Single-cell analysis and dimension reduction

Yun William Yu

Some slides adapted from Jian Ma

March 30, 2026

02-752 — Advanced Algorithms for Biological Problems
(Spring 2026)

Carnegie Mellon University

Cell types in our body

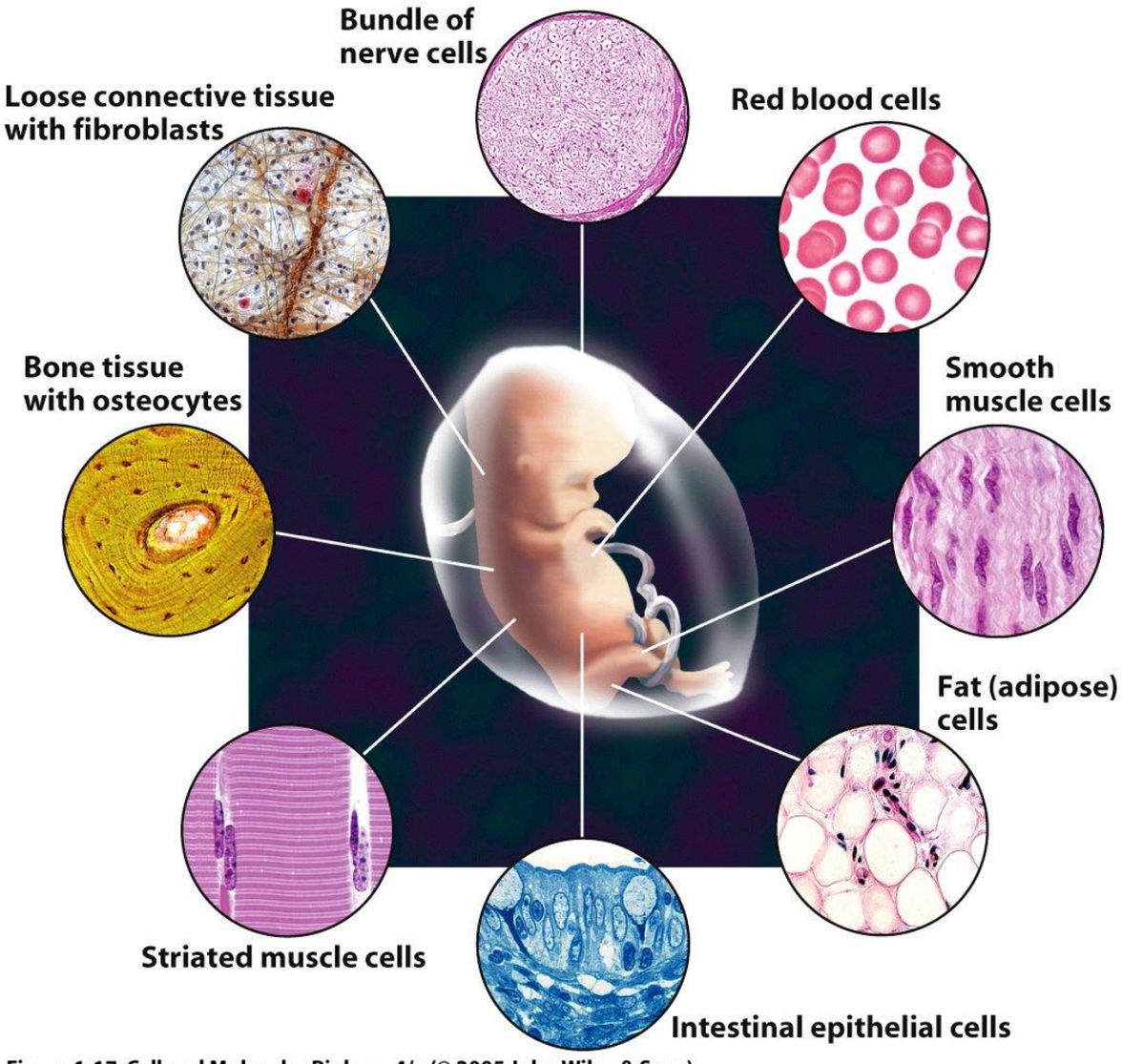
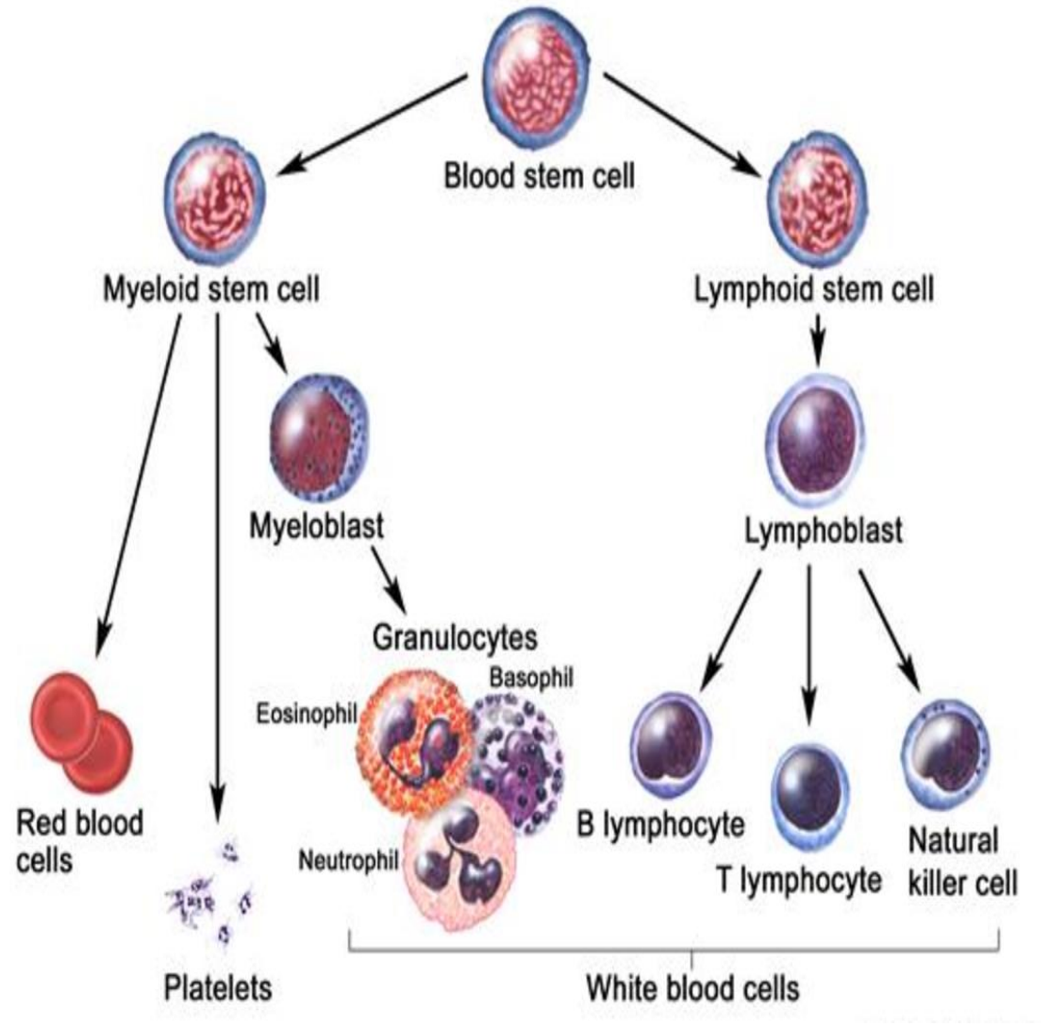
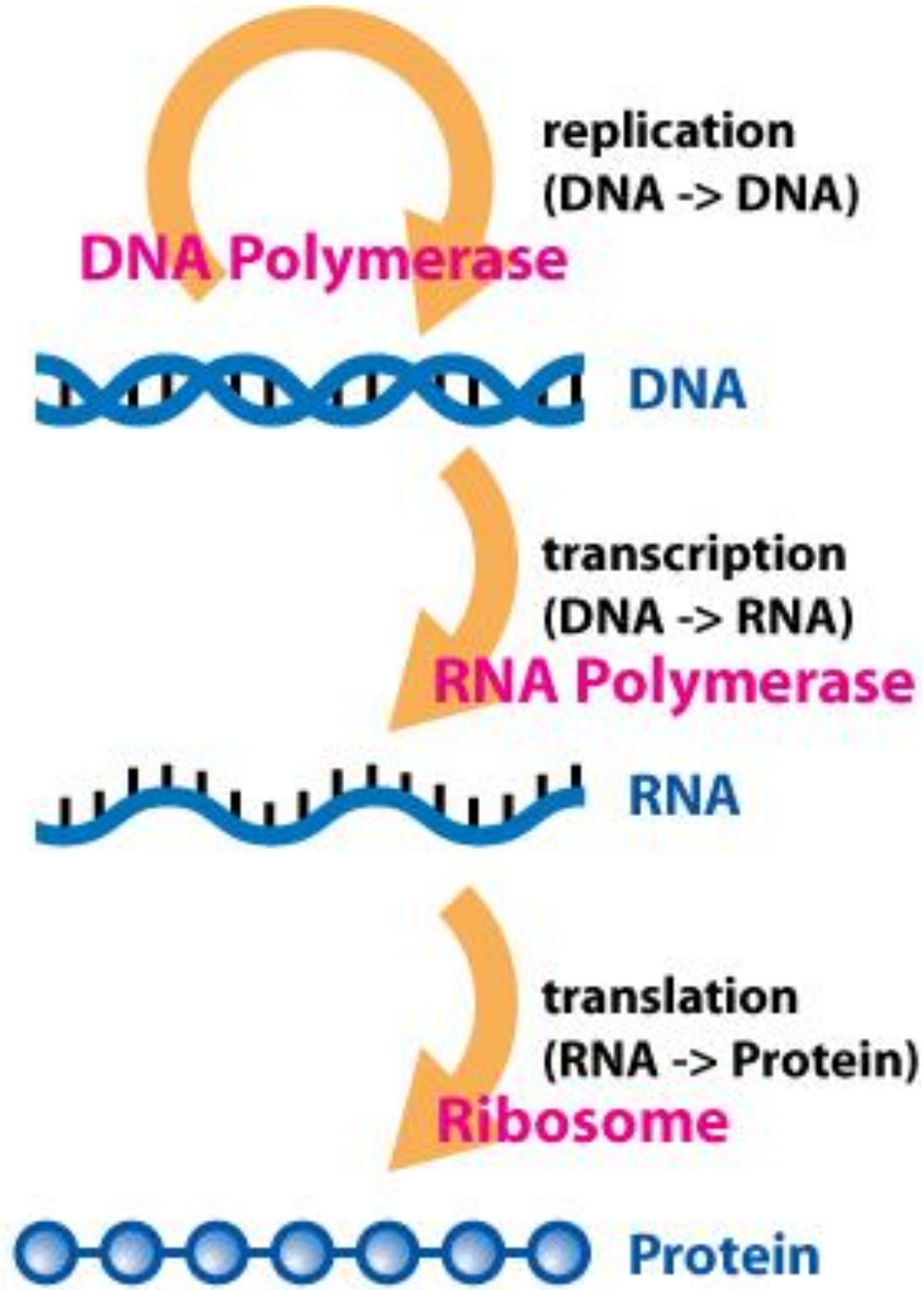


Figure 1-17 Cell and Molecular Biology, 4/e (© 2005 John Wiley & Sons)



