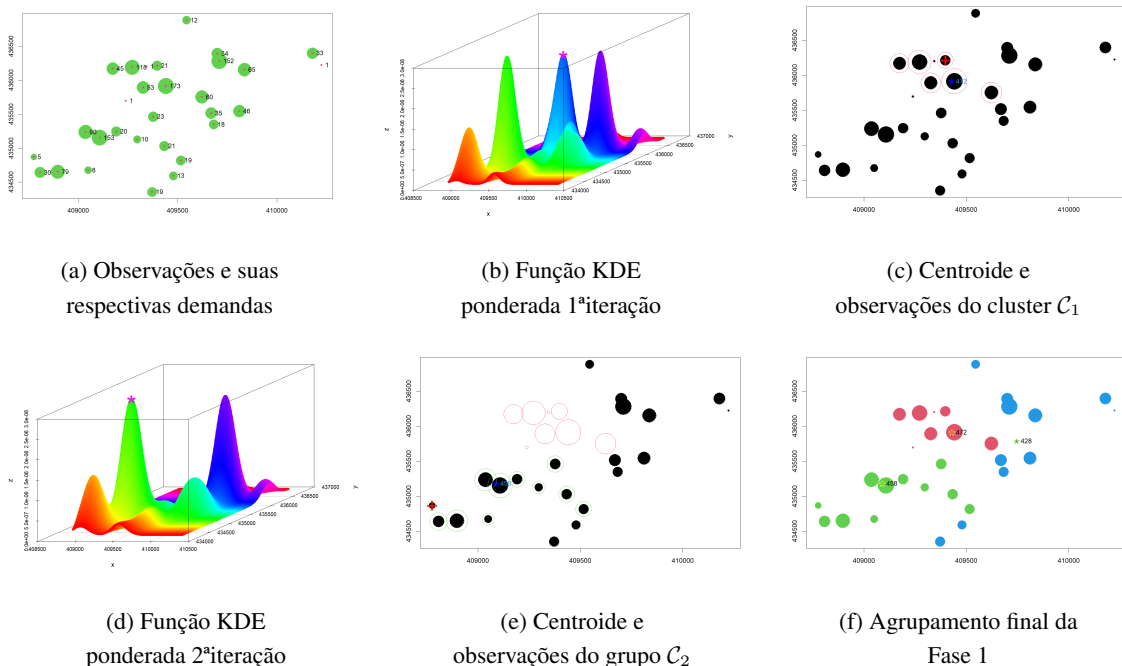


Figura 1: Ilustração do funcionamento da Fase 1 do Algoritmo WKDE-CCCP para o exemplo.

meira dedicada à obtenção de centroides pela múltipla otimização da função KDE ponderada. Para construção da função KDE, adotou-se o parâmetro $\alpha = 0.5$ na equação (4) para o cálculo da matriz de largura de banda, H , a qual, aplicada na equação (3) (linha 3, Algoritmo 1), gera a função cuja superfície é ilustrada na Figura 1b.

Na sequência, o *WKDE-CCP* seleciona um ponto do conjunto $\bar{X}$ para ser utilizado como ponto inicial no processo de otimização, representado pelo símbolo + em vermelho na Figura 1c. A partir desse ponto, é determinado o ponto de máximo da função KDE ponderada (linha 4), o qual corresponde ao centroide (c_1) do primeiro cluster, indicado pelo símbolo * em azul na Figura 1c e também pelo símbolo * em magenta na Figura 1b.

Paralelamente à obtenção do centroide, o algoritmo define, em cada iteração (linha 5), um limitante inferior para a capacidade do cluster a ser formado. Esse procedimento é necessário, pois o *WKDE-CCP* busca determinar exatamente o número de clusters especificado. Sem a imposição de um valor mínimo para a capacidade inicial, poderia haver subutilização do sistema, devido a capacidade máxima informada pelo usuário, resultando, ao final do processo, em menos clusters do que o desejado. Esse limitante inferior é calculado como o valor inteiro do quociente entre a demanda ainda não alocada e o número de clusters restantes. Na primeira iteração do exemplo, o limitante inferior é dado por $w' = \lfloor \frac{1358}{3} \rfloor = 452$. Ressalta-se que esse limitante inferior não constitui uma restrição rígida do problema, mas sim um mecanismo auxiliar para garantir a formação do número estipulado de clusters.

