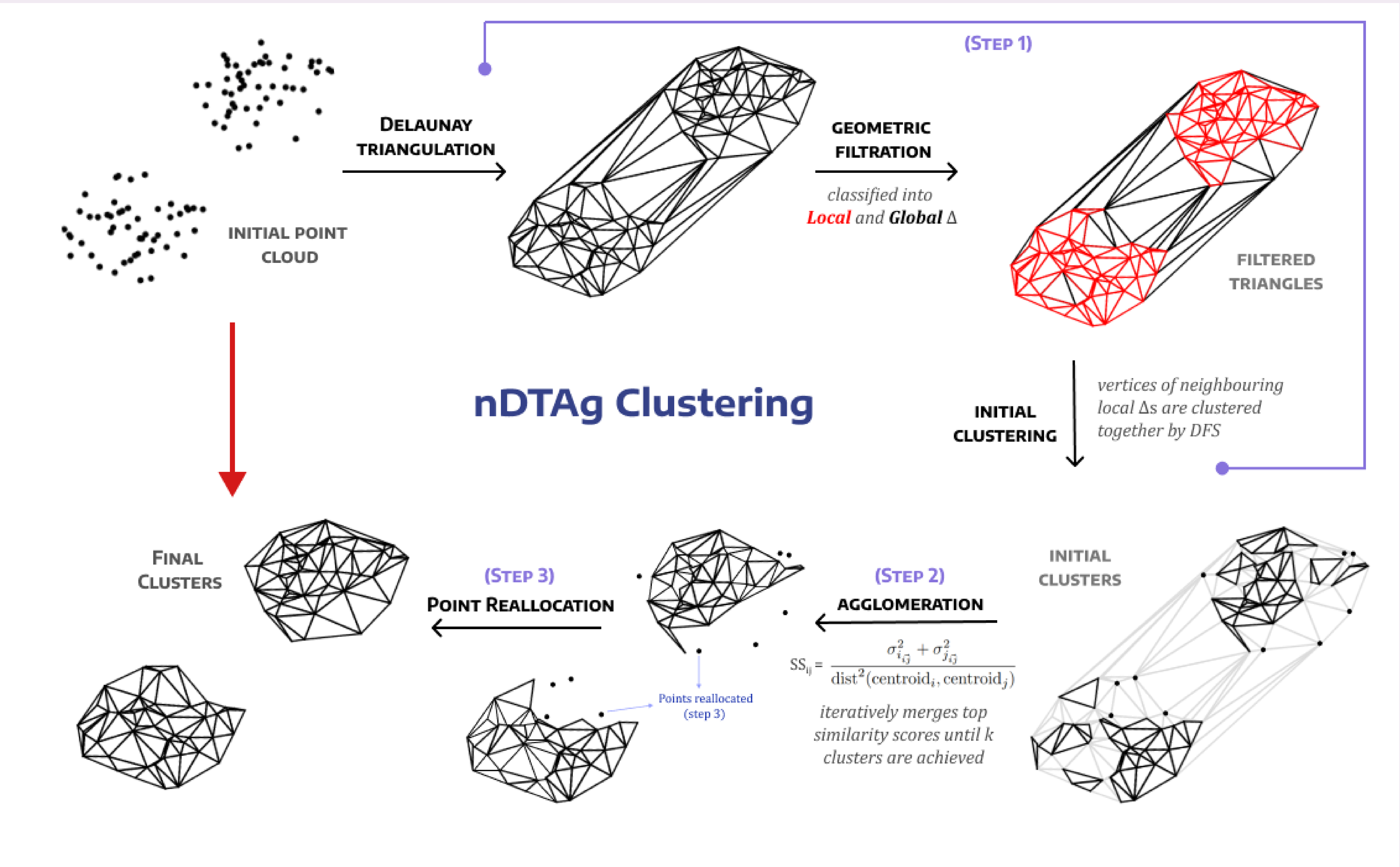
nDTAg – A Delauney Triangulation based Agglomerative Clustering in n-Dimensions

Pranav Debbad[1] , Kanish Thiagarajan[1], Minu Reghunath[1], Sasinas Alias Harita[1], and Ramanathan Muthuganapathy1

1. Indian Institute of Technology Madras (Advanced Geometric Computing Lab)

INTRODUCTION



This work proposes a clustering algorithm nDTAg that leverages Delaunay Triangulation for the spatial clustering of point clouds. The illustrated results in 2D, 3D and higher dimensional spaces demonstrate the potential of our method.
By leveraging Delaunay graphs, the algorithm simplifies the representation of n-dimensional datasets, enabling efficient spatial clustering. Extensive testing and comparison on various datasets demonstrates the effectiveness of the proposed approach, particularly in handling non-linearly separable data as well as datasets with heterogeneous density, producing well-separated clusters

As highlighted in the figure above, nDTAg is comprised of 3 key steps

Preliminary Clustering:
Step 1 generates a preliminary set of clusters by filtering the DT graph on the basis of edge lengths. It computes geometric thresholds, derived from global and local measures of edge lengths, that are applied to classify the Delaunay edges. Vertices of the edges that satisfy these criteria are grouped into preliminary clusters using a graph-based search.

