Annals of Operations Research
https://doi.org/10.1007/s10479-026-07259-x

ORIGINAL RESEARCH

A p-median based approach to constrained clustering

Zhifeng Guo[1,2] · **Jesse R. O'Hanley**[3,4] · **Stuart Gibson**[5] · **Maria Paola Scaparra**[3,4]

Received: 24 January 2025 / Accepted: 30 April 2026

Abstract
Constrained clustering is a semi-supervised method for dividing multi-attribute datasets into groups of similar elements that incorporates restrictions on how data points should or should not be clustered together. In this study, we investigate the application of the p-median model coupled with instance-level constraints, specifically must-link and cannot-link constraint, for performing constrained clustering. A Lagrangian relaxation procedure is proposed to efficiently solve large problem instances. To further speed up solution times, a simple but highly effective variable reduction technique is devised for identifying a small set of potential cluster centers. We test our modeling framework and solution approach on a set of benchmark datasets and several real-world household electricity consumption datasets. We find that high-quality solutions with small optimality gaps can be obtained with moderate computational effort. In addition, analysis of the largest household electricity consumption dataset reveals that constrained p-median clusters have less extreme variability with respect to average energy usage, a more even spread of cluster sizes, and much less overlap between high- and low-income households within the same clusters compared to classic k-means clustering.

Keywords Semi-supervised learning · Constrained clustering · Instance-level constraints · p-median problem · Lagrangian relaxation · Household electricity consumption

1 Introduction

Clustering analysis is a widely employed unsupervised data mining technique for dividing multi-attribute datasets into groups of similar elements (Rokach & Maimon, 2005; Xu & Tian, 2015; Xu & Wunsch, 2005). It has application in a wide array of fields, such as image

✉ Jesse R. O'Hanley
j.ohanley@kent.ac.uk

1 School of Management, Hefei University of Technology, Hefei, China

2 Anhui Provincial Key Laboratory of Philosophy and Social Sciences for Smart Management of Energy & Environment and Green & Low Carbon Development, Hefei, China

3 Kent Business School, University of Kent, Canterbury, UK

4 Centre for Logistics and Sustainability Analytics, University of Kent, Canterbury, UK

5 School of Physical Sciences, University of Kent, Canterbury, UK

segmentation (Coleman & Andrews, 1979), molecular biology (Nugent & Meila, 2010; Oyelade et al., 2016), customer relationship management (Ngai et al., 2009), recommendation systems (Phanich et al., 2010), social network analysis (Pham et al., 2011), and information retrieval (Leuski, 2001; McCallum et al., 2000). The main clustering algorithms commonly in use are: k-means clustering (MacQueen 1967), hierarchical clustering (Johnson, 1967), the Gaussian Mixture Model (McLachlan & Basford, 1988), spectral clustering (Ng et al., 2001), self-organizing maps (Kohonen, 1990), and Nonnegative Matrix Factorization (Paatero & Tapper, 1994). Clustering algorithms such as these can work either directly on the original set of features or new features generated through feature engineering (e.g., dimensionality reduction and feature aggregation).

Adding human intuition or subjectivity to a clustering algorithm can enhance its validity and the interpretability of clusters. In some cases, for example, an analyst may have prior domain knowledge regarding how elements should or should not be clustered together. Such information can come from experts, physical reasoning, or logical deduction. Semi-supervised clustering (aka constrained clustering) is a process of incorporating such information into a clustering algorithm with the aim of generating more logical clusters. Constrained clustering has been used in several domains and applications, such as text mining (Basu et al., 2004a, b), video object classification (Yan et al., 2004), and RNA sequencing data (Tian et al., 2021). For a recent survey on semi-supervised clustering, the reader is referred to Taha (2023).

In this paper, we investigate the use of the p-median problem (Hakimi, 1964) coupled with instance-level constraints for performing constrained clustering. Instance-level constraints place restrictions on cluster membership of data point pairs and include "must-link" and "cannot-link" constraints (Wagstaff et al., 2001). Respectively, these constraints require a given pair of points to be in or not be in the same cluster. An example where cannot-link constraints might be advantageous is to prevent high- and low-income customers from appearing in the same clusters when doing a customer segmentation analysis. Although consumption levels of the two groups may be similar, formulating rational customer sales and relationship management strategies may necessitate these two customer groups be treated separately. Conversely, in the context of image segmentation, it may be desirable to have nearby pixels grouped together, in which case must-link constraints should be included.

A p-median problem with instance-level constraints can be readily formulated as a mixed-integer linear program (MILP). To efficiently solve large problem instances with verifiable optimality bounds, we propose a Lagrangian relaxation algorithm. We further investigate an efficient variable reduction technique involving the use of standard k-means to identify a subset of potential cluster centers that significantly reduces computational times, while still producing demonstrably good solutions.

Research on constrained clustering can be broadly divided into machine learning and mathematical programming based approaches (González-Almagro et al., 2023), with the vast majority of studies falling into the former category. Among the more well-known machine learning algorithms for incorporating instance-level constraints are COP-KMeans (Wagstaff et al., 2001), PCKMeans (Basu et al., 2004a), HMRF-KMeans (Basu et al., 2004b) and CMWK-Means (de Amorim, 2012). Each of these extends the classic k-means model and has been shown to produce dramatic increases in cluster accuracy on both artificially constructed and real-world datasets. However, being in essence construction and local search heuristics, their degree of optimality is unknown and they need to be run repeatedly to ensure some level of confidence in solution quality. Additionally, while computational overhead may be low, not all guarantee that clustering constraints will be satisfied (e.g., PCKMeans and HMRF-KMeans).

Incorporation of constraints, other than instance-level constraints, has also been investigated in the machine learning literature. In Ng (2000), the k-means algorithm is modified to ensure that clusters all have a fixed number of elements. Bradley et al. (2000), meanwhile, add constraints to k-means clustering to avoid empty clusters or clusters with few points. Ge et al. (2007) extend this further by imposing minimum variance of clusters in addition to a minimum number of points in each cluster in order to find clusters which are balanced in terms of cardinality and variance.

Mathematical programming is another powerful tool for both unconstrained and constrained clustering. For unconstrained clustering, a number of approaches have been adopted, including MILP solved by a variety of exact and heuristic methods (Brusco, 2003; Sağlam et al., 2006; Xia & Peng, 2005), dynamic programming (Os & Meulman, 2004), and semi-definite programming (Aloise & Hansen, 2009; Peng & Wei, 2007). Constrained clustering has been addressed by the likes of Xia (2009) and Babaki et al. (2014). Xia (2009) proposed an adaptation of Tuy's cutting plane algorithm to find optimal solutions to k-means clustering with must-link and cannot-link constraints. Numerical experiments show the algorithm is not only able to find better solutions than COP-KMeans, but also efficiently solve medium to large problem instances involving many thousands of data points, usually in fractions of a second and only several minutes in the worst case. Babaki et al. (2014), meanwhile, propose the use of column generation to solve minimum sum-of-squares clustering with must- and cannot-link constraints as well as anti-monotone constraints other than cannot-link (e.g., constraints limiting the maximum cluster size or overlap with a subset of points). Although an exact method with guaranteed optimality bounds, the solution approach of Babaki et al. (2014) is computationally intensive and only tested on small instances involving less than 200 data points with at most 35 dimensions and 4 labels. More recently, Calvete et al. (2024) present a nonlinear mixed integer program and heuristic solution method to balance the size of clusters subject to a constraint on assignment distance.

Vinod (1969) was the first to recognize that the p-median problem could be used in unconstrained clustering. The p-median problem is a classic facility location model that seeks to locate p facilities in order to minimize the average distance between a set of customer points and their nearest assigned facility (Daskin & Maass, 2015). In the context of clustering, p-median distance corresponds to a dissimilarity measure between data points. The model is equivalent to the k-medoids model, well-known in the field of machine learning. The suitability of the p-median problem for unconstrained clustering has been investigated by various authors, including Narula et al. (1977), Mulvey and Crowder (1979), Klastorin (1985), Brusco and Köhn (2008), and Benati and García (2014).

