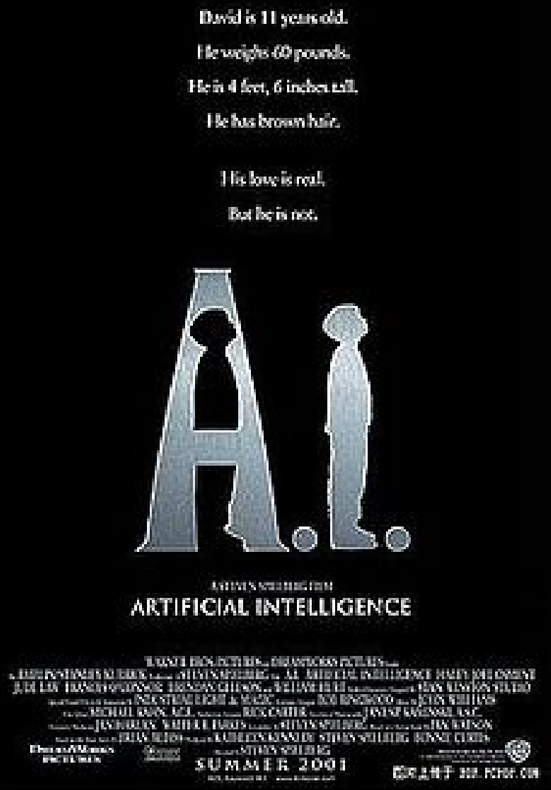
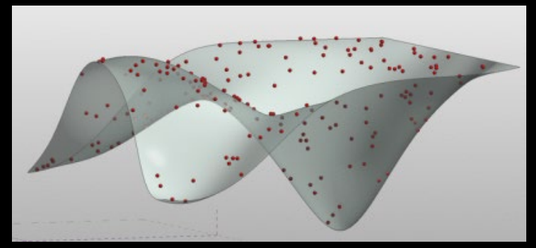
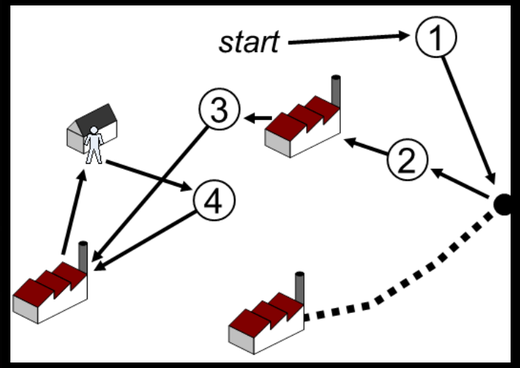
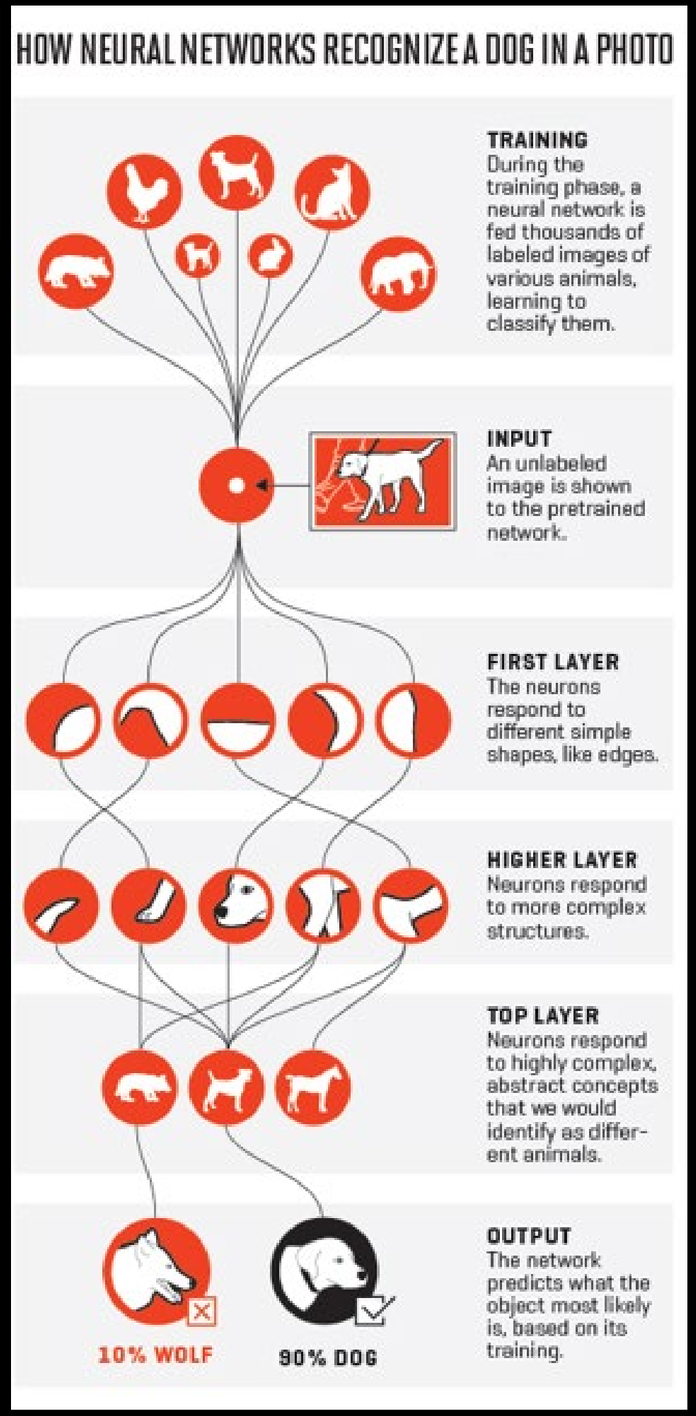

David is 11 years old.
He weighs 60 pounds.
He is 4 feet, 6 inches tall.
He has brown hair.
His love is real.
But he is not.
A.I.
ARTIFICIAL INTELLIGENCE
SUMMER 2001




start
1
2
3
4


data
information
patterns
relationships
large
process
database
analyzing
useful
discover
customer
customers
searching
effective
amounts
using
done
business
processing
businesses
behavior
look
relationship
summarizing
computerized
transactions
warehousing
supermarket
marketing
applications
pre-existing
history
uncover
Exploring
held
pattern
digging
depends
stored
find
search
gathered
mining
statistical
implies
establish
class
preferences
learn
habits
group
trends
Web
understand
important
detect
practice
analyze
meaning
way
dicing
databases
predict
order
management
increase
algorithms
hidden
raw
automatically
decrease
huge
part
preexisting
companies
manually
contained
programs
identify
new
turn
strategies
used
sorting
use
gathering
perspectives
develop
via
better
well
collecting
computer
slicing
collection
becomes
activity
sales
tons
costs
CRM
batches
within
sophisticated
shopping
detailed
obvious
capabilities
wade
future
correlations
fraud
software
one


HOW NEURAL NETWORKS RECOGNIZE A DOG IN A PHOTO
TRAINING
During the training phase, a neural network is fed thousands of labeled images of various animals, learning to classify them.
INPUT
An unlabeled image is shown to the pretrained network.
FIRST LAYER
The neurons respond to different simple shapes, like edges.
HIGHER LAYER
Neurons respond to more complex structures.
TOP LAYER
Neurons respond to highly complex, abstract concepts that we would identify as different animals.
OUTPUT
The network predicts what the object most likely is, based on its training.
10% WOLF
90% DOG

Basic Machine learning Methods (1)
Dimension reduction

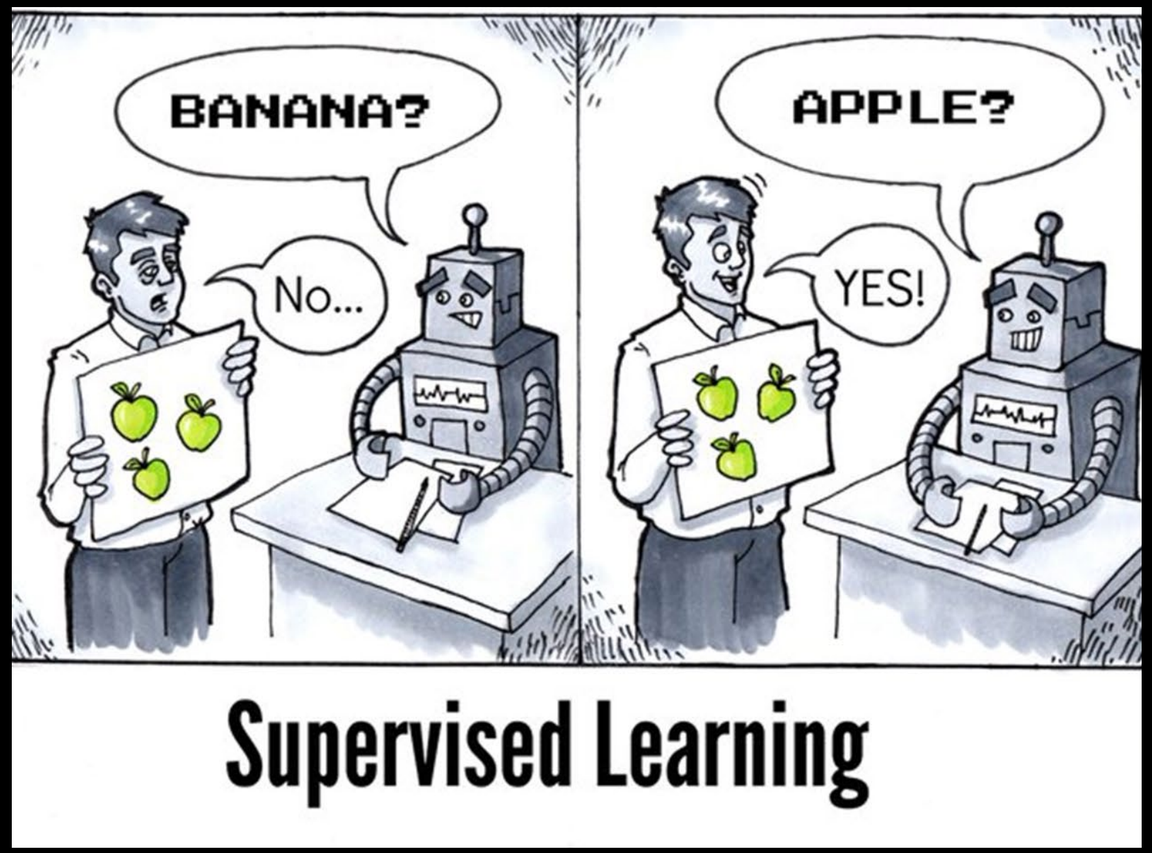
Fengling Chen

2021.06.09

Outline

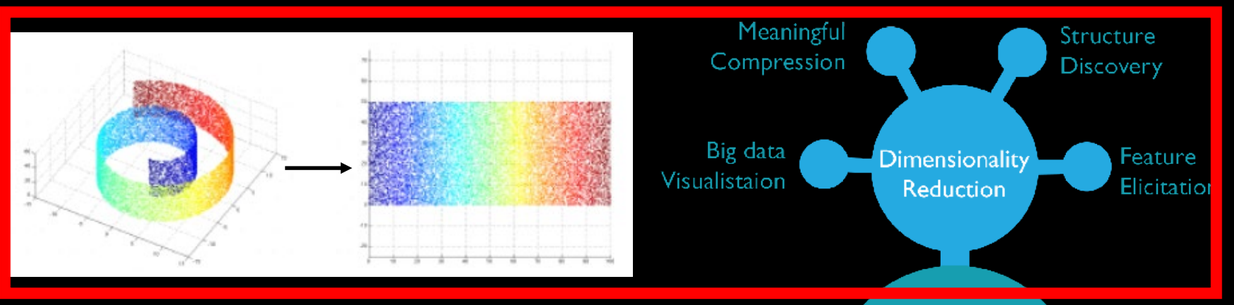
- The general concept of machine learning
- Dimension reduction
- PCA & tSNE & UMAP
- Recommendation