Iterative Agglomeration:
Step 2 employs a directional variance-based agglomeration process. Clusters that are in close proximity and share similar directional variance are iteratively merged. This process continues until the user defined desired number of clusters, k, is achieved.

Point Reallocation:
During Step 3 , points initially identified as unclassified (those that did not meet the criteria for initial clustering in step 1) are re-evaluated. Using less conservative proximity measures, these points are either reassigned to the most appropriate clusters, or they retain the unclassified label.

REFERENCES

[1] Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed tristique ligula nec gravida hendrerit. Etiam venenatis mattis auctor. Nulla nec mi sodales, imperdiet libero non, malesuada nunc.
[2] Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed tristique ligula nec gravida hendrerit. Etiam venenatis mattis auctor. Nulla nec mi sodales, imperdiet libero non, malesuada nunc.
[3] Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed tristique ligula nec gravida hendrerit. Etiam venenatis mattis auctor. Nulla nec mi sodales, imperdiet libero non, malesuada nunc.
[4] Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed tristique ligula nec gravida hendrerit. Etiam venenatis mattis auctor. Nulla nec mi sodales, imperdiet libero non, malesuada nunc.

METHODOLOGY

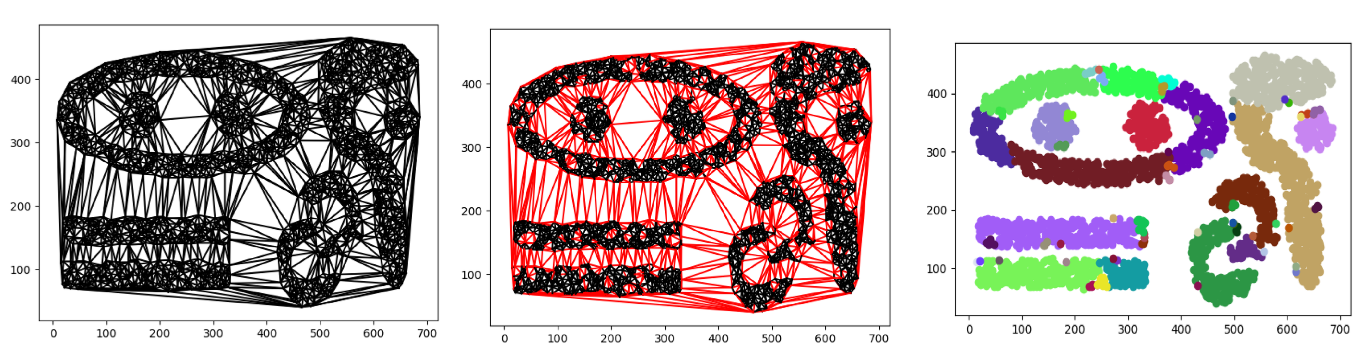
Preliminary clustering

Utilizing statistical measures of the edge lengths in the DT to identify a preliminary set of spatial clusters, an edge length threshold λ(p) for each node p in the Delaunay graph is derived from the mean and standard deviation of the lengths of all edges (i.e. global mean μg and global standard deviation **σ**g respectively) in the graph, as well as the mean length of Delaunay edges containing vertex p, denoted μ(p).

This cutoff value is computed as: $\lambda(p) = \mu_g + \alpha \cdot \left(\frac{\mu(p)}{\mu_g}\right) \cdot \sigma_g$

where **α** is a tunable that states how many local standard deviations are allowed until an edge is classified as long.

Next, a depth-first-search is implemented on the graph to identify a connected group of local simplices within the DT. Once this set of neighbouring local simplices has been clustered, another random local simplex is visited and the process is repeated until no more local simplices remain



Delaunay triangulation of a point set — After Geometric Filtration — After DFS Search