Uma vez definidos o centroide e o limitante inferior, o algoritmo procede à alocação das observações ao cluster C_1 . Inicialmente, calcula-se o vetor de distâncias entre todas as observações do conjunto $\bar{X}$ e o centroide c_1 (linha 6), ordenando-as em ordem crescente (linha 7). Com base nessa ordenação, o vetor de demandas também é reordenado (linha 8). Em seguida, realiza-se uma busca sequencial pelas observações mais próximas de c_1 , de modo a acumular uma demanda total situada entre o limitante inferior e o superior estabelecidos (linha 9). Assim, o algoritmo determina

o subconjunto de observações que atenderá à capacidade mínima requerida. No exemplo, as oito observações mais próximas de c_1 somam uma demanda de 472 unidades. Essas observações estão destacadas por uma circunferência vermelha na Figura 1c. Após a alocação (linhas 11 a 13), as observações designadas e suas respectivas demandas são removidas do conjunto de dados (linhas 14 e 15), encerrando a primeira iteração da Fase 1 do WKDE-CCP.

Na segunda iteração, o algoritmo repete o mesmo procedimento: define a função KDE ponderada e, por meio de técnicas de otimização, identifica um novo ponto de máximo, designado como centroide c_2 . Essas etapas podem ser observadas nas Figuras 1d e 1e. Ressalta-se que, na Figura 1e, apenas os pontos pretos foram utilizados na construção da função KDE; as circunferências vermelhas apenas ilustram as observações já alocadas na iteração anterior.

No início da segunda iteração, as observações remanescentes acumulavam uma demanda total de 886 unidades. Considerando que o objetivo é formar três clusters e um já foi definido, o limitante inferior é $w' = \lfloor \frac{886}{2} \rfloor = 443$. Com o centroide c_2 e o limitante inferior estabelecidos, o algoritmo aloca 11 observações ao cluster C_2 , totalizando uma demanda de 458 unidades, como indicado pelas circunferências verdes na Figura 1e, concluindo assim a segunda iteração da Fase 1.

Com o final da segunda iteração, e tendo o problema um total de 3 clusters, o cluster C_3 é formado por todas as observações ainda não atribuídas a nenhum cluster (linha 18). Para o exemplo, 11 observações não foram alocadas a nenhum cluster nas primeiras iterações, sendo que essas acumulam uma demanda de 428 unidades. Dessa forma, essas são atribuídas ao cluster C_3 finalizando assim o agrupamento da Fase 1 do Algoritmo WKDE-CCP, conforme Figura 1f.

Concluída a Fase 1, inicia-se a Fase 2 (Algoritmo 2), dedicada ao refinamento da solução obtida. Esta etapa consiste em um procedimento iterativo que busca realocar observações entre clusters de modo a reduzir o valor da função objetivo — definida como a soma das distâncias das observações aos seus respectivos centroides — sem violar a restrição de capacidade máxima de cada cluster. A cada iteração, o algoritmo atualiza os centroides, calcula a matriz de diferenças associadas às possíveis trocas de observações e identifica os melhores candidatos à realocação, priorizando as movimentações que promovem a maior redução na função objetivo.

No Algoritmo WKDE-CCP, a Fase 2 é dividida em dois loops: *troca simples* (linhas 4 a 13) e *permutação* (linhas 14 a 32). Na troca simples, verifica-se se uma observação pertencente a um cluster encontra-se mais próxima do centroide de outro cluster. Caso a realocação dessa observação proporcione uma redução no valor da função objetivo, a troca é considerada benéfica. No entanto, a realocação somente é realizada se o cluster de destino possuir capacidade ociosa suficiente para acomodar a demanda da nova observação; caso contrário, a observação permanece alocada em seu cluster original.

Na permutação, o algoritmo explora pares de clusters em busca de observações que, ao serem permutadas, resultem em uma redução do valor da função objetivo, respeitando simultaneamente as restrições de capacidade máxima de cada cluster. Esse procedimento permite ajustes mais refinados na solução, especialmente em situações nas quais a troca simples não é viável devido a limitações de capacidade.

