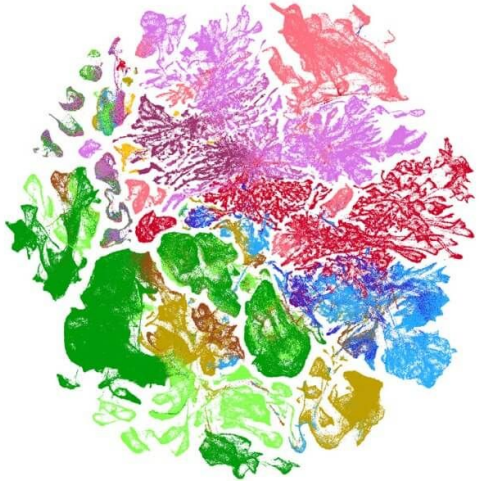
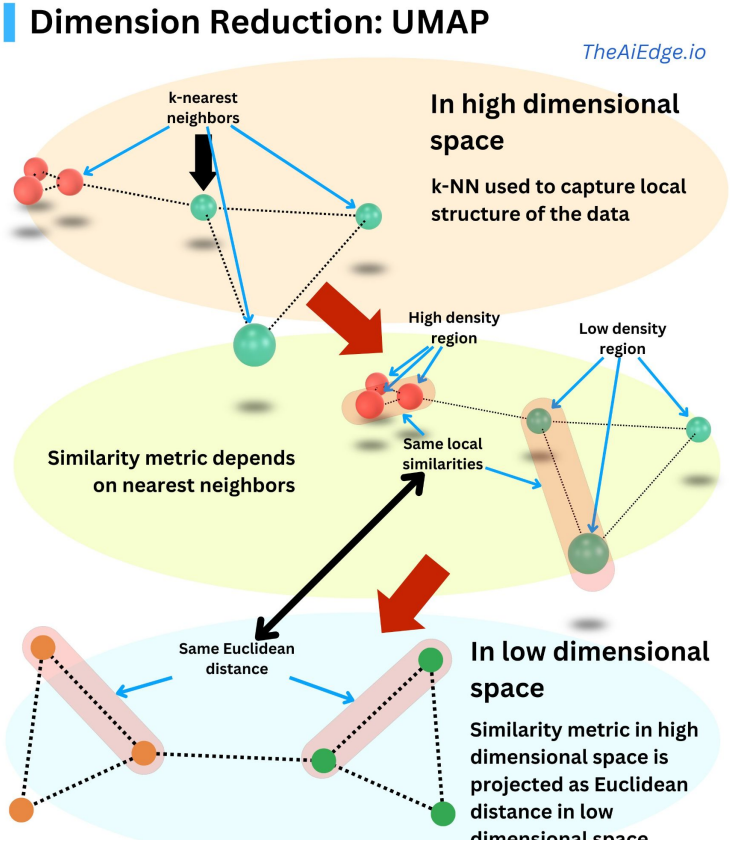
Uniform Manifold Approximation and Projection



UMAP (2018)

- Dimension reduction technique that can be used for visualization.

- UMAP constructs a high-dimensional graph representation of data, then optimizes a low-dimensional graph that preserves relationships.



UMAP

Advantages:

- Increased speed (over t-SNE); computationally efficient and scalable; adjustable parameters.

- Unlike PCA, UMAP can capture nonlinear structures in data

- Better preservation of local structure global structure.

- UMAPs and t-SNEs on real world data: https://pair-code.github.io/understanding-umap/

Limitations of UMAP

- Different results can be produced for the same data set depending on the random initialization and optimization process.

- It does not have an inverse function that can map the lower-dimensional embedding back to the original high-dimensional space.

- Umap is not a metric learning technique, so it does not guarantee that the distances or angles in the lower-dimensional embedding are proportional or equivalent to those in the original high-dimensional space.

- It is sensitive to the choice of parameters, such as n_neighbors and min_dist, which can affect the shape and size of the clusters and the distance function used to compare the data points

Source: https://www.linkedin.com/advice/3/what-advantages-disadvantages-umap-over-other

UMAP Applications

- Machine learning: clustering, classification, anomaly detection.