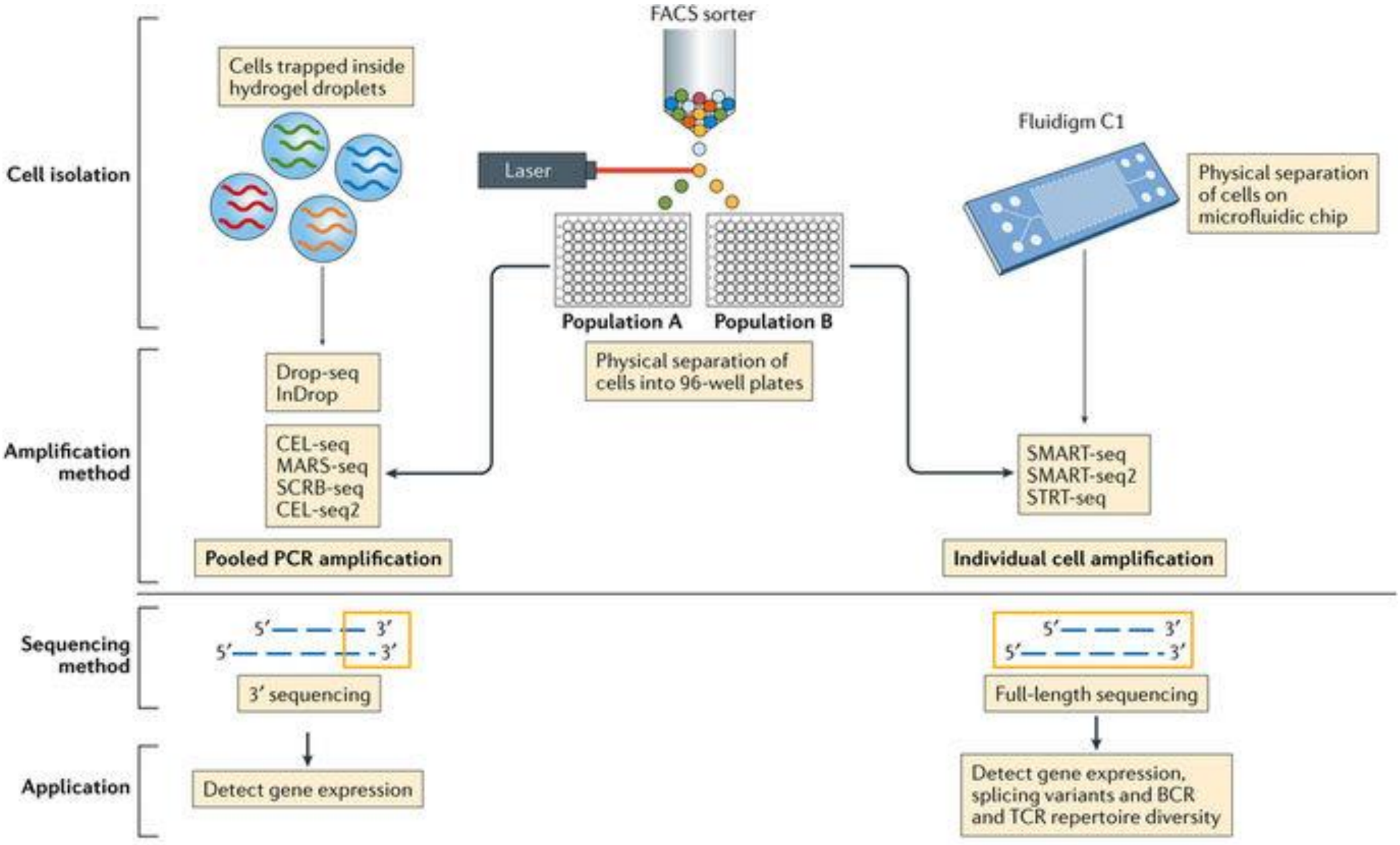
RNA expression distinguishes cell type

- DNA is (to first-order approximation) the same for all the cells in your body.
- Obviously, lots of different types of cells, so the differences have to be later in the central dogma.
- Can analyze using the proteins and other phenotype of cell morphology.
- Or, can do at scale by measuring RNA expression levels.

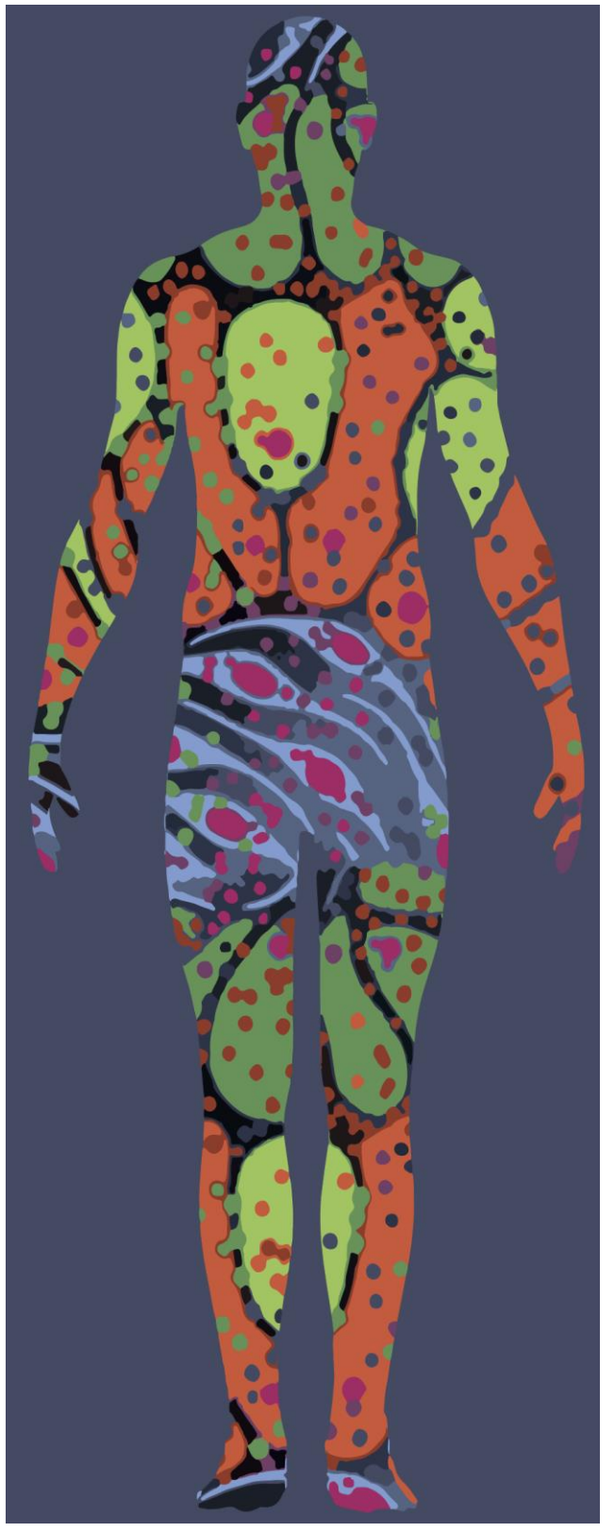


https://en.wikipedia.org/wiki/Central_dogma_of_molecular_biology#/media/File:Central_Dogma_of_Molecular_Biochemistry_with_Enzymes.jpg

Single-cell RNA-seq (scRNA-seq)



Nature Reviews | Immunology



HUMAN CELL ATLAS

CHAN ZUCKERBERG INITIATIVE

About Initiatives Careers

Advancing human potential and promoting equal opportunity.

RNA-seq counts matrix

1	2	3	4	5	6	7	8	9	10	11	12	13
Geneid	MCL1-DL	MCL1-DK	MCL1-DJ	MCL1-DI	MCL1-DH	MCL1-DG	MCL1-LF	MCL1-LE	MCL1-LD	MCL1-LC	MCL1-LB	MCL1-LA
100008567	0	0	3	2	2	0	4	0	0	0	14	10
100009600	20	34	31	23	23	36	1	5	15	15	20	11
100009609	0	0	0	0	0	0	0	0	0	0	0	0
100009614	0	0	0	0	0	0	0	0	0	0	0	0
100009664	1	0	0	0	0	1	1	0	0	3	0	0
100012	0	0	0	0	0	0	0	0	0	0	0	0
100017	555	633	1000	1097	1026	1083	388	344	764	734	712	696
100019	1092	1403	1926	2268	2672	4136	632	567	1562	1984	2265	2849
100033459	0	0	0	0	1	0	1	0	0	0	0	0
100034251	7	11	3	1	0	1	1652	1519	116	24	8	7
100034361	42	43	34	38	30	49	29	24	43	39	59	52
100034363	1	1	2	4	0	0	0	1	0	0	1	0
100034684	0	0	1	0	1	0	0	1	2	3	21	8
100034726	6	1	11	5	6	8	5	4	8	8	12	11
100034729	0	0	0	0	0	0	0	0	0	0	0	0
100034739	5	2	4	5	3	6	2	0	0	5	10	5
100034748	7	6	7	19	10	10	0	3	9	16	20	4
100036518	0	0	0	0	0	0	0	0	0	0	0	0
100036520	0	0	0	0	0	0	0	0	0	0	0	1
100036521	228	234	244	247	229	220	243	244	213	232	358	334
100036523	4	3	9	5	6	10	5	5	3	9	7	8
100036537	4	2	1	2	0	0	0	0	1	0	0	0
100036568	3	2	3	4	2	5	5	3	0	0	0	2
100036571	0	0	0	0	0	1	0	0	0	0	0	0
100036572	2	0	0	0	1	0	0	1	0	0	0	0

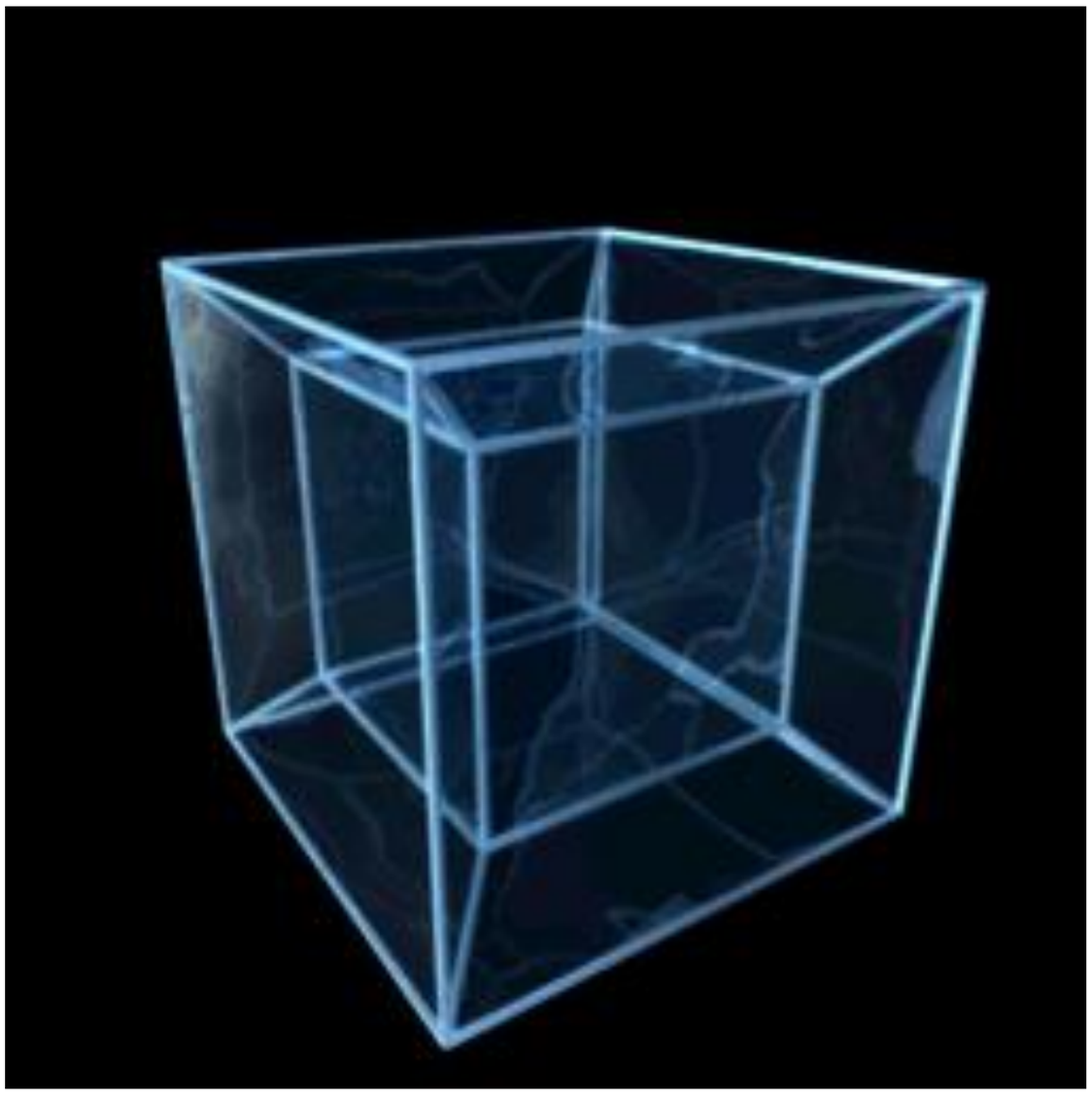
https://training.galaxyproject.org/training-material/topics/transcriptomics/images/rna-seq-reads-to-counts/count_matrix.png