To our knowledge, use of the p-median problem for constrained clustering has only been examined by Randel et al. (2019). The authors introduce an MILP formulation of the p-median problem with instance-level constraints and propose an efficient variable neighborhood search (VNS) heuristic. The heuristic is tested on a set of benchmark datasets generated by Xia (2009) with solution quality assessed by comparing against branch and bound (CPLEX). They find that the VNS heuristics is able to find optimal to near optimal solutions in a matter of seconds to minutes for instances with 600 or more data points.

In summary, different semi-supervised algorithms have their own advantages and disadvantages. For machine learning based approaches, algorithms are typically based on traditional clustering paradigms and adopt various strategies to incorporate instance-level constraints. Their solution quality is difficult to assess as no optimality bounds are available and they have rarely been compared against global optimization methods. In addition, machine learning algorithms are usually highly specialized in that they are designed to only handle must-link and cannot-link constraints, but cannot be easily generalized to incorporate

other types of constraints (e.g., anti-monotone constraints). Compared with machine learning algorithms, mathematical programming overcomes many of these disadvantages, but generally has a much higher computational overhead and, therefore, may be limited to tackling small problem sizes. Additional research is clearly needed on the development of alternative mathematical programming approaches capable of producing verifiable near-optimal solutions to large, constrained clustering problems. This includes decomposition techniques like Lagrangian relaxation.

The remainder of this paper is structured as follows. In the next section, we provide a mathematical formulation of our problem along with our proposed solution methodology. This is followed by a presentation of results of numerical experiments on a set of benchmark datasets and several real-world household electricity consumption datasets. We conclude with a critical discussion of the implications of our findings and propose avenues for future research.

2 Methodology

2.1 Constrained p-median model

Consider the following notation. Let N, indexed by i, j, and k, be a set of data points. The distance metric d_{ij} represents the dissimilarity between points i and j. The number of cluster centers (i.e., medians) is denoted by p. Set $\mathcal{ML}$ defines the pairs of points (i, j) such that i and j must be in the same cluster (i.e., must-link pairs), while set $\mathcal{CL}$ defines the pairs of points (i, j) such that i and j must be assigned to different clusters (i.e., cannot-link pairs). The decision variables of the problem are given below.

$$y_j = \begin{cases} 1 \text{ if data point } j \text{ is selected as a cluster center} \\ 0 \text{ otherwise} \end{cases}$$

$$x_{ij} = \begin{cases} 1 \text{ if data point } i \text{ is assigned to the cluster centered on point } j \\ 0 \text{ otherwise} \end{cases}$$

With this in place, an MILP formulation of the p-median problem with instance-level constraints, as originally proposed by Randel et al. (2019), is given as follows.

$$\min \sum_{i \in N} \sum_{j \in N} d_{ij} x_{ij} \tag{1}$$

$$s.t.$$

$$\sum_{j \in N} y_j = p \tag{2}$$

$$\sum_{j \in I} x_{ij} = 1 \quad \forall i \in N \tag{3}$$

$$x_{ij} \leq y_j \quad \forall i, j \in N \tag{4}$$

$$x_{ij} - x_{kj} = 0 \quad \forall (i, k) \in \mathcal{ML}, j \in N \tag{5}$$

$$x_{ij} + x_{kj} \leq 1 \quad \forall (i, k) \in \mathcal{CL}, j \in N \tag{6}$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j \in N \tag{7}$$

$$y_j \in \{0, 1\} \quad \forall j \in N \tag{8}$$

The objective (1) minimizes the sum of distances between all points and their assigned clusters. Constraint (2) requires that exactly p points be selected as cluster center. Equalities (3) ensure that each data point is assigned to exactly one cluster, while inequalities (4) require that data points only be assigned to selected cluster centers. Constraints (5) and (6) define the must-link and cannot-link constraints, respectively. More specifically, (5) states for points i and k that must belong to the same cluster, if $x_{ij} = 1$, then $x_{kj} = 1$ and vice versa. For points i and k that cannot be in the same cluster, (6) specifies if $x_{ij} = 1$, then $x_{kj} = 0$ and vice versa. Finally, constraints (7) and (8) impose binary restrictions on the decision variables. Note that without inequalities (5)–(6), the problem reduces to a standard p-median model.

Compared with k-means, a p-median type model has obvious advantages for constrained clustering. First and foremost, it can be solved directly to find a global optimum. As pointed out by Steinley (2015), other advantages include the fact that (1) centers are selected from the points in a dataset (as opposed to being the central point of a cluster), thereby allowing one to identify the most representative element of each cluster; (2) because dissimilarity between points is usually taken as Euclidean distance (instead of quadratic distance), it is typically more robust to outliers within a dataset; and (3) the flexibility of having a more generic predefined distance metric means that distances do not need to be constantly recomputed each time centers are updated and there is no imposition of having triangle inequality or distance symmetry violations.

On the downside, however, as the number of data points increases, the number of variables and constraints in a p-median model grows dramatically. This can quickly make even moderately sized datasets computationally intractable with an exact approach. To partially overcome the challenge of solving large problem instances, we present below a Lagrangian relaxation algorithm together with a simple variable reduction technique to help reduce the number of potential center points and further speed up the Lagrangian procedure.

2.2 Lagrangian relaxation

2.2.1 Lower bound problem

Lagrangian relaxation is a classic decomposition method in which complicating constraints are relaxed and included in the objective function as penalties. The relaxed problem becomes easier to solve, while still providing useful information such as an error bound on optimality.

In our implementation, we chose to relax equality constraints (3) with Lagrangian multipliers $\mu_i, \forall i \in N$. The result is the following dual problem:

$$z_{LR}(\boldsymbol{\mu}) = \max_{\boldsymbol{\mu}} \min_{\boldsymbol{x}, \boldsymbol{y}} \sum_{i \in N} \sum_{j \in N} \left(d_{ij} - \mu_i\right) x_{ij} + \sum_{i \in N} \mu_i \tag{9}$$

subject to constraints (2) and (4)–(8). For fixed values of the Lagrangian multipliers μ_i, the resulting minimization problem over variables x_{ij} and y_j can be solved using a branch and bound solver like CPLEX. This yields a lower bound to the original problem (1)–(8).

It is worth noting that a more natural choice would be to relax inequalities (4), which would decouple the original problem into two simpler subproblems involving the selection of cluster center variables y_j and cluster assignment variables x_{ij} separately. However, previous studies (Daskin & Maass, 2015) and preliminary tests on our part showed that relaxing (3) produced much tighter lower bounds, albeit at the expense of solving a much more difficult relaxed problem at each iteration.

2.2.2 Upper bound problem

To try and produce a feasible solution and valid upper bound to the original problem, we consider two approaches. The first is to fix the centers $y_j = 1$ selected in the relaxed dual problem and then use a branch and bound solver (e.g., CPLEX) to determine the x_{ij} assignment variables. Although guaranteed to find the best possible assignment of points to centers, the obvious disadvantage of this is the potentially long run times, especially for large problem instances.

A second way of finding a feasible upper bound is to use a heuristic for assigning data points to centers, with centers fixed based on the solution to the relax dual problem. For this, we implemented a modification of the approach proposed by Randel et al. (2019). The heuristic is divided into two main steps.

Step 1 Create super points from must-link constraints. Temporarily ignoring cannot-link constraints, assign each regular and super point to its nearest available center that does not violate any cannot-link constraints.

Step 2 Examine the list of point pairs with cannot-link constraint violations and try to restore feasibility by keeping one of the two points at its current center. For the point being kept, determine the net change in objective from moving all points having cannot-link constraint violations with the point in question to their next closest center without a constraint violation. Keep whichever point achieves the lowest net increase in objective and continually update the list of violated constraint pairs.

In Step 1, sample points involved in must-link constraints, which need to be assigned to the same cluster center, are merged into super points and the distances between the super points and all other points updated. For example, assuming points p_1 and p_9 have a must-link constraint, a new super point s_1 is created and the distance between s_1 and every other point p_k is changed to $d_{1k} = d_{1k} + d_{9k}$. Note that the choice of specifying point p_1 or p_9 as the super point root is arbitrary. For further details, please refer to Randel et al. (2019). Once super points have been created, Step 1 ends by assigning each point to its nearest cluster center, while temporarily ignoring cannot-link constraints.