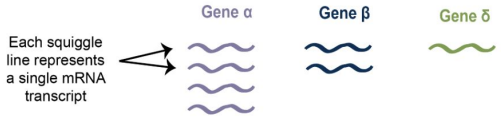
- Image analysis; social media analysis.

- Genomics and transcriptomics:
  - Understanding population structure.
  - Phenotype correlations.
  - Clustering cells or tissues based on gene expression.

UMAP in Transcriptomics

Isolate the nuclei from the cells in the sample of brain tissue and extract the RNA found in each nucelus.

Sequence the mRNA transcripts found in each cell's nucleus in order to determine which genes each brain cell was expressing.



Count the number of mRNA transcripts found for each gene. This allows us to quantify how much each cell was expressing each gene.

	Gene α	Gene β	Gene δ
# of mRNA transcripts found in Cell 1	4	2	1

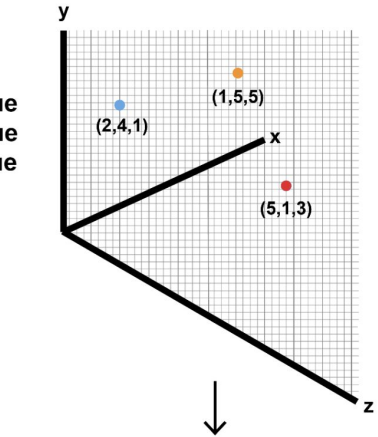
Create a table that shows how much each cell was expressing each gene.

UMAP in Transcriptomics

	Gene α	Gene β	Gene δ
# of mRNA transcripts found in Cell 1	2	4	1
# of mRNA transcripts found in Cell 2	1	5	5
# of mRNA transcripts found in Cell 3	5	1	3
repeat count for thousands of cells...	...	...	...

Repeat this process for THOUSANDS of cells. Remember, this means we are counting how much EACH cell was expressing EACH gene. If we wanted to create a table that listed the data in full, this data table would have thousands of rows.

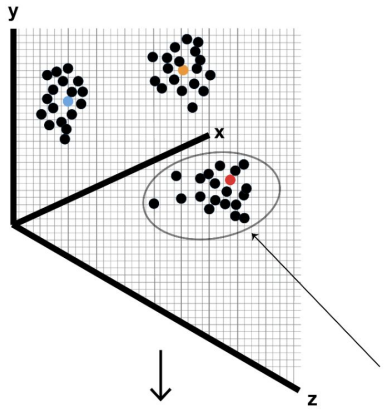
x = gene α value
y = gene β value
z = gene δ value



If we wanted to create a graph that plotted the inital data for cell 1, cell 2, and cell 3 and their relative amount of expression of gene alpha, gene beta, and gene delta, we would need a 3D graph like the one on the left.

UMAP in Transcriptomics

x = gene α value
y = gene β value
z = gene δ value



We can repeat this process for the thousands of cells that were collected from the brain tissue sample. Notice that the cells begin to cluster based on how similar their gene expression for gene alpha, gene beta, and gene delta is to one another. These clusters help us identify which cells may be more similar and/or dissimilar to one another!

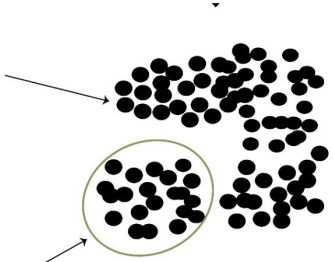
when we plot the gene expression data for more cells, we notice that cell 3 (red dot) clusters next to these other cells from the sample

	Gene α	Gene β	Gene δ	repeat for thousands of genes...
# of mRNA transcripts found in Cell 1	2	4	1	...
# of mRNA transcripts found in Cell 2	1	5	5	...
# of mRNA transcripts found in Cell 3	5	1	3	...
repeat count for thousands of cells...	...	...	...	...

In addition to collecting data on gene expression for thousands of cells, scientists will add another layer of complexity by measuring the gene expression of these thousands of cells for THOUSANDS of genes. A table displaying this data would have thousands of rows and thousands of columns. Since the graph would now have much more than just 3 dimensions, we will need a special type of tool to graphically represent this data in a way that humans can visualize.

