User guidelines
Uncertainty modeling via polyhedral uncertainty sets for energy

Julien Vaes [*]   Vassilis M. Charitopoulos [†]

June, 2023

1 Introduction

This document supports the paper Vaes and Charitopoulos (2022) entitled *A data-driven uncertainty modelling and reduction approach for energy optimisation problems.*

This document details the implementation to generate uncertainty based on polyhedral uncertainty sets that can be found here on Github.

2 Robust optimisation problem.

We consider the following scenario-based linear adaptive robust optimisation (ARO) problem, which is recurrently used to formulate energy problems (*e.g.* unit commitment):

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimise}} \quad & \boldsymbol{c}^{\mathrm{T}} \boldsymbol{x} + \sum_{k \in \mathcal{K}} p_k \left(\max_{\boldsymbol{u}_k \in \mathcal{U}_k} \min_{\boldsymbol{y}_k} \boldsymbol{b}_k^{\mathrm{T}} \boldsymbol{y}_k, \right) && \text{(1a)} \\ \text{subject to} \quad & \mathbf{A}\, \boldsymbol{x} \leq \boldsymbol{d}, && \text{(1b)} \\ & \forall k \in \mathcal{K} : \boldsymbol{h}_k - \mathbf{T}_k\, \boldsymbol{x} - \mathbf{M}_k\, \boldsymbol{u}_k \leq \mathbf{W}_k\, \boldsymbol{y}_k, && \text{(1c)} \end{aligned}$$

where $\boldsymbol{x}$ and $\boldsymbol{y}_k$ with $k \in \mathcal{K}$ are respectively the first and second stage variables, $\mathcal{K}$ is the set of distinct operation conditions/scenarios, $\mathcal{U}_k$ is the uncertainty set associated to the uncertain parameters in the operation conditions $k \in \mathcal{K}$, and p_k is the weight (or probability of occurrence) associated to scenario k.

3 Data driven uncertainty set generation

In this section, we explain more deeply the method we previously presented in Vaes and Charitopoulos (2022) to generate polyhedral uncertainty sets in the case of data scarcity: we assume here w.l.o.g. that uncertainty corresponds to the daily profiles

[*] Department of Chemical Engineering, The Sargent Centre for Process Systems Engineering, University College London, Torrington Place, London WC1E 7JE, UK (j.vaes@ucl.ac.uk).

[†] Department of Chemical Engineering, The Sargent Centre for Process Systems Engineering, University College London, Torrington Place, London WC1E 7JE, UK (v.charitopoulos@ucl.ac.uk).

(hourly values) of $n \in \mathbb{N}$ attributes. A realisation of uncertainty of a given day i is consequently a vector $\boldsymbol{u}^{(i)} \in \mathbb{R}^{24n}$ defined as the concatenation of the daily profile $\boldsymbol{u}^{(i,k)} \in \mathbb{R}^{24}$ of each attribute $k \in [\![1, n]\!]$:

$$\boldsymbol{u}^{(i)} := \begin{bmatrix} \boldsymbol{u}^{(i,1)} \\ \vdots \\ \boldsymbol{u}^{(i,k)} \\ \vdots \\ \boldsymbol{u}^{(i,n)} \end{bmatrix} \tag{2}$$

Let $\boldsymbol{u}^{(i)} \in \mathbb{R}^{24n}$, $i \in [\![1, m]\!]$, be m historical data points. Given a data point $\boldsymbol{u}^{(i)} \in \mathbb{R}^{24n}$, we denote by $\mathbf{U}^{(i)} \in \mathbb{R}^{n\times 24}$ the matrix whose rows correspond to the daily profiles of each attribute:

$$\mathbf{U}^{(i)} := \quad \begin{bmatrix} \boldsymbol{u}^{(i,1)} & \dots & \boldsymbol{u}^{(i,k)} & \dots & \boldsymbol{u}^{(i,n)} \end{bmatrix}^{\mathrm{T}}. \tag{3}$$

As we will see later on, this second notation eases the expression of the continuity constraints when we desire to generate a succession of daily profiles (see Section 4). The collection of all data points is noted $\boldsymbol{u} := \left\{\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(m)}\right\}$ or $\mathbf{U} := \left\{\mathbf{U}^{(1)}, \dots, \mathbf{U}^{(m)}\right\}$.

As an initial step, based a correlation analysis (*e.g.* by using `pandas.DataFrame.corr` in Python), we define a set $\mathcal{A}$ of disjoint groups of attributes such that the data of the attributes inside a group are strongly correlated while being weakly correlated with any attribute of another group. The idea is then to separately generate an uncertainty set for each group $a \in \mathcal{A}$ of attributes as if they were independent. The (weak) correlation between each group will in a second stage be captured by defining scenarios, which are defined as the combination of specific group-scenario for each group of attributes.

We now present the steps needed for generating a PUS based on a lower dimensional space for each group $a \in \mathcal{A}$. For every group $a \in \mathcal{A}$, we denote by $\boldsymbol{u}_a := \{\boldsymbol{u}_a^1, \dots, \boldsymbol{u}_a^m\}$ the collection of the m data points related to the n_a attributes associated to a. A data point in group a is thus an element of $\mathbb{R}^{24n_a}$. First, we should standardise the data of each uncertainty component such that it has zero mean and unit variance. The standardised data points are denoted with $\boldsymbol{v}_a^{(i)}$, $i \in [\![1, m]\!]$, and are computed as follows

$$\boldsymbol{v}_a^{(i)} := \frac{1}{\boldsymbol{\sigma}_a} \circ \left(\boldsymbol{u}_a^{(i)} - \boldsymbol{\mu}_a\right), \tag{4}$$

where $\boldsymbol{x} \circ \boldsymbol{y}$ defines the Hadamard product between $\boldsymbol{x}$ and $\boldsymbol{y}$, where $\boldsymbol{\mu}_a$ and $\boldsymbol{\sigma}_a$ are respectively vectors in $\mathbb{R}^{24n_a}$ where each element k corresponds to the sample mean and standard deviation of the component k over all m data points, *i.e.* of the vector $\left[u_a^{(1)}[k], \dots, u_a^{(m)}[k]\right]^{\mathrm{T}}$, and where $\frac{1}{\boldsymbol{\sigma}_a}$ should be understood componentwise. This standardization step is performed to associate an equal weight to each uncertainty component.