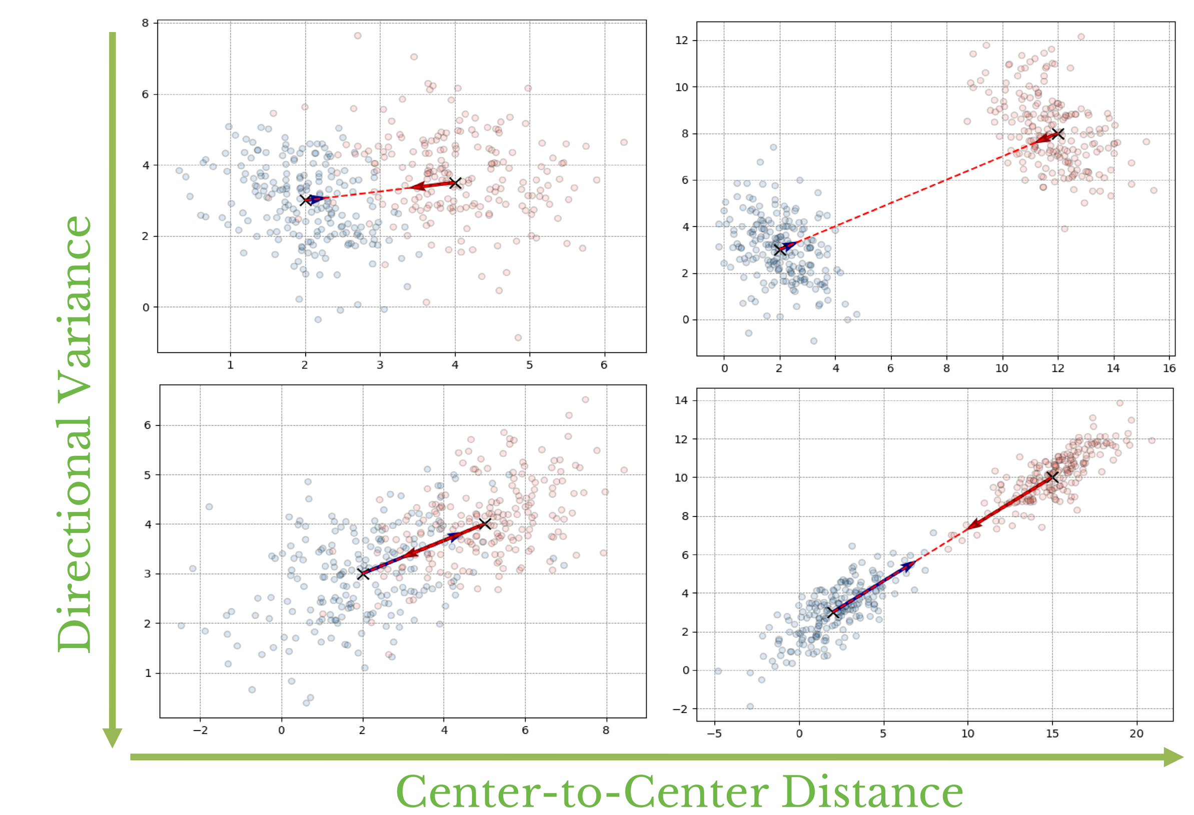
Iterative Agglomeration

Following a bottom-up, agglomerative strategy, the initial clusters are progressively merged based on a similarity score. This merging process is performed pairwise by calculating the variance of the clusters along the unit vector joining their centroids, normalized by the square of the centre-to-centre distance between the two cluster centroids, avoiding scale-dependent biases in the similarity measure. The similarity score between two clusters is computed as:

$$\mathrm{SS}_{i,j} = \frac{\sigma^2_{i_{\vec{ij}}} + \sigma^2_{j_{\vec{ij}}}}{\mathrm{dist}^2(\mathrm{centroid}_i, \mathrm{centroid}_j)}$$

- $\sigma^2_{i_{\vec{ij}}}$: Variance of cluster 1 along the ij vector
- $\sigma^2_{j_{\vec{ij}}}$: Variance of cluster 2 along the ij vector
- $\mathrm{dist}(\mathrm{centroid}_i, \mathrm{centroid}_j)$: Euclidean distance between the centroids of clusters i and j

Cluster pairs with higher cumulative directional variance show higher similarity. Likewise, cluster pairs with smaller C2C distances also exhibit higher similarity,



RESULTS

The clustering algorithms used for benchmarking were implemented using the Scikit-learn library. The scores measure the similarity between the ground truth and the result of the clustering algorithm on 5 datasets:

PC1 – Aggregation, PC2 – Cure, PC3 – Chainlink, PC4 – Zelnik3, PC5 – Longsquare

Algorithm	*PC1* Score	*PC2* Score	*PC3* Score	*PC4* Score	*PC5* Score
nDTAg	**0.99441**	**0.93776**	**1**	**1**	**0.99730**
k-Means	0.72569	0.69482	0.308	0.68208	0.87777
MeanShift	0.88401	0.75576	0.44265	0.40322	0.80828
Agglomerative	0.75	0.8125	0.68027	0.71477	0.87777
DBSCAN	0.85387	0.82795	1	1	0.83333

Additionally, results are shown for 7D dataset in the form of 3D feature subspaces: