Clustering of metallic alloys based on chemical composition and thermophysical properties

Meradji Anais and **Gacem Aicha**

Abstract :

This project focuses on the clustering of metallic glass alloys based on their thermophysical properties and chemical composition.

The dataset contains the following features: **alloy composition**, **glass transition temperature** (Tg), **crystallization temperature** (Tx), **liquidus temperature** (Tl), and **maximum diameter** (Dmax).

After cleaning and preparing the data (applying different preprocessing steps that are necessary before creating any machine learning model) we standardize the numerical features and apply K-means clustering and Hierarchical clustering.

The goal is to group alloys with similar properties and compositions.

This approach will result in the discovery of patterns in the data and could support future research in predicting and designing new metallic glasses.

1. Data exploration

Alloys (compostion)	Tg(K)	Tx(K)	Tl(K)	Dmax (mm)
Ag30.8 Ca30.8 Mg23.1 Cu15.4	413	432	803	2,5
Ag38.4 Mg30.8 Ca30.8	394	426	805	0,5
Ag38.5 Ca30.8 Mg23 Cu7.7	384	416	854	2
Ag38.5 Mg38.5 Ca15.4 Cu7.7	405	436	842	0,5

Our data contains five features.

We notice that the first feature (alloys composition) isn't perfectly organized, making it not suitable for further analysis.

The first step we're going to do is to separate the chemical elements of every alloy.

1. Separation of alloys composition

In order to do that, we're going to use a **regular expression**.

```
pattern = r'([A-Z][a-z]*)(\s?[0-9]*\.?[0-9]+)'
```

This regular expression matches a chemical element symbol ([A-Z][a-z]*) followed by an optional space and a number (\s?[0-9]*\.?[0-9]+), which represents its percentage in the alloy.

Remark: The dataset contains two **invalid** chemical elements: **Mm** and **L**. We dropped both of these columns and we were left with 43 valid elements.

Result: The final dataset contains the extracted elements and their percentages along with the original data.

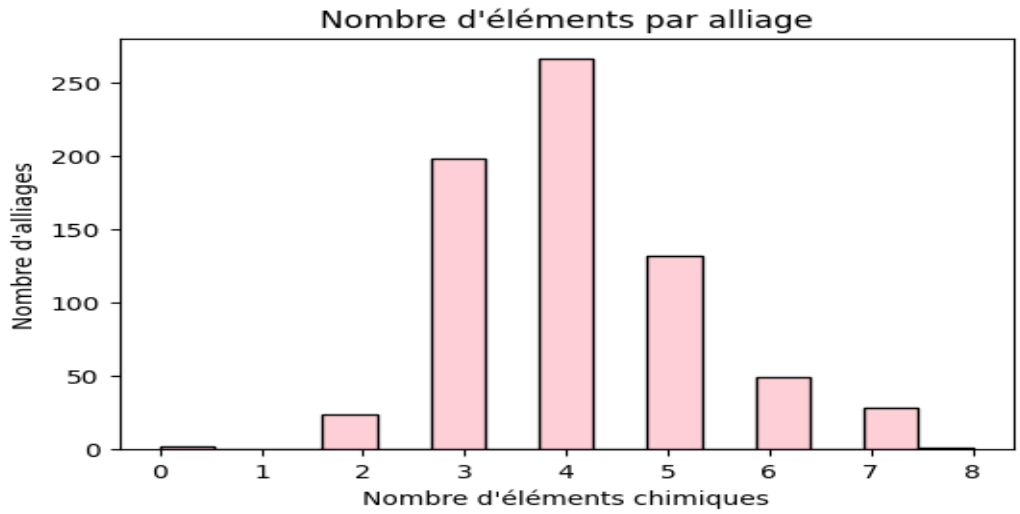
```
[ ]  print(data.columns)

Index(['Ag', 'Ca', 'Mg', 'Cu', 'Au', 'Si', 'Pd', 'Nd', 'Al', 'Ni', 'Fe', 'Zn',
       'Ce', 'Ga', 'Nb', 'La', 'Co', 'B', 'Ta', 'Cr', 'C', 'Mo', 'Zr', 'Ti',
       'Hf', 'In', 'Y', 'Sn', 'P', 'W', 'Er', 'Mn', 'Dy', 'Tb', 'Gd', 'Tm',
       'Ho', 'Pr', 'S', 'Be', 'Sc', 'Sm', 'V', 'Tg(K)', 'Tx(K)', 'Tl(K)',
       'Dmax (mm)'],
      dtype='object')
```