UMAP in Transcriptomics

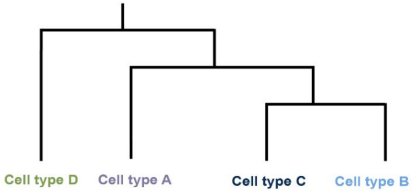
Each dot represents a single nucleus isolated from a single brain cell



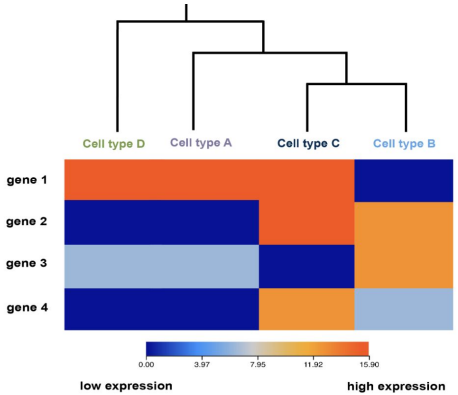
Identify clusters in the data--these clusters represent cells that are more like each other than they are like any other cells

UMAP

In order to plot this many-dimensional graph in a way humans can visualize, we use a dimensionality reduction tool, such as a UMAP, to plot it in a 2D space. Dimensionality reduction is a technique that helps represent many-dimentional data in just two or three dimensions.



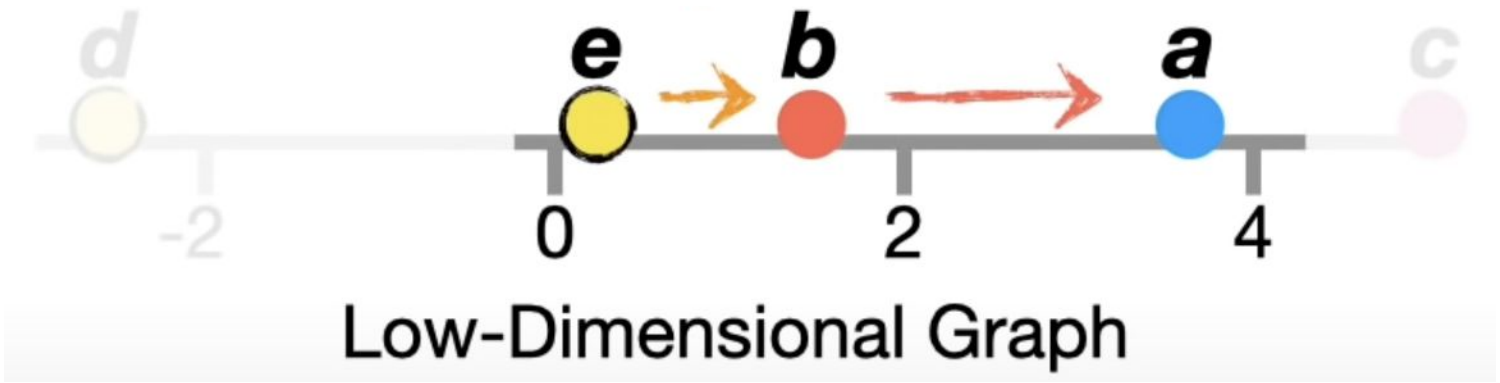
Organize the clusters identified in the UMAP to construct a dendrogram that displays hierarchical relationships between the clusters based on each cell type's similarity and dissimilarity of gene expression.



Use a heatmap below the dendrogram to compare the level of gene expression between each cell type for specific genes of interest.