Visualizing high dimensions is hard

Point (0)

.

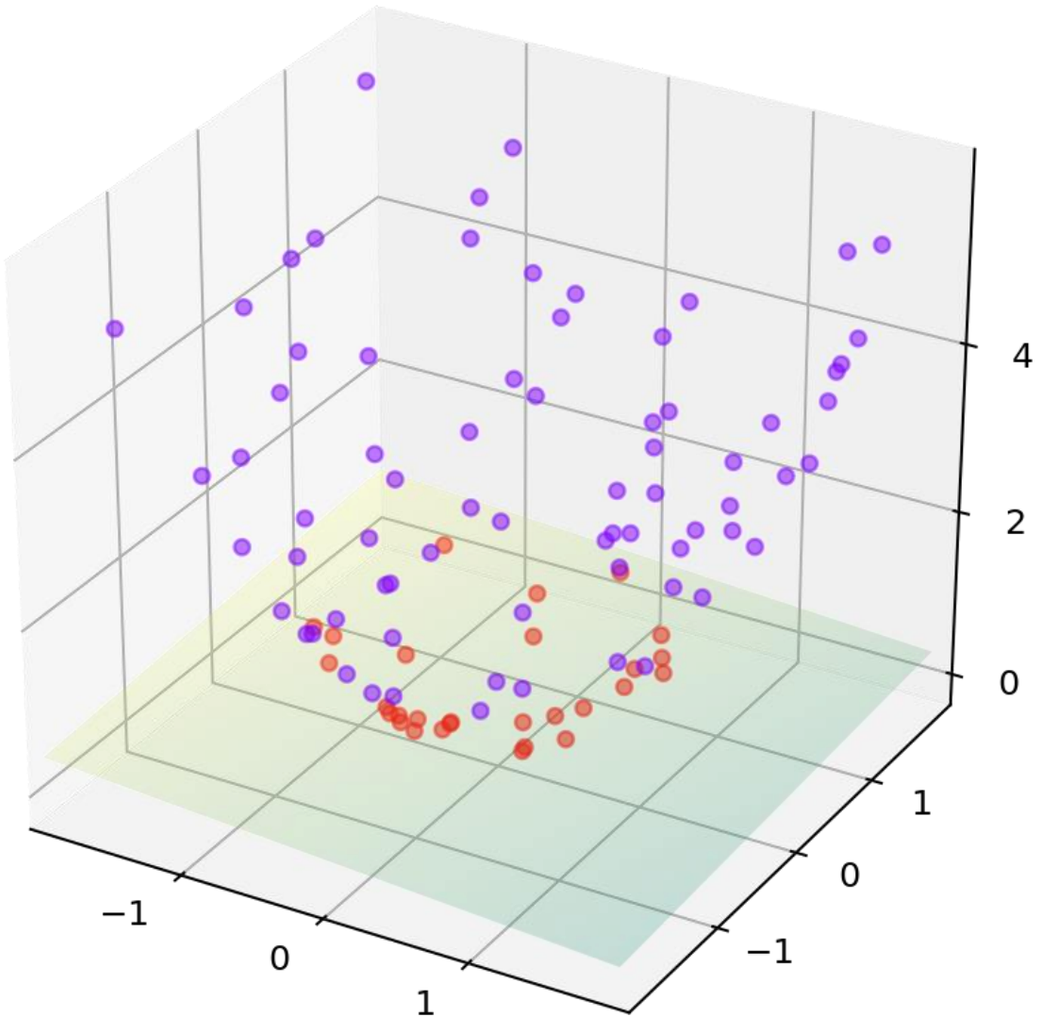
Tesseracts are 4-cubes



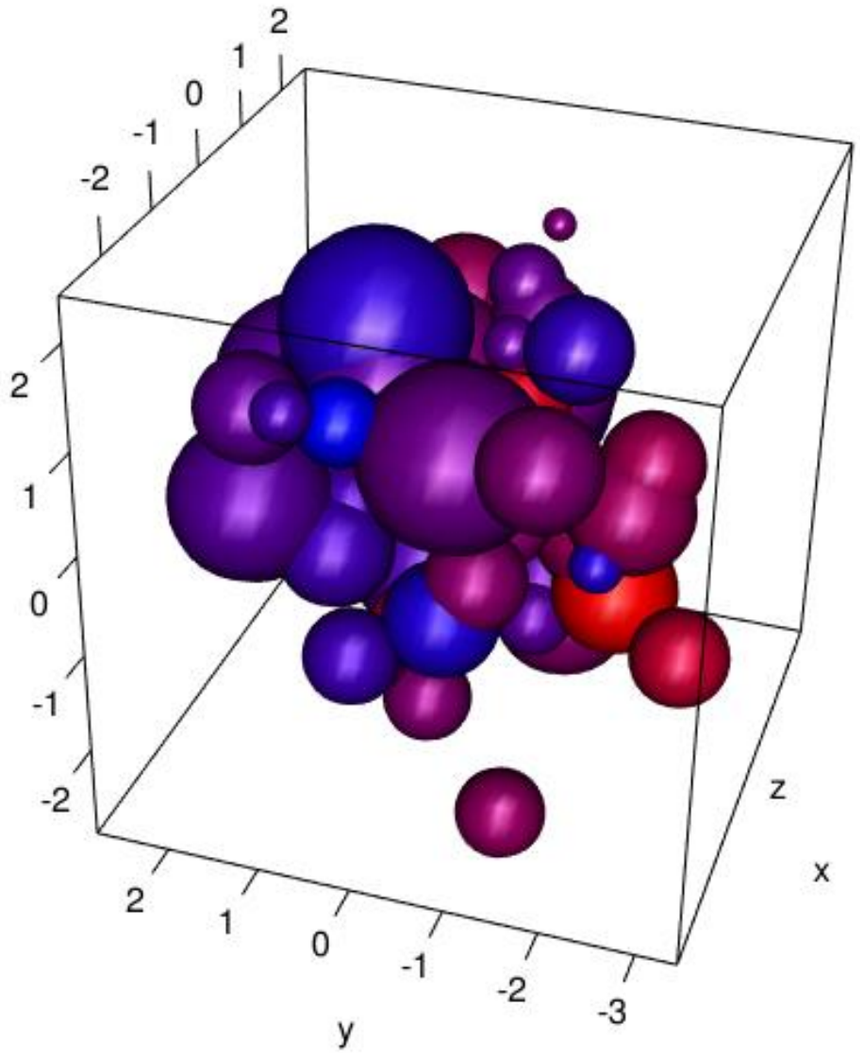
Rotating tesseracct

https://en.wikipedia.org/wiki/Hypercube

Sometimes can use other properties (size, color) for higher-dimensional scatter plots



https://en.wikipedia.org/wiki/Kernel_method#/media/File:Kernel_trick_idea.svg



https://stackoverflow.com/questions/53896144/rgl-3d-scatterplot-controlling-size-of-spheres-from-4th-dimension-bubble-plot

Dimensionality reduction: Some Assumptions

- High-dimensional data often lies on or near a much lower dimensional, curved manifold.

- A good way to represent data points is by their low-dimensional coordinates.

- The low-dimensional representation of the data should capture information about high-dimensional pairwise distances.

The basic idea of non-parameteric dimensionality reduction

- Represent each data-point by a point in a lower dimensional space.

- Choose the low-dimensional points so that they optimally represent some property of the data-points (e.g., the pairwise distances).
  - Many different properties have been tried.

Two types of dimensionality reduction

- **Global methods** assume that all pairwise distances are of equal importance.
  - Choose the low-D pairwise distances to fit the high-D ones (using magnitude or rank order).

- **Local methods** assume that only the local distances are reliable in high-D.
  - Put more weight on modeling the local distances correctly.

Linear methods of reducing dimensionality

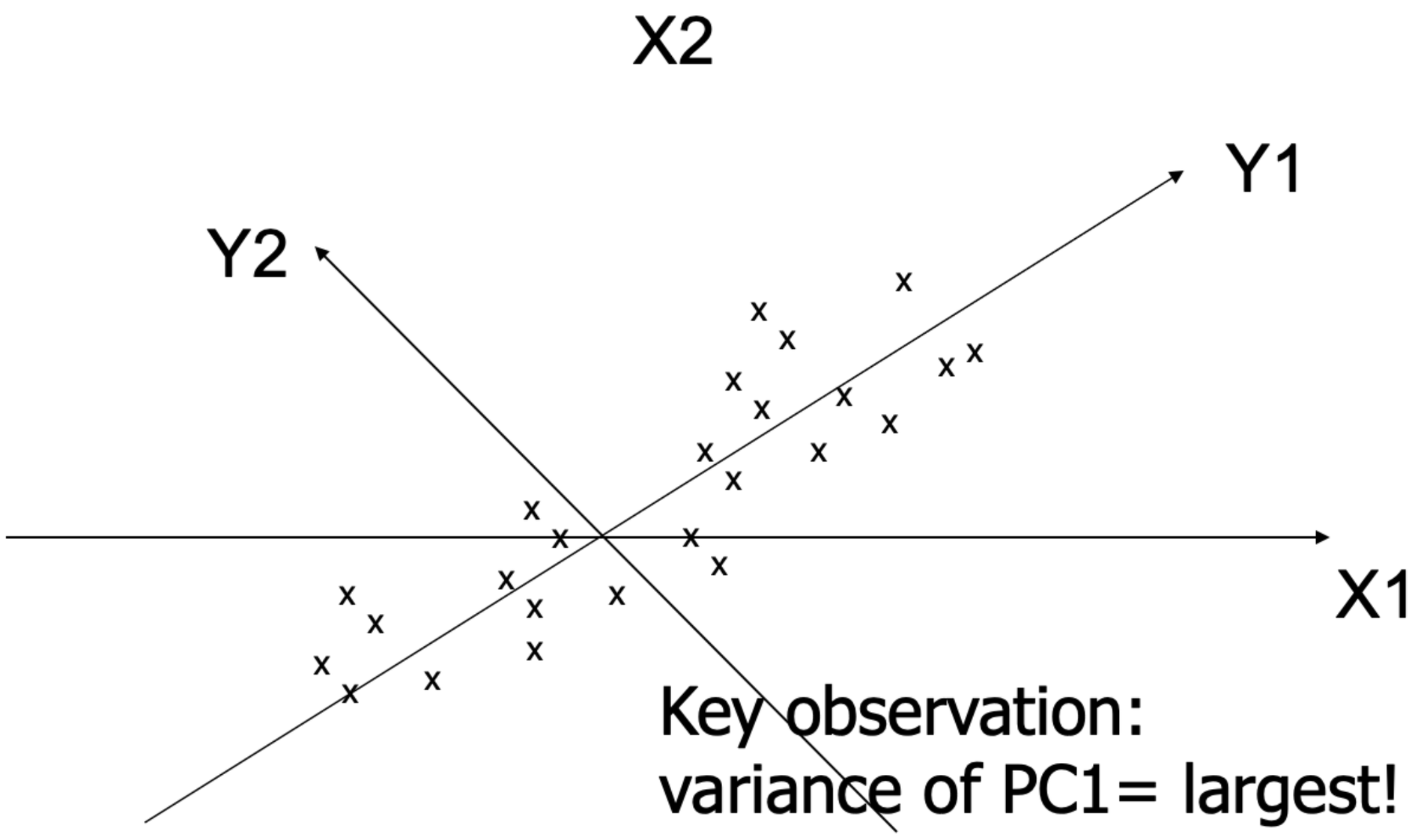
- **PCA** finds the directions that have the most variance.
  - By representing where each datapoint is along these axes, we minimize the squared reconstruction error.

