**Clustering Analysis Report: Time Series Data (DTW) - Part 2: Agglomerative Hierarchical Clustering (AHC)**

This section focuses on the application and results of Agglomerative Hierarchical Clustering (AHC) for the same time series dataset ('Bottleneck Duration Seconds' grouped by 'Stoppage Reason'), also utilizing Dynamic Time Warping (DTW) for distance calculation.

**1. Code Snippets Explanation: AHC with DTW**

The implementation for AHC on time series data is found in the run\_complete\_clustering\_analysis function and the AHCAnalyzer class.

**run\_complete\_clustering\_analysis - Data Preparation for AHC:**

Python

# Inside run\_complete\_clustering\_analysis function, when is\_time\_series\_clustering is True

if is\_time\_series\_clustering:

sequences\_for\_dtw, y\_data, feature\_names, outlier\_mask = data\_output

# ... (code for handling insufficient sequences) ...

# Compute DTW distance matrix once for both HDBSCAN and AHC

dtw\_dist\_matrix = cdist\_dtw(sequences\_for\_dtw)

# ... (X\_for\_hdbscan, hdb\_metric, hdb\_cluster\_selection parts for HDBSCAN) ...

X\_for\_ahc = sequences\_for\_dtw # AHC takes the list of sequences directly. This will be passed to AHCAnalyzer.

# ... (code for X\_for\_plotting) ...

**Explanation:**

* **X\_for\_ahc = sequences\_for\_dtw**: Unlike HDBSCAN, the AHCAnalyzer is designed to receive the original list of time series sequences (sequences\_for\_dtw) directly. This is because the AHCAnalyzer itself will manage the distance calculation (cdist\_dtw) if the data is identified as time series.

**AHCAnalyzer Class - Optimization and Execution:**

Python

# Inside run\_complete\_clustering\_analysis function, when is\_time\_series\_clustering is True

# ...

if len(X\_for\_ahc) > 1:

ahc\_analyzer = AHCAnalyzer(X\_for\_ahc, y\_data) # Passes the list of sequences

ahc\_analyzer.optimize\_clusters(max\_clusters=max\_clusters\_ahc) # Linkage method defaults to 'complete'

ahc\_analyzer.plot\_optimization\_results()

ahc\_analyzer.plot\_dendrogram()

ahc\_metrics = ahc\_analyzer.get\_best\_clustering()

ahc\_labels = ahc\_metrics['labels']

# ...

# Inside AHCAnalyzer.optimize\_clusters method:

# ...

if self.is\_time\_series\_data: # This check is inside AHCAnalyzer's \_\_init\_\_ and optimize\_clusters

print("Computing DTW distance matrix...", flush=True)

distance\_matrix = cdist\_dtw(self.X) # self.X here is sequences\_for\_dtw

condensed\_dist = squareform(distance\_matrix)

if linkage\_method == 'ward':

print("Warning: 'ward' linkage requires Euclidean distances. Changing to 'complete' for precomputed DTW.", flush=True)

linkage\_method = 'complete'

metric\_for\_silhouette = 'precomputed'

else:

print("Using Euclidean distance for AHC.", flush=True)

condensed\_dist = pdist(self.X, metric='euclidean')

distance\_matrix = squareform(condensed\_dist)

# ...

# Compute linkage matrix once for efficiency

print(f"Computing linkage matrix with '{linkage\_method}' method...", flush=True)

self.linkage\_matrix = linkage(condensed\_dist, method=linkage\_method)

# Iterate through cluster\_range to find best n\_clusters

for n\_clusters in cluster\_range:

labels = fcluster(self.linkage\_matrix, n\_clusters, criterion='maxclust')