Algorithm GR Greedy reassignment procedure for restoring problem feasibility

$V \leftarrow \emptyset$
for each $i \in N$
  $S_i \leftarrow \emptyset$
end for
for each cannot-link constraint $(i,k) \in \mathcal{CL}$
  if cannot-link constraint (i,k) violated **then**
    $V \leftarrow V \cup (i,k), S_i \leftarrow S_i \cup k, S_k \leftarrow S_k \cup i$
  end if
end for
for each $(i,k) \in V$
  $w_i = \sum_{\ell \in S_i} \Delta d(\ell), w_k = \sum_{\ell \in S_k} \Delta d(\ell)$
  if $w_i = \infty$ **and** $w_k = \infty$ **then**
    stop feasibility cannot be restored
  else if $w_i < w_k$ **then**
    for each $\ell \in S_i$
      reassign ℓ from $Y(\ell)$ to $O(\ell)$
      remove $(i,\ell) \in \mathcal{CL}$ or $(\ell,i) \in \mathcal{CL}$ from V
    end for
  else
    for each $\ell \in S_k$
      reassign ℓ from $Y(\ell)$ to $O(\ell)$
      remove $(k,\ell) \in \mathcal{CL}$ or $(\ell,k) \in \mathcal{CL}$ from V
    end for
  end if
end for

Step 2 involves recovering feasibility for the solution found in the first step. The procedure for doing this is outlined in Algorithm GR (GR for greedy reassignment). The notation for Algorithm GR is defined as follows. Let $S_i = \left\{\ell \in N | (i,\ \ell) \in \mathcal{CL},\ x_{ij} = x_{\ell j} = 1\right\}$ be the subset of points having cannot-link constraint violations with point i (note S_i never contains a cluster center given how Step 1 is implemented), let $Y(i) = \left\{j \in J' | x_{ij} = 1\right\}$ denote the cluster center of point i, and let $O(i) = \arg\min_{j \in J'} \left\{d_{ij} | \nexists k \in N,\ (i,\ k) \in \mathcal{CL},\ x_{kj} = 1\right\}$ be the nearest "open" center to point i that does not violate any cannot-link constraints, with $J' = \left\{j | y_j = 1\right\}$ being the set of p currently selected centers. The net cost of reassigning point i from $Y(i)$ to $O(i)$ is given by $\Delta d(i) = d_{iO(i)} - d_{iY(i)}$ if $O(i)$ exists, $\Delta d(i) = \infty$ otherwise.

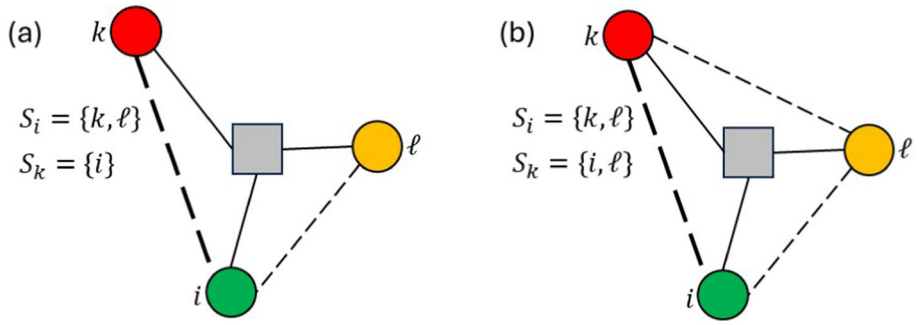

Fig. 1 Illustrative example of the bulk reassignment of multiple points to restore feasibility. Points i, k, and ℓ are all assigned to the same center (grey square). Dashed lines indicate there exists a cannot-link constraint between two points. The point pair being evaluated is (i, k) in both cases. In case (**a**), the options are either to keep point i at its current center and reassign the points in S_i to different centers or keep point k at its current center and reassign the points in S_k to different centers. For either move, no potential cannot-link violation can be created by moving points in S_i or S_k to their nearest open centers, including the same open center. In case (**b**), the options are the same, however, cannot-link violations will be created if all points in S_i are moved to the same open center or all points in S_k are moved to the same open center. (Color figure online)

In lines 1–10 of Algorithm GR, the various sets are initialized. This includes the set of violated cannot-link constraints V involving point pairs (i, k) along with the subset of points S_i and S_k having cannot-link constraint violations with points i and k, respectively. In lines 11–26, an attempt is made to iteratively remove all violated constraints in V. This is done by first computing the total net cost of keeping point i at its current center and reassigning points S_i to their next closest open centers versus keeping point k at its current center and reassigning points S_k to their next closest open centers (terms w_i and w_k, respectively, line 12). Note that the evaluation of reassignment costs is potentially order dependent in the sense that both functions $O(\cdot)$ and $Y(\cdot)$ need to be updated any time a point is temporarily or permanently reassigned. As such, total net cost terms w_i and w_k can vary depending on the sequence of points in summations $\sum_{\ell \in S_i} \Delta d(\ell)$ and $\sum_{\ell \in S_k} \Delta d(\ell)$ if and when potential constraint violations would be introduced by moving two points (ℓ, ℓ') in S_i or S_k to the same center, such that $(\ell, \ell') \in \mathcal{CL}$ (see Fig. 1). If neither point i nor point k can be kept at the current center (condition on line 13 is true), the algorithm stops (line 14). Otherwise, the move with the lower total net cost is determined (lines 15 and 20) and the reassignment of points S_i (lines 15–19) or S_k (lines 20–24) is carried out. Each time an individual point is reassigned (line 17 or 22), the set of violated constraints V is updated (lines 18 and 23).

Our algorithm for restoring feasibility differs from Randel et al. (2019). Whereas Randel et al. iteratively move a point violating at least one cannot-link constraint from its current center to its nearest open center until feasibility is restored (or not), in our implementation, we look at point pairs involved in violated constraints, making a greedy retainment of one of the points (while potentially involving the bulk reassignment of multiple points) and stopping after one complete pass. This has the advantage of potentially having fewer reassignments to restore feasibility and, therefore, shorter run times compared to Randel et al.

It is also worth noting that our restore feasibility routine, like Randel et al. (2019), is not guaranteed to produce a feasible solution. In the event this occurs, no feasible upper bound will be produced, but the Lagrangian relaxation algorithm can still continue using the best known upper bound to compute updated Lagrangian multipliers (see below).

2.2.3 Subgradient optimization

To find the best lower bound possible, we used subgradient optimization to update the Lagrangian multipliers in an iterative fashion. Let z^*_{UB} and $z_{LR}(\boldsymbol{\mu}^t)$ denote the values of the best known upper bound and the current lower bound at iteration t, respectively, and let x^t_{ij} be the partial solution to the relaxed problem at iteration t (i.e., the values of the cluster assignment variables). Multipliers were updated at every iteration according to the rule:

$$\mu_i^{t+1} = \mu_i^t - \alpha^t \left(\sum_{j \in J} x_{ij}^t - 1 \right) \tag{10}$$

$$\alpha^t = \frac{\theta^t \left(z^*_{UB} - z_{LR}(\boldsymbol{\mu}^t) \right)}{\sum_{i \in N} \left(\sum_{j \in N} x_{ij}^t - 1 \right)^2} \tag{11}$$

where μ_i^t is the multiplier value at iteration t, α^t it step size at iteration t, and θ^t is a user-defined constant, which, in the usual way, is initially set to 2 and divided by 2 if the lower bound fails to improve after a set number of iterations, in our case 3 iterations.

Starting values for the Lagrangian multipliers were all set to a fixed value:

$$\overline{d} = \sum_{i \in N} \sum_{j \in N} \frac{d_{ij}}{|N|^2} \tag{12}$$

This is simply the average distance between each pair of points. It is used for normalization purposes. Lastly, stopping conditions for the Lagrangian procedure included whether: (i) a maximum number of iterations t_{max} was reached; (ii) the relative difference between the upper and lower bounds was below some threshold ε; and (iii) the θ^t tuning parameter was less than some threshold θ_{min}. In our experimentation, we set $t_{max} = 200$ and $\varepsilon = 0.0001$ (i.e., 0.01%). Parameter θ_{min} was changed depending on the dataset, $\theta_{min} = 0.002$ for the benchmark datasets and $\theta_{min} = 0.0002$ for the medium and large UK household electricity consumption datasets (see below for details).