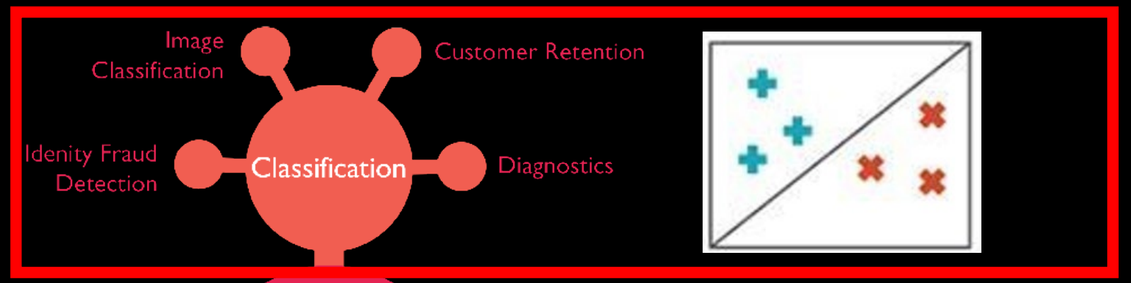
Supervised and unsupervised learning

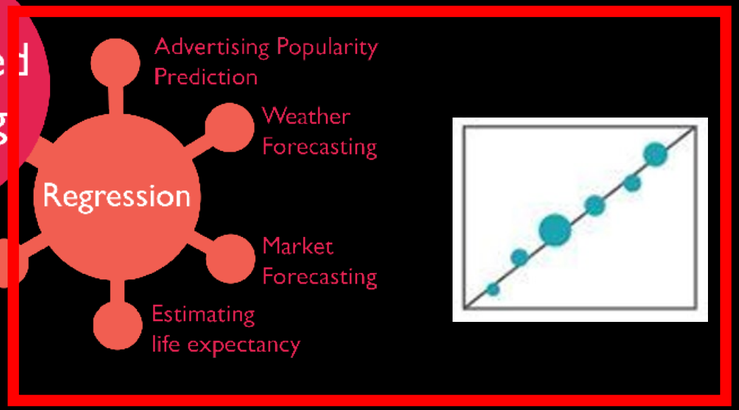


（X,Y）

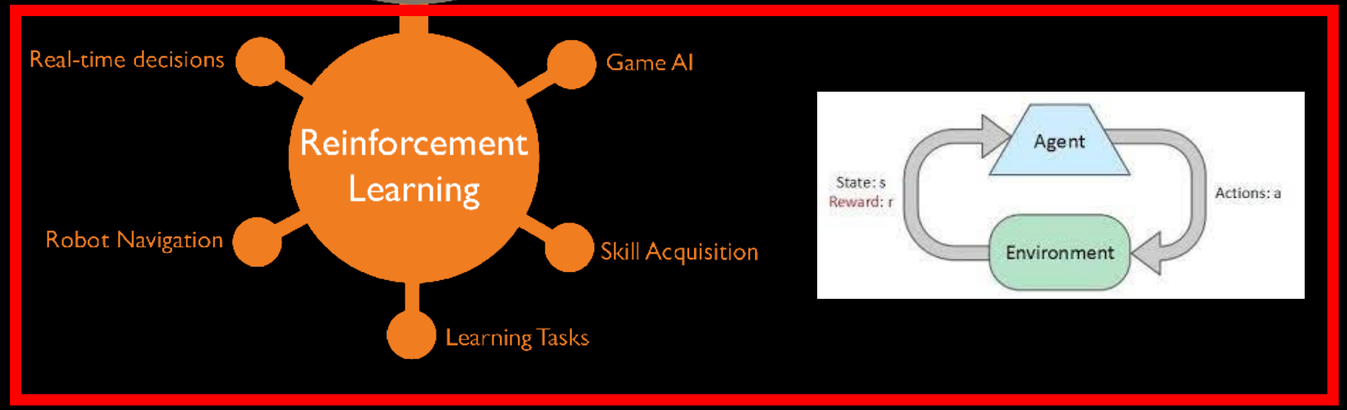
Basic tasks of machine learning







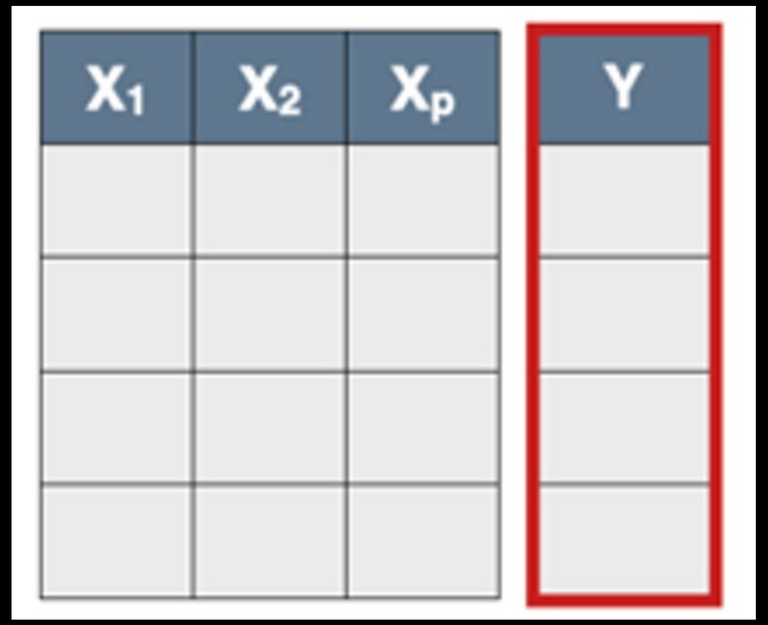
Machine Learning



Input of machine learning methods

feature label

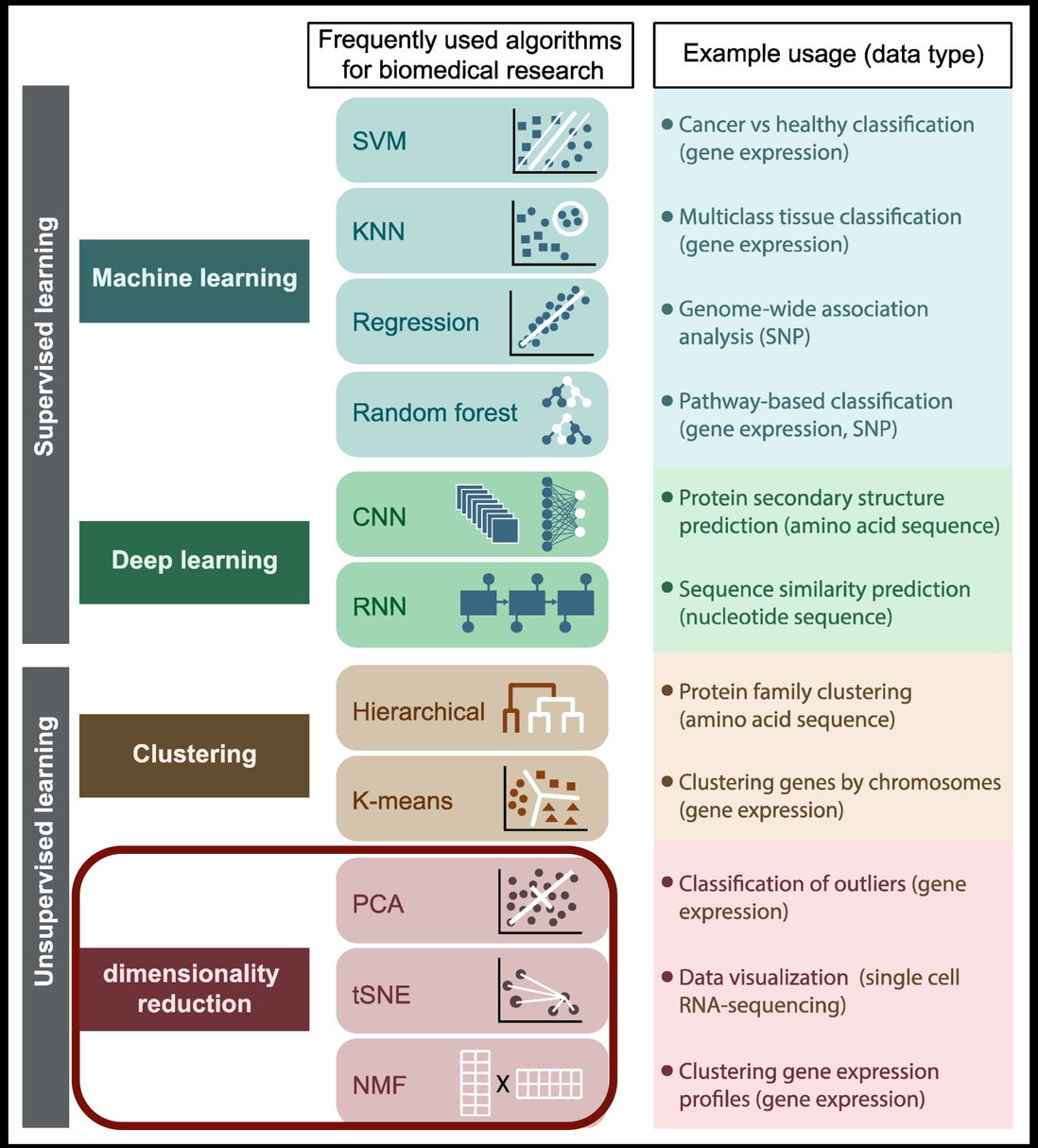
Sample



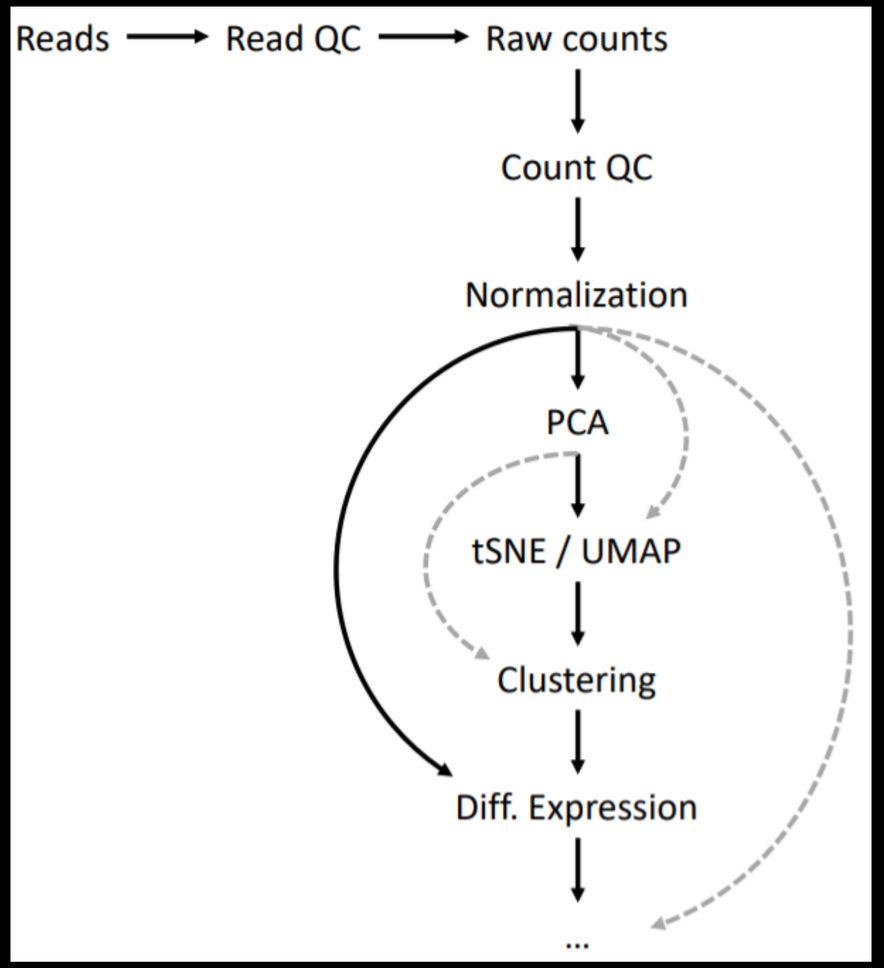
Example: Titanic disaster on Kaggle

PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	**Survived**
1	3	, Mr. Owen	male	22	1	0	A/5 21171	7.25		S	**0**
2	1	Bradley (Flo	female	38	1	0	PC 17599	71.2833	C85	C	**1**
3	3	inen, Miss.	female	26	0	0	N/O2. 3101	7.925		S	**1**
4	1	cques Heat	female	35	1	0	113803	53.1	C123	S	**1**
5	3	Mr. William	male	35	0	0	373450	8.05		S	**0**
6	3	ran, Mr. Jar	male		0	0	330877	8.4583		Q	**0**

Top used algorithms in biological data analysis



When we are dealing with multi-samples



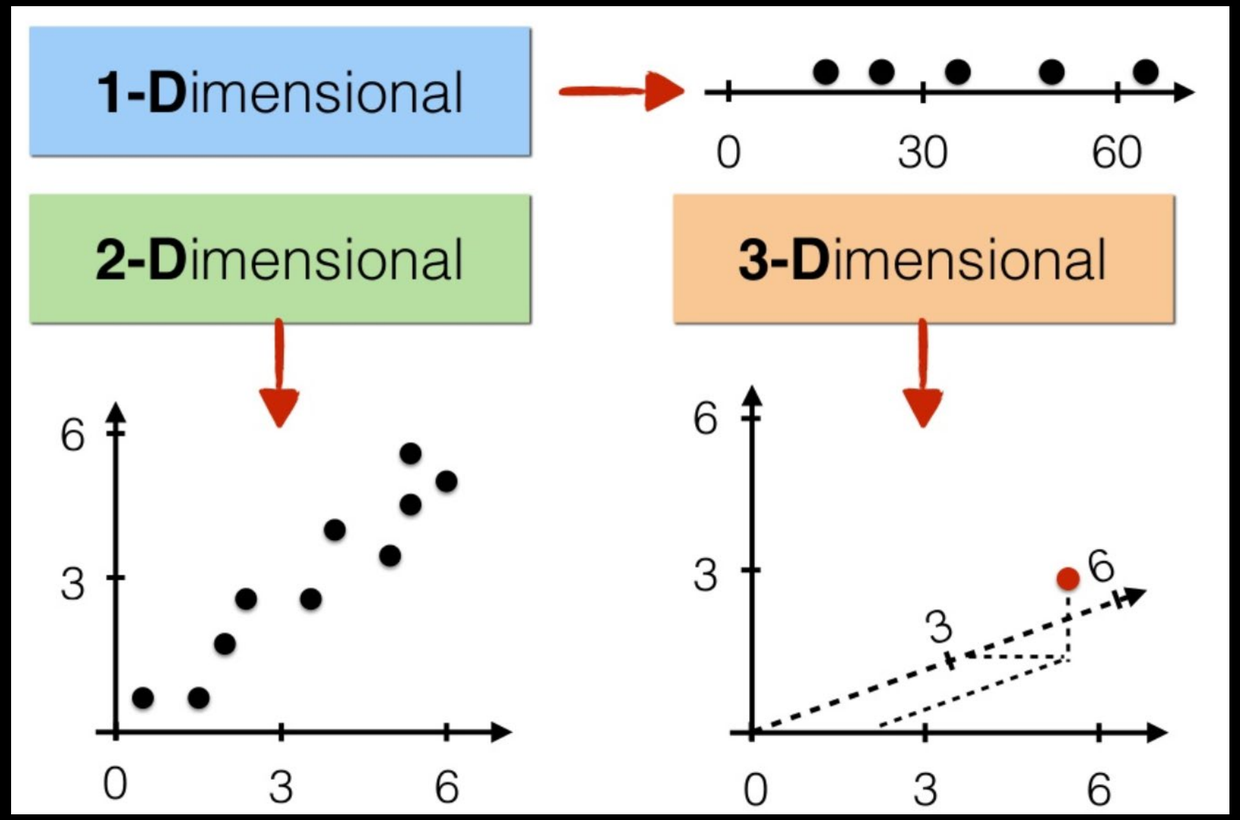
A general single cell analysis workflow

What is dimension?

feature (genes)

X_1	X_2	X_p

Sample (cell)



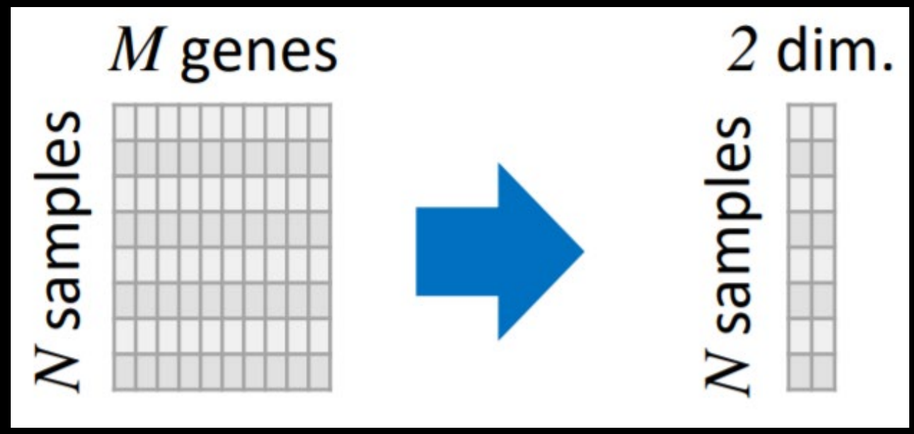
What is dimension reduction?

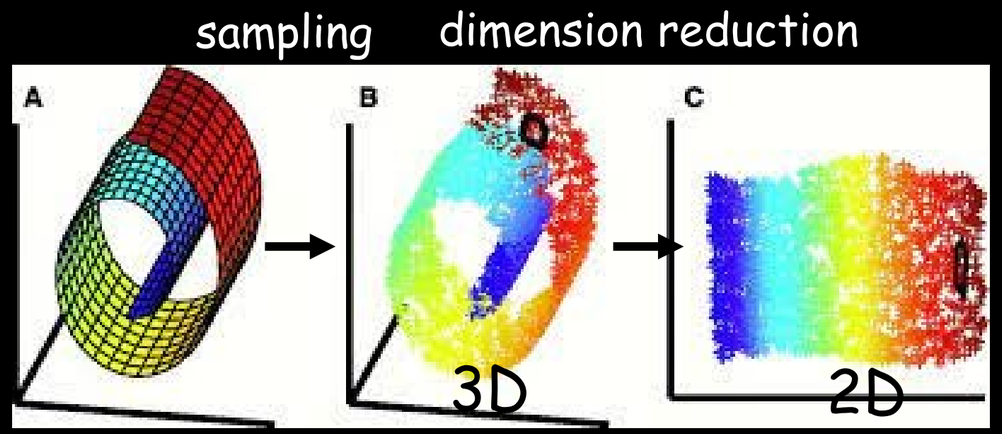
feature (genes)

Sample (cell)

X_1	X_2	X_p

dimension: p

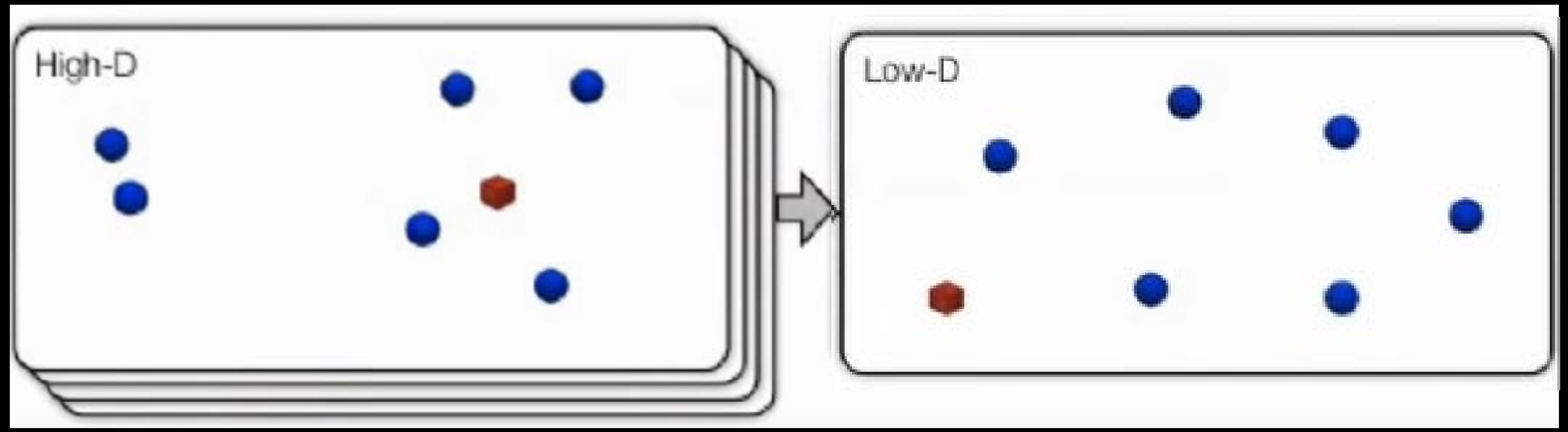




Aim: low-dimensional representation retains meaningful properties of the original data, ideally close to its intrinsic dimension.
- i.e. distances are preserved as well as possible

What is dimension reduction?

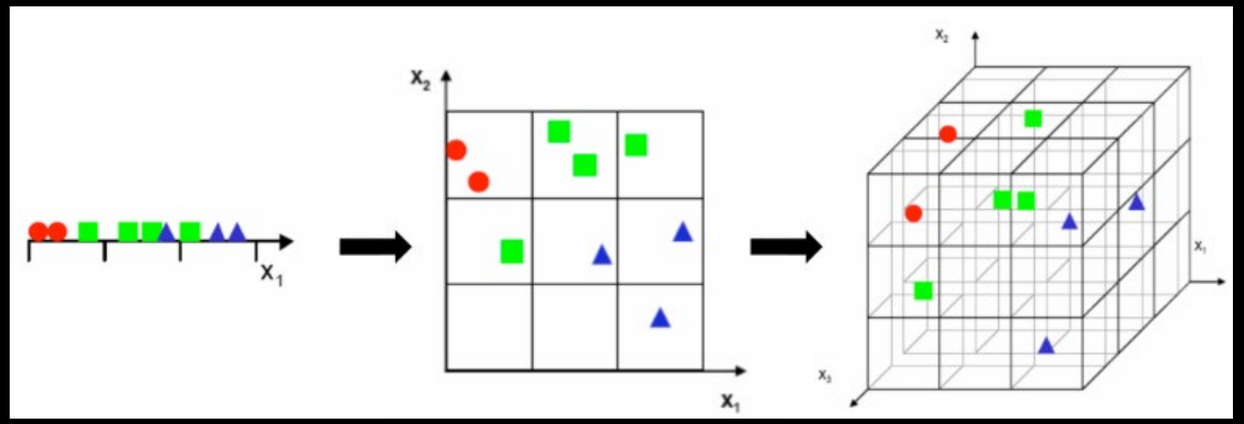
Build a low dimension map in which distances between samples reflecting similarity in high dimension



Minimize some objective function that measure the difference between similarity in high dimension and low dimension

Why we need dimension reduction?

Curse of dimensionality

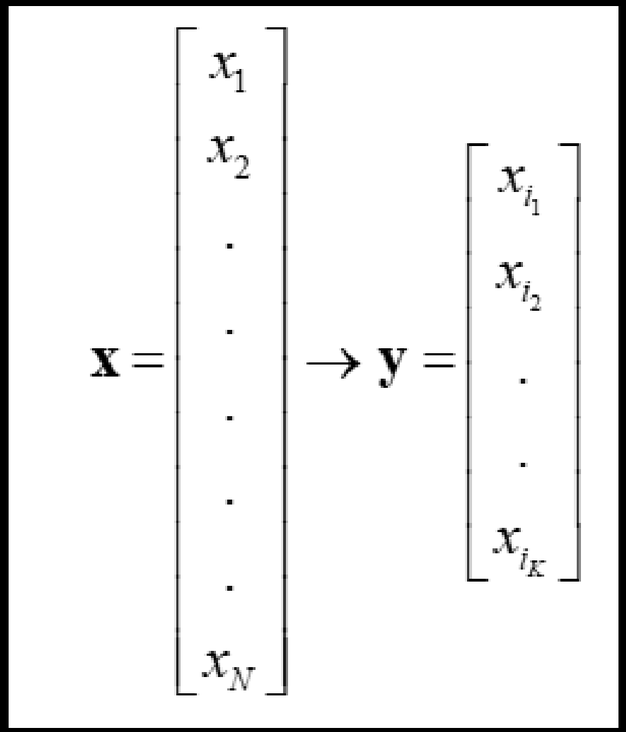


Statistics need repetition!

1. Remove redundant and irrelevant features, captures the "essence" of the data.
2. Simpler to compute and analytically tractable. (clustering)
3. Visualize high dimensional data. (2D;3D)

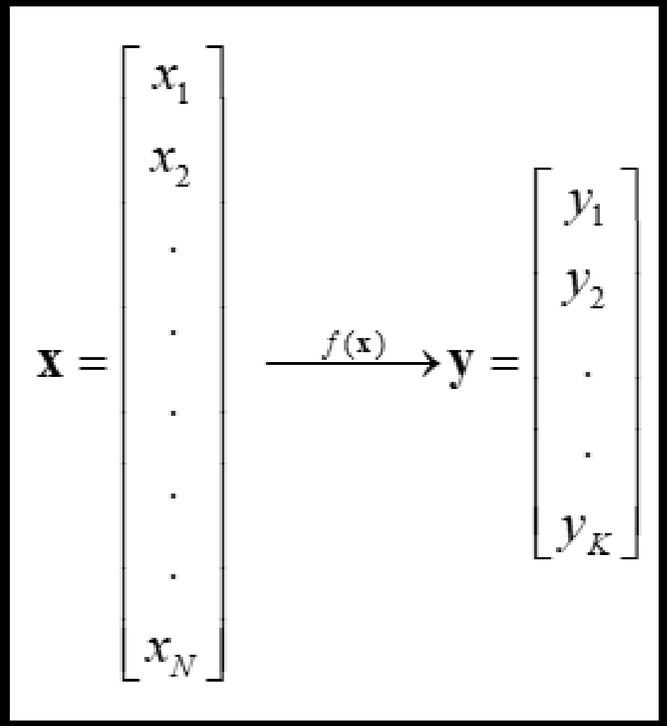
What is dimension reduction?

Feature selection: chooses a subset of the original features.