- **Principal Components Analysis (PCA) —** Unsupervised technique used to reduce the dimensionality of the data

- Can be used to:
  - Reduce number of dimensions in data
  - Find patterns in high-dimensional data
  - Visualize data of high dimensionality

- Example applications:
  - Face recognition
  - Gene expression analysis
  - Single cell analysis

Y1 is the first eigen vector, Y2 is the second.

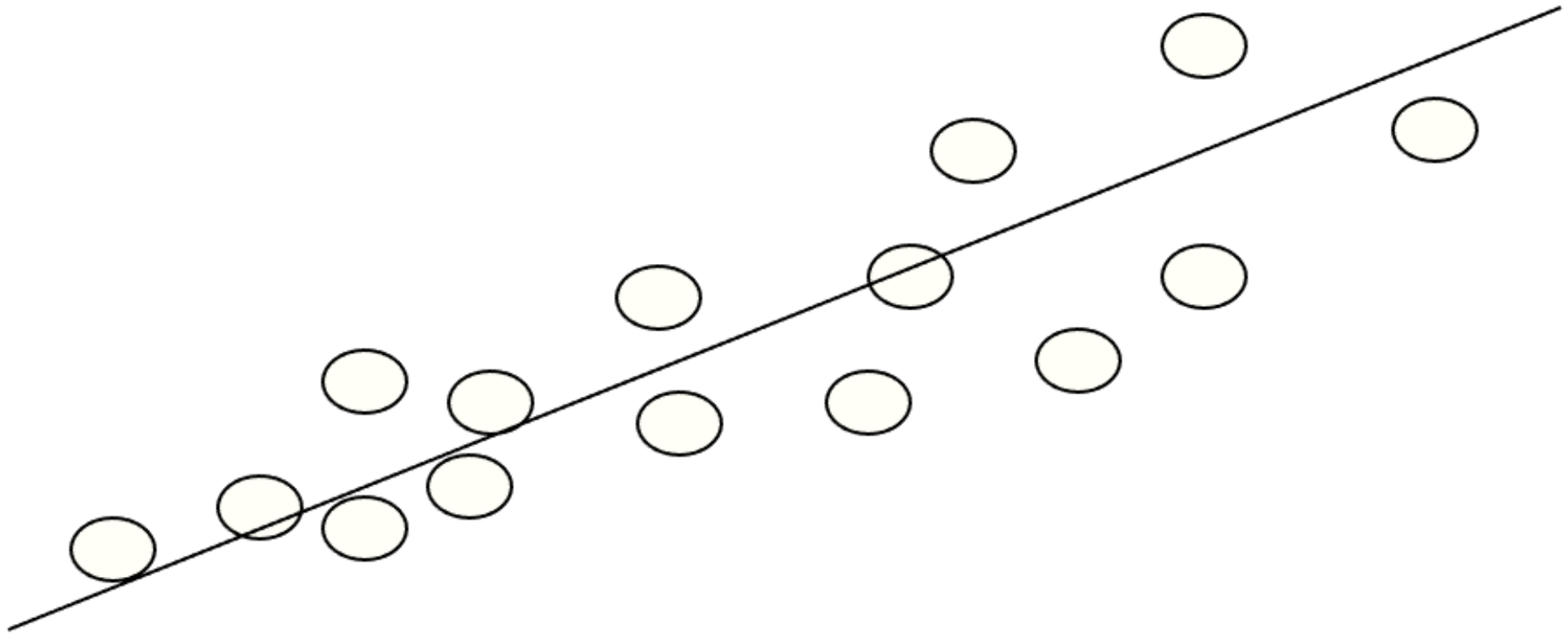


Key observation:
variance of PC1= largest!

Algebraic Interpretation – 1D

- Given m points in a n dimensional space, for large n, how can we project them on a 1 dimensional line?

- Choose a line that fits the data so the points are spread out well along the line

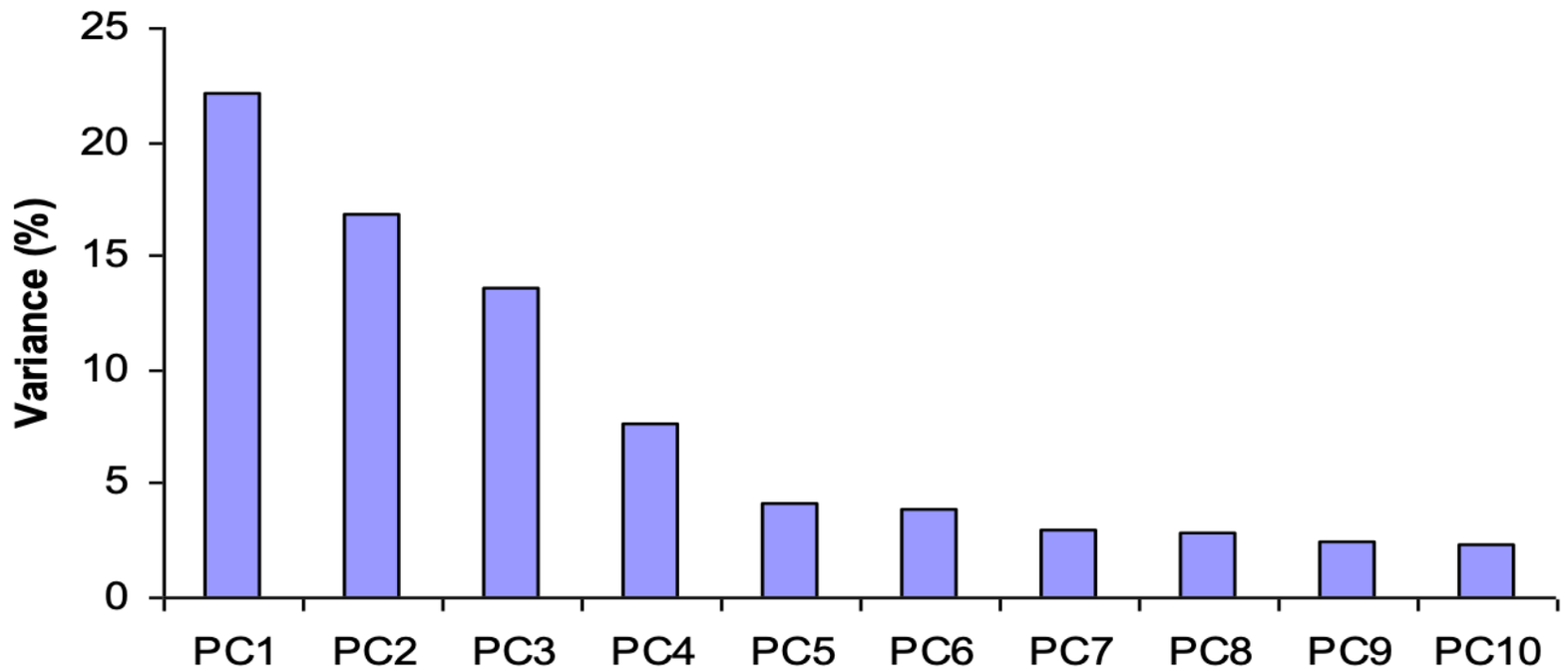


Principal components

- Principal component 1 (PC1): The eigenvalue with the largest absolute value will indicate that the data have the largest variance along its eigenvector, the direction along which there is greatest variation

- Principal component 2 (PC2): the direction with maximum variation left in data, orthogonal to the 1.

- In general, only few directions manage to capture most of the variability in the data.



Steps of PCA

- Let X be the $m \times n$ data matrix, where each row corresponds to a data point and each column a variable.
- Let $\bar{X} = \frac{1}{m} X\mathbf{1}$ be the row mean.
- Adjust the original data by the mean: $X' = X - \mathbf{1}\bar{X}$.
- Compute the covariance matrix C of X' by $C = \frac{1}{m-1} X^T X$
- Find the eigenvectors and eigenvalues of C.
  - i.e. vectors e such that $Ce = \lambda e$, where λ is an eigenvalue.

Eigenvalues

- Calculate eigenvalues λ and eigenvectors **x** for covariance matrix:
  - Eigenvalues λ_j are used for calculation of [% of total variance] (V_j) for each component *j*:

$$V_j = 100 \cdot \frac{\lambda_j}{\sum_{x=1}^{n} \lambda_x}$$

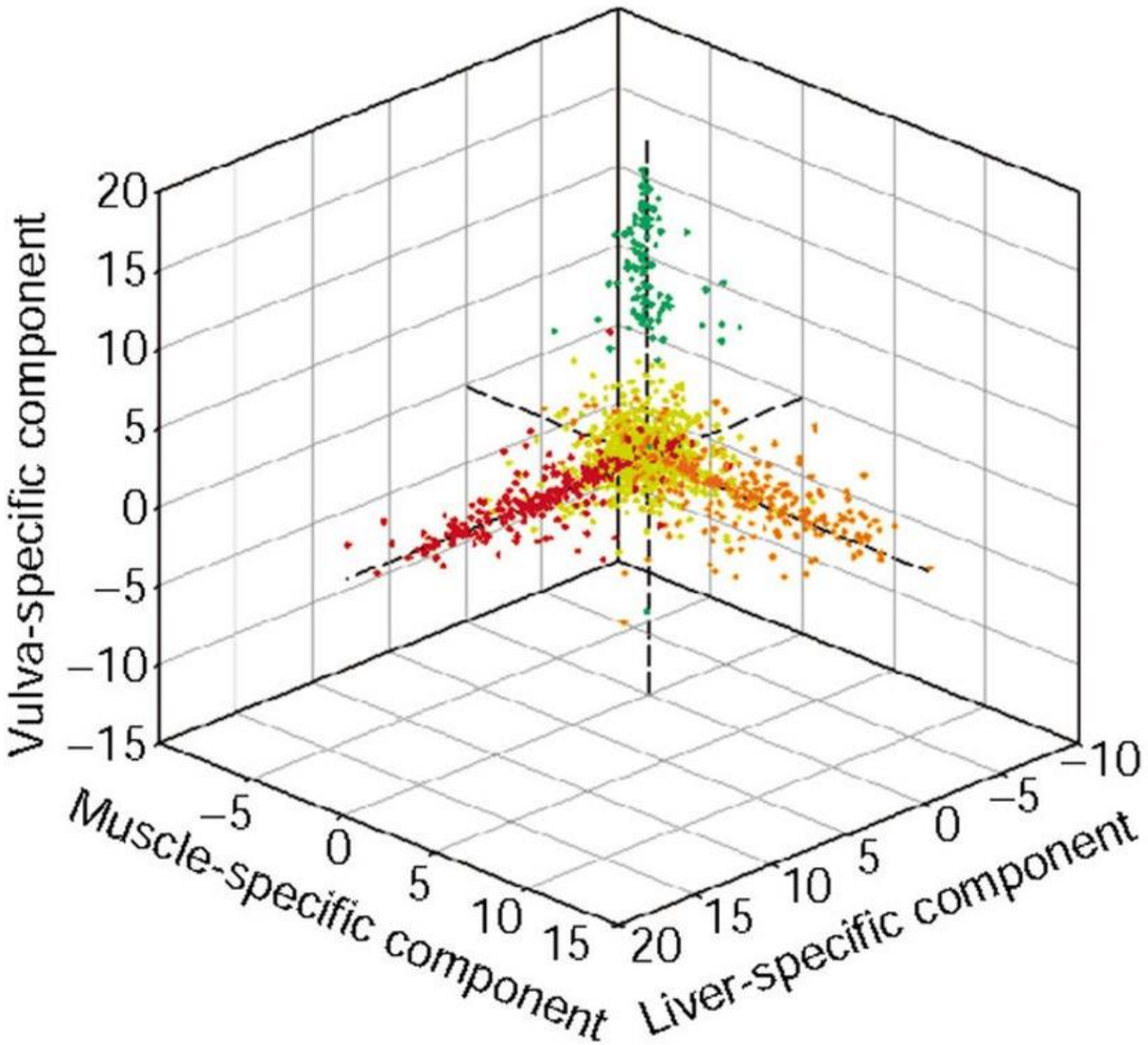
- Plot each cell using the values for the top Eigenvectors (usually 2, though can go higher).

Exercise – SVD for PCA

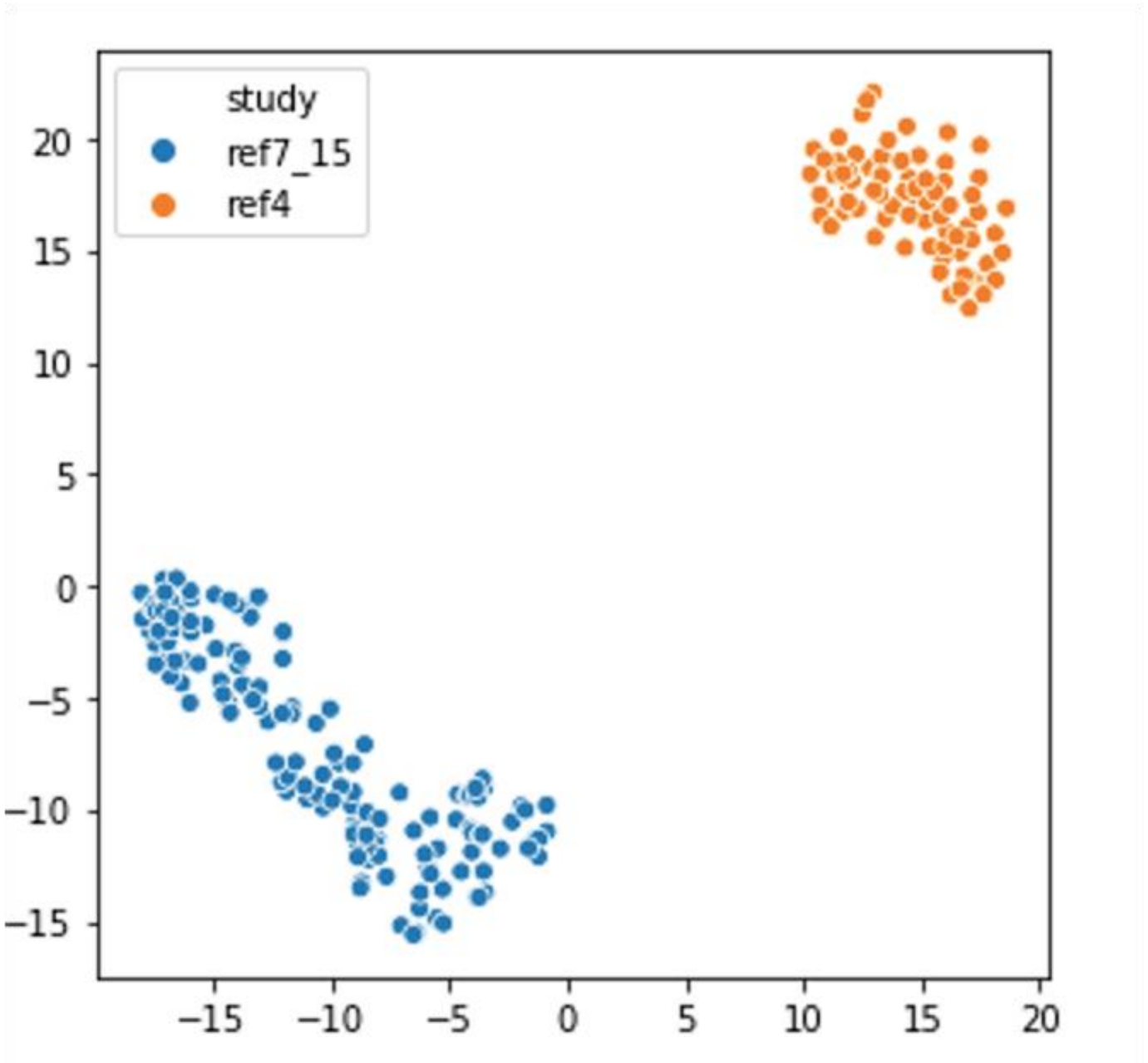
- Recall that the singular value decomposition of an $m \times n$ matrix M is a factorization $M = U\Sigma V^T$, where U and V are real matrices with orthonormal columns of size $m \times r$ and $n \times r$ respectively, and Σ is a nonnegative real $r \times r$ diagonal matrix.
- Instead of using an eigendecomposition, show how you can use singular value decomposition instead.
- Why might one want to use an SVD-solver instead of an eigensolver for PCA?

Applications – Gene expression analysis

- Group genes based on their expression
- PCA is a commonly used either instead of or prior to clustering



PCA for cells from two lung development studies

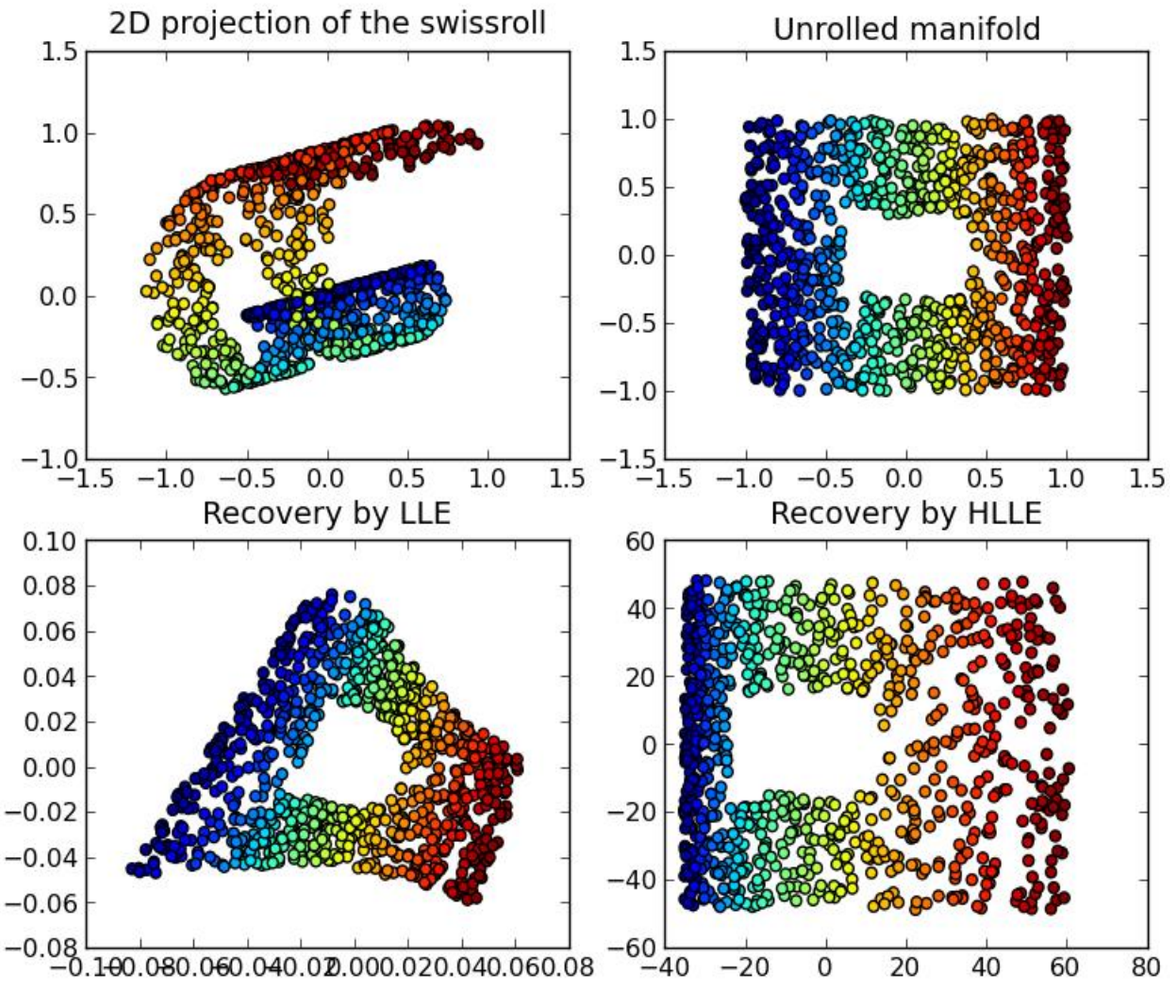


Nonlinear dimensionality reduction

- SVD and related methods like PCA allows you to reduce the dimensionality of a dataset down linearly.

- What if your dataset is actually nonlinear?

