# Clustering in Nucleolus Estimations for P2P Energy Exchange

An Analysis of Clustering Methods for Nucleolus Estimation

Philip Flatz-Stransky

Master of Science Thesis

Delft University of Technology

Delft Center for Systems and Control

# Clustering in Nucleolus Estimations for P2P Energy Exchange

**An Analysis of Clustering Methods for Nucleolus Estimation**

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Philip Flatz-Stransky

April 11, 2025

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

# Abstract

With the increasing electrification of domestic energy usage and the inherent demands on distribution networks, there are growing incentives for consumers to produce and consume energy locally. Cooperative strategies between consumers can alleviate upstream demand through scheduling the charging and discharging of battery systems on the basis of predicted knowledge of consumers' solar panel generation, consumption habits, and electricity prices. These consumers are called prosumers for their ability to consume and produce energy. Smart scheduling keeps generated energy stored for local consumption, and can help time prosumers' consumption of energy from the distribution networks when energy is cheap. One issue lies in the fair allocation of cost savings amongst prosumers. A well-suited solution concept, the nucleolus, becomes problematic to solve for larger numbers of prosumers. Building upon research done on estimating the nucleolus, this thesis investigates how different methods of clustering prosumers perform in nucleolus estimation.

# Table of Contents

# List of Figures

# List of Tables

# Acknowledgements

I would like to thank my supervisor Dr.ing. Sergio Grammatico for taking me on board, and for his assistance during the writing of this thesis. I would also like to thank my supervisor Dr.ing. Aitazaz Ali Raja for his time in guiding me with my thesis.

Delft, University of Technology
April 11, 2025

Philip Flatz-Stransky

Chapter 1

# Introduction to Peer-to-Peer Energy Markets

## 1-1 Motivations for P2P Energy Exchange

With increased availability of Distributed Energy Resources (DER), it has become attainable for many consumers to generate and store power at home, and be more proactive with their consumption with the help of smart devices. DERs are constituted of electricity generation sources, energy storage devices, and flexible loads [24]. In The Netherlands alone, 2018 saw an increase of 1500MW in electricity generation from solar panels, to a total of 4400MW [5]. The availability and decreasing cost of green technology that allows for local generation, energy storage and scheduling of loads can only point to an ever increasing adoption of such technologies on the consumer level.

For wider infrastructure-level solutions, Peer-to-Peer (P2P) energy markets are a class of designs that make consumer investments in DERs more effective by reducing costs of consumer coalitions beyond what would be possible with consumers acting independently. Currently, consumers can minimize costs by consuming energy during low rate periods, and by generating energy locally. This approach is inefficient, since consumers are limited to minimizing their own costs. Cost minimization of a P2P platform involves coordinating multiple DERs dynamically, minimizing cumulative costs of the P2P market community.

P2P markets are also proposed to reduce loads on electricity networks, which are only expected to increase due to, amongst others, the propagation of EV's and the phasing out of domestic gas use.

P2P markets can improve energy security by providing economic incentive for the purchase of additional DERs, reducing reliance on centralized sources. Upstream faults with energy suppliers or transmission networks are better dealt with by a P2P network that can to some extent meet the energy demands of its constituent prosumers (producer & consumer). Energy security becomes cheaper for wholesale suppliers because aggregate effects of prosumer loads show less variability [24] and are easier to estimate.

Prosumers are constrained to a top-down hierarchical approach as any external energy trading is done solely through conventional retailers which dictate prices. P2P networks and the decentralization of energy production liberalizes the market by reducing dependency on upstream suppliers. This turns a hierarchical market approach to a bottom up approach, potentially allowing prosumers to choose their energy suppliers on the basis of criteria such as the type of renewable energy, $CO_2$ emissions, and location of energy production.

In The Netherlands, it is generally considered that generating more energy than the prosumer consumes is financially detrimental given the lower returns on this excess generation [38], deincentivizing prosumers from having generation capacities larger than individual demand. Generated excess energy is sent upstream and deducted from the prosumer's bill at the original price. New legislation is expected to incrementally devalue any energy fed into upstream networks, which further incentivizes local use of generated energy. Such legislation feeds the need for aggregating prosumers into larger groups by means of effective management of local DERs.

Chapter 2

# The Nucleolus Solution Concept in P2P Energy Exchange

Game theory is a suitable technique that can be used for these kinds of allocation problems. In fact, one of the earliest examples of a game theoretic solution was the bankruptcy problem in the Babylonian Talmud ([2],[1],[33]). A solution to the contested garment problems in the Talmud are presented that coincide with the nucleolus, a cooperative game theory solution concept that will be evaluated in this thesis. [2] offers interesting insight into how these solutions were conceived considering the time they were written: the *equal division of the contested sum*, which is specifically mentioned in the classical text, and a physical interpretation, the Rule of Linked Vessels. This offers additional perspective on the meaning of the nucleolus.

The nucleolus combines some fairness criteria (*symmetry* and *dummy variable* axioms [33]) and the notion of stability [34]. These principles justify investigating nucleolus payoff allocations of the grand coalition cost savings versus allocations from distributed optimization ([25], [36]) where prosumers are assumed to not be capable of making tactical decisions, such as coordinating the defection and formation of subcoalitions (with possibly better outcomes for the defecting prosumers), or non-cooperative game theory, which generally suffers the same issue (besides [26] which gives outcomes in the core).

## 2-1 Cooperative Game & Nucleolus Solution Problem Formulations

### 2-1-1 Introduction

Han *et al.* [13] study the impact of a cooperating coalition of prosumers, some having Energy Storage (ES) systems, on the total coalitional costs. Due to generally lower feed-in tariffs (the commercial supplier buy-in price), it is preferred to schedule ES systems to retain energy rather than sell it off for a sub-optimal price. This is typically done in a non-cooperative fashion; each ES owner schedules ES operation based on their individual usage profile. This paper shows that coordinated ES operation is beneficial to the coalition cost. Energy management consists of centralized control using an optimization objective that minimizes coalitional costs. An additional mechanism is required to incentivize coordination by carefully establishing individual prosumer payoffs. To this purpose, Han *et al.* [13] choose the nucleolus and Shapley solution concepts for the allocation of coalition cost savings. Grand coalition load profiles are generated, showing that cooperative ES scheduling is beneficial in reducing loads on upstream distribution networks, with cooperative ES allowing for better local balance of energy supply and demand. The paper shows that cost reduction is also possible without ES, and that the game is balanced (which is proven in this papers extension [15]), meaning that a stabilizing (prosumer incentivizing) allocation of profits is possible, maintaining mutual interest amongst prosumers to form the coalition.

## 2-1-2 Cooperative Game Formulation

### Objective Function for a Coalition of Prosumers

Before defining the cost function, relevant variables are

$q_{it}$: Prosumer $i$ net (considering Photovoltaics (PV) generation and prosumer demand, disregarding ES operation) energy consumption (+) or generation (−), at timestep $t$, with unit [kWh].
$b_{it}$: Prosumer $i$ ES charging (+) or discharging (−) at timestep $t$ energy [kWh].
$p_t^b$: Price of energy bought by prosumers from the grid at timestep $t$, unit [£ kWh].
$p_t^s$: Price of energy sold by prosumers to the grid at timestep $t$, unit [£ /kWh].

For any coalition $\mathcal{S} \in 2^{\mathcal{N}}$ subset to the grand coalition $\mathcal{N}$ ($\mathcal{S} \subseteq \mathcal{N}$), the lowest coalitional energy coast $C(\mathcal{S})$ is such that the energy cost function $F_{\mathcal{S}}(\mathbf{b})$ is minimized, $\mathbf{b}$ being the set of vectors $\mathbf{b}_i$, each containing elements $b_{it}$ for each time step $t$. The authors state that the cost of purchasing energy to match net coalitional demand is $p_t^b \sum_{i\in\mathcal{S}} [q_{it} + b_{it}]^+$, and for an excess this is $p_t^s \sum_{i\in\mathcal{S}} [q_{it} + b_{it}]^-$.

This results in the objective function being (slightly mistakenly; prices should be applied to net coalitional loads rather than individual prosumer net loads) defined as

$$C(\mathcal{S}) = \min_{\mathbf{b}} F_{\mathcal{S}}(\mathbf{b}) = \min_{\mathbf{b}} \sum_{t=1}^{K} \left\{ p_t^b \sum_{i\in\mathcal{S}} [q_{it} + b_{it}]^+ + p_t^s \sum_{i\in\mathcal{S}} [q_{it} + b_{it}]^- \right\} \tag{2-1}$$

where the intent was most likely to define the cost function to be

$$C(\mathcal{S}) = \min_{\mathbf{b}} F_{\mathcal{S}}(\mathbf{b}) = \min_{\mathbf{b}} \sum_{t=1}^{K} \left\{ p_t^b \left[ \sum_{i\in\mathcal{S}} (q_{it} + b_{it}) \right]^+ + p_t^s \left[ \sum_{i\in\mathcal{S}} (q_{it} + b_{it}) \right]^- \right\} \tag{2-2}$$

which corresponds exactly to the objective function used in a proof by the same authors (eq. 1 in [15]). Operation variables $\mathbf{b}$ are subject to a set of constraints.

1) **Power Constraint**: The charge and discharge energy $b_{it}$ released or consumed per timestep $t$ are both limited in the form (with prediction horizon $K$)

$$\underline{b}_i \leq b_{it} \leq \bar{b}_i, \quad \forall i \in \mathcal{S}, \forall t \in [1, K] \tag{2-3}$$

2) **Energy Constraint**: the energy stored cannot exceed a prosumer's energy capacity $e_i$. Here, $SoC_i$ is the state of charge percentage. The term in the summation is the added energy in each timestep $t$, accounting for charging and discharging efficiencies $\eta$.

$$0 \leq e_i SoC_i^0 + \sum_{t=1}^{k} \left( [b_{it}]^+ \eta_i^{in} + [b_{it}]^- / \eta_i^{out} \right) \leq e_i, \quad \forall i \in \mathcal{S}, \forall k \in [1, K] \tag{2-4}$$

3) **Cycle Constraint**: At final timestep $t = K$, the initial state of charge must be returned (net change over all time steps is 0)

$$\sum_{t=1}^{K} \left( [b_{it}]^+ \eta_i^{in} + [b_{it}]^- / \eta_i^{out} \right) = 0, \quad \forall i \in \mathcal{S} \tag{2-5}$$

The minimization is changed slightly as done in [13] to remove the binary effect of the two $[.]^+, [.]^-$ terms in the objective function, and becomes

$$\begin{aligned}
\min_{\mathbf{b}^+, \mathbf{b}^-, \mathbf{L}^+, \mathbf{L}^-} \quad & \sum_{t=1}^{K} \left\{ p_t^b \sum_{i \in \mathcal{S}} L_{it}^+ + p_t^s \sum_{i \in \mathcal{S}} L_{it}^- \right\} \\
\text{s.t.} \quad & L_{it}^- \leq 0 \leq L_{it}^+ \\
& b_{it}^+ + b_{it}^- + q_{it} \leq L_{it}^+ \\
& b_{it}^+ + b_{it}^- + q_{it} = L_{it}^+ + L_{it}^- \\
& 0 \leq b_{it}^+ \leq \bar{b}_i \\
& \underline{b}_i \leq b_{it}^- \leq 0 \\
& 0 \leq e_i SoC_i^0 + \sum_{t=1}^{k} \left( b_{it}^+ \eta_i^{in} + b_{it}^- / \eta_i^{out} \right) \leq e_i, \forall k \in [1, K] \\
& \sum_{t=1}^{K} \left( b_{it}^+ \eta_i^{in} + b_{it}^- / \eta_i^{out} \right) = 0
\end{aligned} \tag{2-6}$$

This effectively sets $L_{it}^+ = \max\{0, b_{it}^+ + b_{it}^- + q_{it}\}$ and $L_{it}^- = \min\{0, b_{it}^+ + b_{it}^- + q_{it}\}$.

The implementation in this thesis changes this minimization problem to that in equation 5-1 to make explicit the fact that the coalitional energy cost $C(\mathcal{S})$ is computed over the *net* load.

### Cooperative Game Model

The characteristic function for a coalition $\mathcal{S}$ is defined as the cost savings of forming the coalition.

$$v(\mathcal{S}) = \sum_{i \in \mathcal{S}} C(\{i\}) - C(\mathcal{S}) \tag{2-7}$$

An imputation $\mathbf{x} \in \mathbb{R}^N$ is a vector of payoff allocations, payments to prosumers $i \in N$, originating from savings of forming the grand coalition $\mathcal{N}$. There are two criteria for the imputation:

1) **Efficiency criterion**: The total grand coalitions cost savings must be equal to the sum of all allocation $\mathbf{x}$ elements $x_i$.

$$\sum_{i \in \mathcal{N}} x_i = v(\mathcal{N}) \tag{2-8}$$

2) **Individual rationality**: The grand coalition $\mathcal{N}$ cost savings $x_i$ must be bigger than the non-coalitional individual cost savings $v(\{i\})$. In this case, $v(\{i\}) = 0$ as a prosumer cannot generate coalitional cost savings in the non-cooperative case.

$$x_i \geq v(\{i\}), \forall i \in \mathcal{N} \tag{2-9}$$

Besides criteria 2-8, 2-9, another criterion is needed to guarantee that under a payoff allocation vector $\mathbf{x}$, any smaller coalition $\mathcal{S}$ does not provide its prosumers with better cost savings than the grand coalition. To examine this, the excess $\varepsilon(\mathbf{x}, \mathcal{S})$ is used.

$$\varepsilon(\mathbf{x}, \mathcal{S}) = v(\mathcal{S}) - \sum_{i \in \mathcal{S}} x_i \tag{2-10}$$

For the grand coalition to be stable (disincentivizing prosumers from splintering into smaller coalitions), the excess $\varepsilon(\mathbf{x}, \mathcal{S})$ should always be negative.

The set of valid imputations $\mathcal{I}$ following from the efficiency and individual rationality criteria is defined as

$$\mathcal{I} := \left\{ \mathbf{x} \in \mathbb{R}^N \mid \sum_{i \in \mathcal{N}} x_i = v(\mathcal{N}), x_i \geq v(\{i\}), \forall i \in \mathcal{N} \right\} \tag{2-11}$$

The core $\mathcal{C}$ is a combination of $\mathcal{I}$ and the negative excess condition: an efficient, individually rational set of imputations $\mathbf{x}$ that guarantee the stability of the grand coalition with negative excess for all $\mathcal{S} \in 2^{\mathcal{N}}$.

$$\mathcal{C} := \left\{ \mathbf{x} \in \mathcal{I} \mid \varepsilon(\mathbf{x}, \mathcal{S}) \leq 0, \forall \mathcal{S} \in 2^{\mathcal{N}} \right\} \tag{2-12}$$

It is important to note that some games will have empty cores, meaning that the grand coalition is unstable.

### The Nucleolus

The nucleolus is a solution concept that gives an allocation that is in the core if the game is *balanced.* A game is balanced if for any balanced map $\alpha : 2^{\mathcal{N}} \to [0, 1]$, where

$$\sum_{\mathcal{S} \in 2^{\mathcal{N}}} \alpha(\mathcal{S}) 1_i^{\mathcal{S}} = 1, \forall i \in \mathcal{N}$$

$$1_i^S = \begin{cases} 1, & \text{if } i \in \mathcal{S} \\ 0, & \text{otherwise} \end{cases}$$

it holds that

$$\sum_{\mathcal{S} \in 2^{\mathcal{N}}} \alpha(\mathcal{S}) v(\mathcal{S}) \leq v(\mathcal{N}) \tag{2-13}$$

Balancedness can be interpreted as follows [22]. Assume each player $i$ has a unit of time to be distributed amongst all coalitions he is part of. Each coalition $\mathcal{S}$, active for that full unit of time, generates $v(\mathcal{S})$. Each coalition generates $\alpha(\mathcal{S})v(\mathcal{S})$, when all agents involved in $\mathcal{S}$ participate in it for a fraction of the given time unit $\alpha(\mathcal{S})$. Balancedness then indicates that no distribution of time across coalitions can generate a greater cumulative output than the grand coalition $v(\mathcal{N})$.

Han *et al.* empirically show that the nucleolus is in the core in [13], meaning the game is likely to be balanced. An extension of this paper ([15]) includes a proof for balancedness. This means that the nucleolus is a solution concept that generates stabilizing allocations given this Peer-to-Peer (P2P) framework.

The nucleolus $\nu$ is an imputation such that for any sub-coalition $j$

$$\epsilon_j(\nu) \leq \epsilon_j(\mathbf{x}), \forall \mathbf{x} \in \mathcal{I}, \forall j \in 2^N - 2 \tag{2-14}$$

the vector of excesses $\epsilon(\nu)$ is lexicographically minimized (the inequality sign refers to a lexicographic comparison of vectors), and is lexicographically smaller than *any* other excess vector $\epsilon_j(\mathbf{x})$. A vector $a = [0, 0, -5, -5]$ is lexicographically smaller than $b = [0, 0, 0, -10]$ because the entry $b(3) > a(3)$. The nucleolus is a unique payoff vector.

## 2-1-3 Nucleolus Computation

Han *et al.* ([13]) give the following algorithm for solving the nucleolus

**1)** Find the sub-coalitions with the largest excess:

$$LP_1 : \varepsilon_1 = \min_{\mathbf{x},\varepsilon} \varepsilon \tag{2-15}$$

$$\text{s.t.} \sum_{\forall i \in \mathcal{N}} x_i = v(\mathcal{N}) \tag{2-16}$$

$$v(\mathcal{S}) - \sum_{\forall i \in \mathcal{S}} x_i \leq \varepsilon, \forall \mathcal{S} \notin \{\emptyset, \mathcal{N}\} \tag{2-17}$$

Combining 2-17 with the minimization objective, $\varepsilon$ is reduced until it reaches a lower bound constrained by the largest excess corresponding to some set of subcoalitions $\mathcal{S} \notin \{\emptyset, \mathcal{N}\}$. This can be one or more sub-coalitions which will be entered into (now empty) set $\mathfrak{S}_l$.

**2)** Find the sub-coalitions with the next largest excess, considering that allocations in $\mathbf{x}$ given to prosumers belonging to sub-coalitions in $\mathfrak{S}_l$ must create the correct excesses $\varepsilon_l, \forall l \in [1, j-1]$ determined in previous iterations. Iterating $LP_j$ results in the unique nucleolus imputation $\mathbf{x}^*$.

$$LP_j : \varepsilon_j = \min_{\mathbf{x},\varepsilon} \varepsilon \tag{2-18}$$

$$\text{s.t.} \sum_{\forall i \in \mathcal{N}} x_i = v(\mathcal{N}) \tag{2-19}$$

$$\sum_{\forall i \in \mathcal{N}} x_i = v(\mathcal{S}) - \varepsilon_l, \quad \forall \mathcal{S} \in \mathfrak{S}_l, \forall l \in [1, j-1] \tag{2-20}$$

$$v(\mathcal{S}) - \sum_{\forall i \in \mathcal{S}} x_i \leq \varepsilon, \quad \forall \mathcal{S} \notin \{\emptyset, \mathfrak{S}_l, \mathcal{N}\}, \forall l \in [1, j-1] \tag{2-21}$$

The additional constraint $2-20$ specifies the imputation $\mathbf{x}$ must be such to respect the already allocated cost savings $v(\mathcal{S}) - \varepsilon_l$ determined prior to all agents in all coalitions $\mathcal{S}$ in $\mathfrak{S}_l$.

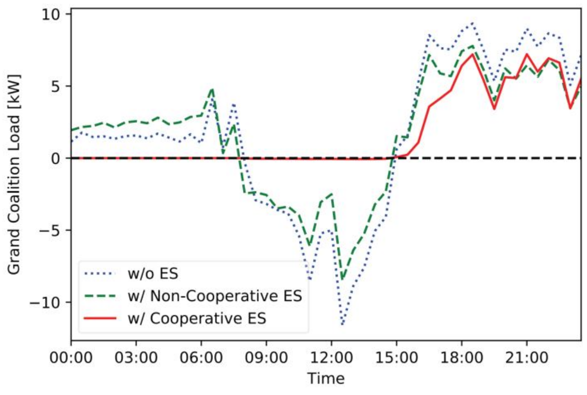


**Figure 2-1:** Loads of 1) non-cooperative individual agents without ES, 2) non-cooperative individual agents with ES, 3) agents with cooperative ES [13].

### 2-1-4 Case Studies

#### Load Balancing

Figure 2-1 shows that the grand coalition load with non-cooperative ES reduces reverse power flow, and lessens the peak load. Cooperative ES operation flattens the coalition net demand/-generation curve even more.

#### Nucleolus & Balancedness

Han *et al.* [13] state that the largest excess of the nucleolus imputation is always negative across many runs of prosumer constellations, so the game is likely to be balanced. This is an empirical observation, however balancedness is proven in later paper [15].

#### Nucleolus and Marginal Contribution

The nucleolus and Shapley values for each prosumer can be plotted against one another (figure 2-2). The first thing to note is that the imputations follow the diagonal. The Shapley as a cooperative game theory solution concept calculates prosumer payoffs according to their marginal contribution to the grand coalition. The diagonal relation between the Shapley and nucleolus payoffs shows that a relation between nucleolus payoffs and marginal contribution also holds for the nucleolus to a large extent.

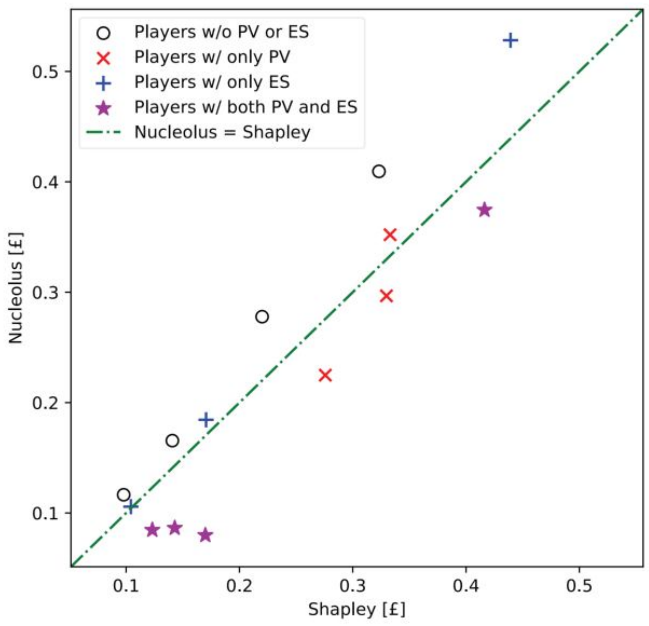


**Figure 2-2:** Cost savings according to Shapley and nucleolus solutions for all prosumers, plotted against eachother.

# Chapter 3

# Nucleolus Estimation in P2P Markets

## 3-1 Introduction

In [13], it was seen that the calculation of the nucleolus was computationally intensive. In [16], Han *et al.* explore clustering prosumers into $K$ clusters to reduce the number of required analyzed coalitions from $\mathcal{O} \sim (2^N)$ (for $2^N - 2$ possible coalitions $\mathcal{T} \subset \mathcal{N}$) to a $\mathcal{O} \sim (2^K N)$ problem, with $2^{K-1}(N+2)-1$ coalitions that need to be evaluated to estimate the nucleolus. This chapter details the techniques and results of that paper. The nucleolus estimation is shown to be accurate in comparison to full nucleolus computation.

As prosumer load and generation profiles characterize the prosumer's role in the grand coalition, it makes sense to cluster prosumers based on these profiles. Alternatively, the marginal contribution of each prosumer to the grand coalition can be a clustering feature, as the nucleolus allocation and marginal contribution to the grand coalition are linked. K-means, hierarchical and Gaussian mixture modelling are chosen as clustering techniques.

## 3-2 Clustering Techniques

Prosumers are clustered into $K$ clusters on the basis of prosumer feature profiles. The first feature profile is the *grand coalition cooperative energy profile*, the net energy demand of a prosumer across all time intervals as it cooperatively participates in the grand coalition. The second is the *grand coalition marginal contribution profile.* These two will be described in section 3-4.

The set of $K$ clusters are defined as $cl_{\mathcal{K}} = \{cl_1, cl_2, \cdots, cl_K\}$. The clustering assignment is $\mathbf{g} : g_i = j | i \in cl_j, \forall i \in \mathcal{N}$. The clustering methods are chosen on the basis of paper [29], where clustering techniques are evaluated using various cluster validity indices.

### 3-2-1 K-means Clustering

K-means is a clustering method that iterates between constructing clusters based on the Euclidian distance to cluster centroids and updating cluster centroids. The two steps are:

1) Generate clusters from centroid profiles, by assigning players to clusters with the smallest Euclidean distance $d(\mathbf{f}_j, \mathbf{c}_j)$. Centroid profiles $cl_j$ are initialized as randomly generated profiles of the chosen feature profile. In this thesis, feature profiles will all have dimensions $f_{it} \in \mathbb{R}^{48}$.

$$d\left(\mathbf{f}_i, \mathbf{c}_j\right) = \sqrt{\sum_{t}^{R} \left(f_{it} - c_{jt}\right)^2} \tag{3-1}$$

$$g_i \leftarrow \arg\min_{j} d\left(\mathbf{f}_i, \mathbf{c}_j\right) \tag{3-2}$$

2) The generated clusters are used to compute new centroids $\mathbf{c}_j$ for each cluster. Centroids are a minimization of the squared Euclidean distance for all prosumers $i \in cl_j$, or equivalently by calculating the average feature profile of all prosumers.

$$\mathbf{c}_j = \arg\min_{\mathbf{c}} \sum_{i \in cl_j} d^2\left(\mathbf{f}_i, \mathbf{c}\right) = \frac{\sum_{i \in cl_j} \mathbf{f}_i}{|cl_j|} \tag{3-3}$$

The k-means algorithm is prone to settling in local minima. It is the norm to repeat the algorithm with different random initializations of $\mathbf{c}_j$ to find $g_i, \mathbf{c}_j$ that belong to the iteration with the lowest cost function $\sum_{j \in K} \sum_{i \in cl_j} d^2\left(\mathbf{f}_i, \mathbf{c}\right)$. The authors limit k-means iterations to 1000, each with different initial centroids.

### 3-2-2 Hierarchical Clustering

Hierarchical clustering was another method that performed well in paper [29]. Clusters are formed from two most similar clusters (clusters with the smallest Euclidean distance as described for k-means clustering). Initial clusters are the individual prosumer feature profiles. The final cluster assignment of feature profiles is achieved when the predetermined number of clusters $K$ is reached, however a predetermined number of clusters is not required for the evolution of cluster formation such as with k-means, as the number of clusters reduces in each step.

Hierarchical clustering operates on the basis of a linkage criterium, of which there are four; *single*, *complete*, *average*, and *Ward*. This defines how the Euclidean distance between clusters is measured. Single linkage regards the distance (3-1) between the two closest profiles of two clusters, while complete linkage regards the distance between the two farthest profiles. Average linkage looks at the average distance of all possible pairs between the two clusters, with each pair consisting of one feature profile from both clusters. The Ward linkage method computes, for all possible pairs of clusters, the sum of squared distances between each feature profile and the centroid of the cluster pair, and forms a new cluster from the pair which minimizes this objective function.

The authors only consider the Ward linkage for nucleolus estimation case studies, as it has the least sensitivity to outliers. Sensitivity to noise and outliers is generally unwanted as a clustering method that discriminates outliers has a tendency to mischaracterize cluster formations when data is noisy [32].

A clustering method's sensitivity to outliers is also reflected in the unevenness of cluster sizes, as this indicates that the clustering method may subject the data to chaining. Chaining is an unwanted effect of noise that extends clusters into regions that should be represented by other clusters. This reason for choosing the Ward method is also specified in [29].

### 3-2-3 Gaussian Mixture Model (GMM) Clustering

Unlike the previous distance based methods, GMM is a probabilistic method that defines a probability distribution function as the sum of $K$ Gaussian distributions, defined by $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K\}$, where $\boldsymbol{\theta}_j = \left(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)$ In this case, $\mu_j$ is a $R$-dimensional vector of a Gaussian mean, and $\boldsymbol{\Sigma}_j$ is a $R \times R$ covariance matrix. The probability that an instance occurs can be defined as a weighted combination of these $K$ Gaussians, where $\alpha_j \geq 0$, $\sum_{j=1}^{K} \alpha_j = 1$.

$$p\left(\mathbf{f}_i \mid \boldsymbol{\theta}\right) = \sum_{j=1}^{K} \alpha_j p\left(\mathbf{f}_i \mid \boldsymbol{\theta}_j\right) \tag{3-4}$$

Values $\{\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\alpha}\}$ need to be found with methods such as expectation-maximization so that the weighted sum of Gaussians best represents the true probability density function of all feature profiles. The feature profile instance $\mathbf{f}_i$ has the probability it belongs to cluster $k$ (defined by Gaussian $\theta_{\mathbf{k}}$ and scaling factor $\alpha_k$) defined as

$$p\left(k \mid \mathbf{f}_i\right) = \frac{\alpha_k p\left(\mathbf{f}_i \mid \boldsymbol{\theta}_k\right)}{\sum_{j=1}^{K} \alpha_j p\left(\mathbf{f}_i \mid \boldsymbol{\theta}_j\right)}, \quad k = 1, \ldots, K \tag{3-5}$$

Han *et al.* [16] run GMM with different covariance structures (full, diagonal, spherical and tied covariance matrices) using 100 prosumer loads, the most even cluster sizes are generated by a full covariance matrix. As this is an indicator of good clustering characteristics in the prevalence of noise, only full covariance matrices are considered in the case studies of this paper.

## 3-3 Nucleolus Estimation

As the nucleolus calculation considers all $2^N - 1$ possible coalitions $\mathcal{T}$, clustering can help reduce the number of coalitions $\mathcal{T}$ by clustering prosumers strategically. In an older paper [14], nucleolus estimations were based on considering $K$ clusters (nucleolus computation complexity $\mathcal{O}(2^K)$ rather than $N$ prosumers (computation complexity $\mathcal{O}(2^N)$ for the formation of coalitions $\mathcal{T}$. The results show that there is no way to differentiate between prosumers in individual clusters, and that prosumer payoffs within these clusters are arbitrary. Han *et al.* considerably improve nucleolus estimation in this paper [16].

Han *et al.* modify the method of [14], adding coalitions $\mathcal{T}$ that consider the individuality of individual prosumers. The method is described in steps as follows:

**1. Clustering**
Prosumers are clustered based on one of two possible features, the *grand coalition cooperative energy profile* and *grand coalition marginal contribution profile*. These will be explained in the next section 3-4. The set of $K$ clusters is $cl_{\mathcal{K}} := \{cl_j \mid j \in \mathcal{K}\}$ with $\mathcal{K} = \{1, 2, \ldots, K\}$.

**2. Cluster Coalitions\Cluster Combinations**
All combinations of clusters are formed and converted into coalitions $\mathcal{T}$. Specifically, just as $cl_{\mathcal{K}}$ was the set of *all* clusters, each $cl_{\mathcal{U}}$ points to a subset of clusters, so that it holds that $\forall cl_{\mathcal{U}} \subseteq cl_{\mathcal{K}}$. There are $2^K - 1$ possible combinations of clusters $cl_j$. Ultimately, all coalitions formed in this step are put in set $\mathfrak{U} = \{\mathcal{T} \mid \mathcal{T} = Q\left(cl_{\mathcal{U}}\right), \forall cl_{\mathcal{U}} \subseteq cl_{\mathcal{K}}\}$, where $Q(cl_{\mathcal{U}})$ converts a set of clusters into a coalition of prosumers.

**3. Diversity-based Pairing Coalitions**
A set $\mathfrak{S}_i$ is created for each prosumer $i$ that consists of all coalitions in $\mathfrak{U}$ that exclude prosumer $i$ (defined as $\mathbf{U}_i$), but then with prosumer $i$ added to those coalitions. Formally, $\mathfrak{S}_i = \{\mathcal{T} \mid \mathcal{T} = Q\left(cl_{\mathcal{U}}\right) \cup \{i\}, \forall cl_{\mathcal{U}} \in \mathbf{U}_i\}$. Adding prosumer $i$ to cluster coalitions consisting of clusters with mutually different characteristics gives a high probability that the greatest payoff to prosumer $i$ is in one of these coalitions. The number of coalitions $\mathcal{T}$ added in this step is bounded by $2^{K-1}N$.

**4. Collecting Coalition Samples for Nucleolus Estimation**
Sampled coalitions are grouped into set $\mathfrak{T}^{cl_{\mathcal{K}}} = \mathfrak{S} \cup \mathfrak{U}$ which are used for nucleolus estimation. This set size is bounded by $\left|\mathfrak{T}^{cl_{\mathcal{K}}}\right| \leq |\mathfrak{U}| + |\mathfrak{S}| = 2^{K-1}(N+2) - 1$.

**Intuition of Steps 1-4**
Step 2 generates a set of coalitions $\mathcal{T}$. These coalitions will have large coalitional cost savings $v(\mathcal{T})$ due to differences in cluster characteristics which improves the marginal contribution of clusters to the coalition. However, the individuality of each prosumer is not considered as the nucleolus estimation contains only the set of coalitions $\mathfrak{U}$. In such a case, prosumers can at most be considered arbitrary participants of their respective cluster. Figure 3-4 shows that considering only prosumer clusters is not enough to estimate an accurate nucleolus payoff.

Step 3 is the solution to this problem. In this step, the individuality of each prosumer is characterized by a set of coalitions $\mathfrak{S}$ which likely includes a coalition which offers each prosumer $i$ the highest payoff (judging from the positive correlation between a prosumer's marginal contribution and nucleolus payoff). By including these coalitions in the nucleolus estimation, the nucleolus estimation must minimize the excesses of each prosumer's set of sub-coalition stemming from diversity-based pairing, removing any reason for a prosumer to be dissatisfied with its payoff.

## 3-4 Clustering Features

The clustering of prosumers is based on two feature profiles. The first is the prosumers load profile as part of the grand coalition. The second is the grand coalition marginal contribution profile.

### 3-4-1 Grand Coaliton Cooperative Energy Profiles

A prosumer's load profile as it operates within the grand coalition is a logical feature profile, as it captures prosumer behavior in the context of participating in the grand coalition. The computation of these grand coalition cooperative energy profiles $\mathbf{p}_i$ only requires the optimal cooperative ES operations $\mathbf{b}^{\mathcal{N}}$ (equation 3-6, where $F_t^{\mathcal{N}}$ refers to the grand coalition cost, see equation 2-1), and the net energy consumption without ES $\mathbf{q}_i$ (see 3-7).

$$\mathbf{b}^{\mathcal{N}} = \arg\min_{\mathbf{b}} \sum_{t=1}^{R} F_t^{\mathcal{N}}(\mathbf{b}) \tag{3-6}$$

$$\mathbf{p}_i = \mathbf{q}_i + \mathbf{b}_i^{\mathcal{N}}, \forall i \in \mathcal{N} \tag{3-7}$$

An example of clustering grand coalition load profiles is shown below in figure 3-1a.

### 3-4-2 Grand Coalition Marginal Contribution Profiles

Another feature profile that Han *et al.* consider is the grand coalition marginal contribution profile, since a prosumer's nucleolus payoff is strongly linked to its marginal contribution to coalitions. Profiles $\{\Delta\mathbf{F}_1, \Delta\mathbf{F}_2, \ldots, \Delta\mathbf{F}_N\}$ can be found as follows

$$\Delta F_{it} = [F_t^{\mathcal{N}\backslash\{i\}}\left(\mathbf{b}^{\mathcal{N}\backslash\{i\}}\right) + F_t^{\{i\}}\left(\mathbf{b}^{\{i\}}\right)] - F_t^{\mathcal{N}}\left(\mathbf{b}^{\mathcal{N}}\right) \tag{3-8}$$

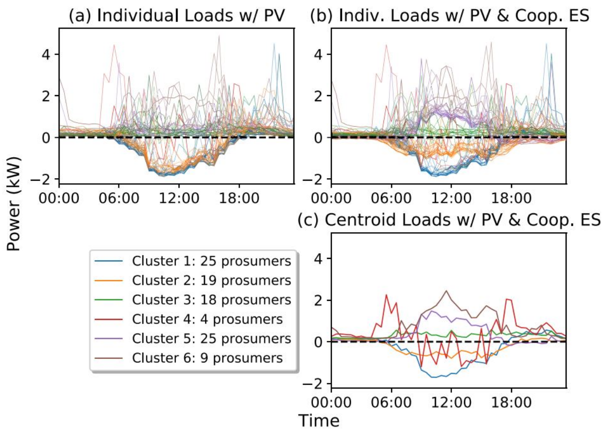
How these feature profiles look like in practice is shown in figure 3-1b.

## 3-5 Nucleolus Estimation Evaluation using Stratified Random Sampling

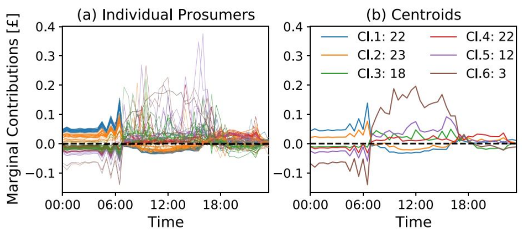
Since the case studies in later section 3-6 examine games of up to 400 prosumers, calculating the nucleolus value to compare the nucleolus estimation technique of this paper to is not feasible. An independent and consistent base-line method of estimating the nucleolus is necessary.

Han *et al.* use stratified random sampling. Given a prosumer set $N$, all possible permutations of this set are $\pi(\mathcal{N})$, each permutation defined by $O \in \pi(\mathcal{N})$. Strata are defined as $P_{il} := \{O \in \pi(\mathcal{N}) \mid O(l) = i, \forall i, l \in [1, N]\}$, meaning each stratum $P_{il}$ contains all permutations $O \in \pi(\mathcal{N})$ where prosumer $i$ is in position $l$.

For consistent and balanced coalition sampling, it is required that we sample across all prosumers and coalition sizes evenly. The steps are as follow:



**(a)** Clustering prosumers based on grand coalition load profiles. Also shows the load profiles of all prosumers in the non-cooperative case.



**(b)** Clustering prosumers based on marginal contribution profiles.

**Figure 3-1:** Clustered feature profiles

1) Iteration starts at an initial sample size of $m = 1$.
2) $m$ random samples from the full set of permutations in a stratum $P_{il}$ are entered into $M_{il}$.
3) $\mathfrak{R}_{il}$ turns $Pre^i(O) \cup i$ for each permutation $O$ in $M_{il}$ into a coalition. Formally $\mathfrak{R}_m := \{\mathcal{T} \mid \mathcal{T} = Pre^i(O) \cup i, \forall O \in M_{il}, \forall i, l \in [1, N]\}$.
4) $v(\mathcal{T})$ is calculated for all coalitions $\mathcal{T}$ in $\mathfrak{R}$ using equation 2-7. The number of positive excesses given the nucleolus estimation is $\mid \varepsilon_m^{pos} \mid$.
5) The number of positive excesses must converge as sample size $m$ increases $(\frac{|\varepsilon_m^{\text{pos}}|}{|\mathfrak{R}_m|} - \frac{|\varepsilon_{m-1}^{\text{pos}}|}{|\mathfrak{R}_{m-1}|} \leq \delta)$. In the case study, the value is $\delta = 0.5\%$. At this point of convergence, it is considered that a valid percentage of positive excesses $\frac{|\varepsilon_m^{\text{pos}}|}{|\mathfrak{R}_m|}$ is found. This is a measure of the nucleolus estimation performance.

The point of this is to collect an independent sampled set of sub-coalitions $\mathcal{T}$ for which the nucleolus estimation can be tested against. If this test gives positive excesses $\varepsilon_m(\boldsymbol{v}^{est}, \mathcal{T})$, this means that the nucleolus estimate has not managed to keep all excesses of sub-coalitions $\mathcal{T}$ negative. A nucleolus estimation that has the lowest percentage of $\frac{|\varepsilon_m^{\text{pos}}|}{|\mathfrak{R}_m|}$ performs the best. The authors use the term *divisible excess* for a positive excess $\varepsilon_m^{\text{pos}}(\boldsymbol{v}^{est}, \mathcal{T})$ for which $\frac{\varepsilon_m^{\text{pos}}(\boldsymbol{v}^{est}, \mathcal{T})}{|\mathcal{T}|} \geq €0.01$.

The nucleolus estimation quality indicator *coalitions with divisible excesses* is critical in the case studies, since it provides an independent measure for the quality of the nucleolus estimation.

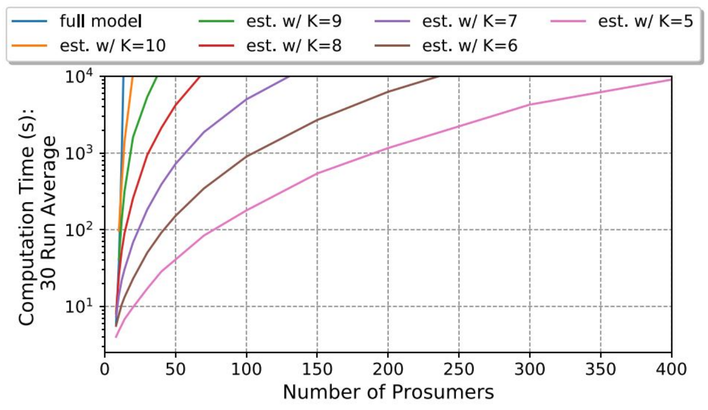


**Figure 3-2:** Computation times of nucleolus estimations according to prosumer size and cluster number $K$.

CLUSTERING COMPUTATION TIME (S) AVG. OVER 10 RUNS

| **No. prosumers** ($K$) | **14** (10) | **20** (9) | **30** (8) | **50** (7) | **100** (6) |
|---|---|---|---|---|---|
| K-means | 5.3 | 4.0 | 3.5 | 4.1 | 5.1 |
| Hierarchical-*ward* | 0.6 | 0.5 | 0.5 | 0.6 | 0.5 |
| GMM-*full* | 2.2 | 1.8 | 1.7 | 2.1 | 1.7 |

**Figure 3-3:** Computation times of clustering according to prosumer size, cluster number $K$, and clustering technique.

## 3-6 Case Studies

All case studies are completed using Photovoltaics (PV) and Energy Storage (ES) adoption rates of 50%, all randomly assigned. Domestic load and PV data covers a 24 hour timespan. This is predictive data, for which the grand coalition Peer-to-Peer (P2P) ES operation and the corresponding nucleolus estimation must be calculated.

### 3-6-1 Computation Time: Nucleolus

Applying k-means to grand coalition cooperative energy profiles, the nucleolus computation times are shown in figure 3-2. The nucleolus of games up to 400 prosumers can be computed within $10^4 s$ ($\approx 2.75h$), with varying cluster numbers $K$.

### 3-6-2 Computation Time: Clustering

Lets say it is required for the nucleolus to be calculated within $10^3 s$ ($\approx$ 15min). From figure 3-2, the maximum number of clusters per 14, 20, 30, 50 and 100 player games is $K = 10, 9, 8, 7, 6$ respectively. The time it takes to form these clusters is shown in figure 3-3. Considering the $10^3 s$ time limit, the clustering step only takes about $5.3s$ at worst (with k-means, for $N = 14, K = 10$). The clustering computation time is not a critical factor in relation to the nucleolus computation time.

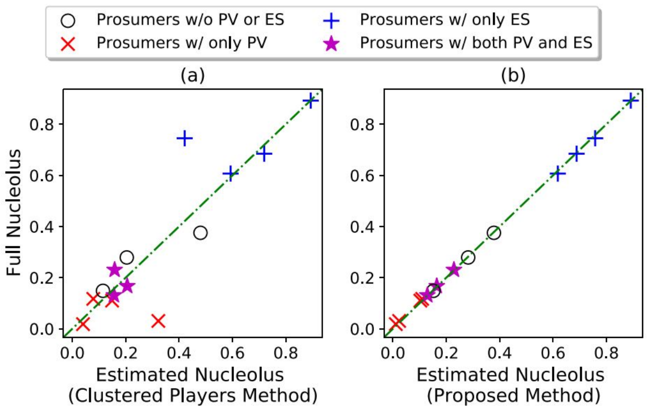


**Figure 3-4:** Full nucleolus prosumer payoff allocations plotted against the proposed method (right) and a nucleolus estimation payoff only considering clustered sets $\mathfrak{U}$ (left).

### 3-6-3 Nucleolus Estimation Accuracy

The right figure in figure 3-4 shows the payoff allocations of the full nucleolus calculation plotted against the nucleolus estimation. The left plot shows results from a previous paper ([14]), where the nucleolus estimation excludes coalition sets $\mathfrak{S}_i$ (diversity-based pairing coalitions) from the nucleolus estimation, showing that adding coalitions from diversity-based pairing into the nucleolus estimation computation improves the nucleolus estimation dramatically.

Figure 3-5 shows the percentage of coalitions with divisible excesses. This refers to the aforementioned percentage of sampled coalitions for which $\frac{\varepsilon_m^{\text{pos}}(\boldsymbol{v}^{est},\mathcal{T})}{|\mathcal{T}|} \geq €0.01$ holds . Increasing percentages show that a higher proportion of test coalitions (originating from stratified random sampling, section 3-5) are dissatisfied with the payoff allocation of the nucleolus estimation.

Figure 3-5 shows the 10-run average percentage of coalitions with divisible excess, taking $(N, K)$ pairs corresponding to a $10^3 s$ nucleolus estimation computation time limit (using figure 3-2). GMM with a full covariance matrix considering grand coalition marginal contribution feature profiles provides the best results.

### 3-6-4 Scaling Up - $10^3$s to $10^4$s Computation Time Limit

The top plot of figure 3-6 shows the percentage of coalitions with divisible excess for $(N, K)$ pairs taken according to a $10^4 s$ nucleolus estimation computation time bound. Nucleolus estimation performance declines as the ratio $N/K$ is increased. The bottom chart shows the largest player dissatisfaction for each game, quantified in $£$.

It is important to note that while the range of percentage of coalitions with divisible excesses is wide, in most cases a good proportion of them result in 0% of coalitions having divisible excesses. Consequently, the observation is that prosumer composition can have a large influence on the quality of the nucleolus estimation.