K<<N

Feature extraction: finds a set of new features (i.e., through some mapping f()) from the existing features



K<<N

The mapping f() could be linear or non-linear

Dimension reduction methods

PCA	linear	Matrix Factorization	1901
ICA	linear	Matrix Factorization	
MDS	non-linear	Matrix Factorization	
Sparce NNMF	non-linear	Matrix Factorization	2010
cPCA	non-linear	Matrix Factorization	2018
ZIFA	non-linear	Matrix Factorization	2015
ZINB-WaVE	non-linear	Matrix Factorization	2018
Diffusion maps	non-linear	graph-based	2005
Isomap	non-linear	graph-based	2000
t-SNE	non-linear	graph-based	2008
- BH t-SNE	non-linear	graph-based	2014
- Flt-SNE	non-linear	graph-based	2017
LargeVis	non-linear	graph-based	2018
UMAP	non-linear	graph-based	2018
PHATE	non-linear	graph-based	2017
scvis	non-linear	Autoencoder (MF)	2018
VASC	non-linear	Autoencoder (MF)	2018

PCA: Seek a projection preserving as much as information in the low-dimension.

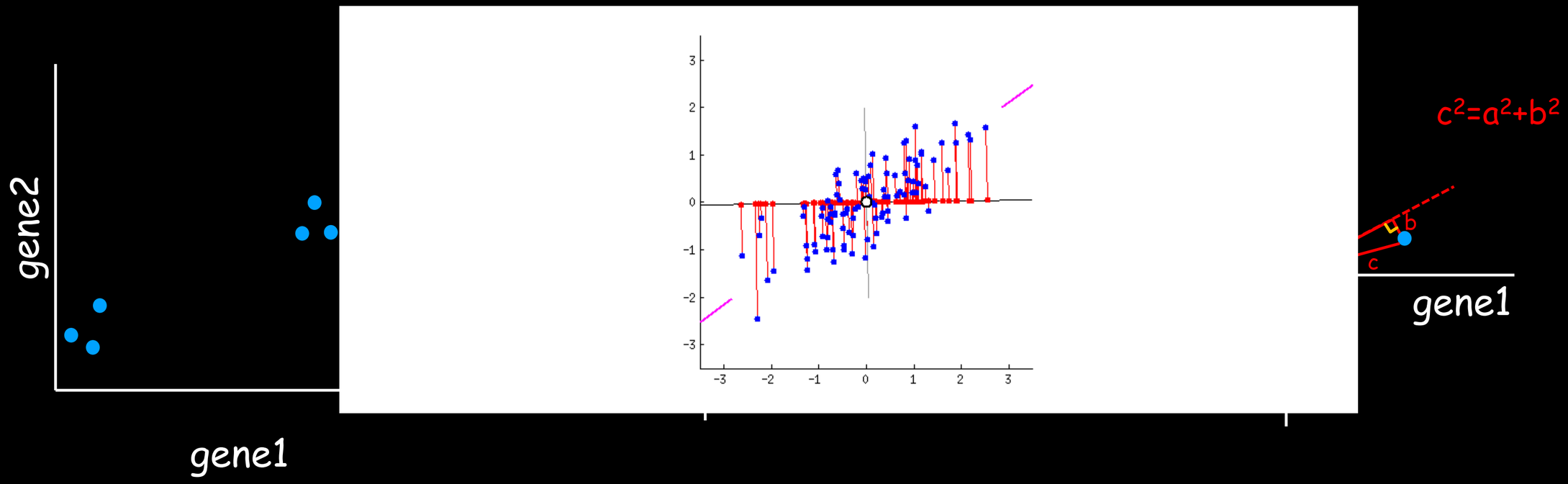
ICA: Making features as independent as possible.

t-SNE: Distance in low dimension between samples reflecting similarities in high dimension

Isomap, UMAP: Finding the low dimensional manifold

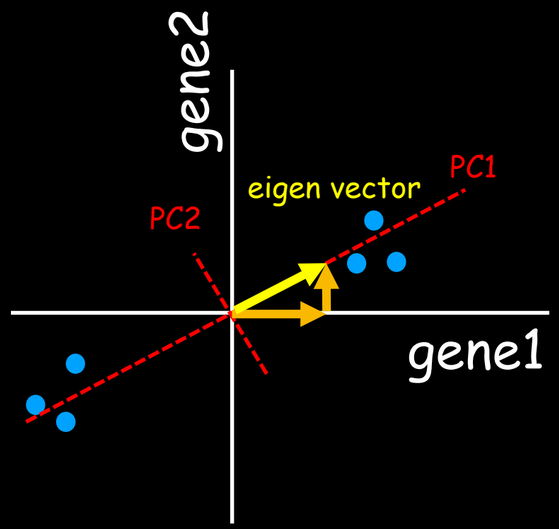
PCA: principle component analysis

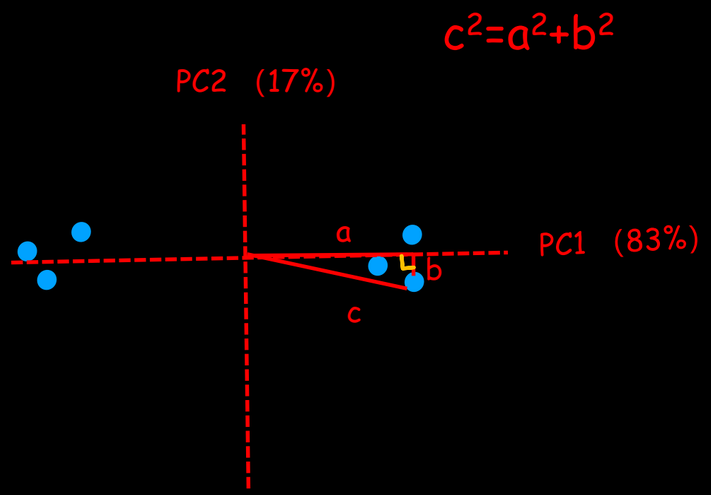
Seek a linear projection preserving as much as information in the low-dimension.



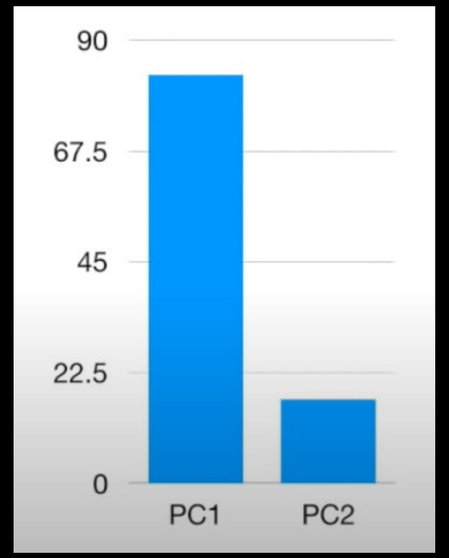
maximizing variance=minimizes the squared error

PCA: principle component analysis





Scree plot



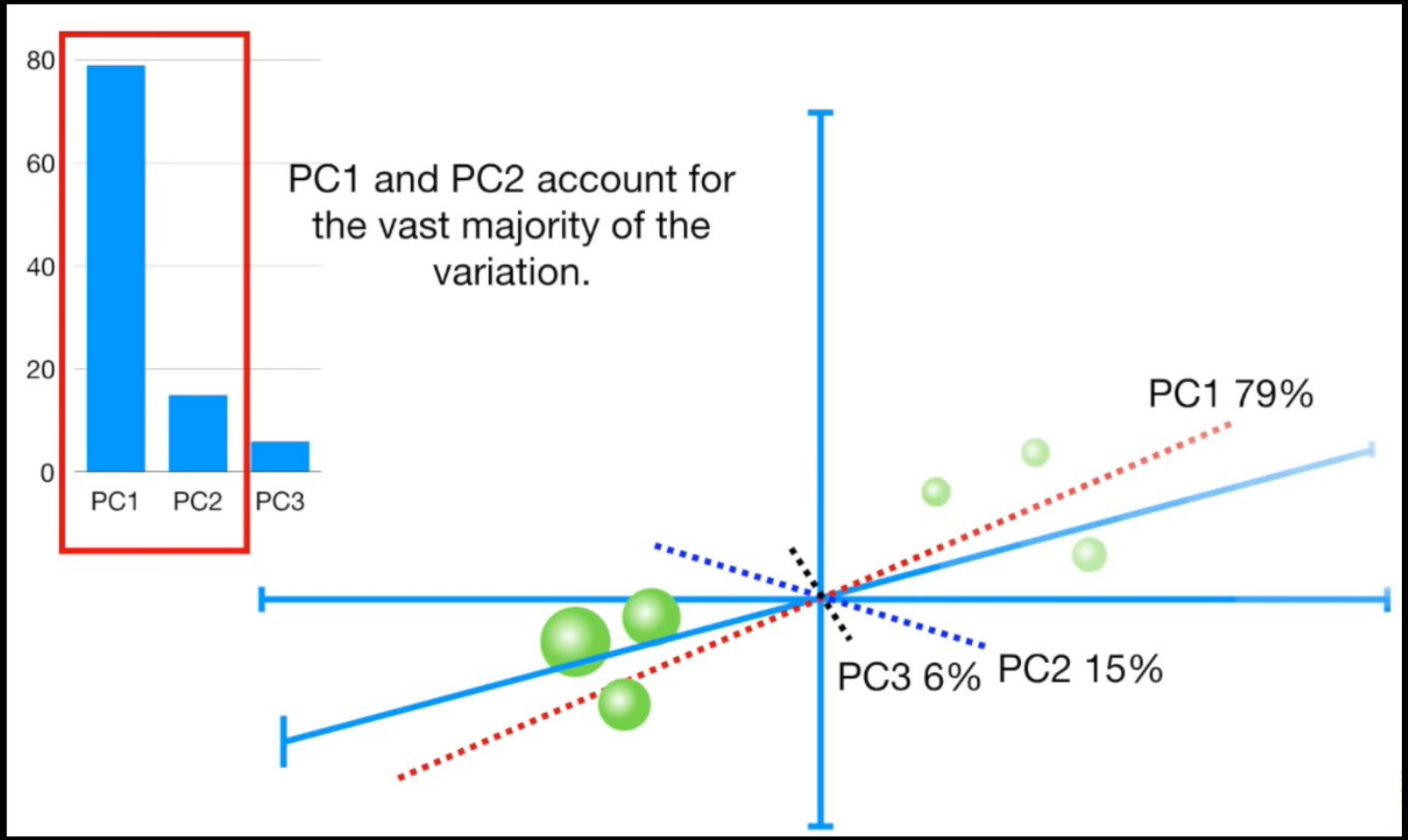
Loading scores : PC1 direction takes
3 (0.95) part of gene1 and
1 (0.32) part of gene2.

$(a_1^2 + a_2^2 + a_3^2 + a_4^2 + a_5^2 + a_6^2)/(n-1)$=Variation for PC1

$(b_1^2 + b_2^2 + b_3^2 + b_4^2 + b_5^2 + b_6^2)/(n-1)$=Variation for PC2

An eigen-decomposition of the covariance matrix of X.
or singular value decomposition (SVD) of the data matrix X.

PCA: principle component analysis

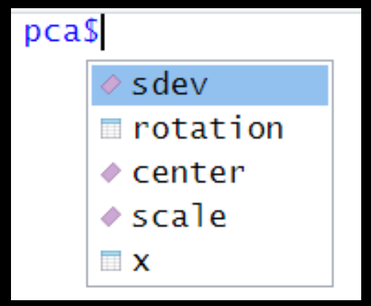


PCA in R

PCA implementation prcomp & sklearn.decomposition.PCA & Seurat

```
> head(data.matrix)
      wt1 wt2 wt3 wt4 wt5 ko1 ko2 ko3 ko4 ko5
gene1 955 936 947 964 873 213 196 192 187 204
gene2 737 746 768 794 768 598 529 555 514 487
gene3  15  22  17  12  24 706 747 765 763 788
gene4 376 381 372 375 389 203 155 211 192 175
gene5  93  91  85 109  90 750 790 773 690 751
gene6 620 600 621 593 637 774 807 752 759 780
```

pca <- prcomp(t(data.matrix), scale=TRUE)




```
> pca$x
            PC1        PC2        PC3
wt1  -9.334055  2.0689554  0.6708543
wt2  -8.948258 -0.8777832 -2.3414271
wt3  -9.081600  1.3803095  1.0274838
wt4  -9.161428 -1.0619666 -0.9767081
wt5  -9.154765 -1.4815547  1.5380232
ko1   8.947310 -0.7564945  0.6500502
ko2   9.239133 -1.0010251 -0.3132664
ko3   9.295162  2.0750376 -0.6398703
ko4   8.785678 -1.0865512  1.3128123
ko5   9.412823  0.7410729 -0.9279518
```

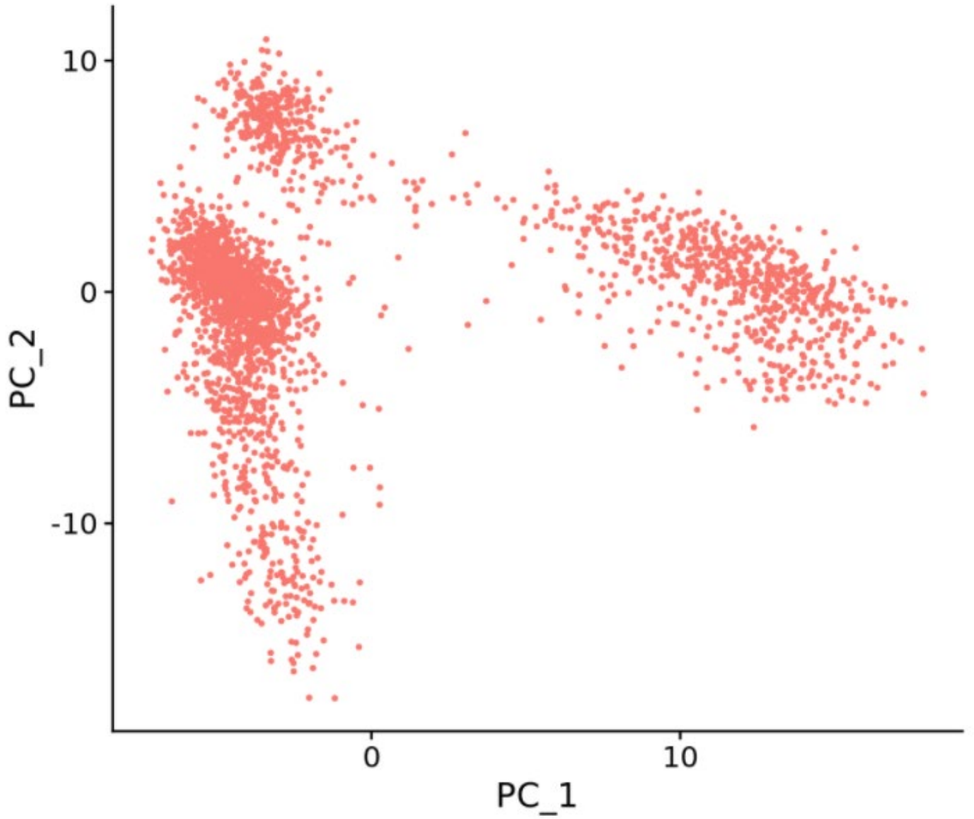
```
> pca$rotation
                PC1           PC2
gene1   -0.10358449  0.0191589459
gene2   -0.10041501 -0.0357415770
gene3    0.10370520  0.0068182441
gene4   -0.10265363  0.0178409733
gene5    0.10360275  0.0053777231
gene6    0.10143657 -0.0257539608
gene7    0.10377272 -0.0094503181
gene8    0.10275326 -0.0619288577
gene9    0.10339587 -0.0209273030
gene10  -0.10034875  0.0332112204
```

```
> summary(pca)
Importance of components:
                          PC1     PC2     PC3
Standard deviation     9.6322 1.40985 1.23928
Proportion of Variance 0.9278 0.01988 0.01536
Cumulative Proportion  0.9278 0.94767 0.96303
```

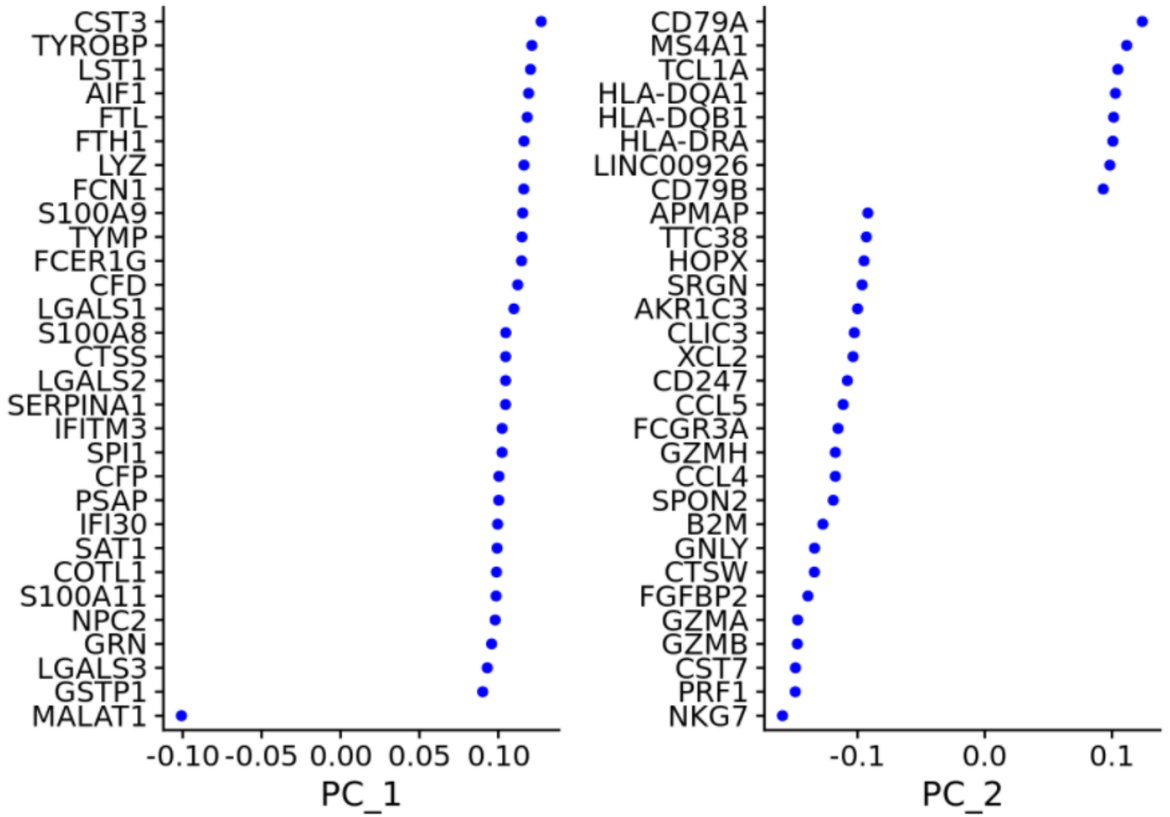
https://github.com/StatQuest/pca_demo/blob/master/pca_demo.R