Next, we perform a principal component analysis (PCA) and express the data in the PCA basis: this allows for finding the directions along which the data has greatest variance. Mathematically, there exists an orthogonal matrix $\mathbf{P}_a \in \mathbb{R}^{24n_a\times 24n_a}$ such that, any data point $\boldsymbol{v}_a^{(i)}$, $i \in [\![1, m]\!]$, expressed in the original basis can be expressed in the PCA basis using the following linear transformation:

$$\boldsymbol{w}_a^{(i)} = \mathbf{P}_a\, \boldsymbol{v}_a^{(i)}. \tag{5}$$

As $\mathbf{P}_a$ is an orthogonal matrix, the linear transformation from the PCA basis to the original basis is given by $\boldsymbol{v}_a^{(i)} = \mathbf{P}_a^{\mathrm{T}}\, \boldsymbol{w}_a^{(i)}$. Given the data expressed in the PCA basis, we perform dimensionality reduction by retaining only the first few components that explain the most the variability inside the data. This can for instance be enforced by defining a threshold for the contribution in the explained variance ratio under which components would be ignored (see `explained_variance_ratio_` associated to `PCA` in Python). Let $r_a \leq 24n_a$ denote the number of components retained. As the data is strongly correlated inside any given group $a \in \mathcal{A}$, we expect to have $r_a \ll 24n_a$. Let $\bar{\boldsymbol{w}}_a^{(i)} \in \mathbb{R}^{r_a}$ denote the data points in the truncated PCA basis, where only the first r_a components are kept, *i.e.*

$$\bar{\boldsymbol{w}}_a^{(i)} \coloneqq \boldsymbol{w}_a^{(i)}[1 : r_a]. \tag{6}$$

We can define a linear function from $\mathbb{R}^{r_a}$ to $\mathbb{R}^{24n_a}$, which maps any point $\bar{\boldsymbol{w}}_a$ in the truncated PCA basis to a point in the original standardised basis as $\bar{\boldsymbol{w}}_a \mapsto \bar{\mathbf{P}}_a^T \bar{\boldsymbol{w}}_a$, where $\bar{\mathbf{P}}_a \coloneqq \mathbf{P}_a[1 : r_a, :] \in \mathbb{R}^{r_a \times 24n_a}$.

To define the typical operational conditions associated to a group $a \in \mathcal{A}$ of attributes, we perform clustering (*e.g.* K-means) on the data points expressed in the truncated PCA basis. Clustering in the PCA basis rather than in the original basis allows to give relatively more importance to the directions explaining the most the variance and obtain more meaningful clusters (Ding and He, 2004). Let $\mathcal{K}_a$ denote the set of clusters related to $a \in \mathcal{A}$. We call each cluster $k_a \in \mathcal{K}_a$ a *group-scenario* and define its probability of occurrence p_{k_a} as the proportion of data points attributed to this cluster.

We now follow a similar approach as Ning and You (2018) to construct a polyhedral uncertainty set for each group-scenario $k_a \in \mathcal{K}_a$. To this end, we estimate the marginal probability density function $\hat{f}_{a,k_a,r}$ along every principal component direction $\boldsymbol{d}_{a,r}$ with $r \in \llbracket 1, r_a \rrbracket$. This can be done using the empirical distribution function or a more convoluted approach such as a kernel smoothing method (*e.g.* KDE) (Chen, 2017). To this estimated marginal pdf, we denote by $\hat{F}_{a,k_a,r}$ the associated cdf, and by $\hat{F}^{-1}_{a,k_a,r}$ the quantile function, *i.e.*

$$\hat{F}^{-1}_{a,k_a,r}(\alpha) = \min \left\{ \xi \in \mathbb{R} \,\middle|\, \hat{F}_{a,k_a,r}(\xi) \geq \alpha \right\}. \tag{7}$$

We now define the polyhedral uncertainty set (PUS) related to the group of attributes $a \in \mathcal{A}$ and the cluster $k_a \in \mathcal{K}_a$, which we call a *group-scenario PUS*, as follows:

$$\bar{\mathcal{W}}^{pol}_{a,k_a} \coloneqq \left\{ \bar{\boldsymbol{w}} \in \mathbb{R}^{r_a} \,\middle|\, \begin{array}{l} \mathbf{0} \leq \boldsymbol{z}^-, \boldsymbol{z}^+ \leq \mathbf{1}, \\ \mathbf{1}^{\mathrm{T}} \left(\boldsymbol{z}^- + \boldsymbol{z}^+ \right) \leq \Phi_{a,k_a} \\ \forall i, j \in \llbracket 1, s_{a,k_a} \rrbracket : z_i^- + z_i^+ + z_j^- + z_j^+ \leq \Psi_{a,k_a} \\ \boldsymbol{\xi}^{lb}_{a,k_a} = \left[\hat{F}^{-1}_{a,1,k_a}\left(\alpha_{a,k_a}\right), \dots, \hat{F}^{-1}_{a,r_a,k_a}\left(\alpha_{a,k_a}\right) \right]^{\mathrm{T}} \\ \boldsymbol{\xi}^{ub}_{a,k_a} = \left[\hat{F}^{-1}_{a,k_a,1}\left(1 - \alpha_{a,k_a}\right), \dots, \hat{F}^{-1}_{a,k_a,r_a}\left(1 - \alpha_{a,k_a}\right) \right]^{\mathrm{T}} \\ \boldsymbol{\lambda} = \frac{1}{2}(\boldsymbol{z}^+ - \boldsymbol{z}^- + \mathbf{1}) \\ \bar{\boldsymbol{w}} = \boldsymbol{\xi}^{lb}_{a,k_a} \circ (1 - \boldsymbol{\lambda}) + \boldsymbol{\xi}^{ub}_{a,k_a} \circ \boldsymbol{\lambda} \end{array} \right\}, \tag{8}$$

where $\mathbf{0}$ and $\mathbf{1}$ are vectors of respectively zeros and ones of size r_a, and where vectors inequalities must be understood componentwise. This set is similar to the gamma uncertainty set proposed by (Bertsimas and Sim, 2004), which enables to control the degree of conservatism: the larger the volume of the PUS, the more conservative is the resulting robust optimisation problem (Bertsimas and Sim, 2004). The uncertainty set is parametrised by α_{a,k_a}, Φ_{a,k_a} and Ψ_{a,k_a}. First, α_{a,k_a} is used to exclude both tails of the marginal pdf along each principal component axis $\boldsymbol{d}_{a,r}$. Then, Φ_{a,k_a} limits the cumulative dispersion from the nominal value along all retained PCA axes. Finally, the parameter Ψ_{a,k_a} additionally limits the pairwise dispersion along the first s_{a,k_a} PCA components. This parameter is important to exclude unlikely data points from the uncertainty that would otherwise not be cut with the general budget constraint parametrised by Φ_{a,k_a}.