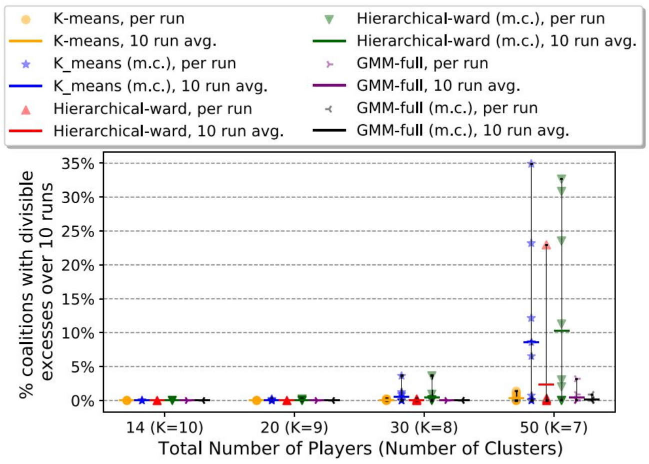


**Figure 3-5:** Percentage of coalitions with divisible excesses - averaged over 10 runs.

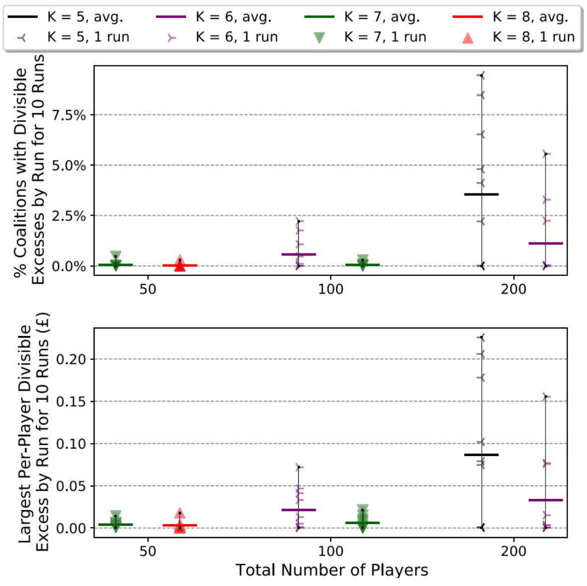


**Figure 3-6:** Divisible excesses per game size $N$ and cluster size $K$ (top). The most dissatisfied player allocations per game size $N$ and cluster size $K$ (bottom).

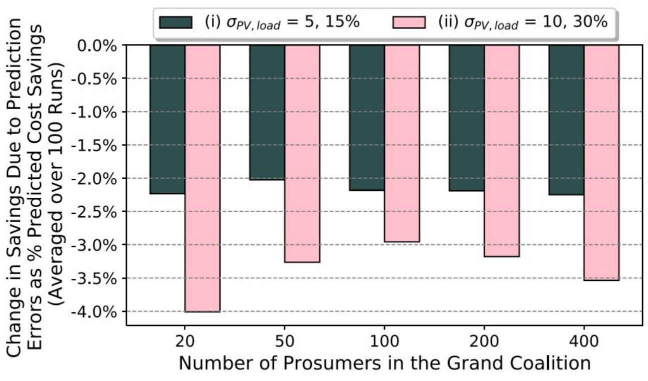


**Figure 3-7:** Averaged grand coalition profit reductions for two sets of standard deviations.

### 3-6-5 Monte Carlo Analysis of Uncertainty Impact on Grand Coalition Total Profits

A Monte Carlo analysis applies the prediction-optimal ES operations to a simulation in which actual PV generation and load profiles are drawn from normal distributions, with predicted values as means. The actual PV generation profile is applied to all PV installations, assuming they are all located in the same area. This experiment is repeated 100 times for different prosumer compositions. The average reduction of grand coalition profits are shown in figure 3-7. The averaged profit reduction resulting from uncertainties in predictions is limited to 4%, with $\sigma_{PV,load} = 15, 30\%$. It follows that the sum of nucleolus payoffs must be reduced by at most 4% as well.

Chapter 4

# Nucleolus Estimation Case Studies

## 4-1 Introduction

An interesting field of work would be to investigate and develop the notion of diversity-based pairing (explained in point 3 under 3-3) to improve nucleolus estimations. This can be accompanied by a general investigation of other clustering methods often used for the clustering of load profiles. Han *et al.* state [16]:

*The purpose of choosing a selection of popular clustering techniques is to demonstrate the effectiveness of the proposed method; a comprehensive investigation of the best possible clustering technique to use is left for future work.*

Section 4-2 will detail case studies that develop the notion of diversity-based pairing, by applying modifications to clustering techniques used by Han *et al.* [16]. Specifically, the following case studies are outlined:

- Case Study 1: Distance-adjusted Clustering
  Modifications of clustering methods considering distances between cluster centers and prosumers (section 4-2-1).
- Case Study 2: Considering the Primary Prosumer as a Cluster Centroid
  Considering prosumers as cluster centroids in diversity-based pairing (section 4-2-2).

Section 4-3 details investigations into a selection of alternative clustering methods considered in problems involving the clustering of consumer load profiles in literature. Specifically:

- Case Study 1: Performance Comparisons between K-medians and K-means
  Investigating the effect on nucleolus estimation performance with k-medians' reduced sensitivity to outliers compared to k-means (section 4-3-1).

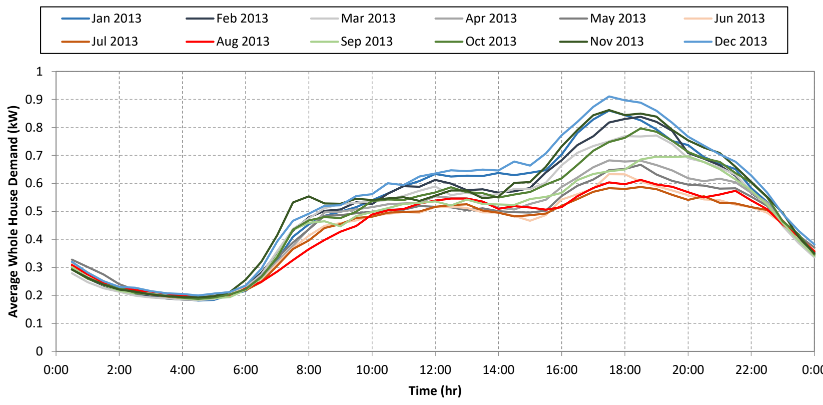


**Figure 4-1:** Monthly averaged demand profiles as shown in Customer-Led Network Revolution trials [30].

- Case Study 2: Fuzzy C-Means & Fuzzy C-Medians
  Applying fuzzy c-means clustering and evaluating nucleolus estimation performance. Two additional optional case studies (1. automatic fuzzifier parameter selection, 2. fuzzy c-medians) could be completed (section 4-3-2).

- Case Study 3: Comparing Hierarchical-Ward, Hierarchical-average, and Hierarchical-complete Clustering
  The degree to which even cluster sizes are generated changes incrementally as the Ward linkage method is replaced by the complete, and then the average linkage. This case study evaluates nucleolus estimation performance of each method to see how nucleolus estimation is affected by clustering with various degrees of sensitivity to outliers (section 4-3-3).

- Case Study 4: GMM-PCA
  This case study investigates if the quality of nucleolus estimation improves with reduced axes (the *Curse of Dimensionality*) (section 4-3-4).

- Case Study 5: Shape-based Clustering
  Shape-based clustering through the normalization of clustering feature profiles (section 4-3-5).

### 4-1-1 Generating Prosumer Load Profiles

Han *et al.* [16] use demand profiles sourced from the Customer-Led Network Revolution [31]. This 11GB database contains individual domestic net load profiles and domestic PV generation profiles. Both are sampled at one minute intervals. Demand profiles can be generated by subtracting the PV generation from the net load profiles.

PV generation profiles are sourced from PVWatts, which gives one-hour profiles for all days in a year. As Han *et al.* use models assuming a uniform PV profile for all prosumers, it is most likely that they use a random day (out of 31 July days) for individual nucleolus estimations. Additionally, the authors consider adoption rates of 50% for each PV and ES independently.

### 4-1-2 Marginal Contribution Profile Implementation

The marginal contribution profile used in these case studies is slightly different from the one used by Han *et al.* [16] (eq. 3-8). Instead of considering costs of coalitions at isolated time intervals, each time stamp $t$ will consider the cumulative cost across intervals $r = (0, \ldots, t)$.

$$\Delta F_{it} = \left[ \sum_{r=1}^{t} F_r^{\mathcal{N}\backslash\{i\}} \left( \mathbf{b}^{\mathcal{N}\backslash\{i\}} \right) + \sum_{r=1}^{t} F_r^{\{i\}} \left( \mathbf{b}^{\{i\}} \right) \right] - \sum_{r=1}^{t} F_r^{\mathcal{N}} \left( \mathbf{b}^{\mathcal{N}} \right) \tag{4-1}$$

where it was established in equation 5-1 that coalition $\mathcal{T}$ has coalitional cost $F_t^{\mathcal{T}} \left( \mathbf{b}^{\mathcal{T}} \right)$ at time interval $t$ with optimal ES operation $\mathbf{b}^{\mathcal{T}}$

$$F_t^{\mathcal{T}} \left( \mathbf{b}^{\mathcal{T}} \right) = r_t^{im} \max \left( 0, \sum_{i \in \mathcal{T}} b_{it}^{+} + b_{it}^{-} + q_{it} \right) + r_t^{ex} \min \left( 0, \sum_{i \in \mathcal{T}} b_{it}^{+} + b_{it}^{-} + q_{it} \right) \tag{4-2}$$

$$\mathbf{b}^{\mathcal{T}} = \arg\min_{\mathbf{b}} \sum_{t=1}^{R} F_t^{\mathcal{T}}(\mathbf{b}) \tag{4-3}$$

Anyhow, this feature profile is just as valid as the original marginal contribution profile as defined by Han *et al.*, and the difference is caused by a miscalculation early on in the computation of case study results.

## 4-2 Case Studies: Set 1 - Modified Clustering Methods

In diversity-based pairing (Han *et al.* [16]), a prosumer forms coalitions with all possible combinations of clusters which it is not part of. This is repeated for all individual prosumers, and the set of coalitions generated is used in the lexicographical minimization algorithm. In this context, the particular prosumer which is paired with all other clusters will be called the *primary prosumer*.

Case studies in this section focus on modified clustering methods. The modifications are minor adjustments in the cluster formation in diversity-based pairing. As only marginal improvements are expected at most, it is enough to apply modifications to the three best performing methods of Han *et al.* [16]:

1. GMM, using the grand coalition marginal contribution profiles.
2. GMM, using the grand coalition cooperative energy profile.
3. k-means, using the grand coalition cooperative energy profile.

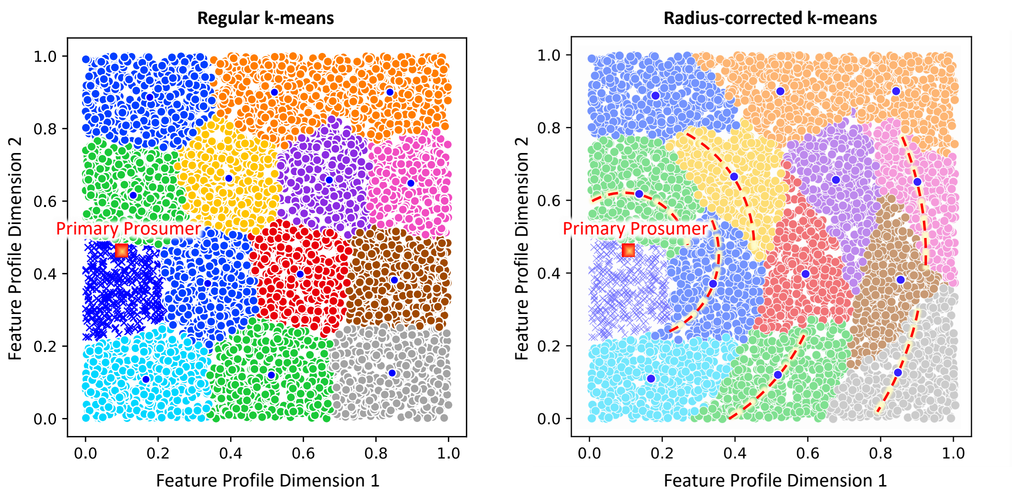


**Figure 4-2: Left**: the result of regular k-means clustering. **Right**: modified k-means clustering that prefers clusters having equal distance between all its prosumers and the primary prosumer.

### 4-2-1 Case Study 1: Distance-adjusted Clustering

This case study investigates if clusters should be modified to account for the position of the primary prosumer in diversity-based pairing, rather than solely being a measure of similarity between prosumers regardless of the position of the primary prosumer. Figure 4-2 shows an example of correcting k-means clusters. The modified distance function between any prosumer $i$ and any cluster $j$ (omitting the primary prosumer's cluster and its prosumers) is

$$\begin{aligned}\mathrm{dist}_{mod}\left(\mathrm{pros}_i, \mathrm{cluster}_j\right) =&\mathrm{dist}\left(\mathrm{pros}_i, \mathrm{cluster}_j\right) + \\ &m\big|\mathrm{dist}\left(\mathrm{prim.pros}, \mathrm{cluster}_j\right) - \mathrm{dist}\left(\mathrm{pros}_i, \mathrm{prim.pros}\right)\big|\end{aligned} \tag{4-4}$$

The idea is that if the purpose of diversity-based pairing is to combine any primary prosumer with clusters that are diverse in comparison to it, it might be beneficial to modify clusters to also account for the distance between primary prosumers and clusters. Variable $m$ changes the degree of the modification (later section 6-1-2 explains how this variable is chosen).

### 4-2-2 Case Study 2: Primary Prosumers as Cluster Centroids

In diversity-based pairing, the primary prosumer is combined with combinations of clusters that it does not partake in. However, if the primary prosumer is close to the boundary of its cluster, it seems unfair to exclude prosumers in its own cluster (that might be farther away), while including prosumers from neighboring clusters that are closer to the primary prosumer. A 2D example of this is shown in figure 4-3, where the primary prosumer is combined with cluster combinations including the cluster containing prosumer B, while prosumer A, despite being farther, is excluded from any cluster combinations.

A seperate case study can investigate the effect of considering the primary prosumer as a cluster centroid. The primary prosumer can then be paired with combinations of all possible clusters excluding itself.

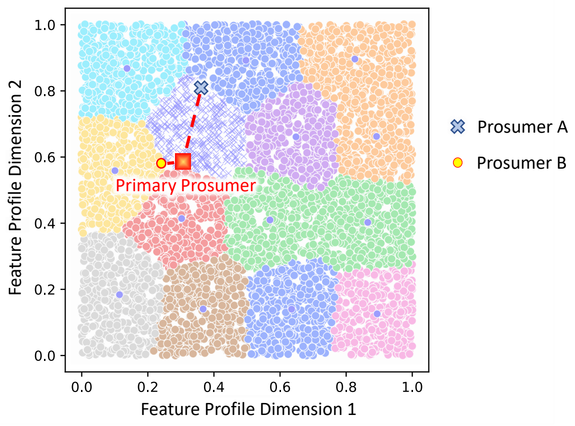


**Figure 4-3:** As it stands, in diversity-based pairing, the primary prosumer would be included in coalitions that exclude prosumers in its own cluster (despite them being farther away, such as Prosumer A), but include prosumers nearer to the primary prosumer if they belong to different clusters (such as Prosumer B).

The disadvantage to such a method would be the need to recluster all prosumers $N$ times for each nucleolus estimation, when each prosumer assumes the role of primary prosumer. However, benefit may be gotten from the increased accuracy of the nucleolus estimation that, in combination with the additionally required computational time, is competitive with other methods.

### Modified k-means

Modifying k-means can be done by fixing the mean of a cluster to be the primary prosumer will have its cluster be centered around it.

### Modified GMM

For the cluster to which the primary prosumer belongs, the notion of clustering by distance to the primary prosumer in k-means is replaced by clustering by likelihood of belonging to a cluster centered around the primary prosumer for GMM. Figure 4-5 show the non-modified GMM (figure 4-5a) and modified GMM (figure 4-5b) clustering results, where the modified case takes prosumer 8 (P8 in figure) to be the primary prosumer, and fixes the mean of the Gaussian to that position throughout the EM algorithm.

The resulting Gaussian does not accurately represent its prosumers anymore; the mean is not truly the mean of the cluster assignments, and this in turn warps the covariance calculations. A fringe case is shown in figure 4-4, where the primary prosumer (prosumer 24) is chosen to be at the upper limit of dimension 2 and as such cannot be the average of a multi-prosumer cluster, however the modified GMM still produces an acceptable result worth studying.

From numerous manual applications of code that perform this modification in 2D problems as shown in figure 4-5b, resulting clusters are usually either 1) modified in line with what would be expected, 2) unchanged from the non-modified clustering assignments or 3) not preferred

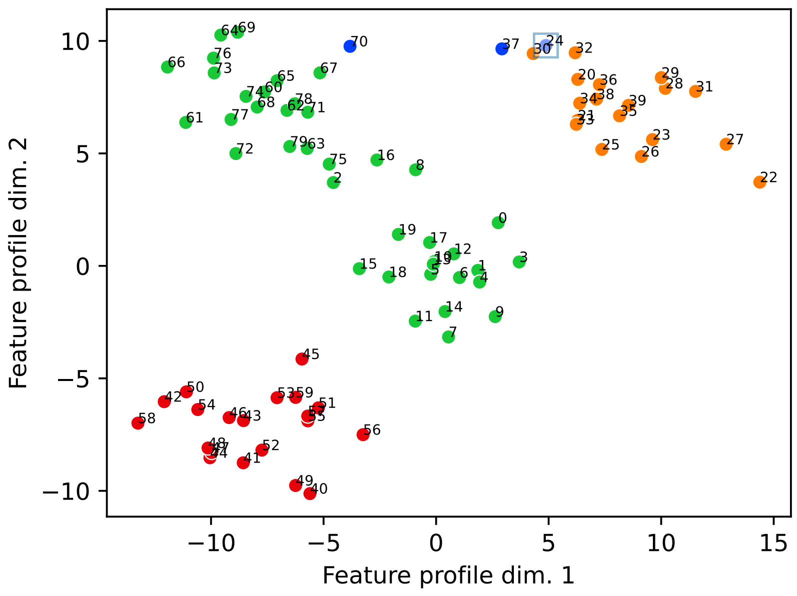
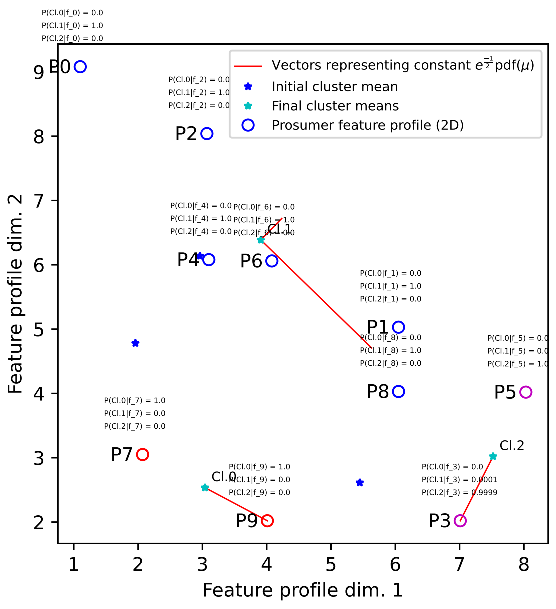


**Figure 4-4:** The blue cluster is the cluster associated with primary prosumer (prosumer 24), and the modified GMM algorithm fixes this cluster's mean to be the position of prosumer 24. Initial covariance matrices are taken to be spherical as in later studies using GMM.


**(a)** Clustering 9 prosumers with regular GMM clustering. Colors of primary prosumers indicate cluster assignments.

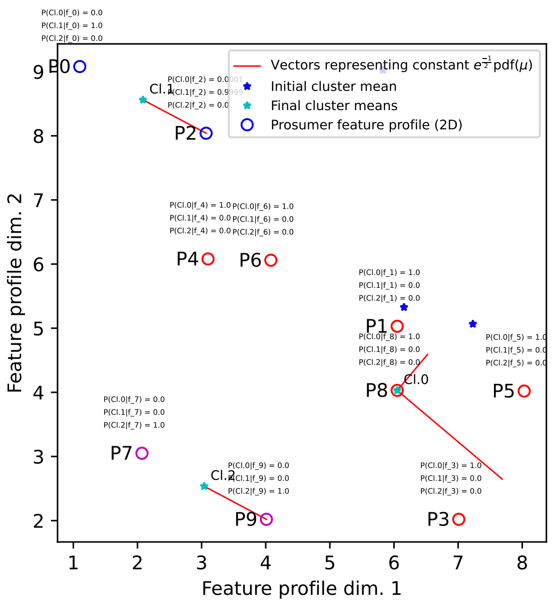

**(b)** The same feature profiles of 9 prosumers, clustered by modified GMM. The primary prosumer in this case is prosumer 8 (P8), with cluster 0 being centered around this prosumer.

**Figure 4-5:** Modified GMM clusters with one Gaussian mean being fixed.

over the non-modified clustering results as the outcome might include single-prosumer clusters. The third scenario is easily detectable and avoidable by resorting to non-modified cluster combinations for that particular primary prosumer (along with being less likely to happen with a larger number of prosumers $N$).

One problem to consider with this method is that by equating the mean of a cluster to the position of the primary prosumer, it is not guaranteed that the primary prosumer will actually belong to this cluster, and this does happen in case studies occasionally.

Details on the expectancy-maximization algorithm can be found in later section 4-3-4.

## 4-3 Case Studies: Set 2 - Alternative Clustering Methods

The section prior (4-2) outlines two case studies that suggest modifications to be applied to the three best performing methods in the paper by Han *et al.* [16].

This section 4-3 will detail some case studies that analyze alternative clustering methods, where either clustering technique or choice of feature profile can be fundamentally different to what is used by Han *et al* [16]. A list of case studies outlines in this section:

Case Study 1: Performance Comparisons between K-medians and K-means
Case Study 2: Fuzzy C-Means (with two optional sub-case studies)
Case Study 3: Comparing Hierarchical-Ward and Hierarchical-average Clustering
Case Study 4: GMM-PCA
Case Study 5: Shape-based Clustering

### 4-3-1 Case Study 1: Performance Comparisons between K-medians and K-means

#### K-medians

K-medians is an alternative clustering method to k-means, that can be useful when the effect of outliers in k-means is such that the mean is drawn away from the majority of instances [41]. Its relevance in the context of clustering feature profiles is that it has even lower sensitivity to outliers compared to k-means, a characteristic preferred by Han *et al.*

In k-medians, the medians (or *geometric medians*) $c_k$ of clusters $k \in K$ minimize the $L_1$-norm of the distances between instances and their respective medians ([17], [20], [21]). For this reason the median is often referred to as the $L_1$-center, compared to the ($L_2$-center) mean (eq. 4-10). The k-medians objective function is shown in equation 4-5. $d(x, c_k)$ is a vector with elements containing distances from prosumers assigned to a cluster $k$ to the cluster median $c_k$.

$$F = \sum_{k=1}^{K} \|d(x, c_k)\|_1, \quad \forall x \in \text{cl.}k \tag{4-5}$$

Given that the Euclidean norm will be utilized as the measure of distance, considering that the $L_1$-norm of ($L_2$) Euclidean distances is taken, the k-medians objective function becomes

$$F = \sum_{k=1}^{K} \sum_{x \in \text{cl.}k} \|x - c_k\|_2 \tag{4-6}$$

The geometric median $c_k$ of a cluster $k$ minimizes the $L_1$-norm of ($L_2$) Euclidean distances between instances and itself

$$c_k = \underset{y}{\text{argmin}} \|d(x, y)\|_1 = \underset{y}{\text{argmin}} \sum_{x \in \text{cl.}k} \|x - y\|_2 \tag{4-7}$$

This contrasts from k-means, where the objective function is

$$F = \sum_{k=1}^{K} \|d(x, c_k)\|_2, \quad \forall x \in \text{cl.}k \tag{4-8}$$

which given that $\sqrt{\cdot}$ is strictly increasing is the same as minimizing (again considering the Euclidean distance vector $d(x, c_k)$ with elements $\|x - c_k\|_2$ for each prosumer instance $x$)

$$F = \sum_{k=1}^{K} \sum_{x \in \text{cl.}k} \|x - c_k\|_2^2 \tag{4-9}$$

This is based on the definition of the mean $c_k$

$$c_k = \underset{y}{\text{argmin}} \|d(x, y)\|_2 = \underset{y}{\text{argmin}} \sum_{x \in \text{cl.}k} \|x - y\|_2^2 \tag{4-10}$$

Weiszfeld's algorithm can be used to iterate towards the geometric median.

Given the Euclidean measure of distance, Weiszfeld's algorithm ([27]) requires the repetitions of the following computations

$$\bar{x}^{k+1} = \frac{\sum_i^N \frac{x_i}{\|x_i - \bar{x}^k\|_2}}{\sum_i^N \frac{1}{\|x_i - \bar{x}^k\|_2}} \tag{4-11}$$

until the median settles at $x_{k+1} \approx x_k$. The sum of distances from a center to all instances is a convex function, so the Weiszfeld algorithm descends into a global minimum.

## 4-3-2 Case Study 2: Fuzzy C-Means & Fuzzy C-Medians

### Fuzzy C-Means

Rajabi *et al.* [29] find that FCM performs the clustering of load profiles almost as well as hierarchical clustering judging by a set of CVI's. Kim *et al.* [19] find that FCM works almost as well as k-means across a wide range of numbers of clusters, and performs better than hierarchical clustering over the whole domain of cluster numbers.

Consideration must be given to the computational time of FCM. Rajabi *et al.* [29] find that the computational time of FCM is much longer than other clustering methods, anywhere between roughly 5 to 100 times slower than k-means, depending on the fuzzifier parameter.

### Fuzzy C-Means - Methodology

Fuzzy c-means is a clustering method that considers the association of each instance to each cluster, in contrast to k-means which categorizes instances discretely amongst clusters, each cluster ignoring instances allocated to other clusters in the determination of the updated cluster mean entirely. FCM can be considered a fuzzy analogue to k-means clustering.

The fuzzy c-means algorithm, minimizing the objective function $F = \sum_{j=1}^{K} \sum_{i=1}^{N} u_{ij}^m \|x_i - c_j\|_2^2$, has the following two steps ([4],[23]):

**1.** Initialize the algorithm with $K$ clusters allocated randomly. Calculate the membership matrix $U = (u_{ij}) \in \mathbb{R}^{N \times K}$ using

$$u_{ij} = \frac{1}{\sum_{k=1}^{K} \left( \frac{\|\mathbf{x}_i - \mathbf{c}_j\|_2}{\|\mathbf{x}_i - \mathbf{c}_k\|_2} \right)^{\frac{2}{m-1}}} \tag{4-12}$$

**2.** Define new cluster centroids as the weighted average of the new membership degrees.

$$c_j = \frac{\sum_{i=1}^{N} u_{ij}^m x_i}{\sum_{i=1}^{N} u_{ij}^m} \tag{4-13}$$

**3.** Reiterate steps 1 and 2 until $\left\| U^{(k+1)} - U^{(k)} \right\| < \epsilon$, where the norm can for instance indicate an array of all modulus values $|u_{ij}^{(k+1)} - u_{ij}^{(k)}|$.

The effect of fuzzifier parameter is such that with $m \to 1$, the membership function returns results same as k-means ($u_{ij} \to 1$ for the nearest cluster, and $u_{ij} \to 0$ for all other clusters). As $m$ grows, the fuzziness of clusters increases towards the point where memberships of a prosumer belonging to $K$ clusters are all equally $\frac{1}{N}$.

### Case Study A: Regular Fuzzy C-Means

This case study involves clustering load profiles for a range of $m \in \{1.6, \ldots, 3\}$ in steps of 0.2 (and including 1.9 as the optimal value found in Rajabi *et al.* [29] for similar half-hour load profiles). This range is wide enough to include various $m$ values found to be optimal for case studies analyzing FCM on load profiles ([42], [29]). While Rajabi *et al.* show that step changes of resolution $m = 0.05$ can detect smaller changes in CVI performances, steps of $m = 0.2$ would be enough to establish the general trends of CVI's as a function of fuzzifier $m$. This keeps the computational time required for all experiments within reason.

### Case Study B - Fuzzy C-Medians

#### Kersten's Fuzzy C-Medians

Papers that study fuzzy c-medians applications implement fuzzy c-medians as outlined by Kersten [18]. Instead of using the geometric median, Kersten finds each cluster's median constructed by the median of each cluster's fuzzy instances mapped onto $P$ individual axes, considering instances $x \in \mathbb{R}^P$. Specifically, the objective function $F = \sum_{j=1}^{K} \sum_{i=1}^{N} u_{ij}^m \sum_{d=1}^{p} |x_i(d) - c_j(d)|$ is minimized by iterating over steps:

**1.** Initialize the algorithm with $K$ clusters allocated randomly. Calculate the membership matrix $U = (u_{ij}) \in \mathbb{R}^{N \times K}$ using

$$u_{ij} = \frac{1}{\sum_{k=1}^{K} \left( \frac{\|\mathbf{x}_i - \mathbf{c}_j\|_1}{\|\mathbf{x}_i - \mathbf{c}_k\|_1} \right)^{\frac{1}{m-1}}} \tag{4-14}$$

**2.** Define new cluster centroids $c_k$ such that $c_k[d], \forall d \in P$ is the weighted median of the 1D projection of instances onto axis $d$, where instances are weighed with $u_{ij}^m$.

#### Euclidean Fuzzy C-Medians

Although not found in literature, it is possible to find fuzzy c-median clusters using the Euclidean distance with the geometric median as a centering statistic instead of Kersten's approach of considering medians across $P$ axes. Figure 4-6 shows that the geometric median and median are not the same. The objective function replaces the $L_2$-norm in the fuzzy c-means objective function with the $L_1$-norm

$$F = \sum_{j=1}^{K} \sum_{i=1}^{N} u_{ij}^m \|x_i - c_j\|_2 \tag{4-15}$$

Deriving the objective function $\frac{dF}{du_{ij}}$ considering the constraints $\sum_{j=1}^{K} u_{ij} = 1, \forall i \in \{1 \ldots N\}$ using Lagrangian multipliers [8], the optimal membership function is

$$u_{ij} = \frac{1}{\sum_{k=1}^{K} \left( \frac{\|\mathbf{x}_i - \mathbf{c}_j\|_2}{\|\mathbf{x}_i - \mathbf{c}_k\|_2} \right)^{\frac{1}{m-1}}} \tag{4-16}$$

Unlike fuzzy c-means, where the new means are identified by setting $\frac{dF}{dc_j} = 0$ and solving for means $c_j$, there is no closed form solution for the median center, and Weiszfeld's iterative

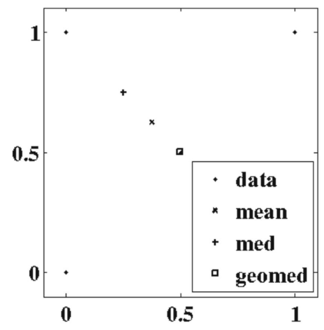


**Figure 4-6:** Different centers for dataset $\begin{bmatrix} 0 & 0 & \frac{1}{2} & 1 \\ 0 & 1 & \frac{1}{2} & 1 \end{bmatrix}$. The axis-wise median and the geometric median are not the same as shown in this simple example [17].

approach can take its place. For a set of prosumers with arbitrary weights $w_i$, the weighted geometric median corresponds to $\underset{\bar{x}}{\text{argmin}} \sum_i^N w_i \left\|x_i - \bar{x}\right\|_2$, and the Weiszfeld algorithm for the weighted geometric median is [7]

$$\bar{x}^{k+1} = \frac{\sum_i^N \frac{w_i x_i}{\|x_i - \bar{x}^k\|_2}}{\sum_i^N \frac{w_i}{\|x_i - \bar{x}^k\|_2}} \tag{4-17}$$

Now considering the Euclidean fuzzy c-medians objective function $F = \sum_{j=1}^K \sum_{i=1}^N u_{ij}^m \left\|x_i - c_j\right\|_2$, the Weiszfeld algorithm iterates

$$c_j^{k+1} = \frac{\sum_i^N \frac{u_{ij}^m x_i}{\|x_i - c_j^k\|_2}}{\sum_i^N \frac{u_{ij}^m}{\|x_i - c_j^k\|_2}} \tag{4-18}$$

Weiszfeld's algorithm is a gradient method and must conform to the gradient descent equation $c_j^{k+1} = c_j^k - s(c_j^k)\frac{dF}{dc_j}$ which it does with step size $s(c_j^k)$ and objective function gradient $\frac{dF}{dc_j}$ being

$$s(c_j^k) = \frac{1}{\sum_i^N \frac{u_{ij}^m}{\left\|x_i - c_j^k\right\|_2}} \tag{4-19}$$

$$\frac{dF}{dc_j} = \sum_i^N \left[ \frac{u_{ij}^m (x_i - c_j)}{\left\|x_i - c_j\right\|_2} \right] \tag{4-20}$$

Figure 4-7 shows the Euclidean fuzzy c-medians algorithm converging to the centers of multiple Cauchy distributions. The Cauchy distribution is a practical distribution for this purpose as it is defined by its median. It was found that the per-axis and geometric median are very similar for these distributions, and the fuzzy c-medians algorithm converges to the geometric median. The Cauchy distribution is the Student's t-distribution with one degree of freedom.

Figure 4-8 shows the Euclidean fuzzy c-medians algorithm converging to Gaussian distributions, for one case where each cluster has an offset Gaussian to represent outliers, and for another case where each cluster consists only of only one Gaussian. The centers converge to the centers of the Gaussians well for both cases with little effect from the outlier Gaussians.

The final test (figure 4-9) duplicates clusters of the type shown in figure 4-6. Each center converges to the geometric median.

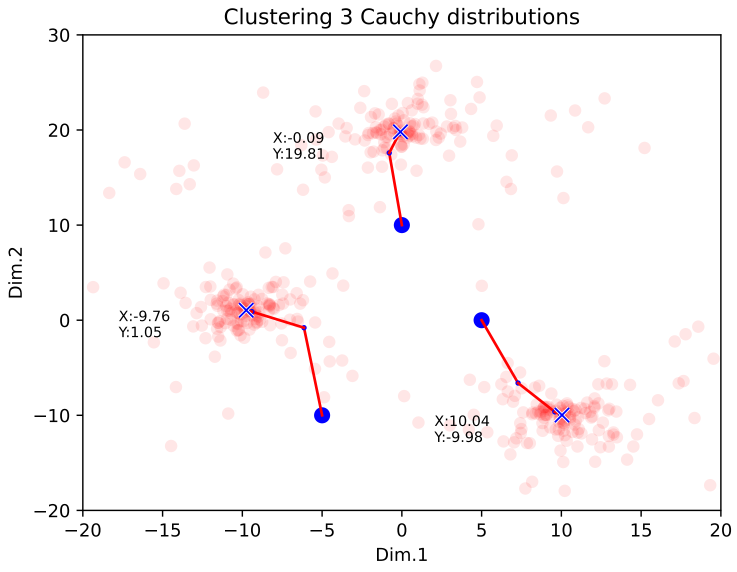


**Figure 4-7:** The evolution of Euclidean fuzzy c-medians towards the median centers of three Cauchy distributions, which are located at $[-10, 1], [0, 20], [10, -10]$, each cluster consisting of 150 samples.

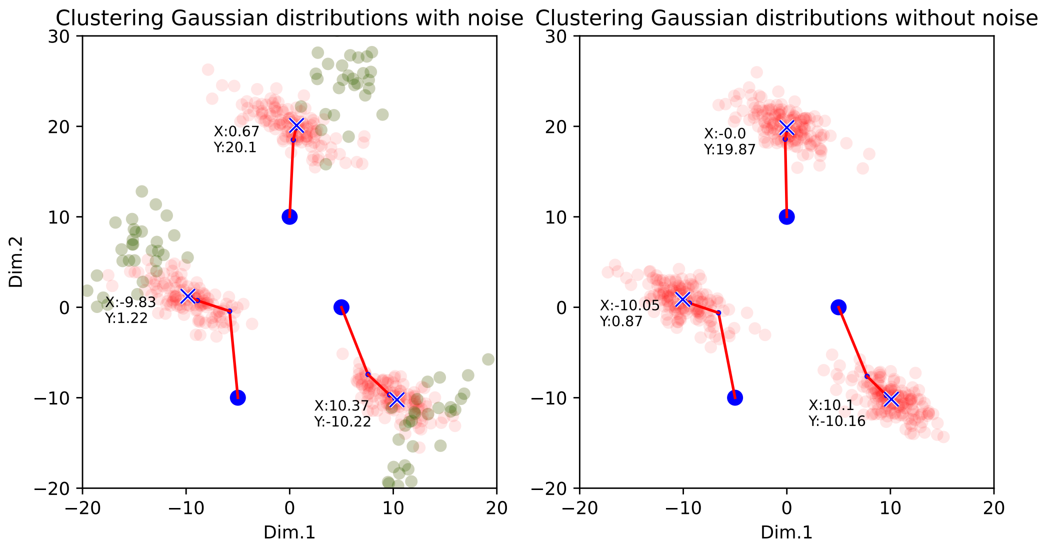


**Figure 4-8:** **Left:** Three clusters are consisted of 120 samples of Gaussian distributions, along with 30 samples of another Gaussian for offset noise. **Right:** The same Gaussians (sample size 150 each) without the Gaussian noise.
A comparison of the two plots shows that the Euclidean fuzzy c-medians centers are relatively stable including or excluding the presence of offset noise.

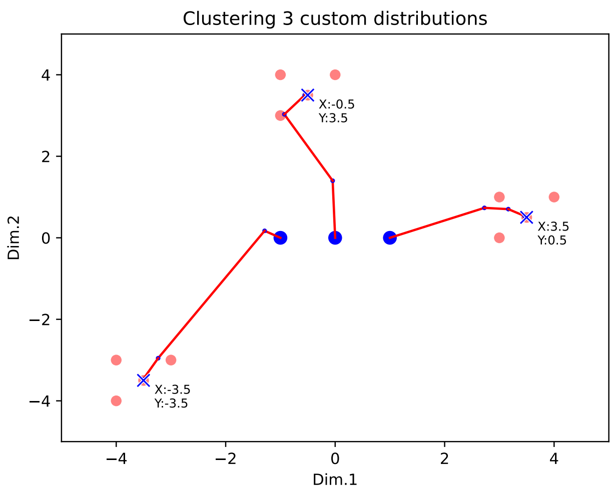


**Figure 4-9:** Applying Euclidean fuzzy c-medians on three clusters of datasets identical to figure 4-6. The centers settle at the geometric medians.

### 4-3-3 Case Study 3: Comparing Hierarchical-Ward, Hierarchical-average, and Hierarchical-complete Clustering

#### Hierarchical Clustering

Hierarchical clustering is a technique that generates clusters either by splitting large clusters into smaller clusters (divisive), or by joining small clusters into larger ones (agglomerative).

Han *et al.* [16] use the agglomerative approach, and it is noted both here and in Rajabi *et al.* [29] that the agglomerative method is the most preferred method for clustering load profiles. One issue with the divisive approach is that two instances, nearby eachother, can still belong to different clusters. The agglomerative approach insures that on an individual basis, each instance is clustered with its nearest neighbor.

#### Linkage Mechanisms

Rajabi *et al.* [29] found that the Ward linkage was the best-performing linkage compared to the single and centroid linkages, despite worse cluster validation index (CVI) performance. The Ward linkage generates even cluster sizes and does not extract outliers, while the single and centroid linkages tend to overwhelmingly group load profiles into one cluster, and distribute outliers amongst the remaining clusters. The Ward linkage was found to be the best in even distribution of instances by Han *et al.* [16] as well, out of the Ward, single, complete and average linkages. However, the results obtained by Han *et al.* solely consist of a single run of clustering. A full case study can compare cluster sizes over multiple iterations, backed by resulting qualities of nucleolus estimations.

A dendrogram showing different cluster development tendencies for the Ward and single linkages is shown in figure 4-10. Horizontal lines in the dendrogram show the composition of clusters as the sum of two sub-ordinate clusters. The height (y-axis) of horizontal lines

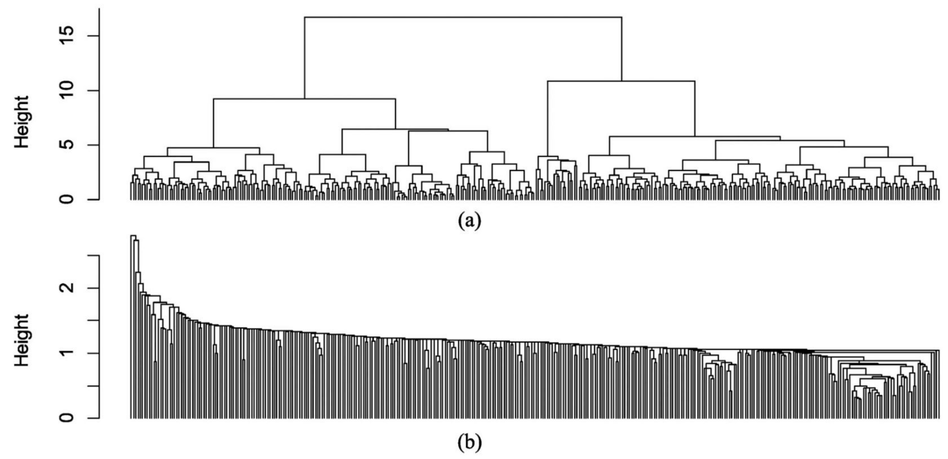


**Figure 4-10:** An example of an agglomerative hierarchical clustering dendrograms. **a)** A dendrogram for the Ward linkage, where cluster sizes grow evenly as the number of clusters is reduced. **b)** A dendrogram for the single linkage, where a single cluster grows as the number of clusters is reduced, resulting in one large cluster and a few smaller ones for outliers. [29].

represent the distance between the two subordinate clusters according to some distance metric and the linkage type.

Chicco [6] also found the single linkage to have the best CVI performance. Despite CVI performances being slightly worse for the average linkage with Euclidean distance (Minkowski exponent $p = 2$), the author states that using the average linkage is more effective for a balance between isolating outliers and somewhat maintaining even cluster sizes. The Ward linkage performed the worst according to CVI's. An interesting observation by Chicco is that CVI's generally represent the attitude of the clustering method towards isolating outliers.

In conclusion, the Ward linkage produces the most even cluster sizes. The average and complete linkages have a stronger tendency to isolate outliers, but maintain some ability to have even cluster sizes for the bulk of the instances. Any benefit of prioritizing even cluster sizes by choosing the Ward linkage is best shown through a comparison of nucleolus estimation quality with the average and complete linkage methods.

### 4-3-4 Case Study 4: GMM-PCA

Rajabi *et al.* [29] use Principal Component Analysis (PCA) to reduce half-hour load profiles ($\mathbb{R}^{48}$ data) into load profiles with 5 to 6 principal components. The authors state that GMM might not work optimally if considering all 48 components of the time-series, and that GMM generally produces better results with limited variables. CVI performance settles as the number of components are increased up to 8 components.

In this case study, the nucleolus estimation performances of clustering methods applied to reduced-dimensionality datasets will be analyzed. In the PCA step, the number of principal components required to reach 99% and 95% variance explained are analyzed. This number of principal components (out of 48) needed to reach a certain variance explained target is one way to reveal the inherent dimensionality of datasets, if ignoring phenomena such as manifolds. Datasets with high inherent dimensionalities suffer from the Curse of Dimensionality

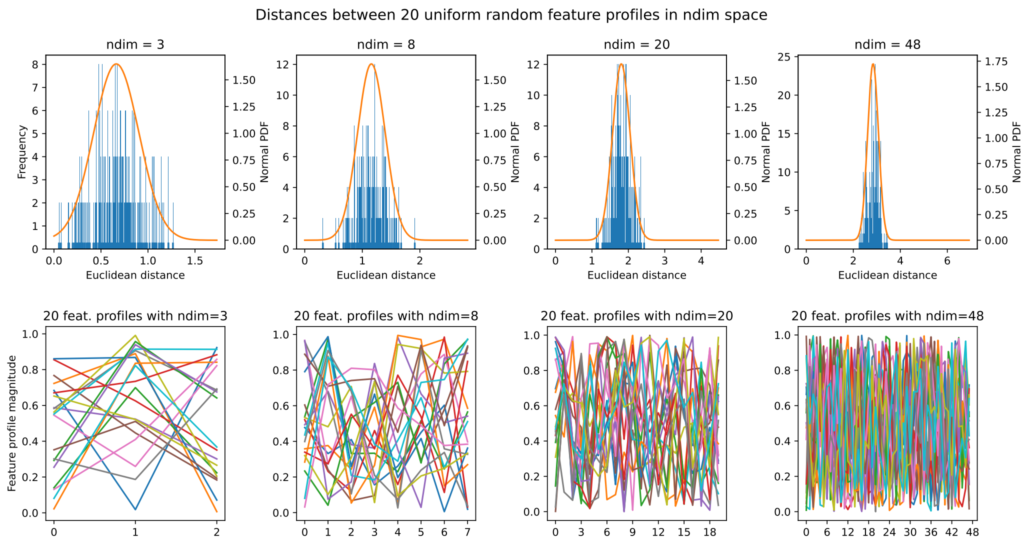


**Figure 4-11:** An example of the creeping effect of Curse of Dimensionality with increasing number of dimensions a prosumer's feature profile contains. Feature profiles are uniformly generated in $ndim$ space, which is not the case with true prosumer feature profiles. It can be investigated whether the Curse of Dimensionality is a problem for our data through the use of PCA.

([40], [37], [3], [9]). where the Euclidean distance metric and Gaussian distribution aren't accurate ways of differentiating between prosumers. Figure 4-11 shows how the notion of distance degrades for uniformly distributed prosumer feature profiles with increasing numbers of dimensions. Gaussian distributions suffer from a similar effect; most datapoints accumulate in the outer shell of the distribution, which doesn't coincide with the shape of a Gaussian distribution which assumes most instances occur nearby the mean.

Considering that figure 4-12 [16] shows k-means and GMM clusters visually aligning with what would be expected, all relevant data for clustering is most likely in a low-dimensional subspace or follows a manifold within $\mathbb{R}^{48}$ space that allows for Euclidean distances and Gaussian distributions to still work well in these areas densely populated by feature profiles. The distribution of prosumer feature profiles is then far enough from being uniformly distributed for the Curse of Dimensionality to be a problem. As a result, it is expected that GMM-PCA cannot perform better than regular GMM in terms of nucleolus estimation performance.

## Gaussian Mixture Models (GMM)

GMM is a probabilistic method that defines a jointed probability distribution function as the weighted average of $K$ Gaussian distributions, defined by $\boldsymbol{\theta} = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K\}$, where $\boldsymbol{\theta}_j = \left(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\right)$ In this case, $\mu_j$ is a $R$-dimensional vector of a Gaussian mean, and $\boldsymbol{\Sigma}_j$ an $R \times R$ covariance matrix. The joint PDF can be defined as a weighted combination of these $K$ Gaussians, where $\alpha_j \geq 0$, $\sum_{j=1}^{K} \alpha_j = 1$.

$$p\left(\mathbf{f}_i \mid \boldsymbol{\theta}\right) = \sum_{j=1}^{K} \alpha_j p\left(\mathbf{f}_i \mid \boldsymbol{\theta}_j\right) \tag{4-21}$$

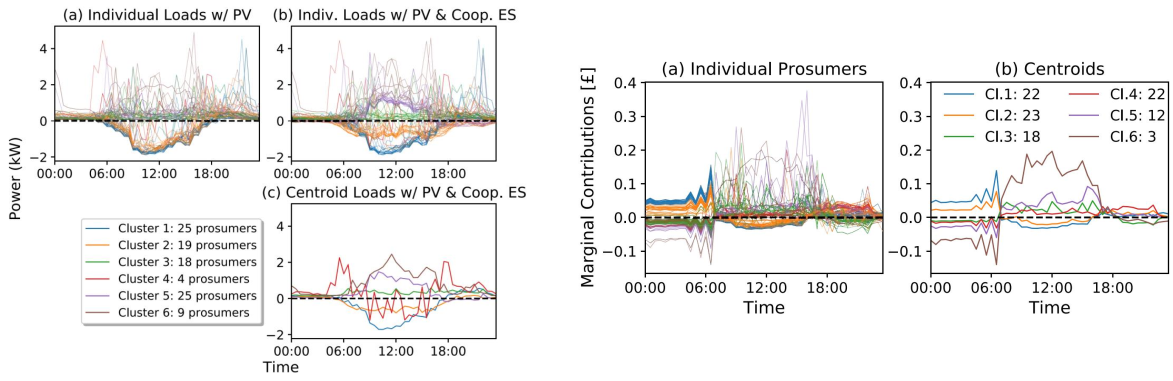


**Figure 4-12: Left:** Clustering 100 prosumers with k-means using the grand coalition cooperative energy profiles shows clusters that align reasonably with what would be expected. **Right:** Clustering 100 prosumers with GMM using the grand coalition marginal contribution profiles also shows clusters forming that align with what would be expected.