PCA in single cell RNA-seq data

```
DimPlot(pbmc, reduction = "pca")
```

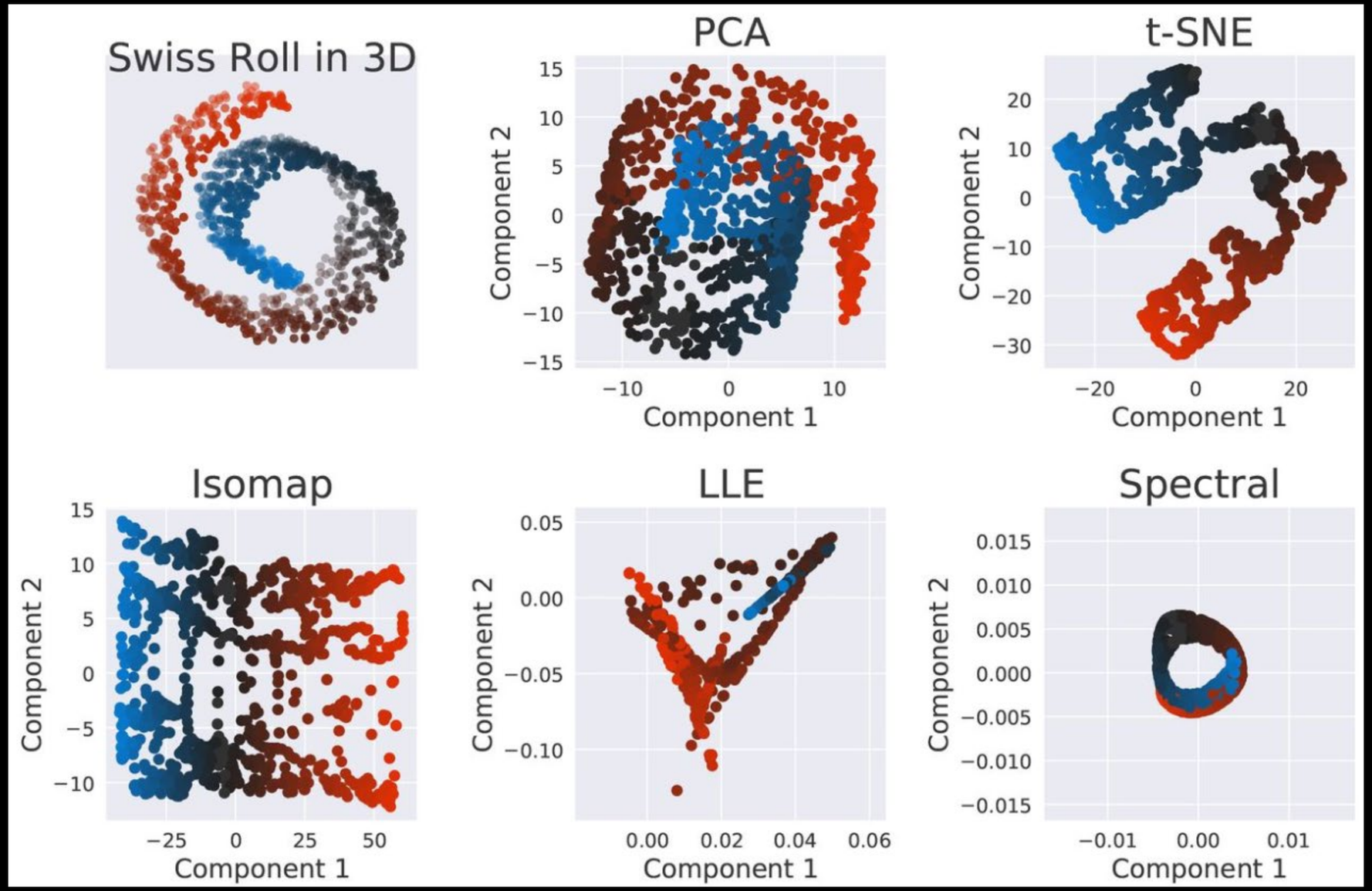



```
VizDimLoadings(pbmc, dims = 1:2, reduction = "pca")
```



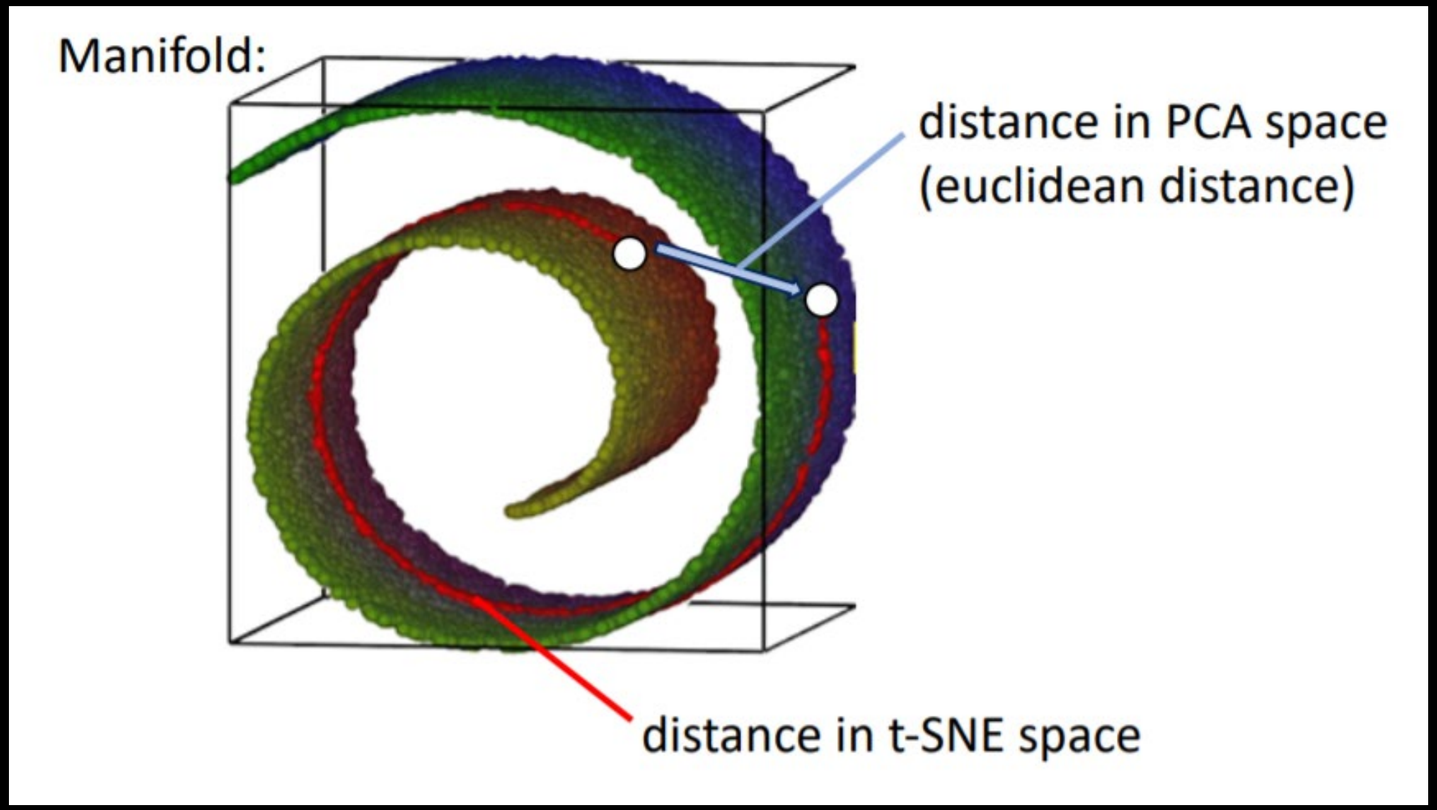
Seurat - Guided Clustering Tutorial

PCA applied at Swiss Roll dataset



tSNE: t-distributed stochastic neighbor embedding

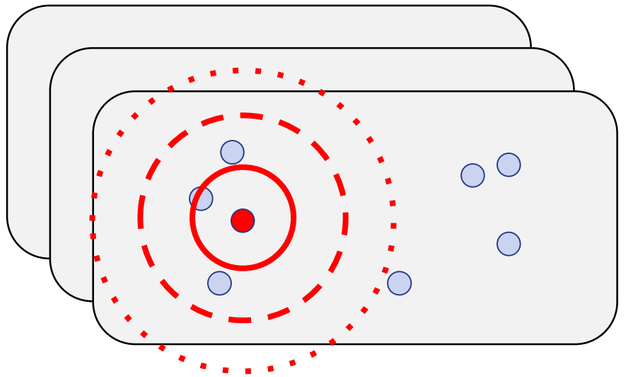
It is a graph-based NON-LINEAR dimensionality reduction



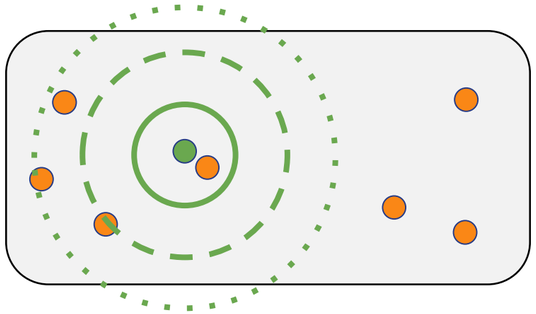
PCA preserves the large pairwise distances in the map.

→ tSNE preserves small pairwise distance

How t-SNE works



High-Dimensional Space



Low-Dimensional Space

Similarity Computation → Embedding Optimization

$X = \{\mathbf{x}_1 \ldots \mathbf{x}_N\}$

$p_{ij} =$ Similarity between *i* and *j in H-Dim*

$P =$ Probability distribution encoding similarities

$Y = \{\mathbf{y}_1 \ldots \mathbf{y}_N\}$

$q_{ij} =$ Similarity between *i* and *j in L-Dim*

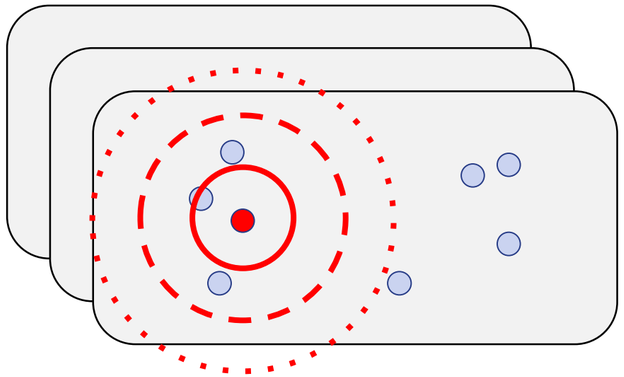
$Q =$ Probability distribution encoding similarities

Entropy-based cost function

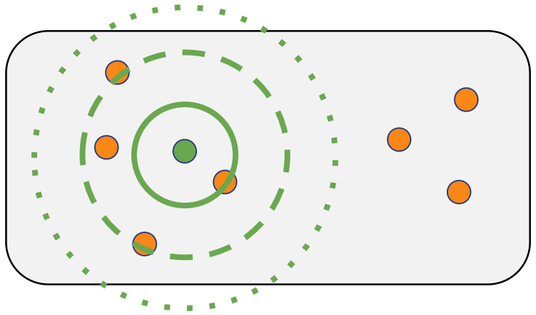
$$C(P,Q) = KL(P||Q)$$

$$\frac{\delta C}{\delta \mathbf{y}_i} = 4(F_i^{\text{attr}} - F_i^{\text{rep}})$$

How t-SNE works



High-Dimensional Space



Low-Dimensional Space

Similarity Computation → Embedding Optimization

$X = \{\mathbf{x}_1 \ldots \mathbf{x}_N\}$

$p_{ij} =$ Similarity between *i* and *j in H-Dim*

$P =$ Probability distribution encoding similarities

$Y = \{\mathbf{y}_1 \ldots \mathbf{y}_N\}$

$q_{ij} =$ Similarity between *i* and *j in L-Dim*

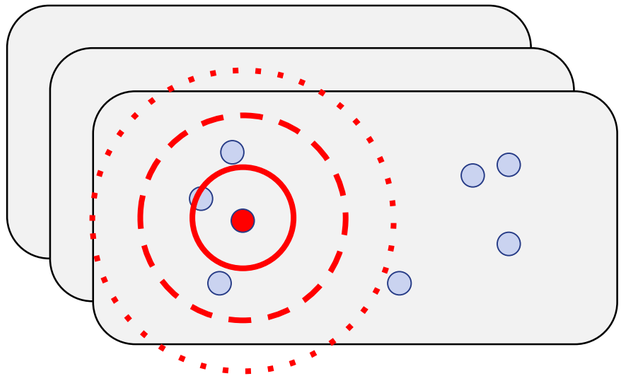
$Q =$ Probability distribution encoding similarities

Entropy-based cost function

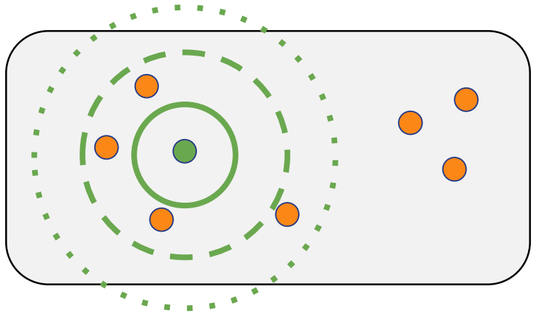
$$C(P,Q) = KL(P||Q)$$

$$\frac{\delta C}{\delta \mathbf{y}_i} = 4(F_i^{\text{attr}} - F_i^{\text{rep}})$$

How t-SNE works



High-Dimensional Space



Low-Dimensional Space

Similarity Computation → Embedding Optimization

$X = \{\mathbf{x}_1 \ldots \mathbf{x}_N\}$

$p_{ij} =$ Similarity between *i* and *j in H-Dim*

$\boldsymbol{P} =$ Probability distribution encoding similarities

$Y = \{\mathbf{y}_1 \ldots \mathbf{y}_N\}$

$q_{ij} =$ Similarity between *i* and *j in L-Dim*

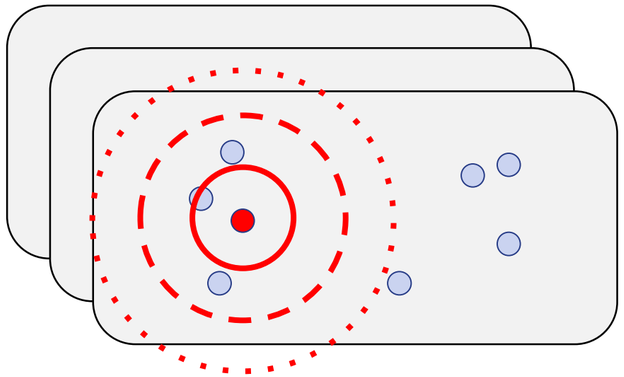
$\boldsymbol{Q} =$ Probability distribution encoding similarities

Entropy-based cost function

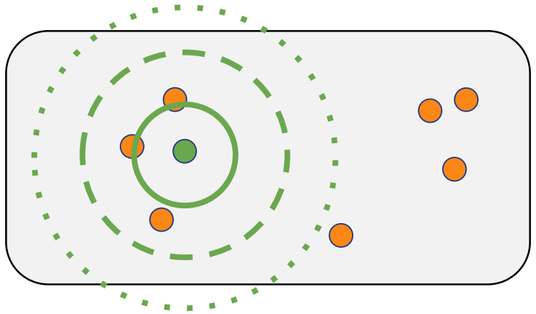
$$C(P,Q) = KL(P||Q)$$

$$\frac{\delta C}{\delta \mathbf{y}_i} = 4(F_i^{\text{attr}} - F_i^{\text{rep}})$$

How t-SNE works



High-Dimensional Space



Low-Dimensional Space

Similarity Computation → Embedding Optimization

$$X = \{\mathbf{x}_1 \ldots \mathbf{x}_N\}$$

$p_{ij} =$ Similarity between *i* and *j in H-Dim*

$\boldsymbol{P} =$ Probability distribution encoding similarities

$$Y = \{\mathbf{y}_1 \ldots \mathbf{y}_N\}$$

$q_{ij} =$ Similarity between *i* and *j in L-Dim*

$\boldsymbol{Q} =$ Probability distribution encoding similarities

Entropy-based cost function

$$C(P,Q) = KL(P||Q)$$

$$\frac{\delta C}{\delta \mathbf{y}_i} = 4(F_i^{\text{attr}} - F_i^{\text{rep}})$$

How t-SNE works

$$p_{ij} = \frac{\exp\left(-\|x_i - x_j\|^2 / 2\delta^2\right)}{\sum_k \sum_{l \neq k} \exp\left(-\|x_k - x_l\|^2 / 2\delta^2\right)}$$

$$q_{ij} = \frac{\left(1 + \|y_i - y_j\|^2\right)^{-1}}{\sum_k \sum_{l \neq k} \left(1 + \|y_k - y_l\|^2\right)^{-1}}$$

$$\mathrm{KL}(\mathrm{P}\|\mathrm{Q}) = \sum_i \sum_{j \neq i} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

Large p_{ij} modeled by small q_{ij} ? Big penalty!

Small p_{ij} modeled by large q_{ij} ? Small penalty!

Hence, t-SNE mainly preserve local similarity structure.