Para o nosso exemplo, tomando como ponto de partida a solução da Fase 1 (Figura 1f) o Algoritmo WKDE-CCP realizou na sua primeira iteração 5 trocas simples (Figura 2a) e uma permutação (Figura 2b).

Na Figura 2a, as observações que mudaram de cluster são indicadas por um quadrado sobre o ponto correspondente. Por exemplo, a observação situada na região superior central do gráfico, originalmente pertencente ao cluster C_3 (cor azul), foi realocada para o cluster C_1 (cor vermelha), conforme indicado pela cor do quadrado. Como resultado das trocas, o cluster C_1 ganhou duas observações e perdeu uma; o cluster C_2 ganhou uma e perdeu uma; e o cluster C_3 ganhou duas

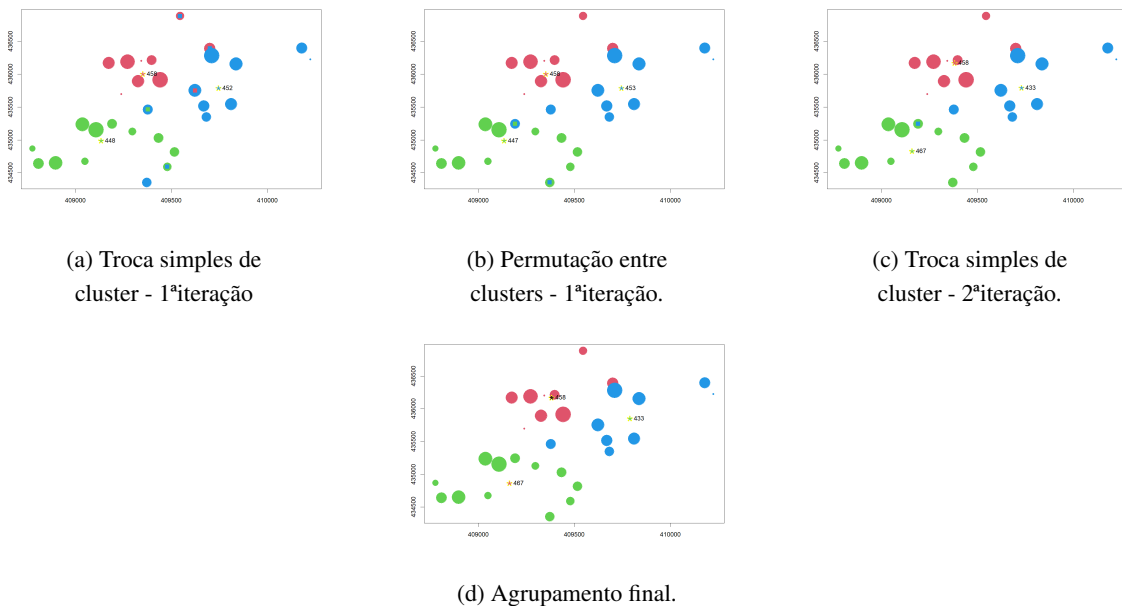


Figura 2: Detalhamento do processo de refinamento (Algoritmo WKDE-CCP Fase 2).

e perdeu três observações. Após essas realocações, as capacidades dos clusters passaram a ser de 458, 448 e 452 unidades, respectivamente.

Cabe destacar que, nesta etapa, embora sete observações estivessem mais próximas de centroides de outros clusters do que de seus próprios, apenas cinco trocas foram efetivamente realizadas. Essa diferença decorre de dois fatores inter-relacionados: a prioridade para trocas que promovem maior redução da função objetivo e a capacidade residual dos clusters de destino. O Algoritmo WKDE-CCP prioriza, em cada iteração, as trocas que proporcionam a maior diminuição na função objetivo. No entanto, mesmo que uma troca seja vantajosa do ponto de vista da distância, ela só é realizada se o cluster de destino possuir capacidade ociosa suficiente para acomodar a demanda da observação.