We have so far generated a PUS for each group-scenario $k_a \in \mathcal{K}_a$ for each $a \in \mathcal{A}$. Let $\mathcal{K}^\times := \bigtimes_{a\in\mathcal{A}} \mathcal{K}_a$ denote the Cartesian product of the sets of group scenarios. The probability $\hat{p}_k$ associated to $k \in \mathcal{K}^\times$ is estimated as the proportion of the data points such that the uncertainty part related to a is attributed to cluster k_a for all $a \in \mathcal{A}$. To take into account the weak correlation between groups of attributes (so far assumed to be independent), we retain only the most probable combinations of group-scenarios $k := \{k_a \in \mathcal{K}_a, a \in \mathcal{A}\} \in \mathcal{K}^\times$, keeping those with associated probability greater than a threshold probability $\tilde{p}$. The more conservative we desire to be, the greater the acceptance probability threshold $\tilde{p}$ should be. The set of scenarios $\mathcal{K}$ is then constituted of the combinations in $\mathcal{K}^\times$ satisfying the probability threshold $\tilde{p}$. The probability of occurrence p_k of each scenario $k \in \mathcal{K}$ used in (1a) is then defined as the scaled probability when the scenarios that do not satisfy the acceptance threshold are excluded, *i.e.* $p_k = \hat{p}_k / \sum_{k\in\mathcal{K}} \hat{p}_k$.

Finally, the polyhedral uncertainty set in the original basis related to a scenario $k = \{k_a \in \mathcal{K}_a, a \in \mathcal{A}\} \in \mathcal{K}$ is then defined as follows:

$$\mathcal{U}_k := \left\{ \boldsymbol{u} \in \mathbb{R}^{24n} \;\middle|\; \forall a \in \mathcal{A} : \begin{cases} \bar{\boldsymbol{w}}_a \in \bar{\mathcal{W}}^{pol}_{a,k_a} \\ \boldsymbol{v}_a = \bar{\mathbf{P}}_a^T \bar{\boldsymbol{w}}_a \\ \boldsymbol{u}_a = \boldsymbol{\sigma}_a \circ \boldsymbol{v}_a + \boldsymbol{\mu}_a, \end{cases} \right\}. \tag{9}$$

To conclude, the method presented in this paper to generate scenario-based polyhedral uncertainty sets from historical data is summarised as follows:

- **Step 1**: Collect historical data points $\left\{\boldsymbol{u}^{(i)}, i \in [\![1, m]\!]\right\}$.
- **Step 2**: Derive a set $\mathcal{A}$ of disjoint groups of attributes such that the attributes inside a group are weakly correlated with any attribute of another group.
- **Step 3**: For each group $a \in \mathcal{A}$:

  (a) Standardise the data according to (4).

  (b) Perform a principal component analysis and express the data in the PCA basis according to (5).

  (c) Dimensionality reduction: retain the r_a first few principal components which explain the most the variability in the data, (see Equation (6)).

  (d) Derive a set $\mathcal{K}_a$ of group-scenarios for the group of attributes using a clustering technique (*e.g.* K-means) on the truncated PCA data $\bar{\boldsymbol{w}}_a$.

  (e) For each cluster/group-scenario $k_a \in \mathcal{K}_a$, estimate the pdf, cdf and quantile function of the data along each retained PCA axis $\boldsymbol{d}_{a,r}$. $r \in [\![1, r_a]\!]$ (*e.g.* KDE).

  (f) Define the polyhedral uncertainty set according to (8) for the group of attributes $a \in \mathcal{A}$ and the group-scenario $k_a \in \mathcal{K}_a$.
- **Step 4**: For any combination of group-scenarios, keep only those with associated probability w_k greater than $\tilde{p}$.

4 Uncertain set for several successive days

In some applications (*e.g.* unit commitment), we are interested in taking into account of the uncertainty over several days. Given the succession of m daily scenarios

$k_{1\to m} := (k_1,\dots,k_m) \in \mathcal{K} \times \dots \times \mathcal{K}$, we can define an associated PUS based on the definition of $\mathcal{U}_k$ in (9) as follows:

$$\mathcal{U}_{k_{1\to m}} := \left\{ \boldsymbol{u} \in \mathbb{R}^{24nm} \;\middle|\; \begin{cases} \boldsymbol{u} = [\boldsymbol{u}_1,\dots,\boldsymbol{u}_m] \\ \forall i \in [\![1,m]\!] : \boldsymbol{u}_i \in \mathcal{U}_{k_i} \\ \forall i \in [\![1,m-1]\!] : |(\mathbf{U}_{i+1}[:,1] - \mathbf{U}_i[:,24]) - \boldsymbol{\mu}_\Delta| \circ \frac{1}{\boldsymbol{\sigma}_\Delta} \leq c \end{cases} \right\}, \tag{10}$$

where $\mathbf{U}_i \in \mathbb{R}^{n\times 24}$ is the matrix representation of $\boldsymbol{u}_i$, where the inequality $\leq$ should be understood componentwise, and where $\boldsymbol{\mu}_\Delta$ and $\boldsymbol{\sigma}_\Delta$ are in $\mathbb{R}^n$ and correspond respectively to the sample mean and standard deviation of the difference between the first value and last value of the previous day in the daily profile for each n attributes. The third constraint in (10) enforces continuity between the daily profiles with the parameter $c > 0$ as it limits the step size between successive profiles.

5 Numerical example

We present here how the code available on GITHUB can be used in practice.

To do so, we provide an example on how to derive polyhedral uncertainty sets with the notebook `uncertainty_modeling.ipynb` available on Github.

We provide the notebook `reproducibility_paper_escape.ipynb` also available on Github that allows to reproduce the images of our paper Vaes and Charitopoulos (2022).

5.1 Step 0: Initialize the Python environment

In order to run scripts provide on our GITHUB page, it is necessary to create the appropriate Python environment.

To do so, we have include the file requirements.txt to create an environment based on this file, the steps should be followed:

1. Clone or download the project repository that contains the requirements.txt file.
2. Open a terminal or command prompt and navigate to the project directory.
3. (Optional) It is recommended to create a new virtual environment to isolate this project's dependencies from your system-wide Python installation. You can create a virtual environment using the following command:

   ```
   python3 -m venv myenv
   ```

   This will create a new directory called `myenv` (you can choose a different name if you prefer) that contains the necessary files for the virtual environment.
4. Activate the virtual environment:
   - In the terminal, run the appropriate command based on your operating system:
     - For Windows:

       ```
       myenv\Scripts\activate
       ```

     - For macOS/Linux:

       ```
       source myenv/bin/activate
       ```

- You will see `(myenv)` prefix in your terminal, indicating that the virtual environment is active.

5. Install the project dependencies using the `requirements.txt` file:

   ```
   pip install -r requirements.txt
   ```

   This command will read the `requirements.txt` file and install all the necessary packages and their versions into the current virtual environment.

6. Once the installation is complete, you can run your code within the activated virtual environment, and it will use the installed packages and versions specified in the `requirements.txt` file.