$X = \{\mathbf{x}_1 \ldots \mathbf{x}_N\}$

$p_{ij} =$ Similarity between *i* and *j in H-Dim*

$\boldsymbol{P} =$ Probability distribution encoding similarities

$Y = \{\mathbf{y}_1 \ldots \mathbf{y}_N\}$

$q_{ij} =$ Similarity between *i* and *j in L-Dim*

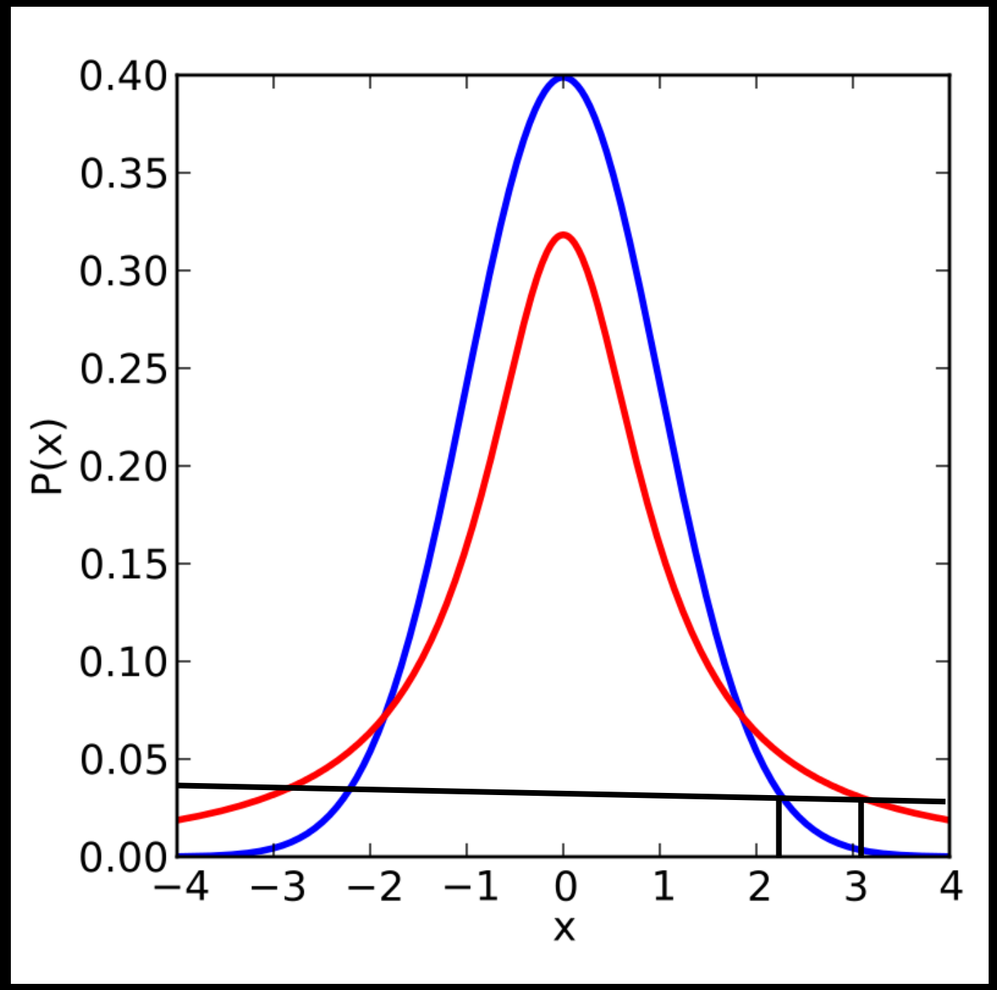
$\boldsymbol{Q} =$ Probability distribution encoding similarities

Entropy-based cost function

$$C(P,Q) = KL(P||Q)$$

$$\frac{\delta C}{\delta \mathbf{y}_i} = 4\left(F_i^{\mathrm{attr}} - F_i^{\mathrm{rep}}\right)$$

Why a Student-t distribution

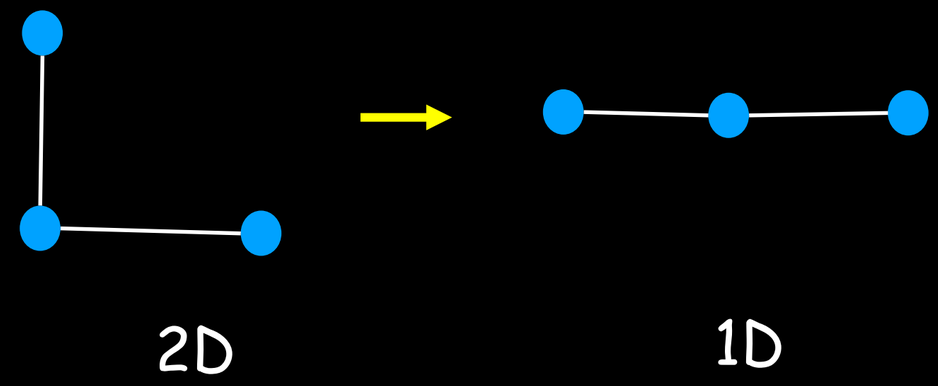


$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_k \sum_{l \neq k} (1 + \|y_k - y_l\|^2)^{-1}}$$

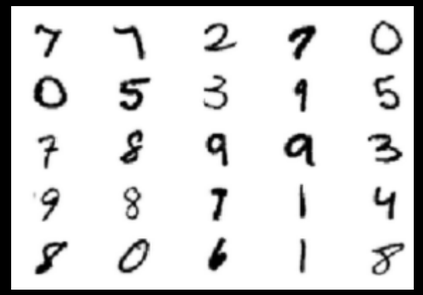
Dissimilar points should be modeled far apart in low dimension.
Avoiding crowd problem

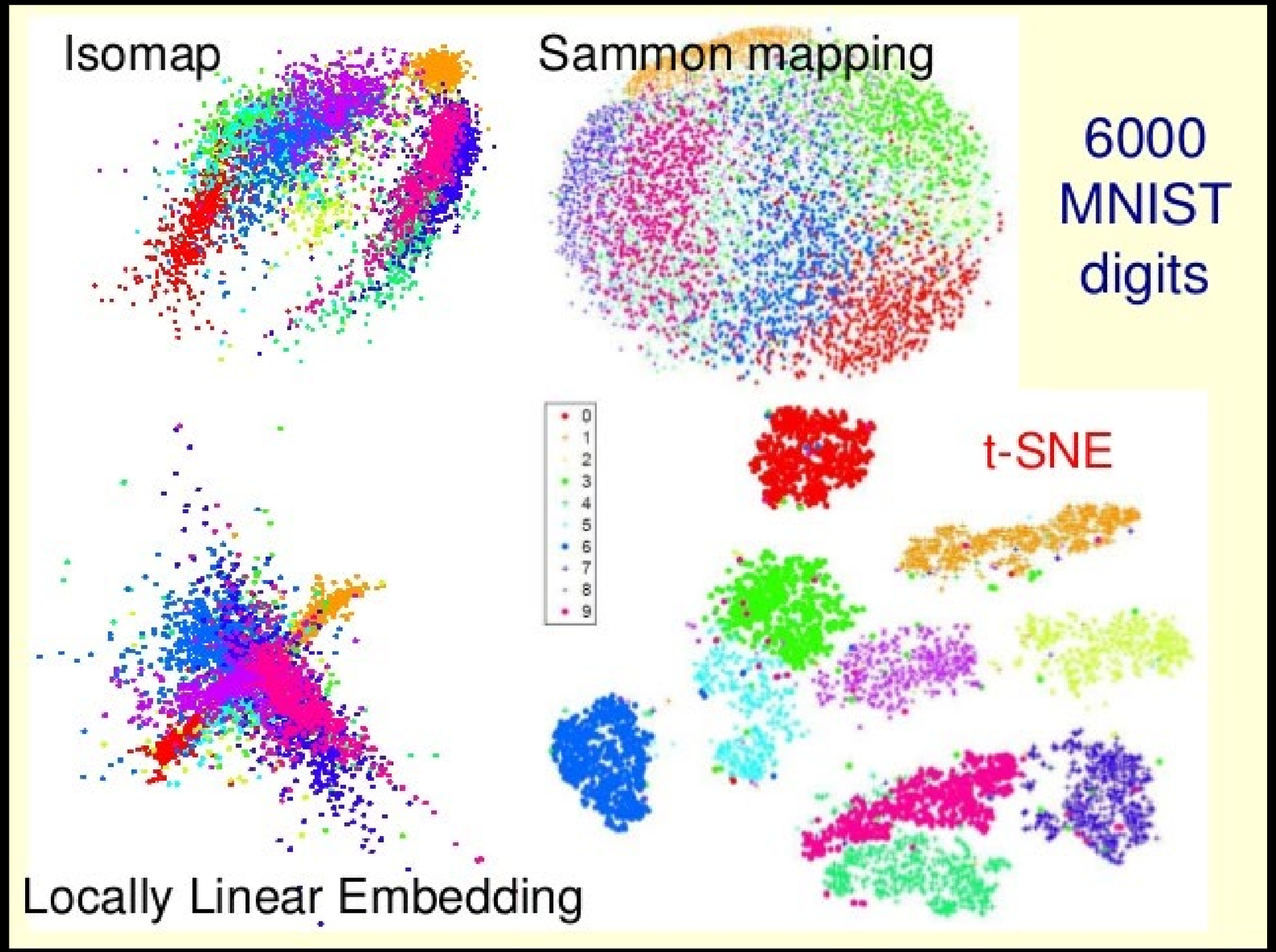
Student's *t*

Gaussian

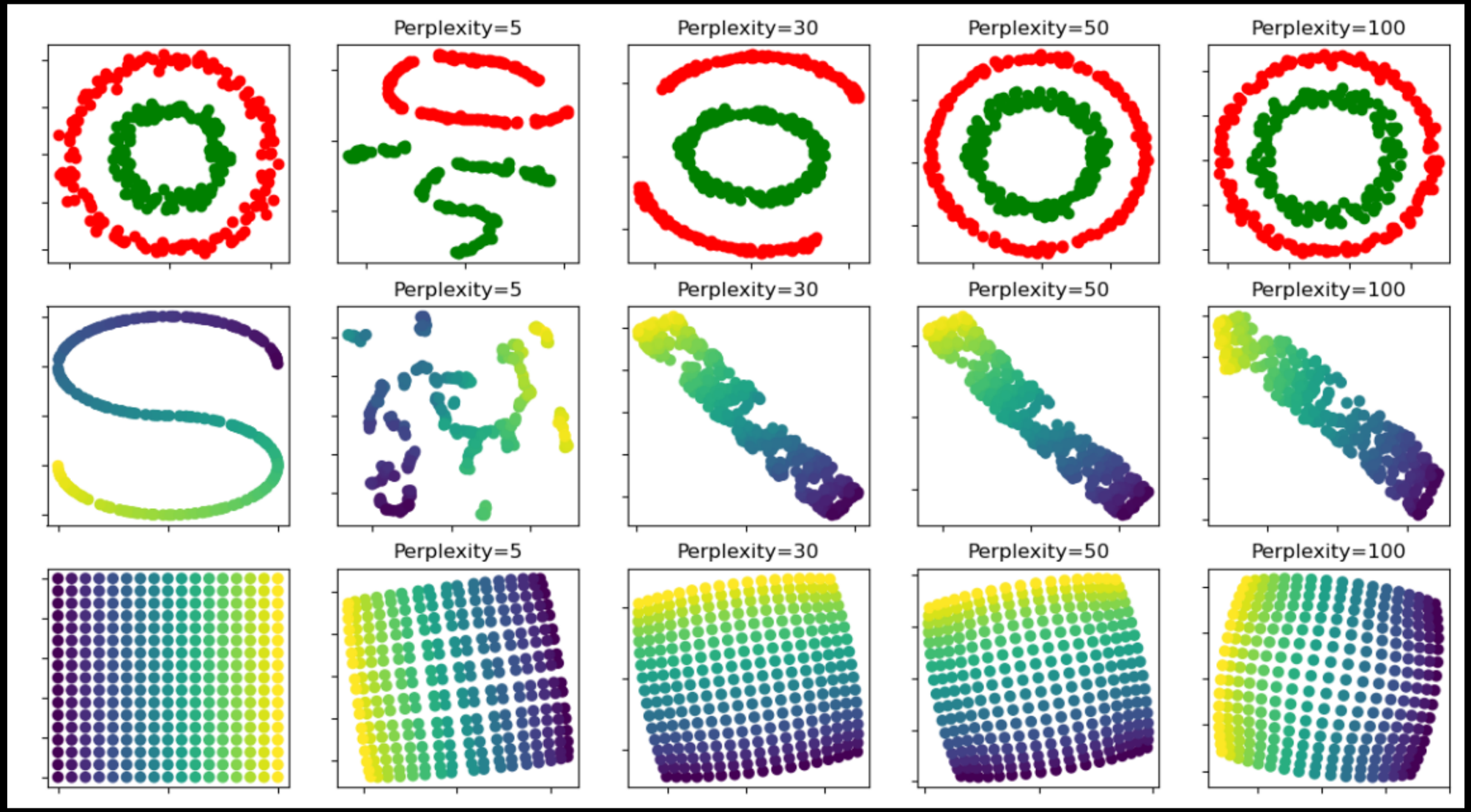


t-SNE on MNIST digits



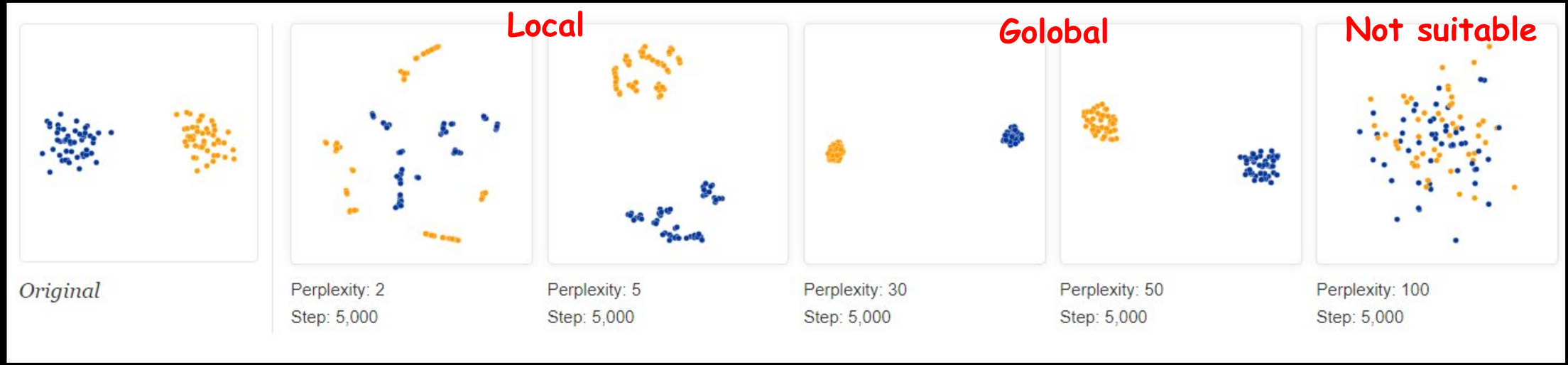


t-SNE on two concentric circles and the S-curve

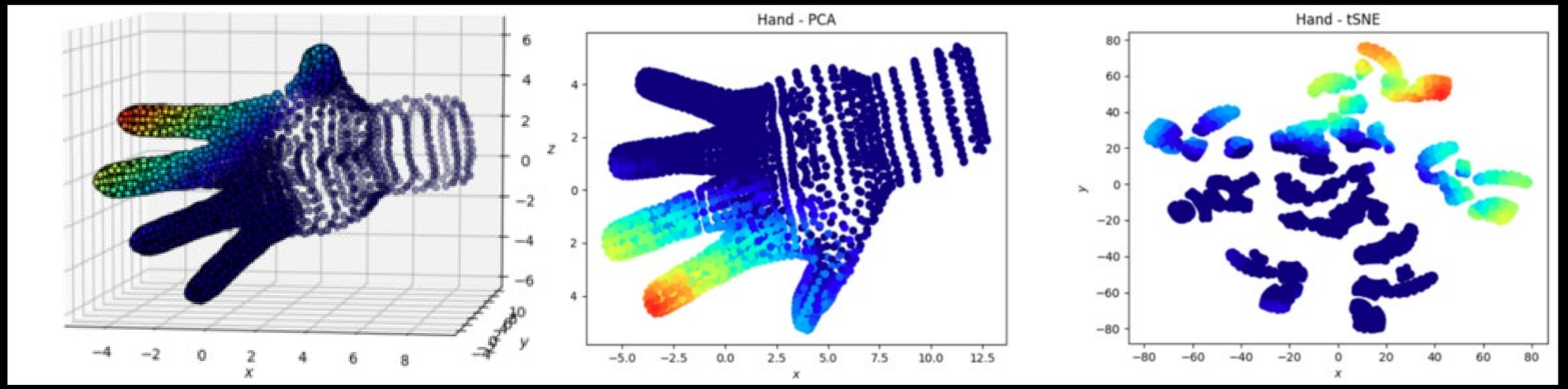


How to Use t-SNE Effectively

Perplexity (hyperparameters) really matter



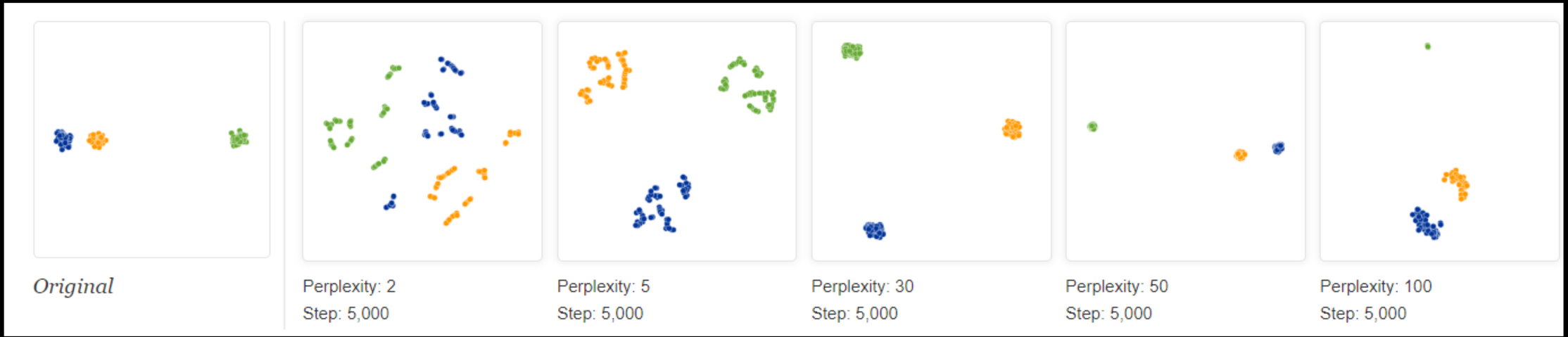
t-SNE captures local structures with low Perplexity



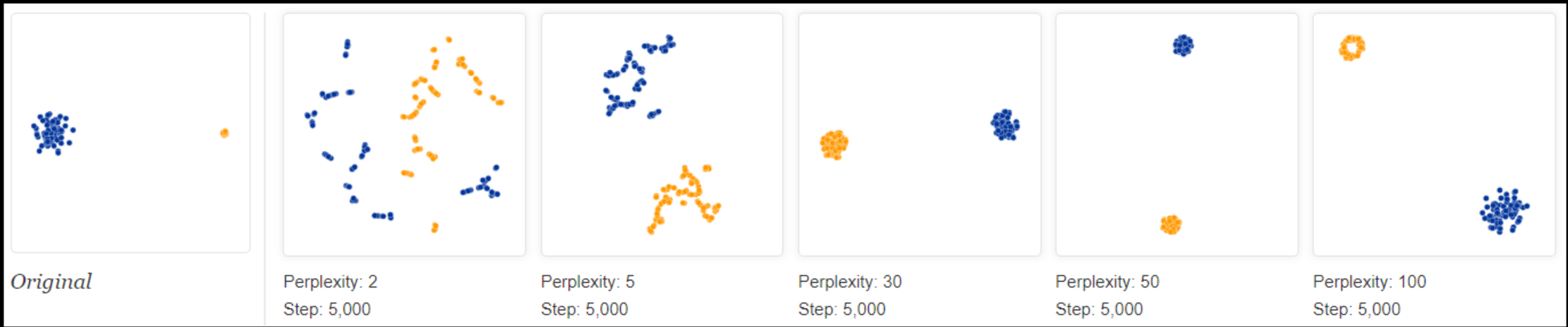
https://distill.pub/2016/misread-tsne/

How to Use t-SNE Effectively

Distances between clusters might not mean anything



Cluster sizes in a t-SNE plot mean nothing



https://distill.pub/2016/misread-tsne/

Important notes about t-SNE

1. tSNE implementation - O(n log n) Rtsne & sklearn.manifold.TSNE & Seurat

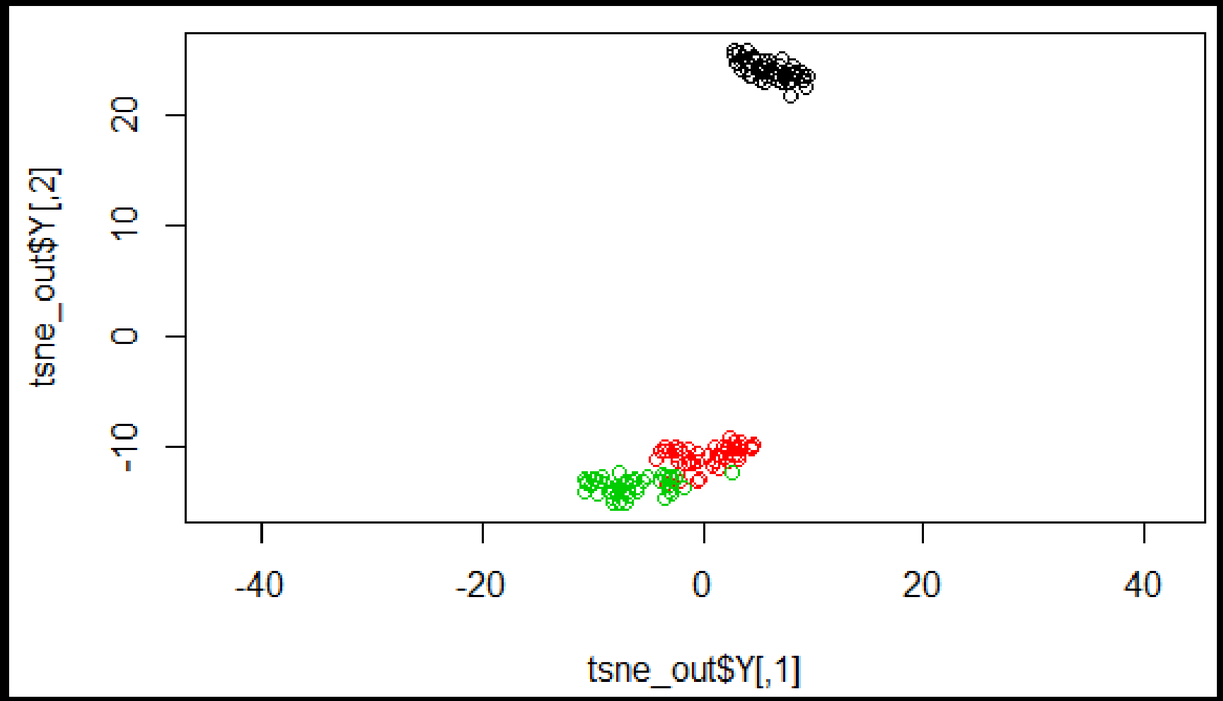
2. It can be run from the top PCs (e.g.: PC1 to PC10)

3. Unlike PCA, it is a stochastic algorithm, so it will never produce the same output (unless you use a seed() to lock the random estimators).