Weighting compensates for unevenly populated distributions of feature profiles, allowing the joint distribution function to vary the overall sizes of Gaussians to account for the number of prosumers belonging to them.

The variables $\{\boldsymbol{\mu}, \boldsymbol{\Sigma}, \boldsymbol{\alpha}\}$ need to be found with methods such as expectation-maximization for the weighted sum of Gaussians to best represents the true probability density function representing all feature profiles. The feature profile instance $\mathbf{f}_i$ has the probability it belongs to cluster $k$ (defined by Gaussian $\theta_{\mathbf{k}}$ with weight $\alpha_k$) defined as

$$p\left(k \mid \mathbf{f}_i\right) = \frac{\alpha_k p\left(\mathbf{f}_i \mid \boldsymbol{\theta}_k\right)}{\sum_{j=1}^{K} \alpha_j p\left(\mathbf{f}_i \mid \boldsymbol{\theta}_j\right)}, \quad k = 1, \ldots, K \tag{4-22}$$

In all case studies involving GMM, the full covariance structure will be used as it generates the most even cluster sizes, a good clustering characteristic in the prevalence of noise.

### Expectation-Maximization

Expectation-maximization happens in two steps:

#### 1) Expectancy Step:

Gaussian parameters $\mu_k$, $\Sigma_k$ and weights $\alpha_k$ are fixed. Initially, these are chosen randomly. A common choice is to select the initial covariances of the clusters to be the data covariance. Weights can be initialized as $\frac{1}{K}$. All posterior probabilities $p\left(k \mid \mathbf{f}_i\right)$ are computed as

$$p\left(k \mid \mathbf{f}_i\right) = \frac{p\left(\mathbf{f}_i \mid \boldsymbol{\theta}_k, \alpha_k\right)}{\sum_{j=1}^{K} p\left(\mathbf{f}_i \mid \boldsymbol{\theta}_j, \alpha_j\right)}, \quad k = 1, \ldots, K \tag{4-23}$$

given that the weighted likelihood of a prosumer $f_i$ belongs to a particular multivariate normal distribution is

$$p\left(\mathbf{f}_i \mid \boldsymbol{\theta}_j, \alpha_j\right) = \alpha_j \frac{\exp\left\{-\frac{1}{2}\left(\mathbf{f}_i - \boldsymbol{\mu}_j\right)^{\top} \boldsymbol{\Sigma}_j^{-1}\left(\mathbf{f}_i - \boldsymbol{\mu}_j\right)\right\}}{(2\pi)^{\frac{R}{2}} \left|\boldsymbol{\Sigma}_j\right|^{\frac{1}{2}}} \tag{4-24}$$

**2) Maximization Step:**
Considering probabilities $p\,(k \mid \mathbf{f}_i)$, the Gaussian parameters $\mu_k, \Sigma_k$ and weights $\alpha_k$ are computed.

Weights of Gaussian $k$ are updated to be the sum of all feature profiles' probabilities of stemming from $k$, normalized for the resulting PDF to have a total probability of 1.

$$\alpha_k = \frac{\sum_k p\,(k \mid \mathbf{f}_i)}{N} \tag{4-25}$$

Gaussian means $\mu_k$ are updated per the mean of feature profiles, weighted by each feature profile's probabilities of belonging to $k$. This insures that Gaussian clusters move towards a position that better represents the underlying prosumer feature profiles' probabilities.

$$\mu_k = \frac{\sum_i [p\,(k \mid \mathbf{f}_i)\, f_i]}{\sum_i p\,(k \mid \mathbf{f}_i)} \tag{4-26}$$

Similarly the full-covariance matrix is updated as

$$\Sigma_k = \frac{\sum_i [p\,(k \mid \mathbf{f}_i)\,(f_i - \mu_k)^T\,(f_i - \mu_k)]}{\sum_i p\,(k \mid \mathbf{f}_i)} \tag{4-27}$$

Iterations of the two steps above result in finding a local minimum. For a global minimum, multiple iterations of GMM are required, of which the solution with the highest maximum likelihood estimation is chosen by computing and comparing the log likelihoods $l\,(\boldsymbol{\theta} \mid \mathbf{f})$ of each model. The highest log likelihood model is the best fit.

### Log-scale GMM

An issue encountered in the GMM-PCA case studies was that the GMM algorithm could not find valid cluster assignments (have one or more empty clusters) in some cases. GMM initialization using k-means clusters is not an alternative, as the final GMM clusters remain identical to the k-means initialization. Initializing GMM with a small covariance as done by default in the `SKLearn GaussianMixture` package would solve this issue for a particular dataset, but result in `NaN` values in $p\,(k \mid \mathbf{f}_i)$ for other datasets. This was fixed by adopting log-scale GMM as used in the `SKLearn GaussianMixture` package instead of the standard GMM method. This method is used in every application of GMM in this thesis.

Standard GMM

$$p\,(\mathbf{f}_i \mid \boldsymbol{\theta}_j, \alpha_j) = \alpha_j \mathcal{N}\,(\mathbf{f}_i \mid \boldsymbol{\theta}_j)$$

$$p\,(k \mid \mathbf{f}_i) = \frac{p\,(\mathbf{f}_i \mid \boldsymbol{\theta}_k, \alpha_k)}{\sum_{j=1}^{K} p\,(\mathbf{f}_i \mid \boldsymbol{\theta}_j, \alpha_j)}, \qquad k = 1, \ldots, K$$

$$l\,(\boldsymbol{\theta} \mid \mathbf{f}) = \sum_{i=1}^{N} \log \left[ \sum_{j=1}^{K} p\,(\mathbf{f}_i \mid \boldsymbol{\theta}_j, \alpha_j) \right]$$

Log GMM

$$\log p\,(\mathbf{f}_i \mid \boldsymbol{\theta}_j, \alpha_j) = \log \alpha_j + \log \mathcal{N}\,(\mathbf{f}_i \mid \boldsymbol{\theta}_j)$$

$$p\,(k \mid \mathbf{f}_i) = \exp \left[ \log p\,(\mathbf{f}_i \mid \boldsymbol{\theta}_k, \alpha_k) - \log \left[ \sum_{j=1}^{K} \exp\left(\log p\,(\mathbf{f}_i \mid \boldsymbol{\theta}_j, \alpha_j)\right) \right] \right]$$

$$k = 1, \ldots, K$$

$$l\,(\boldsymbol{\theta} \mid \mathbf{f}) = \sum_{i=1}^{N} \log \left[ \sum_{j=1}^{K} \exp\left(\log p\,(\mathbf{f}_i \mid \boldsymbol{\theta}_j, \alpha_j)\right) \right]$$

The log-scale GMM equations above have been deduced from the `SKLearn GaussianMixture` files `_gaussian_mixture.py` and `_base.py`. For faster computation of the multivariate distribution function values, $\log \mathcal{N}(\mathbf{f}_i \mid \boldsymbol{\theta}_j)$ is computed with the method used in `scipy.stats.multivariate_normal`, detailed in [12]. Another crucial step is the use of `SciPy` function `logsumexp()` in the calculations of $p(k \mid \mathbf{f}_i)$ and log likelihood for numerical reasons.

The issue with the standard GMM is that most values in $\mathcal{N}(\mathbf{f}_i \mid \boldsymbol{\theta}_j)$ are zero with smaller covariance matrix initializations (for example a covariance matrix with $1e^{-6}$ diagonal values), where $\mathcal{N}(\mathbf{f}_i \mid \boldsymbol{\theta}_j)$ is taken to be the exponent of $\log \mathcal{N}(\mathbf{f}_i \mid \boldsymbol{\theta}_j)$, while $\log \mathcal{N}(\mathbf{f}_i \mid \boldsymbol{\theta}_j)$ itself detects non-zero values. As a result, using $\log p(\mathbf{f}_i \mid \boldsymbol{\theta}_j, \alpha_j)$ with function `logsumexp()` gives a working $p(k \mid \mathbf{f}_i)$ unlike standard GMM which can contain mainly `NaN`'s.

The covariance initialization adopted from the same `SKLearn GaussianMixture` function takes the covariance to be the identity matrix multiplied by $1e{-6}$, a default regularization factor.

## Principal Component Analysis

Principal component analysis is a technique that generates a set of orthogonal axes, ordered by their statistical significance. The first principal component (PC) is the axis that generates maximum variance of the data.

The computation of the principal components and the variance along these axes is found with the eigenvectors (direction of principal axes) and eigenvalues (variance along principal axes) of the covariance matrix. This outcome allows for simple dimension reduction; principal components with the smallest variances can be disregarded.

For data in $\mathbb{R}^R$ space, the variance contributed by a principal axis $PC_r, r \leq R$ compared to total data variance can be calculated as

$$\text{Var. Contr.}_{(PC_r)} = \frac{\text{Var.}_{(PC_r)}}{\sum_r^R \text{Var.}_{(PC_r)}} \tag{4-28}$$

A common method to set a bound to the number of principal components is to set the target cumulative variance of the first $Q$ principle components $\sum_r^Q (\text{Var. Contr.}_{(PC_r)})$ to 99%.

### 4-3-5 Case Study 5: Shape-based Clustering

So far, all clustering methods cluster feature profiles by magnitude at each time interval. Characteristics determined by ES charging, discharging limits and capacity, and PV generation characteristics, are all well represented in these feature profiles.

This case study clusters prosumers not by magnitude of load, but by the shape of the load profile. The only difference with shape-based clustering is that the load profile of each prosumer must be normalized before the clustering step. Like this, prosumers with similar consumption behaviours and characteristics (PV, ES ownership) can be grouped regardless of differences in feature profile magnitudes. Generally, this method may work better in scenarios where PV and ES capacities are not binary (either 0 or at fixed capacities defined in Han *et al.* [16]), as no variable PV and ES capacities exist which could be better grouped by shape after a normalizing step.

# Chapter 5

# Computation of the Nucleolus Estimation

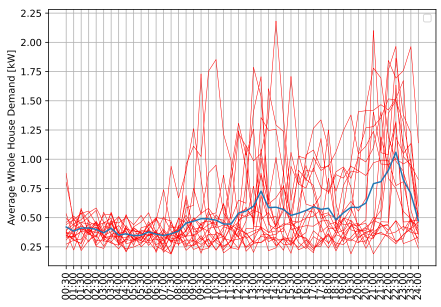
## 5-1 Load Profile Data

Prosumer profiles have been sourced from [30] (download link in [31]). This dataset gives daily individual prosumer demand and generation profiles. To reduce the resolution from minute to half-hour intervals, the average kilowatt reading of all minutes within each half-hour are taken. A plot of 20 load profiles drawn from a batch of 2782 individual load profiles (after filtering for load profiles with incomplete or erroneous data, and sorting for July) is shown in figure 5-1a. PV data has been sourced from PVWatts [28] using variables used by Han *et al.*: 4kW PV systems, a fixed 20 degree tilt, in London Gatwick. Half-hour measurements are interpolated from hourly interval data by taking the average of half-hour prior and half-hour after measurement. Individual PV profiles correspond to specific days within the PVWatts dataset.

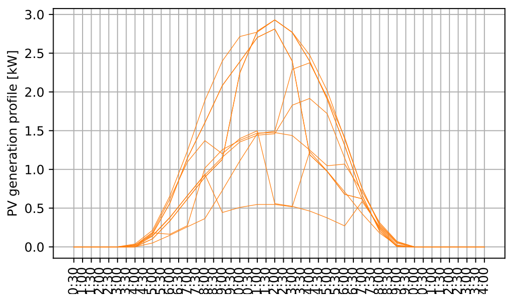
## 5-2 Variables

Variables are taken from Han *et al.* [16]:

| | |
|---|---|
| Electricity price (00:00-07:00) ($r_t^{im}$): £0.08/kWh | Minimum battery SoC ($\underline{SoC_i}$): $0.2e$ |
| Electricity price (07:00-24:00) ($r_t^{im}$): £0.18/kWh | Maximum battery SoC ($\overline{SoC_i}$): $0.95e$ |
| Electricity price (feed in) ($r_t^{ex}$): £0.0379/kWh | Initial battery SoC ($SoC_i^0$): $0.5e$ |
| Charging efficiency $\eta_{\text{in}}$: 0.95 | Battery discharge limit $\underline{b}_i$: $-3.2kW \times 0.5hrs$ |
| Discharging efficiency $\eta_{\text{out}}$: 0.95 | Battery charge limit $\overline{b}_i$: $3.5kW \times 0.5hrs$ |
| Individual battery capacity $e$: 7 kWh | |



**(a)** 20 prosumer demand profiles, with the average plotted in blue.



**(b)** A set of 10 PV generation profiles based on data from PVWatts ([28]), as would be used for 10-run nucleolus estimations.

**Figure 5-1:** An example of 20 prosumer demand profiles, along with 10 sets of PV data as used in the case studies.

## 5-3 Cooperative P2P Operation - Linear Programming

### 5-3-1 Minimization Problem

The minimization problem is to find battery operations $b_{it}^+$ and $b_{it}^-$ that minimizes the overall cost of a coalition of prosumers. With $L_t^+ = max(0, \sum_{i\in N}(b_{it}^+ + b_{it}^- + q_{it}))$, and $L_t^- = min(0, \sum_{i\in N}(b_{it}^+ + b_{it}^- + q_{it}))$, the objective function to minimize is

$$\min_{\mathbf{b}^+,\mathbf{b}^-,\mathbf{L}^+,\mathbf{L}^-} \quad \sum_{t=1}^{48} \left[ r_t^{im} L_t^+ + r_t^{ex} L_t^- \right] \tag{5-1}$$

$$\text{s.t.} \quad L_t^- \leq 0 \leq L_t^+ \tag{5-2}$$

$$\sum_{i\in N}(b_{it}^+ + b_{it}^- + q_{it}) \leq L_t^+ \tag{5-3}$$

$$\sum_{i\in N}(b_{it}^+ + b_{it}^- + q_{it}) = L_t^+ + L_t^- \tag{5-4}$$

$$0 \leq b_{it}^+ \leq \overline{b}_i \tag{5-5}$$

$$\underline{b}_i \leq b_{it}^- \leq 0 \tag{5-6}$$

$$e_i\underline{SoC}_i \leq e_i SoC_i^0 + \sum_{t=1}^{r}\left(b_{it}^+\eta_i^{\text{in}} + b_{it}^-/\eta_i^{\text{out}}\right) \leq e_i\overline{SoC}_i, \quad \forall r \in [1, 48] \tag{5-7}$$

$$\sum_{t=1}^{48}\left(b_{it}^+\eta_i^{in} + b_{it}^-/\eta_i^{\text{out}}\right) = 0 \tag{5-8}$$

### 5-3-2 Constraints

The following constraints are applied:

5-2: Pairs the correct sign of net coalition demand with the appropriate import or export price.

5-3, 5-4: Complete the definition of $L_t^+ = max(0, \sum_{i\in N}[b_{it}^+ + b_{it}^- + q_{it}])$, and $L_t^- = min(0, \sum_{i\in N}[b_{it}^+ + b_{it}^- + q_{it}])$

5-5: Sets bounds on the battery charging variables ([kWh])

5-6: Sets bounds on the battery discharging variables ([kWh])

5-7: Sets bounds any instantaneous battery SoC ([kWh])

5-8: Net battery charge over all time intervals $t \in \{1, ..., 48\}$ must be zero.

### 5-3-3 Python Implementation

#### Objective function

The chosen state vector of dimension $[96(n+1) \times 1]$ is

$$x = [\,\underbrace{\overbrace{[-b^+-]}^{[1x48]}[-b^--]}_{i=1}\cdots\underbrace{[-b^+-][-b^--]}_{i=n}\;\vdots\;\overbrace{[-L^+-]}^{[1x48]}[-L^--]]^T$$

A valid objective function in form $\min c^T x$ has

$$c^T = [\,\underbrace{\overbrace{[-0-]}^{[1x48]}[-0-]}_{i=1}\cdots\underbrace{[-0-][-0-]}_{i=n}\;\vdots\;\overbrace{[-r^{im}-]}^{[1x48]}[-r^{ex}-]]$$

#### Constraint 5-2 ($L_{it}^- \leq 0 \leq L_{it}^+$)

The constraint splits into
1) $-L_t^+ \leq 0$
2) $L_t^- \leq 0$.
Considering the state vector $x$, a vectorized version of these constraints would be:
1)

$$\underbrace{\begin{bmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \\ 0 & & 0 \end{bmatrix}}_{[48\times 96n]}\underbrace{\begin{bmatrix} -1 & & \\ & \ddots & \\ & & -1 \end{bmatrix}}_{[48\times 48]}\underbrace{\begin{bmatrix} 0 & & \\ & \ddots & \\ & & 0 \end{bmatrix}}_{[48\times 48]} x \leq \underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}}_{[48\times 1]} \tag{5-9}$$

2)

$$\underbrace{\begin{bmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \\ 0 & & 0 \end{bmatrix}}_{[48\times 96n]}\underbrace{\begin{bmatrix} 0 & & \\ & \ddots & \\ & & 0 \end{bmatrix}}_{[48\times 48]}\underbrace{\begin{bmatrix} 1 & & \\ & \ddots & \\ & & 1 \end{bmatrix}}_{[48\times 48]} x \leq \underbrace{\begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}}_{[48\times 1]} \tag{5-10}$$

**Constraint 5-3 ($\sum_{i\in N}(b^+_{it} + b^-_{it} + q_{it}) \leq L^+_t$)**

$$\left[\underbrace{\begin{matrix}1 & & & 1 & & \\ & \ddots & & & \ddots & \\ & & 1 & & & 1\end{matrix}}_{i=1, dim=[48\times 96]}\middle|\cdots\middle|\underbrace{\begin{matrix}1 & & & 1 & & \\ & \ddots & & & \ddots & \\ & & 1 & & & 1\end{matrix}}_{i=n, dim=[48\times 96]}\middle|\underbrace{\begin{matrix}-1 & & \\ & \ddots & \\ & & -1\end{matrix}}_{[48\times 48]}\middle|\underbrace{\begin{matrix}0 & & \\ & \ddots & \\ & & 0\end{matrix}}_{[48\times 48]}\right] x \leq \underbrace{\begin{bmatrix}-\sum_i(q_{i,t=1})\\ \vdots \\ -\sum_i(q_{i,t=48})\end{bmatrix}}_{[48\times 1]} \tag{5-11}$$

**Constraint 5-4 ($\sum_{i\in N}(b^+_{it} + b^-_{it} + q_{it}) = L^+_t + L^-_t$)**

$$\left[\underbrace{\begin{matrix}1 & & & 1 & & \\ & \ddots & & & \ddots & \\ & & 1 & & & 1\end{matrix}}_{i=1, dim=[48\times 96]}\middle|\cdots\middle|\underbrace{\begin{matrix}1 & & & 1 & & \\ & \ddots & & & \ddots & \\ & & 1 & & & 1\end{matrix}}_{i=n, dim=[48\times 96]}\middle|\underbrace{\begin{matrix}-1 & & \\ & \ddots & \\ & & -1\end{matrix}}_{[48\times 48]}\middle|\underbrace{\begin{matrix}-1 & & \\ & \ddots & \\ & & -1\end{matrix}}_{[48\times 48]}\right] x \leq \underbrace{\begin{bmatrix}-\sum_i(q_{i,t=1})\\ \vdots \\ -\sum_i(q_{i,t=48})\end{bmatrix}}_{[48\times 1]} \tag{5-12}$$

**Constraint 5-5, 5-6 ($0 \leq b^+_{it} \leq \bar{b}, \underline{b}_i \leq b^-_{it} \leq 0$)**

Both constraints are enforced by applying bounds on the state vector $x$ as follows

$$\Big[\underbrace{\overbrace{[-0-]}^{[1x48]}[-\underline{b}_i-]}_{i=1}\cdots\underbrace{[-0-][-\underline{b}_i-]}_{i=n}\ \Big\vert\ \overbrace{[-\texttt{None}-]}^{[1x48]}[-\texttt{None}-]\Big]^T \tag{5-13}$$

$$\leq x \leq$$

$$\Big[\underbrace{\overbrace{[-\bar{b}_i-]}^{[1x48]}[-0-]}_{i=1}\cdots\underbrace{[-\bar{b}_i-][-0-]}_{i=n}\ \Big\vert\ \overbrace{[-\texttt{None}-]}^{[1x48]}[-\texttt{None}-]\Big]^T \tag{5-14}$$

**Constraint 5-7 ($e_i\underline{SoC}_i \leq e_iSoC^0_i + \sum_{t=1}^r \left(b^+_{it}\eta^{\textsf{in}}_i + b^-_{it}/\eta^{\textsf{out}}_i\right) \leq e_i\overline{SoC}_i, \quad \forall r \in [1, 48]$)**

In block (diagonal) matrix form, two inequality constraints are

$$\left[\underbrace{\begin{matrix}A_{i=1} & \cdots & 0\\ \vdots & \ddots & \\ 0 & & A_{i=n}\end{matrix}}_{[48n\times 96n]}\middle|\underbrace{\begin{matrix}0 & \cdots & 0\\ \vdots & & \vdots\\ 0 & \cdots & 0\end{matrix}}_{[48n\times 96]}\right] x \leq \underbrace{\begin{bmatrix}B_{i=1}\\ \vdots\\ B_{i=n}\end{bmatrix}}_{[48n\times 1]} \tag{5-15}$$

and

$$-\left[\underbrace{\begin{matrix}A_{i=1} & \cdots & 0\\ \vdots & \ddots & \\ 0 & & A_{i=n}\end{matrix}}_{[48n\times 96n]}\middle|\underbrace{\begin{matrix}0 & \cdots & 0\\ \vdots & & \vdots\\ 0 & \cdots & 0\end{matrix}}_{[48n\times 96]}\right] x \leq \underbrace{\begin{bmatrix}C_{i=1}\\ \vdots\\ C_{i=n}\end{bmatrix}}_{[48n\times 1]} \tag{5-16}$$

with each prosumer $i$ with assigned ES has

$$A_i = \underbrace{\left[\begin{array}{cccc|cccc} \eta_i^{\text{in}} & 0 & \dots & 0 & \frac{1}{\eta_i^{\text{out}}} & 0 & \dots & 0 \\ \eta_i^{\text{in}} & \eta_i^{\text{in}} & \dots & 0 & \frac{1}{\eta_i^{\text{out}}} & \frac{1}{\eta_i^{\text{out}}} & \dots & 0 \\ \vdots & & \ddots & & \vdots & & \ddots & \\ \eta_i^{\text{in}} & & & \eta_i^{\text{in}} & \frac{1}{\eta_i^{\text{out}}} & & & \frac{1}{\eta_i^{\text{out}}} \end{array}\right]}_{[48\times 96]}, B_i = \underbrace{\begin{bmatrix} e_i\overline{SoC}_i - e_iSoC_i^0 \\ \vdots \\ e_i\overline{SoC}_i - e_iSoC_i^0 \end{bmatrix}}_{[48\times 1]}, C_i = \underbrace{\begin{bmatrix} e_iSoC_i^0 - e_i\underline{SoC}_i \\ \vdots \\ e_iSoC_i^0 - e_i\underline{SoC}_i \end{bmatrix}}_{[48\times 1]}$$

For prosumers without ES, $B_i$ and $C_i$ are replaced with zero vectors.

**Constraint 5-8 ($\sum_{t=1}^{48}\left[b_{it}^{+}\eta^{\text{in}} + b_{it}^{-}\frac{1}{\eta^{\text{out}}}\right] = 0$)**

This constraint which forces a zero net change in battery SoC is implemented to limit the arbitrage period to the forecast horizon, insuring the coalitional cost savings (and allocated nucleolus payoffs) are not inflated by a "short-sighted" sell-off of previously stored energy.

The constraint can be formulated as

$$\left[\begin{array}{ccc|ccc} D_{i=1} & \dots & 0 & 0 & \dots & 0 \\ \vdots & \ddots & & \vdots & & \vdots \\ 0 & & D_{i=n} & 0 & \dots & 0 \end{array}\right] x = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} \qquad (5\text{-}17)$$

with the blocks of size $[n\times 96n]$, $[n\times 96]$ and $[n{\times}1]$,

where

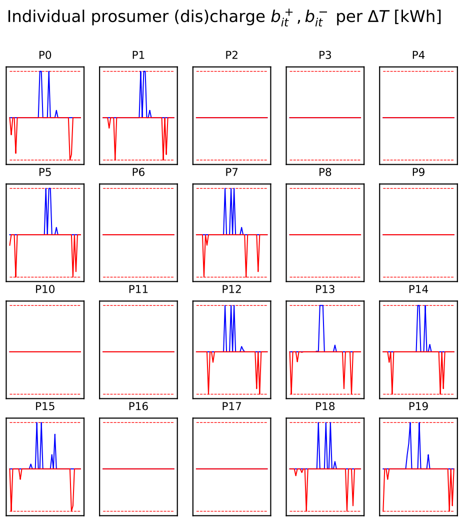
$$D_i = [\overbrace{[-\eta_i^{\text{in}}-]}^{[1\text{x}48]}[-\frac{1}{\eta_i^{\text{out}}}-]]$$

## 5-4 Cooperative P2P Operation - Linear Programming Tests

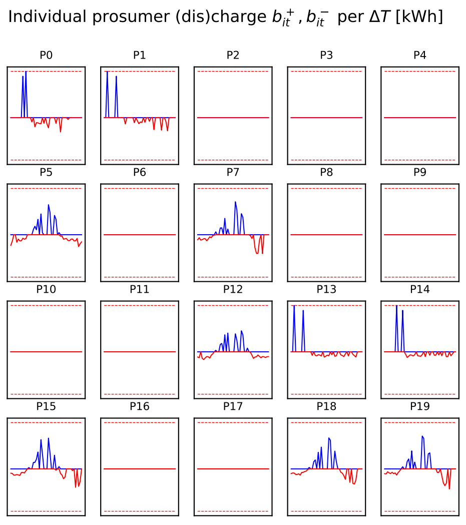
This section covers a number of checks that can be done to confirm operation of a coalition using cooperative ES operation.

### 5-4-1 Prosumer battery operation

The first check is to confirm that for any $t \in [1, 48]$, $b_{it}^{+}b_{it}^{-} = 0$. Figure 5-2 shows two example. The first example is fora 20 prosumer coalition, and the other is for the same 20 prosumers in a non-coalitional setting. $b_{it}^{+}b_{it}^{-} = 0$ holds true for both cases. Prosumers without ES have $b_{it}^{+}, b_{it}^{-} = 0$. Charge and discharge limits $\overline{b}_i, \underline{b}_i$ are shown as dashed lines.



**(a)** N=20 coalitional battery operations.



**(b)** N=20 non-coalitional battery operations.

**Figure 5-2:** Prosumer battery operations for a 20-prosumer coalitional and non-coalitional case. Battery charge and discharge limits $\underline{b}_i$ and $\overline{b}_i$ are shown as dashed lines.

### 5-4-2 Net coalitional load $L_t^+, L_t^-$

The second check is to confirm if the minimization formulation generates net coalition loads $L_t^+, L_t^-$ such that $L_t^+ L_t^- = 0$. The results are shown in figure 5-3. One interesting side note is that in this example, the net load is never negative as a result of the cooperative scheme. All imported and generated energy is effectively consumed, and no loss is made by exporting energy to the grid at reduced prices.

### 5-4-3 Cooperative and non-cooperative coalition cost evolution

The cooperative and non-cooperative cost evolution plots are shown in figure 5-4.

### 5-4-4 ES operation and coalition net demand matching

Correct cooperative battery operation has to match the coalition net demand. The cooperative case is shown in figure 5-5a. Here, cooperative battery operation matches the net coalition demand well. ES discharge never exceeds net coalition demand, meaning energy stored is never sold at feed-in prices. ES charging matches available excess PV generation, reducing reliance on imported electricity.

5-5b shows the non-cooperative ES operation in the context of coalition demand for comparison. Worse matching characteristics of the ES operation are observed. This is also reflected in the simultaneous charging and discharging of the aggregate ES system, where the coalition cost is increased by redundant charging and discharging efficiencies.

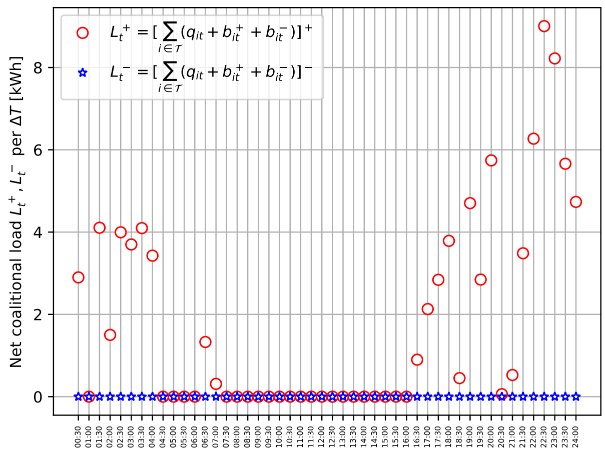
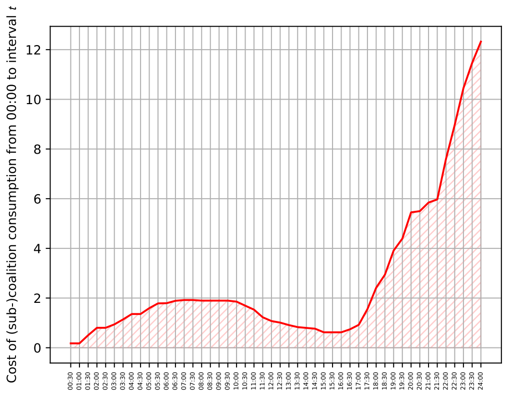
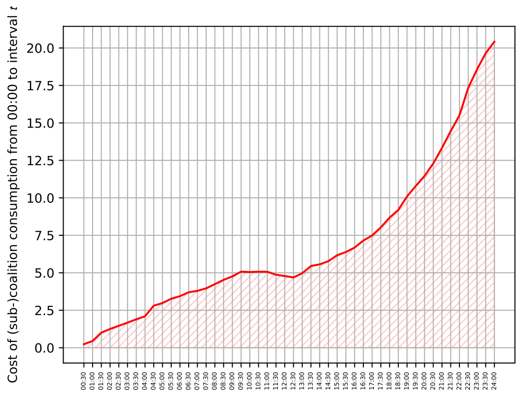


**Figure 5-3:** A plot of the magnitude of net coalitional load $L_t^+, L_t^-$ for a coalition $\mathcal{T}$ with N=20.

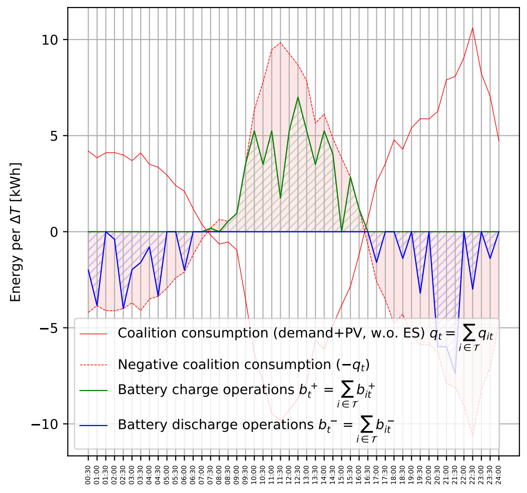


**(a)** Cooperative cost evolution for an N=20 coalition.

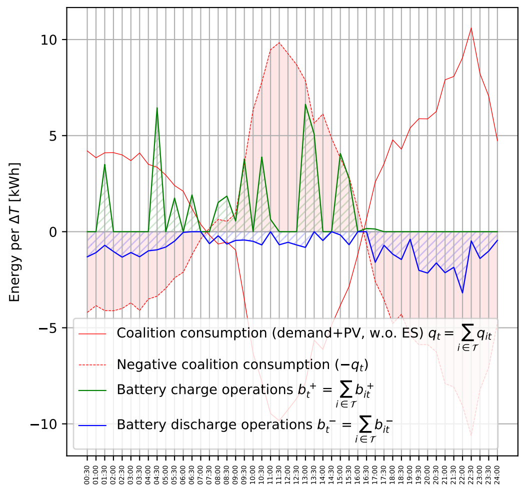


**(b)** Non-cooperative cost evolution for the same N=20 coalition.

**Figure 5-4:** The cooperative and non-cooperative cost evolutions of an N=20 coalition.



**(a) Cooperative ES operation**: Cooperative battery operation matches the net coalition demand well. ES discharge never exceeds net coalition demand, meaning energy stored is never sold at feed-in prices. ES charging matches available excess PV generation, reducing reliance on imported electricity.



**(b) Non-cooperative ES operation**: Worse matching characteristics of the ES operation are observed. This is also reflected in the simultaneous charging and discharging of the aggregate ES system, where the coalition cost is increased by losses of simultaneous charging and discharging efficiencies.

**Figure 5-5:** The cooperative and non-cooperative cost evolution of an N=20 coalition.

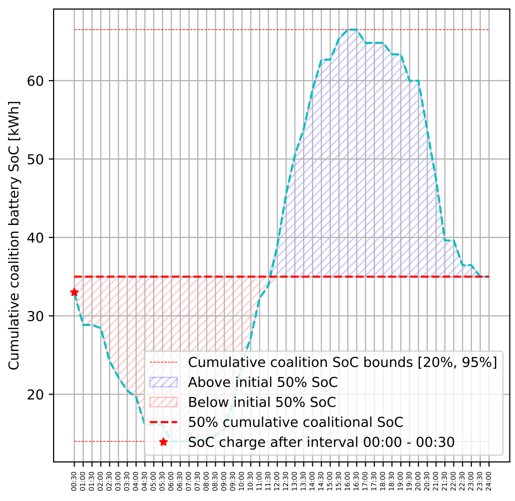


**Figure 5-6:** Cumulative coalition ES SoC.

### 5-4-5 Aggregate coalition SoC evolution

Figure 5-6 shows the state of charge of all ES batteries acting as a single battery. Performance is conform to the constraint $\sum_{t=1}^{48}\left[b_{it}^{+}\eta^{\text{in}} + b_{it}^{-}\frac{1}{\eta^{\text{out}}}\right] = 0$ resulting in a final SoC of 50%. The state of charge is also conform to a maximum and minimum battery SoC (min: $n_{ES}\left(\underline{SoC} \times e\right)$, max: $n_{ES}\left(\overline{SoC} \times e\right)$), where $\underline{SoC} \times e$ is the minimum SoC for any prosumer ES considering $\underline{SoC}$ and battery capacity $e$ is equal for any prosumer with allocated ES, and $n_{ES} = 10$, as 50% of prosumers in the coalition are allocated ES.

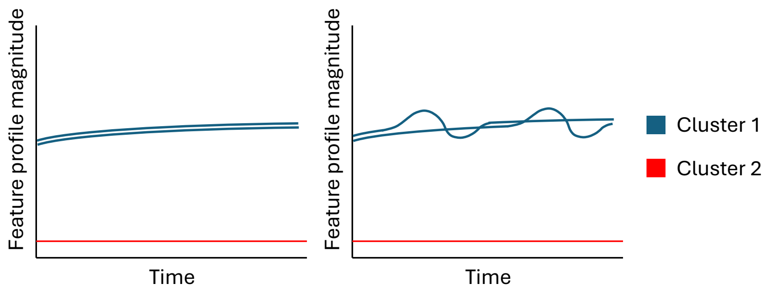


**Figure 5-7:** The randomness of ES (dis)charge peaks affects distances between prosumers. The left figure shows two prosumers in cluster 1, with identical feature profiles and ES operations as a result of incorporating CEPmin or MCPmin. In the right figure, the distance between the prosumers in cluster 1 increases due to non-synchronous ES operations, resulting in the relative distance between both prosumers and a third prosumer in cluster 2 becoming smaller.

## 5-5 Comparison of ES Operation to Plots in Han *et al.*

Figure 5-8 shows that the ES operation presented in Han *et al.* [16] shows different characteristics from the results presented here. The authors' battery operations (figure 5-8c) seem to be identical across all prosumers, which is not a specific outcome of their minimization problem. This would coincide with setting all battery operations in figure 5-2a equal to each other. While this section will not try to exactly duplicate the feature profiles in Han *et al.*, their feature profiles show that less erratic ES operations are possible compared to the ones resulting from the LP in 5-3-1. This section investigates this.

An issue that could stem from (dis)charge peaks caused by the randomness in optimal ES operations present in figure 5-8a is that the Euclidean distance between prosumers that should otherwise form one cluster becomes larger, making the relative distance to load profiles in other clusters smaller, potentially resulting in suboptimal clusters (figure 5-7).

Figure 5-8b shows feature profiles where each prosumer has equal battery operations. Instead of computing the optimization for one single ES system, a shortcut to compute shared ES battery operation given the minimization problem in equation 5-1) is to sum the resulting ES operations $b_t^{sum} = \sum_i b_{it}$, and give each prosumer their portion of the sum of operations. Conditions that all battery variables $\overline{b}_i, \underline{b}_i, \underline{SoC}_i, \overline{SoC}_i, SoC_i^0$ are maintained, and are also equal for all prosumers. Figure 5-8a and 5-8b shows the comparison between individual cooperative ES operations compared to an even division of battery operations $b_{it} = \frac{b_t^{sum}}{n_{ES}}$. The resulting ES operations in figure 5-8b are equal across all prosumers, just as shown by Han *et al.* [16] in figure 5-8c. However, the erraticness of ES operations is still much larger. This equalness in ES operations can only hold if all parameters of prosumers are identical.

Another attempt to remove high peaks of battery (dis)charging considers an individual (dis)charge limit $\overline{b}_i, \underline{b}_i$ that is also minimized in the LP problem 5-18, giving results shown in figure 5-9. The advantage of this method is that it can increase longevity of ES systems with slower charging rates and reduce losses caused by unnecessarily high charging currents. Visually, it often provides a better split amongst cluster groups and may improve clustering results, which might in turn be reflected in better nucleolus estimations. This new feature profile can be considered in case studies. A nice coincidence is that it resembles the feature profiles used by Han *et al.* quite well.

The modified objective function has weights somewhat arbitrarily chosen such that if $\bar{b}_i = \bar{\bar{b}}_i$ and $\underline{b}_i = \underline{\underline{b}}_i$ with all $N$ prosumers owning ES, the objective function does not change by more than £0.005. $\bar{\bar{b}}_i$ and $\underline{\underline{b}}_i$ are the maximum battery charge and discharge limits, listed in section 5-2. Any other weights could be chosen as long as an excessive cost reduction by $\bar{b}_i$ and $\underline{b}_i$ leads to a comparatively even greater cost increase in $\sum_{t=1}^{48} \left[ r_t^{im} L_t^+ + r_t^{ex} L_t^- \right]$. A check for this is to compare $\sum_{t=1}^{48} \left[ r_t^{im} L_t^+ + r_t^{ex} L_t^- \right]$ for both minimization problem outputs to confirm they are equal for any set of feature profiles used for simulation, but for certainty this check would have to be applied to all subcoalitions for which the costs are computed, which is far from practical.

An analytical approach showing that an excessive reduction of $\bar{b}_i$ and $\underline{b}_i$ leads to a comparatively even greater cost increase in $\sum_t \left[ r_t^{im} L_t^+ + r_t^{ex} L_t^- \right]$ for any coalition, compares the cost reduction in $\frac{0.0025}{\sum_{i \in N} \bar{\bar{b}}_i} \sum_{i \in N} \bar{b}_i + \frac{0.0025}{\sum_{i \in N} \underline{\underline{b}}_i} \sum_{i \in N} \underline{b}_i$ by reducing $\bar{b}$, $\underline{b}$ to 0 from $\bar{\bar{b}}$, $\underline{\underline{b}}$ , to the loss in profit of a lowest-profit ES operation scenario where one battery (instead of 15, for a lowest profit scenario where only one prosumer has a battery) charges up by $min(\bar{\bar{b}}, -\underline{\underline{b}})$ in half an hour, and discharges this same amount in the next half-hour, for the least possible profit: ES utilization can vary between A) storing PV energy instead of buying day-tariff energy, B) storing PV energy instead of buying night-tariff energy, and C) storing night-tariff energy instead of buying day-tariff energy. Situation B is the least profitable with $0.08 \frac{£}{\text{kWh}}$ generated, given the prices of energy listed in section 5-2. Considering (dis)charging efficiencies of 95%, the profit is $0.08(0.95)^2 min(\bar{\bar{b}}, -\underline{\underline{b}})$. With $\bar{\bar{b}} = \frac{3.5}{2}\text{kWh}, \underline{\underline{b}} = -\frac{3.2}{2}\text{kWh}$, the profit is £0.1155. Reducing $\bar{b}$, $\underline{b}$ to 0 from $\bar{\bar{b}}$, $\underline{\underline{b}}$ decreases the objective function by $2(0.0025) = £0.005$. As a result, to minimize the objective function, optimization software cannot afford to excessively reduce $\bar{b}$ and $\underline{b}$ as the cost increase in $\sum_{t=1}^{48} \left[ r_t^{im} L_t^+ + r_t^{ex} L_t^- \right]$ will be much bigger.

With $L_t^+ = max(0, \sum_{i \in N} (b_{it}^+ + b_{it}^- + q_{it}))$, and $L_t^- = min(0, \sum_{i \in N} (b_{it}^+ + b_{it}^- + q_{it}))$, the minimization problem becomes

$$\min_{\mathbf{b}^+, \mathbf{b}^-, \mathbf{L}^+, \mathbf{L}^-, \underline{\mathbf{b}}, \bar{\mathbf{b}}} \quad \sum_{t=1}^{48} \left[ r_t^{im} L_t^+ + r_t^{ex} L_t^- \right] + \frac{0.0025}{\sum_{i \in N} \bar{\bar{b}}_i} \sum_{i \in N} \bar{b}_i + \frac{0.0025}{\sum_{i \in N} \underline{\underline{b}}_i} \sum_{i \in N} \underline{b}_i \tag{5-18}$$

$$\text{s.t.} \quad L_t^- \leq 0 \leq L_t^+ \tag{5-19}$$

$$\sum_{i \in N} (b_{it}^+ + b_{it}^- + q_{it}) \leq L_t^+ \tag{5-20}$$

$$\sum_{i \in N} (b_{it}^+ + b_{it}^- + q_{it}) = L_t^+ + L_t^- \tag{5-21}$$

$$0 \leq b_{it}^+ \leq \bar{b}_i \tag{5-22}$$

$$0 \leq \bar{b}_i \leq \bar{\bar{b}}_i \tag{5-23}$$

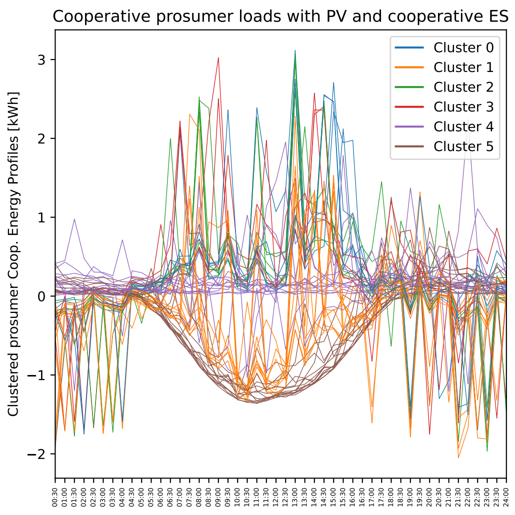
$$\underline{b}_i \leq b_{it}^- \leq 0 \tag{5-24}$$

$$\underline{\underline{b}}_i \leq \underline{b}_i \leq 0 \tag{5-25}$$

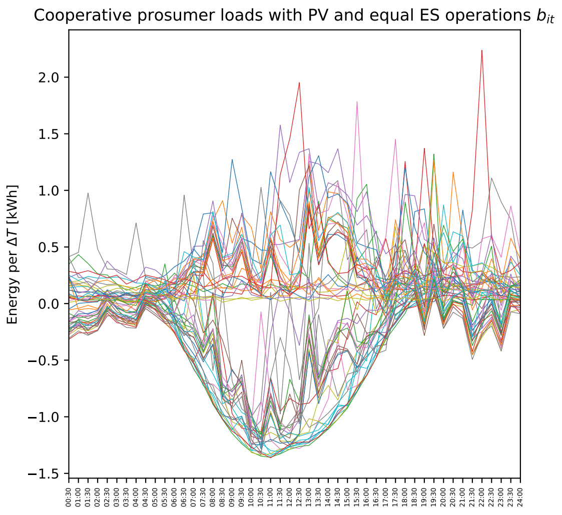
$$e_i \underline{SoC}_i \leq e_i SoC_i^0 + \sum_{t=1}^{r} \left( b_{it}^+ \eta_i^{\text{in}} + b_{it}^- / \eta_i^{\text{out}} \right) \leq e_i \overline{SoC}_i, \quad \forall r \in [1, 48] \tag{5-26}$$

$$\sum_{t=1}^{48} \left( b_{it}^+ \eta_i^{in} + b_{it}^- / \eta_i^{\text{out}} \right) = 0 \tag{5-27}$$

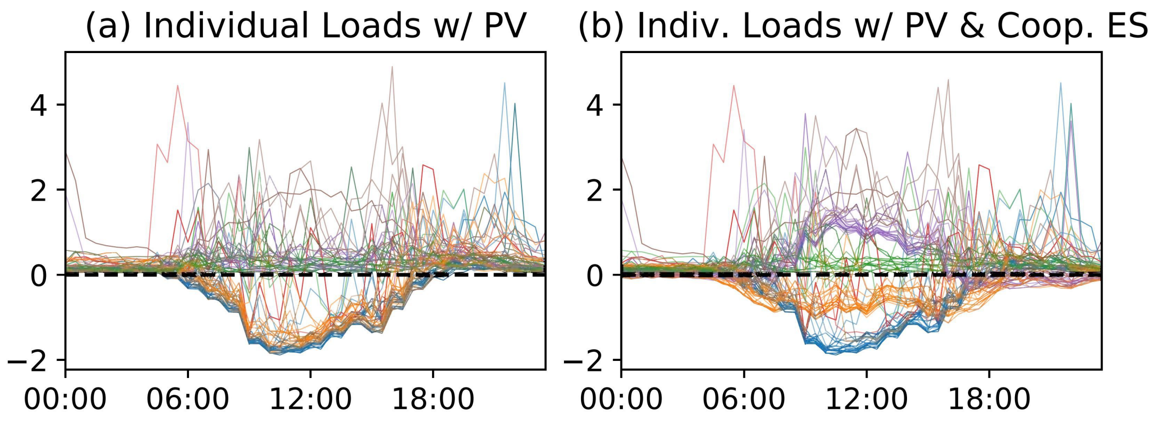
ES operations for both minimization problems considering dataset 9 are shown in figure 5-10. It is best mentioned here that with both costs $\sum_{t=1}^{48}\left[r_t^{im}L_t^+ + r_t^{ex}L_t^-\right]$ being the same using both LP problems, the nucleolus estimation code can use any ES operation from any one of both LP problems as long as coalitional costs savings $v(\mathcal{T}) = \sum_{i\in\mathcal{T}} C(\{i\}) - C(\mathcal{T})$ only consider $C = \sum_{t=1}^{48}\left[r_t^{im}L_t^+ + r_t^{ex}L_t^-\right]$ without the additional $\frac{0.0025}{\sum_{i\in N}\bar{\bar{b}}_i}\sum_{i\in N}\bar{b}_i + \frac{0.0025}{\sum_{i\in N}\underline{\underline{b}}_i}\sum_{i\in N}\underline{b}_i$. This can save time developing code. Only the clustering code has to adopt the correct LP problem for the desired ES operation.