Um exemplo desse comportamento pode ser observado na Figura 2a, onde uma observação situada na região inferior central do gráfico, apesar de estar claramente mais próxima do centroide do cluster C_2 , não foi inicialmente realocada devido à limitação de capacidade do cluster no momento da avaliação. Posteriormente, após outras trocas liberarem capacidade, o cluster C_2 passou a ter capacidade ociosa suficiente para absorver essa observação. Esse comportamento justifica a necessidade de um processo de permutação entre clusters. A Figura 2b apresenta o resultado do processo de permutação entre clusters. No exemplo considerado, ocorreu uma única permutação entre observações dos clusters C_2 e C_3 .

Ao término da iteração, o Algoritmo WKDE-CCP atualiza os centroides, recalcula o valor da função objetivo e verifica se houve redução. Caso a função objetivo diminua, uma nova iteração é iniciada; caso contrário, o refinamento é encerrado. Em nosso exemplo, na segunda iteração, o algoritmo realizou apenas uma troca simples, conforme ilustrado na Figura 2c, que propiciou uma pequena redução na função objetivo, suficiente para iniciar a terceira iteração. No entanto, nesta iteração, não foram realizadas novas trocas ou permutações, e o valor da função objetivo permaneceu inalterado. A ausência de melhoria constituiu o critério de parada para o processo de refinamento, consolidando assim a solução final do Algoritmo WKDE-CCP, representada na Figura 2d.

4. Experimentos numéricos

Nesta seção, apresentamos resultados numéricos que evidenciam e qualificam o Algoritmo WKDE-CCP como um método eficaz para o problema de agrupamento com capacidades. O Algoritmo WKDE-CCP foi codificado no *software R* (versão 4.4.2) por meio da interface RStudio e os testes computacionais realizados em um notebook equipado com um processador Intel(R) Core(TM) i7-1255U de 12ª geração, com uma frequência de 1,70 GHz e 16 GB de memória RAM. O sistema operacional utilizado foi o Windows 11 Pro de 64 bits.

Os testes numéricos foram realizados sobre um *benchmark*² composto por quatro conjuntos de instâncias: dois conjuntos introduzidos por [Lorena e Senne, 2003] chamados “sjc” e “p3038”; dois conjuntos introduzidos por [Negreiros e Palhano, 2006b] chamados “ta” e “doni”, totalizando 25 instâncias. A Tabela 1 mostra as características das instâncias: nome, número de pontos (m), número de clusters (k), capacidade máxima dos clusters (w) e demanda total das observações (Q). Além de apresentar os valores *benchmark* para a mínima soma das distâncias entre os pontos do conjunto e seus respectivos centros (função objetivo, f_b) e tempo de execução (t_b) e os resultados do Algoritmo WKDE-CCP.

Para cada uma das instâncias apresentadas na Tabela 1, realizamos a renicialização do Algoritmo WKDE-CCP 20 vezes. Utilizamos o parâmetro $\alpha = 0.5$ na equação (4) para construção das funções KDE ponderadas (3). Apresentamos o resultado da execução que propiciou o menor valor da função objetivo (f) e o seu respectivo tempo de execução (t). Além disso, definimos duas medidas de comparação, sendo: $gap_f = \left(\frac{f - f_b}{f_b} \right) \cdot 100$, a diferença percentual entre a solução do WKDE-CCP e a solução *benchmark* e $speed_up = \frac{t_b}{t}$, o ganho de desempenho em tempo de execução do algoritmo proposto em relação ao tempo *benchmark*.

Com base nos resultados da Tabela 1, o Algoritmo WKDE-CCP obteve valores próximos dos melhores resultados conhecidos em várias instâncias testadas. Para instâncias menores, como “ta25”, “ta50”, “ta60” e “ta90”, o algoritmo conseguiu alcançar exatamente os valores ótimos já registrados, demonstrando sua precisão e eficiência na solução do problema.

Para instâncias de maior porte, como “doni6” e “p3038-1000”, o WKDE-CCP apresentou resultados competitivos, com *gaps* inferiores a 10% em relação às melhores soluções conhecidas. Além disso, o tempo de processamento foi significativamente menor que o *benchmark*, evidenciando a eficiência computacional do algoritmo proposto para problemas de grande porte.