- What techniques do we have to reduce dimensionality while preserving structure?
  - E.g. t-SNE and UMAP, among others.



https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction#/media/File:Lle_hlle_swissroll.png

Linear methods of reducing dimensionality

- Multi-Dimensional Scaling arranges the low-dimensional points so as to minimize the discrepancy between the pairwise distances in the original space and the pairwise distances in the low-D space.
- Metric Multi-Dimensional Scaling

- Find low dimensional representatives, y, for the high-dimensional data-points, x, that preserve pairwise distances as well as possible.
- An obvious approach is to start with random vectors for the y's and then perform steepest descent by following the gradient of the cost function.

$$Cost = \sum_{i<j} (d_{ij} - \hat{d}_{ij})^2$$

$$d_{ij} = \| x_i - x_j \|^2$$

$$\hat{d}_{ij} = \| y_i - y_j \|^2$$

Other non-linear methods of reducing dimensionality: Sammon

- Multi-Dimensional Scaling can be made non-linear by putting more weight on the smaller distances. A popular version is the Sammon mapping:

high-D distance

low-D distance

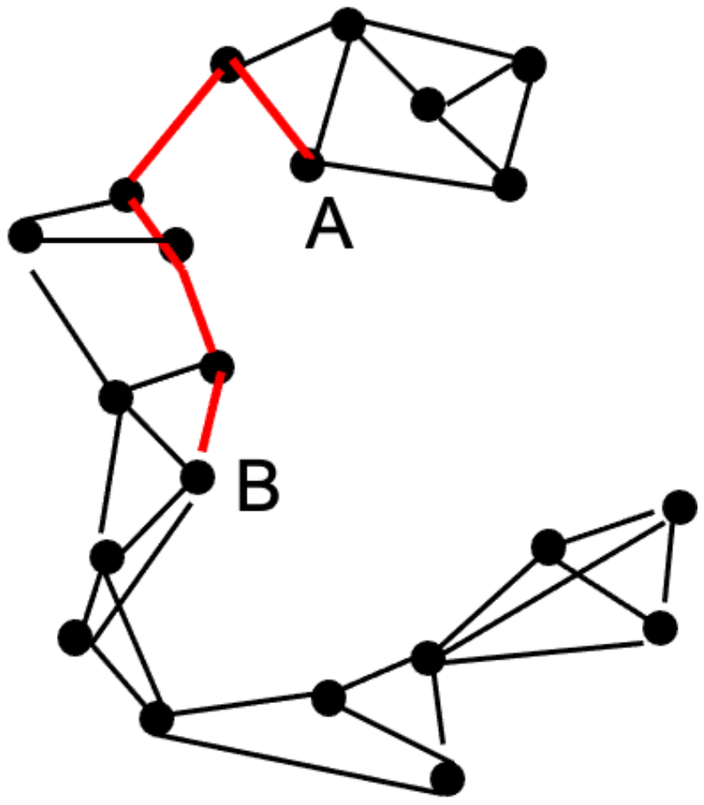
$$Cost = \sum_{i\,j} \left(\frac{\| \mathbf{x}_i - \mathbf{x}_j \| - \| \mathbf{y}_i - \mathbf{y}_j \|}{\| \mathbf{x}_i - \mathbf{x}_j \|} \right)^2$$

- Non-linear MDS is also slow to optimize and also gets stuck in different local optima each time.

- It puts too much emphasis on getting very small distances exactly right.
- It produces embeddings that are circular with roughly uniform density of the map points.

Addressing uniform circularity: Graph based methods

- **Isomap** is a dimensionality reduction technique based on graphs
- Connect each datapoint to its K nearest neighbors in the high-dimensional space.
- Put the true Euclidean distance on each of these links.
- Then approximate the manifold distance between any pair of points as the shortest path in this “neighborhood graph”.



Addressing impact of short distances: Probabilistic local MDS

- It is more important to get local distances right than non-local ones, but getting infinitesimal distances right is not infinitely important.
  - All the small distances are about equally important to model correctly.
  - Stochastic neighbor embedding has a probabilistic way of deciding if a pairwise distance is “local”.

Local embedding

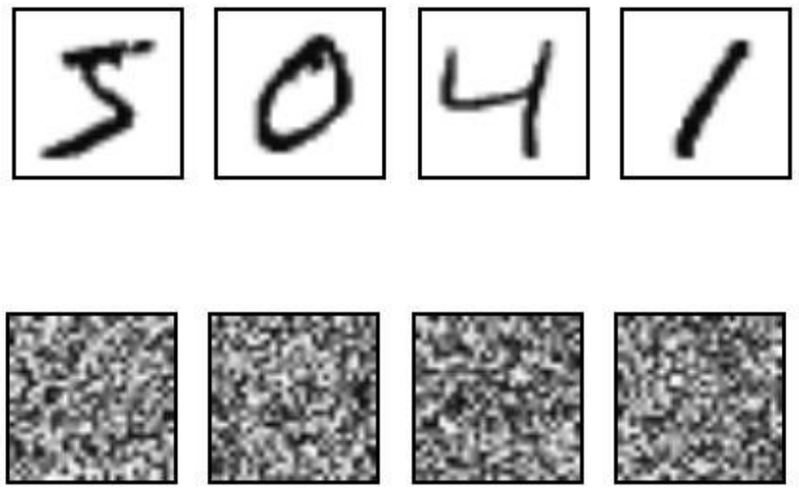
- t-SNE is an alternative dimensionality reduction algorithm.

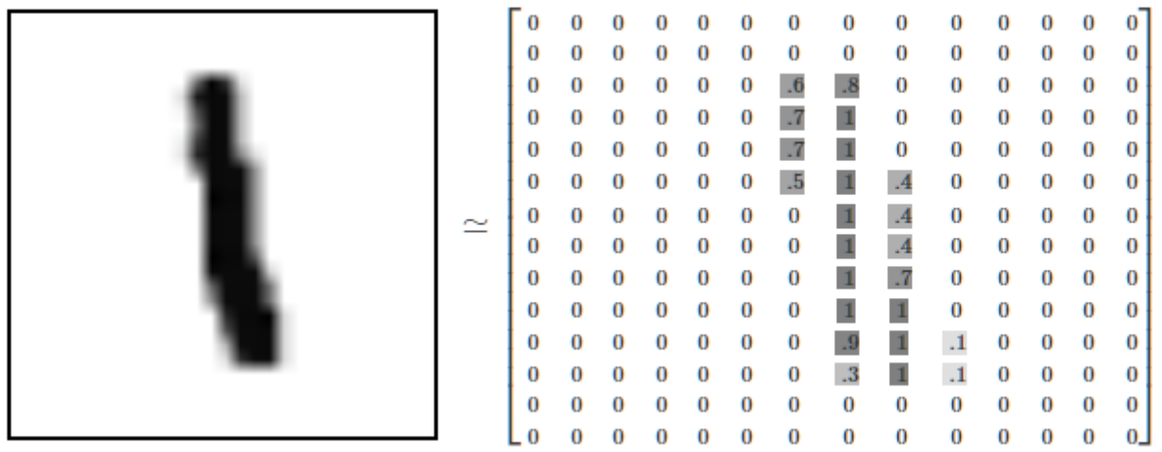
- PCA tries to find a **global** structure
  - Low dimensional subspace
  - Can lead to local inconsistencies — Far away point can become nearest neighbors

- t-SNE tries to preserve **local** structure
  - Low dimensional neighborhood should be the same as original neighborhood.

- Unlike PCA almost only used for visualization
  - No easy way to embed new points

MNIST

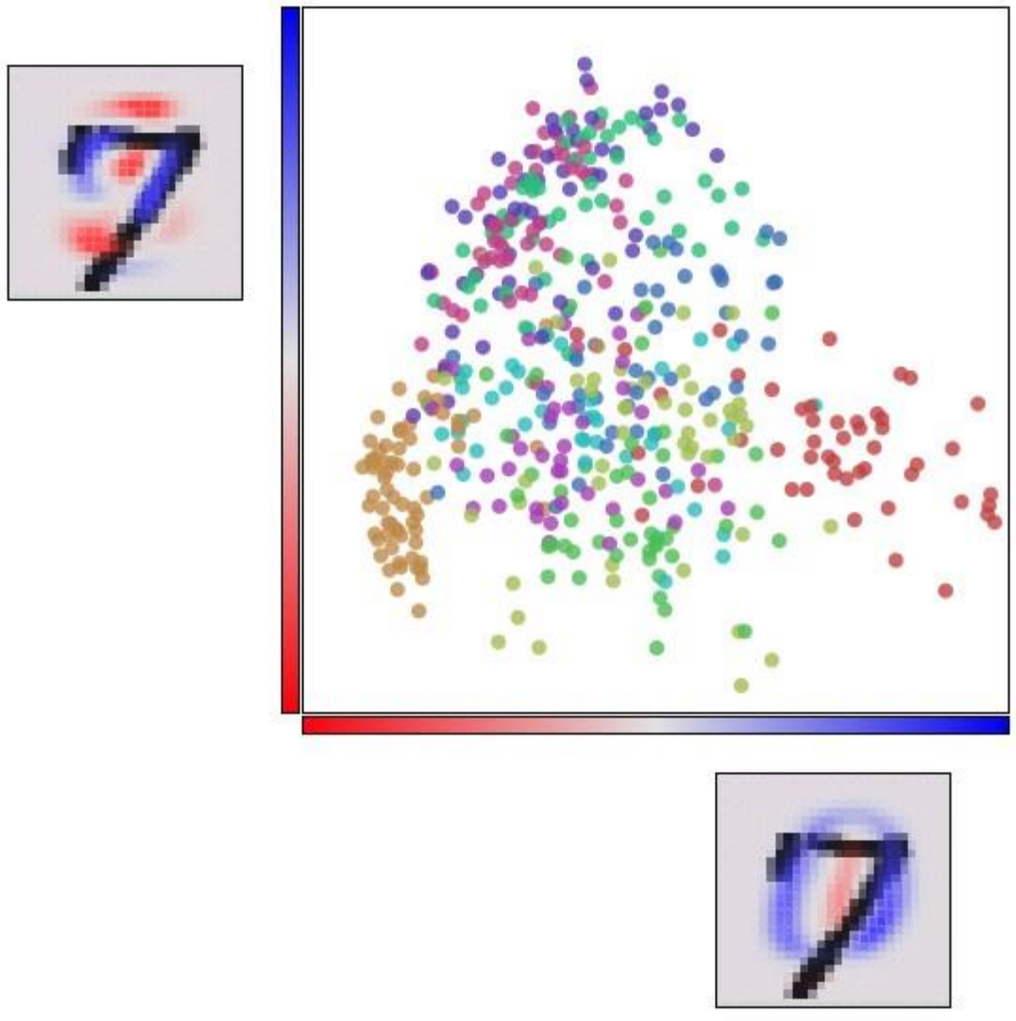




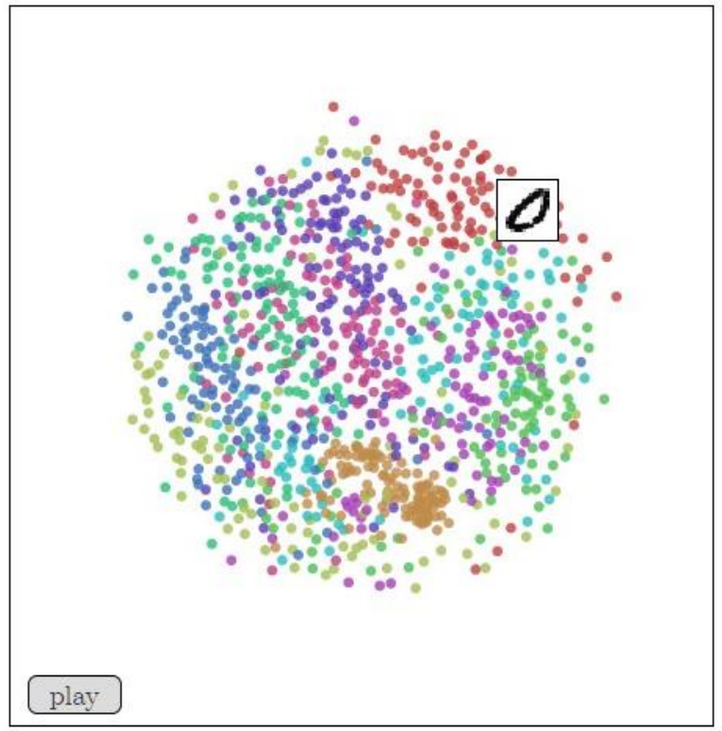
https://www.cnblogs.com/jins-note/p/9719157.html

L. Bottou et al., "Comparison of classifier methods: a case study in handwritten digit recognition," Proceedings of the 12th IAPR International Conference on Pattern Recognition, Vol. 3 - Conference C: Signal Processing (Cat. No.94CH3440-5), Jerusalem, Israel, 1994, pp. 77-82 vol.2, doi: 10.1109/ICPR.1994.576879.