**(a)** Prosumer loads with cooperative ES operation computed with minimization problem 5-1. Note the peaks of ES charging and discharging.

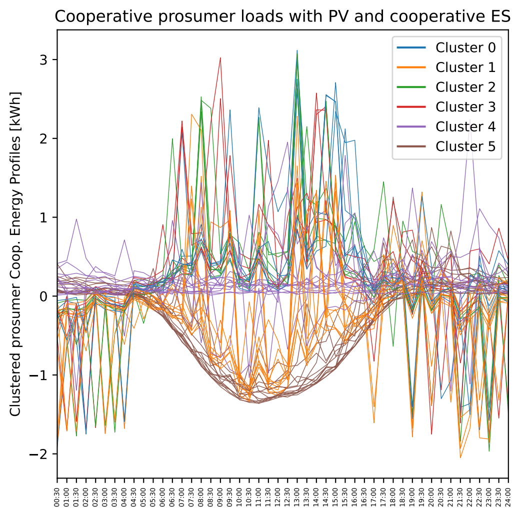


**(b)** The same simulation with the same cost function value, except each prosumer with ES is assigned $b_{it} = b_t^{sum}/n_{ES}$ (with $b_t^{sum} = \sum_i b_{it}$), which is identical across all prosumers with ES.

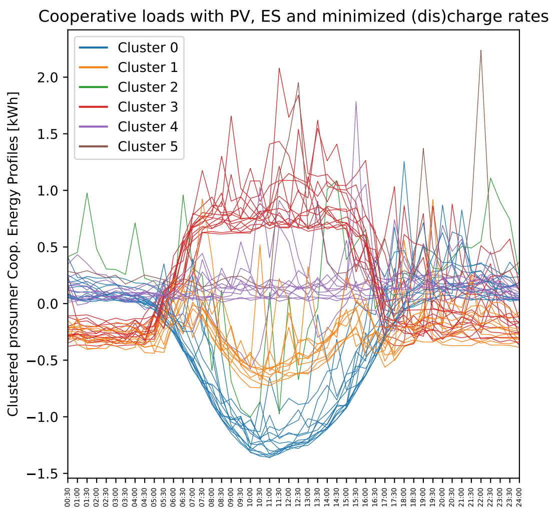


**(c)** Both plots regard one simulation, where the difference between them is the cooperative ES operation. Cluster assignments in both plots are equal, and prosumers are grouped into 6 clusters. The right plot shows that ES operation is uniform amongst all ES owning prosumers.

**Figure 5-8:** A comparison between plots in Han *et al.* [16] (subfigure **c**), the results of clustering purely with the minimization function specified in Han *et al.* (subfigure **a**), and results of splitting the sum of battery operations evenly amongst prosumers (subfigure **b**), to attempt to replicate the style of ES operations evident in subfigure **c**.

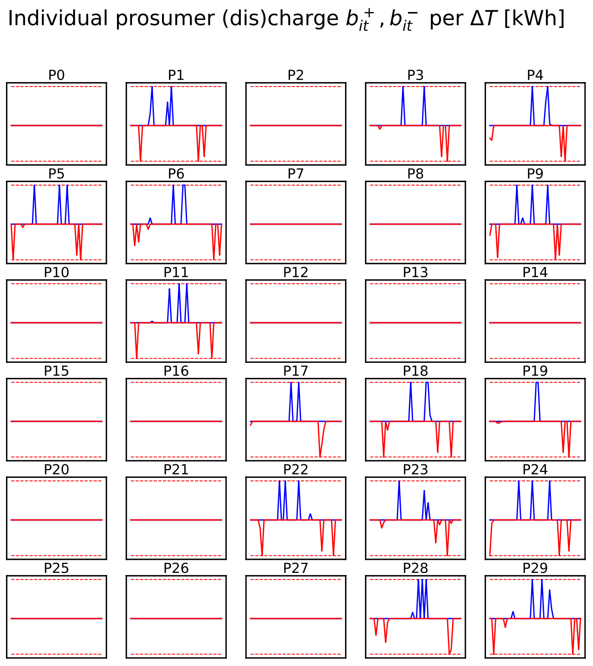


**(a)** Prosumer loads with cooperative ES operation computed with minimization problem 5-1, where (dis)charge limits $\overline{b}_i, \underline{b}_i$ are fixed bounds.

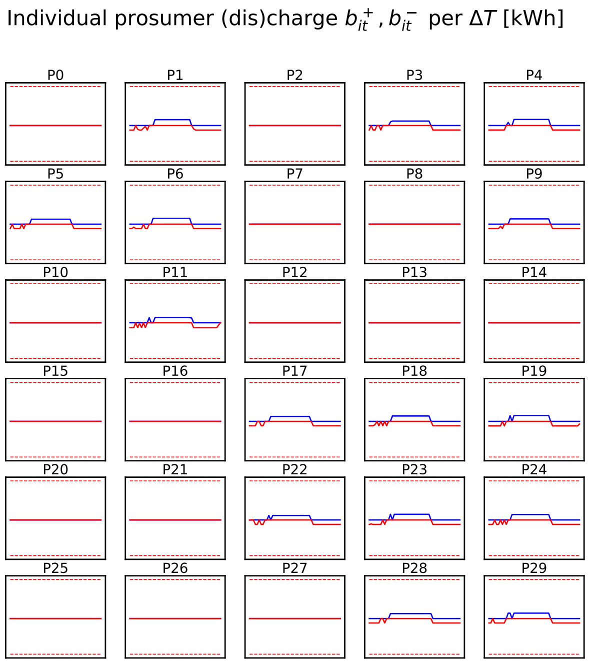


**(b)** The same loads, combined with cooperative ES operation including $\overline{b}_i, \underline{b}_i$ as additional variables in the minimization objective function. The grand coalition cost is the same in both plots.

**Figure 5-9:** These figures compare the cooperative ES operation as **a)** the result of the minimization problem as specified by Han *et al.*, **b)** the result of a modified optimization problem where $\overline{b}_i, \underline{b}_i$ are additional variables to be minimized. The prevalence of ES peaks are removed, giving a better visual overview of the 4 main categories of prosumer (with and without PV/ES).



**(a)** Battery operations without minimized maximum (dis)charge rates.



**(b)** Battery operations with minimized maximum (dis)charge rates.

**Figure 5-10:** Prosumer battery operations for a 30-prosumer coalitional and non-coalitional case, considering dataset 9. The coalitional cost $\sum_{t=1}^{48}\left[r_t^{im}L_t^{+}+r_t^{ex}L_t^{-}\right]$ for the left and right case are exactly the same. Battery charge and discharge limits $\underline{b}_i$ and $\overline{b}_i$ are shown as dashed lines.

## 5-6 Nucleolus Computation and Evaluation

Having defined the cost savings formulation for any coalition in section 5-3-1, the nucleolus algorithm can be implemented.

### 5-6-1 Nucleolus Estimation Algorithm

The algorithm for estimating the nucleolus used by Han *et al.* [16] is shown in algorithm 1.

**Algorithm 1** Lexicographical Minimization of Excesses

```
$\mathfrak{T}_0 \leftarrow \{\mathcal{N}\}$
$w(\mathcal{N}) \leftarrow v(\mathcal{N})$
$a \leftarrow 1$
$\mathfrak{T}^* \leftarrow \mathfrak{T}^{\mathcal{N}} = \{\mathcal{T} | \mathcal{T} \subseteq \mathcal{N}\}$
while $a \leq 2^N$ do
    $LP_a: \; \varepsilon_a = \min_{\mathbf{x}_a, \varepsilon} \varepsilon$
        s.t. $\sum_{\forall i \in \mathcal{T}} x_{a_i} = w(\mathcal{T}), \forall \mathcal{T} \in \mathfrak{T}_{a-1}$        (5-28)
             $v(\mathcal{T}) - \sum_{\forall i \in \mathcal{T}} x_{a_i} \leq \varepsilon, \forall \mathcal{T} \in \mathfrak{T}^* \backslash \mathfrak{T}_{a-1}$
    $\mathbf{T}_a \leftarrow \{\mathcal{T} | v(\mathcal{T}) - \sum_{\forall i \in \mathcal{T}} x_{a_i} = \varepsilon_a, \forall \mathbf{x}_a\}$        ▷ binding constraint(s)
    $w(\mathcal{T}) \leftarrow v(\mathcal{T}) - \varepsilon_a, \forall \mathcal{T} \in \mathbf{T}_a$
    $\mathfrak{T}_a \leftarrow \mathfrak{T}_{a-1} \cup \mathbf{T}_a$
    if $\mathfrak{T}_a = \mathfrak{T}^*$ then
        end while
    else
        $a \leftarrow a + 1$
        continue while
$\mathbf{x}^* = \mathbf{x}_a$
return $\mathbf{x}^*$
```

### 5-6-2 Nucleolus Algorithm - Binding Constraints

The equation $\{\mathcal{T} | v(\mathcal{T}) - \sum_{\forall i \in \mathcal{T}} x_{a_i} = \varepsilon_a, \forall \mathbf{x}_a\}$ in algorithm 1 refers to the set of all coalitions which have binding inequality constraints in a particular minimization iteration (eq. 5-28).

A binding constraint is a constraint that lies coincident to *all* possible optimal solutions of an LP. In the nucleolus computation, binding constraints in each iteration of the LP are identified by looking at the dual variables. The complementary slackness theorem ([39]) states that $(b - Ax^*)^T y^* = 0$, where $b - Ax^* \geq 0$ and $y \geq 0$. If the optimization result returns a positive dual variable $y^*$, the corresponding primal constraint must be binding ($Ax^* = b$).

It appears to be a common mistake ([11]) that binding constraints are taken to be constraints where $A_i x^* = b_i$ holds, which is too broad of a definition. A solution $x^*$ can lie against two constraints, one binding and the other non-binding, however moving the non-binding constraint does not move the minimized objective function.

#### Dual variables in SciPy

The SciPy package `linprog` shows non-zero dual variables as negative values, however they should be regarded as positives.

### 5-6-3 LP method for lexicographical minimization in SciPy

Relying on `highs-ds` or `highs-ipm` in `scipy.optimize.linprog` can cause errors in the lexicographical minimization of the nucleolus estimation payoff vector. For $(N, K) = (20, 9)$, `highs-ds` works for 9 out of 10 iterations, while `highs-ipm` works for 6 out of 10 iterations. Method `highs` should be avoided as it chooses between `highs-ds` and `highs-ipm` automatically. Nucleolus estimation code can use `highs-ds` primarily, and in case of a fault, repeat the calculation with `highs-ipm`. For most case studies, there is only a very small likelihood of both methods not working (two case studies in sections 6-2-2 and 6-2-3 are exceptions, however enough results were able to have been computed to show the clustering modifications do not perform favorably). Impact on computation time can be disregarded as faults usually occur near the start of the lexicographical minimization.

This issue might be dealt with by addressing the numerical issues mentioned in Guajardo & Jörnsten [11].

### 5-6-4 Verifying the Nucleolus Algorithm

The nucleolus algorithm results coincide with results found in examples listed by Guajardo & Jörnsten [11]. Special note has to be taken for the last two examples (3.5 and 3.6), where the nucleolus is an imputation that represents the optimal cost-sharing amongst consumers, rather than the sharing of cost *savings*. With the algorithm in Han *et al.* [16], the last two nucleoli are achieved by taking $v(\mathcal{S})$ as a negative number, as it can be considered that

$$\text{coalitional cost savings} = -\text{coalitional cost}$$

Considering these last two games this way shows monotonicity, i.e. for example 3.5: $v(3, 4) = -12 \geq v(3) + v(4) = -21$.

Results have also been checked against three examples in a forum discussion [10].

## 5-7 Nucleolus Estimation - Example

An example of nucleolus estimation is presented. Subcoalition values (cost savings $v(\mathcal{T})$ for some coalition $\mathcal{T}$) come from a simulation using input data as presented in this thesis, however someone can use the coalition values below as inputs into nucleolus estimation code to verify the resulting nucleolus estimations and lexicographical ordering of excesses and corresponding subcoalitions. For a clustering algorithm that allocated prosumers such that $cl_1 = \{0, 1\}$ and $cl_2 = \{2, 3\}$, considering diversity-based pairing, the reduced set of subcoalitions excludes subcoalitions $\{0, 2\}$, $\{0, 3\}$, $\{1, 2\}$, $\{1, 3\}$. Coalition values considered for the full nucleolus are (with red cells showing coalition values excluded from the nucleolus estimation)

| Coalition $\mathcal{T}$ | {0} | {1} | {2} | {3} | {0,1} | {0,2} | {0,3} | {1,2} | {1,3} |
|---|---|---|---|---|---|---|---|---|---|
| $v(\mathcal{T})$ | 0 | 0 | 0 | 0 | 0 | 0.9859 | 0 | 0.7249 | 0 |

| Coalition $\mathcal{T}$ | {2,3} | {0,1,2} | {0,1,3} | {0,2,3} | {1,2,3} | {0,1,2,3} |
|---|---|---|---|---|---|---|
| $v(\mathcal{T})$ | 0.6774 | 1.3521 | 0 | 1.3174 | 1.1784 | 1.5867 |

Nucleoli:
The nucleolus payoff allocations for the full nucleolus and nucleolus estimations are identical:

| Prosumer | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| Full nucleolus | 0.2041 | 0.1347 | 1.1306 | 0.1173 |
| Nucleolus estimation | 0.2041 | 0.1347 | 1.1306 | 0.1173 |

Binding coalitions in each LP iteration:

| Full nucleolus subcoalitions | Excess | Nucleolus estimation subcoalitions | Excess |
|---|---|---|---|
| {0,1,2} | -0.1173 | {0,1,2} | -0.1173 |
| {3} | -0.1173 | {3} | -0.1173 |
| {0,2,3} | -0.1347 | {0,2,3} | -0.1347 |
| {1} | -0.1347 | {1} | -0.1347 |
| {0} | -0.2041 | {0} | -0.2041 |
| {1,2,3} | -0.2041 | {1,2,3} | -0.2041 |
| {1,3} | -0.2520 | | |
| {0,3} | -0.3215 | | |
| {0,1} | -0.3388 | {0,1} | -0.3388 |
| {0,2} | -0.3488 | | |
| {0,1,3} | -0.4561 | {0,1,3} | -0.4561 |
| {1,2} | -0.5404 | | |
| {2,3} | -0.5705 | {2,3} | -0.5705 |
| {2} | -1.1306 | {2} | -1.1306 |

## 5-8 Nucleolus Estimation Evaluation - Stratified Random Sampling & Divisible Excess

This section details the implementation of the nucleolus estimation evaluation technique used by Han *et al.* [16] to quantify the quality of nucleolus estimation (section 3-5).

### 5-8-1 Stratified Random Sampling Implementation

The set of all possible permutations $\pi(\mathcal{N})$ as described in Han *et al.* [16] is a hypothetical construct as computation time is excessive for larger $N$. Considering this, the steps used to compute the percentage of sampled coalitions with divisible excess is as follows

1. Set initial sample size $m = 1$.
2. For each stratum sample set $M_{il}$, if the sample size $|M_{i,l}| < m_{il}^{max}$, generate one new unique sub-coalition sample. The maximum sample size is $m_{il}^{max} = min(m, max.comb.)$ for $l \geq 1$, and $m_{il}^{max} = 1$ for $l = 0$. $max.comb.$ is the maximum number of combinations $N-1$ objects (prosumers) with $l$ samples can generate, and defines the maximum number of possible sampled coalitions a particular stratum $M_{il}, l \geq 1$ can contain.

   Considering indexing from 0 ($i, l \in (0, N-1)$), a sample is created by taking the prosumer set $[0, \ldots, N-1]$, then removing prosumer $i$ from this list, and shuffling the order of remaining prosumers. Prosumer $i$ is reinserted in position $l$, and any prosumers after position $l$ are cut out of the list. The sub-coalition is added to $M_{il}$ if successfully tested against other sampled sub-coalitions in $M_{il}$ for uniqueness.
3. Calculate $v(\mathcal{T})$ for all sampled coalitions in $M_{il}, \forall i \in [0, N-1], \forall l \in [0, N-2]$. This is a slightly reduced set with respect to Han *et al.*, where sampled coalitions in column $M_{i,l=N-1}$ are excluded as they all refer to permutations of the grand coalition which are redundant as they can never generate a divisible excess. Sample excesses $\varepsilon_m(\boldsymbol{v}^{est}, \mathcal{T}), \forall \mathcal{T} \in M_{i\in[0,N-1],l\in[0,N-2]}$ are computed. The number of sampled coalitions with positive excess $|\varepsilon_m^{pos}|$ is counted across all $M_{i,l}$.
4. Repeat steps 1-4 with $m = m + 1$, unless $|\frac{|\varepsilon_m^{\text{pos}}|}{|\mathfrak{R}_m|} - \frac{|\varepsilon_{m-1}^{\text{pos}}|}{|\mathfrak{R}_{m-1}|}| \leq 0.5\%$. $\mathfrak{R}_m$ is the set of all sampled coalitions in $M$ with $m$ samples specified for each $M_{il}$. If this ending condition is met, the number of divisible excesses (where it holds that the fraction $\frac{\varepsilon_m(\boldsymbol{v}^{est}, \mathcal{T})}{|\mathcal{T}|} \geq 0.005\pounds$) can be counted, and the fraction of sampled subcoalitions with divisible excess can be computed.

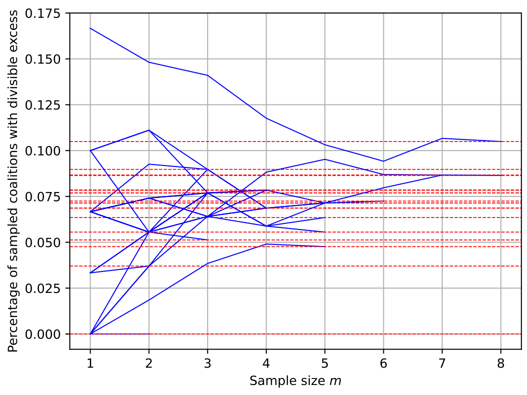
Figure 5-11a shows a plot containing 20 runs of the stratified sampling algorithm applied to a single game of 6 prosumers, grouped into 3 clusters randomly. The 6 prosumers and their cluster assignment are constant across all 20 runs. The plot shows the evolution of percentage of coalitions with divisible excess as a function of sample size $m$.

Figure 5-11b shows a similar figure, except the terminating condition in step 4 is defined more strictly as $|\frac{|\varepsilon_m^{\text{pos}}|}{|\mathfrak{R}_m|} - \frac{|\varepsilon_{m-1}^{\text{pos}}|}{|\mathfrak{R}_{m-1}|}| \leq 0.005\frac{|\varepsilon_m^{\text{pos}}|}{|\mathfrak{R}_m|}$ and $|\frac{|\varepsilon_{m-1}^{\text{pos}}|}{|\mathfrak{R}_{m-1}|} - \frac{|\varepsilon_{m-2}^{\text{pos}}|}{|\mathfrak{R}_{m-2}|}| \leq 0.005\frac{|\varepsilon_{m-1}^{\text{pos}}|}{|\mathfrak{R}_{m-1}|}$.

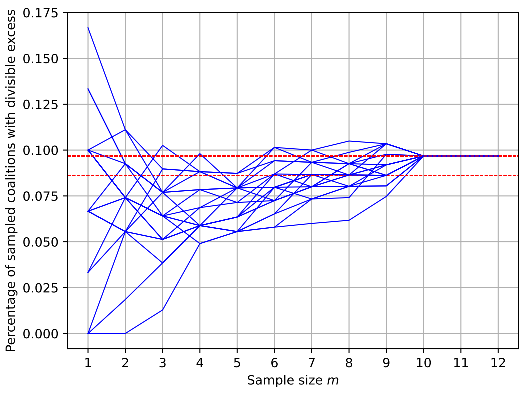
| Try | Dataset 1 | D.2 | D.3 | D.4 | D.5 | D.6 | D.7 | D.8 | D.9 | D.10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.0583 | 0 | 0 | 0.0364 | 0.0232 | 0.0086 | 0.0665 | 0 | 0.0811 | 0.0946 |
| 2 | 0.0628 | 0 | 0 | 0.0541 | 0.0086 | 0.0057 | 0.0637 | 0 | 0.0386 | 0.1181 |
| 3 | 0.0885 | 0 | 0 | 0.0408 | 0.0212 | 0.0115 | 0.0694 | 0 | 0.0685 | 0.0968 |
| 4 | 0.0667 | 0 | 0 | 0.0466 | 0.0212 | 0.0028 | 0.0787 | 0 | 0.0637 | 0.0947 |
| 5 | 0.0628 | 0 | 0 | 0.0444 | 0.0114 | 0.0028 | 0.0577 | 0 | 0.0714 | 0.1030 |

**Table 5-1:** This table shows the resulting fraction of sampled coalitions with divisible excesses for 10 separate datasets, where each dataset has its own prosumer demand profiles, distribution of PV and ES, and PV profile. Data is gathered 5 times across all 10 datasets to show that results coincide acceptably with multiple applications of stratified random sampling. All data points regard $(N, K) = (14, 5)$.

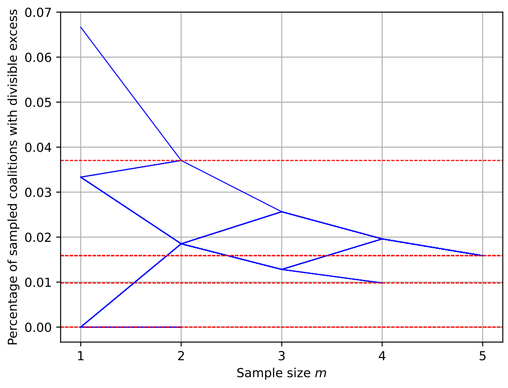
Additionally, smaller tests with $(N, K) = (14, 5)$ show that the nucleolus estimations have reasonable variations in results (one such test is shown in table 5-1). Furthermore, a later section will run nucleolus estimations and compare them to the full nucleoli for $N = 15$. Figure 8-12 shows that the stratified random sampling scheme is good; the estimated fractions of sampled subcoalitions with divisible excess tracks the true fraction of all subcoalitions with divisible excess quite well.



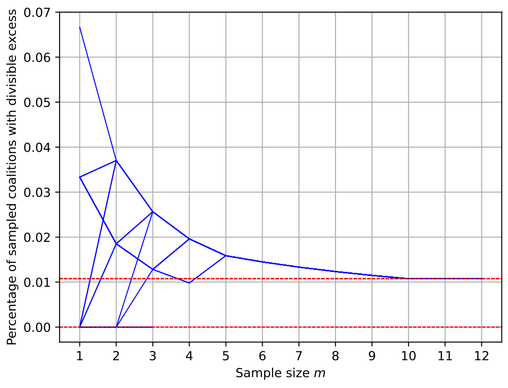
**(a)** This figure shows the evolution as a function of increasing $m$ until the terminating condition in step 4 is met.



**(b)** 20 iterations of stratified random sampling show better consensus with stricter terminating condition: $\left|\frac{|\varepsilon_m^{\text{pos}}|}{|\mathfrak{R}_m|} - \frac{|\varepsilon_{m-1}^{\text{pos}}|}{|\mathfrak{R}_{m-1}|}\right| \leq 0.005\frac{|\varepsilon_m^{\text{pos}}|}{|\mathfrak{R}_m|}$ and $\left|\frac{|\varepsilon_{m-1}^{\text{pos}}|}{|\mathfrak{R}_{m-1}|} - \frac{|\varepsilon_{m-2}^{\text{pos}}|}{|\mathfrak{R}_{m-2}|}\right| \leq 0.005\frac{|\varepsilon_m^{\text{pos}}|}{|\mathfrak{R}_m|}$.



**(c)** This figure shows 20 applications of stratified random sampling to another set of 6 prosumers grouped into 3 clusters, with terminating condition in step 4.



**(d)** 20 iterations of stratified random sampling for this second set of 6 prosumers also shows better consensus with stricter terminating conditions.

**Figure 5-11:** Each plot shows 20 applications of stratified random sampling to the nucleolus estimations of 6 prosumers, grouped into 3 clusters. Plots **a)** and **b)** compare results between the terminating condition used in Han *et al.* and a stricter terminating condition. Plots **c)** and **d)** compare terminating conditions for a second set of 6 prosumers, grouped into 3 clusters.

Chapter 6

# Nucleolus Estimation - Modified Clustering Case Study Results

## 6-1 Case Study 1 - Distance-adjusted Clustering

Briefly introduced in section 4-2-1, this section analyzes the effects of modifying clusters to account for the distances of prosumers to the primary prosumer on the nucleolus estimation quality.

Computational times mentioned from here on consider multi-core processing, which parallelizes the nucleolus estimations and nucleolus estimation evaluations for each of 10 iterations as shown in figure 6-1.

### 6-1-1 Choice of $(N, K)$

A simulated Peer-to-Peer (P2P) microgrid with $N = 30$ and $K = 5$ is used to experiment with the modification of clusters. This choice of $N$ and $K$ balances:

1) The quality of the nucleolus estimation with non-modified clustering. A reasonable percentage of sampled coalitions with divisible excesses is needed to properly evaluate if clustering modifications reduce, maintain or worsen the divisible excesses. Taking $(30, 5)$ makes these percentages positive in almost all cases.

2) The time needed to compute the percentage of coalitions with divisible excess. For larger $N$, the nucleolus estimation computation time becomes relatively insignificant in comparison to the computation time required for stratified random sampling. For $(30, 5)$, computing and evaluating the nucleolus estimation for all 10 iterations takes an average of $4333s$, of which the nucleolus estimation takes $503s$, and stratified random sampling takes $3830s$. This average is drawn all computations used to create all 10 Gaussian distributions of the non-modified divisible excess percentages in figure 6-2.

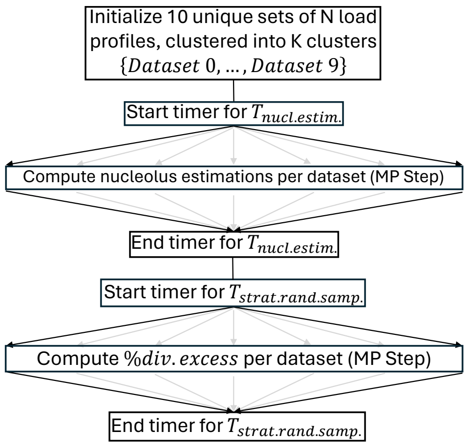

Initialize 10 unique sets of N load profiles, clustered into K clusters $\{Dataset\ 0, \ldots, Dataset\ 9\}$
Start timer for $T_{nucl.estim.}$
Compute nucleolus estimations per dataset (MP Step)
End timer for $T_{nucl.estim.}$
Start timer for $T_{strat.rand.samp.}$
Compute $\%div.excess$ per dataset (MP Step)
End timer for $T_{strat.rand.samp.}$

**Figure 6-1:** This figure shows how multiprocessing is used to parallelize the nucleolus estimation and nucleolus estimation evaluation (using stratified random sampling) to keep computational time manageable in the case studies.

3) To populate the feature space adequately so that minor modifications to the cluster borders result in reassignments of prosumers from one cluster to another.

In Han *et al.* [16] it was seen that the relations between nucleolus estimation performances of clustering methods were reasonably consistent across $(N, K)$ pairs (figure 3-5, 3-6), so to quantify the performance of additional clustering methods, a single $(N, K) = (30, 5)$ pair is considered for now. A later part of this thesis re-runs nucleolus estimations for $(N, K) = (15, 5)$ in order to analyze nucleolus estimations compared to the full (true) nucleoli. Here, some nucleolus estimation performances are worse despite a better ratio $\frac{N}{K}$, so better performance for improved ratios $\frac{N}{K}$ cannot be taken for granted.

Also, the focus on percentages of subcoalitions with divisible excesses, rather the magnitude of these divisible excesses is justified on the basis of figure 3-6, where the bottom plot shows the magnitude of excesses is proportional to their prevalence.

## 6-1-2 Nucleolus estimation using distance-modified K-means

Figure 6-2 shows the percentage of sampled coalitions with divisible excess for all 10 datasets, for two categories of modified k-means clusters. The results are compared to Gaussian distributions which approximate the non-modified k-means nucleolus estimation data. These Gaussians are established by repeating stratified random sampling 10 times , and using this sample set of resulting divisible excess percentages to visualize the variance in divisible excesses (considering Bessel's correction for the unbiased estimation of sample variance).

The equation for modifying k-means clusters was

$$\begin{aligned}\text{dist}_{mod}\left(\text{pros}_i, \text{cluster}_j\right) =& \text{dist}\left(\text{pros}_i, \text{cluster}_j\right) + \\ & m\left|\text{dist}\left(\text{prim.pros}, \text{cluster}_j\right) - \text{dist}\left(\text{pros}_i, \text{prim.pros}\right)\right|\end{aligned} \tag{6-1}$$

The value of $m$ is incrementally increased in 10 steps between a minimum value $m_{min}$ and a maximum value $m_{max}$, established in two different ways, leading to two categories of modified k-means clusters.

### Category 1: Setting dataset-specific bounds $m_{min}, m_{max}$

This category of results is gained by, for each dataset individually, looking at the average percent change in prosumer cluster reassignments across all $N$ primary prosumers. $m$ is increased in steps of 0.1. The lower bound $m_{min}$ is set to the lowest value of $m$ where the average percentage of reassigned prosumers is greater than 0.5%. The upper bound $m_{max}$ is set to the first value of $m$ that corresponds to a prosumers reassigned percentage within $20 \pm 1\%$.

With each dataset having its specific range $[m_{min}, m_{max}]$, this range is split into 10 steps to gradually alter the k-means clusters. For all $N$ primary prosumers per dataset, clusters not containing the primary prosumer are modified per equation 4-4 if and only if the primary prosumer is not alone in its cluster. Single-prosumer cluster are skipped entirely in the diversity-based pairing step because single-prosumer clusters are already grouped with other clusters in the cluster combinations step, and their payoff as a result is unique, compared to prosumers in larger clusters that need diversity-based pairing to receive unique payoffs. Modified clustering methods only regard the diversity-based pairing step.

### Category 2: Primary prosumer-specific bounds $m_{min}, m_{max}$, with filters $m_{max} \leq \{6, 3\}$

An improvement on the first category of results might be to modify clusters within bounds $m_{min}, m_{max}$ set for each primary prosumer individually rather than with the average across primary prosumers. A small minority of primary prosumers produce outliers for the bound $m_{max}$. In one case, the 20% reassignment condition matched to $m_{max} = 454.9$. Beyond this, some cases never reach the 20% reassignment condition, regardless of the value of $m_{max}$. This means that for such a particular primary prosumer, clusters have to be radically distorted to reach the 20% reassignment terminating condition, at which point the modified clusters are meaningless (the cluster only represents the distance to the primary prosumer, not the similarity of prosumers in the cluster itself; in figure 4-2 such clusters would create cluster rings around the primary prosumer). For this reason a filter is incorporated that only saves sub-coalitions of primary prosumers combined with modified clusters if the bound $m_{max}$ is under some value. With $m_{max}$ above this value, the primary prosumer is combined with non-modified clusters as in Han *et al.* [16]. Two tests are run with filters for $m_{max} \geq 6$ and $m_{max} \geq 3$.

### Results Analysis

Figure 6-2 shows the results. While some results suggest improved nucleolus estimations in comparison to the non-modified case (dataset 1 with category 2 modifications using filter $m_{max} \leq 3$, dataset 3 with all modifications besides category 2 modifications using $m_{max} \leq 3$, dataset 5 with modified k-means using dataset-specific bounds on $m$, and dataset 9 for all modified k-means), there is no clear factor which indicates which sets of prosumers might benefit from such modification and from which particular strategy, and for which value of $m$.

It is ambiguous which k-means modification method perform the best. Dataset 1 and 5 for example show opposite results, where modified results all perform differently. Nucleolus estimation improvements also depend on the the incremental step of $m$. Dataset 9 shows

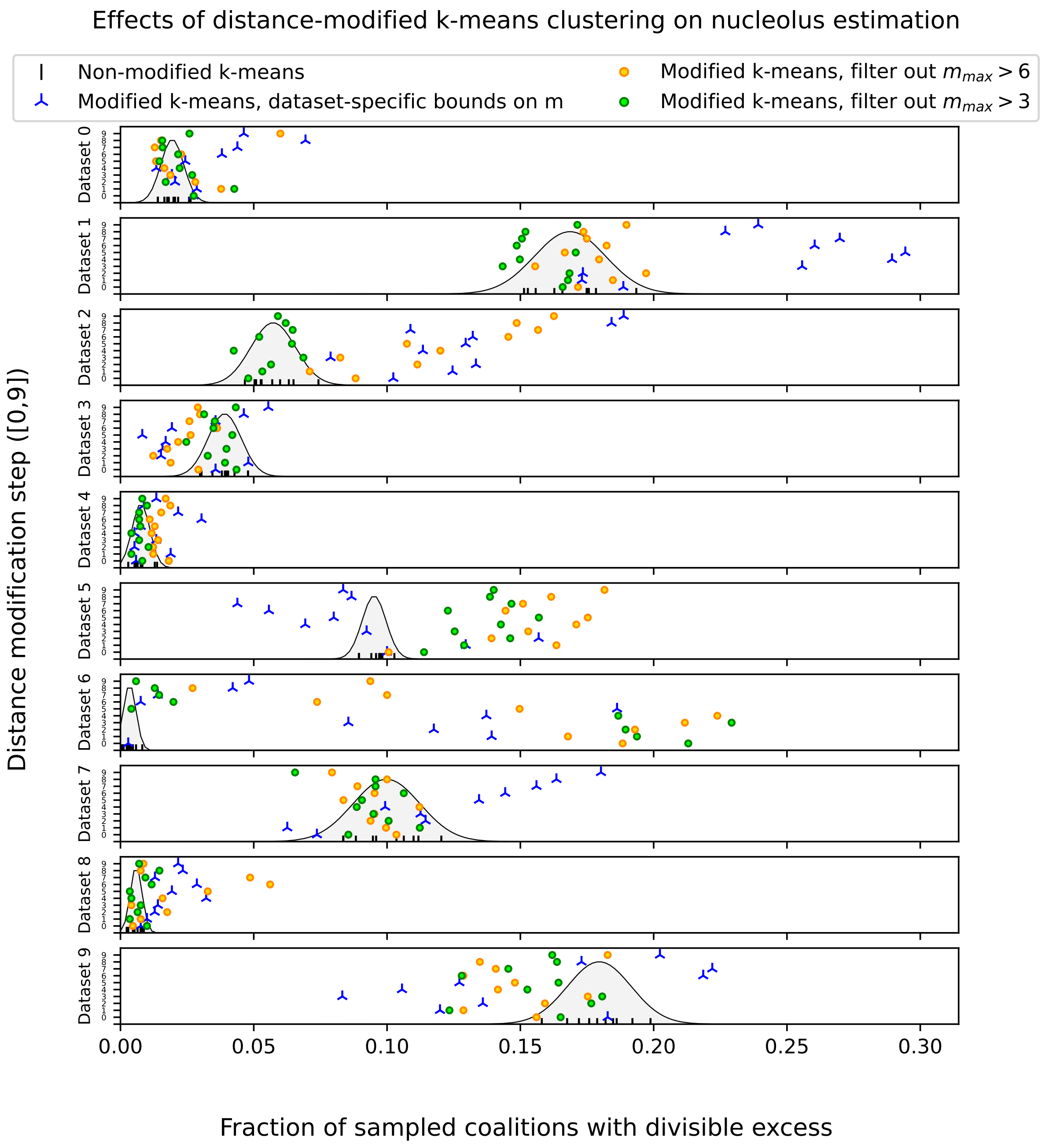


**Figure 6-2:** Distance-modified k-means results. The black Gaussians represent 10 independent tries of stratified random sampling to give a range of estimated fractions of sampled subcoalitions with divisible excess.

| Dataset | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| $m_{min}$ | 0.6 | 0.6 | 0.5 | 1.3 | 1.1 | 0.7 | 0.4 | 0.7 | 0.9 | 1 |
| $m_{max}$ | 3.4 | 7.8 | 5.6 | 7.6 | 7.3 | 4.6 | 1.9 | 6.8 | 4.2 | 8 |

**Table 6-1:** Values of $m_{min}, m_{max}$ for each dataset in modified k-means with dataset specific bounds on $m$

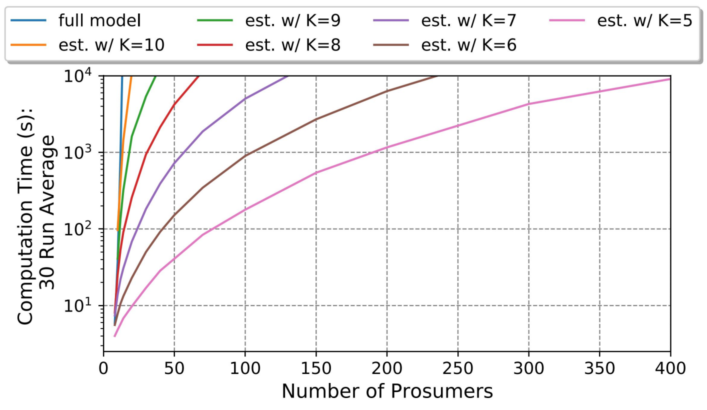


**Figure 6-3:** Average nucleolus estimation times across 30 runs from Han *et al.* [16].

modified k-means using dataset-specific bounds on $m$ to be significantly better than non-modified clustering for steps 1 to 5, while performing worse for steps 6 to 9. The opposite is for dataset 5, where modified k-means using dataset-specific bounds on $m$ performs better for steps 4 to 9, and worse for the lower steps compared to non-modified k-means.

**Effect of $m_{max}$ on nucleolus estimation performance (for Category 1 modifications):** Table 6-1 shows the values of $m_{min}, m_{max}$ for each dataset in modified k-means with dataset-specific bounds on $m$. While it was considered that iterations with a high value for $m_{max}$ might perform worse as the 20% reassignment condition needs a higher deformation of the Voronoi cells (at which point the cluster represents only the distance to the primary prosumer, and doesn't represent the similarity of prosumers in the cluster itself), no visible relation can be found between the results and the the values of $m_{min}, m_{max}$.

## Computation Times

Table 6-2 displays the computational times required to compute all data shown in figure 6-2. For clarity, computing the nucleolus once per (dataset, clustering method) combination would be enough. However, the nucleolus estimation is recomputed for each repetition of stratified random sampling, in order to gather computational times for nucleolus estimations. Computational times are measured as shown in figure 6-1.

Given that all computation times are specified for 10 datasets, within this multiprocessing framework it can be said that the nucleolus estimation for $(N, K) = (30, 5)$ is roughly $45s$, which is close to the individual nucleolus estimation times (for individual iterations) established by Han *et al.* [16]. In figure 6-3 this is shown to be roughly $20s$.

| K-Means Modification Method | Nucleolus Estim. Comp. Time [s] | Strat. Rand. Samp. Comp. Time [s] | Total [s] |
|---|---|---|---|
| Non-mod. k-means (avg comp. time for 10 datasets, across 10 S.R.S. repetitions) | 503 | 3830 | 43330 |
| Mod. k-means with dataset-specific bounds $m_{min}$, $m_{max}$ (avg. for 10 datasets, across 10 $m$ steps) | 461 | 3724 | 41850 |
| Mod. k-means with primary prosumer-specific bounds $m_{min}$, $m_{max}$ (avg. for 10 datasets, across 10 $m$ steps and 2 filters) | $\frac{446+427}{2}$ | $\frac{3061+3104}{2}$ | 70380 |
| Total Case Study Comp. Time [s] | | | 155560 |

**Table 6-2:** Distance-modified k-means computational times, for all data presented in figure 6-2

#### Conclusions

Results currently do not show any consistent way of improving the nucleolus estimation with distance-modified clustering. Future work can consider increasing the number of stratified random sampling data points to generate Gaussian probability distribution functions that represent the results. Comparing PDF's of modified and non-modified datasets allows for comparisons with higher statistical significance, using the Student's T-test.

## 6-2 Case Study 2 - Primary Prosumers as Cluster Centroids

Introduced in section 4-2-2, this section looks at the effects of equating one out of $K$ cluster centroids to the primary prosumer's feature profile in diversity-based pairing.

### 6-2-1 K-Means (Cooperative Energy Profile)

For this modification, the k-means algorithm has to be applied an additional $N$ times for each primary prosumer. The k-means algorithm is modified in the centroids update step. Each centroid $k$ is redefined as usual besides cluster 0, which is defined to be the primary prosumer's cooperative energy profile. There are some exceptions to the application of modified clustering in diversity-based pairing.

#### Exceptions

Table 6-3 shows when modified k-means or non-modified k-means is applied. There are two exceptions to the application of modified clustering.

| Primary prosumer in cluster | | |
|---|---|---|
| **Non-mod.** | **Mod** | **Clustering approach in diversity-based pairing** |
| alone | alone | No diversity-based pairing (Exception 1) |
| alone | not alone | Modified k-means clustering |
| not alone | alone | Non-modified k-means clustering (Exception 2) |
| not alone | not alone | Modified k-means clustering |

**Table 6-3:** This table shows what clustering approach is used, depending on if the primary prosumer is singularly assigned to its cluster in modified and non-modified k-means clustering.

| Dataset | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Fraction of singular primary prosumers | $\frac{17}{30}$ | $\frac{14}{29}$ | $\frac{16}{30}$ | $\frac{15}{29}$ | $\frac{16}{29}$ | $\frac{15}{30}$ | $\frac{19}{30}$ | $\frac{18}{29}$ | $\frac{17}{30}$ | $\frac{14}{29}$ |

**Table 6-4:** This table shows how many primary prosumers are singularly allocated to a cluster in the modified k-means scheme, given that they are allocated to clusters with other prosumers in the non-modified scheme.

**Exception 1**
The purpose of diversity-based pairing was explained by Han *et al.* [16] to be a method that allows for differentiation between prosumer payoffs. The primary prosumer is paired with all possible combinations of clusters that exclude it. This means that if the primary prosumer is alone in its non-modified and modified cluster assignments, the diversity-based pairing step for that particular primary prosumer is redundant as the primary prosumer has already been paired to other clusters in the cluster combinations step.

**Exception 2**
Given that roughly half of the primary prosumers end up alone in their clusters using modified k-means (table 6-4), another exception is to not use the primary prosumer-centric k-means clusters, and instead use the non-modified k-means clusters for these primary prosumers. Any clustering result that has the primary prosumer be alone in its cluster is most likely to be suboptimal, especially considering that non-modified clustering assignments have very few prosumers that are alone in their clusters. Applying diversity-based pairing with the non-modified clusters for these primary prosumers effectively reduces sensitivity to outliers, allowing for efficient usage of the available clusters.

**Results**

Figure 6-4 shows the results of a series of experiments.

**1. Non-modified k-means**
The black Gaussian distributions summarize regular, non-modified k-means, with 10 repetitions of nucleolus estimation and stratified random sampling (with one common k-means clustering assignment per dataset) to account for deviations in stratified random sampling. The recalculation of the nucleolus payoffs could be skipped but are repeated alongside repetitions of stratified random sampling to get a good overview of the computational times.

**2. Modified k-means**

The red Gaussians show the divisible excess results for modified k-means. It can be seen that the modified k-means at no point notably outperforms non-modified k-means.

**3. Modified k-means (step-wise k-means cost filter)**

This experiment is conducted to see if limiting the degradation of k-means quality (setting a limit on the cost of primary prosumer-centric k-means results) improves results. The modified k-means costs for primary prosumers can vary from the non-modified k-means cost to varying degrees (some primary prosumers even have modified k-means assignments with better costs compared to the non-modified assignment). Each dataset has its own range of k-means cost limits. The maximum cost for each dataset is the maximum cost found across all modified k-means assignments. The minimum cost limit is set to be the dataset's non-modified k-means cost.

If a primary prosumer has a k-means cost above the limit, the primary prosumer is subject to diversity-based pairing using the non-modified k-means cluster assignment.

Results show that, besides a few outliers, the k-means cost filter generally improves the results as the k-means cost limit reaches the minimum value (cost filter step 0), suggesting that this modified k-means method worsens nucleolus estimation performance by increasing the k-means cost by forcing each primary prosumer to be its own cluster centroid.

#### General Analysis

While different k-means modification methods give different results, no experiment produces results that notably outperform the non-modified k-means case. Datasets 2 and 9 show slightly better results, but this is more a consequence of the additional variance in modified k-means divisible excesses caused by the additional primary prosumer clustering assignments.

#### Computational Time

Table 6-5 shows the computation times of the nucleolus estimation and nucleolus estimation evaluation (stratified random sampling) across all 10 datasets (considering multiprocessing as shown in figure 6-1). Using $(N, K) = (30, 5)$, the last two experiments roughly double the time necessary for the nucleolus estimation, required for the $N$ additional k-means applications for each primary prosumer.

## 6-2-2 GMM (Cooperative Energy Profile)

Much like the modified k-means, the modified GMM algorithm is slightly tweaked to have one cluster centroid set to be equal to the primary prosumer feature profile, as described in section 4-2-2. The rules in table 6-3 are also applied here.

#### Errors in nucleolus lexicographical minimization (`scipy.optimize.linprog`)

While errors in the nucleolus lexicographical minimization come up rarely, errors are easily avoided by finding a new cluster assignment, or by a slight modification of any parameter.