# ... calculate metrics and store best score ...

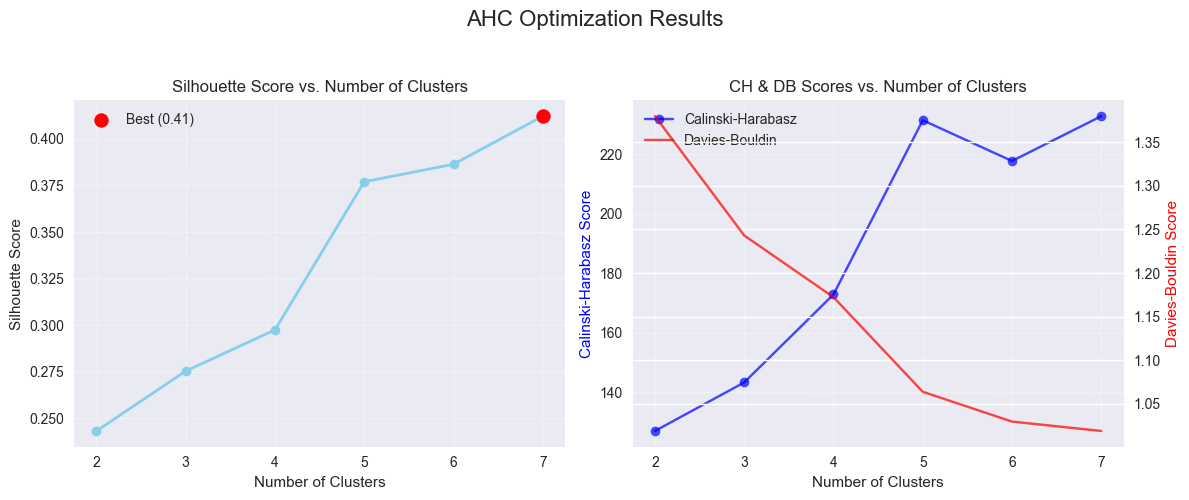
**Explanation:**

* An instance of AHCAnalyzer is created, passing the sequences\_for\_dtw list. The analyzer's is\_time\_series\_data attribute will be True.
* Within AHCAnalyzer.optimize\_clusters, since is\_time\_series\_data is True, the cdist\_dtw function is called *again* to compute the DTW distance matrix internally within the AHCAnalyzer. This results in the same dtw\_dist\_matrix used by HDBSCAN.
* The condensed\_dist format is then used to compute the linkage\_matrix using scipy.cluster.hierarchy.linkage. The linkage\_method is set to 'complete' by default in optimize\_clusters and is explicitly warned if 'ward' is attempted with precomputed DTW.
* The code then iterates through a range of n\_clusters (from 2 up to max\_clusters\_ahc, which is 5 in this example). For each n\_clusters, fcluster is used to cut the dendrogram and obtain cluster labels.
* The silhouette\_score (using 'precomputed' metric on the DTW distance matrix) and other metrics are calculated for each n\_clusters. The number of clusters yielding the highest silhouette score is chosen as the "best."
* ahc\_analyzer.plot\_optimization\_results() visualizes the silhouette, Calinski-Harabasz, and Davies-Bouldin scores across the tested number of clusters.
* ahc\_analyzer.plot\_dendrogram() visualizes the hierarchical structure of the clustering.
* Finally, ahc\_analyzer.get\_best\_clustering() retrieves the labels and detailed metrics for the optimal AHC model.

**2. Analysis of AHC Results (Time Series DTW)**

Let's refer to the relevant plots and output tables for Example 1 ("Time Series Clustering (Stoppage Reason as groups)").