4. To add more samples, you need to re-run the algorithm from start.

tSNE in R

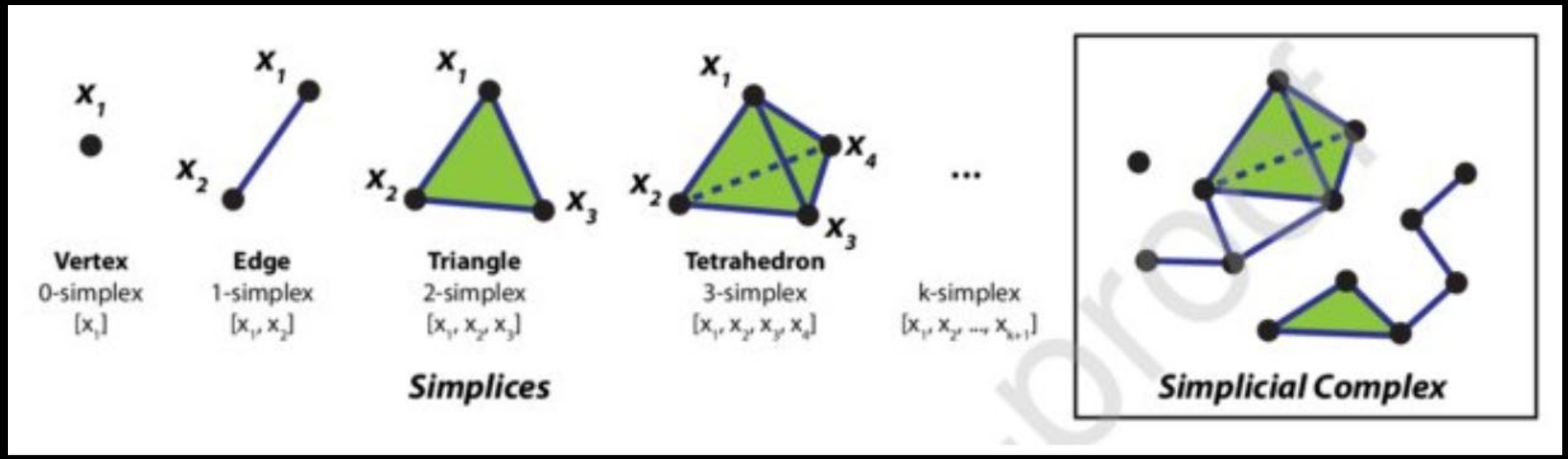
```
library(Rtsne)
iris_unique <- unique(iris) # Remove duplicates
iris_matrix <- as.matrix(iris_unique[,1:4])
# Set a seed if you want reproducible results
set.seed(42)
tsne_out <- Rtsne(iris_matrix,pca=FALSE,perplexity=30,theta=0.0) # Run TSNE
# Show the objects in the 2D tsne representation
plot(tsne_out$Y,col=iris_unique$Species, asp=1)
```



Umap: uniform manifold approximation and projection

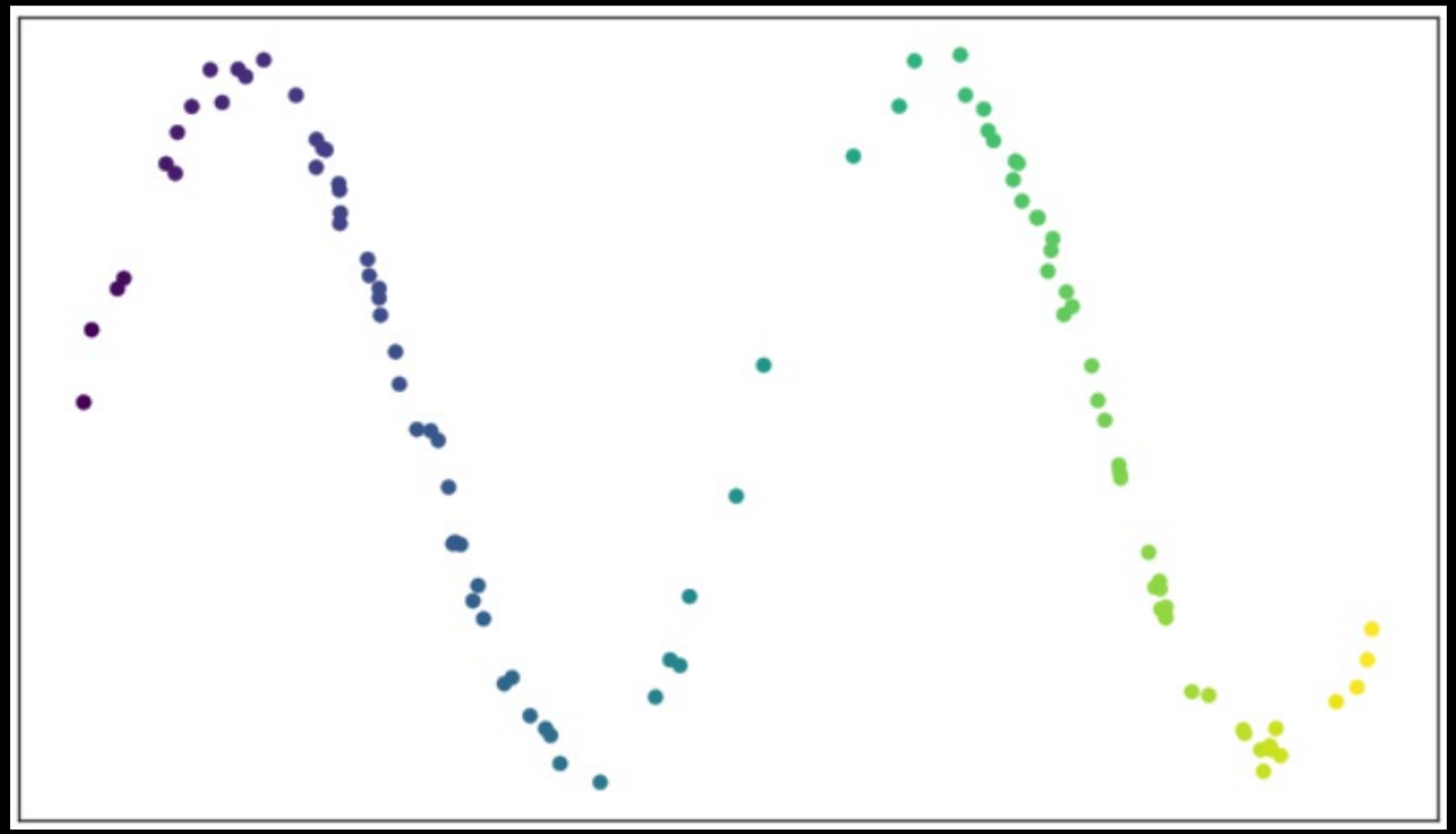
It is based on topological structures in multidimensional space (simplices)
UMAP has strong theoretical foundations.

Low dimensional simplices



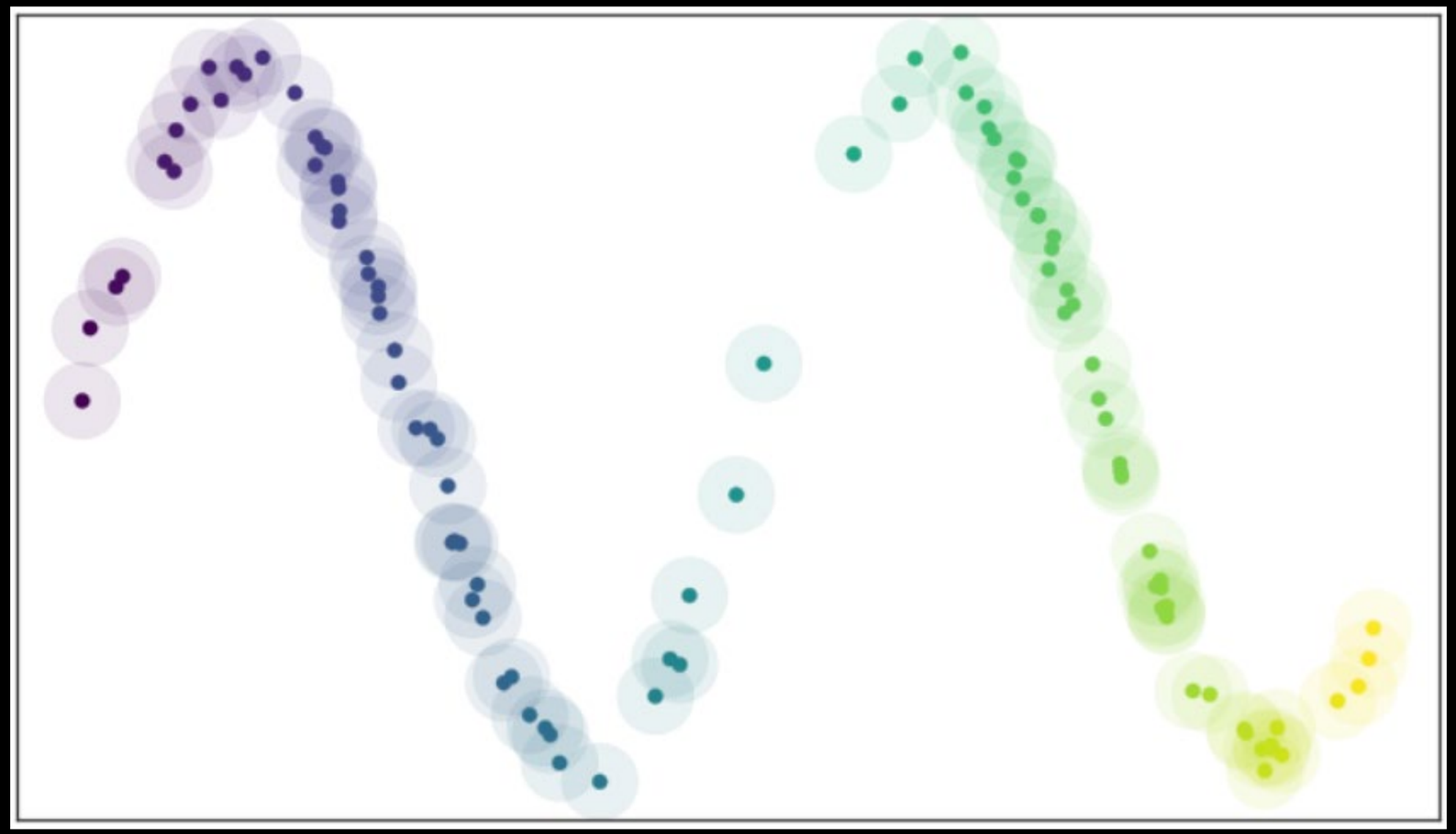
A large class of topological spaces can be constructed by gluing together simplices of various dimensions along their faces. (Nerve theorem)

How to represent data with simplices



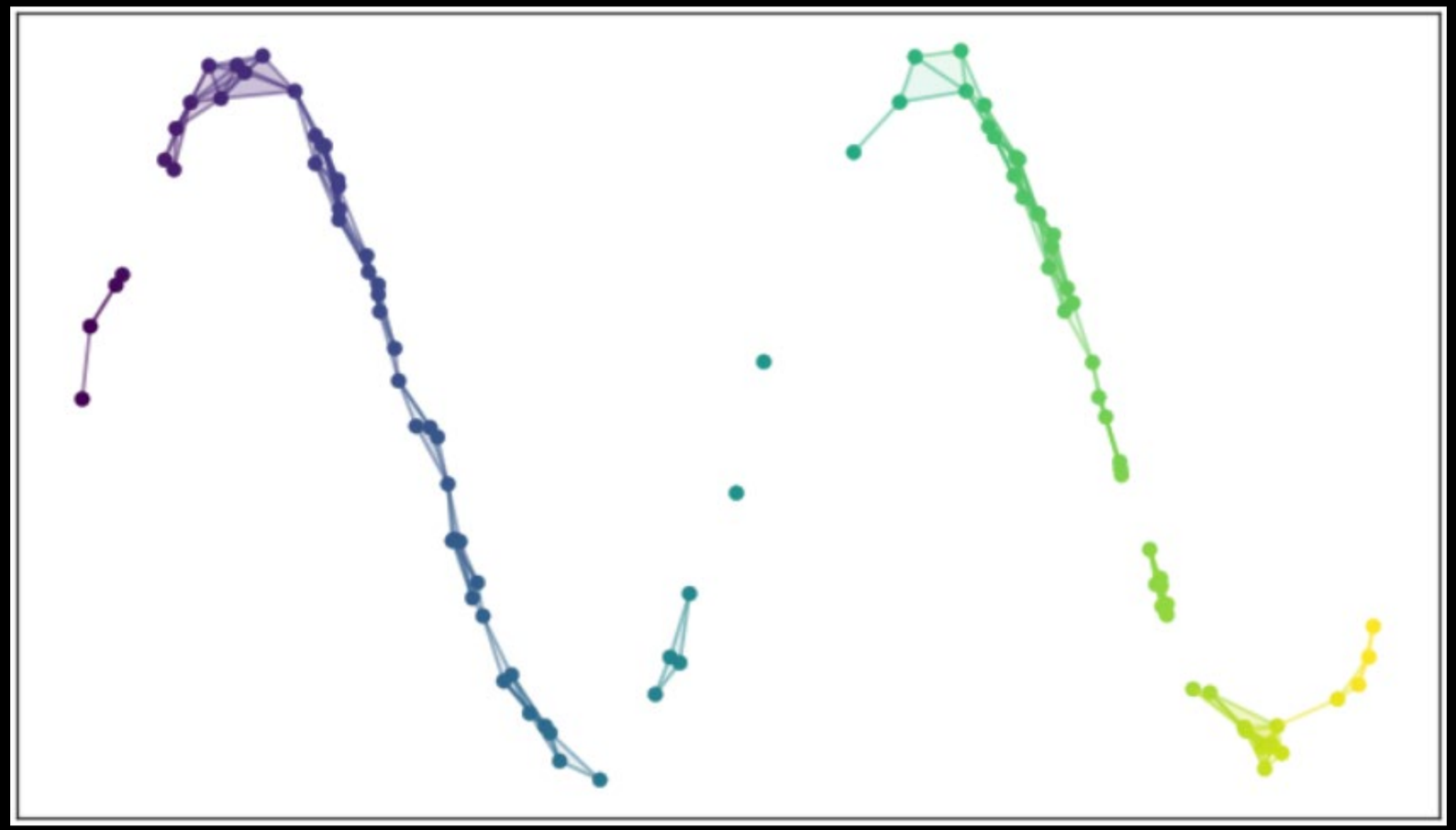
Test data set of a noisy sine wave

How to represent data with simplices



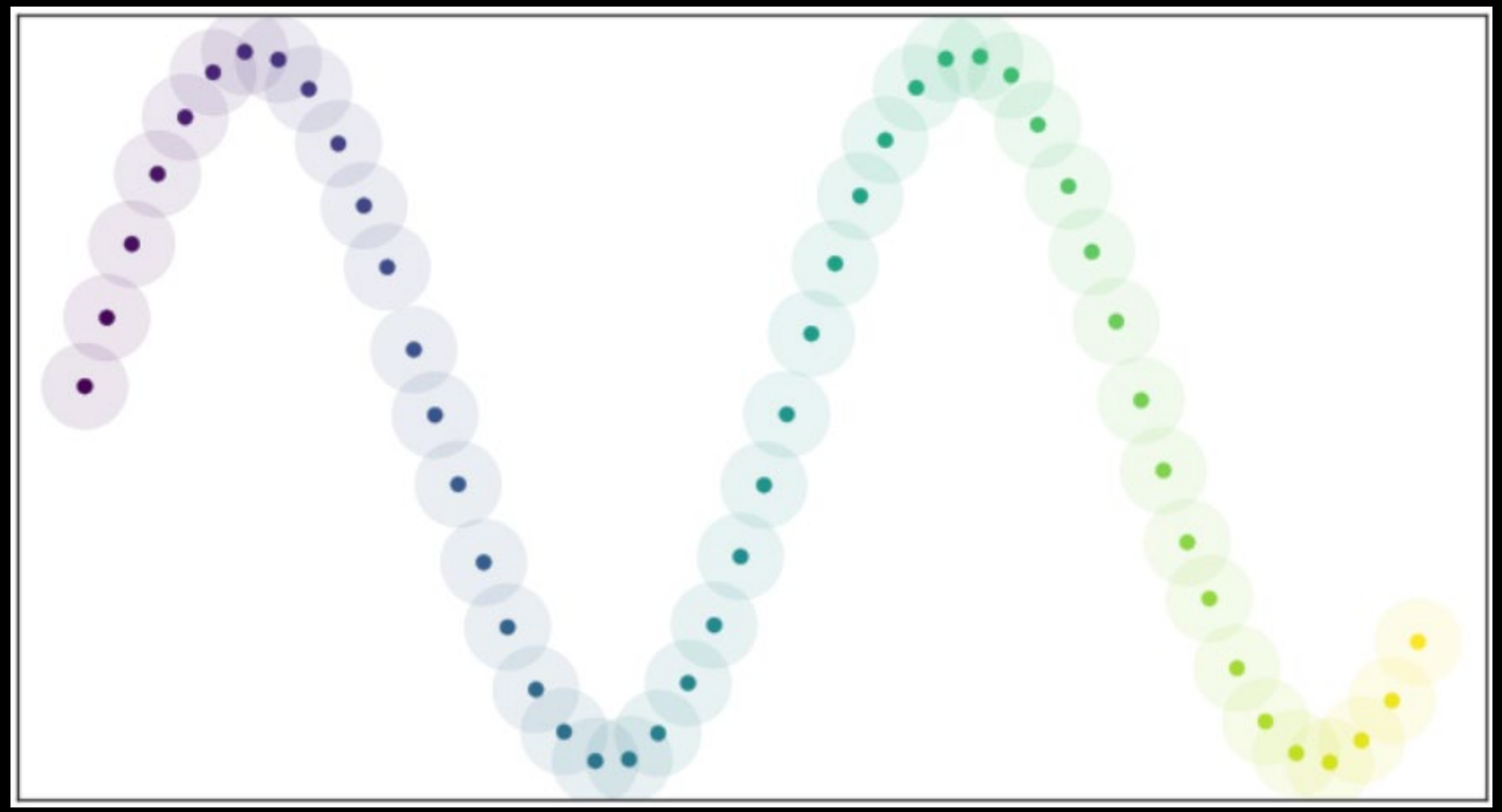
A basic open cover of the test data.