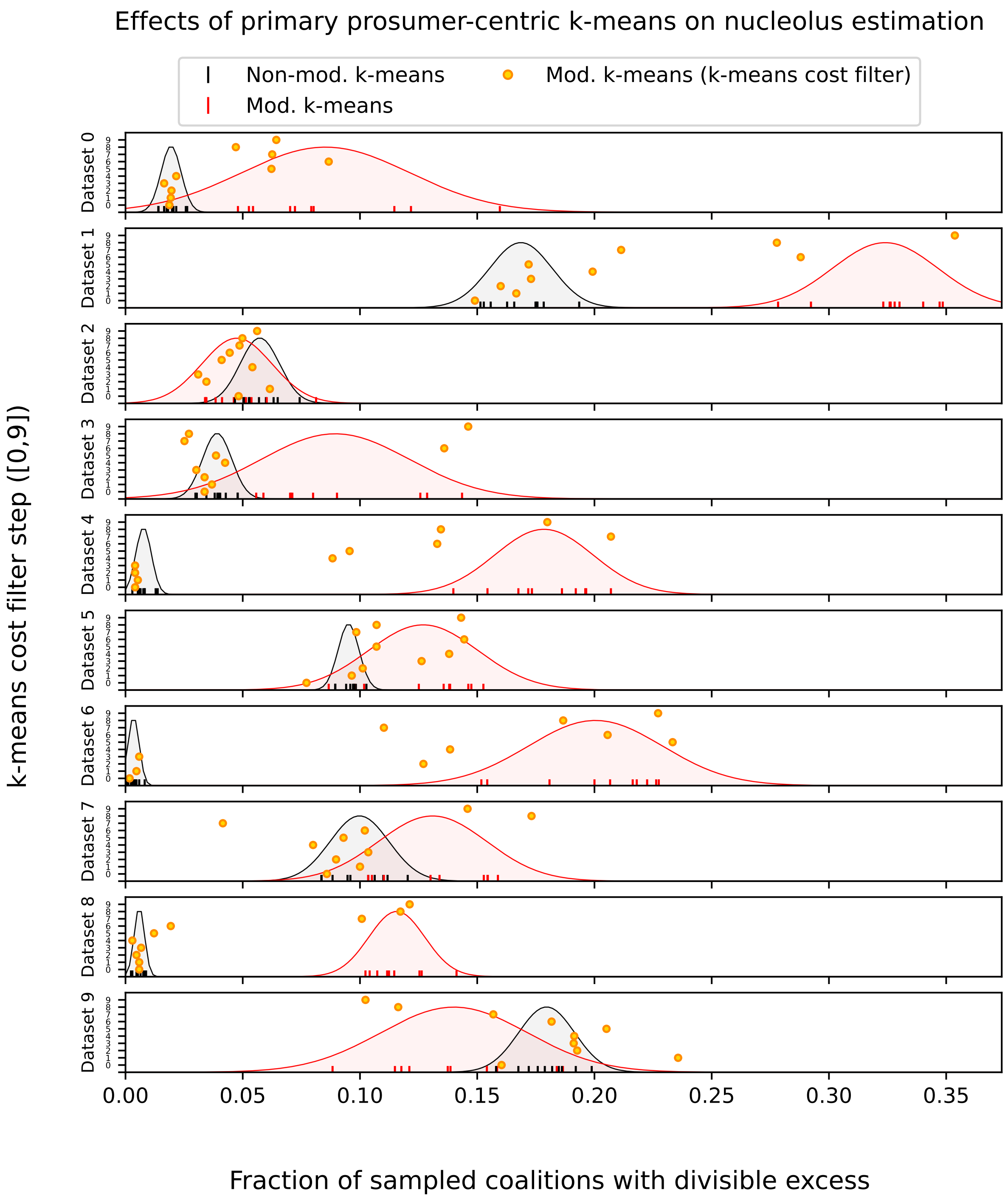


**Figure 6-4:** This figure shows the fraction of sampled coalitions with divisible excesses using non-modified k-means, modified k-means and modified k-means with a cost function filter.

| K-Means Modification Method | Nucleolus Estim. Comp. Time [s] | Strat. Rand. Samp. Comp. Time [s] | Total [s] |
|---|---|---|---|
| 1. Non-mod. k-means (avg comp. time for 10 datasets, across 10 S.R.S. repetitions) | 503 | 3830 | 43330 |
| 2. Mod. k-means: avg. for 10 datasets, across 10 repetitions | 953 | 3647 | 46000 |
| 3. Mod. k-means: avg. for 10 datasets, across 10 k-means cost steps) | 1040 | 3814 | 48540 |
| Total Case Study Comp. Time [s] | | | 137870 |

**Table 6-5:** This table specifies computation times for the nucleolus estimation and stratified random sampling across all 10 datasets, averaged across 10 repetitions.

In the results presented in this document so far, the only nucleolus estimation error was encountered in section 6-1-2, where the nucleolus estimation could not be calculated for one specific dataset for the value of $m = 5$. Modifying this value slightly to $m = 4.8$ consequently avoided the same minimization problem.

For this modified GMM scheme however, a nucleolus estimation is much more likely to terminate in an error than not. The code is modified to repeat nucleolus estimations across all 10 datasets repeatedly (relying on changes in clustering assignments for primary prosumers in the diversity-based pairing step) until a nucleolus estimation is returned without error. The set of sub-coalitions which return a valid nucleolus estimate are saved and later used to find the percentage of sampled coalitions with divisible excess.

The errors presented here are most likely an extension of the issue described in 5-6-3. The same errors are generated with either regular GMM or the improved log-scale GMM described in section 4-3-4.

#### Results

Figure 6-5 shows the nucleolus estimations found. The results show worse nucleolus estimation properties compared to the non-modified GMM case. These results are calculated with older data using regular GMM, not the improved log-scale GMM described in section 4-3-4, as no change was found in nucleolus estimation errors, and the performance of modified GMM was equally worse using log-scale GMM.

### 6-2-3 GMM (Marginal Contribution Profile)

Using the marginal contribution profile rather than the cooperative energy profile, the results of modified and non-modified diversity-based pairing are shown in figure 6-6. The modified cluster allocations cause the same errors in nucleolus estimation as modified GMM using the cooperative energy profile, where the minimization fails in a majority of cases. With enough repetitions, some functioning cluster allocations are found for certain datasets (0,2,3,5,6,8,9), where performance is better for all non-modified cases. Only the results for dataset 8 are inconclusive as all divisible excesses (both for modified and non-modified clustering) are 0.

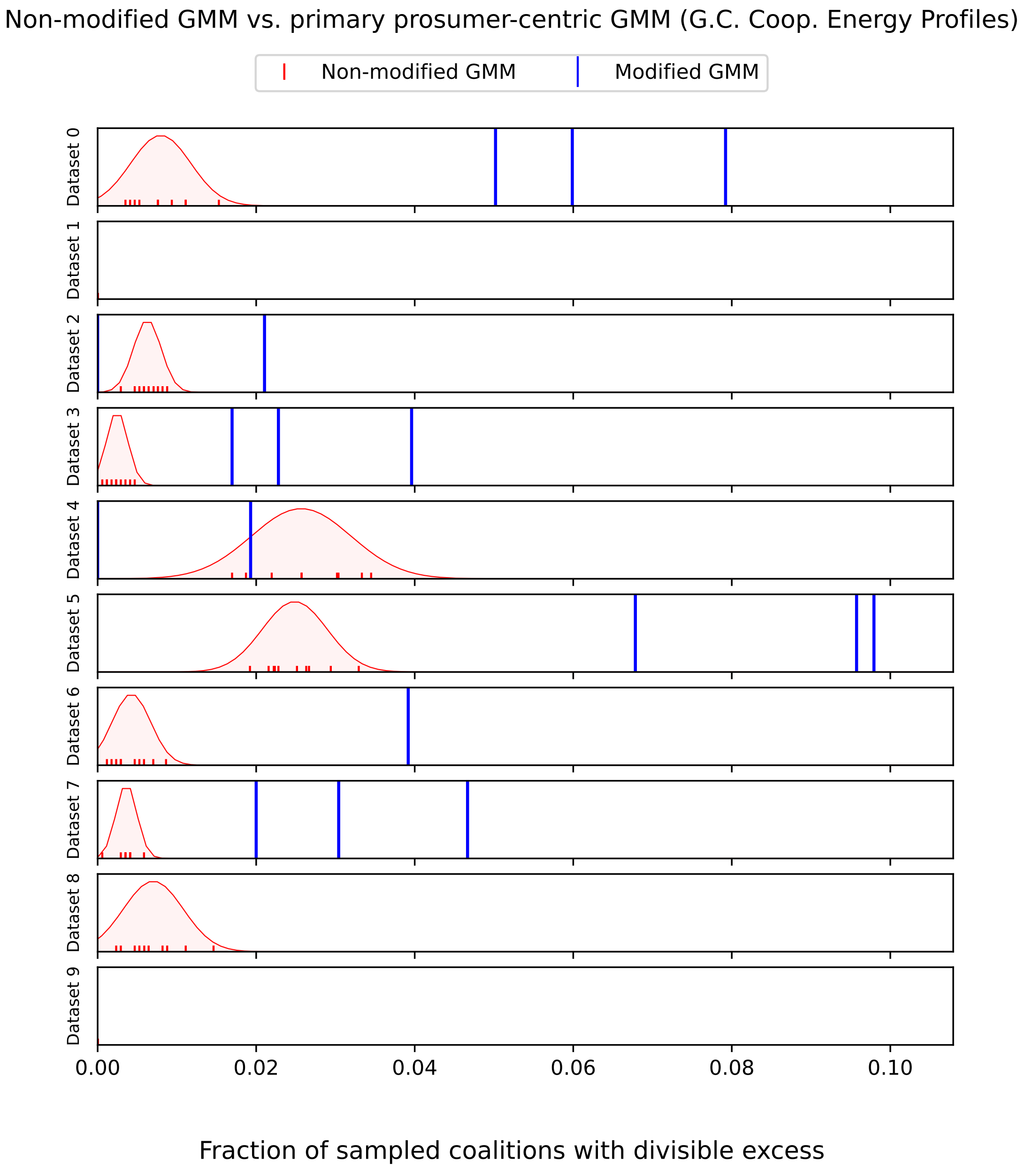


**Figure 6-5:** The results of modified GMM with primary prosumer-centric GMM being used in diversity-based pairing, using the grand coalition cooperative energy profiles.

As in the previous cases, the rules in table 6-3 are applied here also, however checks if the primary prosumer is alone in its modified cluster have not been necessary so far; across all 10 datasets (30 prosumers each) only one cluster was found where the primary prosumer was the sole prosumer in its cluster.

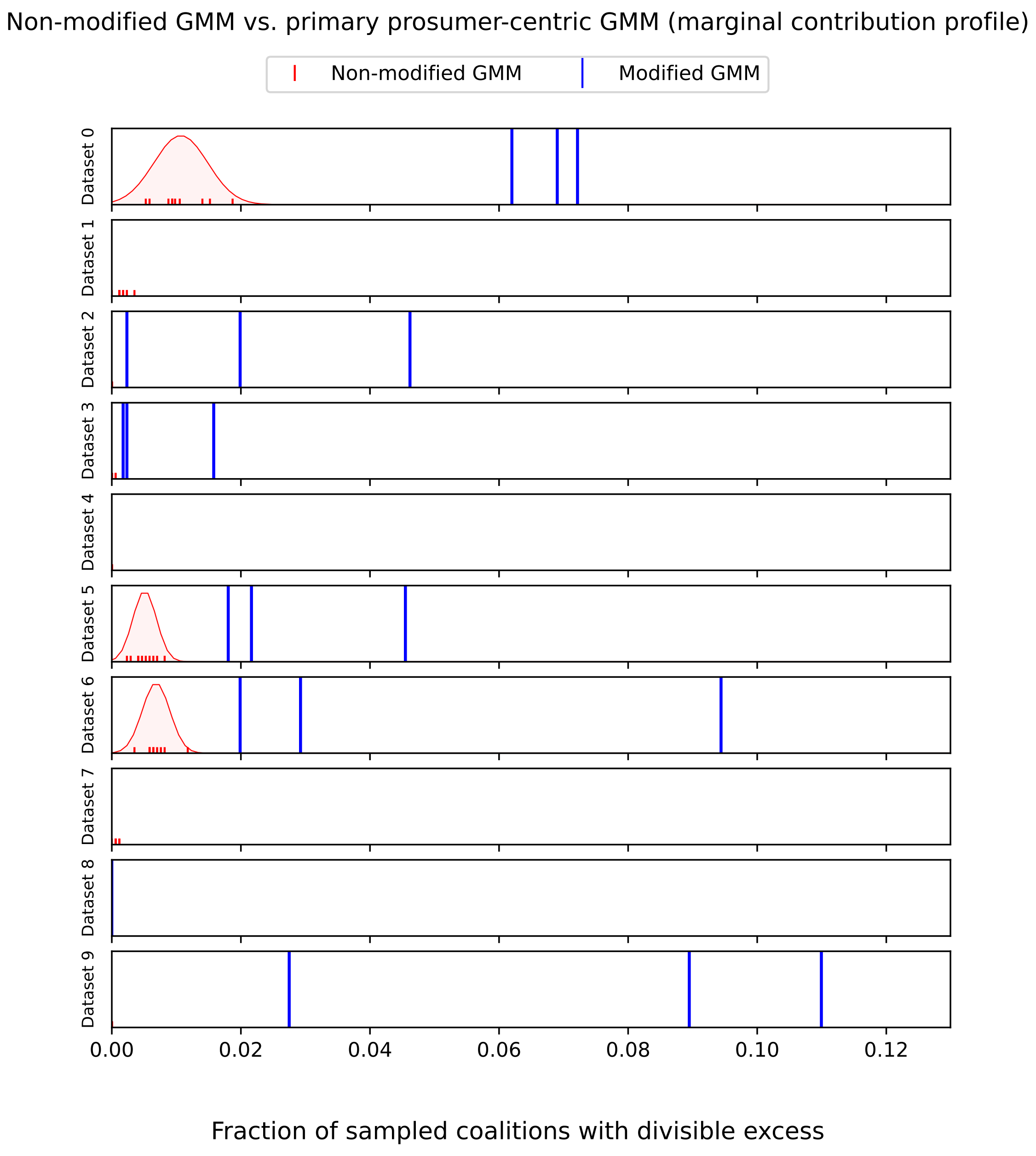


**Figure 6-6:** The results of modified GMM compared to non-modified GMM, using the marginal contribution profile.

Chapter 7

# Nucleolus Estimation - Alternative Clustering Case Study Results

Case studies analyzing additional clustering methods will make use of four feature profiles:

- MCP: The marginal contribution profile,
- CEP: the cooperative energy profile,
- MCPmin: the marginal contribution profile with minimized battery (dis)charge limits $\underline{b}_i, \overline{b}_i$,
- CEPmin: the cooperative energy profile with minimized battery (dis)charge limits $\underline{b}_i, \overline{b}_i$.

CEPmin and MCPmin use ES operations from the modified LP in equation 5-18 . All feature profiles are shown in Appendix A, including the normalized feature profiles used in Case Study 5 (section 7-5).

## 7-1 Case Study 1: Performance Comparisons between K-medians and K-means

The fractions of sampled coalitions with divisible excesses across all datasets is shown in figure 7-1 below. Each Gaussian consists of 10 individual applications of stratified random sampling to one nucleolus estimation. No Gaussians are shown for nucleolus estimations with any 0% divisible excess data point in the 10 stratified sampling results. For both k-medians and k-means, the best clustering assignment is chosen from 1000 separate k-means and k-medians applications. Centers are initialized to be prosumer instances, with a slight offset applied for k-medians to avoid division by zero in the Weiszfeld algorithm.

A few exceptions are implemented for the k-medians algorithm. Much like k-means, k-medians can occasionally give empty clusters, this must be detected. Secondly, a `while` loop iterating

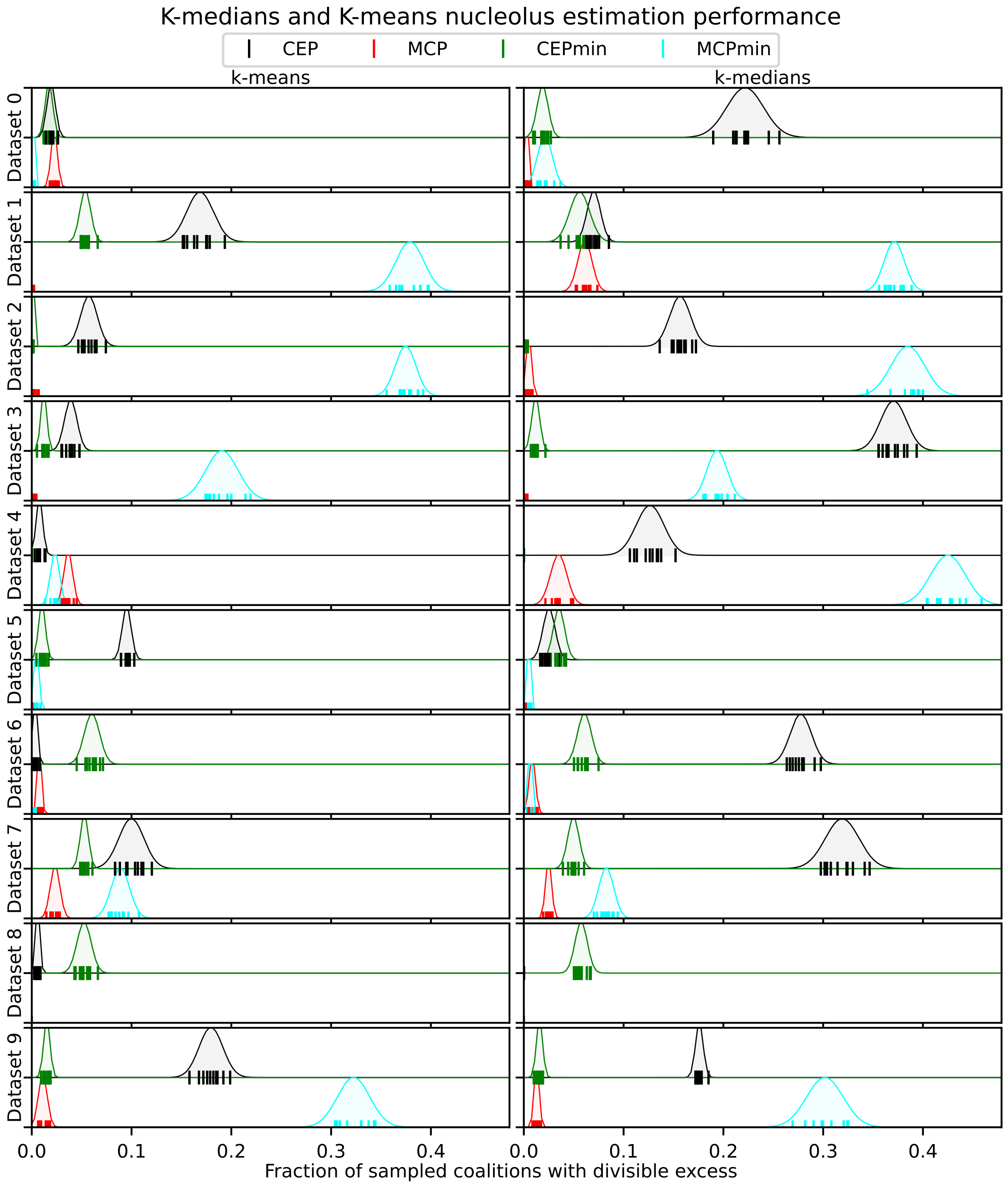


**Figure 7-1:** This figure shows nucleolus estimation results grouped by clustering method and datasets.

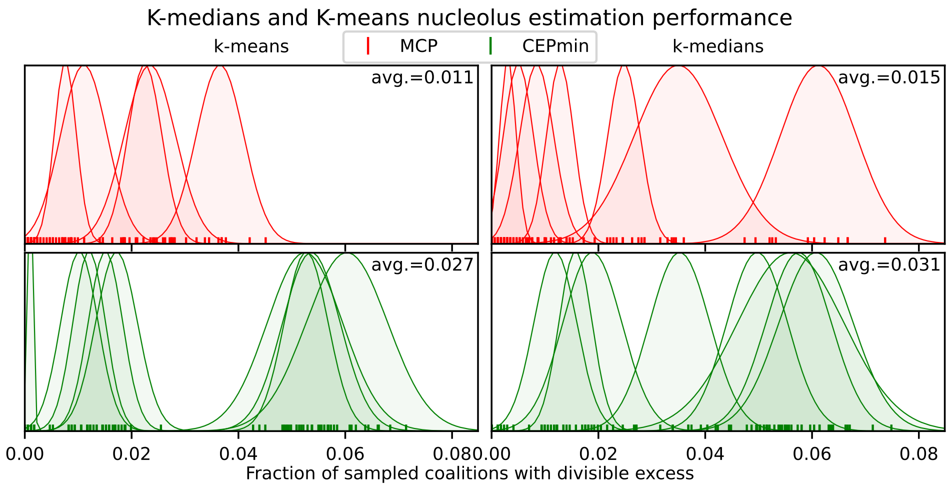


**Figure 7-2:** MCP and CEPmin results are drawn from figure 7-1 and individual dataset results are combined into single plots for better overview of the best general clustering method. Each subplot annotates the mean of each dataset's mean fraction of sampled coalitions with divisible excess, across all datasets. K-means using MCP performs the best.

between new centroids and resulting prosumer allocations occasionally has a centroid bounce around a minimum by a small but non-decreasing distance, never terminating. Any successful calculation of k-medians clusters has the medians settle in well under 50 cycles, so 50 is the chosen threshold. Another solution would be to relax the terminating condition. Specifically, increasing `eps` from 0.001 to 0.003 in `np.all(abs(new _centroids - centroids) < eps)` would solve all cases of this error for k-medians using MCP. Thirdly, within the Weiszfeld iterative process, it rarely happens that a geometric median moves to coincide with a prosumer instance resulting in `NaN` values. In such cases the program resumes with the next k-medians iteration.

### 7-1-1 ES optimization effects on nucleolus estimations

Figure 7-1 shows that regardless of the choice of k-means or k-medians, CEPmin outperforms CEP. Exceptions are k-means datasets 6,8 and k-medians dataset 8. The opposite is true for MCP. MCPmin can occasionally perform marginally better than MCP, but MCP outperforms MCPmin across a large majority of cases, regardless of the choice of k-means or k-medians.

This consistent difference in performances shows that the choice of ES strategies can significantly affect nucleolus estimation performance.

### 7-1-2 Best Results

Figure 7-1 shows that CEP and MCPmin have the worst performance. Figure 7-2 shows the results from figure 7-1 with CEP and MCPmin results removed, and having all results grouped by feature profile and clustering method. Each subplot has the average divisible excess across all 10 datasets, including the results not represented by Gaussians if any dataset has a 0% divisible excess data point.

Figure 7-2 shows MCP performing marginally better than CEPmin for both k-means and k-medians. K-medians offers no performance advantage over k-means.

Variances of cluster sizes per dataset are shown in figure 7-4. A larger cluster variance indicates less even distributions of prosumers amongst clusters. The two worst performing feature profiles (CEP and MCPmin) have the largest cluster variances. Cluster size variances are smaller for CEPmin than MCP, while nucleolus estimation performance was better for MCP than CEPmin. All this suggests that cluster size variance can to some extent be basis for predictions regarding a clustering method's nucleolus estimation performance.

It is interesting to see that k-medians, despite the median being a center less sensitive to outliers, actually returns larger cluster size variances compared to k-means. The robustness towards outliers allows certain clusters be large, considering profiles that would otherwise be separate clusters under k-means as outliers belonging to one cluster, marginalizing only the most non-conforming feature profiles into their own clusters (figure 7-3). This results in increased cluster size variances.

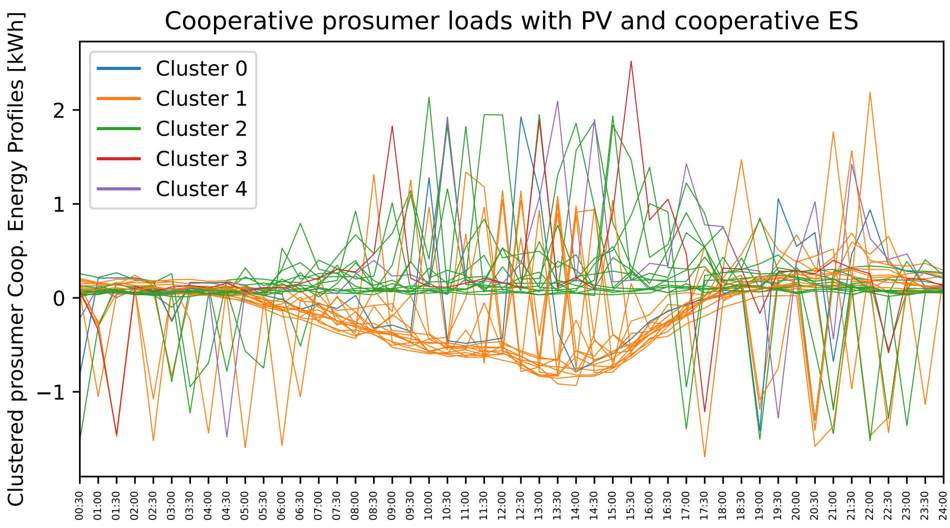


**Figure 7-3:** The best k-medians clustering result out of 1000 iterations using CEP. This figure shows two large clusters forming amongst the two groups of separated data. Clusters seem to form based on ownership of PV, where ES profiles are close enough to profiles without ES that combining both in larger clusters still minimizes the k-medians objective function.

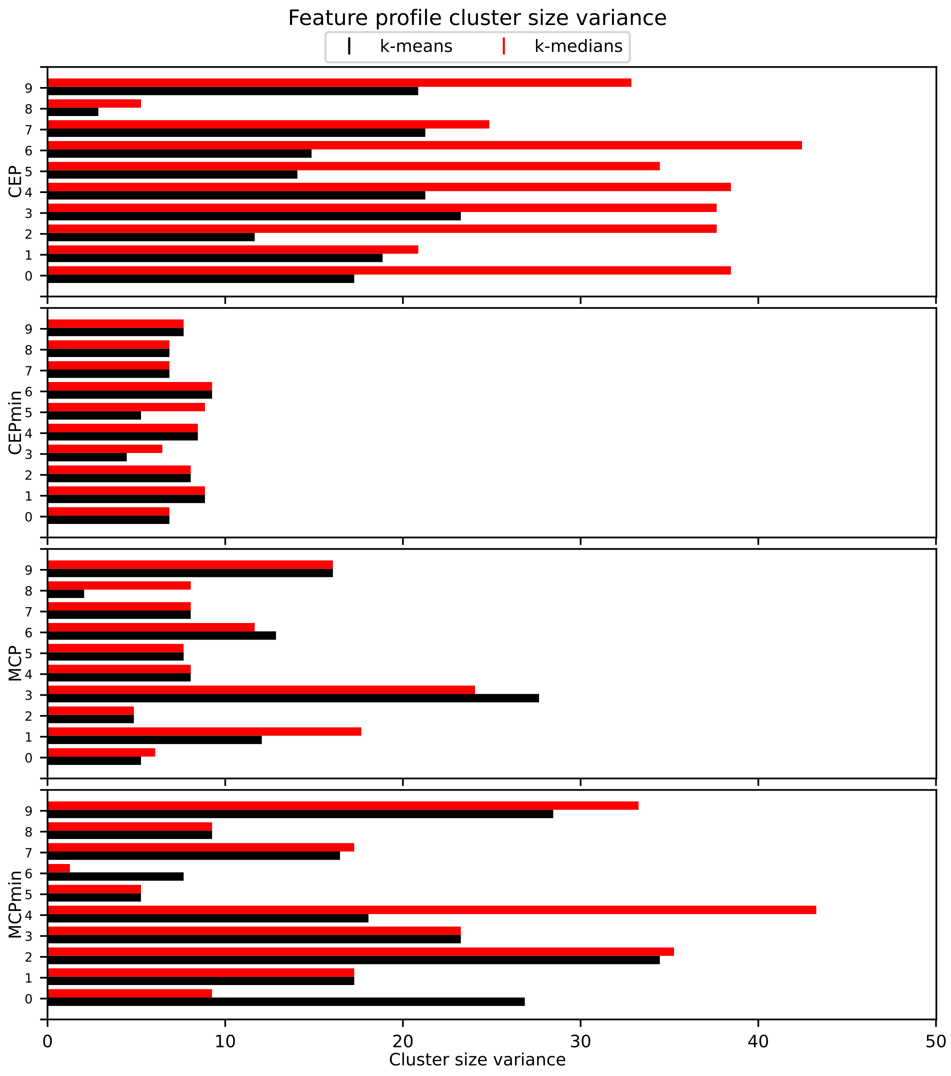


**Figure 7-4:** The figure shows the variances in cluster sizes per dataset, sorted by feature profile. The feature profiles with the worst nucleolus estimations (CEP and MCPmin) have the largest variances. The best performing feature profiles (CEPmin and MCP) have considerably lower variances. Conversely, the best performing feature profile (MCP) has larger variances than CEPmin, the second-best performing feature profile.

## 7-2 Case Study 2: Fuzzy C-Means & Fuzzy C-Medians

### 7-2-1 FCM and FCMed Implementation

Figure 7-5 shows all nucleolus estimation results for fuzzy c-means and fuzzy c-medians. An immediate conclusion that can be drawn is that fuzzy c-means using the CEP is too unreliable. For many datasets, FCM does not return valid clustering assignments to prosumers as one (sometimes more) clusters are not assigned prosumers at all. FCMed is much less likely to find clustering assignments where any cluster is empty. FCM and FCMed are both iterated 100 times, where runs with faulty outcomes are excluded from nucleolus estimation, and are deduced from the 100 iteration limit.

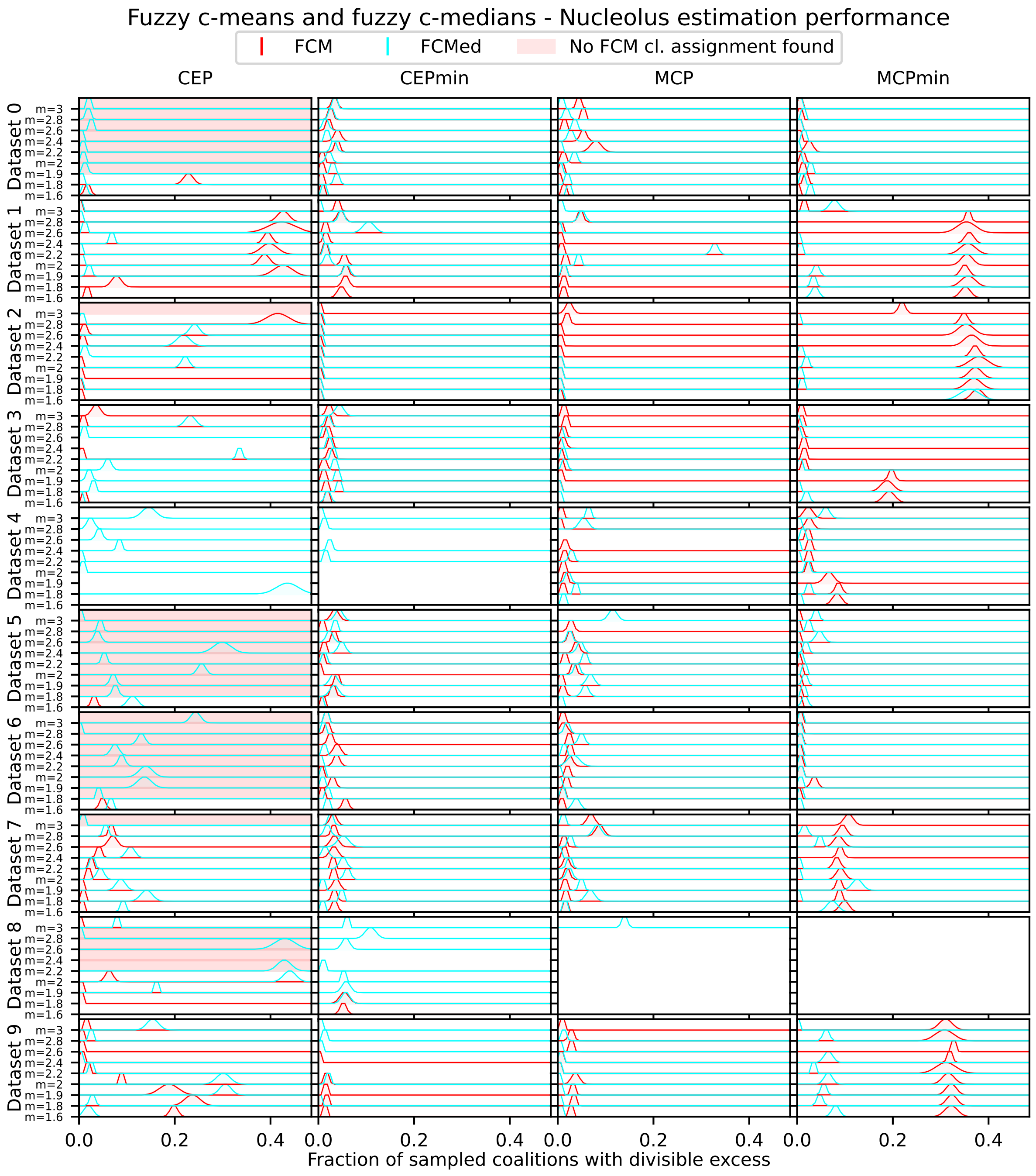


**Figure 7-5:** This figure shows the fractions of coalitions with divisible excesses for different fuzzifier parameters $m$ across 10 datasets. Here, each Gaussian consists of 5 stratified sampling data points.

For FCMed, the loop alternating between memberships and centroids can occasionally have a center which never quite settles, but continuously bounces around its minimum by a small but non-decreasing distance. This was seen in k-medians clustering as well (covered in previous section 7-1). A limit to the number of evolution cycles solves this problem. Choosing less stringent terminating conditions is another solution.

### 7-2-2 Results

Figure 7-5 shows that neither FCM and FCMed with CEP perform well across any datasets for any value of $m$. FCM-MCPmin also shows very large portions of sampled coalitions with divisible excesses, which rules it out as a good clustering algorithm.

Figure 7-6 aggregates the better performing results across all 10 datasets to help look for a feature profile and $m$ that performs well across all 10 datasets. Taking 1% average divisible excess as a cutoff criterium, FCMed-CEPmin with $m = 1.6$, FCM-MCP with $m = 1.6$, FCMed-MCPmin with $m = 2.2$ perform very well.

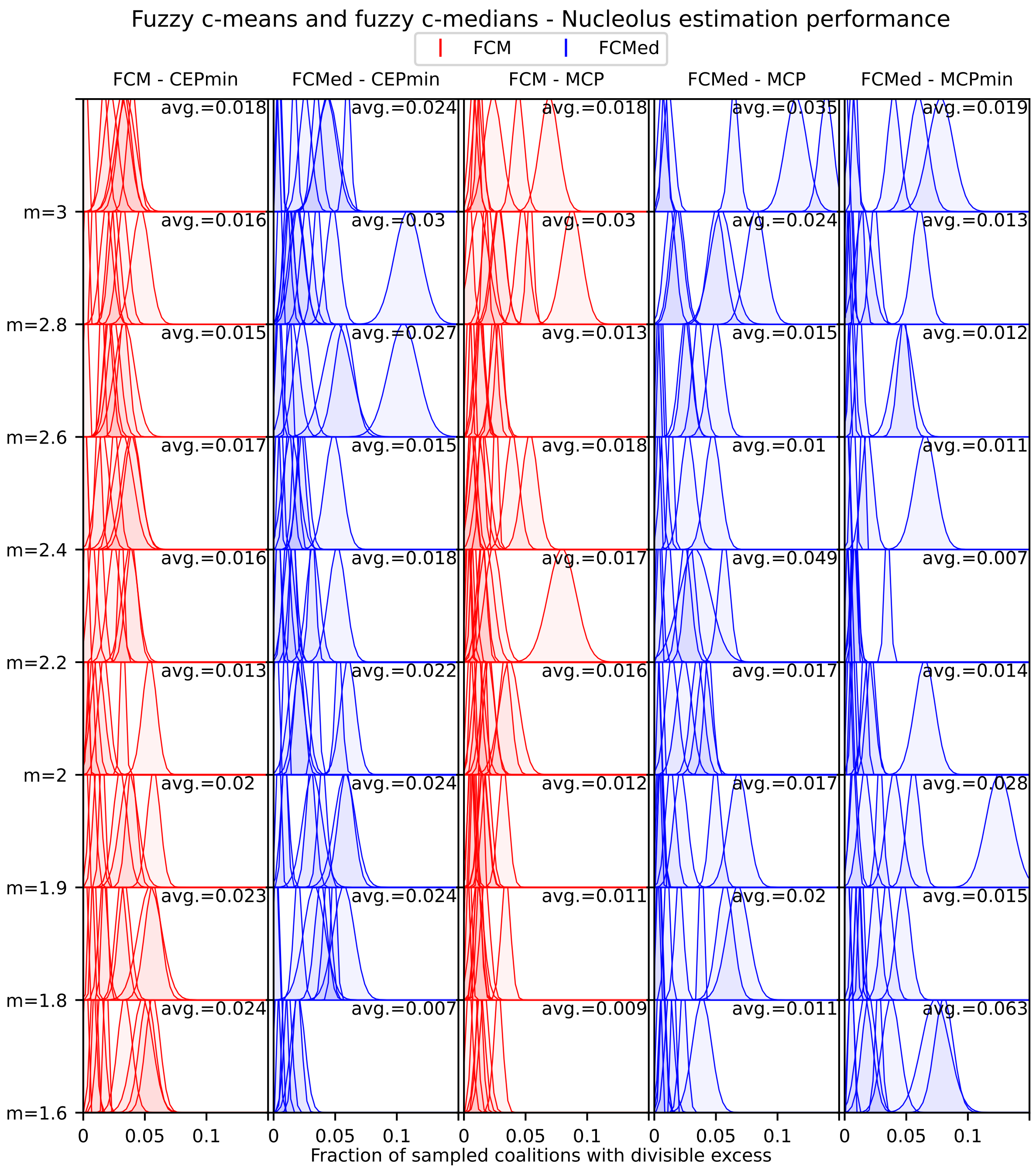


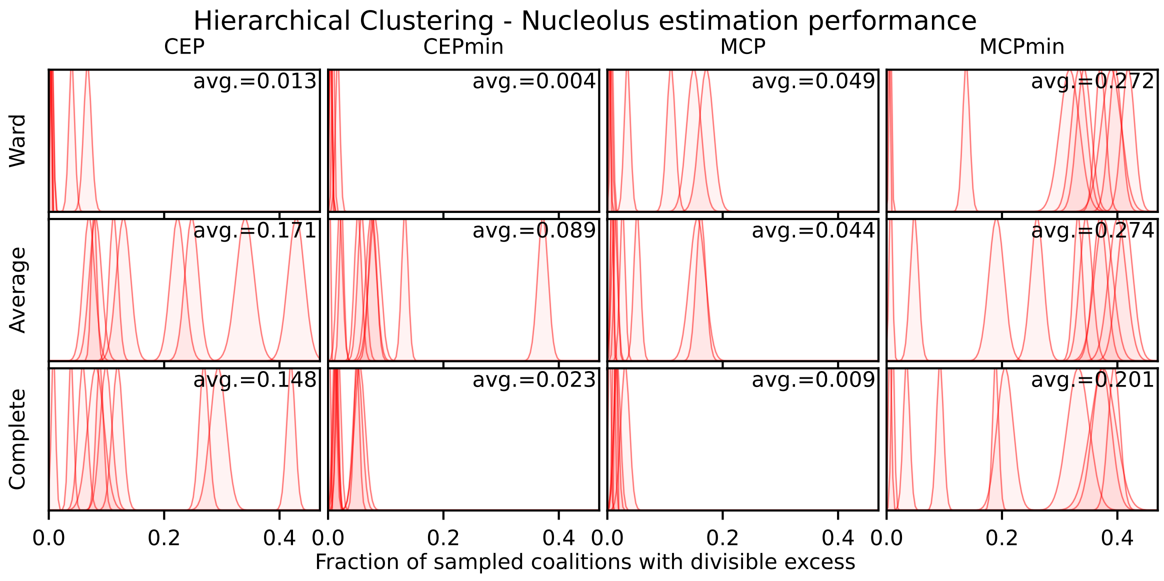
**Figure 7-6:** This figure shows the results of figure 7-5, condensed to show average results across 10 datasets. While Gaussians are only displayed if the minimum divisible excesses is greater than 0, the displayed averages also consider 0% divisible excess results.

## 7-3 Case Study 3: Comparing Ward, Average and Complete Linkage Methods

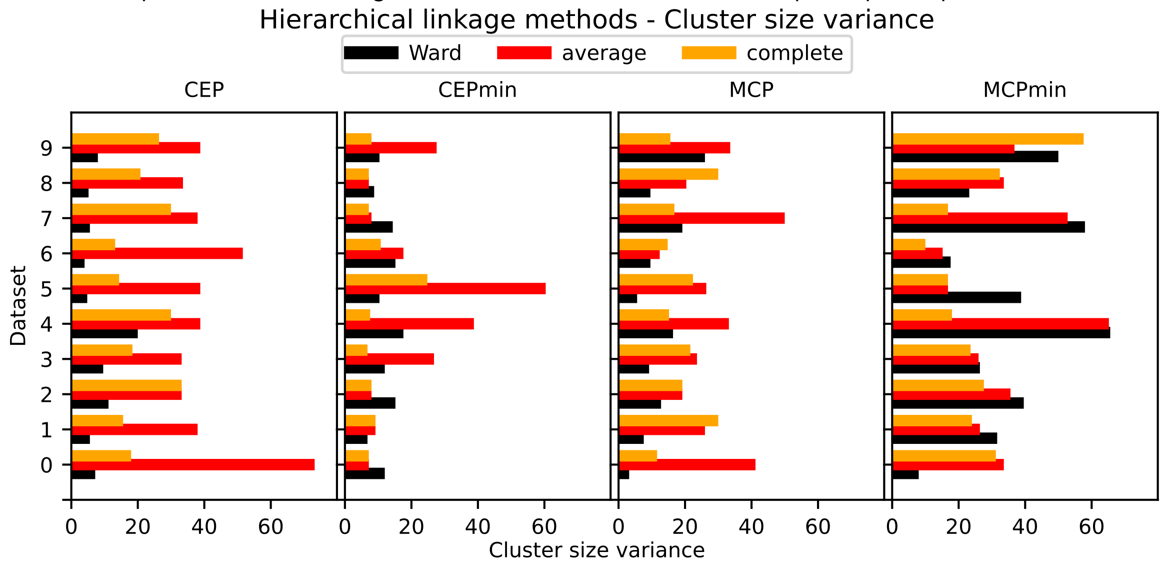
Figure 7-7a shows hierarchical clustering performing very well with Ward-CEPmin and Complete-MCP.

Figure 7-7b shows that a broad trend between clustering performance and cluster size variance exists. However, any direct correlation can be disproven by looking at the following examples:

- MCP: The Ward linkage has marginally smaller cluster size variances compared to the complete linkage, but the complete linkage still outperforms Ward in terms of divisible excess.
- CEPmin: Ward has marginally larger cluster size variances compared to complete linkage but still performs better.
- CEPmin/MCP with Complete linkage: The complete linkage gives better divisible excess results with MCP than with CEPmin, while CEPmin has consistently lower cluster size variances.



**(a)** Results of hierarchical clustering for the Ward, average and complete linkages. Each Gaussian uses 10 data points, and each average considers all 100 divisible excess data points per subplot.



**(b)** Cluster size variances across all datasets and linkages.

**Figure 7-7:** Hierarchical clustering results.

## 7-4 Case Study 4: GMM-PCA

### 7-4-1 PCA

#### Principal Component Analysis, Eigendecomposition and Singular Value Decomposition

In section 4-3-4 it was explained that principal component analysis (PCA) orders a mean-centered data matrix $X$'s principal components (eigenvectors of $X^T X$ if considering the data matrix form in 7-1) based on the variance along the principal components (the corresponding eigenvalues).

For consistency, notation will consider a data matrix $X$ to have rows of individual prosumer feature profiles (equation 7-1). Also, as there is no need for any statistical information of the covariance matrix besides the ordering of the eigenvalues/singular values and corresponding eigenvectors, any mention of $X^T X$ will omit the fraction $\frac{1}{m}$ in the standard covariance equation $\frac{1}{m} X^T X$.

$$X = \begin{bmatrix} — & x_1 & — \\ — & x_2 & — \\ & \vdots & \\ — & x_n & — \end{bmatrix} \in \mathbb{R}^{n \times 48} \tag{7-1}$$

The implementation of PCA here uses the singular value decomposition (SVD). The SVD of a mean-corrected data matrix $B = X - \overline{X}$ (where $\overline{X}$ is a matrix containing $n$ identical rows, each containing the average row of matrix $X$) also produces the eigenvectors $V$, and the square roots of the eigenvalues called singular values. Ordering principal components by the magnitude of the corresponding eigenvalues is identical to ordering principal components by their singular values. Concretely, if the SVD returns $B = U\Sigma V^T$, then

$$\begin{aligned} X^T X &= V\Sigma^T U^T U \Sigma V^T \\ &= V\Sigma^T \Sigma V^T \end{aligned} \tag{7-2}$$

Multiplying both sides with $V$

$$\begin{aligned} (X^T X)V &= V\Sigma^T \Sigma V^T V \\ &= V\Sigma^T \Sigma \end{aligned} \tag{7-3}$$

This form above corresponds to the eigendecomposition of $X^T X$ ($X^T X V = V\Lambda$), showing that eigenvalues $\lambda$ are the squares of singular values in $\Sigma$, and that $V$ is a matrix of eigenvector columns.

With a given data matrix $X$ of size $n \times m$, the PCA procedure applies singular value decomposition to the mean-corrected data $B = X - \overline{X},$ . The singular value decomposition produces three matrices, $U\Sigma V^T$, where $U$ is an orthogonal $n \times n$ matrix, $V^T$ is an orthogonal $m \times m$ matrix, and $\Sigma$ is an $n \times m$ matrix with diagonal values being the singular values, ordered so that $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$ . The original, full-dimensioned data $B$ can be projected onto the new reference frame spanned by the eigenvectors (columns of $V$) with $T = BV$, with $T$ being the new data coordinates with respect to the principal components. Dimensionality reduction can be done by reducing the number of columns in $V$, which reduces the dimensionality of the data $T$.