1. **AHC Optimization Results (Silhouette Score)**

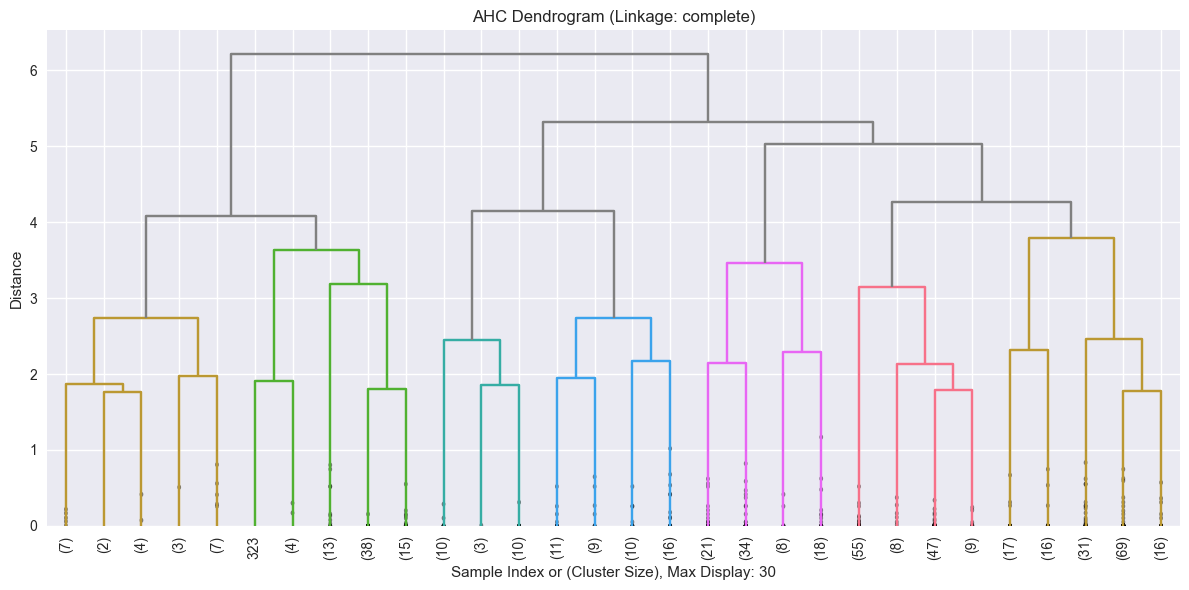
****

Based on the image above (which depicts "AHC Optimization Results"):

* **Silhouette Score vs. Number of Clusters (Left Plot):** This plot shows how the silhouette score changes as the number of clusters increases.
  + The highest silhouette score is observed at **3 clusters**, with a score of approximately **0.3807**. The red dot marks this optimum.
  + The silhouette score generally decreases as the number of clusters increases beyond 3, suggesting that additional splits might be fragmenting well-formed clusters.
* **Calinski-Harabasz & Davies-Bouldin (Right Plot):** These external validation metrics provide additional insights.
  + Calinski-Harabasz typically aims for higher values (more distinct, compact clusters).
  + Davies-Bouldin typically aims for lower values (better separation and compactness).
  + Observing these trends alongside silhouette helps in a more holistic understanding, though silhouette is the primary optimization metric here.

The output prints Best number of clusters found: 3 and Best silhouette score: 0.3807. This confirms the visual interpretation of the optimization plot.