MNIST PCA and MDS

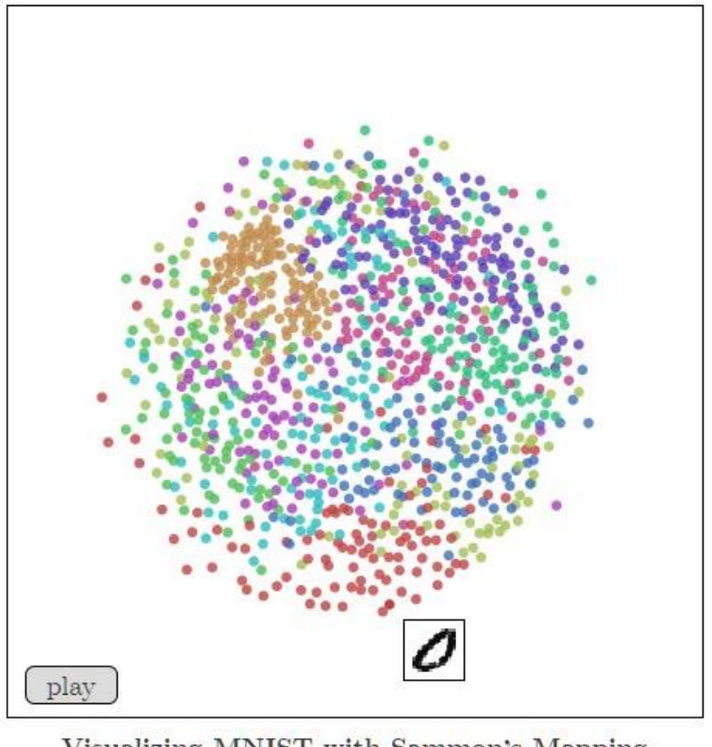


Visualizing MNIST with PCA



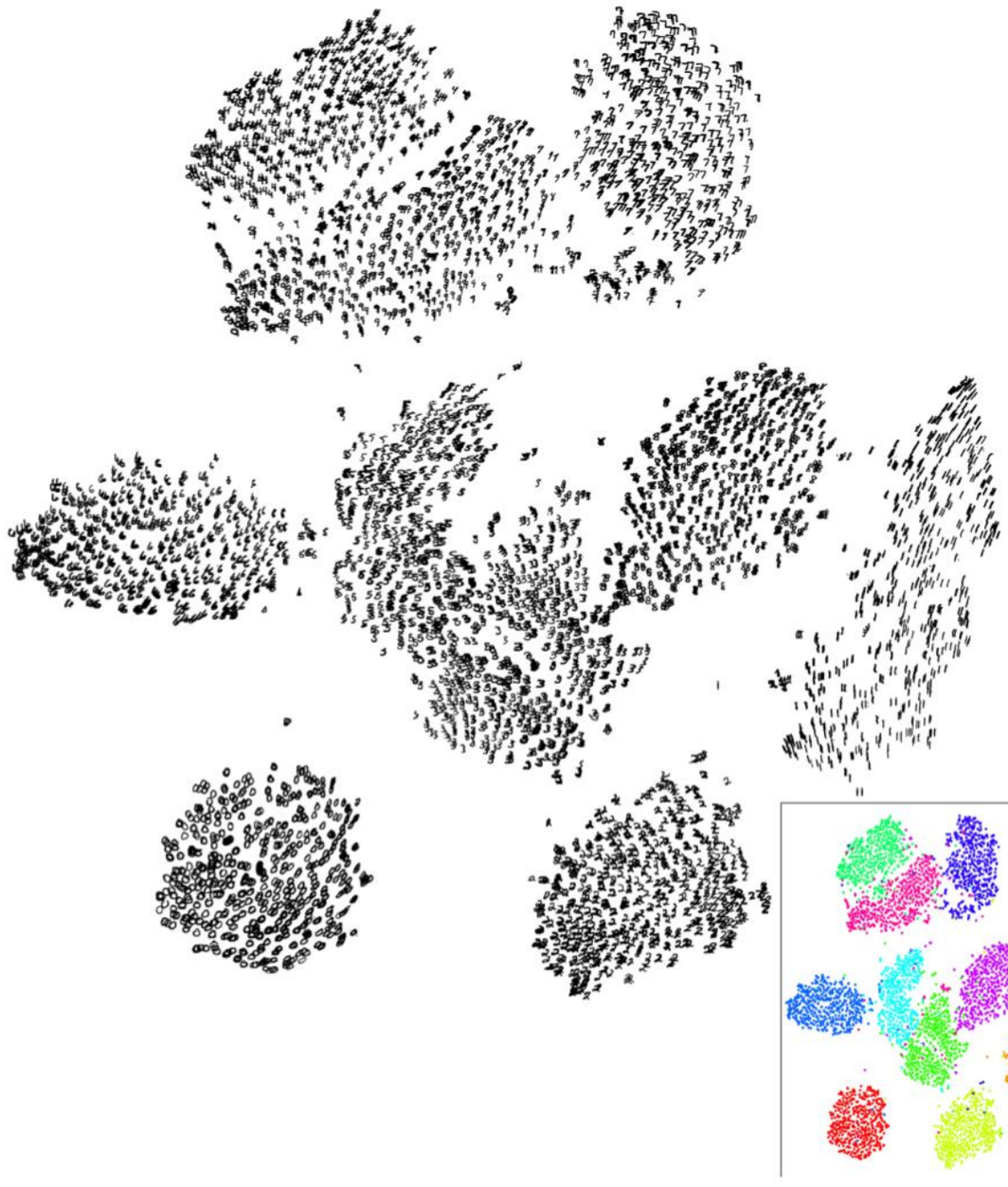
Visualizing MNIST with MDS

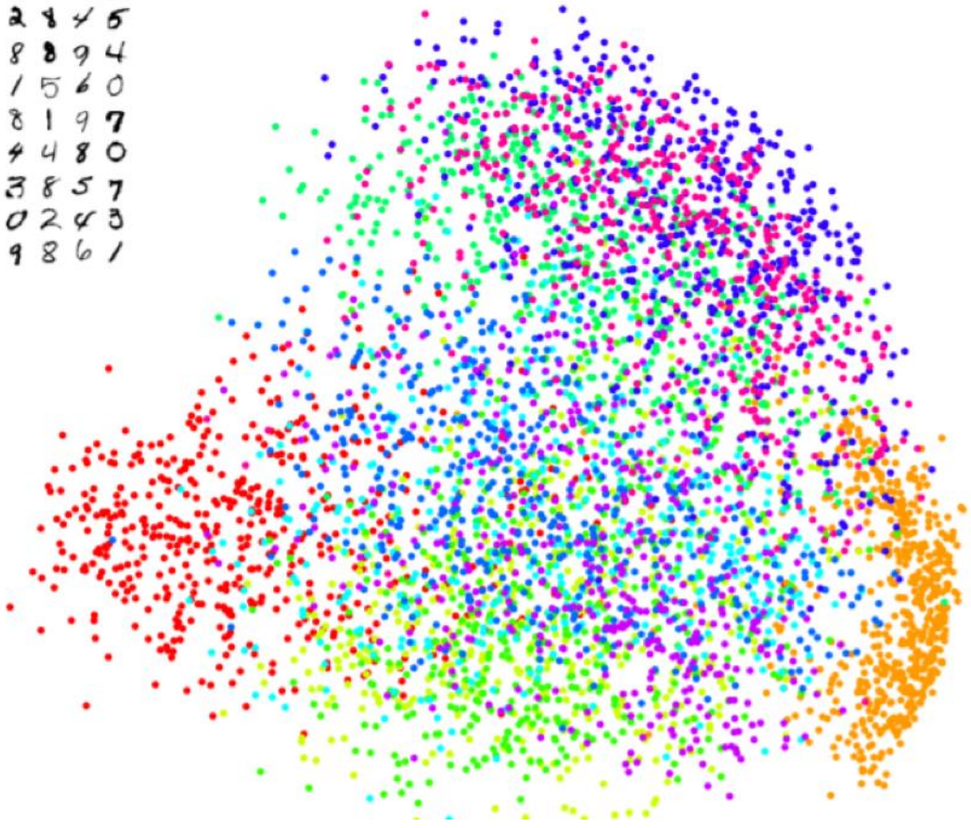
https://www.cnblogs.com/jins-note/p/9719157.html



Visualizing MNIST with Sammon's Mapping

tSNE vs. PCA





Stochastic Neighbor Embedding (SNE)

- First convert / encode each high-dimensional similarity into the probability that one data point will pick the other data point as its neighbor.

- Intuition: Random walk between data points.
  - High probability to jump to a close point

- Find low dimensional points such that their neighborhood distribution is similar.

- To evaluate a map:

  - Use the pairwise distances in the low-dimensional map to define the probability that a map point will pick another map point as its neighbor.
  - Compute the Kullback-Leibler divergence between the probabilities in the high-dimensional and low-dimensional spaces.

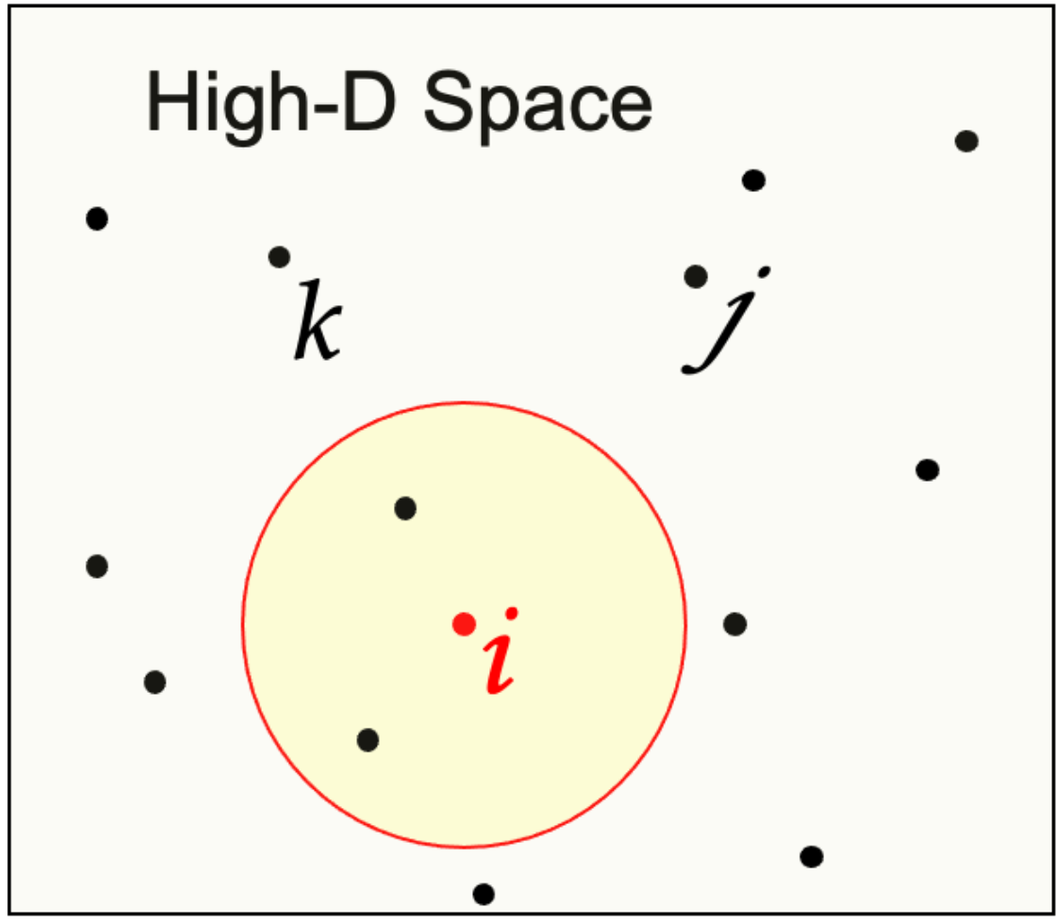
Neighborhood distributions

- Consider the neighborhood around an input data point $\mathbf{x}_i \in \mathbb{R}^d$
- Imagine that we have a Gaussian distribution centered around $\mathbf{x}_i$
- Then the probability that $\mathbf{x}_i$ chooses some other datapoint $\mathbf{x}_j$ as its neighbor is in proportion with the density under this Gaussian
- A point closer to $\mathbf{x}_i$ will be more likely than one further away

A probabilistic local method

- Each point in high-D has a conditional probability of picking each other point as its neighbor.