## SVD Implementation

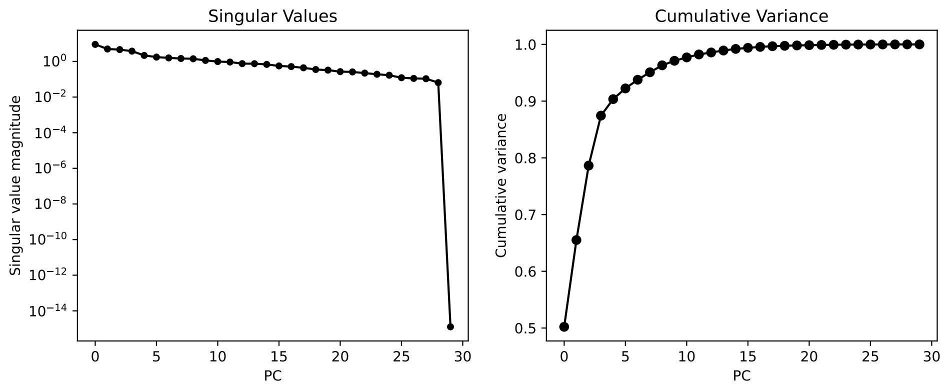
The cutoff condition for the number of principal components is selected on the basis of plots showing the variance as a function of the selected number of principal components. Figure 7-8 shows two such plots. Cutoff percentages of 99% and 95% are used. The cumulative variance of the first $I$ singular values in a $J \times J$ covariance matrix $\Sigma$ is

$$\frac{\sum_{i=1}^{I} s_i^2}{\sum_{j=1}^{J} s_j^2} \tag{7-4}$$

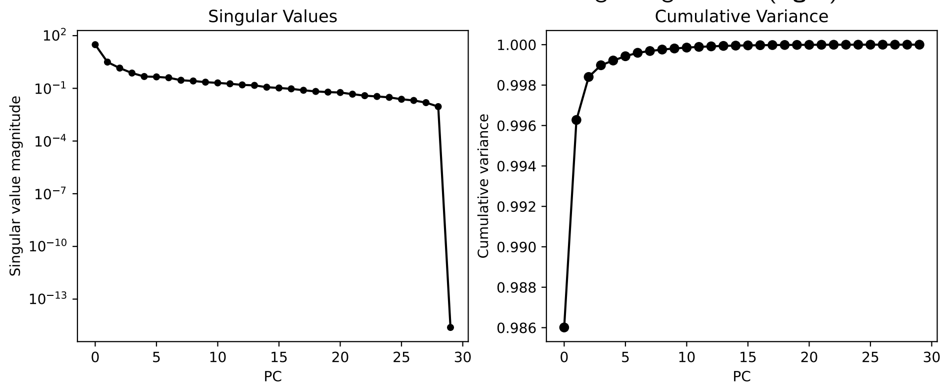
With $min(n, m)$ singular values given by `np.linalg.svd(..., full_matrices=False)`, and the number of non-zero singular values being $rank(B)$, the last singular value is always 0 if $n \leq m$, as $rank(B) \leq min(n, m) - 1$. This is deduced by considering a mean-corrected matrix $B$ with varying numbers of data points in $\mathrm{R}^2$ and $\mathrm{R}^3$, for which the table below shows the number of non-zero singular values:

| number of data points $n$: | 1pt | 2pt | 3pt | 4pt | 5pt |
|---|---|---|---|---|---|
| 2D ($m = 2$): | 0 SV's | $\leq$ 1 SV | $\leq$ 2 SV's | $\leq$ 2 SV's | $\leq$ 2 SV's |
| 3D ($m = 3$): | 0 SV's | $\leq$ 1 SV | $\leq$ 2 SV's | $\leq$ 3 SV's | $\leq$ 3 SV's |

An example of a typical plots for cumulative variance and singular value magnitudes is shown in figure 7-8a. A unique situation is shown in figure 7-8b where the first singular value is already above the 95% cutoff percentage. The code is written so that in these rare exceptions, it takes the first singular value regardless of the variance that this singular value captures.



**(a)** An example of typical plots of singular value magnitudes (**left**) and cumulative variances as a function of the selected number of largest eigenvalues (**right**).



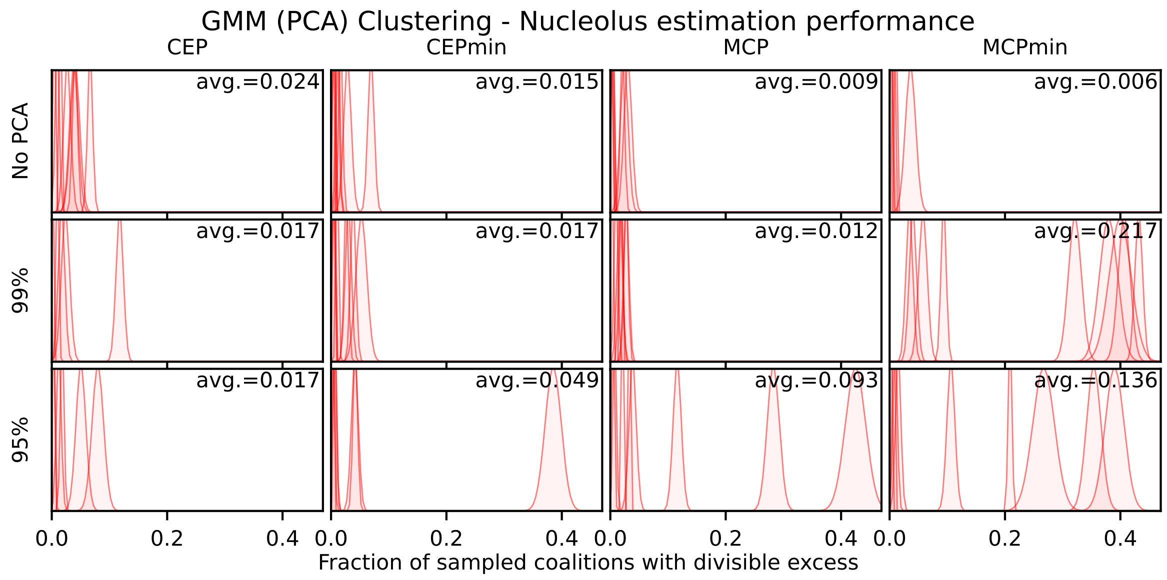
**(b)** A rare situation in which the first PC captures 98.6% of the data variance.

**Figure 7-8:** Two examples showing singular value magnitudes and cumulative variances.

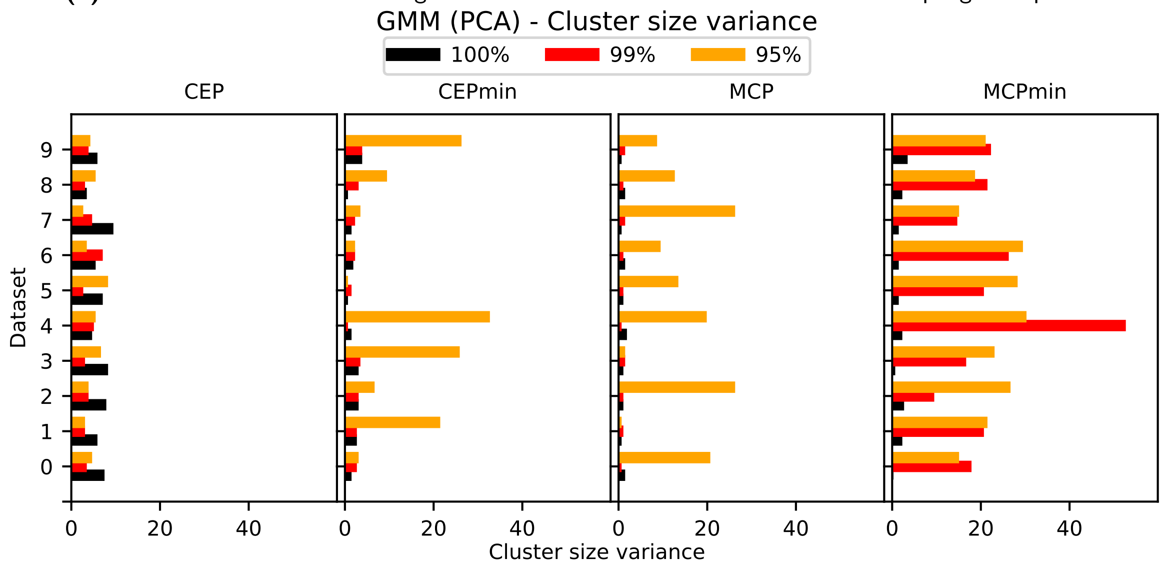
### 7-4-2 GMM-PCA Results

Figure 7-9a shows the nucleolus estimation performance results across 10 datasets, with each Gaussian consisting of 5 separate stratified sampling data points for one nucleolus estimation. The figure shows that in the tradeoff between variability loss and reduced dimensionality, the nucleolus estimation algorithm generally prefers to have complete data. Considering the low inherent dimensionality of feature profiles and the number of prosumers $N$, the GMM algorithm does not suffer from the Curse of Dimensionality.

Figure 7-9b shows cluster size variances. This figure once again shows broad correlations between cluster size variance and nucleolus estimation performance. Also here there are exceptions; GMM using CEPmin with a 95% cutoff shows only one Gaussian located at 40% of sampled coalitions with divisible excesses, while there are 4 datasets with notably high percentages of coalitions with divisible excess that perform better than would be judged by their cluster size variances.



**(a)** The results of GMM clustering with PCA. Gaussians consist of 5 stratified sampling data points.



**(b)** Cluster size variances across all datasets and linkages.

**Figure 7-9:** GMM clustering results with PCA.

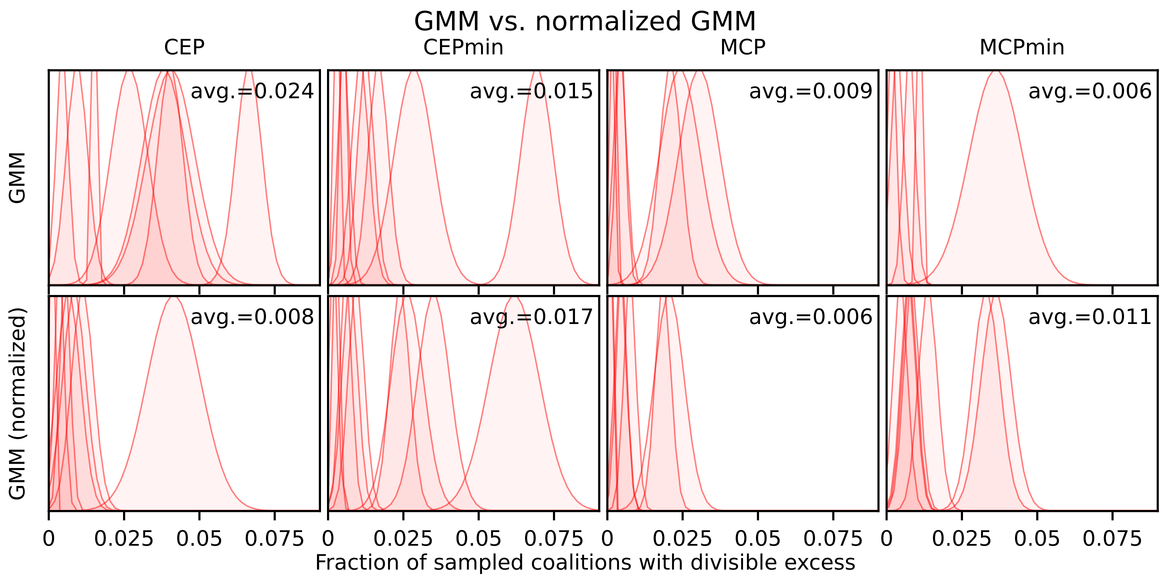
## 7-5 Case Study 5: Shape-based Clustering

This case study looks at the performance of normalized feature profiles for a selection of experiments. Feature profiles are normalized as follows:

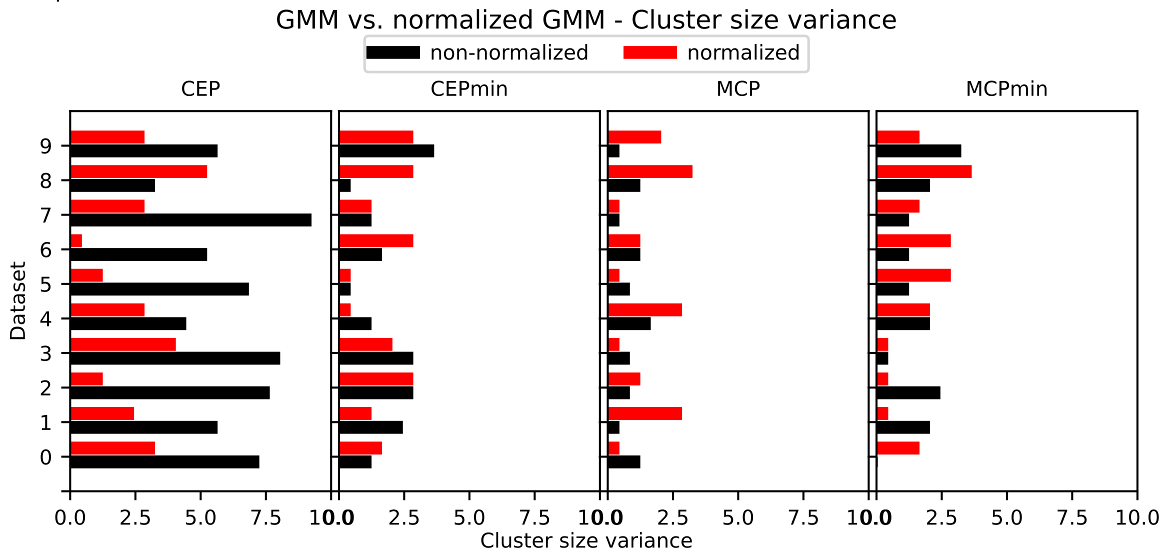
$$\tilde{\mathbf{f}}_i = \frac{1}{\max\left(|\mathbf{f}_i|\right)}\mathbf{f}_i \tag{7-5}$$

### 7-5-1 Shape-based GMM

Here, GMM is applied to regular and normalized feature profiles. Figure 7-10a shows the nucleolus estimation performances, and figure 7-10b the cluster size variances. Notable improvements are found specifically for GMM-CEP, with more uniform cluster sizes for the normalized CEP profile. Other feature profiles do not show any large differences in nucleolus estimation.



**(a)** The results of GMM using regular and normalized feature profiles. Each Gaussian consists of 5 data points.

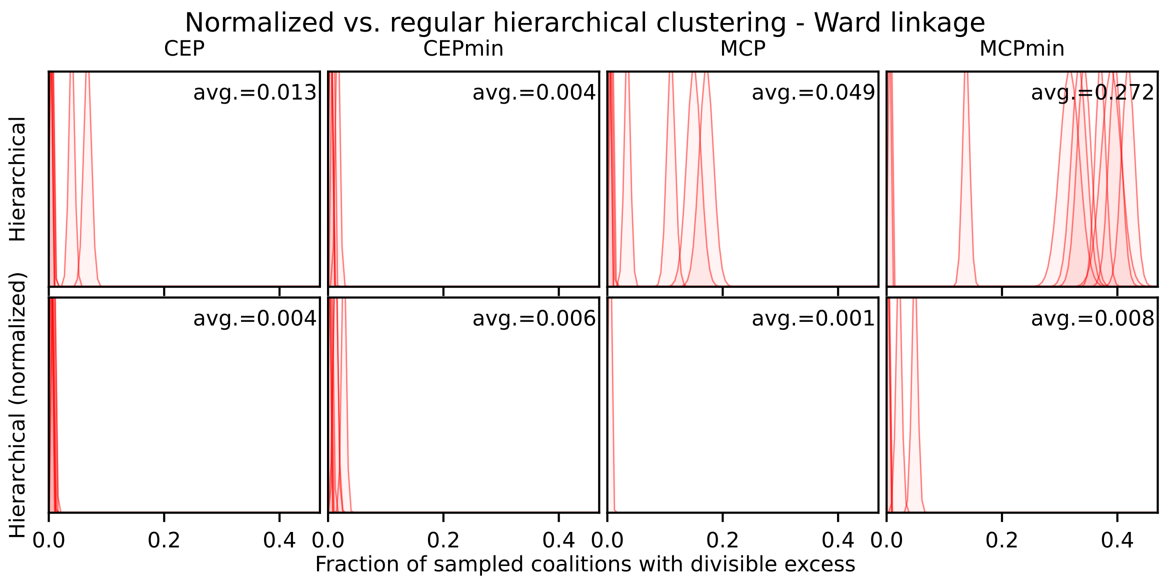


**(b)** Gmm cluster size variances for regular and normalized feature profiles.

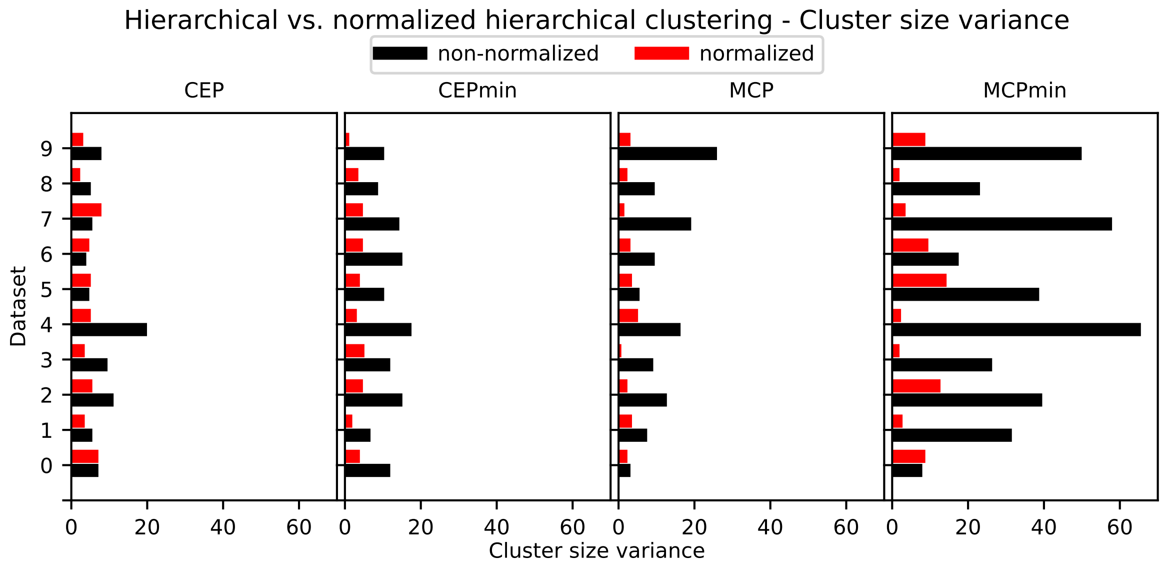
**Figure 7-10:** GMM with regular and normalized feature profiles.

## 7-5-2 Shape-based Hierarchical (Ward linkage)

Figure 7-11a shows that normalized hierarchical clustering performs outstandingly. Figure 7-11b shows that all cluster size variances are reduced with the normalization of feature profiles.



**(a)** The results of hierarchical-Ward using regular and normalized feature profiles. Regular feature profile Gaussians consist of 10 data points each. Normalized feature profile Gaussians consist of 5 data points each



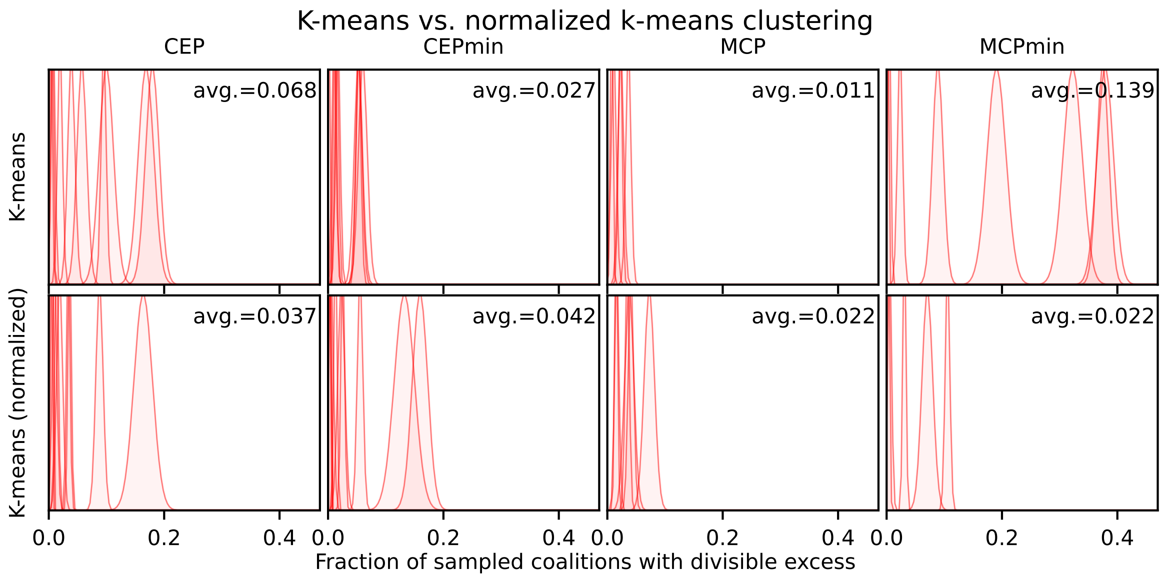
**(b)** Cluster size variances are reduced using the normalization of feature profiles.

**Figure 7-11:** Hierarchical clustering results using regular and normalized feature profiles.

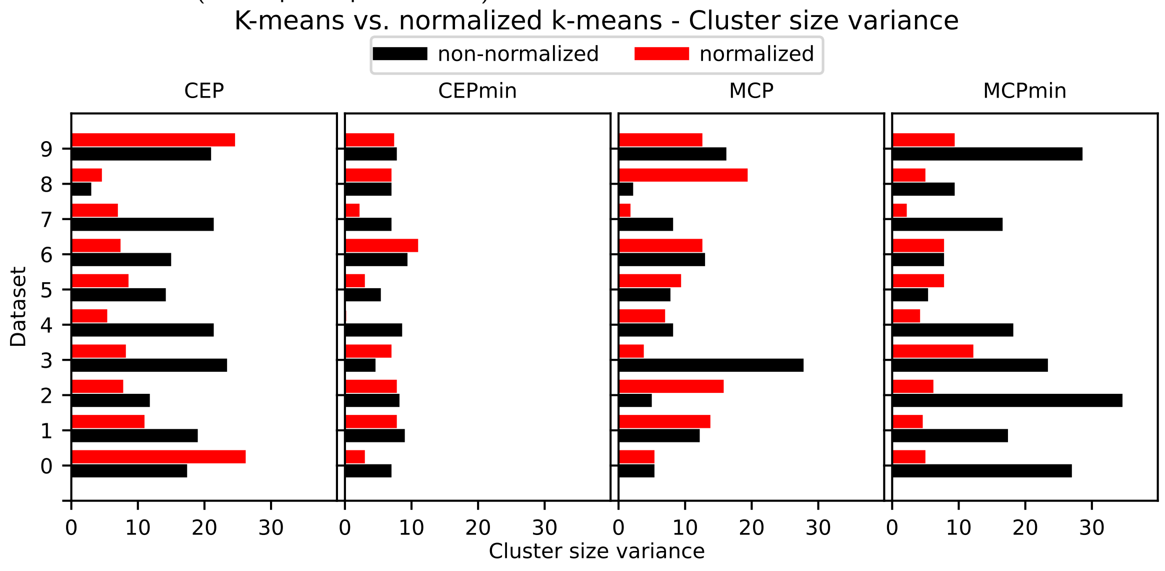
Future work can include the complete linkage in the comparison of normalized and non-normalized hierarchical clustering, as figure 7-7a shows the complete linkage to work well with MCP.

### 7-5-3 Shape-based K-means

Shape-based K-means improves for normalized CEP and MCPmin, while it degrades the quality of nucleolus estimations with CEPmin and MCP (figure 7-12).



**(a)** Nucleolus estimation performances for regular k-means (10 data points per Gaussian), and normalized k-means (5 data points per Gaussian).

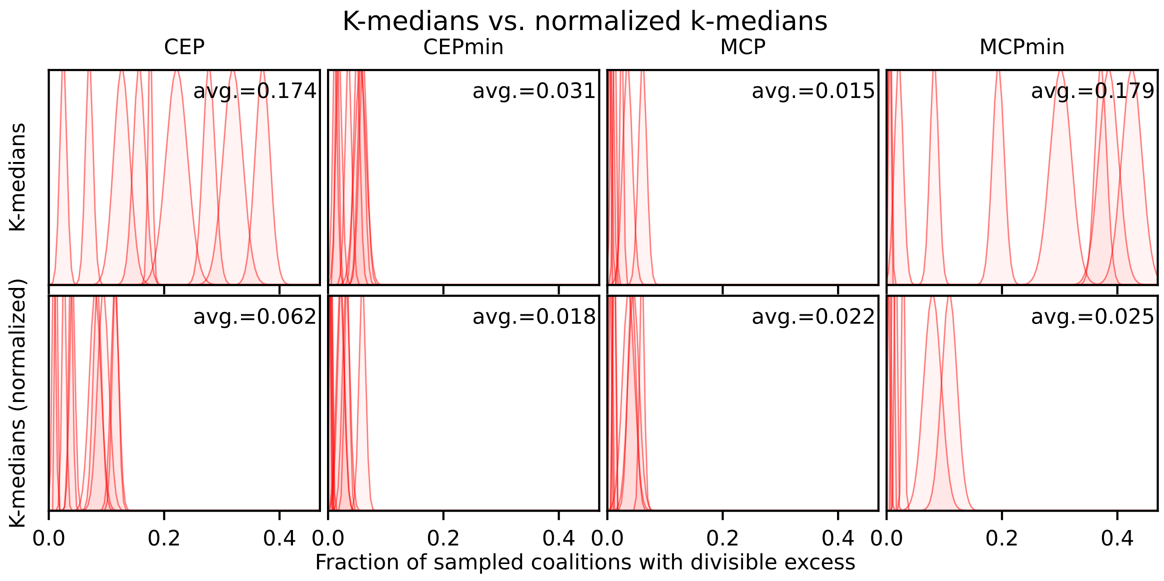


**(b)** Cluster size variances for k-means with regular and normalized feature profiles.

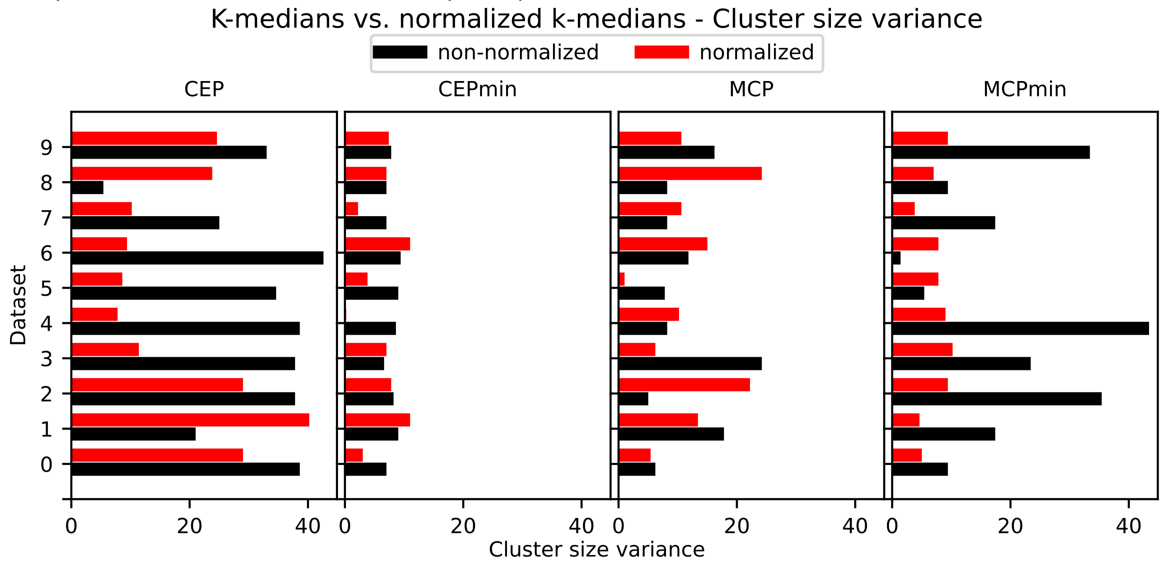
**Figure 7-12:** K-means results with regular and normalized feature profiles.

### 7-5-4 Shape-based K-medians

Figure 7-13 shows the K-medians nucleolus estimation performance improves with the normalization of all feature profiles besides MCP, which only changes by a marginal degree.



**(a)** Nucleolus estimation performance for k-medians. Regular k-medians Gaussians consist of 10 data points. Normalized results use 5 data points per Gaussian.



**(b)** Cluster size variances for k-medians with regular and normalized feature profiles.

**Figure 7-13:** K-medians with regular and normalized feature profiles.

Chapter 8

# Comparisons between Nucleolus Estimations and Full Nucleoli

Repeating the work done in this section with $(N, K) = (15, 5)$ allows the resulting nucleolus estimations to be compared to the full nucleolus for $N = 15$. Given the nucleolus estimation framework, the full nucleolus is computed by setting $(N, K) = (15, 15)$, and skipping the diversity-based pairing section of code.

This section aims to show that evaluating the nucleolus estimation performances of clustering methods using the fraction of sampled subcoalitions with divisible excess (under the stratified random sampling framework) gives an accurate measure of the disparity between nucleolus estimations and full nucleoli, and that choosing clustering methods that give low fractions of sampled subcoalitions with divisible excess will give better nucleolus estimations.

The disparity between nucleoli and their estimates (conventionally measured by the *fraction of sampled subcoalitions with divisible excess*) are evaluated by looking at:

- Euclidean distances between nucleoli and their estimates (section 8-1)
- The fairness of how nucleolus payoff deviations are distributed across prosumers (section 8-2)
- The ability of stratified random sampling to estimate the true fraction of all $2^N$ subcoalitions with divisible excess (section 8-3)

Section 8-5 compares the nucleolus estimation performance of different clustering methods between $(N, K) = (15, 5)$ and $N = (30, 5)$.

## 8-1 Euclidean distance between nucleoli and nucleolus estimations

Figure 8-1 shows that, on average, choosing clustering methods that minimize the fraction of sampled subcoalitions with divisible excess does result in nucleolus estimations with smaller

Euclidean distance deviations of estimated nucleolus payoffs. The y-axis shows the average euclidean distance between nucleolus estimations $\tilde{\nu}$ and true nucleoli $\nu$, where $\nu \in \mathbb{R}^N$ contains payoffs for all $N$ prosumers. On the x-axis, the average fractions of sampled subcoalitions with divisible excess are drawn over all 10 datasets, where stratified random sampling is repeated 5 times for each dataset. The average Euclidean distance considers the average over all 10 datasets.

The computation of a full nucleolus allows the computation of the true fraction of all $2^N$ subcoalitions with divisible excess resulting from a corresponding nucleolus estimation, omitting the stratified random sampling procedure. Figure 8-2 shows the same as figure 8-1, but using the true fractions of all $2^N$ subcoalitions with divisible excess to show that deviations between estimated and true divisible excess results does not change the Euclidean distance-divisible excess relation. Later figures 8-12 and 8-13 show that stratified random sampling in general produces good estimations of the true fraction of $2^N$ subcoalitions with divisible excess.

With either the estimated or true fraction of subcoalitions with divisible excess, many data points still show large variance when considering the line of best fit. This most likely has to do with the cores of datasets not being spherical. A nucleolus estimation can be close to the true nucleolus, but still not be in, or even near, the core, and have a larger associated fraction of subcoalitions with divisible excess, if the core is very narrow in the direction of this nucleolus estimation. Another nucleolus estimation can be farther from the true nucleolus, but be either in the core, or much closer to it (with a lower fraction of subcoalitions with divisible excess), if the core is elongated in the direction of this nucleolus estimation. Future work could confirm if the Euclidean distance relation with the fraction of subcoalitions with divisible excess better follows the average line of best fit when more than 10 datasets are simulated, to average out the effect of narrow or wide cores in the directions of estimated nucleolus payoff vectors.

Red-dashed lines of best fit are the first principal components of standardized data matrices (to have zero mean and a population standard deviation of one). This PC is transformed back onto the correct axes along with all data points.

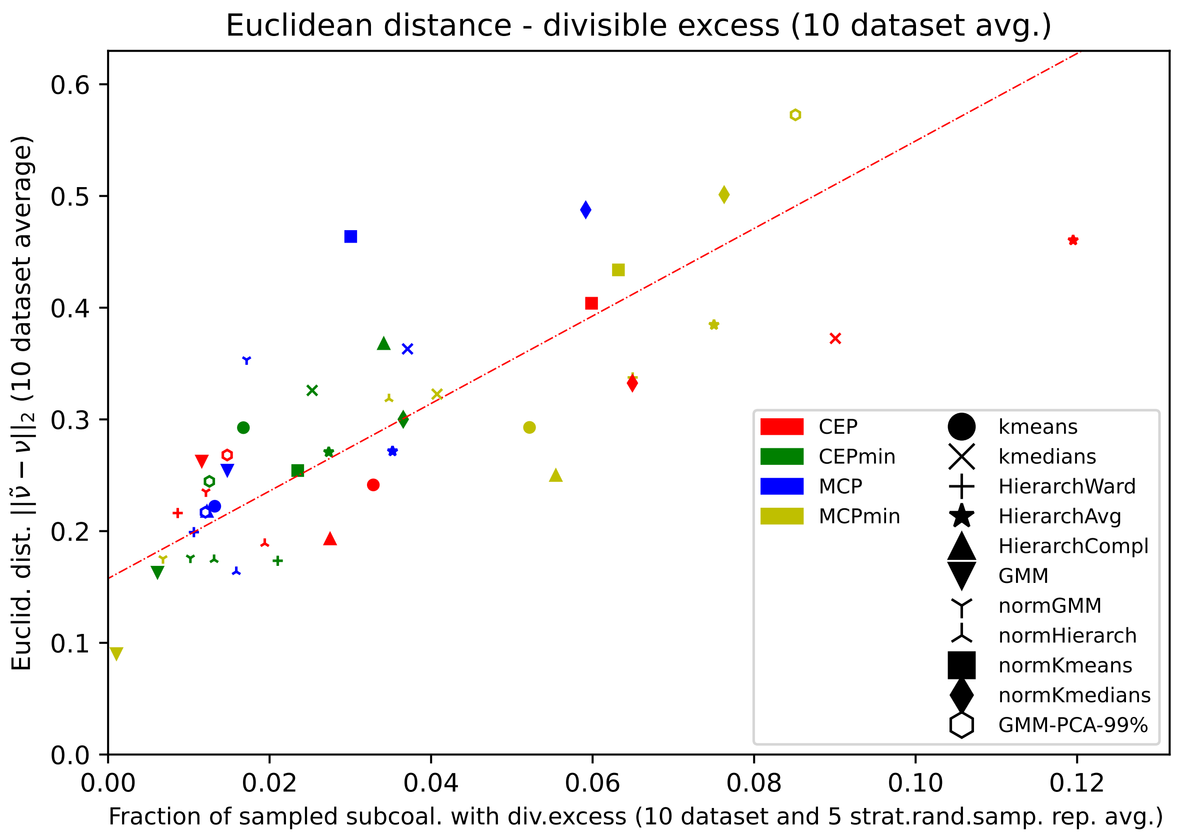


**Figure 8-1:** A linear relation is seen when plotting the average fraction of sampled subcoalitions with divisible excess against the average euclidean distances between full nucleoli $\nu$ and nucleolus estimations $\tilde{\nu}$. Averages are drawn across 10 datasets, and the divisible excess averages are drawn over 10 datasets with 5 iterations of stratified random sampling each.

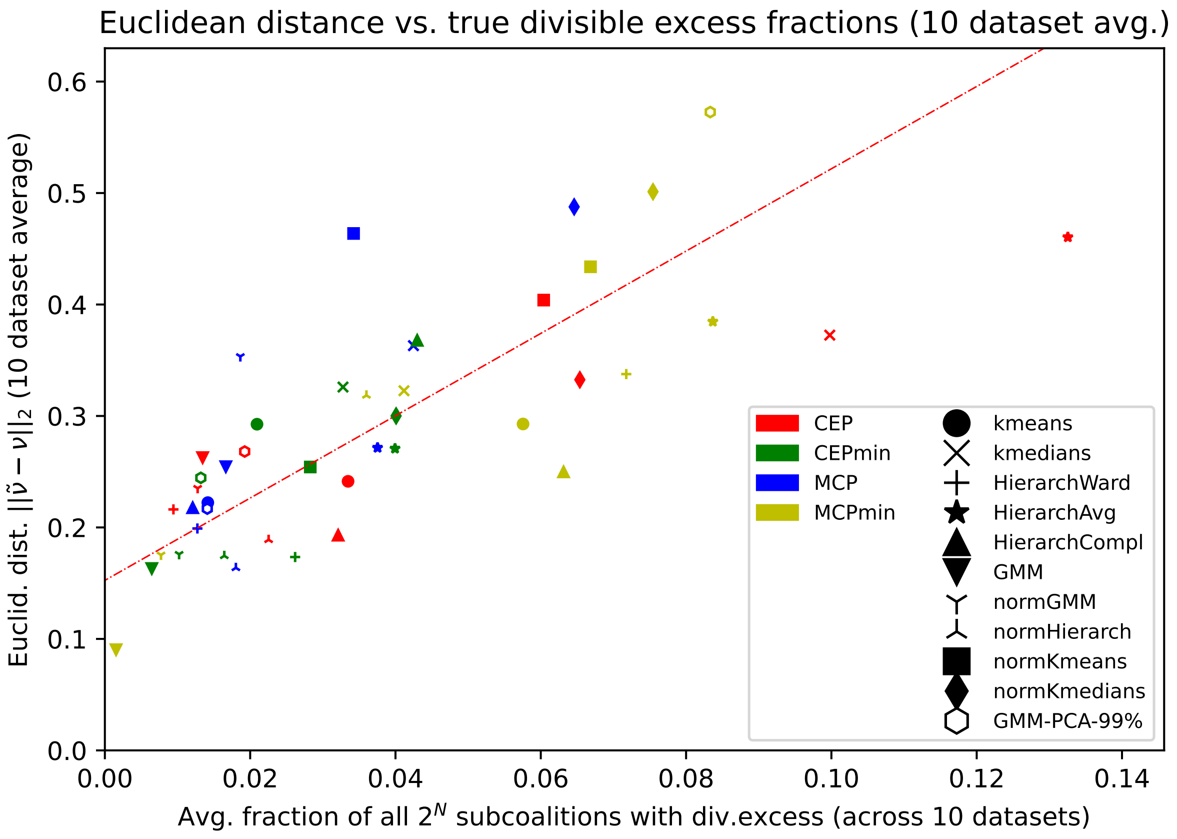


**Figure 8-2:** An identical figure to 8-1, however the x-axis shows the true fractions of all $2^N$ subcoalitions with divisible excess.

## 8-2 Nucleolus estimation deviations and fairness of deviations

The previous section established a linear relation between the fractions of sampled subcoalitions with divisible excess and the Euclidean distance between nucleoli and nucleolus estimations. *How* the improved Euclidean distance is seen in prosumer payoffs, and if there is any improvement in how fairly deviations (deviations being the per-prosumer difference between the nucleolus $\nu_i$ and its estimate $\tilde{\nu}_i$, $|\nu_i - \tilde{\nu}_i|$ for any prosumer $i \in N$) are distributed across prosumers, using clustering methods with better nucleolus estimation performance, is not established yet.

Figure 8-3 shows a hypothetical scenario where two nucleolus estimations can have the exact same Euclidean distance to the true nucleolus, while the fairness of deviations is better in the right figure compared to the left figure.

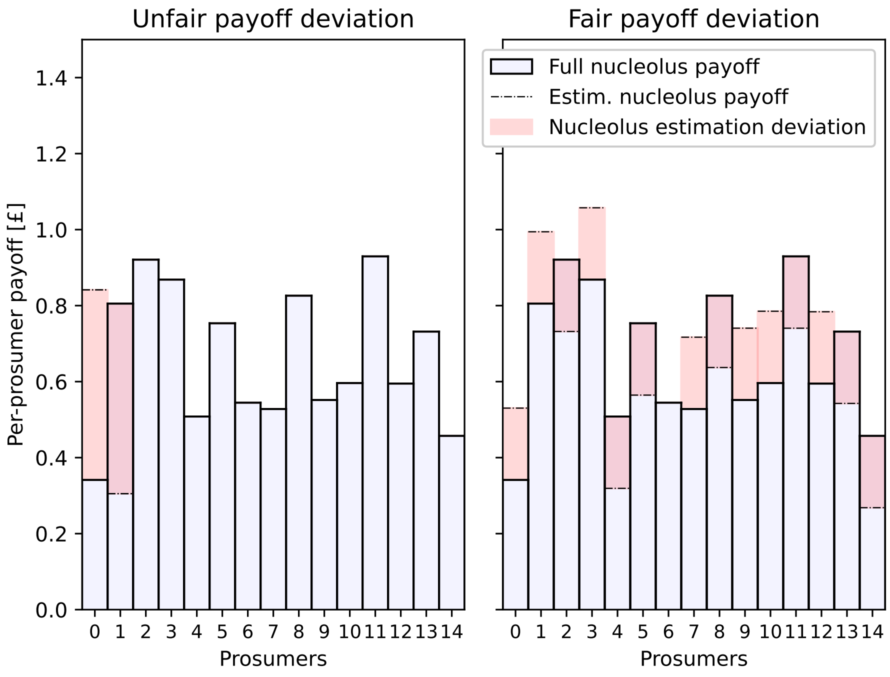


**Figure 8-3:** Two different nucleolus estimations are shown. The left figure shows the deviations between the nucleolus and nucleolus estimation divided amongst prosumer 0 and 1. A more fair nucleolus estimation is shown in the right figure, where the deviations are spread evenly across all prosumers.

While figure 8-3 shows a generic example of fair and unfair distributions of nucleolus estimation payoff deviations, the next few subsections (8-2-1 to 8-2-3) will look at the logical steps taken to form a metric for the fairness of prosumer deviations, the *variance of normalized payoff deviations*.

### 8-2-1 Maximum payoff deviations

Each nucleolus estimation will have one prosumer $i \in N$ with a maximum deviation between its estimated payoff $\tilde{\nu}_i$ and full nucleolus payoff $\nu_i$. Figure 8-4 shows the maximum per-prosumer payoff deviations (£) for different clustering methods, averaged across all 10 datasets. The figure shows clustering methods which perform better in terms of divisible excess, on average generate nucleolus payoffs that minimize the maximum deviations. This is a

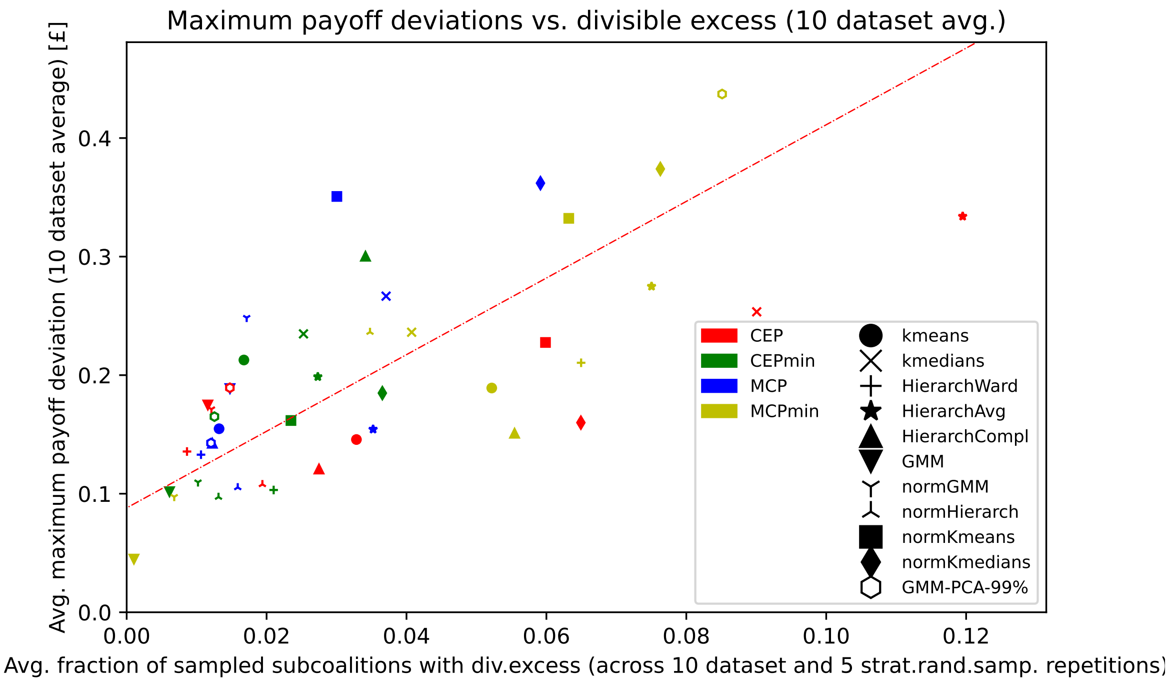


**Figure 8-4:** Clustering methods that give smaller fractions of sampled subcoalitions with divisible excess, tend to produce nucleolus estimations with smaller maximum prosumer deviations.

good characteristic to check, but it does not say much in terms of fairness. It may be thought that minimizing maximum payoff deviations should mean a more even distribution of payoff deviations across prosumers, but a comparison of figure 8-4 and the Euclidean distances (fig. 8-1) show a lot of similarities, suggesting that this metric just tracks the Euclidean distance instead of being a metric of fairness. Furthermore, later figure 8-6 shows a decrease in both maximum payoff deviations (payoff deviations A to payoff deviations C) and fairness.

This all means that while it is good to check that the maximum payoff deviations do not increase for better performing clustering methods, measuring the maximum payoff deviations as a metric for fairness does not work.

### 8-2-2 Variance of payoff deviations

The variance of $N$ payoff deviations is an interesting aspect of a clustering methods' nucleolus estimation performance, with results shown in figure 8-5. However, the variance does not necessarily quantify the fairness of how payoff deviations are distributed. Given two nucleolus estimations, the estimated nucleolus payoff with a smaller Euclidean distance to the true nucleolus can still have worse payoff deviation distribution, which is not necessarily indicated by an increase in the variance. This is shown in figure 8-6, where payoff deviations C are a more unfair distribution of payoffs amongst prosumers, in comparison to payoff deviations A. Despite this, the variance in C is still lower than the variance in A.

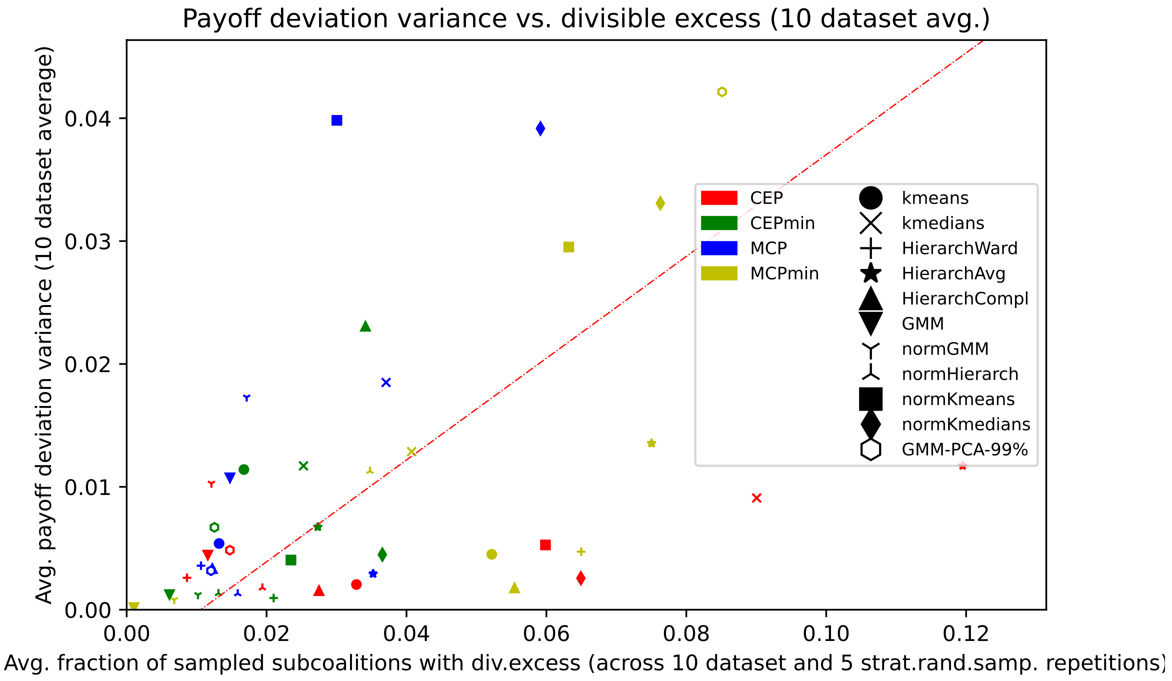


**Figure 8-5:** On average, the variance of payoff deviations $|\nu_i - \tilde{\nu}_i|$ across prosumers $i \in N$ decreases with clustering methods that produce better nucleolus estimation results (smaller fractions of sampled subcoalitions with divisible excess).