	Ag	Ca	Mg	Cu	Au	Si	Pd	Nd	Al	Ni	...	Pr	S	Be	Sc	Sm	V	Tg(K)	Tx(K)	Tl(K)	Dmax (mm)
0	30.8	30.8	23.1	15.4	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	413.0	432.0	803.0	2.5
1	38.4	30.8	30.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	394.0	426.0	805.0	0.5
2	38.5	30.8	23.0	7.7	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	384.0	416.0	854.0	2.0
3	38.5	15.4	38.5	7.7	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	405.0	436.0	842.0	0.5
4	46.2	30.7	23.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	399.0	426.0	765.0	0.7

2. Dataset exploration

- Shape of the dataset: (700, 47)



We notice that most alloys are made of 4 elements.

We notice that Copper, Iron and Zirconium are the most common elements composing our alloys.

Now that we know what we are working with, we can engage with the **data preprocessing** step which is a crucial step in order to clean up and transform our dataset.

2. Data preprocessing

1. Checking for duplicate rows

```
duplicated_rows = data[data.duplicated()]
print(len(duplicated_rows))
```

```
9
```

```
[ ] data.drop_duplicates(inplace=True)
```

```
[ ] duplicated_rows = data[data.duplicated()]
    print(len(duplicated_rows))
```

```
0
```

2. Checking for missing values

```
==================================================
ANALYSE DES VALEURS MANQUANTES
==================================================

1. Nombre de valeurs manquantes par colonne :
Tg(K)        2
Tx(K)        2
Tl(K)        2
Dmax (mm)    2
dtype: int64

2. Pourcentage de valeurs manquantes par colonne :
Tg(K)        0.29 %
Tx(K)        0.29 %
Tl(K)        0.29 %
Dmax (mm)    0.29 %
dtype: object
```

```
Colonne 'Tg(K)' imputée par la médiane (670.50)
Colonne 'Tx(K)' imputée par la médiane (731.00)
Colonne 'Tl(K)' imputée par la médiane (1145.00)
Colonne 'Dmax (mm)' imputée par la médiane (3.00)
```

We replaced the missing values with the median.

3. Checking for outliers

```
==================================================
RAPPORT DES VALEURS ABERRANTES
==================================================

Nombre d'outliers par élément (Z-score > 3):

La              43
Ca              33
Co              26
Ti              25
Ni              24
C               23
Ag              21
Zn              19
Ce              19
Si              18
Mo              17
Pd              16
Hf              16
Cr              13
Al              13
Ga              12
Sn              11
```

The outlier analysis (**Z-score > 3**) revealed that the elements La, Ca, and Co had the highest number of extreme values, with 43, 33, and 26 outliers respectively, indicating significant variability in their distributions.