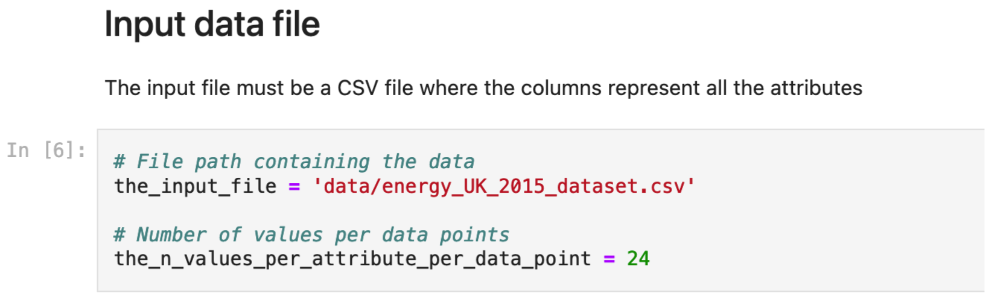
By following these steps, you will have recreated the environment based on the `requirements.txt` file, ensuring that all the required packages and versions are installed and ready to use.

5.2 Step 1: Collect historical data points

As an input, the code need two things (see Figure 1):

1. A file path to a `csv` where the data is as follows: each column corresponds to concatenated data points of an attribute (see Figure 2).

2. A `int` which corresponds to the number of values in a data points.



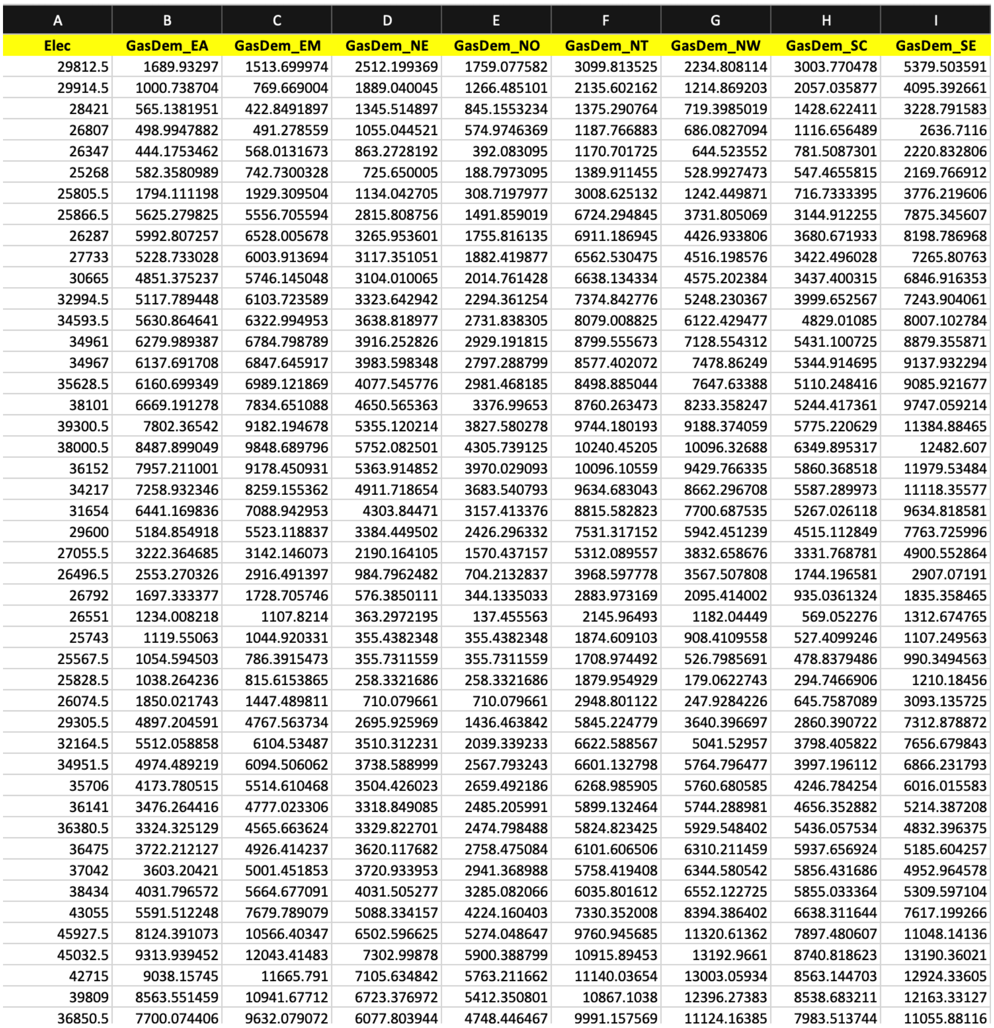
Input data file

The input file must be a CSV file where the columns represent all the attributes

In [6]:

```
# File path containing the data
the_input_file = 'data/energy_UK_2015_dataset.csv'

# Number of values per data points
the_n_values_per_attribute_per_data_point = 24
```