2.3 Variable reduction technique

To reduce the computational burden of having to solve very large MILPs for datasets of medium/large size, we adopted a two-stage solution approach (see Fig. 2). In the first stage, unconstrained cluster centers are identified using k-means or some other similarly fast algorithm (e.g. k-medoids). The points closest to each k-means cluster center (or the actual centroids for k-medoids) are then taken as the set of candidate medians M for a constrained p-median model, with $j \in M$ replacing $j \in N$ in (1)–(8). This reduced model is then solved in the second stage to find the final set of p cluster centers. The Lagrangian relaxation algorithm remains exactly the same, except for the denominator in Eq. (12), which changes from $|N|^2$ to $|N||M|$.

There is a lot of flexibility in how to apply k-means for selecting candidate centers. One approach is to simply use a single fixed value of $k > p$. For example, given a dataset with 1000 points, 250 must-link constraints, and 250 cannot-link constraints, if one wished to find $p = 10$ clusters and set $k = 30$, the number of variables and constraints would both decrease by two orders of magnitude, specifically going from 1,001,000 to 30,030 variables and from 1,501,001 to 46,001 constraints, which is a considerable reduction.

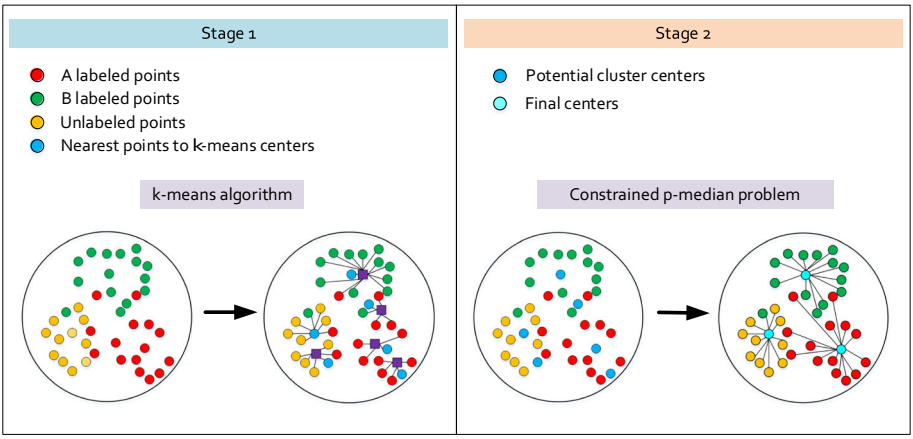

Fig. 2 Two-stage variable reduction technique for constrained clustering. Data points labeled A and B are assumed to have cannot-link constraints. Values for k and p are equal to 5 and 3, respectively

While this would undoubtedly make the constrained p-median problem much easier to solve, there are obvious concerns about the quality of the resulting solutions. One might expect, at least for some clusters, that many of the chosen cluster centers for a constrained p-median model would be near to the cluster centers obtained by unconstrained k-means with $k = p$. For $k > p$, however, it may be that none of the cluster centers for k-means are close in any sense to the $k = p$ centers, especially if k is only marginally larger than p.

To address this, more elaborate procedures might involve: (i) setting $k = p$, but instead of taking the points nearest to the k-mean cluster centers, using the $r \geq 2$ closest points to each center or (ii) using a series of increasing k values, starting with $k = p$, to iteratively build up a candidate set of constrained cluster centers. For the latter approach, different series of k values can be chosen, such as increments or multiples of p.

3 Experimental results

The optimization model and solution methods were implemented in Python using the docplex package and CPLEX callable libraries version 22.1.1. Runs of the k-mean algorithm were executed with the sklearn package for Python. All experiments were performed on the same multi-core Lenovo ThinkBook laptop (13th Gen Intel i7-13700H processor, 2.40 GHz per core) with 32 GB of RAM and running Windows 10 64-bit operating system.

3.1 Datasets

We tested the performance of the model, the variable reduction technique, and Lagrangian relaxation algorithm on set of 8 small-sized benchmark datasets (see Table 1), as well as several medium-sized and one large-sized UK household electricity consumption datasets. Benchmark datasets are the same ones used by Randel et al. (2019), thus allowing a direct comparison with their VNS heuristic. Details of the benchmark datasets are provided in Xia (2009).

Table 1 Summary of benchmark datasets

Instance	n	f	p	Configuration 1*		Configuration 2*	
				$\lvert\mathcal{ML}\rvert$	$\lvert\mathcal{CL}\rvert$	$\lvert\mathcal{ML}\rvert$	$\lvert\mathcal{CL}\rvert$
Soybean	47	35	4	2	12	4	2
Protein	116	20	6	9	6	13	9
Lris	150	4	3	6	6	8	4
Wine	178	13	3	22	13	36	22
Lonosphere	351	34	2	26	18	61	32
Control	600	60	6	30	15	45	30
Balance	625	4	3	78	47	109	63
Yeast	1484	8†	10	148	89	260	148

Respectively, parameters n, f, and p refer to the number of data points, the number of features, and the number of classes/clusters

**Note* the number of must- and cannot-link constraints reported in Xia (2009) and Randel et al. (2019) refer to the number of points involved, not pairwise restrictions, which are given here

† Correction to Xia (2009)

The household electricity consumption dataset derives from the UK Power Networks' Low Carbon London project (AECOM Building Engineering, 2018). The project, which ran from late 2007 to the end of 2010, measured electricity consumption at 30-min intervals in over 14,000 households across the UK. The dataset contains energy consumption time series (kWh per 30 min), unique household identifier, date and time, and Acorn household segmentation type (CACI, 2024). Acorn provides a geo-demographic breakdown of the UK's population (Table A.1). It divides postcodes into 62 types based on detailed social-economic characteristics then aggregates these into 18 intermediate groups and 6 top level categories (HM Land Registry, 2023). Acorn is designed to segment households based on lifestyle characteristics as opposed to property characteristic.

For our analysis, we created a number of datasets from the original UK Power Networks (UKPN) dataset. The large dataset contained household electricity consumption data for 8102 households with a full six months' worth of electricity consumption readings (24 July 2009–20 December 2009). This dataset (denoted UKPN-L) has a total of 150 features that includes Acorn types in the ranges 1–17 and 19–56. Note, there are only two type 55 households and one type 56 household. We attached one of three labels (aka classes) to each household based on their Acorn type. Households of type 1 were designated high-income and given a label "H", while households of type 52–57 were considered low-income and assigned a label "L". In all, there were 296 H-labeled and 187 L-labeled data points. All other households ($n = 7609$) were given a label "O" for other.

Must-link constraints were included to force a subset of high-income (alternatively low-income) households to appear in the same cluster. In all, we added 328 H-to-H and 131 L-to-L must-link constraints by selecting at random, with replacement, two H- and two-L labeled points, respectively, for a total of 459 must-link constraints. Specified numbers of must-link constraints are based on the formula $0.0075 \times n(n-1)/2+0.5$, which is simply 0.0075 times the total number of pairwise combinations rounded up to the nearest whole number, n being the number of data points for a particular label (i.e., 296 or 187). In a similar manner, 1384 random H-to-L cannot-link constraints were generated to prevent high- and low-income households from appearing in the same clusters. The number of cannot-link

Table 2 Summary of medium-sized UK Power Network datasets

Instance	$\lvert\mathcal{ML}\rvert$	$\lvert\mathcal{CL}\rvert$
UKPN-M1	102	310
UKPN-M2	106	329
UKPN-M3	112	342
UKPN-M4	119	336
UKPN-M5	103	287

All datasets have 4000 points, 150 features, and 3 classes

constraints is based on the formula $0.025 \times n_H \times n_L + 0.5$, where n_H and n_L are number of H- and L-labeled points (i.e., 296 and 187, respectively).

From the UKPN-L dataset, we subsequently generated 5 medium sized datasets (denoted UKPN-M1–UKPN-M5) with 4000 randomly selected points each (see Table 2). For the medium-sized datasets, cannot-link and must-link constraints were added for any pair of points that had such constraints in the large dataset.