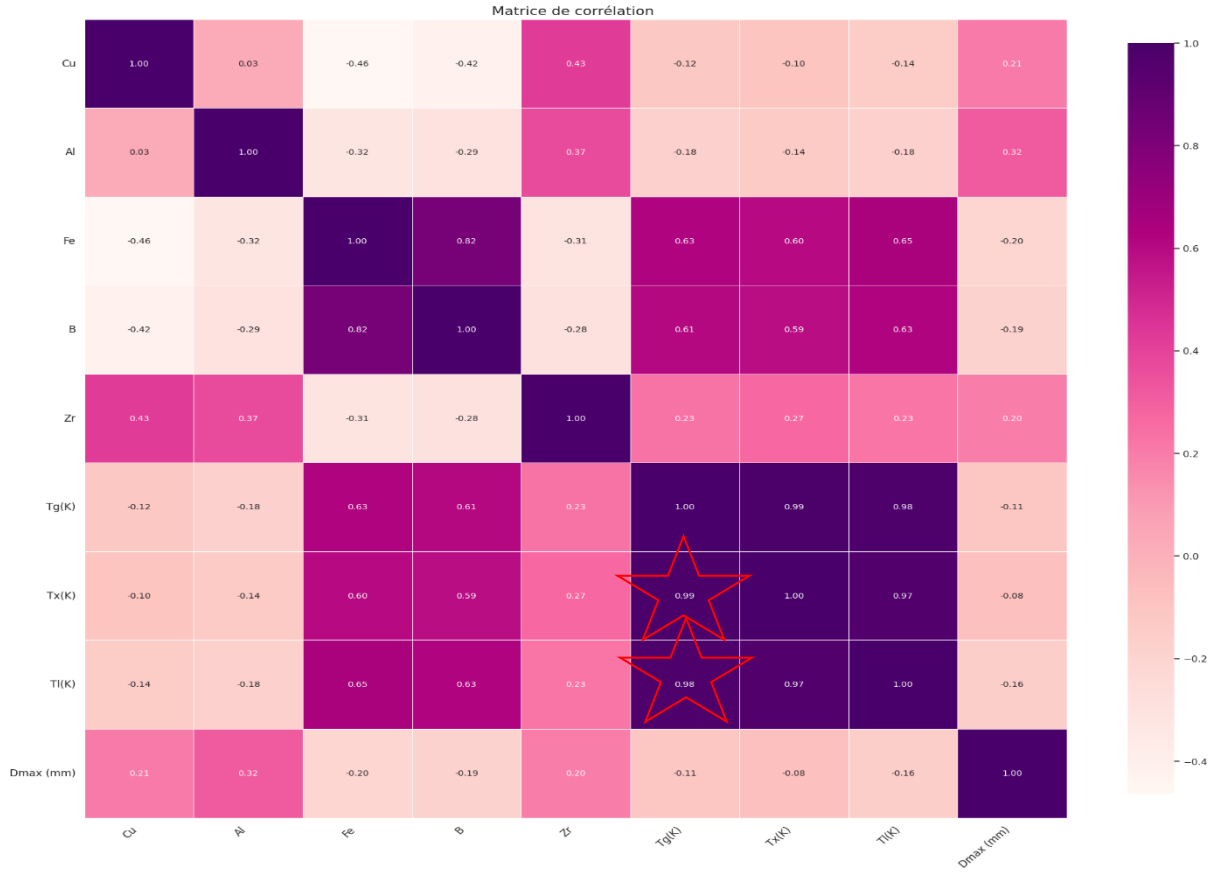
We're going to replace the outliers with the **median**.

4. Normalization of values

```
Normalisation Min-Max terminée. Aperçu :
         Ag        Ca        Mg        Cu   Au
0  0.500813  0.440000  0.256667  0.185542  0.0
1  0.624390  0.440000  0.342222  0.000000  0.0
2  0.626016  0.440000  0.255556  0.092771  0.0
3  0.626016  0.220000  0.427778  0.092771  0.0
4  0.751220  0.438571  0.256667  0.000000  0.0

Données normalisées sauvegardées dans 'BMGs_normalized.csv'
```

5. Checking for correlated features




```
Variables très corrélées  :
Tx(K) est fortement corrélé avec ['Tg(K)'] (corr > 0.9)
Tl(K) est fortement corrélé avec ['Tg(K)', 'Tx(K)'] (corr > 0.9)
```