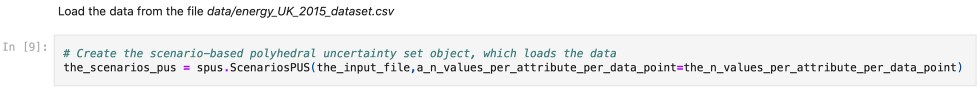
Figure 1



A	B	C	D	E	F	G	H	I
Elec	GasDem_EA	GasDem_EM	GasDem_NE	GasDem_NO	GasDem_NT	GasDem_NW	GasDem_SC	GasDem_SE
29812.5	1689.93297	1513.699974	2512.199369	1759.077582	3099.813525	2234.808114	3003.770478	5379.503591
29914.5	1000.738704	769.669004	1889.040045	1266.485101	2135.602162	1214.869203	2057.035877	4095.392661
28421	565.1381951	422.8491897	1345.514897	845.1553234	1375.290764	719.3985019	1428.622411	3228.791583
26807	498.9947882	491.278559	1055.044521	574.9746369	1187.766883	686.0827094	1116.656489	2636.7116
26347	444.1753462	568.0131673	863.2728192	392.083095	1170.701725	644.523552	781.5087301	2220.832806
25268	582.3580989	742.7300328	725.650005	188.7973095	1389.911455	528.9927473	547.4655815	2169.766912
25805.5	1794.111198	1929.309504	1134.042705	308.7197977	3008.625132	1242.449871	716.7333395	3776.219606
25866.5	5625.279825	5556.705594	2815.808756	1491.859019	6724.294845	3731.805069	3144.912255	7875.345607
26287	5992.807257	6528.005678	3265.953601	1755.816135	6911.186945	4426.933806	3680.671933	8198.786968
27733	5228.733028	6003.913694	3117.351051	1882.419877	6562.530475	4516.198576	3422.496028	7265.80763
30665	4851.375237	5746.145048	3104.010065	2014.761428	6638.134334	4575.202384	3437.400315	6846.916353
32994.5	5117.789448	6103.723589	3323.642942	2294.361254	7374.842776	5248.230367	3999.652567	7243.904061
34593.5	5630.864641	6322.994953	3638.818977	2731.838305	8079.008825	6122.429477	4829.01085	8007.102784
34961	6279.989387	6784.798789	3916.252826	2929.191815	8799.555673	7128.554312	5431.100725	8879.355871
34967	6137.691708	6847.645917	3983.598348	2797.288799	8577.402072	7478.86249	5344.914695	9137.932294
35628.5	6160.699349	6989.121869	4077.545776	2981.468185	8498.885044	7647.63388	5110.248416	9085.921677
38101	6669.191278	7834.651088	4650.565363	3376.99653	8760.263473	8233.358247	5244.417361	9747.059214
39300.5	7802.36542	9182.194678	5355.120214	3827.580278	9744.180193	9188.374059	5775.220629	11384.88465
38000.5	8487.899049	9848.689796	5752.082501	4305.739125	10240.45205	10096.32688	6349.895317	12482.607
36152	7957.211001	9178.450931	5363.914852	3970.029093	10096.10559	9429.766335	5860.368518	11979.53484
34217	7258.932346	8259.155362	4911.718654	3683.540793	9634.683043	8662.296708	5587.289973	11118.35577
31654	6441.169836	7088.942953	4303.84471	3157.413376	8815.582823	7700.687535	5267.026118	9634.818581
29600	5184.854918	5523.118837	3384.449502	2426.296332	7531.317152	5942.451239	4515.112849	7763.725996
27055.5	3222.364685	3142.146073	2190.164105	1570.437157	5312.089557	3832.658676	3331.768781	4900.552864
26496.5	2553.270326	2916.491397	984.7962482	704.2132837	3968.597778	3567.507808	1744.196581	2907.07191
26792	1697.333377	1728.705746	576.3850111	344.1335033	2883.973169	2095.414002	935.0361324	1835.358465
26551	1234.008218	1107.8214	363.2972195	137.455563	2145.96493	1182.04449	569.052276	1312.674765
25743	1119.55063	1044.920331	355.4382348	355.4382348	1874.609103	908.4109558	527.4099246	1107.249563
25567.5	1054.594503	786.3915473	355.7311559	355.7311559	1708.974492	526.7985691	478.8379486	990.3494563
25828.5	1038.264236	815.6153865	258.3321686	258.3321686	1879.954929	179.0622743	294.7466906	1210.18456
26074.5	1850.021743	1447.489811	710.079661	710.079661	2948.801122	247.9284226	645.7587089	3093.135725
29305.5	4897.204591	4767.563734	2695.925969	1436.463842	5845.224779	3640.396697	2860.390722	7312.878872
32164.5	5512.058858	6104.53487	3510.312231	2039.339233	6622.588567	5041.52957	3798.405822	7656.679843
34951.5	4974.489219	6094.506062	3738.588999	2567.793243	6601.132798	5764.796477	3997.196112	6866.231793
35706	4173.780515	5514.610468	3504.426023	2659.492186	6268.985905	5760.680585	4246.784254	6016.015583
36141	3476.264416	4777.023306	3318.849085	2485.205991	5899.132464	5744.288981	4656.352882	5214.387208
36380.5	3324.325129	4565.663624	3329.822701	2474.798488	5824.823425	5929.548402	5436.057534	4832.396375
36475	3722.212127	4926.414237	3620.117682	2758.475084	6101.606506	6310.211459	5937.656924	5185.604257
37042	3603.20421	5001.451853	3720.933953	2941.368988	5758.419408	6344.580542	5856.431686	4952.964578
38434	4031.796572	5664.677091	4031.505277	3285.082066	6035.801612	6552.122725	5855.033364	5309.597104
43055	5591.512248	7679.789079	5088.334157	4224.160403	7330.352008	8394.386402	6638.311644	7617.199266
45927.5	8124.391073	10566.40347	6502.596625	5274.048647	9760.945685	11320.61362	7897.480607	11048.14136
45032.5	9313.939452	12043.41483	7302.99878	5900.388799	10915.89453	13192.9661	8740.818623	13190.36021
42715	9038.15745	11665.791	7105.634842	5763.211662	11140.03654	13003.05934	8563.144703	12924.33605
39809	8563.551459	10941.67712	6723.376972	5412.350801	10867.1038	12396.27383	8538.683211	12163.33127
36850.5	7700.074406	9632.079072	6077.803944	4748.446467	9991.157569	11124.16385	7983.513744	11055.88116

Figure 2

All the computations are dealt with the object `ScenariosPUS` which is generated as in Figure 3 given the file path and the number of values per data points for each attribute:



Load the data from the file *data/energy_UK_2015_dataset.csv*

```
In [9]:
# Create the scenario-based polyhedral uncertainty set object, which loads the data
the_scenarios_pus = spus.ScenariosPUS(the_input_file,a_n_values_per_attribute_per_data_point=the_n_values_per_attribute_per_data_point)
```

Figure 3

5.3 Step 2: groups of attributes

Finding the groups of attributes must be done by analysing the correlation matrix (see Figure 4). Here we detect 3 groups of attributes:

1. **Seasonal** data: electricity demand, gas demand and temperature
2. **Solar** data
3. **Wind** data: on-shore and off-shore

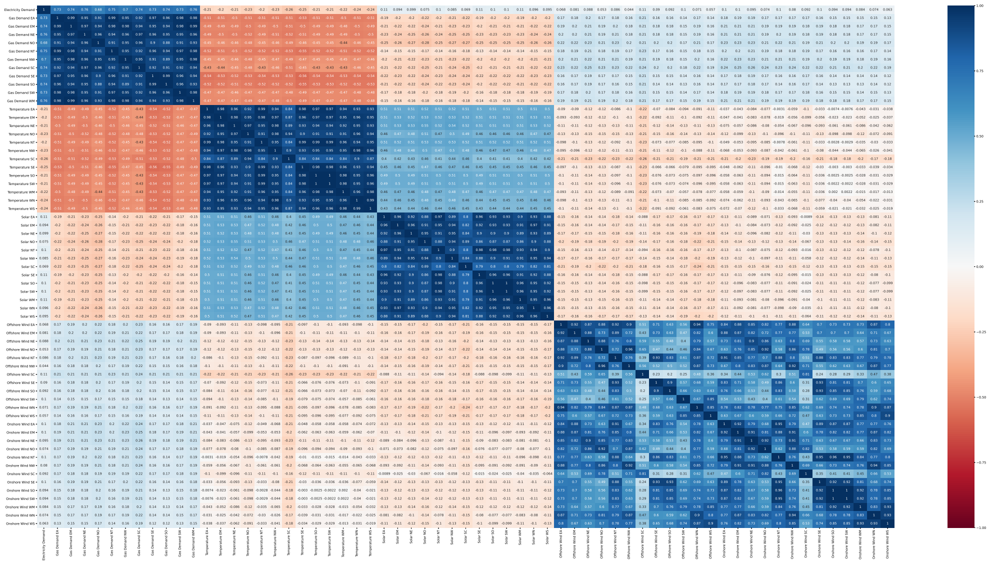


FIGURE 4

5.4 Step 3a → 3c: PCA reduction

To perform the PCA size reduction, we must provide which attributes should be merged together. Note that PCA works only if the number of data points is greater than the initial dimension. This explain why we perform here to successive PCA reduction for each of the group of attributes derived in **Step 2**.

Each sub-PCA reduction takes two parameters:

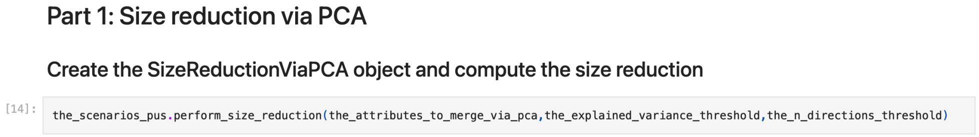
1. `explained_variance_threshold`: when performing PCA, it returns for each PCA axis the explained variance threshold. The first components explain the most variance inside the data. By fixed a threshold on the explained variance, we get rid of the axes that do no justify enough the variance in the data.

2. `n_directions_threshold`: this corresponds to an upper bound on the number of PCA axes that will be retained.

At least one of these parameters must be given, the other can be set to `None`.

Given these parameters, the PCA size reduction is performed with the function (see Figure 5):

```
perform_size_reduction(self,
                       a_attributes_to_merge,
                       a_explained_variance_threshold,
                       a_n_directions_threshold)
```



Part 1: Size reduction via PCA

Create the SizeReductionViaPCA object and compute the size reduction

```
[14]: the_scenarios_pus.perform_size_reduction(the_attributes_to_merge_via_pca,the_explained_variance_threshold,the_n_directions_threshold)
```

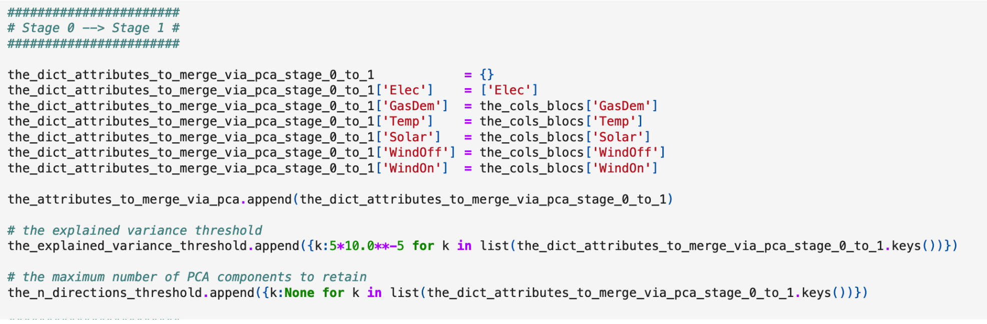
FIGURE 5

In our numerical example we have two steps in the PCA reduction:

Level 1:

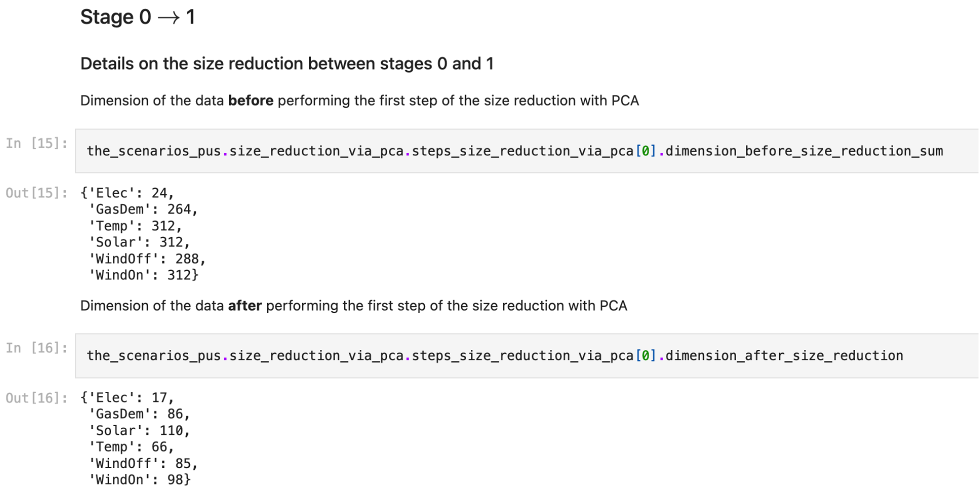
- **Elec**: Elec
- **GasDem**: GasDem_EA, . . . , GasDem_WM
- **GasDem**: GasDem_EA, . . . , GasDem_WM
- **Solar**: Solar_EA, . . . , Solar_WM
- **Temp**: Temp_EA, . . . , Temp_WM
- **WindOff**: WindOff_EA, . . . , WindOff_WM
- **WindOn**: WindOn_EA, . . . , WindOn_WM

For this level the parameters are set to the values `explained_variance_threshold` $= 5 \cdot 10^{-5}$ and `n_directions_threshold` = `None` (see Figure 6). Figure 7 shows the size reduction for each of the new attributes.



```
#######################
# Stage 0 --> Stage 1 #
#######################

the_dict_attributes_to_merge_via_pca_stage_0_to_1            = {}
the_dict_attributes_to_merge_via_pca_stage_0_to_1['Elec']    = ['Elec']
the_dict_attributes_to_merge_via_pca_stage_0_to_1['GasDem']  = the_cols_blocs['GasDem']
the_dict_attributes_to_merge_via_pca_stage_0_to_1['Temp']    = the_cols_blocs['Temp']
the_dict_attributes_to_merge_via_pca_stage_0_to_1['Solar']   = the_cols_blocs['Solar']
the_dict_attributes_to_merge_via_pca_stage_0_to_1['WindOff'] = the_cols_blocs['WindOff']
the_dict_attributes_to_merge_via_pca_stage_0_to_1['WindOn']  = the_cols_blocs['WindOn']

the_attributes_to_merge_via_pca.append(the_dict_attributes_to_merge_via_pca_stage_0_to_1)

# the explained variance threshold
the_explained_variance_threshold.append({k:5*10.0**-5 for k in list(the_dict_attributes_to_merge_via_pca_stage_0_to_1.keys())})

# the maximum number of PCA components to retain
the_n_directions_threshold.append({k:None for k in list(the_dict_attributes_to_merge_via_pca_stage_0_to_1.keys())})
```

Figure 6



Stage 0 → 1

Details on the size reduction between stages 0 and 1

Dimension of the data **before** performing the first step of the size reduction with PCA

```
In [15]:  the_scenarios_pus.size_reduction_via_pca.steps_size_reduction_via_pca[0].dimension_before_size_reduction_sum

Out[15]:  {'Elec': 24,
           'GasDem': 264,
           'Temp': 312,
           'Solar': 312,
           'WindOff': 288,
           'WindOn': 312}
```