Main Ideas Pseudocode

1. Initialize high dimensional points into a low dimensional space
2. Move points until clustering matches high dimensional space

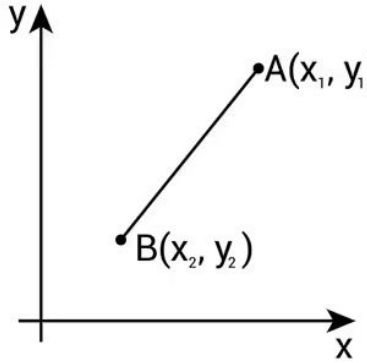


Main Ideas Pseudocode

1. Initialize high dimensional points into a low dimensional space
2. Move points until clustering matches high dimensional space

More in-depth Pseudocode

1. Calculate distances in high dimensions

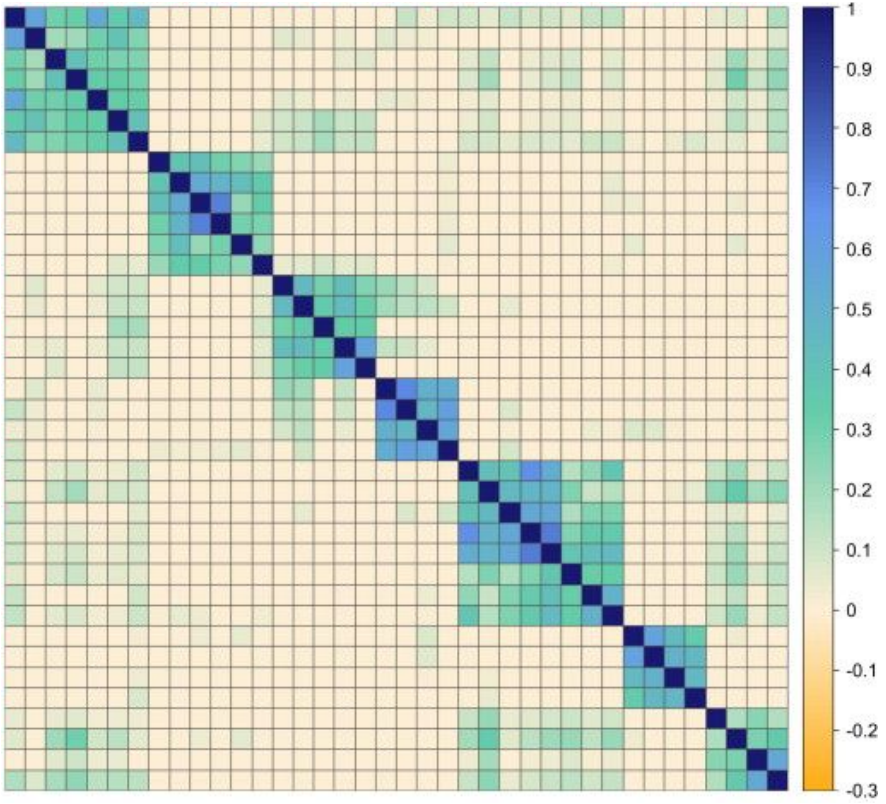


$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

More in-depth Pseudocode

1. Calculate distances in high dimensions