How to represent data with simplices



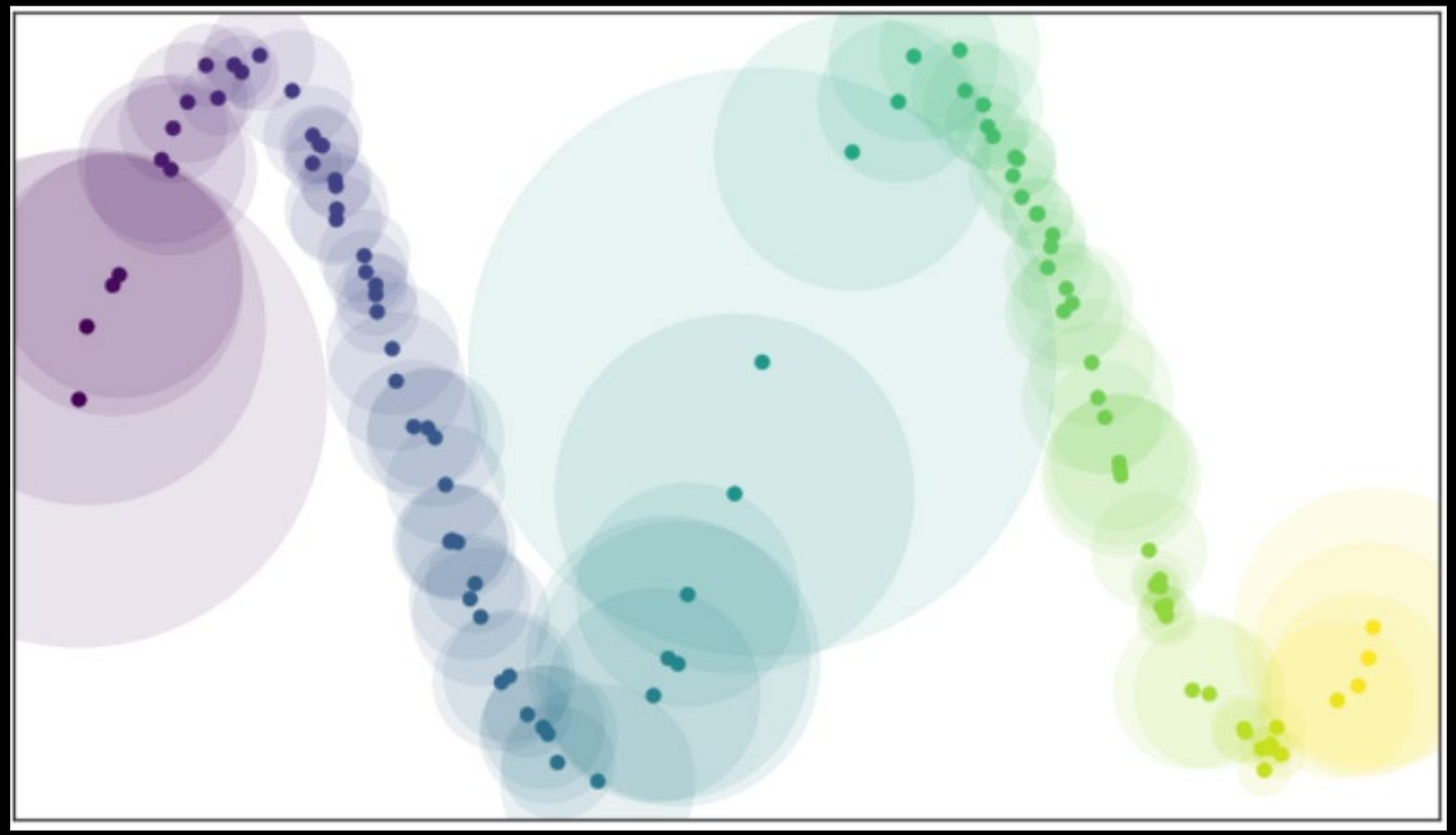
A simplicial complex built from the test data

Adapting topological analysis to real world data



Open balls over uniformly distributed data

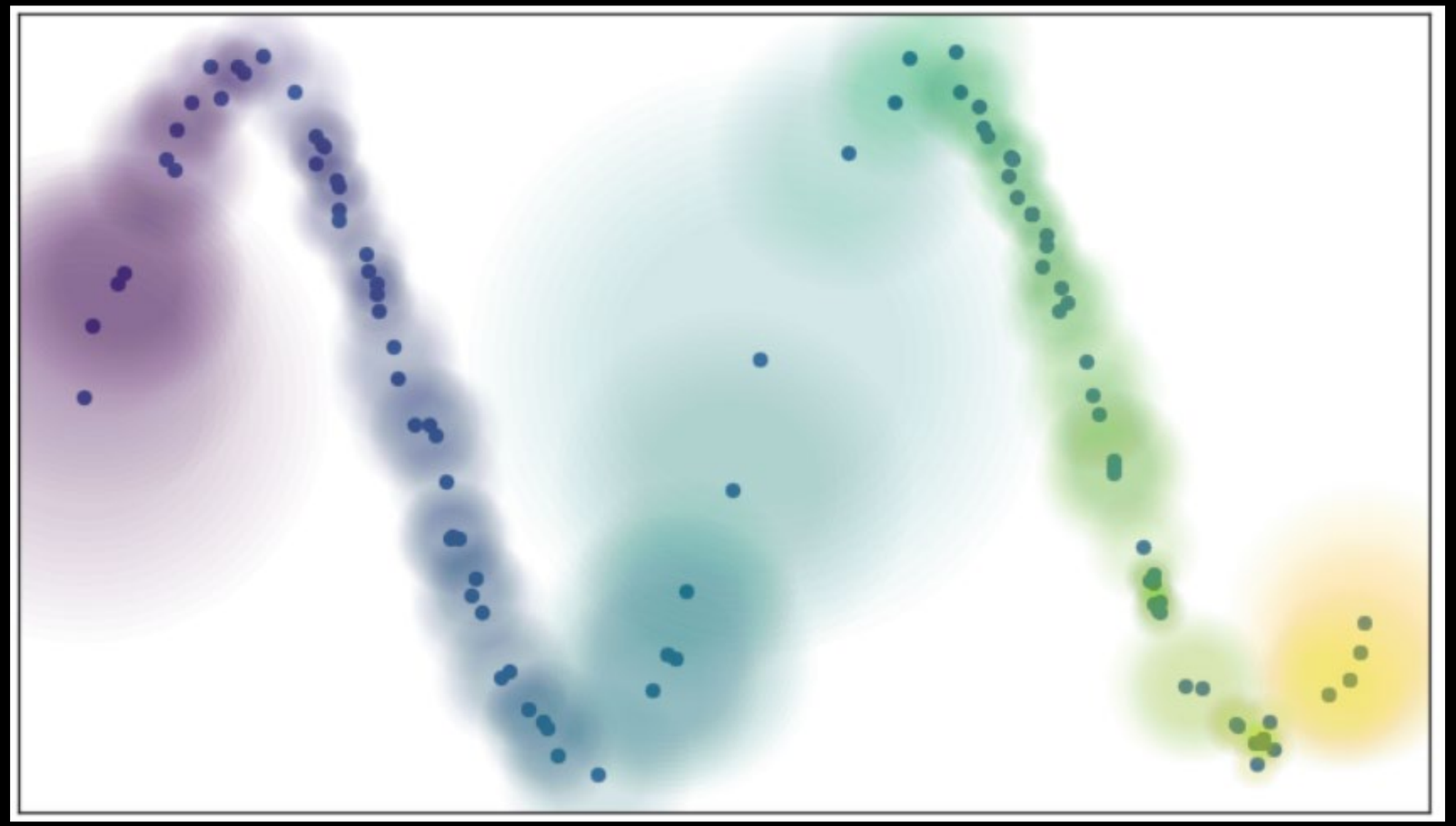
Adapting topological analysis to real world data



Open balls of radius one with a locally varying metric

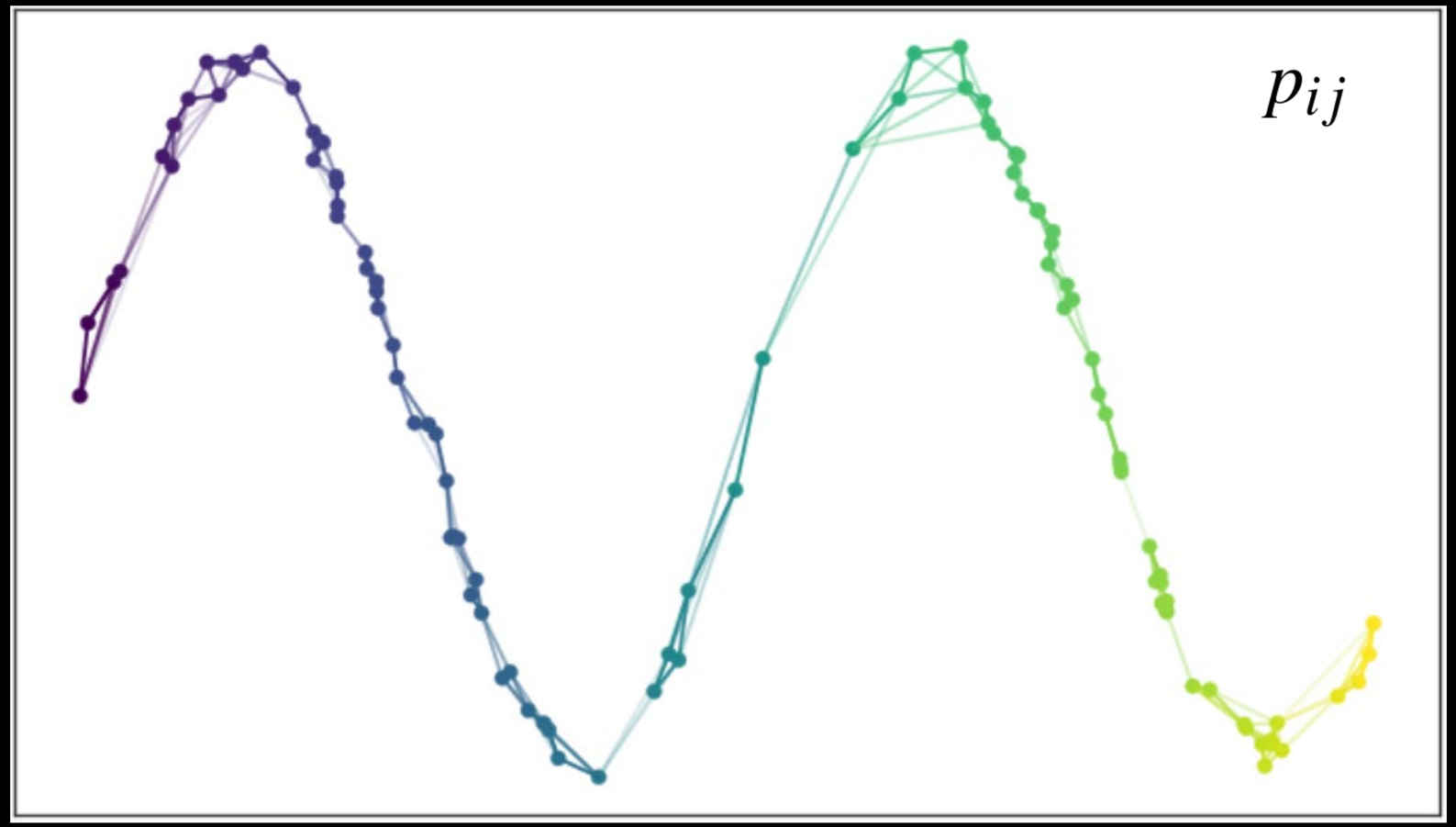
A unit ball about a point in Riemannian space stretches to the k-th nearest neighbor of the point.

Adapting topological analysis to real world data



Fuzzy open balls of radius one with a locally varying metric

Adapting topological analysis to real world data



Graph with combined edge weights

All these stuff give us a good Pij in high dimension.

How Umap works

Riemannian space

$$p_{i|j} = exp(\frac{-max(0, d(x_i, x_{i_j}) - \rho_i)}{\sigma_i})$$

$$p_{ij} = p_{i|j} + p_{j|i} - p_{i|j}p_{j|i}$$

Euclidean space

$$q_{ij} = (1 + a(y_i - y_j)^{2b})^{-1}$$

$$CE(X,Y) = \sum_i \sum_j \left[p_{ij}(X) \log\left(\frac{p_{ij}(X)}{q_{ij}(Y)}\right) + (1 - p_{ij}(X)) \log\left(\frac{1 - p_{ij}(X)}{1 - q_{ij}(Y)}\right) \right]$$

Large p_{ij} modeled by small q_{ij} ? Big penalty!

Small p_{ij} modeled by large q_{ij} ? Big penalty!

Umap could preserve local and global similarity structure.

$X = \{\mathbf{x}_1 \ldots \mathbf{x}_N\}$

$p_{ij} =$ Similarity between *i* and *j in H-Dim*

$P =$ Probability distribution encoding similarities

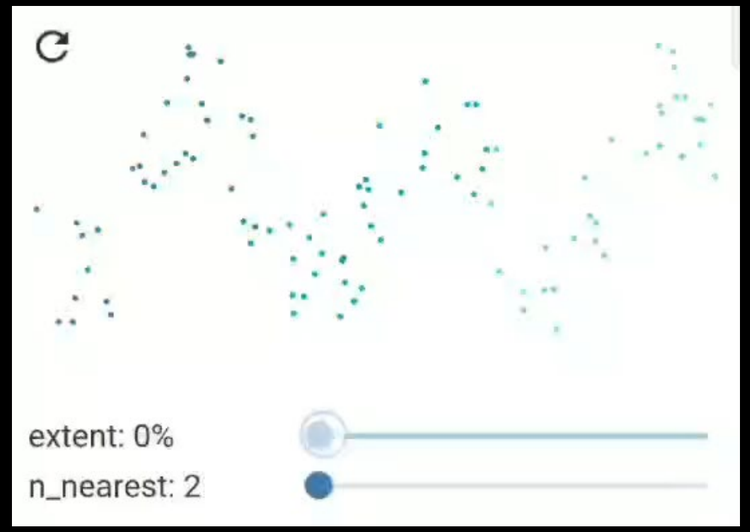
$Y = \{\mathbf{y}_1 \ldots \mathbf{y}_N\}$

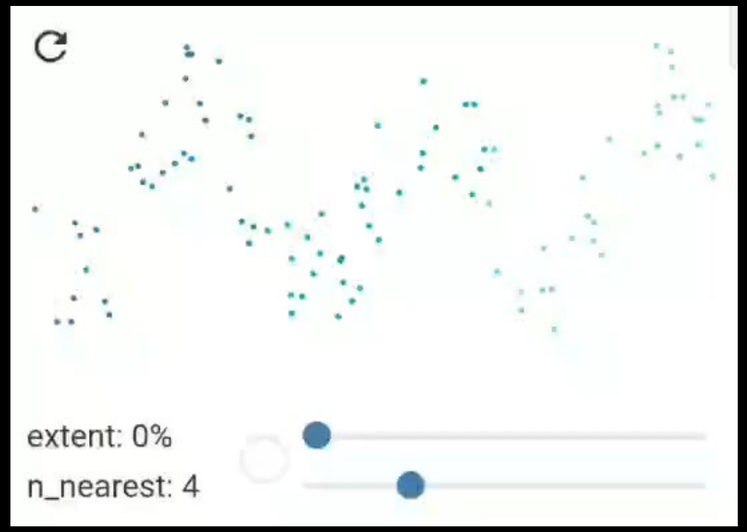
$q_{ij} =$ Similarity between *i* and *j in L-Dim*

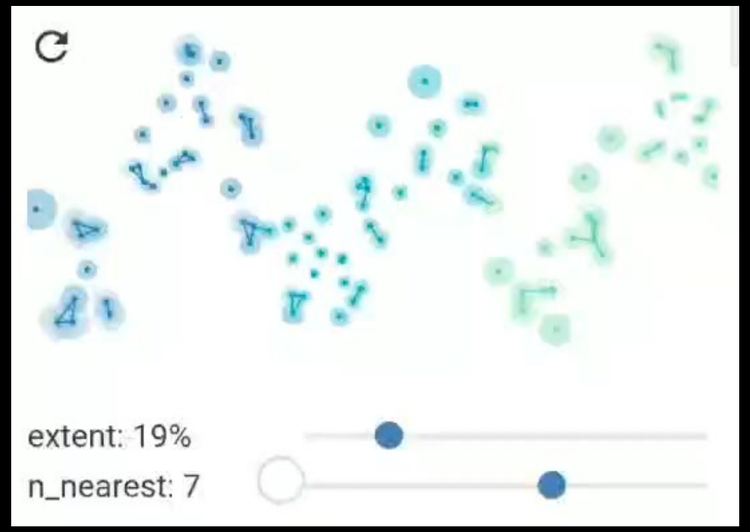
$Q =$ Probability distribution encoding similarities