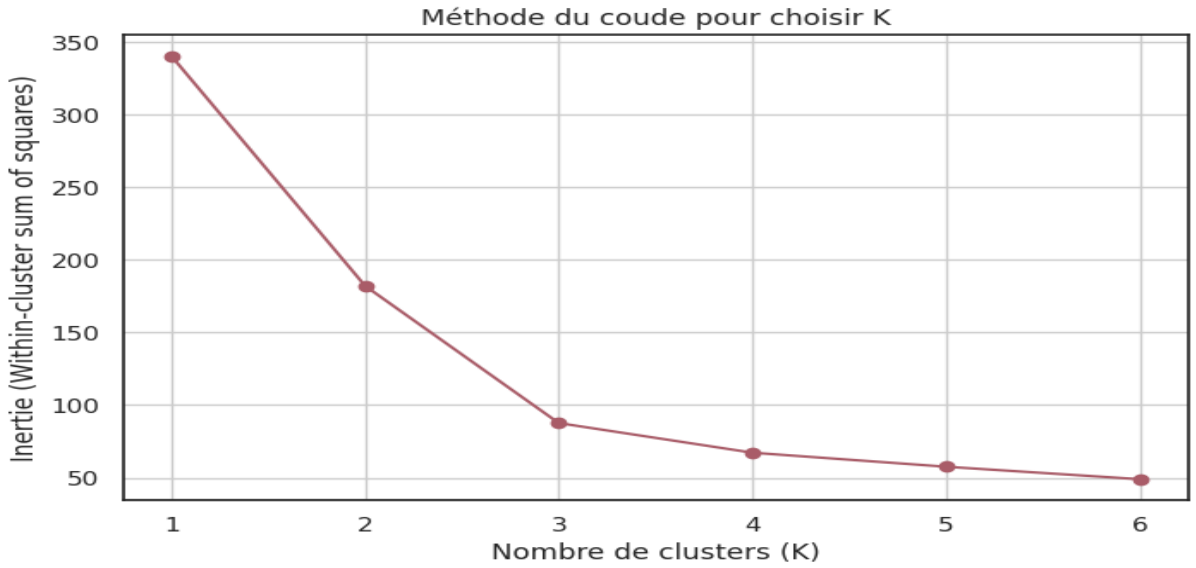
We decided to drop the column Tg(K) (**glass transition temperature)**

6. Clustering

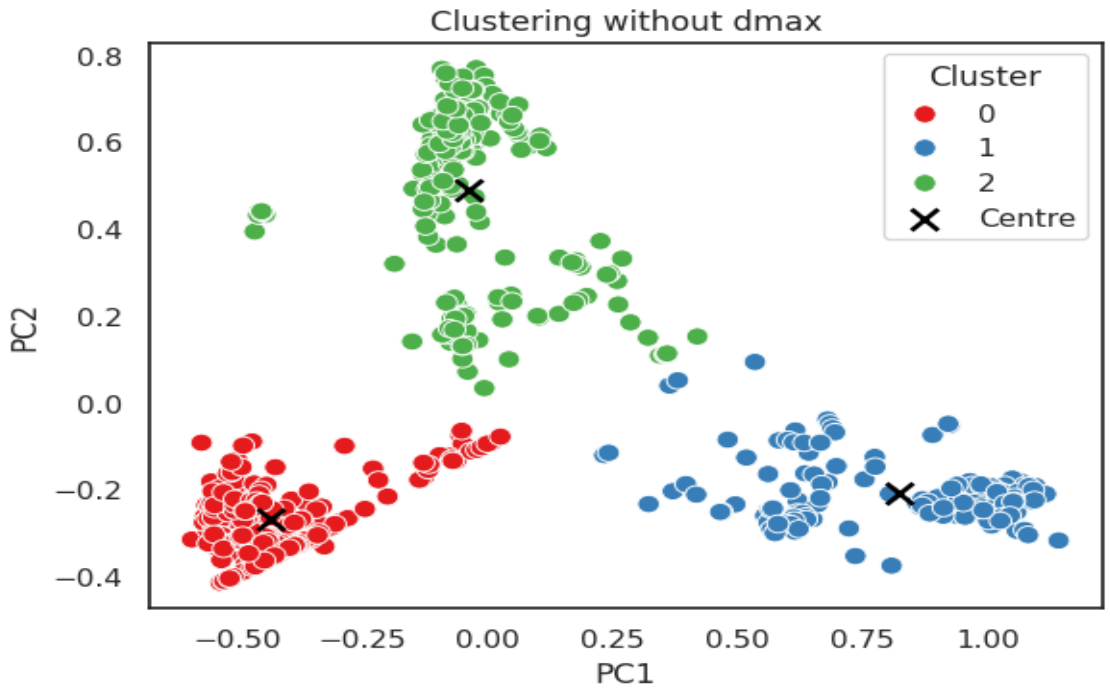
1. K- means without Dmax (**maximum diameter)**

K-means is a clustering algorithm used to partition a set of data points into groups (clusters) based on their similarities.

It aims to minimize the variance within each cluster by iteratively assigning data points to the nearest cluster centroid and recalculating centroids until convergence.