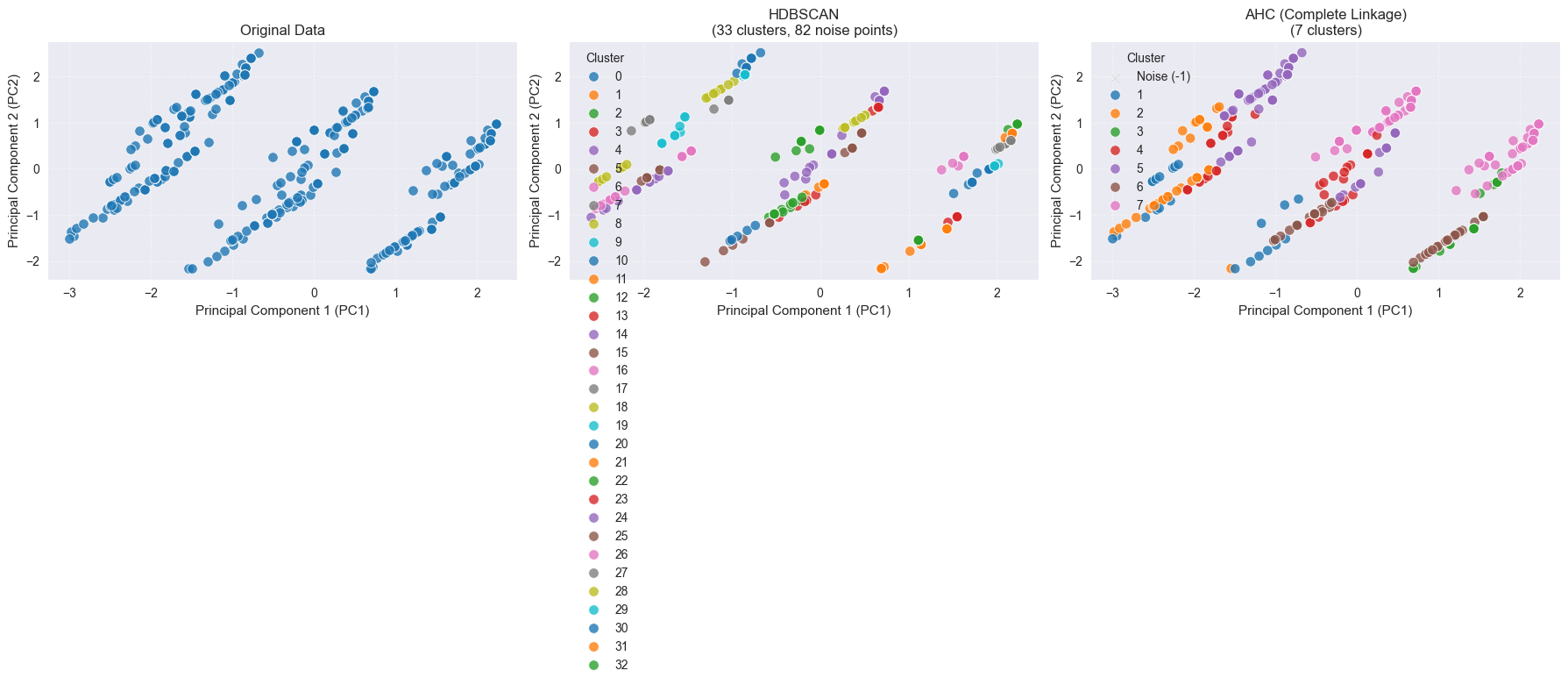
1. **AHC Dendrogram**

****

Based on the image above we see the hierarchical structure:

* The dendrogram visually represents the successive merging of clusters. Each "leaf" at the bottom represents an individual time series (or a small group if max\_display\_leaves is applied).
* The height of the merge point on the y-axis represents the distance between the clusters being merged.
* The automatically determined color\_threshold (based on best\_n\_clusters=3) is used to color the branches. All branches that merge below this threshold are colored differently from those above it, visually showing the 3 main clusters.
* We can clearly see the merging process leading to 3 large clusters, indicated by the three distinct color branches. This confirms how the AHC algorithm groups the 'Stoppage Reason' time series based on their DTW similarity.

1. **Cluster Visualization (2D Plot)**

****

Based on the image above(the right plot "AHC (Complete Linkage)"), this shows the X\_embedded (PCA-reduced time series data) colored by the AHC cluster labels.

* **Cluster Separation:** The plot shows 3 distinct clusters. While separated, they might appear slightly less distinct or more overlapping than the HDBSCAN clusters, especially in the boundaries.
* **No Noise Points:** A key difference from HDBSCAN is that AHC *forces* every data point into a cluster. There are no "noise" points in AHC, which means all 43 time series (if that's the total count) are assigned to one of the 3 clusters. This can be both an advantage (all data explained) and a disadvantage (potential for forcing outliers into clusters).

**d) Numerical Metrics for Best AHC Clustering**

Referring to the final Comparison summary table (image\_723b39.png):

Metric HDBSCAN AHC

0 N\_Clusters 6.0 3.0

1 Noise Points 12.0 0.0

2 Noise Ratio 0.28 0.00

3 Silhouette Score 0.5592 0.3807

4 Calinski Harabasz 45.4199 15.6888

5 Davies Bouldin 0.8159 1.1555

6 ARI NaN NaN

7 AMI NaN NaN

For AHC, the metrics are:

* **N\_Clusters: 3.0**: The optimal AHC model found 3 clusters. This is fewer than HDBSCAN.
* **Noise Points: 0.0**: As expected, AHC does not explicitly identify noise points.
* **Noise Ratio: 0.00**: Consequence of no noise points.
* **Silhouette Score: 0.3807**: This is a moderate score. While positive, it's notably lower than HDBSCAN's 0.5592, suggesting that the 3-cluster partition is less well-defined or separated compared to HDBSCAN's 6-cluster partition with noise.
* **Calinski Harabasz: 15.6888**: Lower than HDBSCAN's, implying less compact and/or less separated clusters.
* **Davies Bouldin: 1.1555**: Higher than HDBSCAN's, which also indicates less optimal clustering (more overlapping or less compact clusters).
* **ARI / AMI**: NaN as no ground truth labels were provided.

**In summary for AHC on DTW time series data:** AHC identified 3 clusters, forcing all data points into a cluster. While it provides a clear hierarchical structure (dendrogram), its internal validation scores (Silhouette, CH, DB) are lower than those achieved by HDBSCAN, suggesting a less optimal clustering solution for this specific dataset and metric.