Cross entropy-based cost function

Hyper-parameters of Umap

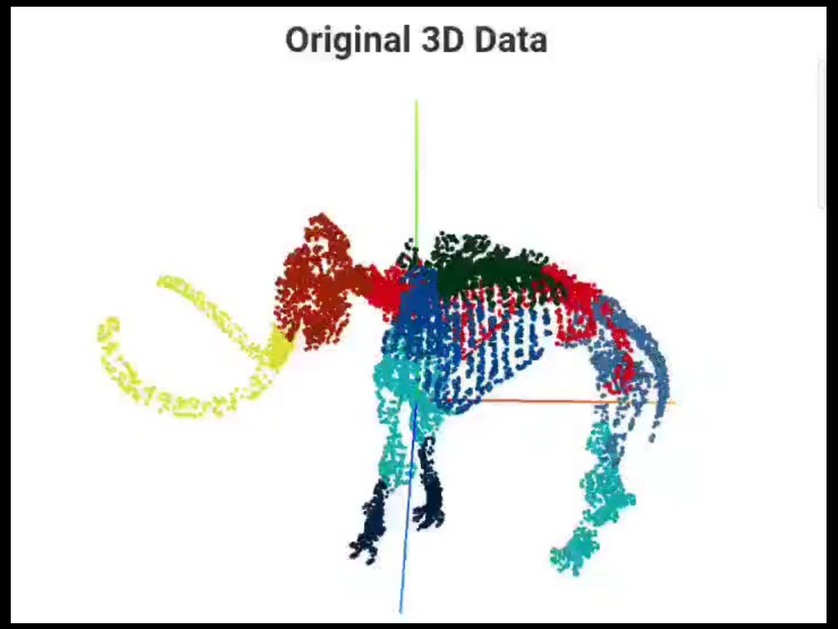


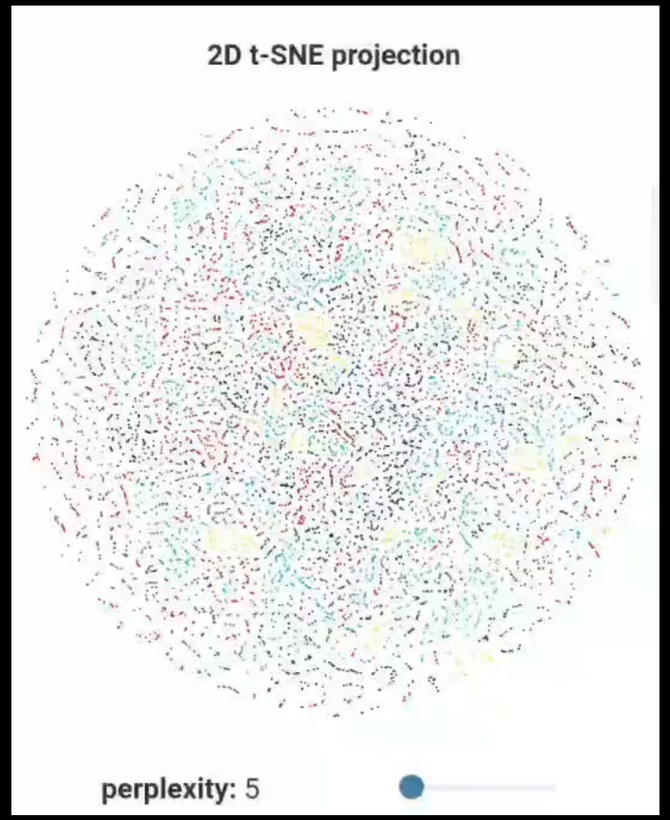


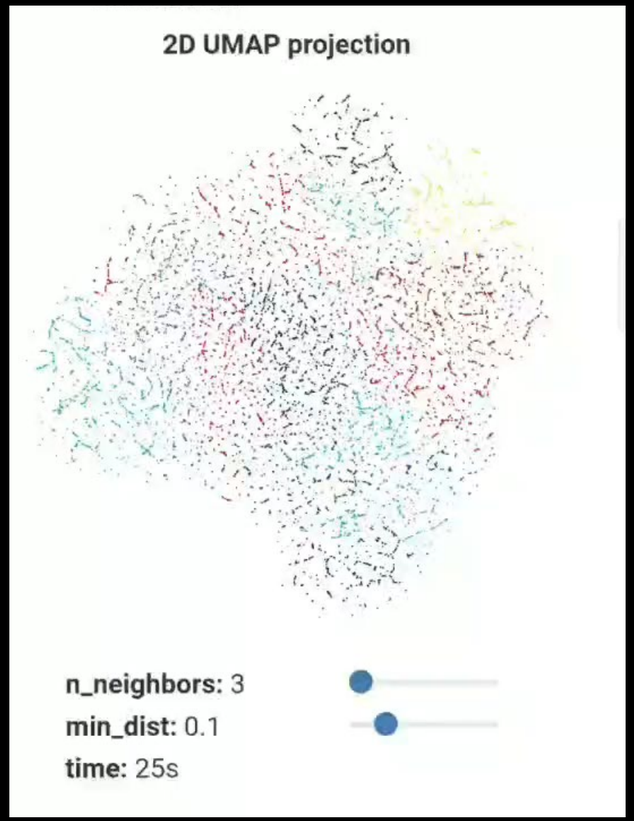


neighbors: number of nearest neighbors.

Hyper-parameters of Umap



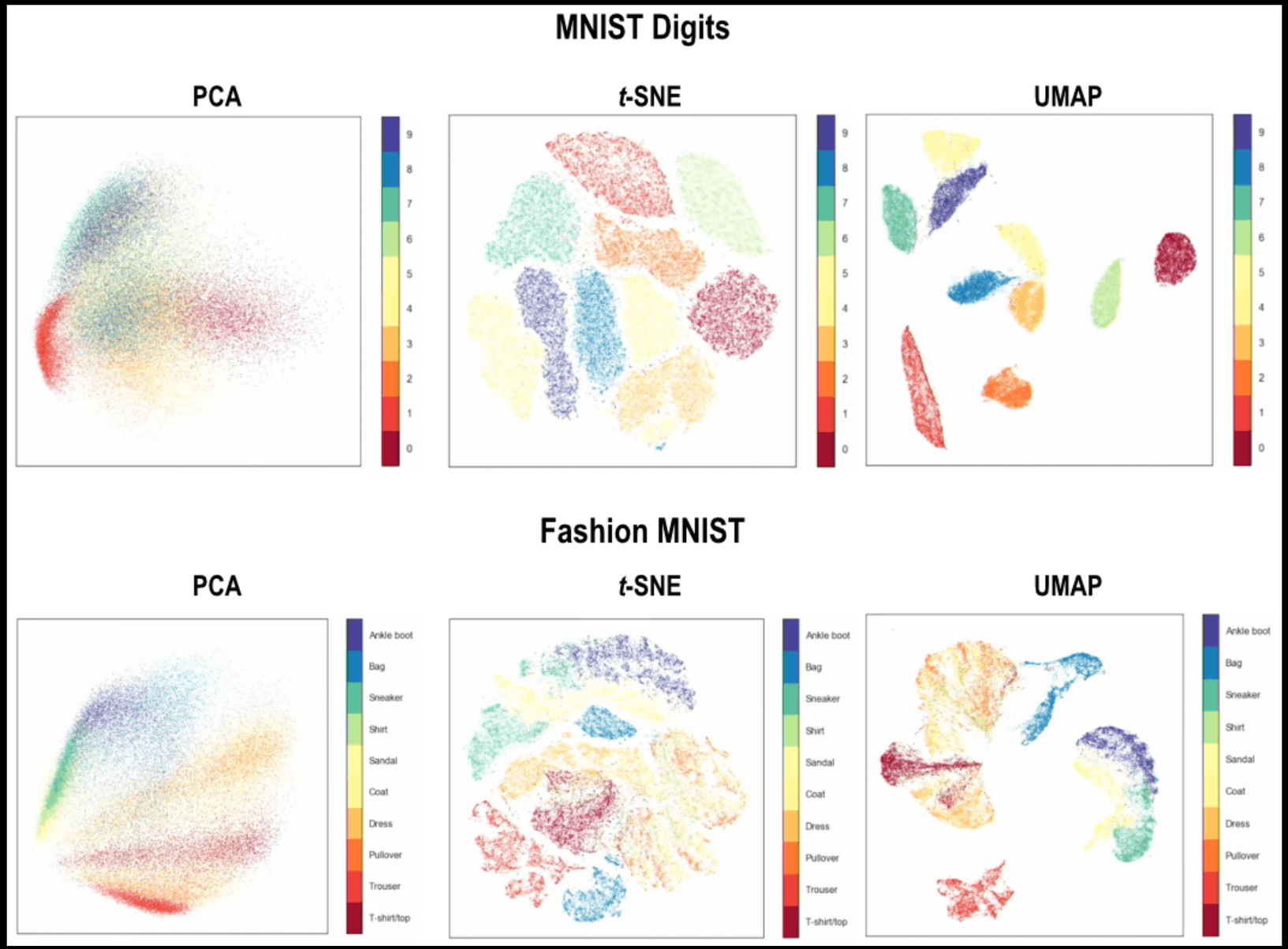




neighbors: number of nearest neighbors.
min_dist: determines how close points appear in the final layout.

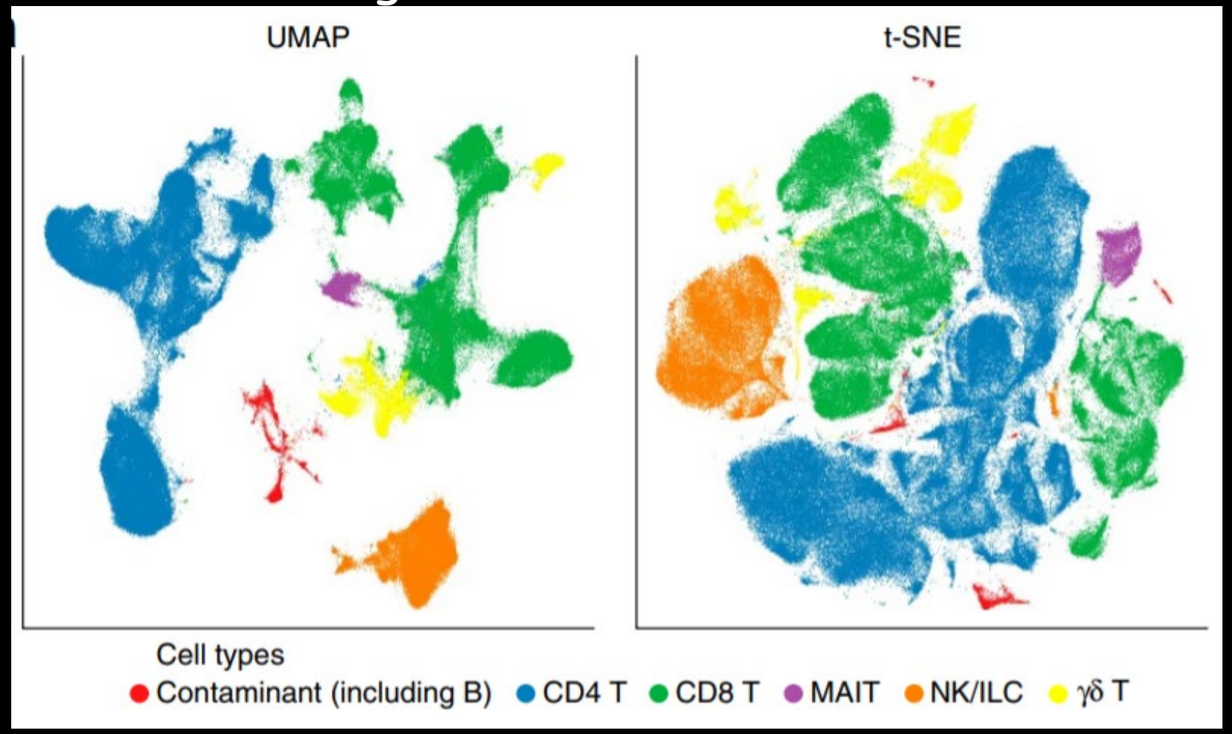
https://pair-code.github.io/understanding-umap/

Umap on MNIST and Fashion MNIST

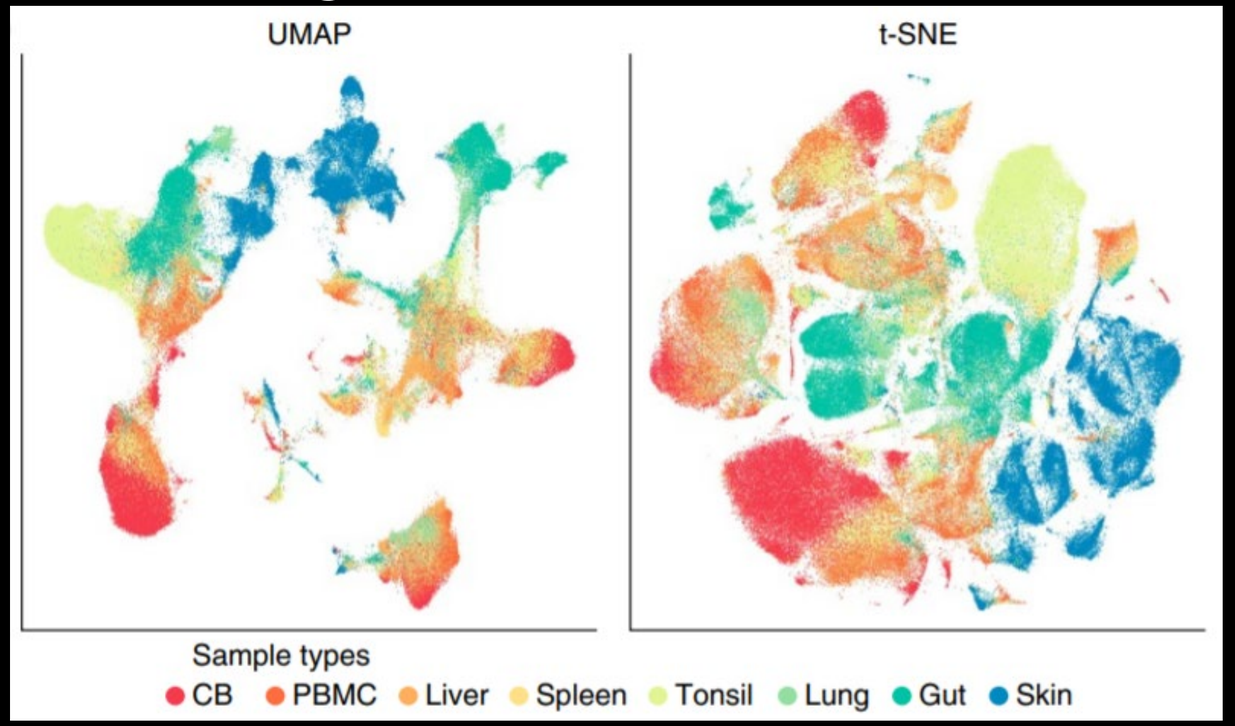


UMAP vs t-SNE

broad cell lineages



tissue of origin



Becht et al. Nature Biotechnology. 2018

Umap is really fast

	UMAP	FIt-SNE	t-SNE	LargeVis	Eigenmaps	Isomap
Pen Digits (1797x64)	9s	48s	17s	20s	**2s**	**2s**
COIL20 (1440x16384)	**12s**	75s	22s	82s	47s	58s
COIL100 (7200x49152)	**85s**	2681s	810s	3197s	3268s	3210s
scRNA (21086x1000)	**28s**	131s	258s	377s	470s	923s
Shuttle (58000x9)	**94s**	108s	714s	615s	133s	–
MNIST (70000x784)	**87s**	292s	1450s	1298s	40709s	–
F-MNIST (70000x784)	**65s**	278s	934s	1173s	6356s	–
Flow (100000x17)	**102s**	164s	1135s	1127s	30654s	–
Google News (200000x300)	**361s**	652s	16906s	5392s	–	–

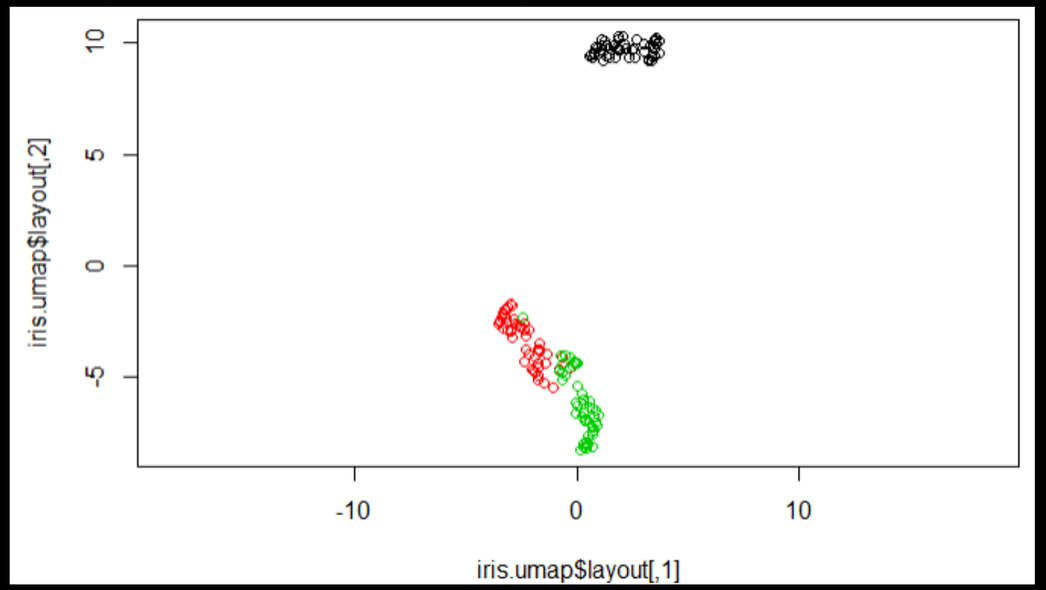
Important notes about Umap

1. Umap implementation - O(n) umap & umap.UMAP() & Seurat
2. It can be run from the top PCs (e.g.: PC1 to PC10)
3. It is no longer completely stochastic as t-SNE
4. Can be applied to new data points
5. Defines both LOCAL and GLOBAL distances

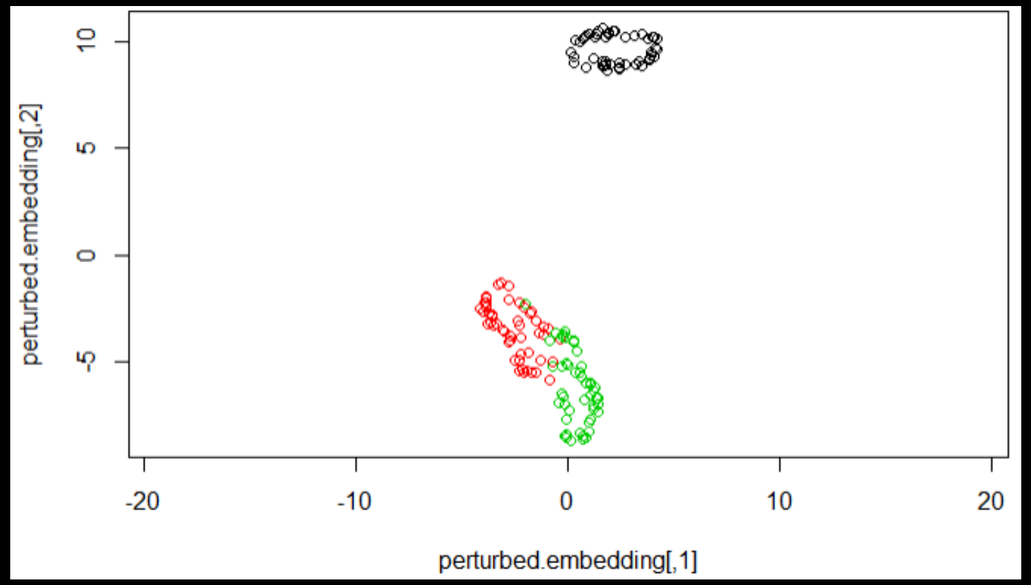
Umap in R

```
library(umap)
# embedd iris dataset
iris.umap = umap(iris[,1:4],n_neighbors=30)
plot(iris.umap$layout,col=iris$Species, asp=1)
# create a dataset with structure like iris, but with perturbation
iris.perturbed = iris[,1:4] + matrix(rnorm(nrow(iris)*4, 0, 0.1), ncol=4)
# project perturbed dataset
perturbed.embedding = predict(iris.umap, iris.perturbed)
plot(perturbed.embedding,col=iris$Species, asp=1)
```

Original data embedding



New data embedding



Recommendation

- StatQuest with Josh Starmer
https://www.youtube.com/c/joshstarmer/playlists

- Coursera class - Machine Learning (Andrew Ng)
https://www.coursera.org/learn/machine-learning

- Dimension reduction
https://www.analyticsvidhya.com/blog/2018/08/dimensionality-reduction-techniques-python/

- Single cell

1. Seurat - Guided Clustering Tutorial
https://satijalab.org/seurat/articles/pbmc3k_tutorial.html
2. Single cell RNA-seq data analysis with R
https://www.csc.fi/web/training/-/scrnaseq

Thanks for your attention