## 8-2-3 Variance of normalized payoff deviations

Figure 8-6 shows a slight degradation in fairness when considering payoff deviations C over A, as $\frac{1.337}{0.54} > \frac{2.4}{1.6}$ . This is not reflected in the variance of deviations, but is reflected in the variance of normalized deviations. Payoff deviations B are equally fairly distributed as in A, and the variance of normalized deviations is the same for both.

Given the nucleolus estimation $\tilde{\nu}$ and the true nucleolus $\nu$, all $N$ payoff deviations are normalized to correspond to an average payoff deviation of 1; all $i \in N$ prosumers' payoff deviations are scaled as $|\nu_i - \tilde{\nu}_i| \left(\sum_{i\in N} \frac{|\nu_i - \tilde{\nu}_i|}{N}\right)^{-1}$. The variance of these normalized payoff deviations is a good measure of the deviation distribution fairness.

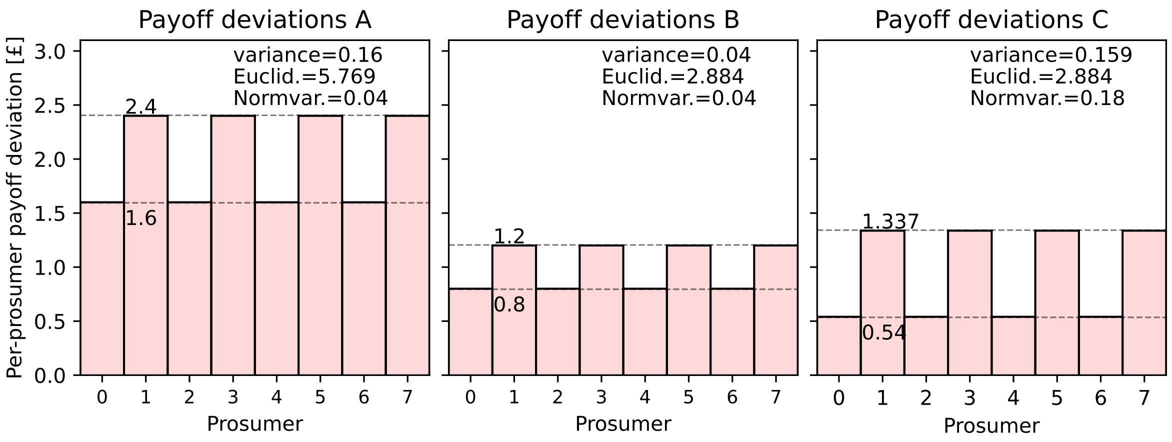


**Figure 8-6:** This figure shows a fictive example of $N = 8$ payoff deviations $|\nu_i - \tilde{\nu}_i|$, expressed in £, generated by three nucleolus estimations (A-C). For each, the variance, euclidean distance and variance of normalized payof deviations is shown.

Figure 8-7 shows the variances of normalized payoff deviations for different clustering methods, plotted against the fractions of sampled subcoalitions with divisible excess, with results being averaged over 10 datasets. The primary PC shows that clustering methods with smaller associated fractions of sampled subcoalitions with divisible excess are more likely to reduce the normalized payoff deviation variance. This means that the fairness of payoff deviation distributions improves, if the aim is to equalize normalized payoff deviations $|\nu_i - \tilde{\nu}_i| \left(\sum_{i\in N} \frac{|\nu_i - \tilde{\nu}_i|}{N}\right)^{-1}$ across all prosumers as much as possible.

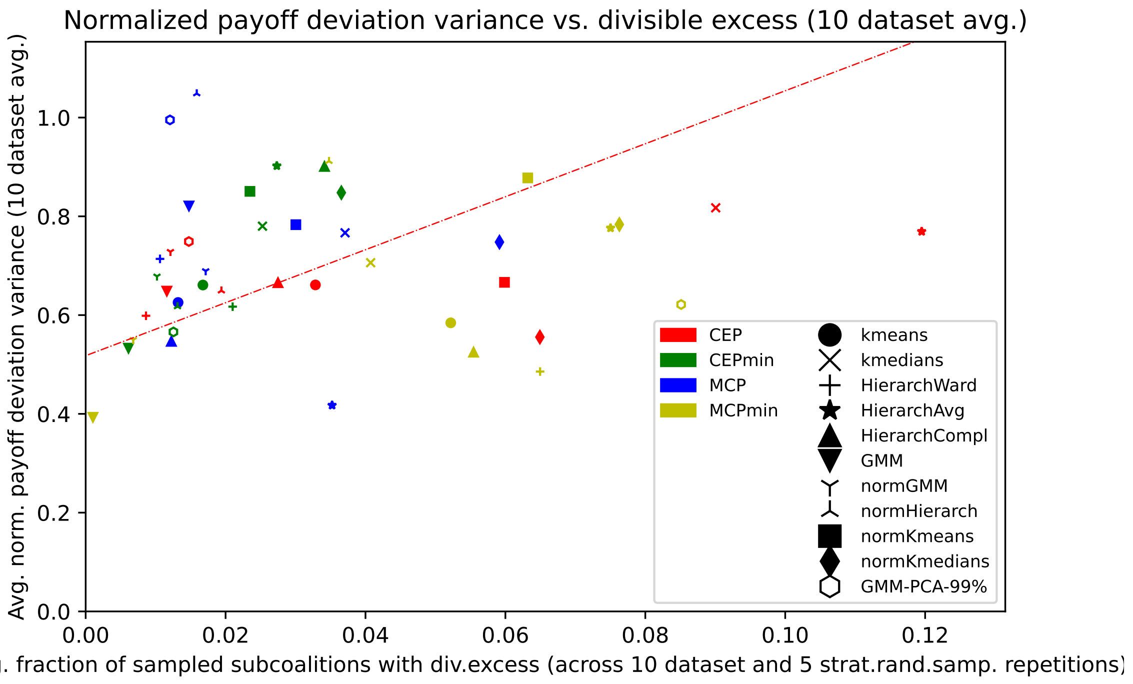


**Figure 8-7:** Clustering methods that reduce the fraction of subcoalitions with divisible excess are more likely to equalize normalized payoff deviations $|\nu_i - \tilde{\nu}_i| \left(\sum_{i \in N} \frac{|\nu_i - \tilde{\nu}_i|}{N}\right)^{-1}$ better.

### 8-2-4 Maximum fractional payoff deviations

Fairness can also be seen from another perspective. Given the full nucleolus $\nu$ and the nucleolus estimate $\tilde{\nu}$ for a particular dataset, instead of equalizing all payoff deviations $|\nu_i - \tilde{\nu}_i|$ across all prosumers $i \in N$, a more fair approach may be to equalize the fractional payoff deviations $\frac{|\nu_i - \tilde{\nu}_i|}{\nu_i}$ across all prosumers instead.

There will be one or more prosumers $i \in N$ for whom the fractional deviation $\frac{|\nu_i - \tilde{\nu}_i|}{\nu_i}$ is maximal. Figure 8-8 shows that better performing clustering methods also minimize maximum fractional deviations. Large values in the y-axis are no cause for concern: figure 8-9 shows that the largest fractional deviations occur for very small nucleolus payoffs. For example, the largest fractional deviation occurs is for an estimated payoff of 0.09413, and a true nucleolus payoff of 0.00016.

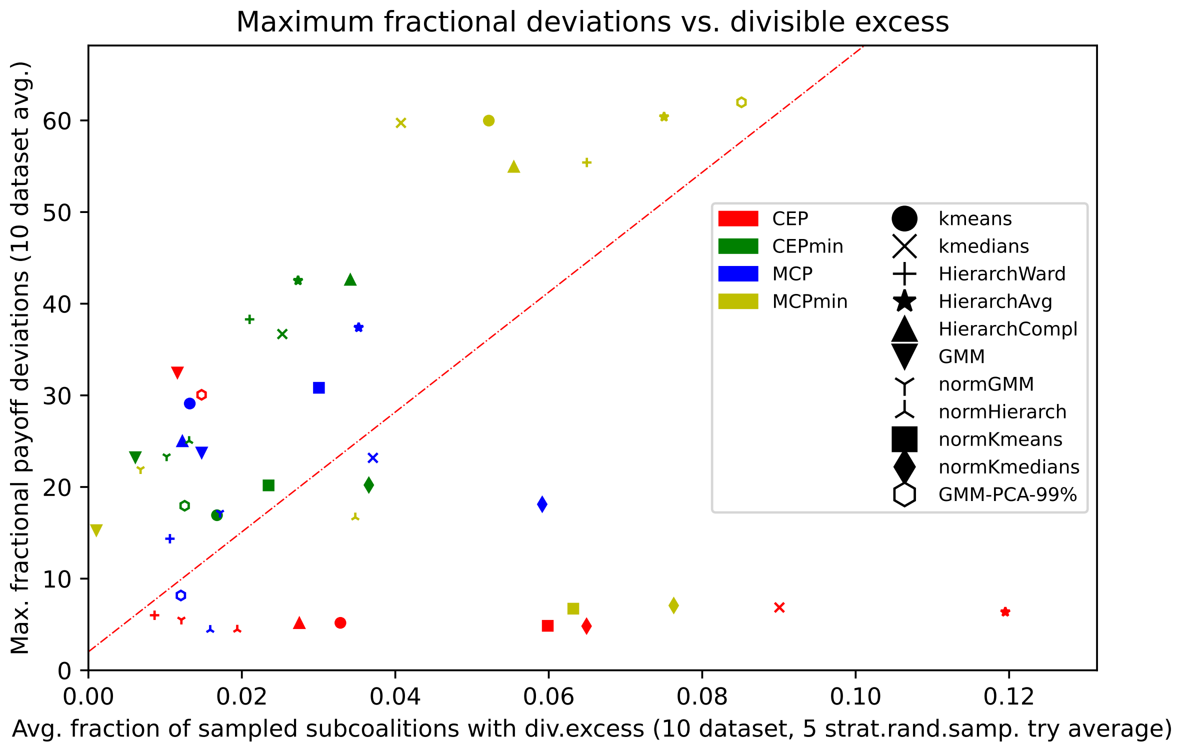


**Figure 8-8:** Clustering methods that perform better in terms of fractions of sampled subcoalitions with divisible excess, also minimize maximum fractional deviations.

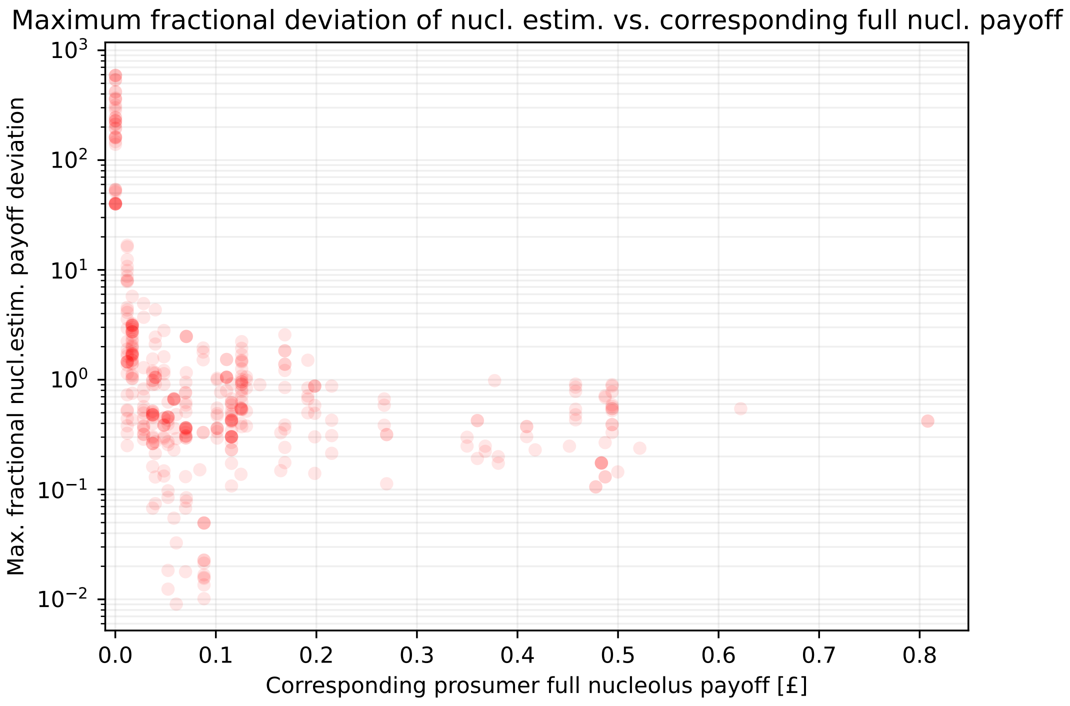


**Figure 8-9:** To explain the scale of the y-axis in figure 8-8, the maximum fractional payoff deviations are plotted against their corresponding true nucleolus payoffs. The largest fractional deviations are associated with very small nucleolus payoffs.

### 8-2-5 Variance of fractional payoff deviations

Figure 8-10 shows the variance of fractional payoff deviations decreasing as clustering methods with better nucleolus estimation performance (smaller fractions of sampled subcoalitions with divisible excess) are chosen.

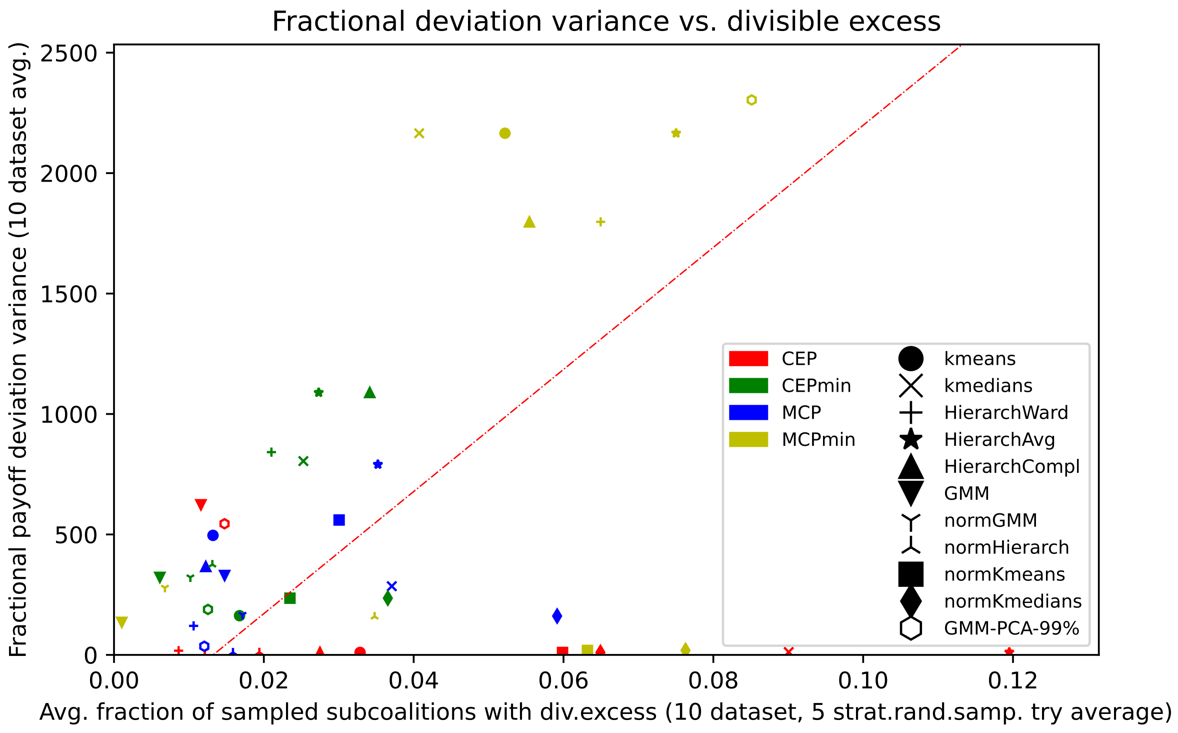


**Figure 8-10:** Variance of fractional payoff deviations decreases with better performing clustering methods.

## 8-2-6 Variance of normalized fractional payoff deviations

Much like how the variance of normalized deviations was a measure of the fairness of payoff deviation distributions, the variance of normalized fractional payoff deviations is a measure of the fairness of fractional payoff distributions. Given an vector $d \in \mathbb{R}^N$ of fractional payoff deviations, normalized to have an average fractional payoff deviation of 1 across all $N$ prosumers

$$d_i = \frac{|\nu_i - \tilde{\nu}_i|}{\nu_i} \left( \frac{\sum_{i \in N} \left( \frac{|\nu_i - \tilde{\nu}_i|}{\nu_i} \right)}{N} \right)^{-1} \tag{8-1}$$

the variance of normalized fractional payoff deviations is calculated as $var(d)$. Figure 8-11 shows that these variances increase for better performing clustering methods. Combined with the results from figure 8-7 which shows variance of payoff deviations $|\nu_i - \tilde{\nu}_i|$ improve, it can be concluded that clustering methods with improved nucleolus estimation performance improve fairness if this is quantified by how evenly normalized payoff deviations $|\nu_i - \tilde{\nu}_i| \left( \sum_{i \in N} \frac{|\nu_i - \tilde{\nu}_i|}{N} \right)^{-1}$ are distributed across prosumers $i$, rather than by how evenly normalized fractional payoff deviations $d_i$ (eq. 8-1) are distributed.

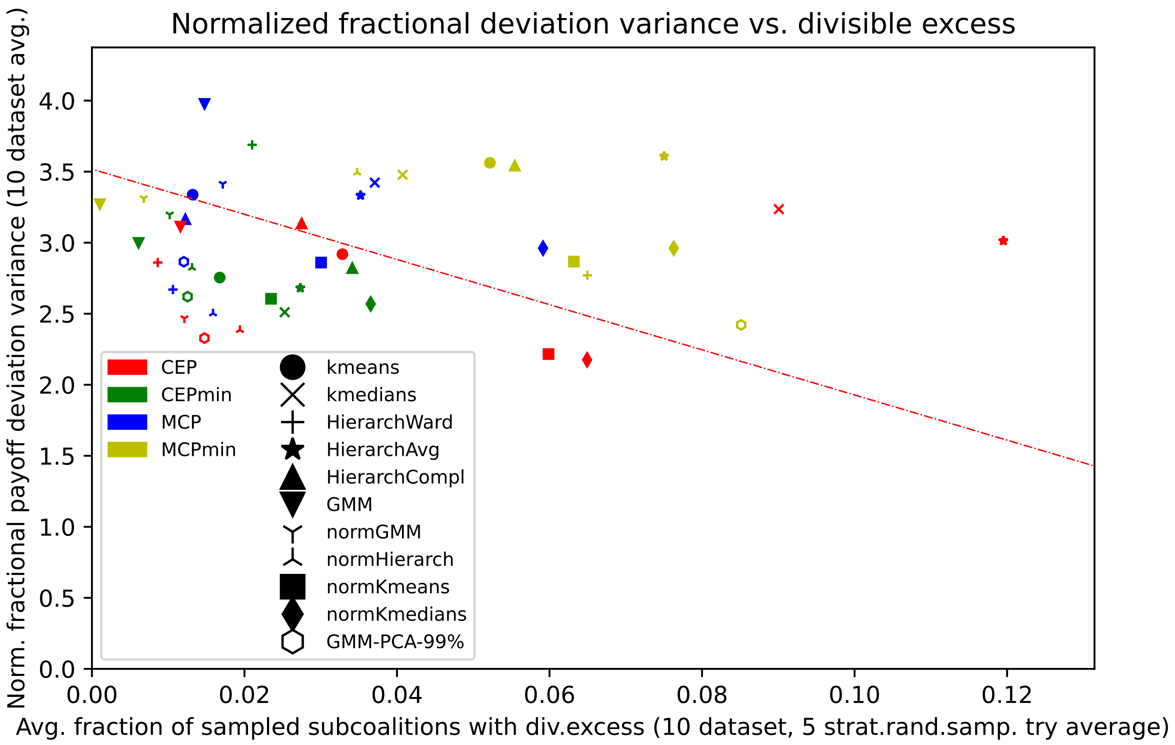


**Figure 8-11:** The variance of normalized payoff deviations tends to worsen for clustering methods with better nucleolus estimation performance.

## 8-3 Stratified random sampling analysis

Given that all $2^N$ subcoalition cost savings $v(\mathcal{T})$ have been computed, it is easy to compare the estimated fractions of sampled subcoalitions with divisible excesses given by stratified random sampling, to the true fraction of subcoalitions with divisible excess. Figure 8-13 shows that stratified random sampling gives good estimations of the true fractions of subcoalitions. Figure 8-12 shows the results for individual datasets, showing reasonable variance around the mean.

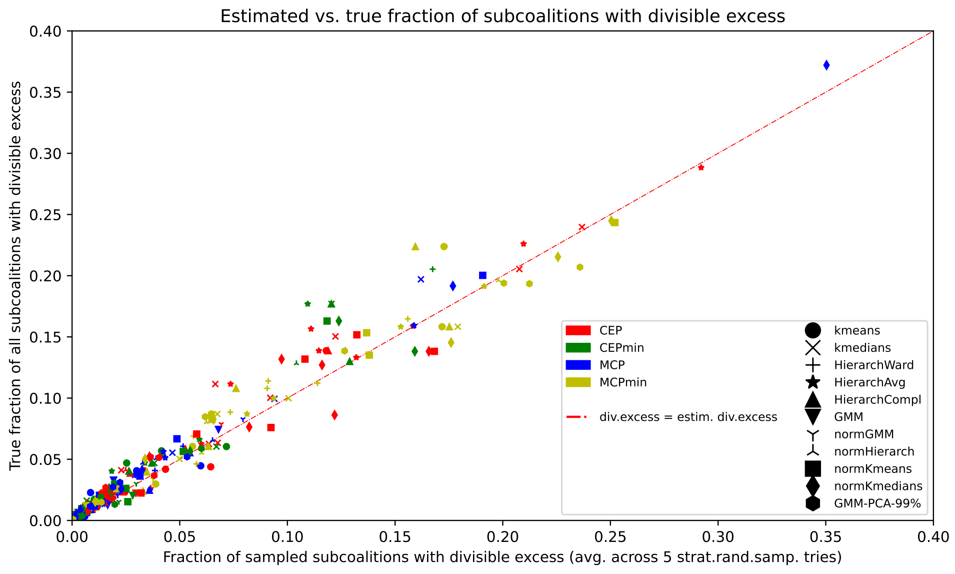


**Figure 8-12:** Estimated vs. true fractions of subcoalitions with divisible excess. Results are shown per dataset

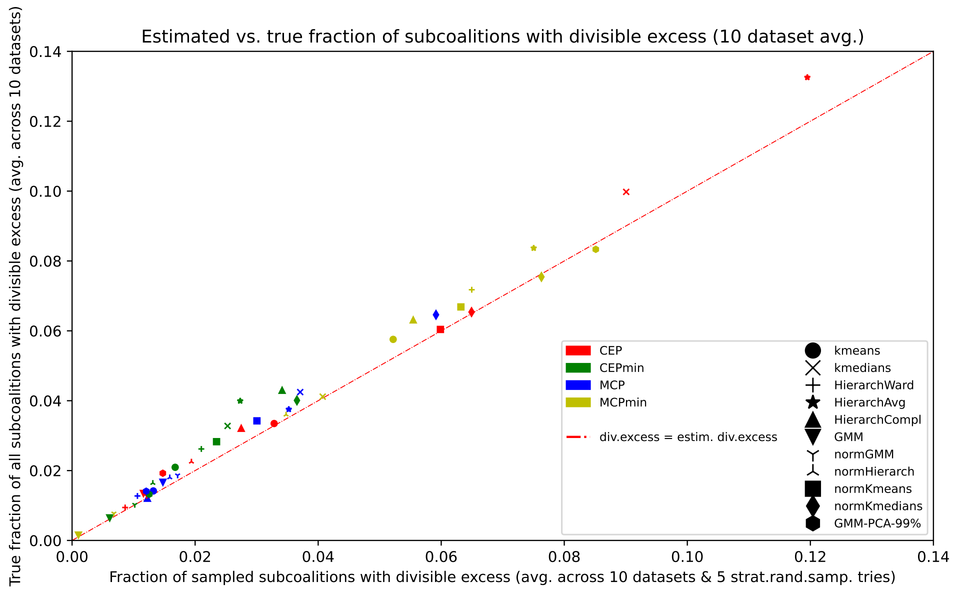


**Figure 8-13:** The same results as in figure 8-12, now averaged over all 10 datasets. Stratified random sampling performs well in tracking the true fractions of subcoalitions with divisible excess.

## 8-4 Conclusion

This section has shown that choosing clustering methods with lower fractions of sampled subcoalitions with divisible excess tends to:

1. Improve the average Euclidean distance between nucleolus estimations and full nucleoli.
2. Reduce the maximum deviations $|\nu_i - \tilde{\nu}_i|$ for corresponding prosumers $i$.
3. Reduce the variance of deviations $|\nu_i - \tilde{\nu}_i|$ across all $i \in N$.
4. Reduce the variance of normalized deviations $|\nu_i - \tilde{\nu}_i| \left(\sum_{i \in N} \frac{|\nu_i - \tilde{\nu}_i|}{N}\right)^{-1}$.
5. Reduce the maximum fractional deviation $\frac{|\nu_i - \tilde{\nu}_i|}{\nu_i}$ for corresponding prosumers $i$.
6. Reduce the variance of fractional deviations $\frac{|\nu_i - \tilde{\nu}_i|}{\nu_i}$ across all $i \in N$.
7. Not improve the variance of normalized fractional deviations $\frac{|\nu_i - \tilde{\nu}_i|}{\nu_i} \left( \frac{\sum_{i \in N} \left(\frac{|\nu_i - \tilde{\nu}_i|}{\nu_i}\right)}{N} \right)^{-1}$
8. Result in nucleolus estimations that reduce the true fraction of *all* $2^N$ subcoalition with divisible excess.

The *variance of normalized deviations* and the *variance of normalized fractional deviations* were considered two potential metrics of fairness.

Points 1 and 4 mean that improvements in clustering methods' nucleolus estimation performance (in terms of divisible excess) will tend to reduce nucleolus estimation payoff deviations as shown in figure 8-14 below

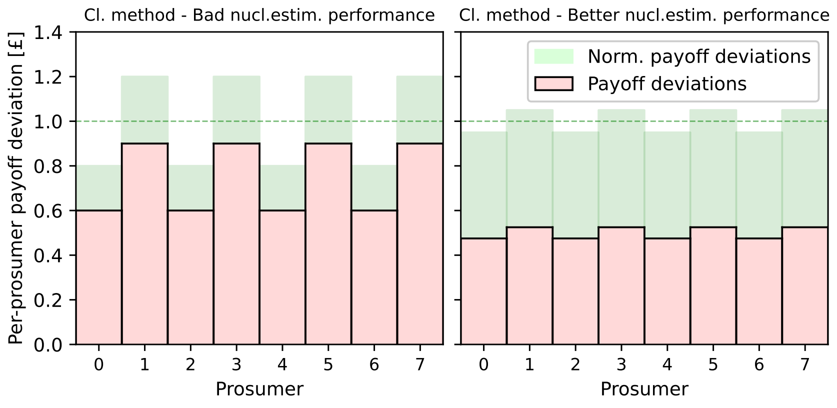


**Figure 8-14:** Implications of improvements in clustering methods' Euclidean distance (point 1) as well as fairness (point 4 - judged by the variance of normalized deviations, shown in green) on nucleolus estimation payoff deviations are vizualized here.

Points 4 and 7 indicate that clustering methods with better nucleolus estimation performance (in terms of divisible excess) will tend to make the distribution of payoff deviations look more like the left subfigure below, in contrast to the right subfigure (figure 8-15).

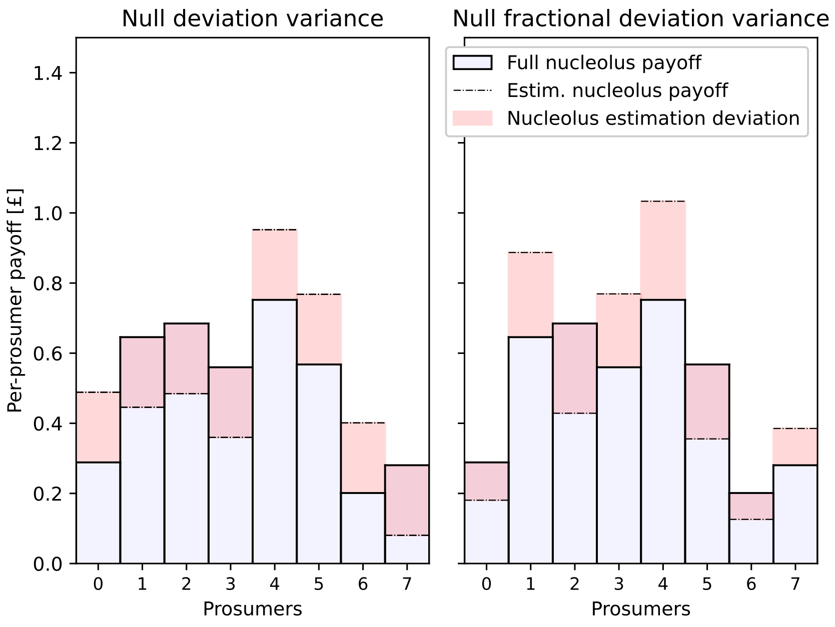


**Figure 8-15:** Two nucleolus estimations are shown for a single true nucleolus. Both estimations have the same Euclidean distance to the true nucleolus. The left subfigure shows a null deviation variance situation (point 4). The right subfigure shows a null fractional deviation variance situation (point 7), where each prosumer's fractional deviation is the same. For completeness, it should be mentioned that the right subfigure fictive example does not consider the condition $\sum(\text{overestimates}) = \sum(\text{underestimates})$, which is always true for any real nucleolus estimation.

## 8-5 Comparisons of N=15 and N=30 nucleolus estimation results

### 8-5-1 K-means and normalized k-means

Looking at regular and normalized (shape-based) k-means, regular k-means using MCP performs the best across all results. An interesting observation is that despite the smaller number of prosumers, some results for $N = 15$ are worse than their equivalents using $N = 30$. For example, normalized k-means performs worse using CEP and MCPmin.

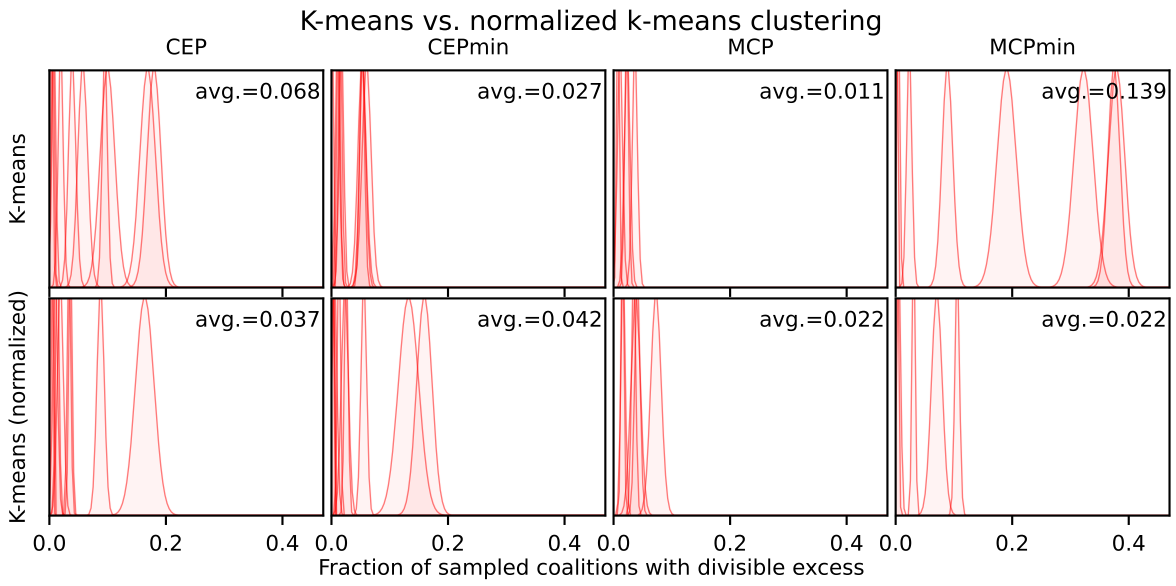


**Figure 8-16:** Nucleolus estimation performances ($N = 30$) for regular k-means (10 data points per Gaussian), and normalized k-means (5 data points per Gaussian).

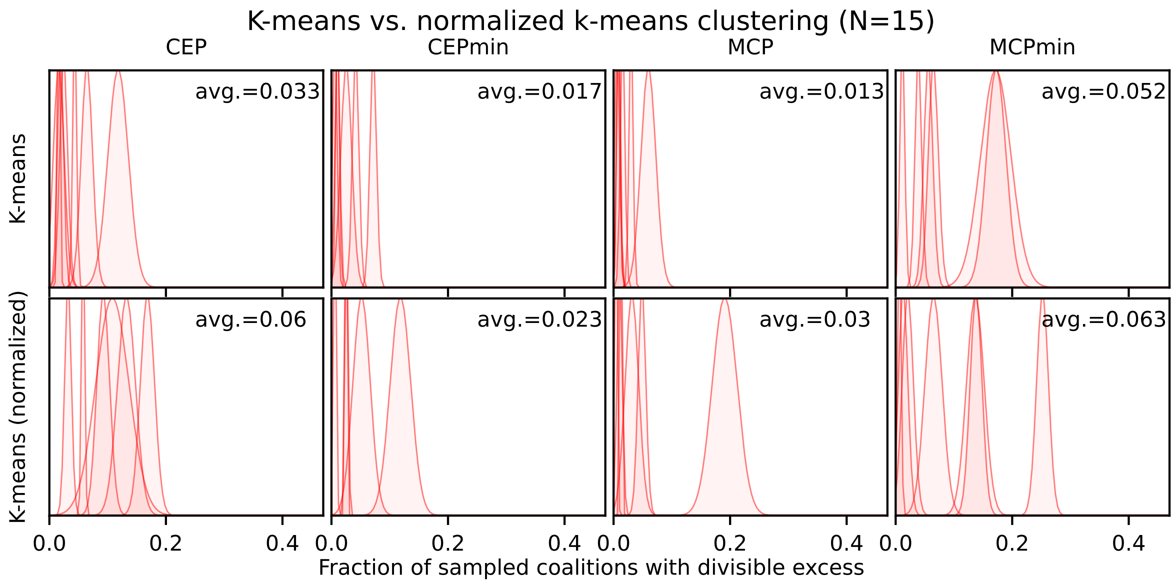


**Figure 8-17:** Nucleolus estimation performances ($N = 15$) for regular k-means (5 data points per Gaussian), and normalized k-means (5 data points per Gaussian).

### 8-5-2 Hierarchical-Ward and normalized Hierarchical-Ward

For hierarchical-Ward clustering, the ambiguity of best-performing clustering methods is equal for both $N = 30$ and $N = 15$. For $N = 15$, there is no notable benefit of shape-based clustering, unlike for $N = 30$.

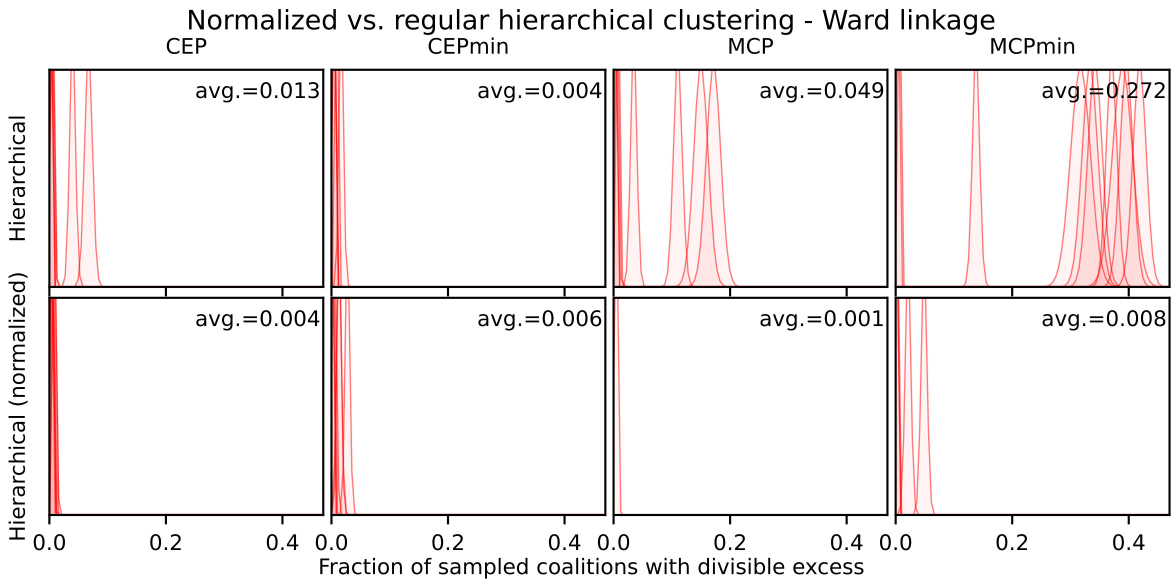


**Figure 8-18:** The results of hierarchical-Ward clustering ($N = 30$) using regular and normalized feature profiles. Regular feature profile Gaussians consist of 10 data points each. Normalized feature profile Gaussians consist of 5 data points each.

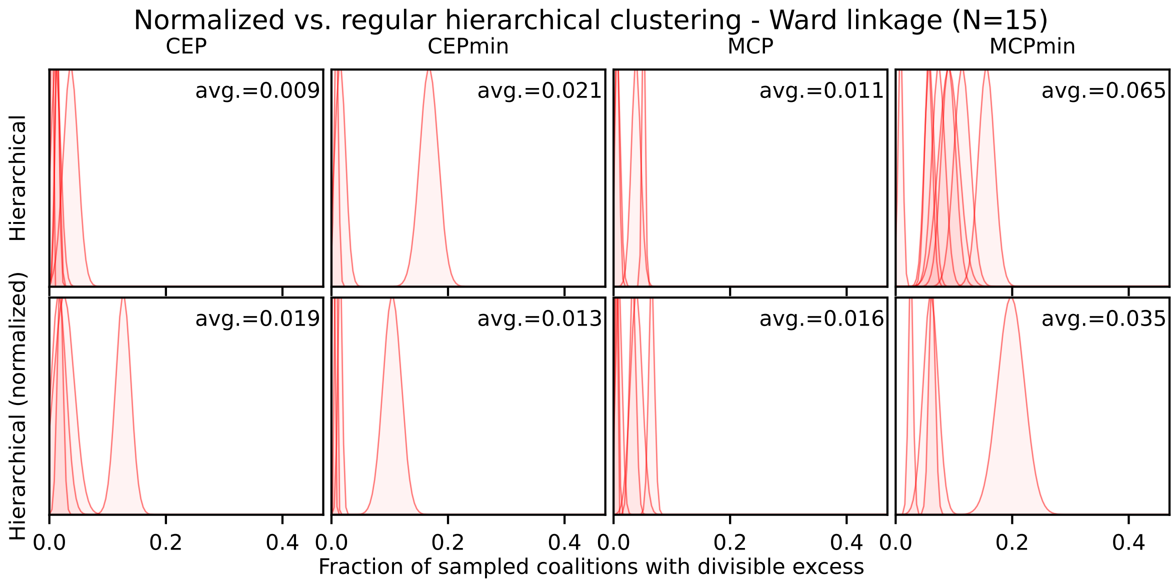


**Figure 8-19:** The results of hierarchical-Ward clustering ($N = 15$) using regular and normalized feature profiles. Regular feature profile Gaussians consist of 5 data points each. Normalized feature profile Gaussians consist of 5 data points each.

### 8-5-3 GMM and normalized GMM

For both $N = 15$ and $N = 30$, no consequent advantage of shape-based clustering is seen. GMM-MCPmin performs very well for both $N$, and was seen to perform the best in terms of average Euclidean distance (fig. 8-1 and 8-2), average maximum absolute payoff deviations (fig. 8-4), and performed very well in average maximum fractional payoff deviation (fig. 8-8) for $N = 15$.

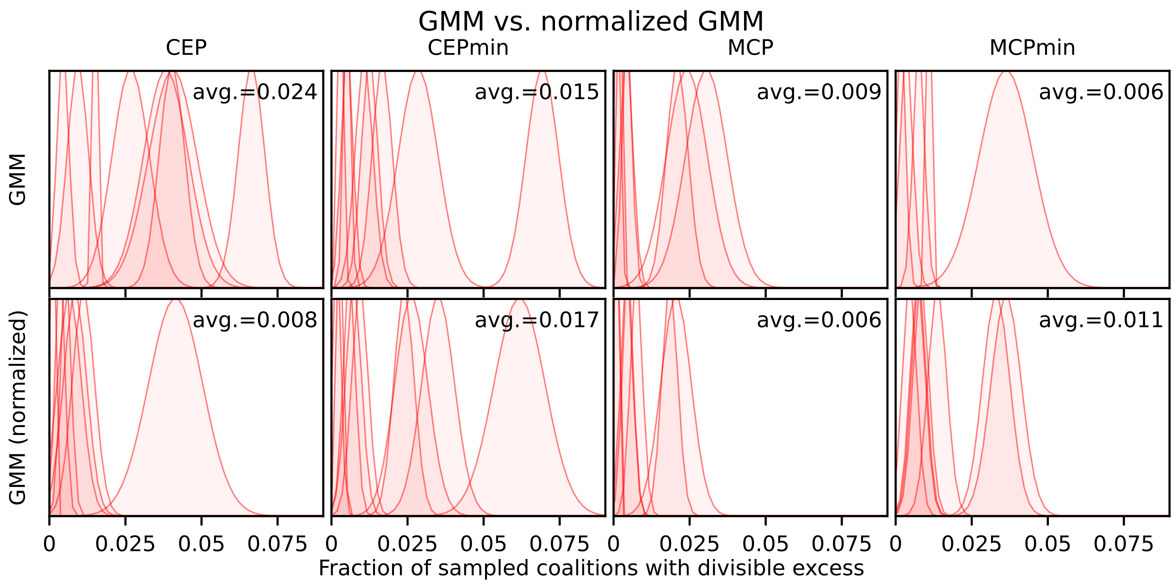


**Figure 8-20:** The results of GMM for $N = 30$ using regular and normalized feature profiles. Each Gaussian consists of 5 data points.

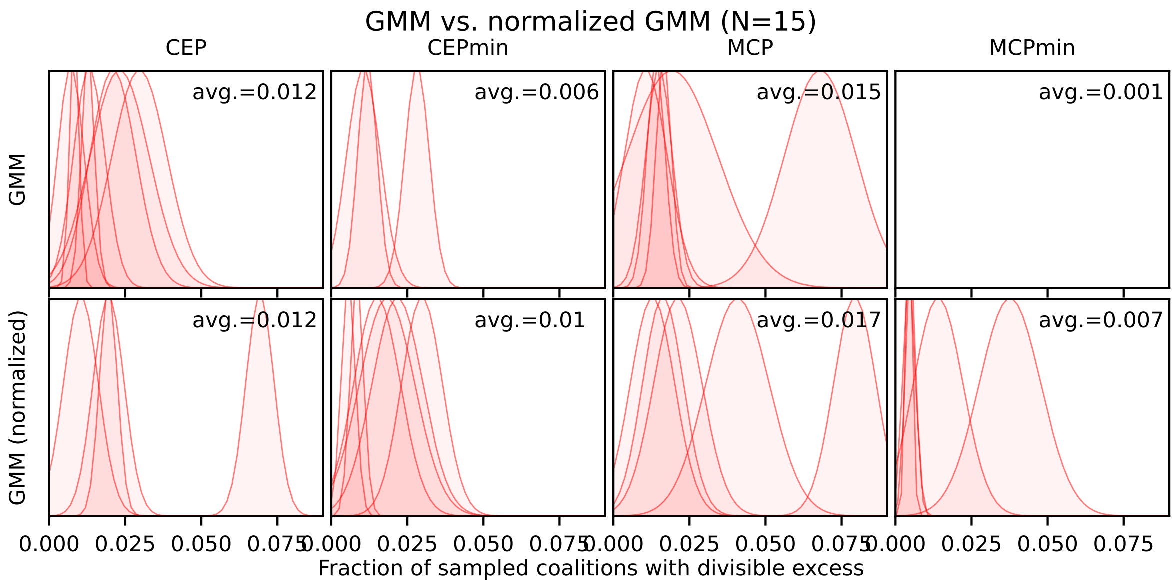


**Figure 8-21:** The results of GMM for $N = 15$ using regular and normalized feature profiles. Each Gaussian consists of 5 data points.

### 8-5-4 K-medians and normalized k-medians

Unlike for $N = 30$, there is no advantage seen in shape-based clustering for $N = 15$ besides for CEP. Across both $N$, k-medians with CEPmin and MCP, and k-medians with normalized CEPmin perform the best.