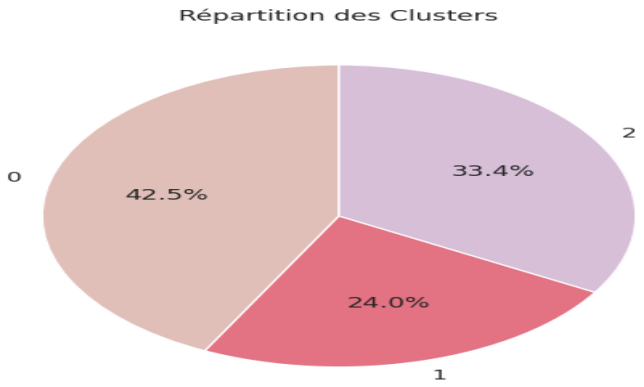


Observation: 3 seems to be the ideal number of clusters.



The clustering reveals three well-separated, compact groups.

Observation: all three clusters maintain distinct regions without significant overlap, indicating a solid clustering result.



Metrics :

```
K-means clustering without dMax scores:
---------------------------------
Silhouette Score : 0.739
Davies-Bouldin Index : 0.376
Calinski_harabasz score : 3118.523
```

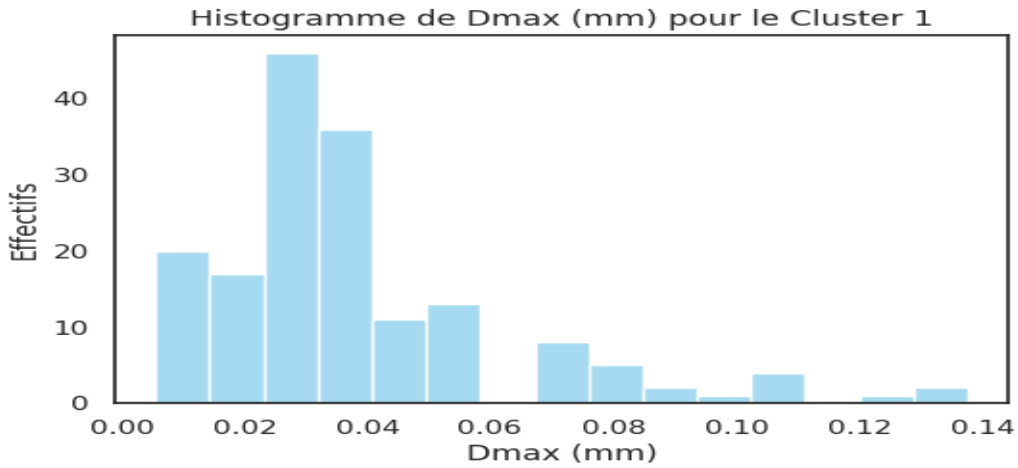
Silhouette score of 0.739 → good cluster separation

Davies-Bouldin index of 0.376 → reasonable cluster compactness

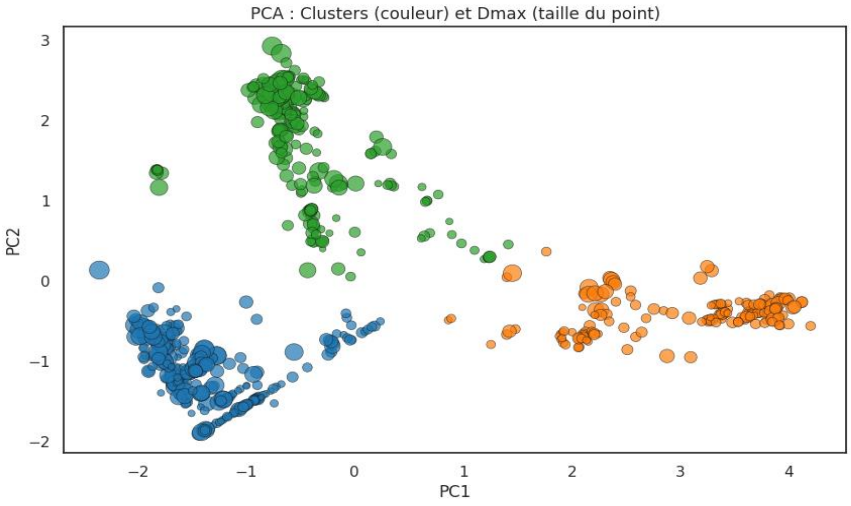
Calinski-Harabasz score of 3118.523 → well-defined cluster boundaries

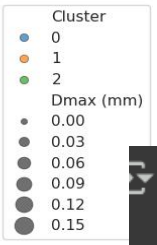
- Is there a link between the values of D max and the clustering results?