Em resumo, os experimentos demonstraram que o WKDE-CCP é um algoritmo eficiente e competitivo para problemas de clusterização com restrições de capacidade. Sua abordagem baseada na otimização do estimador de densidade kernel ponderado permitiu encontrar soluções de qualidade em tempo reduzido, destacando-se como uma alternativa viável.

5. Considerações finais

Neste trabalho, apresentamos o Algoritmo WKDE-CCP para a solução do problema de clusterização capacitada. Diferente de abordagens clássicas, nosso método se baseia na otimização do estimador de densidade kernel ponderado, permitindo a identificação eficiente de clusters que respeitam restrições de capacidade máxima. O algoritmo foi implementado no *software R* e testado com um *benchmark*, demonstrando sua eficácia tanto em termos de qualidade das soluções quanto de tempo computacional.

Os resultados experimentais mostraram que o WKDE-CCP é competitivo, obtendo soluções próximas ou iguais aos melhores valores conhecidos para diversas instâncias. Além disso, o algoritmo apresentou um tempo de execução reduzido em comparação com outras abordagens, evidenciando sua viabilidade para aplicações práticas que demandam eficiência computacional.

²disponível em <https://sites.google.com/site/antoniochaves/publications/data>.

Tabela 1: Instâncias de Benchmark: Características e Resultados Experimentais Comparativos.

Instancias	Características				benchmark		WKDE-CCP			
	m	k	w	Q	f_b	$t_b(s)$	f	$gap_f(\%)$	$t(s)$	$speed_up$
ta25	25	5	6	25	1251.44	0.25	1251.44	0.0	0.26	0.96
ta50	50	5	11	50	4474.52	0.83	4474.52	0.0	0.58	1.43
ta60	60	5	13	60	5356.58	1.06	5356.58	0.0	0.99	1.07
ta70	70	5	17	70	6240.67	0.28	6247.41	0.1	1.15	0.24
ta80	80	7	12	80	5730.28	0.23	5739.71	0.2	0.65	0.35
ta90	90	4	23	90	9069.85	0.67	9069.85	0.0	0.49	1.33
ta100	100	6	17	100	8102.04	6.75	8106.76	0.1	0.35	19.29
sjc1	100	10	720	5807	17359.75	11.84	17573.04	1.2	1.42	8.33
sjc2	200	15	840	9269	33181.65	9.55	33394.25	0.6	1.66	5.75
sjc3a	300	25	740	11255	45366.35	73.11	48040.47	5.9	3.71	19.71
sjc3b	300	30	740	11255	40695.46	97.44	43075.25	5.9	3.01	32.37
sjc4a	402	30	840	15984	61944.85	81.06	65726.00	6.1	4.11	19.72
sjc4b	402	40	840	15984	52214.55	195.75	56179.53	7.6	5.17	37.86
doni1	1000	6	200	1000	3022.26	151.88	3046.63	1.0	1.93	76.69
doni2	2000	6	400	2000	6080.70	98.95	6396.53	5.2	7.09	13.96
doni3	3000	8	400	3000	8446.08	167.65	8523.77	2.2	68.17	2.46
doni4	4000	10	400	4000	10854.48	1294.51	10886.59	1.0	50.10	25.84
doni5	5000	12	450	5000	11134.94	4532.47	11379.39	2.4	64.18	70.62
doni6	10000	23	450	10000	15928.38	4022.72	16122.68	4.3	457.33	7.35
doni7	13221	30	450	13221	18484.13	1929.99	18951.63	2.5	540.24	3.57
p3038-600	3038	600	321	154482	129194.11	1263.08	137137.4	6.1	157.76	8.01
p3038-700	3038	700	273	152913	117295.47	12790.20	123243.3	5.1	260.62	49.08
p3038-800	3038	800	238	152693	109532.61	4630.17	117380.9	7.2	286.83	16.14
p3038-900	3038	900	216	155824	102458.94	34326.91	110106.1	7.5	271.4	126.48
p3038-1000	3038	1000	191	153240	97771.67	52530.25	106059.7	8.5	291.86	179.98

Fonte: Os autores (2025).

Entre as principais contribuições do trabalho, destacamos o uso da otimização da função KDE ponderada para guiar o processo de clusterização, bem como a estratégia de refinamento baseada na realocação e permutação de observações. Esses elementos contribuíram para a obtenção de clusters mais compactos e bem distribuídos, respeitando as restrições impostas.

Para trabalhos futuros, sugerimos a exploração de diferentes funções kernel e estratégias de ajuste da matriz de largura de banda, visando refinar ainda mais a identificação dos centroides. Além disso, a aplicação do WKDE-CCP em conjuntos multidimensionais pode intensificar sua qualidade como algoritmo de clusterização com restrição de capacidade.

Agradecimentos

O presente trabalho foi desenvolvido no âmbito do Núcleo de Inovação e Pesquisa em Matemática Aplicada e Engenharia (Numeng), da Universidade Estadual do Paraná (Unespar), em parceria com o Programa de Pós Graduação em Matemática (PPGM) da Universidade Federal do Paraná (UFPR), e com apoio financeiro da Fundação Araucária.

Referências

Aljarah, I., Habib, M., Faris, H., e Mirjalili, S. (2021). Introduction to evolutionary data clustering and its applications. *Evolutionary Data Clustering: Algorithms and Applications*, p. 1–21.

- França, P. M., Sosa, N. M., e Pureza, V. (1999). An adaptive tabu search algorithm for the capacitated clustering problem. *International Transactions in Operational Research*, 6(6):665–678.
- Gan, G., Ma, C., e Wu, J. (2020). *Data clustering: theory, algorithms, and applications*. SIAM.
- Geetha, S., Poonthalir, G., e Vanathi, P. (2009). Improved k-means algorithm for capacitated clustering problem. *INFOCOMP Journal of Computer Science*, 8(4):52–59.
- Gnägi, M. e Baumann, P. (2021). A matheuristic for large-scale capacitated clustering. *Computers & operations research*, 132:105304.
- Gramacki, A. (2018). *Nonparametric kernel density estimation and its computational aspects*. Springer. ISBN 978-3-319-71687-9. DOI: 10.1007/978-3-319-71688-6.
- Lorena, L. A. N. e Senne, E. L. F. (2003). Local search heuristics for capacitated p-median problems. *Networks and Spatial Economics*, 3:407–419.
- Matioli, L. C., Santos, S., Kleina, M., e Leite, E. A. (2017). A new algorithm for clustering based on kernel density estimation. *Journal of Applied Statistics*, 45(2):347–366. DOI: 10.1080/02664763.2016.1277191.
- Mount, J. e Zumel, N. (2019). *Practical data science with R*. Simon and Schuster. ISBN 978-1-617-29587-4.
- Mulvey, J. M. e Beck, M. P. (1984). Solving capacitated clustering problems. *European Journal of Operational Research*, 18(3):339–348.
- Negreiros, M. e Palhano, A. (2006a). The capacitated centred clustering problem. *Computers & operations research*, 33(6):1639–1663.
- Negreiros, M. e Palhano, A. (2006b). The capacitated centred clustering problem. *Computers Operations Research*, 33(6):1639–1663. ISSN 0305-0548. URL <https://www.sciencedirect.com/science/article/pii/S0305054804003120>.
- R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2019. URL <https://www.R-project.org/>.
- Said, A. e Torra, V. (2019). *Data Science in Practice*. Springer. ISBN 978-3-319-97555-9.
- Scalldelai, D., Matioli, L., e Santos, S. (2024). Treekde: clustering multivariate data based on decision tree and using one-dimensional kernel density estimation. *Journal of Applied Statistics*, 51(4):740–758.
- Scalldelai, D., Matioli, L., Santos, S., e Kleina, M. (2020). Multiclusterkde: a new algorithm for clustering based on multivariate kernel density estimation. *Journal of Applied Statistics*, p. 1–24.
- Silverman, B. W. (1986). *Density estimation for statistics and data analysis*, volume 26. CRC press. ISBN 0-412-24620-1.
- Wand, M. P. e Jones, M. C. (1994). *Kernel smoothing*. Chapman and Hall/CRC. ISBN 978-0-412-55270-0.