2. Calculate similarity score and symmeterize

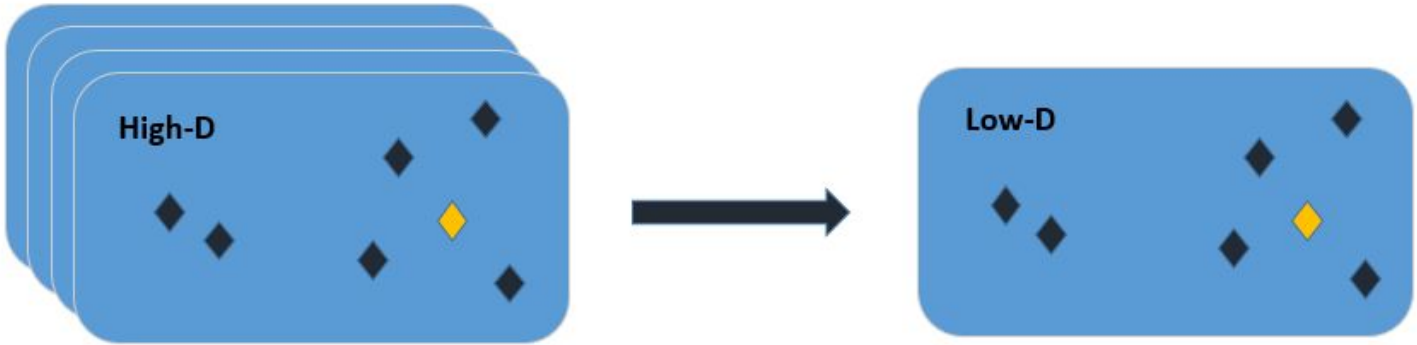


More in-depth Pseudocode

1. Calculate distances in high dimensions

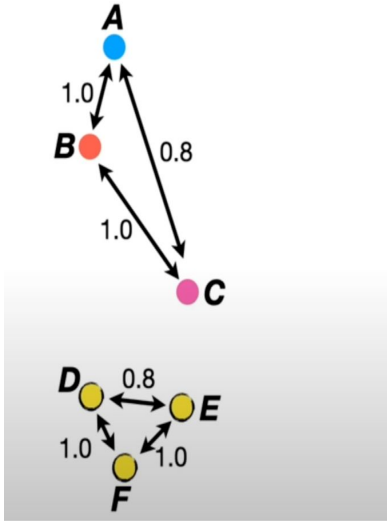
2. Calculate similarity score and symmeterize

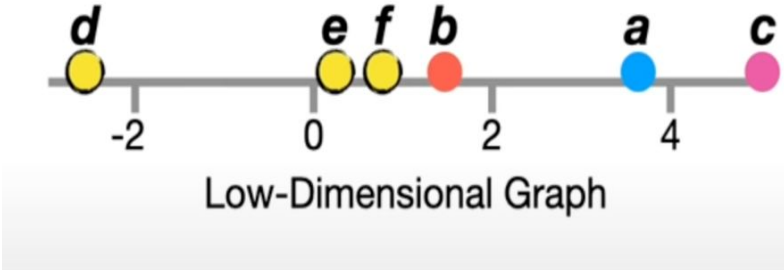
3. Initialize Low Dimension Graph



More in-depth Pseudocode

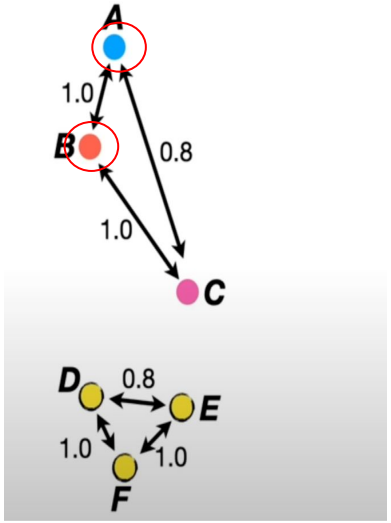
4. Choose Two Points Proportional to Similarity Score

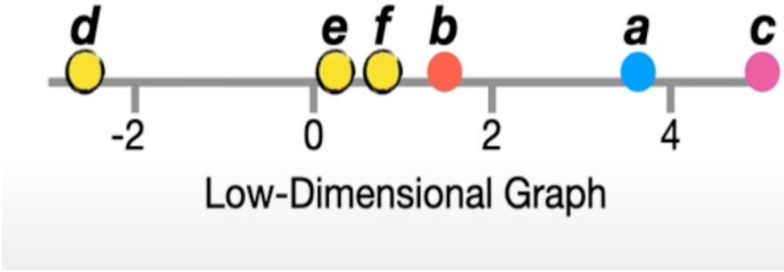




More in-depth Pseudocode

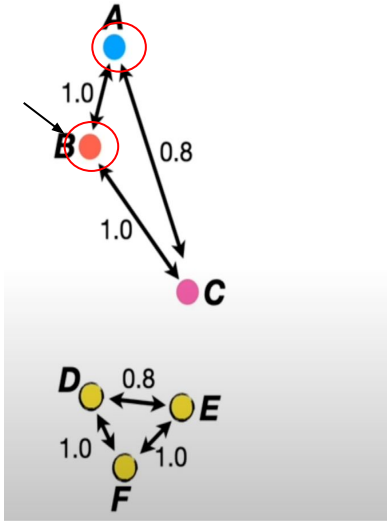
4. Choose Two Points Proportional to Similarity Score