We are going to examine whether alloys with similar **Dmax** (maximum diameter) values are indeed grouped together in the same cluster.



Observation: the histogram shows that the vast majority of alloys in Cluster 1 have Dmax values concentrated between 0.01 and 0.05 mm.





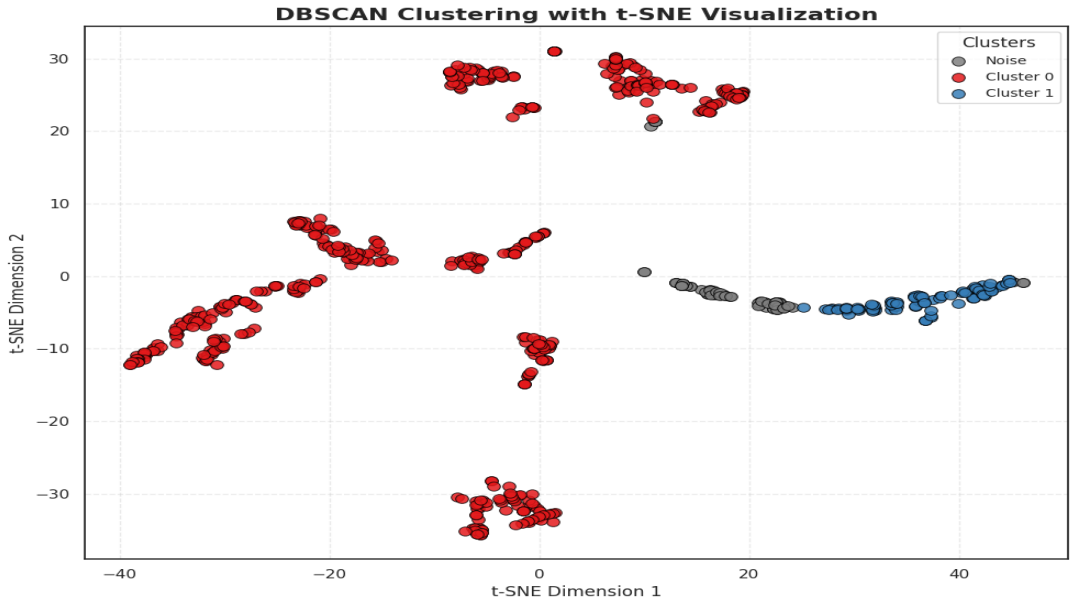
Cluster	Dmax (mm)
1	0.038604
0	0.054353
2	0.060745

The PCA plot shows that clusters correspond to distinct Dmax value ranges, with cluster 1 (orange) having the smallest Dmax values (smallest point sizes), cluster 0 (blue) having intermediate values, and cluster 2 (green) showing the largest Dmax values.

2. DBSCAN

- Why should we use DBSCAN:

DBSCAN excels at identifying clusters of arbitrary shapes, making it particularly well-suited for non-spherical datasets where traditional clustering algorithms like k-means may fail.



PS: t-SNE visualization is used with DBSCAN to **reduce the dimensionality of the data while preserving local structure**, making it easier to **visually identify and interpret the natural clusters** detected by DBSCAN, especially in high-dimensional datasets.

-Metrics:

```
Metrics with DBSCAN
--------------------------------
Silhouette Score  : 0.739
Davies-Bouldin Index : 0.086
Calinski-Harabasz Index : 35020.68
```

Silhouette Score: 0.739 - excellent cluster separation and cohesion.

Davies-Bouldin Index: 0.086 - outstanding cluster distinction with minimal overlap.

Calinski-Harabasz Index: 35020.68 - very high compactness and separation quality.