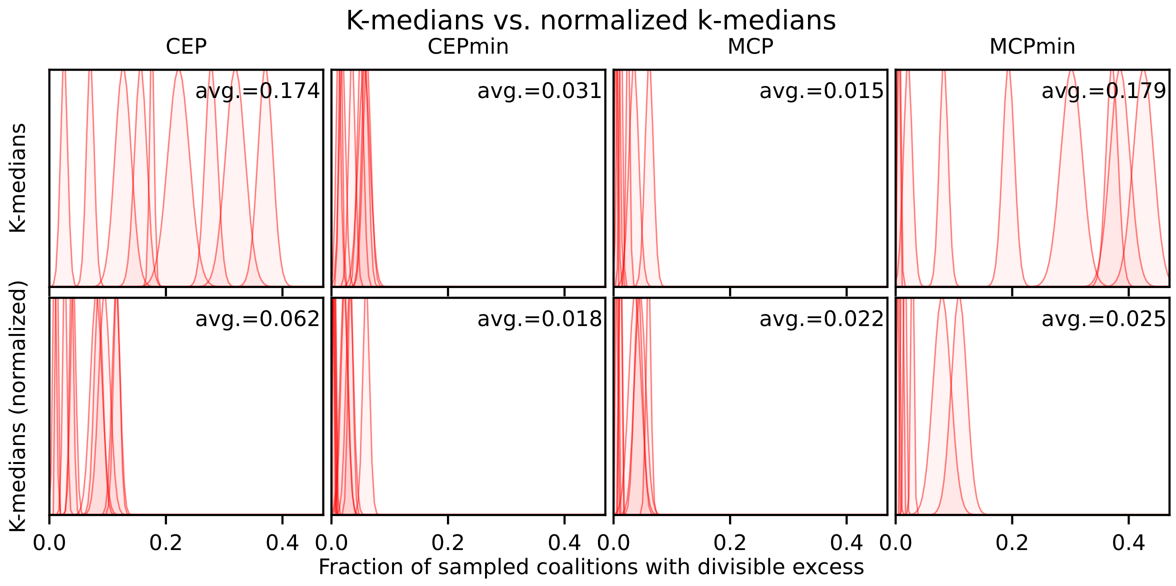


**Figure 8-22:** Nucleolus estimation performance for k-medians ($N = 30$). Regular k-medians Gaussians consist of 10 data points. Normalized results use 5 data points per Gaussian.

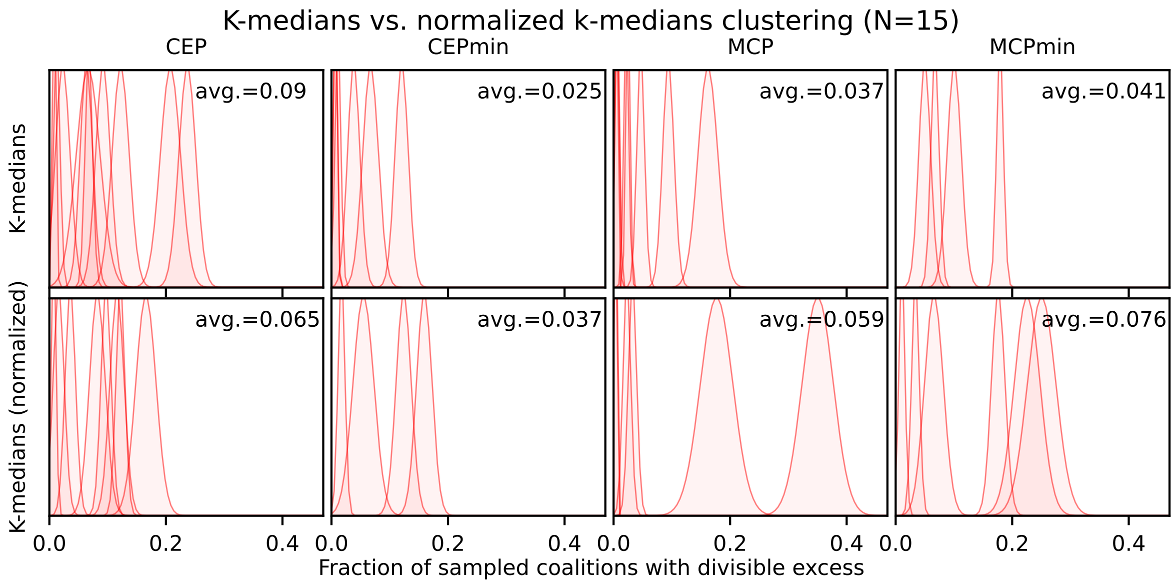


**Figure 8-23:** Nucleolus estimation performance for k-medians ($N = 15$). Regular k-medians Gaussians consist of 10 data points. Normalized results use 5 data points per Gaussian.

## 8-5-5 K-means vs. k-medians

Figures 8-24 and 8-25 below show that out of all k-medians and k-means results, k-means with CEPmin performs the best on average. For both $N = 15$ and $N = 20$, no advantage is gained by using k-medians instead of k-means.

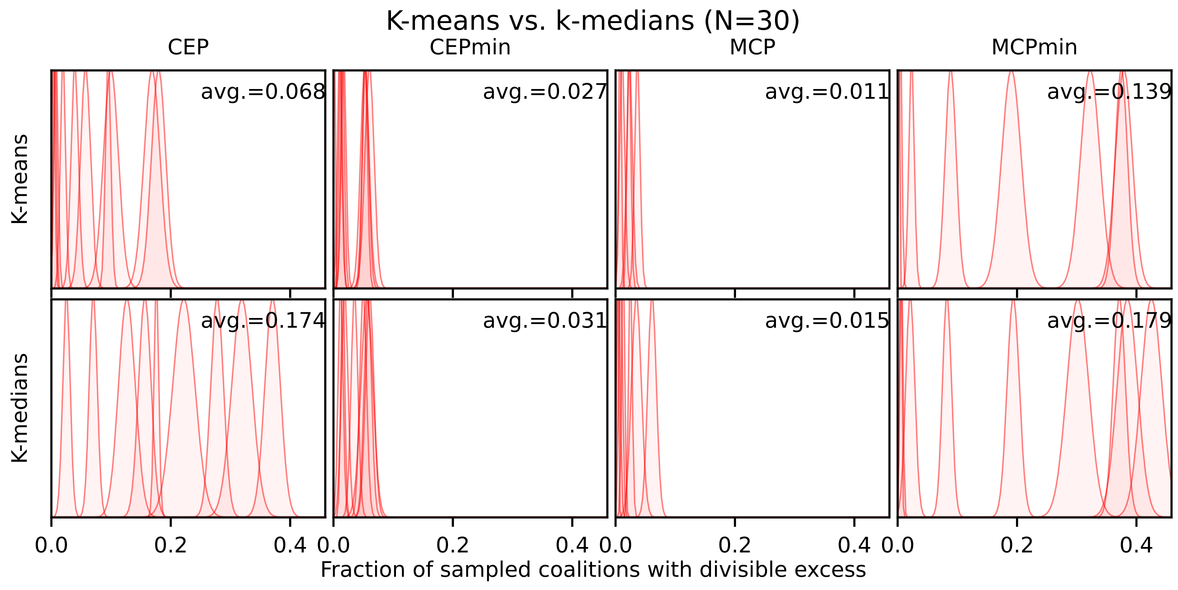


**Figure 8-24:** K-means and k-medians results using $N = 30$.

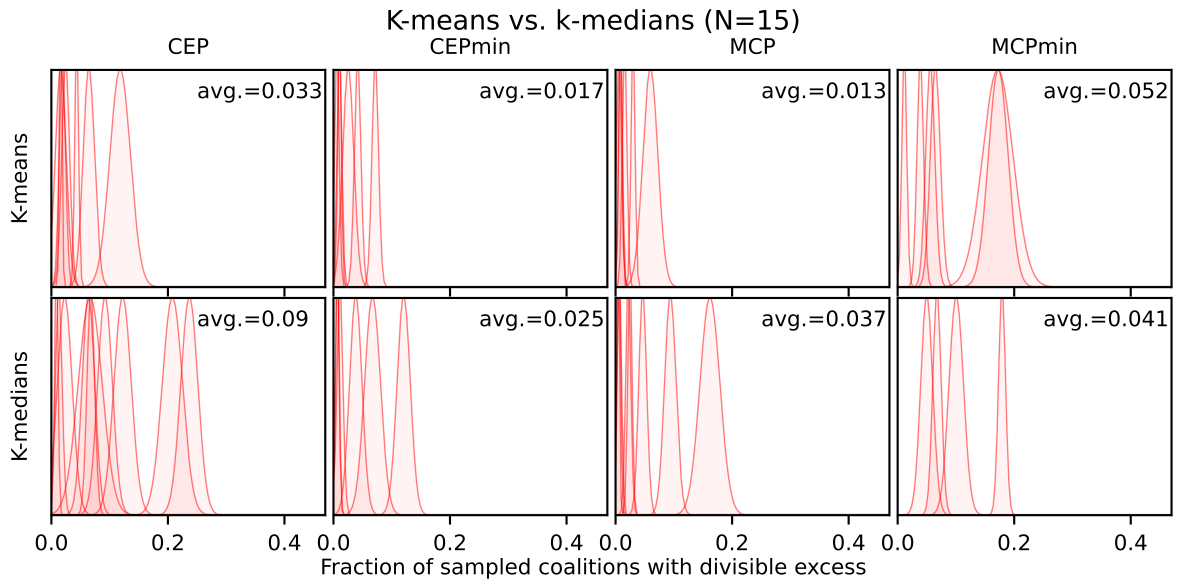


**Figure 8-25:** K-means and k-medians results for N=15.

### 8-5-6 Hierarchical Clustering - Ward/Average/Complete linkages

Considering both $N = 30$ and $N = 15$, the best performing clustering methods with good consistency are hierarchical clustering using Ward-CEP and complete-MCP.

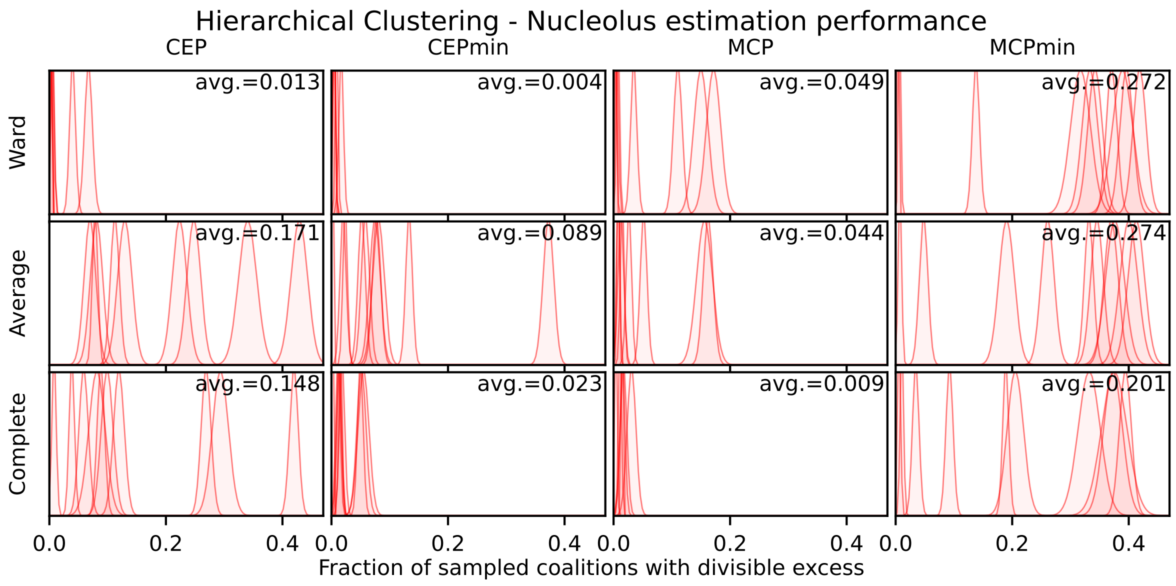


**Figure 8-26:** Results of hierarchical clustering for the Ward, average and complete linkages using $N = 30$. Each Gaussian uses 10 data points.

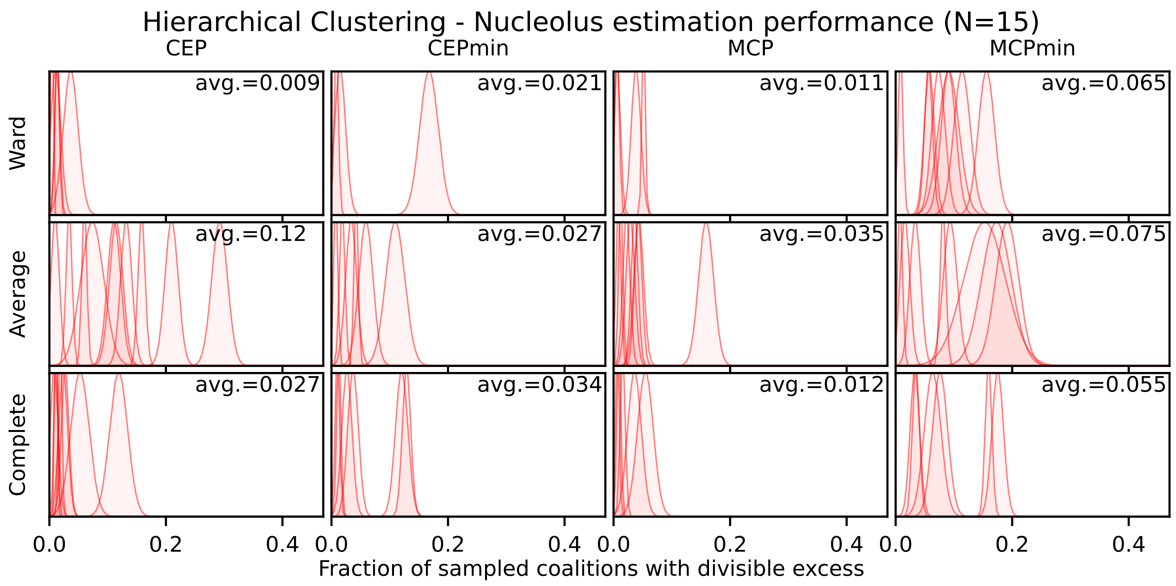


**Figure 8-27:** Results of hierarchical clustering for the Ward, average and complete linkages using $N = 15$. Each Gaussian uses 5 data points.

### 8-5-7 GMM-PCA (99%)

GMM-PCA performs better than GMM for $N = 15$ with the MCP feature profile. This is confirmed in the divisible excess results (fig. 8-29), as well as in the results looking at Euclidean distance (fig. 8-1), and in most other analyses, such as maximum payoff deviations (fig. 8-4), variance of payoff deviations (fig. 8-5), variance of normalized payoff deviations (fig. 8-7), maximum fractional payoff deviations (fig. 8-8), variance of fractional payoff deviations (fig. 8-10), variance of normalized fractional payoff deviations (fig. 8-11).

For $N = 30$, GMM-PCA performs worse with MCP compared to full-dimensionality GMM. No consistent advantage to using GMM on dimensionality-reduced data is shown in these results.

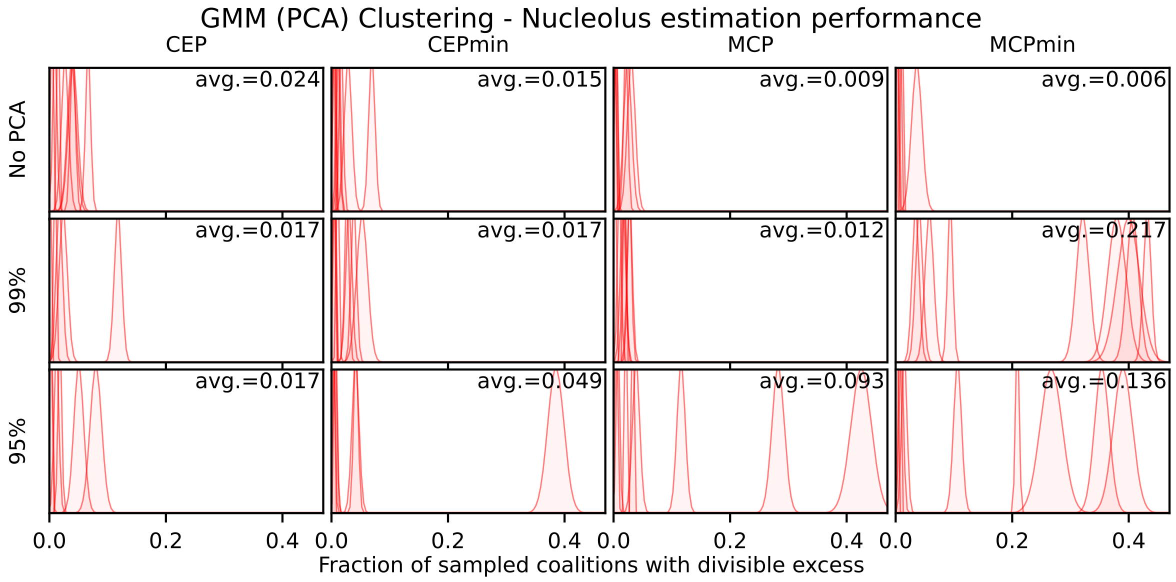


**Figure 8-28:** The results of GMM clustering with PCA, using $N = 30$. Gaussians consist of 5 stratified sampling data points.

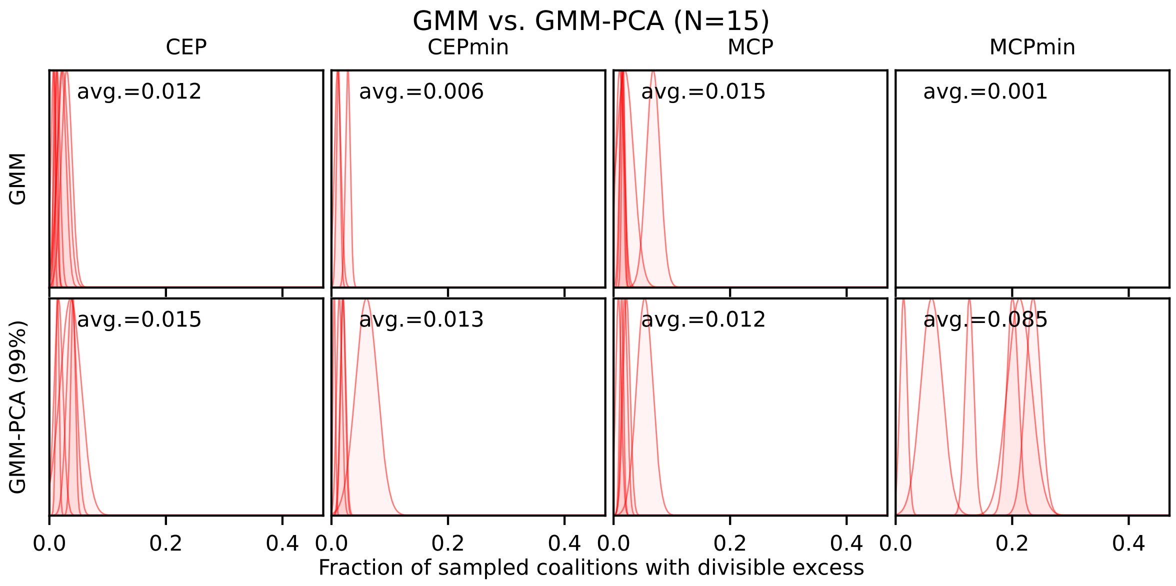


**Figure 8-29:** The results of GMM clustering with PCA, using $N = 15$. Gaussians consist of 5 stratified sampling data points.

## 8-5-8 Conclusions

**Performance degradation with $N = 15$ vs. $N = 30$**
While for most clustering methods, the fractions of sampled subcoalitions decreases for $N = 15$ compared to $N = 30$, in some cases, clustering methods perform worse for $N = 15$ than $N = 30$, both with $K = 5$. Some examples of this are normalized GMM using MCP (figures 8-20 and 8-21), normalized hierarchical clustering using MCPmin (8-18 and 8-19), and normalized k-means using MCPmin (8-16 and 8-17). Seeing how well the average fraction of sampled subcoalitions with divisible excess follows the true average fraction of all $2^N$ subcoalitions with divisible excess (figure 8-13), it is unlikely that variance in divisible excess results causes this.

In any case, nucleolus estimation performance improvements from using smaller $N$ cannot be taken for granted. Results highlight the need to consider various simulations which realistically vary the number of prosumers $N$, from which the clustering method that performs best will be preferred.

**Notable results**
The overall best performing method overall considering this selection of case studies is GMM using the MCPmin feature profiles. The performance of this clustering method, in terms of the fractions of sampled subcoalitions with divisible excess, was very good for both $N = 30$ (figure 7-10a) and $N = 15$ (figure 8-21). GMM-MCPmin also performed very well in most comparisons with full nucleolus payoffs (figures 8-1, 8-2, 8-4, 8-5, 8-7,8-8, 8-10, 8-11).

A list of notable clustering methods that primarily performed well in terms of both Euclidean distances to true nucleoli (fig. 8-1, 8-2), and performed well across both $N = 15, 30$, and then also had good performance across additional analyses (fig. 8-4, 8-5, 8-7,8-8, 8-10, 8-11) is shown below.

- GMM with MCPmin (figures 7-10a, 8-21)
- GMM with CEPmin (figures 7-10a, 8-21)
- normalized hierarchical clustering (Ward) with MCP (figures 7-11a, 8-19)
- normalized GMM with MCPmin (figures 7-10a, 8-21)
- hierarchical clustering (Ward) with CEPmin (figures 7-11a, 8-19)
- normalized GMM with CEPmin (figures 7-10a, 8-21)
- normalized hierarchical clustering (Ward) with CEPmin (figures 7-11a, 8-19)
- normalized hierarchical clustering (Ward) with CEP (figures 7-11a, 8-19)
- hierarchical clustering (Ward) with CEP (figures 7-11a, 8-19)
- hierarchical clustering (complete) with MCP (figures 7-7a, 8-27)
- k-means with MCP (figures 7-12a, 8-17)
- GMM-PCA (99%) with MCP (figures 7-9a, 8-29)

# Chapter 9

# Conclusions and Future Work

## 9-1 Conclusions

### 9-1-1 Best-performing clustering methods

A selection of good performing methods is listed above at the end of section 8-5-8. It was shown that GMM-MCPmin performs best for $N = 15$, and for N=30 GMM-MCPmin had amongst the very best performance as well. This coincides with the results of Han *et al.* [16] where GMM-MCP performed best, assuming that their MCP profiles are generated to be more similar to the MCPmin profiles used here (covered in section 5-5).

For $N = 30$, fuzzy methods (FCMed-CEPmin, FCM-MCP, FCMed-MCP, FCMed-MCPmin for specific values of fuzzifier parameter $m$, shown in figure 7-6) also worked well. However, given that the ideal fuzzifier parameter $m$ depends on the specific feature profiles used as well likely being sensitive to parameters $(N, K)$, fuzzy methods in general seem less practical as finding $m$ is time intensive. Combined with the fact that no fuzzy method outperformed other clustering methods, these other clustering methods can be prioritized.

### 9-1-2 Effects of ES operation

It was shown in section 5-5 that there is no unique set of prosumer ES (dis)charging operations that minimizes the cooperative coalitional cost, and that sharp peaks in ES operation can be reduced by additional constraints (shown in the LP in equation 5-18). While this seems to be a neater method of displaying feature profiles, some results show that certain clustering methods actually prefer MCP over MCPmin. Examples of this are seen for k-means and k-medians (figure 7-1), hierarchical clustering (figure 7-7), and GMM (for both 99% PCA and 95% PCA but not for the no-PCA case, see figure 7-9). This is also reflected in the variances of cluster sizes, with MCP showing lower variances in comparison to MCPmin in all these cases.

Given these results, future work can consider that the ES operations used in clustering by no means also *have* to be used by prosumers. A noisy ES strategy (such as MCP) can be incorporated into feature profiles to form clusters if it helps the clustering method to produce better nucleolus estimations. Once the nucleolus estimation algorithm receives the clustering assignments, the nucleolus estimation algorithm only considers the cost savings of sub-coalitions, which remain equal for any ES operations that optimize costs of coalitions. As there is this freedom in the choice of optimal ES operations, it would make sense to choose the actual ES operation on the basis of any additional criteria that are important in design, for example battery longevity and hardware constraints (where profiles such as CEPmin and MCPmin do better in these regards). This is entirely separate from the choice of which cost-minimizing ES strategy to use for clustering.

### 9-1-3 Evaluation of Stratified Random Sampling as a measure of nucleolus estimation quality

Chapter 8 is in part dedicated to showing that *the fraction of sampled subcoalitions with divisible excess* is a viable metric for the quality of nucleolus estimations. Lower fractions of sampled subcoalitions with divisible excess are tied to smaller Euclidean distances between nucleoli and their estimates (section 8-1), along with improving the fairness of how payoff deviations (differences in payoffs between the nucleolus and its estimate) are spread across prosumers (section 8-2). Furthermore, section 8-3 shows that stratified random sampling accurately approximates the true fraction of all $2^N$ subcoalitions with divisible excess

## 9-2 Future Work

### 9-2-1 Nucleolus estimation instability

Future work can pay more attention to the instabilities involved with computing the nucleolus estimation explained in sections 5-6-3 and 6-2-2. A good start would be to investigate the numerical issues mentioned in Guajardo & Jörnsten [11].

### 9-2-2 Simulating sets of realistic $(N, K)$

Section 8-5-8 noted that a clustering method's nucleolus estimation performance results for one $(N, K)$ pair do not necessarily reflect the performance for other $(N, K)$ pairs. Future case studies should try to simulate scenarios for an accurate range of prosumers.

### 9-2-3 Applying cluster modification methods to new CEPmin and MCPmin profiles

Chapter 6 focused on cluster modification methods, but only regards the CEP and MCP profiles. Given that k-means performs better using CEPmin in comparison to CEP, and that GMM works better for both CEPmin and MCPmin in comparison to CEP and MCP, there is some chance that the cluster modification schemes might perform better overall for these new feature profiles.

### 9-2-4 Prediction divergency effects on grand coalition cost savings

Han *et al.* [16] (described in 3-6-5) ran a Monte Carlo study showing that some variance around predicted values for demand and solar generation profiles resulted in small deviations in the total cost savings of the grand coalition, given that ES operations are optimized for predicted demand and solar generation profiles.

Future work can analyze how skewed predictions effect the total cost savings of the grand coalition. Instead of variance being orientated around the predicted means, true demand and generation profiles can skew away from predicted profiles entirely. These kinds of situations will most likely create a considerably larger reduction in the grand coalition cost savings.

### 9-2-5 Implementing loss functions

Future work could look into incorporating loss functions that model efficiency losses between prosumers, or variable costs of connections between prosumers. This might be crucial in the implementation of P2P networks that span larger geographical areas, where the cost and efficiency of infrastructure will have to be accounted for in the optimization problem. It has to be seen what the consequences (of additional costs and inefficiencies) are on the proof for balancedness as formulated by Han *et al.* in [15]. Maybe a bound on the costs/efficiencies could be established for which balancedness does not hold, or more likely: balancedness might be assumed through empirical measurements of the fraction of sampled subcoalitions with divisible excess, on the basis of nucleolus estimates, which are preferrably as accurate as possible so that any detected divisible excess can be said to stem from excessive costs/inefficiencies rather than bad nucleolus estimations.

### 9-2-6 Implementing smart devices

Smart devices could be incorporated into the model, where the optimization might include a time variable which specifies the time of actuation of smart devices. Optimizing the use of smart devices alongside batteries can further improve independence of Peer-to-Peer (P2P) grids from the power grids, and generate additional cost savings.

### 9-2-7 Implications of increased unfairness in distributions of fractional payoff deviations across prosumers

A worsening of fairness (in the distribution of fractional payoff deviations) with otherwise better performing clustering methods, established in section 8-2-6, definitely does not automatically rule out the use of better performing clustering methods in the first place, even if prosumers would prefer the fair distribution of fractional payoff deviations rather than the fair distribution of regular payoff deviations.

Some loose thoughts to consider for future work:

Given an estimated nucleolus with corresponding fractional payoff deviations shown in A (figure 9-1), the 8 prosumers might be much more likely to accept nucleolus estimation B, which reduces the Euclidean distance between the true nucleolus and its estimate by 90%,

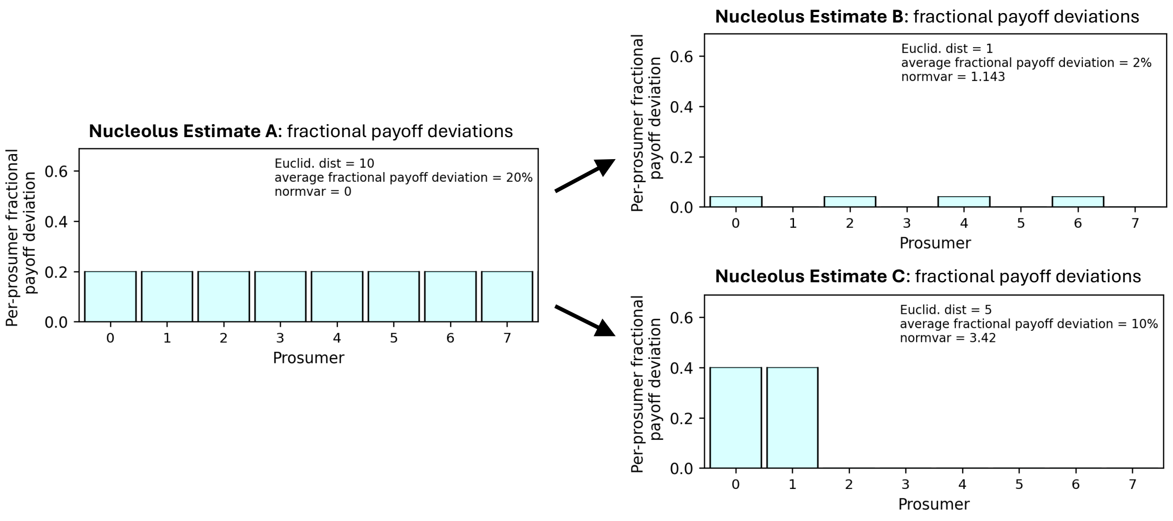


**Figure 9-1:** Three sets of fictive fractional payoff deviations for a coalition of 8 prosumers, corresponding to three separate nucleolus estimations.

but has increased unfairness in the distribution of fractional payoff deviations, evident by the `normvar` metric. Prosumer 0 or 1 (depending on whose nucleolus payoff is underestimated in nucleolus estimate C) will be much less likely to want to partake in the coalition if it recieves underestimates corresponding to nucleolus estimation C, which reduces the Euclidean distance by 50% but with greatly increased unfairness.

In any case, there are definitely imaginable scenarios where the worsening of fairness is worth the improvement in Euclidean distance.

### 9-2-8 Recommender Systems

The prosumer's nucleolus payoff was seen to be quite proportional to its Shapley value (by definition tied to its marginal contribution to coalitions) in figure 2-2. It would be interesting to develop recommender systems, that could recommend whether to increase PV or ES capabilities of the coalition, using prosumers with large (normalized) nucleolus payoffs as a reference, normalized in some way for the magnitude of overall activity in the coalition because prosumers with larger energy flows (more generation, demand, and/or storage) receive larger payoffs.

For example, if it could be established that, over multiple datasets (simulated days), prosumers with battery systems receive larger normalized nucleolus payoffs (normalized by the magnitude of energy flows) in contrast to prosumers with PV systems having comparatively smaller normalized nucleolus payoffs, a valid conclusion may be that investing into more batteries gives the coalition larger cost savings, and the prosumer that decides to invest the sum will receive larger payoffs as a result.

An accurate prediction on the returns of investment would require the additional computations of nucleolus estimations that assume the additional PV or ES as an extra, fictive prosumer (whose payoff will be merged with the payoff of any prosumer making the investment), and these nucleolus estimations must be computed for a wide representative variety of days. The number of clusters $K$ for these parallel nucleolus estimations might be selected to be smaller to speed up computations at the expense of the quality of the returns-on-investment predictions.

# Appendix A

# Feature Profiles

This appendix shows all CEP, CEPmin, MCP, and MCPmin feature profiles used across all case studies in this thesis.

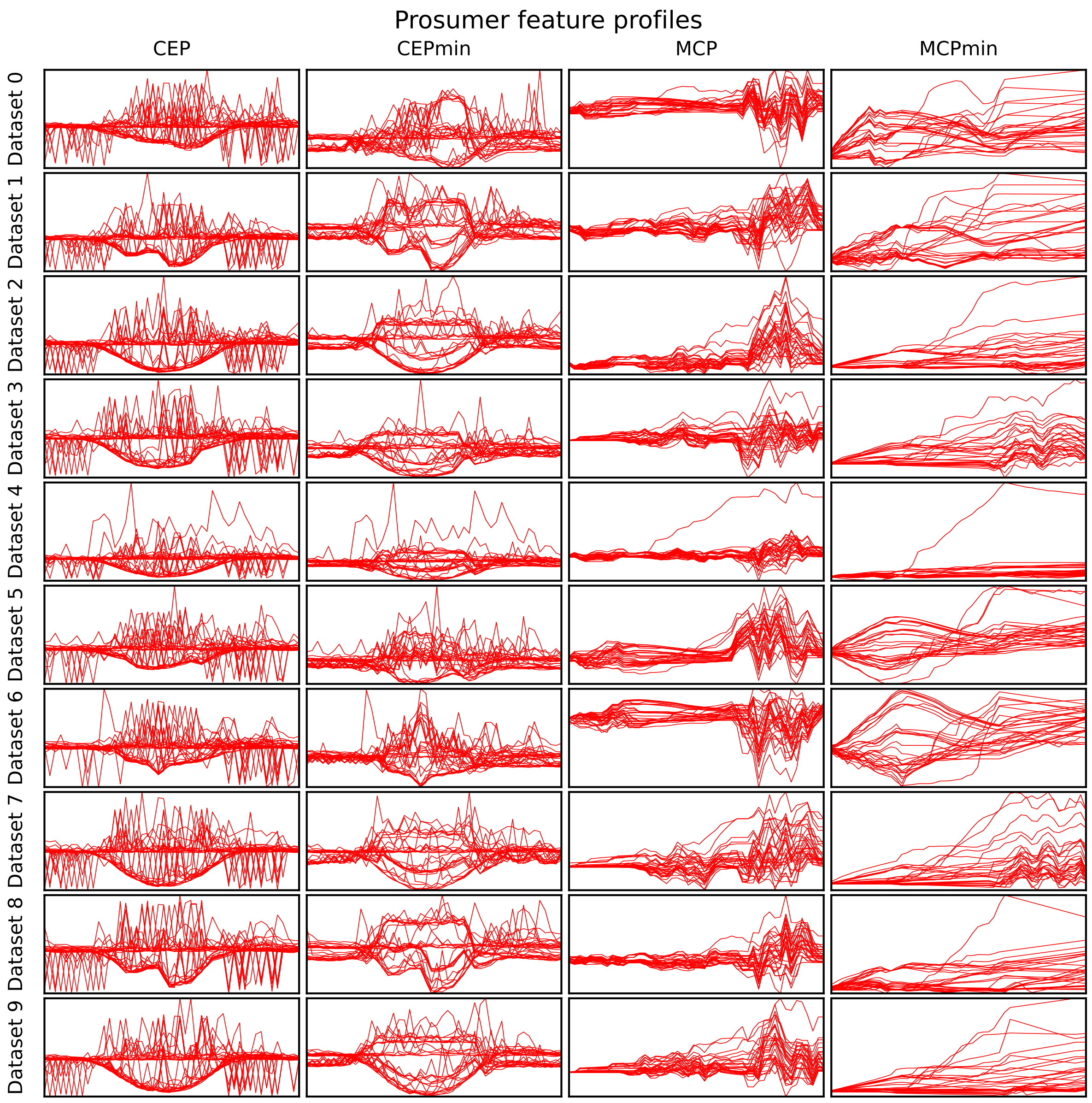


**Figure A-1:** The set of feature profiles used across all case studies.

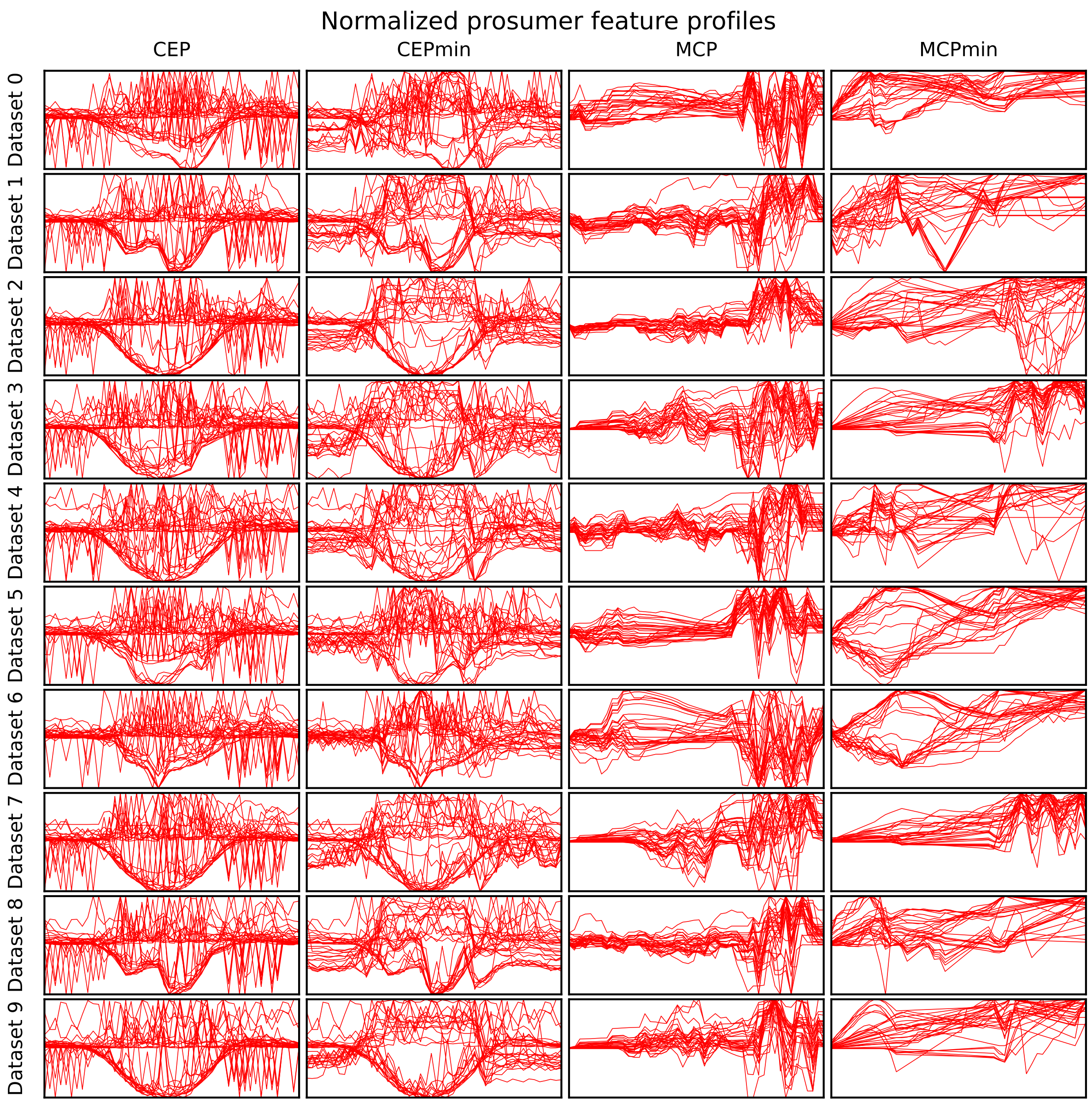


**Figure A-2:** The set of normalized feature profiles used in case study 7-5.

# Bibliography

[1] Robert Aumann. Robert aumann's lecture "the economics of talmud". https://www.youtube.com/watch?v=iWJCSVTl8C8.

[2] Robert J. Aumann. Game theory in the talmud. *Research bulletin Series on Jewish Law and Economics*, June 2002.

[3] Kevin Beyer, Jonathan Goldstein, Raghu Ramakrishnan, and Uri Shaft. When is "nearest neighbor" meaningful? *ICDT 1999. LNCS*, 1540, 12 1997.

[4] James Bezdek. *Pattern Recognition With Fuzzy Objective Function Algorithms*. 01 1981.

[5] CBS. Vermogen zonnepanelen meer dan de helft toegenomen. https://www.cbs.nl/nl-nl/nieuws/2019/17/vermogen-zonnepanelen-meer-dan-de-helft-toegenomen, April 2019.

[6] Gianfranco Chicco. Overview and performance assessment of the clustering methods for electrical load pattern grouping. *Energy*, 42(1):68–80, 2012. 8th World Energy System Conference, WESC 2010.

[7] Leon Cooper and I.Norman Katz. The weber problem revisited. *Computers & Mathematics with Applications*, 7(3):225–234, 1981.

[8] Rudolf Cruse and Christoff Doell. Fuzzy clustering 1. http://fuzzy.cs.ovgu.de/ci/fs/fs_ch09_clustering.pdf.

[9] Julie Delon. The curse of dimensionality. https://mathematical-coffees.github.io/slides/mc08-delon.pdf.

[10] Aleksas Domarkas and Holger Ingmar Meinhardt. https://math.stackexchange.com/questions/1108449/finding-the-nucleolus.

[11] Mario Guajardo and Kurt Jörnsten. Common mistakes in computing the nucleolus. *European Journal of Operational Research*, 241(3):931–935, 2015.

[12] Gregory Gundersen. A fast and numerically stable implementation of the multivariate normal pdf. https://gregorygundersen.com/blog/2019/10/30/scipy-multivariate/.

[13] L. Han, T. Morstyn, and M. McCulloch. Constructing prosumer coalitions for energy cost savings using cooperative game theory. In *2018 Power Systems Computation Conference (PSCC)*, pages 1–7, 2018.

[14] Liyang Han, Thomas Morstyn, Constance Crozier, and Malcolm McCulloch. Improving the scalability of a prosumer cooperative game with k-means clustering. In *2019 IEEE Milan PowerTech*, pages 1–6, 2019.

[15] Liyang Han, Thomas Morstyn, and Malcolm McCulloch. Incentivizing prosumer coalitions with energy management using cooperative game theory. *IEEE Transactions on Power Systems*, 34(1):303 – 313, January 2019.

[16] Liyang Han, Thomas Morstyn, and Malcolm D. McCulloch. Scaling up cooperative game theory-based energy management using prosumer clustering. *IEEE Transactions on Smart Grid*, 12(1):289–300, 2021.

[17] Stephen Keeling and Karl Kunisch. Robust $\ell_1$ approaches to computing the geometric median and principal and independent components. *Journal of Mathematical Imaging and Vision*, 56:pg.6, 02 2016.

[18] P.R. Kersten. Fuzzy order statistics and their application to fuzzy clustering. *IEEE Transactions on Fuzzy Systems*, 7(6):708–712, 1999.

[19] Young-Il Kim, Jong-Min Ko, and Seung-Hwan Choi. Methods for generating tlps (typical load profiles) for smart grid-based energy programs. In *2011 IEEE Symposium on Computational Intelligence Applications In Smart Grid (CIASG)*, pages 1–6, 2011.

[20] Konstantin Makarychev and Liren Shan. Near-optimal algorithms for explainable k-medians and k-means, 2021.

[21] Yury Makarychev. k-means and k-medians under dimension reduction. https://www.youtube.com/watch?v=0yz3_1AmHgw, 2018.

[22] Mihai Manea. Cooperative games. https://ocw.mit.edu/courses/economics/14-126-game-theory-spring-2016/lecture-notes/MIT14_126S16_cooperative.pdf, January 2017.

[23] Matteo Matteucci. A tutorial on clustering algorithms: Fuzzy c-means clustering. https://matteucci.faculty.polimi.it/Clustering/tutorial_html/cmeans.html.

[24] Thomas Morstyn, Niall Farrell, Sarah Darby, and Malcolm Mcculloch. Using peer-to-peer energy-trading platforms to incentivize prosumers to form federated power plants. *Nature Energy*, 3, 02 2018.

[25] Thomas Morstyn and Malcolm D. McCulloch. Multi-class energy management for peer-to-peer energy trading driven by prosumer preferences. *IEEE Transactions on Power Systems*, 34(5):4005 – 4014, Sept 2019.

[26] Thomas Morstyn, Alexander Teytelboym, and Malcolm D. McCulloh. Bilateral contract networks for peer-to-peer energy trading. *IEEE Transactions on Smart Grid*, 10(2):2026–2035, March 2019.

[27] Sebastian Neumayer, Max Nimmer, Simon Setzer, and Gabriele Steidl. On the robust pca and weiszfeld's algorithm, 2019.

[28] NREL. Pvwatts calculation. https://pvwatts.nrel.gov/pvwatts.php.

[29] Amin Rajabi, Mohsen Eskandari, M. Jabbari Ghadi, Li Li, Jiangfeng Zhang, and Pierluigi Siano. A comparative study of clustering techniques for electrical load pattern segmentation. *Renewable and Sustainable Energy Reviews*, 120:109628, 12 2019.

[30] Customer-Led Network Revolution. Dataset (tc5): Enhanced profiling of domestic customers with solar photovoltaics (pv). http://www.networkrevolution.co.uk/project-library/dataset-tc5-enhanced-profiling-solar-photovoltaic-pv-users/, 2014.

[31] Customer-Led Network Revolution. Dataset (tc5): Enhanced profiling of domestic customers with solar photovoltaics (pv). http://www.networkrevolution.co.uk/project-data-download/?dl=TC5.zip#, 2014.

[32] Frederic Ros and Serge Guillaume. A hierarchical clustering algorithm and an improvement of the single linkage criterion to deal with noise. *Expert Systems with Applications*, 128, 03 2019.

[33] W. Saad, Z. Han, M. Debbah, A. Hjorungnes, and T. Basar. Coalitional game theory for communication networks. *IEEE Signal Processing Magazine*, 26(5):77–97, 2009.

[34] Walid Saad, Zhu Han, Merouane Debbah, Are Hjorungnes, and Tamer Basar. Coalitional game theory for communication networks. *IEEE Signal Processing Magazine*, 26(5):77–97, 2009.

[35] Manuel Saldarriaga. Por primera vez, barrio de medellín le venderá energía solar al país. https://www.elcolombiano.com/medellin/barrio-de-medellin-vendera-energia-solar-al-pais-DE21164968. El Colombiano. Thesis cover picture.

[36] Etienne Sorin, Lucien Bobo, and Pierre Pinson. Consensus-based approach to peer-to-peer electricity markets with product differentiation. *IEEE Transactions on Power Systems*, 34(2):994 – 1004, March 2019.

[37] David R. Thompson. 8.2 david thompson (part 2): Nearest neighbors and the curse of dimensionality. https://www.youtube.com/watch?v=dZrGXYty3qc.

[38] Peter van der Wilt. Salderen en terugleververgoeding zonnepanelen. https://www.consumentenbond.nl/zonnepanelen/salderen-en-terugleververgoeding-zonnepanelen, December 2019.

[39] Benjamin van Roy and Kahn Mason. Formulation and analysis of linear programs. https://web.stanford.edu/~ashishg/msande111/notes/chapter4.pdf, September 2005. Introduction to Optimization - Lecture notes [MS&E 111/ENGR 62], pg. 13/28.

[40] Kilian Weinberger. Lecture 4 "curse of dimensionality / perceptron" - cornell cs4780 sp17. https://www.youtube.com/watch?v=BbYV8UfMJSA, 2018.

[41] Christopher Whelan, Gregory K. Harrell, and Jin Wang. Understanding the k-medians problem. 2015.

[42] Kaile Zhou, Shanlin Yang, and Zhen Shao. Household monthly electricity consumption pattern mining: A fuzzy clustering-based model and a case study. *Journal of Cleaner Production*, 141:900–908, 2017.

# Glossary

## List of Acronyms

| | |
|---|---|
| **PV** | Photovoltaics |
| **ES** | Energy Storage |
| **P2P** | Peer-to-Peer |
| **DER** | Distributed Energy Resources |
| **PCA** | Principal Component Analysis |