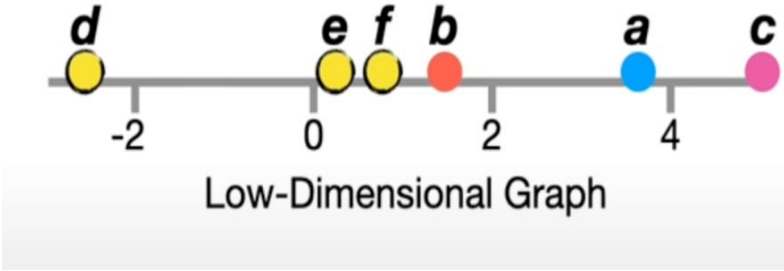




More in-depth Pseudocode

4. Choose Two Points Proportional to Similarity Score

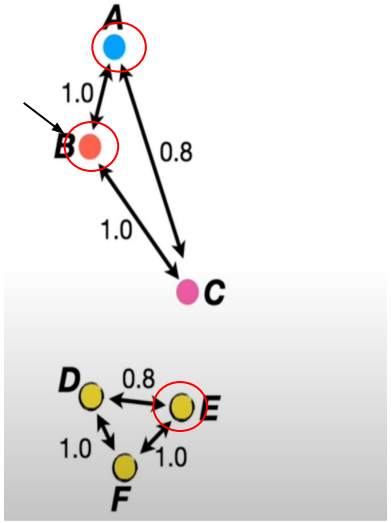


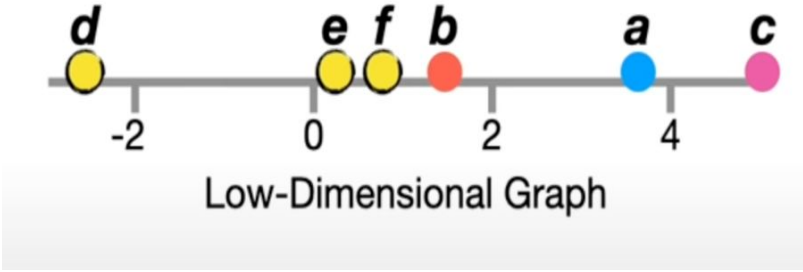


More in-depth Pseudocode

4. Choose Two Points Proportional to Similarity Score

5. Pick Point from Other Cluster to Move



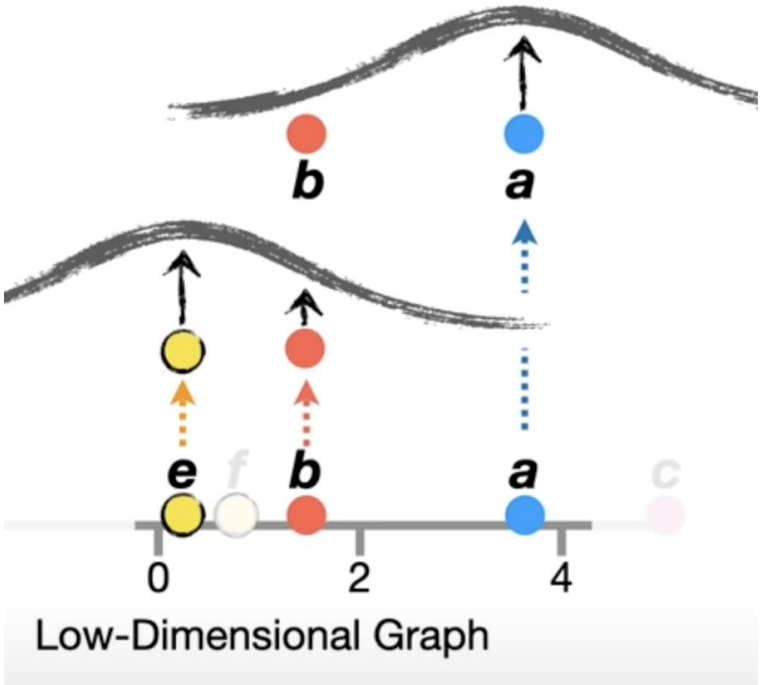


More in-depth Pseudocode

4. Choose Two Points Proportional to Similarity Score

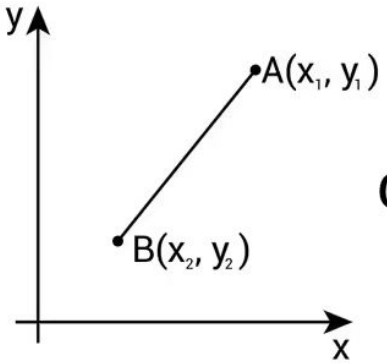
5. Pick Point from Other Cluster to Move