3. Feature engineering results

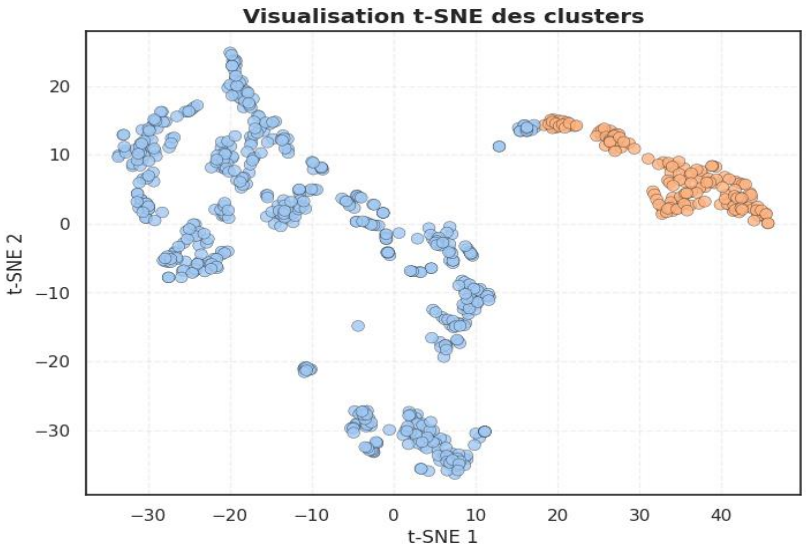
To improve the quality of clustering, we enhanced our dataset through feature engineering by creating new variables from existing data to better represent the properties of the alloys.

We :

- Calculated the number of elements present in each alloy (`Nb_elements`);

- Derived descriptive statistics on the mass percentages of the elements: mean, standard deviation, and maximum value (`Mean_pct`, `Std_pct`, `Max_elem_pct`);
- Computed thermodynamic indicators:
  - `Delta_T = Tx(K) - Tg(K)` (amorphous phase stability range),
  - `Trg = Tg(K) / Tl(K)` (glass-forming ability indicator),
  - `Reduced_Tx = Tx(K) / Tl(K)` (relative thermal stability).

Then we applied KMeans clustering to group the alloys based on their thermodynamic and compositional characteristics. We chose the number of clusters using the elbow method. To visualize the results in two dimensions, we used t-SNE:



The t-SNE plot shows two clear, separate clusters with no overlap. The blue cluster is large and spread out on the left, while the orange cluster is small and tightly grouped in the upper-right.

```
Silhouette Score : 0.810
Davies-Bouldin Index : 0.267
Calinski-Harabasz Score : 4048.572
```

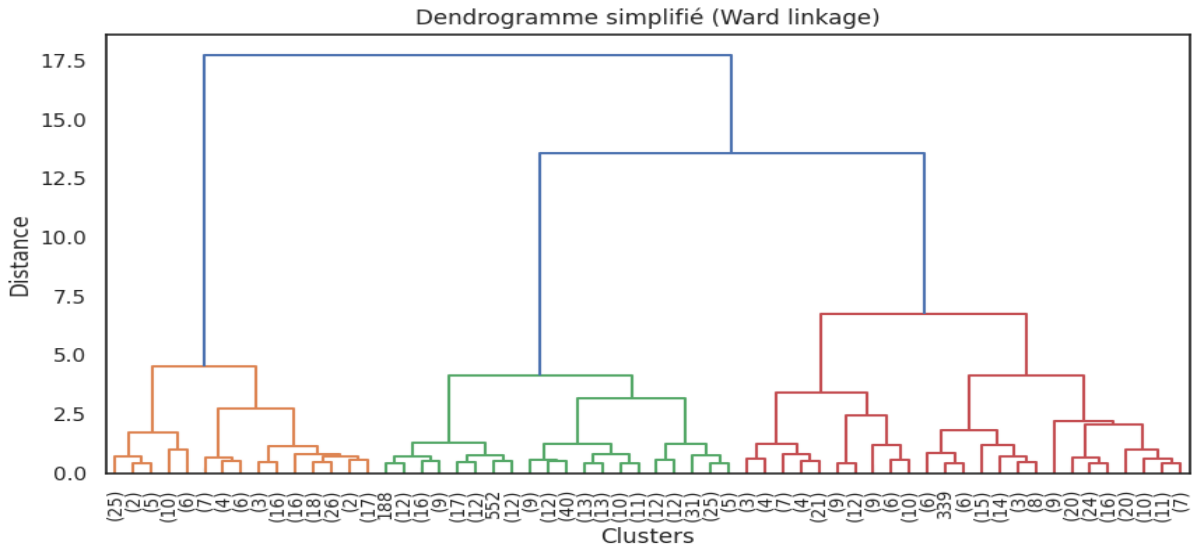
The clustering quality is confirmed by the evaluation metrics:

- **Silhouette Score** (0.810) -> well-separated and cohesive clusters.
- **Davies-Bouldin Index** (0.267) -> clusters are compact and distinct.
- **Calinski-Harabasz Score** (4048.572) -> strong cluster separation relative to their internal variance.

4. Agglomerative Hierarchical Clustering (AHC)

Agglomerative hierarchical clustering is a **bottom-up clustering** method that starts with each data point as its own cluster and iteratively merges the closest pairs until a single hierarchy is formed.

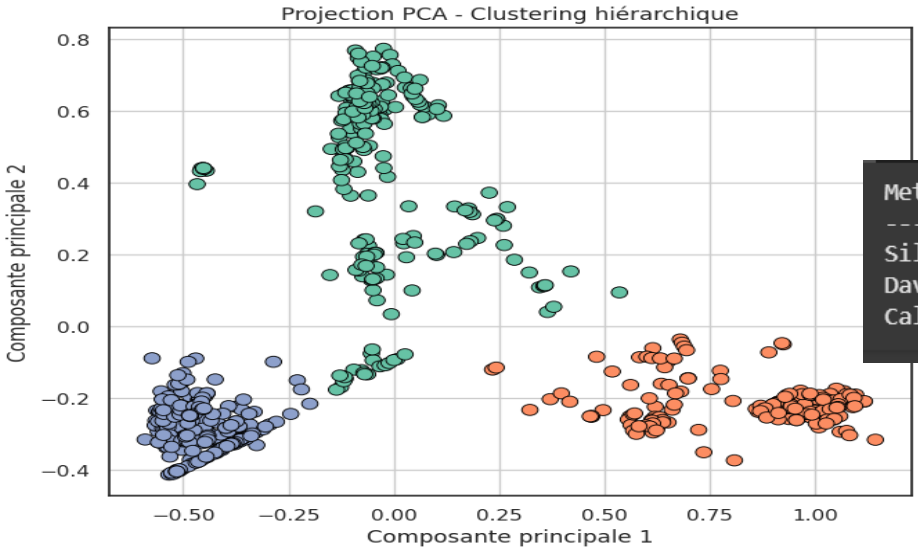
- ## Dendrogram



This dendrogram visualizes hierarchical clustering based on minimizing within-cluster variance.

- **Structure**: Several clear groupings emerge at different heights, indicating varying cluster similarity.
- **Main Split**: A major division around distance 17.5 separates the data into two large families.
- **Subclusters**: Smaller subgroups merge at lower distances (1–4), showing strong local similarity.
- **Cluster Sizes**: Uneven, with some large, complex groups and others more compact.

Results




```
Metrics using Hierarchical clustering
----------------------------------------
Silhouette Score: 0.7084
Davies-Bouldin Index: 0.4341
Calinski-Harabasz Score: 2524.1165
```

This PCA projection effectively visualizes the hierarchical clustering results in a reduced 2D space, highlighting the separation between clusters. The clustering evaluation (**Silhouette:** 0.71; **Davies-Bouldin**: 0.43; **Calinski-Harabasz**: 2524) indicates good internal cohesion and clear separation, with varying compactness across clusters.

Conclusion

This project successfully applied various clustering techniques to group metallic glass alloys based on their chemical composition and thermophysical properties. Through **detailed preprocessing**, **feature engineering**, and **multiple clustering methods** (K-means, DBSCAN, and AHC), we revealed distinct, well-separated groups supported by strong evaluation metrics.

These findings can help better understand alloy behavior and contribute to the development and design of new metallic glasses with desired properties.