Dimension of the data **after** performing the first step of the size reduction with PCA

```
In [16]:  the_scenarios_pus.size_reduction_via_pca.steps_size_reduction_via_pca[0].dimension_after_size_reduction

Out[16]:  {'Elec': 17,
           'GasDem': 86,
           'Solar': 110,
           'Temp': 66,
           'WindOff': 85,
           'WindOn': 98}
```

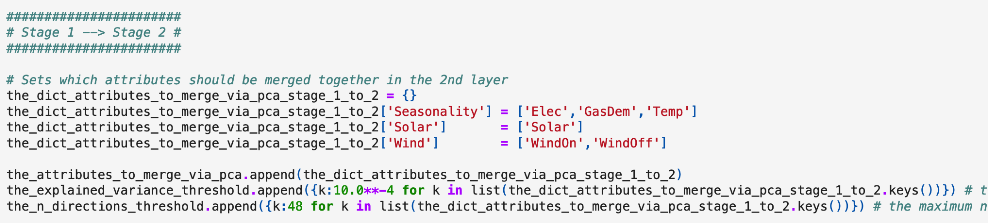
Figure 7

Level 2:

- **Seasonality**: Elec, GasDem, Temp
- **Solar**: Solar

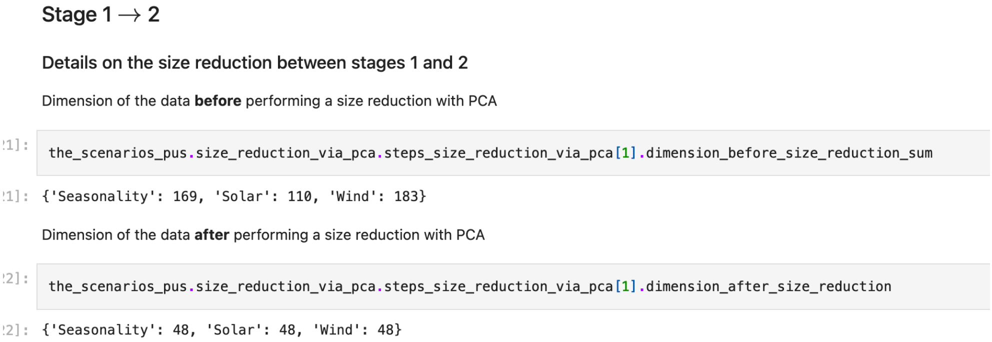
- **Wind**: WindOff, WindOn

For this level the parameters are set to the values `explained_variance_threshold` $= 10^{-4}$ and `n_directions_threshold` $=$ 48 (see Figure 8). Figure 9 shows the size reduction for each of the new attributes.



```
#######################
# Stage 1 --> Stage 2 #
#######################

# Sets which attributes should be merged together in the 2nd layer
the_dict_attributes_to_merge_via_pca_stage_1_to_2 = {}
the_dict_attributes_to_merge_via_pca_stage_1_to_2['Seasonality'] = ['Elec','GasDem','Temp']
the_dict_attributes_to_merge_via_pca_stage_1_to_2['Solar']       = ['Solar']
the_dict_attributes_to_merge_via_pca_stage_1_to_2['Wind']        = ['WindOn','WindOff']

the_attributes_to_merge_via_pca.append(the_dict_attributes_to_merge_via_pca_stage_1_to_2)
the_explained_variance_threshold.append({k:10.0**-4 for k in list(the_dict_attributes_to_merge_via_pca_stage_1_to_2.keys())}) # t
the_n_directions_threshold.append({k:48 for k in list(the_dict_attributes_to_merge_via_pca_stage_1_to_2.keys())}) # the maximum n
```

Figure 8



Stage 1 → 2

Details on the size reduction between stages 1 and 2

Dimension of the data **before** performing a size reduction with PCA

```
:1]: the_scenarios_pus.size_reduction_via_pca.steps_size_reduction_via_pca[1].dimension_before_size_reduction_sum

:1]: {'Seasonality': 169, 'Solar': 110, 'Wind': 183}
```

Dimension of the data **after** performing a size reduction with PCA

```
:2]: the_scenarios_pus.size_reduction_via_pca.steps_size_reduction_via_pca[1].dimension_after_size_reduction

:2]: {'Seasonality': 48, 'Solar': 48, 'Wind': 48}
```

Figure 9

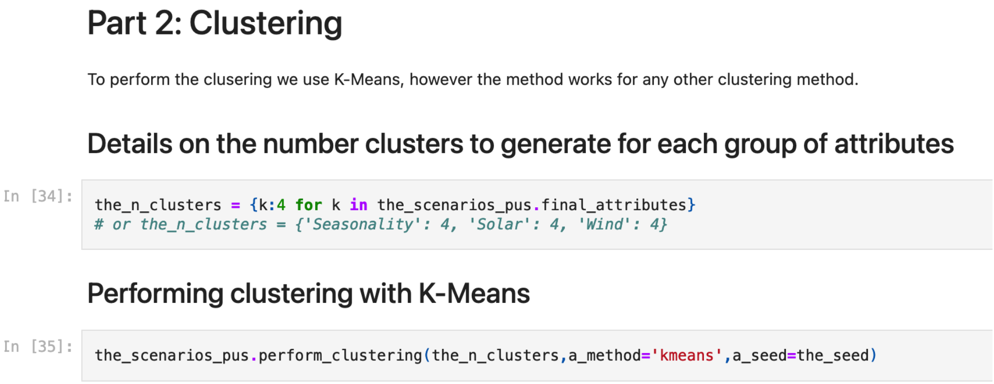
5.5 Step 3d: clustering

For each final attribute (in our example `Seasonality`, `Solar` and `Wind`) we specify the number of cluster that we want to generate (see Figure 10).

Then clustering is performed by calling the function:

```
perform_clustering(self,a_n_clusters,a_method='kmeans',a_seed=None)
```

Different methods are supported for the clustering: `'kmeans'`, `'gmm'`, and `'dpgmm'`.



Part 2: Clustering

To perform the clusering we use K-Means, however the method works for any other clustering method.

Details on the number clusters to generate for each group of attributes

```
In [34]: the_n_clusters = {k:4 for k in the_scenarios_pus.final_attributes}
         # or the_n_clusters = {'Seasonality': 4, 'Solar': 4, 'Wind': 4}
```

Performing clustering with K-Means

```
In [35]: the_scenarios_pus.perform_clustering(the_n_clusters,a_method='kmeans',a_seed=the_seed)
```

Figure 10

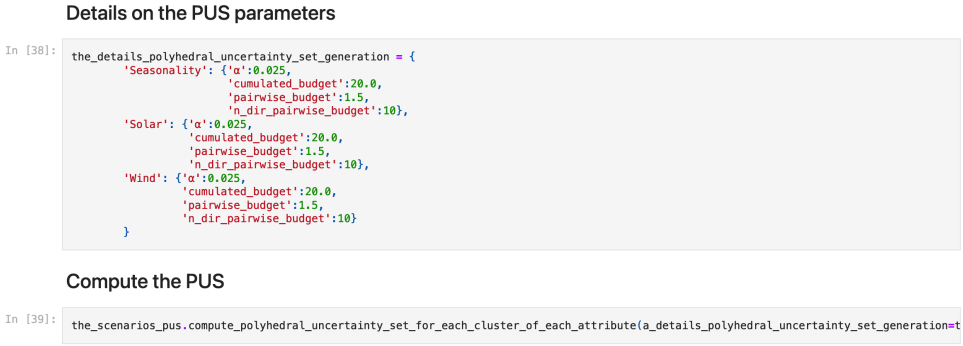
5.6 Step 3e → 3f: PUS of each cluster of each attribute

For each final attribute (in our example `Seasonality`, `Solar` and `Wind`) we must specify different parameters (see Figure 11):

- α: the tail percentage that is ignored on both side of the PCA axes.
- `cumulated_budget`: the cumulated dispersion budget with regard to the cluster mean
- `pairwise_budget`: the pairwise dispersion budget.
- `n_dir_pairwise_budget`: a number corresponding to the number of the first PCA axes that are concerned by the paired wise budget.

Then computation of the PUS for each cluster of each attribute is done by calling the function:

```
compute_polyhedral_uncertainty_set_for_each_cluster_of_each_attribute(self,
a_details_polyhedral_uncertainty_set_generation)
```



Details on the PUS parameters

In [38]:
```
the_details_polyhedral_uncertainty_set_generation = {
        'Seasonality': {'α':0.025,
                        'cumulated_budget':20.0,
                        'pairwise_budget':1.5,
                        'n_dir_pairwise_budget':10},
        'Solar': {'α':0.025,
                  'cumulated_budget':20.0,
                  'pairwise_budget':1.5,
                  'n_dir_pairwise_budget':10},
        'Wind': {'α':0.025,
                 'cumulated_budget':20.0,
                 'pairwise_budget':1.5,
                 'n_dir_pairwise_budget':10}
        }
```

Compute the PUS

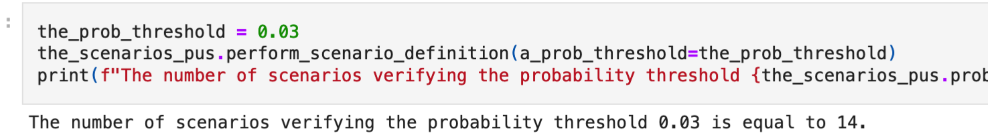
In [39]:
```
the_scenarios_pus.compute_polyhedral_uncertainty_set_for_each_cluster_of_each_attribute(a_details_polyhedral_uncertainty_set_generation=t
```

Figure 11

5.7 Step 4: scenario definition

Given a probability threshold the scenarios are obtained with the function:

```
perform_scenario_definition(self,a_prob_threshold,a_attributes=None)
```



```
the_prob_threshold = 0.03
the_scenarios_pus.perform_scenario_definition(a_prob_threshold=the_prob_threshold)
print(f"The number of scenarios verifying the probability threshold {the_scenarios_pus.prob
```

```
The number of scenarios verifying the probability threshold 0.03 is equal to 14.
```

Figure 12

6 Quering the result

The list of the scenarios generated can be obtained with the function `get_scenarios(self)`, *i.e.*

```
the_scenarios = the_scenarios_pus.get_scenarios()
```

Given a scenario (*e.g.* `the_scenar = the_scenarios[0]`, the first scenarios), its probability can be obtained with the function `get_probability`

```
the_scenar = the_scenarios[0]
the_proba = the_scenar.get_probability()
```

On the other hand, the details on the linear formulation for a realisation of uncertainty inside the scenario is obtained with the function `get_linear_constraints_for_optimisation`, *i.e.*

```
the_details = the_scenar.get_linear_constraints_for_optimisation()
```

This returns a dictionary for each final attribute (in our example for `Seasonality`, `Solar`, `Wind`).

```
the_details_seas = the_details['Seasonality']
```

The details associated to an attribute (*e.g.* `Seasonality`), is twofold:

1. Details on the linear inequality constraint that describe the PUS in the truncated PCA basis, *i.e.* `A_w`, `A_z` and `b`. All the $\boldsymbol{w}$ in $\bar{\mathcal{W}}^{pol}_{a,k_a}$ defined in Equation (8) are equivalent to all the $\boldsymbol{w}$ such that the following linear constraint hold:

$$\texttt{A\_w} \cdot \boldsymbol{w} + \texttt{A\_z} \cdot \boldsymbol{z} \leq \texttt{b}, \tag{11}$$

   with $\boldsymbol{w} \in \mathbb{R}^{r_a}$ and $\boldsymbol{z} \in \mathbb{R}^m$ with m the number of columns of `A_z`.

2. Details on the linear equality constraint that maps a point from the truncated PCA basis to the original uncertainty basis, *i.e.* `A_to_original_basis`, `b_to_original_basis` and `description_var_original_basis`. Given a point $\boldsymbol{w}$ in $\bar{\mathcal{W}}^{pol}_{a,k_a}$, we get the corresponding uncertainty in the original basis with the following linear constraint:

$$\boldsymbol{u} = \boldsymbol{w}^{\mathrm{T}} \cdot \texttt{A\_to\_original\_basis} + \texttt{b\_to\_original\_basis}. \tag{12}$$

   Note that as $\boldsymbol{w}$ is one of the final attribute, $\boldsymbol{u}$ is a vector with the uncertainty value of several initial attribute. The list `description_var_original_basis` details the vector $\boldsymbol{u}$. Each element is a tuple with a name of the initial attribute and the indices of $\boldsymbol{u}$ to which the data corresponds.

References

D. Bertsimas and M. Sim. The Price of Robustness. *Oper. Res.*, 52(1):35–53, 2004. doi:10.1287/opre.1030.0065.

Y. C. Chen. A tutorial on kernel density estimation and recent advances. *Biostat. Epidemiol.*, 1(1):161–187, 2017. doi:10.1080/24709360.2017.1396742.

C. Ding and X. He. K -means clustering via principal component analysis. In *Twenty-first Int. Conf. Mach. Learn. - ICML '04*, page 29. ACM Press, 2004. doi:10.1145/1015330.1015408.

C. Ning and F. You. Data-driven decision making under uncertainty integrating robust optimization with principal component analysis and kernel smoothing methods. *Comput. Chem. Eng.*, 112:190–210, 2018. doi:10.1016/j.compchemeng.2018.02.007.

J. Vaes and V. M. Charitopoulos. A data-driven uncertainty modelling and reduction approach for energy optimisation problems. arXiv, 2022. doi:10.48550/arXiv.2212.01478.