3.2 Benchmark dataset results

3.2.1 Variable reduction technique

For the variable reduction technique, we evaluated a variety of rules for setting parameter k. These included using different single values for k (denoted SV1–SV3), specifically $k = 2p$ (SV1), $k = 5p$ (SV2), and $k = 10p$ (SV3), using the r nearest points to each k-means cluster for $k = p$ (denoted RN1-RN3), specifically $r = 4$ (RN1), $r = 6$ (RN2), and $r = 10$ (RN3), plus the following three ways of combining multiple values of k (denoted MV1–MV3).

- First four 1-step increments: $k = (p,\ p+1,\ p+2,\ p+3,\ p+4)$ (MV1)
- First three half p-step increments: $k = (p,\ p + p/2,\ p + 2 \cdot p/2)$ (MV2)
- First three multiplicative terms: $k = (p,\ 2p,\ 3p)$ (MV3)

Computational results confirm the overall effectiveness of the variable reduction technique (see Tables 3 and A.2–A.4). Rule MV2 achieved the best results in almost half of cases (7 out of 16) based on optimality gap and time. In the remaining cases, rules SV3, RN1, RN3, MV1, and MV3 performed best (1–2 times out of 16). Rules SV1, SV2, and SV3 performed relatively poorly based on average gap and rarely, if ever, obtained optimality. Overall, rule RN3 had the lowest average gap, followed closely by RN2 and MV1.

Irrespectively, variable reduction, in most cases, resulted in little or no degradation in solution quality. For five datasets (Soybean, Protein, Iris, Ionosphere, and Control), an optimal solution was found. For the other three datasets (Wine, Balance, and Yeast), optimality gaps were generally small, ranging from 0.2 to 0.7%. More impressively, solution times were radically reduced, especially for the four larger datasets. On Balance, for example, times were reduced from over 200 to 2 s or less. On Yeast, times were reduced from 88–365 min down to 5–27 s, representing a time savings of 3 orders of magnitude.

3.2.2 Lagrangian relaxation

Overall, the Lagrangian relaxation solution method performed well on the benchmark datasets (see Table 4). Interestingly, optimality gaps were the same regardless of whether CPLEX or

Table 3 Best performing variable reduction (VR) rule on benchmark datasets

Instance	Config.	Without VR	With VR		
		Time (s)	Best rule	Time (s)	Gap (%)
Soybean	1	0.11	RN1	0.05	0.00
	2	0.10	RN1	0.06	0.00
Protein	1	0.70	MV3	0.11	0.00
	2	0.50	MV1	0.11	0.00
Lris	1	2.62	MV2	0.06	0.00
	2	2.51	MV2	0.06	0.00
Wine	1	4.00	SV3	0.26	0.31
	2	5.33	SV3	0.19	0.22
Lonosphere	1	17.44	MV2	0.08	0.00
	2	16.83	MV2	0.10	0.00
Control	1	39.48	MV2	0.59	0.00
	2	38.43	MV2	0.56	0.00
Balance	1	242.16	RN3	2.05	0.38
	2	225.75	MV2	0.52	0.73
Yeast	1	21,900.85	RN3	26.94	0.31
	2	5285.78	MV1	4.95	0.65

Note, results reported here are based on solutions obtained with CPLEX

a heuristic was used to compute feasible upper bounds. Gaps ranged from 0 to 0.3% on the six smaller datasets (Soybean, Protein, Iris, Wine, Ionosphere, Control) and 0.7–1.8% for the largest two datasets (Balance and Yeast). Run times for the heuristic were, on average, half those with a CPLEX upper bound and ranged from under a second to several minutes.

In comparison to the VNS heuristic developed by Randel et al. (2019), our Lagrangian method matched it in terms of optimality gap 9 times (all 0%) and performed worse 6 times. At worst, the Lagrangian had a gap of 1.8%. In only one case did it outperform VNS (Ionosphere, configuration 2). Solution times for the Lagrangian were, in most cases, higher, but there were exceptions. Additionally, it is worth noting that VNS, unlike our approach, does not provide a way of computing an optimality gap. Hence for larger problems, which cannot be solved with a commercial solver like CPLEX, it is not possible to evaluate the solution quality of VNS.

3.3 Medium UK power networks dataset results

Results for the medium UKPN datasets merely confirm the overall effectiveness of the Lagrangian relaxation algorithm (Table 5). Optimality gaps were mostly less than 1%, exceeding 1% only one time. In the majority of cases, optimality gap increased as the number of clusters p increases, but not in a steady manner. In two cases, there was an increase in gap followed by a drop as p became larger. In the other three cases, the gap decreased then increased with increasing p.

Run times ranged from 4.4 to 12.6 min. The average was 8.3 min. In 7 out of 15 cases, the maximum number of iterations was reached. In the other 8 cases, 132 iterations were

Table 4 Performance of Lagrangian relaxation on benchmark datasets with best variable reduction rule (see Table 3) using CPLEX or heuristic GR to compute a feasible upper bound (UB) solution

Instance	Config.	CPLEX UB			Heuristic UB			VNS	
		Gap (%)	Itr	Time (s)	Gap (%)	Itr	Time (s)	Gap (%)	Time (s)
Soybean	1	0.00	30	2.02	0.00	30	2.33	0.00	0.00
	2	0.00	153	8.42	0.00	153	7.22	0.00	0.00
Protein	1	0.00	92	14.31	0.00	92	7.98	0.00	0.01
	2	0.00	112	16.56	0.00	112	9.92	0.00	0.00
Lris	1	0.00	27	2.14	0.00	27	1.32	0.00	0.01
	2	0.00	21	1.62	0.00	21	0.97	0.00	0.00
Wine	1	0.31	200	57.41	0.31	200	30.26	0.00	0.01
	2	0.22	200	64.44	0.22	200	35.36	0.00	7.08
Lonosphere	1	0.00	22	2.58	0.00	22	2.03	0.00	5.13
	2	0.00	20	2.48	0.00	20	2.30	0.02	65.04
Control	1	0.00	70	39.40	0.00	70	20.10	0.00	0.13
	2	0.00	63	37.04	0.00	63	18.82	0.00	0.12
Balance	1	1.48	51	52.22	1.48	51	26.25	0.002	110.42
	2	0.73	60	29.60	0.73	60	17.41	0.00	91.51
Yeast	1	1.84	72	554.54	1.84	72	264.94	0.00	97.78
	2	0.74	62	165.72	0.74	62	80.83	0.004	124.44

Columns under VNS are results reported for Randel et al. (2019)

Table 5 Performance of Lagrangian relaxation on medium-sized UKPN datasets using variable reduction rule MV2 and the heuristic method for computing a feasible upper bound

Instance	p	Best LB	Best UB	Gap (%)	Itr	Time (s)
UKPN-M1	6	196,317.9	197,201.5	0.45	200	412.02
	9	186,552.8	187,673.3	0.60	200	514.60
	12	180,287.2	181,114.3	0.46	200	757.78
UKPN-M2	6	195,304.4	196,300.2	0.51	121	270.01
	9	184,376.3	185,128.5	0.41	200	564.63
	12	178,901.4	180,386.6	0.83	200	731.18
UKPN-M3	6	194,358.2	195,157.9	0.41	122	265.80
	9	184,112.4	184,781.9	0.36	117	336.06
	12	178,164.3	179,480.6	0.74	169	628.06
UKPN-M4	6	194,385.2	194,624.5	0.12	115	273.46
	9	184,051.1	186,064.2	1.09	200	651.06
	12	179,699.6	180,290.5	0.33	200	738.66
UKPN-M5	6	192,704.9	193,354.1	0.34	114	295.97
	9	182,575.0	182,771.7	0.11	147	467.81
	12	176,934.6	177,733.9	0.45	152	579.35

required on average. Both run time and iterations generally show a positive relationship with the number of clusters p.

3.4 Large UK power networks dataset results

3.4.1 k-means clustering

Here we report detailed results comparing k-means clustering with our proposed constrained p-median clustering method on the large UK Power Networks dataset UKPN-L. We begin by computing silhouette scores to determine an 'optimal' number of clusters p (Rousseeuw, 1987) for annual load profiles. We find (Fig. 3) that the preferred number of clusters is in the range 15–17. For the convenience of visualization, we set $p = 15$ for both k-means and constrained clustering.

Results for k-means clustering (see Table 6 and Fig. A.1) reveal a number of distinct clusters that vary considerably in terms of the number of data points contained within them and daily average electricity consumption (measured in kWh/day). The two smallest clusters (no. 14–15), for example, contain just 18 and 6 households, respectively, which amounts to just 0.22% and 0.07% of the full sample. The largest cluster (no. 1), by contrast, contains 1473 data points, equivalent to 18.2% of the total.

Average daily usage for the five larger clusters (no. 1–5) is on the low side, ranging from 4.2 to 14.5 kWh/day. Meanwhile, the five medium-sized clusters (no. 6–10) have moderate average daily usage of 11.6–28.1 kWh/day. Finally, the five smaller clusters (no. 11–16) have moderate to very high average daily usage of between 21.8 and 67.9 kWh/day.

The results also reveal that in spite of there being a fairly large number of clusters, there is a significant overlap between high- and low-income households within most clusters (see Table 7 and Fig. A.1). To measure the degree of overlap within clusters, we computed a

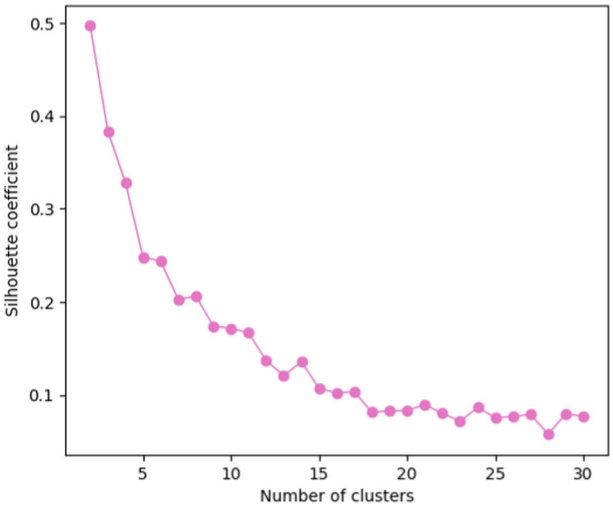

Fig. 3 Average silhouette coefficient versus number of clusters for the UKPN-L dataset

Table 6 Summary statistics for k-means clusters 1–15

Pattern	1	2	3	4	5
Count	1473	1464	1215	1074	904
Proportion (%)	18.18	18.07	15.00	13.26	11.16
Average usage (kWh/day)	8.91	6.63	11.52	4.23	14.45
Pattern	6	7	8	9	10
Count	544	392	307	227	204
Proportion (%)	6.71	4.84	3.79	2.80	2.52
Average usage (kWh/day)	17.83	22.76	18.91	11.56	28.09
Pattern	11	12	13	14	15
Count	136	70	68	18	6
Proportion (%)	1.68	0.86	0.84	0.22	0.07
Average usage (kWh/day)	21.76	29.59	38.15	45.98	67.85

Table 7 Number of H-, L-, and O-labeled households in each k-means cluster

Pattern	H	L	O	Q_H	Q_L	w	D
1	46	36	1391	0.561	0.439	0.170	0.122
2	23	42	1399	0.354	0.646	0.135	0.292
3	45	12	1158	0.789	0.211	0.118	0.579
4	13	62	999	0.173	0.827	0.155	0.653
5	35	11	858	0.761	0.239	0.095	0.522
6	30	3	511	0.909	0.091	0.068	0.818
7	30	3	359	0.909	0.091	0.068	0.818
8	22	2	283	0.917	0.083	0.050	0.833
9	8	4	215	0.667	0.333	0.025	0.333
10	18	8	178	0.692	0.308	0.054	0.385
11	4	2	130	0.667	0.333	0.012	0.333
12	4	0	66	1.000	0.000	0.008	1.000
13	12	0	56	1.000	0.000	0.025	1.000
14	4	2	12	0.667	0.333	0.012	0.333
15	2	0	4	1.000	0.000	0.004	1.000
						wD	0.510

dispersion index for each cluster j as follows.

$$D_j = \left|Q_{Hj} - 0.5\right| + \left|Q_{Lj} - 0.5\right| \tag{13}$$

In (13), Q_{Hj} and Q_{Lj} denote, respectively, the fractions of H- and L-labeled points in cluster j relative to the total number of H- and L-labeled points in the cluster. By assumption, if there are no H- or L-labeled points in cluster j, then $Q_{Hj} = Q_{Lj} = 0$. Metric D_j ranges from 1, indicating no overlap of H- and L-labeled points within the same cluster, down to 0,

indicating an even 50–50 split. The following weighted sum of the D_j's produces an overall measure of label overlap for a full clustering solution:

$$wD = \sum_{j=1}^{p} w_j D_j \tag{14}$$

where w_j is the fraction of H- and L-labeled points in cluster j relative to the total number of H- and L-labeled points in the whole dataset. Like D_j, total weighted dispersion wD approaches 1 if there is no overlap in any cluster.

According to our dispersion index, only clusters 12, 13, and 15 show no overlap ($D = 1$). The others either have very high overlap with $D < 0.385$ (no. 1, 2, 9–11, 14) or moderate to low overlap with D in the range 0.522–0.833 (no. 3–8). Clusters 10 and 14 are noteworthy for the fact they represent high average daily electricity consumption households with a significant proportion of low-income households.

3.4.2 Constrained p-median clustering

On the large UKPN dataset UKPN-L, the Lagrangian relaxation algorithm managed to produce a good quality solution in a reasonable amount of computing time (see Fig. 4 and Table A.5). Convergence of the best lower and upper bounds occurred quickly. Within 20 iterations, the optimality gap decreased from over 500% initially to under 6%. After 51 iterations, the gap was below 1% and at 116 iterations the best feasible upper bound was achieved, finally stopping at 148 iterations. The optimality gap at this point was 0.60%. In terms of run time, each iteration took 9.6 s on average. Total run time was a little under 24 min versus a matter of seconds for standard k-means. It is worth noting that dataset UKPN-L could not be solved using CPLEX even with variable reduction. It simply produced a memory error when trying to solve the linear relaxation.

Compared to k-means, constrained p-median clusters are qualitatively similar in that larger clusters (no. 1–11) tend to have low to moderate average energy usage of 3.6–23.9 kWh/day, while smaller clusters (no. 12–15) have moderate to high energy usage of 22.7–46.1 kWh/day (see Table 8 and Fig. A.2). What stands out, however, is that constrained p-median clusters

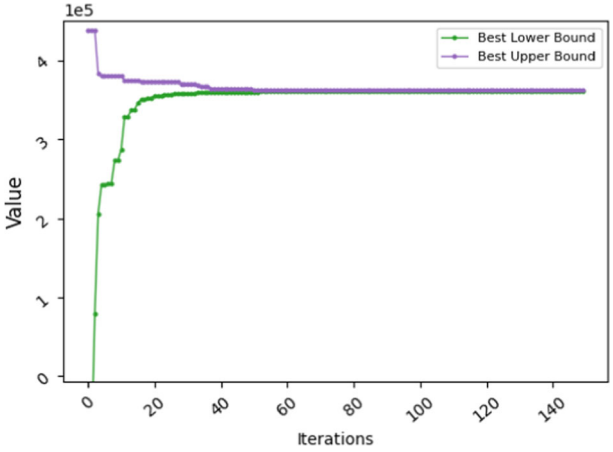

Fig. 4 Best lower and upper bound for Lagrangian relaxation of dataset UKPN-L

Table 8 Summary statistics for constrained p-median clusters 1–15

Pattern	1	2	3	4	5
Count	1129	1120	1011	976	838
Proportion (%)	13.93	13.82	12.48	12.05	10.34
Average usage (kWh/day)	14.99	6.86	9.77	11.84	8.18
Pattern	6	7	8	9	10
Count	763	552	494	306	288
Proportion (%)	9.42	6.81	6.10	3.78	3.55
Average usage (kWh/day)	5.16	18.06	3.58	23.89	20.00
Pattern	11	12	13	14	15
Count	220	159	136	70	40
Proportion (%)	2.72	1.96	1.68	0.86	0.49
Average usage (kWh/day)	14.45	22.70	30.17	31.39	46.08

have: (i) less extreme variability in terms of average energy usage; (ii) a slightly more even spread of cluster sizes (i.e., fewer clusters with extremely few data points); and (iii) far less overlap between high- and low-income households within most clusters (compare Tables 6 and 7 with Tables 8 and 9 and Fig. A.1 with Fig. A.2).

Regarding the first two points, daily average energy consumption for k-means ranges from 4.2 to 67.9 kWh/day. For the constrained p-median solution, that range is reduced considerably down to 3.6–46.1 kWh/day. Additionally, whereas the largest and smallest

Table 9 Number of H-, L-, and O-labeled households in each constrained p-median cluster

Pattern	H	L	O	Q_H	Q_L	w	D
1	260	0	869	0.538	1.000	0.000	1.000
2	0	108	1012	0.224	0.000	1.000	1.000
3	13	10	988	0.048	0.565	0.435	0.130
4	5	13	958	0.037	0.278	0.722	0.444
5	3	19	816	0.046	0.136	0.864	0.727
6	1	24	738	0.052	0.040	0.960	0.920
7	3	2	547	0.010	0.600	0.400	0.200
8	1	6	487	0.014	0.143	0.857	0.714
9	3	1	302	0.008	0.750	0.250	0.500
10	2	0	286	0.004	1.000	0.000	1.000
11	0	0	220	0.000	0.500	0.500	1.000
12	1	4	154	0.010	0.200	0.800	0.600
13	1	0	135	0.002	1.000	0.000	1.000
14	1	0	69	0.002	1.000	0.000	1.000
15	2	0	38	0.004	1.000	0.000	1.000
						wD	0.901

clusters for k-means contain 1473 points (18.2%) and a mere 6 points (0.07%), respectively, for the constrained clustering these values are 1129 (13.9%) and 40 (0.5%), respectively.

The most noticeable difference from k-means (point three above) is the much lower overlap of different label types within clusters (see Table 9 and Fig. A.2). With constrained clustering, the number of clusters with no overlap between H- and L-labeled points increases from 3 to 7 (i.e., $D = 1$ for cluster no. 1, 2, 10, 11, 13–15). Of the remaining clusters, 3 have fairly low overlap with $D \geq 0.714$ (no. 5, 6, 8), 3 have moderate overlap with $0.444 \leq D \leq 0.600$ (no. 4, 9, 12) and only 2 have significant overlap with $D \leq 0.2$ (no. 3, 7). Overall, dispersion (wD) is increased by 80% from 0.510 for the k-means solution to 0.901 for the constrained p-median solution.

4 Conclusion

This paper investigates the application of the p-median model combined with must-link and cannot-link constraints to the problem of constrained clustering. Unlike machine learning algorithms, which have traditionally been used in constrained clustering, our modeling approach has the benefit of producing solutions with verifiable optimality bounds. It can also be generalized to incorporate other types of instance-level constraints, such as a minimum or maximum number of points in each cluster and soft cannot-link constraints limiting the number of pairs of points with cannot-link constraints allowed in the same cluster.

A Lagrangian relaxation procedure is proposed to efficiently solve medium to large problem instances. Subgradient optimization is used in the usual way to update Lagrangian multipliers in the dual problem and a greedy based heuristic is used to find feasible upper bound solutions in each iteration. To speed up solution times, a simple but highly effective variable reduction technique is incorporated, which applies k-means (other similarly fast algorithm) to identify a candidate set of cluster centers.

We test our modeling framework and solution approach on a bank of benchmark datasets as well as several much larger real-world household electricity consumption datasets. We find that high-quality solutions with small optimality gaps, typically under 1%, can be obtained in seconds to minutes, even for problem instances with 8000 plus data points.

In terms of potential lines of future research, a particularly useful one would be to build and test new heuristics for generating feasible upper bound solutions as part of a Lagrangian relaxation procedure. Our heuristic (or at least our implementation), though adequate, is not especially fast and sometimes fails to find the best assignment of data points to cluster centers. Another direction of research would be to explore the adaptation of other solutions methods developed for the p-median problem to solve a constrained p-median problem, specifically methods that provide an optimality bound. This includes decomposition methods (e.g., Benders or Dantzig-Wolfe decomposition), as well as hybrid approaches mixing exact and metaheuristic methods. Alternative model formulations of the underlying p-median problem might also be considered, such as the COBRA (Church, 2003), BEAMR (Church, 2008), integer pseudo-Boolean (Goldengorin & Krushinsky, 2011), and radius constraint (García et al., 2011) models. Finally, while the problem instances used in our experiments all had very sparse cannot-link constraint sets (i.e., all or nearly all points were involved in at most one constraint), this may not always be the case in practice. For problem instances with more dense cannot-link constraint sets, a suitable approach may be problem formulations or solution techniques that rely directly on high-order clique constraints (Murray & Church, 1996, 1997).

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10479-026-07259-x.

Acknowledgements Z. Guo was supported by a University of Kent Vice Chancellor's Research Scholarship. This support is gratefully acknowledged.

Funding University of Kent, Vice Chancellor's Research Scholarship, Zhifeng Guo.

Declarations

Conflict of interest Z. Guo declares that he has no conflict of interest. J.R. O'Hanley declares that he has no conflict of interest. S. Gibson declares that he has no conflict of interest. M.P. Scaparra declares that she has no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

AECOM Building Engineering. (2018). Energy demand research project: Early smart meter trials, 2007-2010. UK Data Service. SN: 7591. https://doi.org/10.5255/UKDA-SN-7591-1.

Aloise, D., & Hansen, P. (2009). A branch-and-cut SDP-based algorithm for minimum sum-of-squares clustering. *Pesquisa Operacional, 29*, 503–516.

De Amorim, R. C. (2012). Constrained clustering with Minkowski weighted k-means. In: *2012 IEEE 13th international symposium on computational intelligence and informatics (CINTI), Budapest, Hungary, 20–22 November 2012* (pp. 13–17). IEEE.

Avella, P., Boccia, M., Salerno, S., & Vasilyev, I. (2012). An aggregation heuristic for large scale p-median problem. *Computers and Operations Research, 39*(7), 1625–1632.

Babaki, B., Guns, T., & Nijssen, S. (2014) Constrained clustering using column generation. In: *Integration of AI and OR techniques in constraint programming, 11th international conference, CPAIOR, Cork, Ireland, 19–23 May 2014, Proceedings* (pp. 438–454). Springer.

Basu, S., Banerjee, A., & Mooney, R. J. (2004a). Active semi-supervision for pairwise constrained clustering. In: *Proceedings of the 2004 SIAM international conference on data mining, Lake Buena Vista (FL), USA, 22–24 April 2004* (pp. 333–344). Society for Industrial and Applied Mathematics.

Basu, S., Bilenko, M., & Mooney, R. J. (2004b). A probabilistic framework for semi-supervised clustering. In: *Proceedings of the 10th ACM SIGKDD international conference on knowledge discovery and data mining, Seattle (WA), USA, 22–25 August 2004* (pp. 59–68). ACM.

Benati, S., & García, S. (2014). A mixed integer linear model for clustering with variable selection. *Computers and Operations Research, 43*, 280–285.

Bradley, P. S., Bennett, K. P., & Demiriz, A. (2000). Constrained k-means clustering. Technical Report MSR-TR-2000-652, Microsoft Research, Redmond (WA), USA, p. 8

Brusco, M. J. (2003). An enhanced branch-and-bound algorithm for a partitioning problem. *British Journal of Mathematical and Statistical Psychology, 56*(1), 83–92.

Brusco, M. J., & Köhn, H. F. (2008). Optimal partitioning of a data set based on the p-median model. *Psychometrika, 73*(1), 89.

CACI. (2024). Acorn: Powerful geodemographics, customer insight and resource targeting. CACI Limited. Available at: https://www.caci.co.uk/datasets/acorn/ Last Accessed from 15 Mar 2024.

Calvete, H. I., Galé, C., & Iranzo, J. A. (2024). Balancing the cardinality of clusters with a distance constraint: A fast algorithm. *Annals of Operations Research*. https://doi.org/10.1007/s10479-024-06017-1

Church, R. L. (2003). COBRA: A new formulation of the classic p-median location problem. *Annals of Operations Research, 122*(1–4), 103–120.
Church, R. L. (2008). BEAMR: An exact and approximate model for the p-median problem. *Computers and Operations Research, 35*(2), 417–426.
Coleman, G. B., & Andrews, H. C. (1979). Image segmentation by clustering. *Proceedings of the IEEE, 67*(5), 773–785.
Daskin, M. S., & Maass, K. L. (2015). The p-median problem. *Location science* (pp. 21–45). Springer International Publishing.
García, S., Labbé, M., & Marín, A. (2011). Solving large p-median problems with a radius formulation. *INFORMS Journal on Computing, 23*(4), 546–556.
Ge, R., Ester, M., Jin, W., & Davidson, I. (2007). Constraint-driven clustering. In: *Proceedings of the 13th ACM SIGKDD international conference on knowledge discovery and data mining, San Jose (CA), USA, 12–15 August 2007* (pp. 320–329). ACM.
Goldengorin, B., & Krushinsky, D. (2011). A computational study of the pseudo-Boolean approach to the p-median problem applied to cell formation. In: *Network optimization, 5th international conference on network optimization, Hamburg, Germany, 13–16 June 2011, Proceedings* (pp. 503–516). Springer.
González-Almagro, G., Peralta, D., De Poorter, E., Cano, J.-R., & García, S. (2023). Semi-supervised constrained clustering: An in-depth overview, ranked taxonomy and future research directions. arXiv 2303.00522.
Hakimi, S. L. (1964). Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations Research, 12*(3), 450–459.
Hansen, P., Brimberg, J., Urošević, D., & Mladenović, N. (2009). Solving large p-median clustering problems by primal–dual variable neighborhood search. *Data Mining and Knowledge Discovery, 19*, 351–375.
HM Land Registry. (2023). Acorn consumer classification (CACI). Official Statistics, Updated 22 Dec 2023. Available at: https://www.gov.uk/government/statistics/quality-assurance-of-administrative-data-in-the-uk-house-price-index/acorn-consumer-classification-caci Last Accessed from 15 Mar 2024.
Johnson, S. C. (1967). Hierarchical clustering schemes. *Psychometrika, 32*(3), 241–254.
Klastorin, T. D. (1985). The p-median problem for cluster analysis: A comparative test using the mixture model approach. *Management Science, 31*(1), 84–95.
Köhn, H. F., Steinley, D., & Brusco, M. J. (2010). The p-median model as a tool for clustering psychological data. *Psychological Methods, 15*(1), 87.
Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE, 78*(9), 1464–1480.
Leuski, A. (2001). Evaluating document clustering for interactive information retrieval. In: *Proceedings of the 10th international conference on information and knowledge management, Atlanta (GA), USA 5–10 October 2001* (pp. 33–40). ACM.
McCallum, A., Nigam, K., & Ungar L. H. (2000). Efficient clustering of high-dimensional data sets with application to reference matching. In: *Proceedings of the 6th ACM SIGKDD international conference on knowledge discovery and data mining, Boston (MA), USA 20–23 August 2000* (pp. 169–178). ACM.
McLachlan, G. J., & Basford, K. E. (1988). Mixture models: Inference and applications to clustering. *Journal of the Royal Statistical Society Series C: Applied Statistics, 38*(2), 384–385.
Mulvey, J. M., & Crowder, H. P. (1979). Cluster analysis: An application of Lagrangian relaxation. *Management Science, 25*(4), 329–340.
Murray, A. T., & Church, R. L. (1996). Analyzing cliques for imposing adjacency restrictions in forest models. *Forest Science, 42*(2), 166–175.
Murray, A. T., & Church, R. L. (1997). Solving the anti-covering location problem using Lagrangian relaxation. *Computers and Operations Research, 24*(2), 127–140.
Narula, S. C., Ogbu, U. I., & Samuelsson, H. M. (1977). An algorithm for the p-median problem. *Operations Research, 25*(4), 709–713.
Ng, A., Jordan, M., & Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. In: *Proceedings of the 14th international conference on neural information processing systems: Natural and synthetic, Vancouver, Canada, 3–8 December 2001* (pp. 849–856). MIT Press.
Ng, M. K. (2000). A note on constrained k-means algorithms. *Pattern Recognition, 33*(3), 515–519.
Ngai, E. W., Xiu, L., & Chau, D. C. (2009). Application of data mining techniques in customer relationship management: A literature review and classification. *Expert Systems with Applications, 36*(2), 2592–2602.
Nugent, R., & Meila, M. (2010). An overview of clustering applied to molecular biology. In: *Statistical methods in molecular biology* (pp. 369–404). Humana Press.
Os, B., & Meulman, J. (2004). Improving dynamic programming strategies for partitioning. *Journal of Classification, 21*(2), 207–230.

Oyelade, J., Isewon, I., Oladipupo, F., Aromolaran, O., Uwoghiren, E., Ameh, F., Achas, M., & Adebiyi, E. (2016). Clustering algorithms: Their application to gene expression data. *Bioinformatics and Biology Insights, 10*, 237–253.

Paatero, P., & Tapper, U. (1994). Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics, 5*(2), 111–126.

Peng, J., & Wei, Y. (2007). Approximating k-means-type clustering via semidefinite programming. *SIAM Journal on Optimization, 18*(1), 186–205.

Pham, M. C., Cao, Y., Klamma, R., & Jarke, M. (2011). A clustering approach for collaborative filtering recommendation using social network analysis. *Journal of Universal Computer Science, 17*(4), 583–604.

Phanich, M., Pholkul, P., & Phimoltares, S. (2010). Food recommendation system using clustering analysis for diabetic patients. In: *Proceedings of 2010 international conference on information science and applications, Seoul, Republic of Korea, 21–23 April 2010* (pp. 1–8). IEEE.

Randel, R., Aloise, D., Mladenovíc, N., & Hansen, P. (2019). On the k-medoids model for semi-supervised clustering. *Variable neighborhood search, 6th international conference on variable neighborhood search, Sithonia, Greece, 4-7 October 2018, revised selected papers* (pp. 13–27). Springer.

Rokach, L., & Maimon, O. (2005). Clustering methods. *Data mining and knowledge discovery handbook* (pp. 321–352). Springer.

Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics, 20*, 53–65.

Sağlam, B., Salman, F. S., Sayın, S., & Türkay, M. (2006). A mixed-integer programming approach to the clustering problem with an application in customer segmentation. *European Journal of Operational Research, 173*(3), 866–879.

Steinley, D. (2015). K-medoids and other criteria for crisp clustering. In: *Handbook of cluster analysis, Chapman and Hall/CRC handbooks of modern statistical methods* (pp. 55–68). CRC Press.

Taha, K. (2023). Semi-supervised and un-supervised clustering: A review and experimental evaluation. *Information Systems, 114*, 102178.

Tian, T., Zhang, J., Lin, X., Wei, Z., & Hakonarson, H. (2021). Model-based deep embedding for constrained clustering analysis of single cell RNA-seq data. *Nature Communications, 12*(1), 1873.

Vinod, H. D. (1969). Integer programming and the theory of grouping. *Journal of the American Statistical Association, 64*(326), 506–519.

Wagstaff, K., Cardie, C., Rogers, S., & Schrödl, S. (2001). Constrained k-means clustering with background knowledge. In: *Proceedings of the 18th international conference on machine learning, Williamstown (MA), USA, 28 June - 1 July 2001* (pp. 577–584). Morgan Kaufmann Publishers Inc.

Xia, Y., & Peng, J. (2005). A cutting algorithm for the minimum sum-of-squared error clustering. In: *Proceedings of the 2005 SIAM international conference on data mining, Newport Beach (CA), USA, 21–23 April 2005*. SIAM, pp. 150–160.

Xia, Y. (2009). A global optimization method for semi-supervised clustering. *Data Mining and Knowledge Discovery, 18*, 214–256.

Xu, D., & Tian, Y. (2015). A comprehensive survey of clustering algorithms. *Annals of Data Science, 2*, 165–193.

Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks, 16*(3), 645–678.

Yan, R., Zhang, J., Yang, J., & Hauptmann, A. (2004). A discriminative learning framework with pairwise constraints for video object classification. In: *Proceedings of the 2004 IEEE computer society conference on computer vision and pattern recognition* (pp. 1–8). IEEE.