6. Minimize Low Dimension Sim. Scores same Cluster. Max Sim. Score other Cluster



Maths for this Algorithm

1. Calculate distances in high dimensions

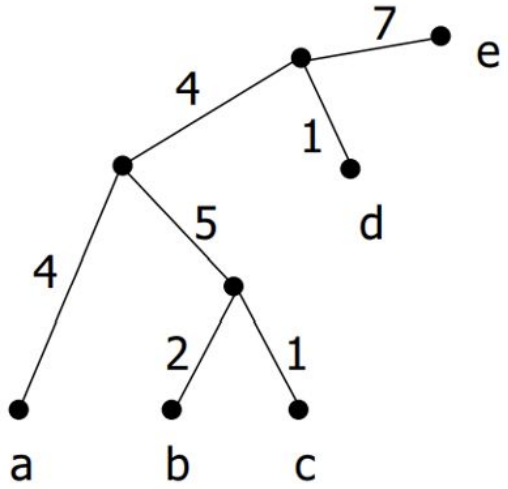


$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Maths for this Algorithm

1. Calculate distances in high dimensions

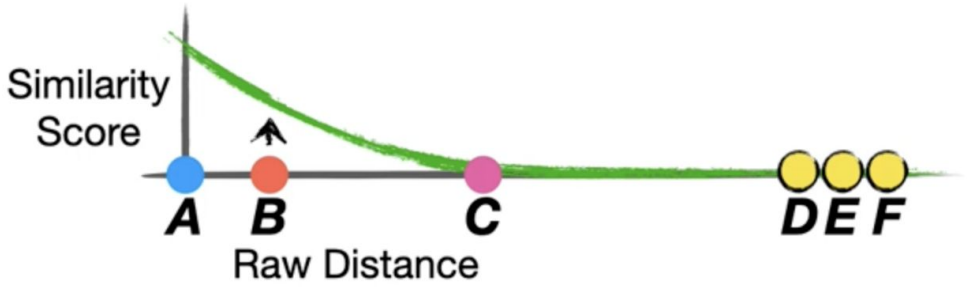
M	a	b	c	d	e
a	0	11	10	9	15
b	11	0	3	12	18
c	10	3	0	11	17
d	9	12	11	0	8
e	15	18	17	8	0



Maths for this Algorithm

1. Calculate distances in high dimensions

2. Calculate similarity score for each point to every other point

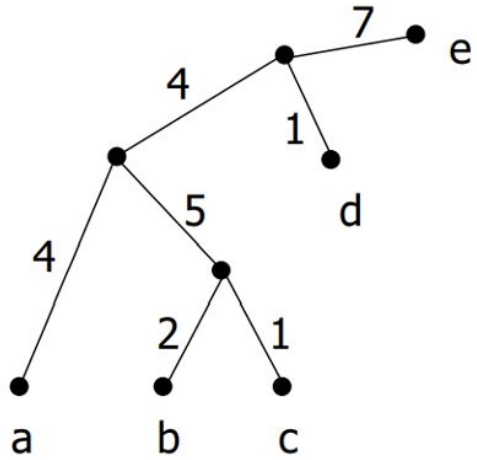


Maths for this Algorithm

1. Calculate distances in high dimensions

2. Calculate similarity score for each point to every other point

M	a	b	c	d	e
a	0	11	10	9	15
b	11	0	3	12	18
c	10	3	0	11	17
d	9	12	11	0	8
e	15	18	17	8	0



$$Sim\ Score = e^{-(raw\ dist. - dist\ nearest\ neighbor)/\sigma}$$

Maths for this Algorithm

1. Calculate distances in high dimensions

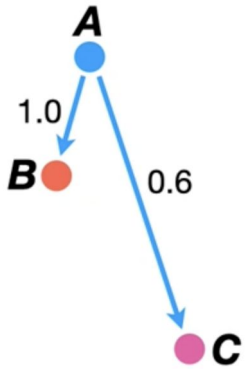
2. Calculate similarity score for each point to every other point

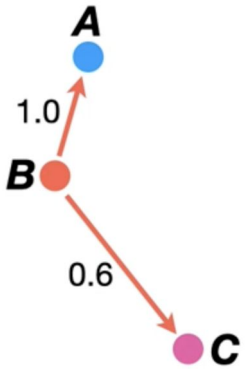
$$\log_2(num.neighbors)$$

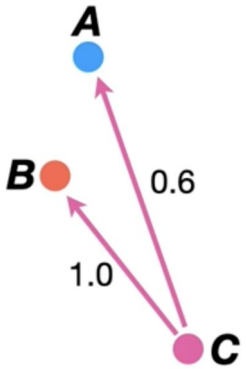
$$Sim\ Score = e^{-(raw\ dist. - dist\ nearest\ neighbor)/\sigma}$$

Maths for this Algorithm

1. Calculate distances in high dimensions
2. Calculate similarity score for each point to every other point
3. Symmetrize the Scores

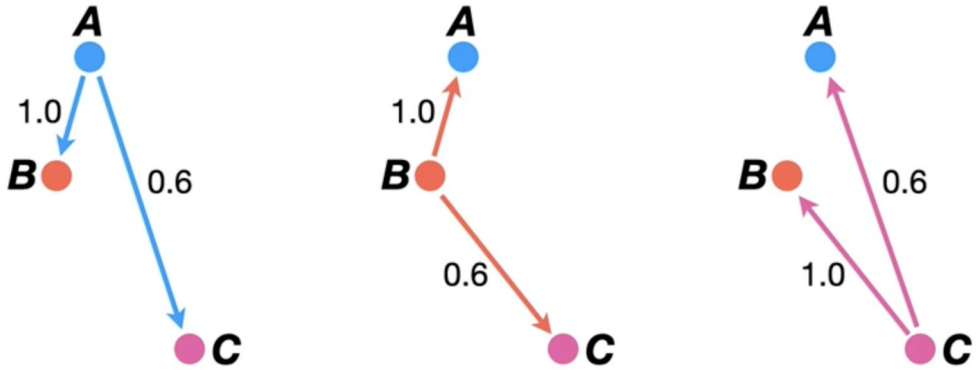






Maths for this Algorithm

1. Calculate distances in high dimensions
2. Calculate similarity score for each point to every other point
3. Symmetrize the Scores



$$Symmetrical\ Score = (S1 + S2) - S1S2$$

Interesting Links

1. https://www.geeksforgeeks.org/spectral-embedding/
2. https://pair-code.github.io/understanding-umap/

Pseudo more in depth

1. Calculate distances
2. Draw a curve over the graph that calculates the similarity scores. The curve is based on the number of predetermined clusters. Common default value for number of neighbors is 15
    a. Take log2 of high dimension neighbors.
    b. This defines shape of the curve
    c. Y axis values on the curve is the similarity. Small number means low similarity
    d. Recursively do this for each point
3. Take score similar to mean to symmeterize
4. Make low dimensional graph (spectral embedding)
5. Randomly pick two points to move proportional to high-dimension score. Higher scores have a better probability to move
6. Flip a coin to pick which one to move
7. Randomly pick point from other cluster to move
8. Figure out how far to move point based on t-distributed bell shape curve.
    a. Minimize in one of the distributions and max in another
    b. Iteratively do this

Pseudo Math (for this week just gonna do similarity scores)

1. Calculate distances
2. Similarity score = e^-(raw_dist - dist_nearest_neighbor)/sigma
    a. Adjust sigma until the sum of the non-zero scores equal log2(number_of_nearest_neighbors)
    b. Calculate similarity score for each different point
3. Symmetrize the scores (S1 + S2) - S1*S2