- The distribution over neighbors is based on the high-D pairwise distances.
  - If we do not have coordinates for the datapoints we can use a matrix of dissimilarities instead of pairwise distances.



$$p_{j|i} = \frac{e^{-d_{ij}^2 / 2\sigma_i^2}}{\sum_k e^{-d_{ik}^2 / 2\sigma_i^2}}$$

probability of picking j given that you start at i

Throwing away the raw data

- The probabilities that each points picks other points as its neighbor contains all of the information we are going to use for finding the manifold.
  - Once we have the probabilities $p_{j|i}$ we do not need to do any more computations in the high-dimensional space.
  - The input could be “dissimilarities” between pairs of datapoints instead of the locations of individual datapoints in a high-dimensional space.

SNE objective

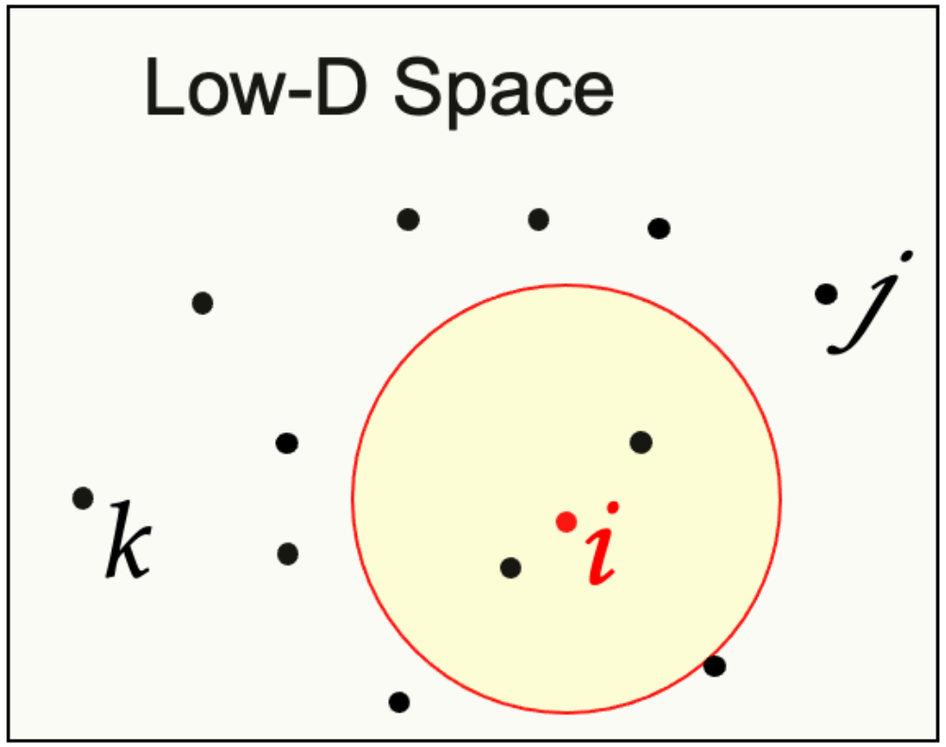
- Given $\mathbf{x}^{(1)}, .., \mathbf{x}^{(N)} \in \mathbb{R}^D$ we define the distribution P_{ij}
- Goal: Find good embedding $\mathbf{y}^{(1)}, .., \mathbf{y}^{(N)} \in \mathbb{R}^d$ for some $d < D$ (normally 2 or 3)
- How do we measure an embedding quality?
- For points $\mathbf{y}^{(1)}, .., \mathbf{y}^{(N)} \in \mathbb{R}^d$ we can define distribution Q similarly the same (notice no σ_i^2 and not symmetric)

$$Q_{ij} = \frac{\exp\left(-||\mathbf{y}^{(i)} - \mathbf{y}^{(j)}||^2\right)}{\sum_k \sum_{l \neq k} \exp\left(-||\mathbf{y}^{(l)} - \mathbf{y}^{(k)}||^2\right)}$$

- Optimize Q to be close to P
  - Minimize KL-divergence
- The embeddings $\mathbf{y}^{(1)}, .., \mathbf{y}^{(N)} \in \mathbb{R}^d$ are the parameters we are optimizing.
  - How do you embed a new point? No embedding function!

Evaluating an arrangement of the data in a low-dimensional space

- Give each datapoint a location in the low-dimensional space.
  - Evaluate this representation by seeing how well the low-D probabilities model the high-D ones.



$$q_{j|i} = \frac{e^{-d_{ij}^2}}{\sum_k e^{-d_{ik}^2}}$$

probability of picking j given that you start at i

The cost function for a low-dimensional representation

$$Cost = \sum_i KL(P_i \,||\, Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

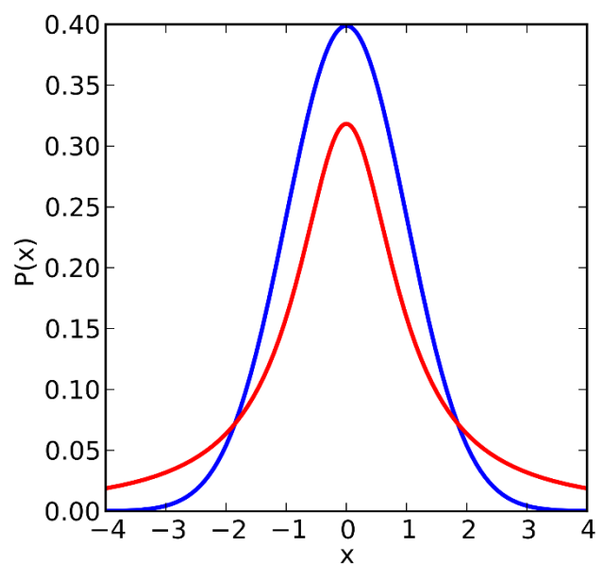
- For points where $p_{j|i}$ is large and $q_{j|i}$ is small we lose a lot.
  - Nearby points in high-D really want to be nearby in low-D
- For points where q_{ij} is large and p_{ij} is small we lose a little because we waste some of the probability mass in the Q_i distribution.
  - Widely separated points in high-D have a mild preference for being widely separated in low-D.

SNE algorithm

- We have P, and are looking for $\mathbf{y}^{(1)}, .., \mathbf{y}^{(N)} \in \mathbb{R}^d$ such that the distribution Q we infer will minimize $L(Q) = KL(P||Q)$ (notice Q on right, uncommon).
- Note that $KL(P||Q) = \sum_{ij} P_{ij} \log\left(\frac{P_{ij}}{Q_{ij}}\right) = -\sum_{ij} P_{ij} \log\left(Q_{ij}\right) + const$
- Can show that $\frac{\partial L}{\partial \mathbf{y}^{(i)}} = \sum_j (P_{ij} - Q_{ij})(\mathbf{y}^{(i)} - \mathbf{y}^{(j)})$
- Not a convex problem! No guarantees, can use multiple restarts.
- Main issue - crowding problem.

Crowding problem

- In high dimension we have more room, points can have a lot of different neighbors
- In 2D a point can have a few neighbors at distance one all far from each other - what happens when we embed in 1D?
- This is the "crowding problem" - we don't have enough room to accommodate all neighbors.
- This is one of the biggest problems with SNE.
- t-SNE solution: Change the Gaussian in Q to a heavy tailed distribution.
  - if Q changes slower, we have more "wiggle room" to place points at.



https://en.wikipedia.org/wiki/Student%27s_t-distribution#/media/File:T_distribution_1df_enhanced.svg

t-SNE

- SNE by Sam Roweis and Geoffrey Hinton in 2002
- Laurens van der Maaten — t-distributed variant (t-SNE) in 2008

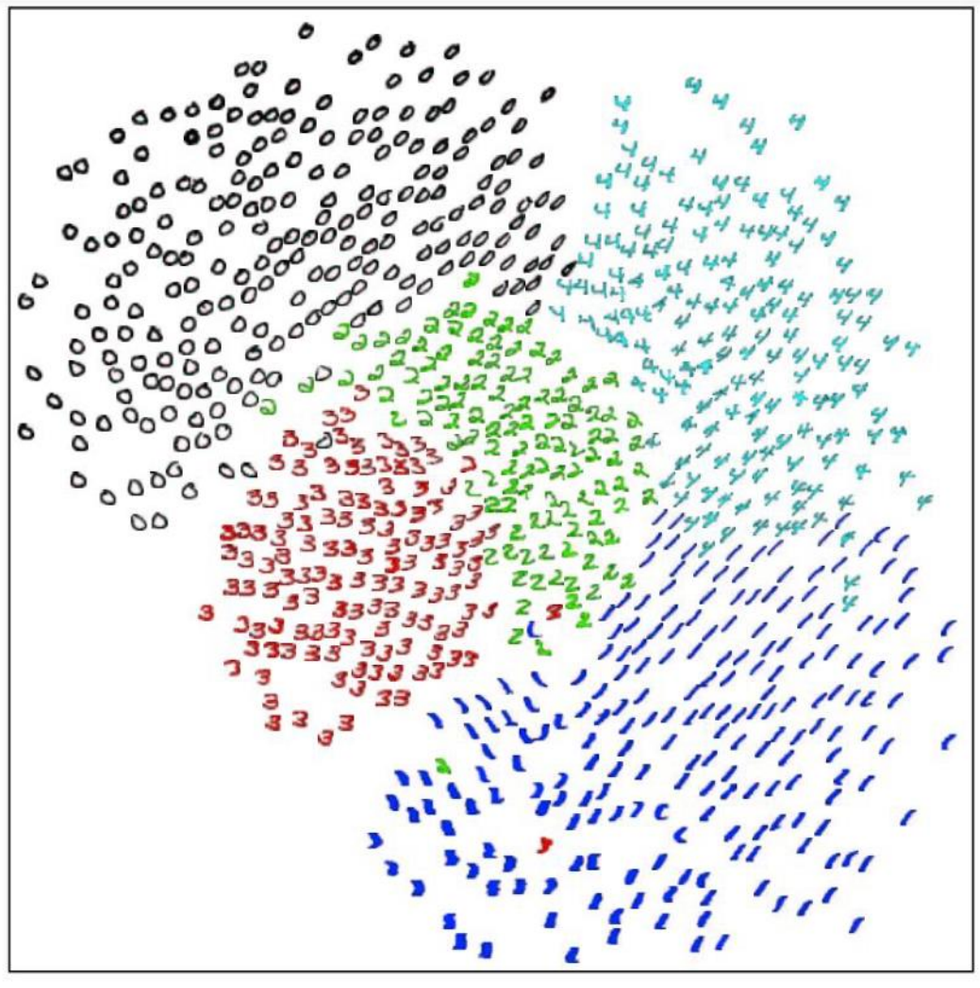
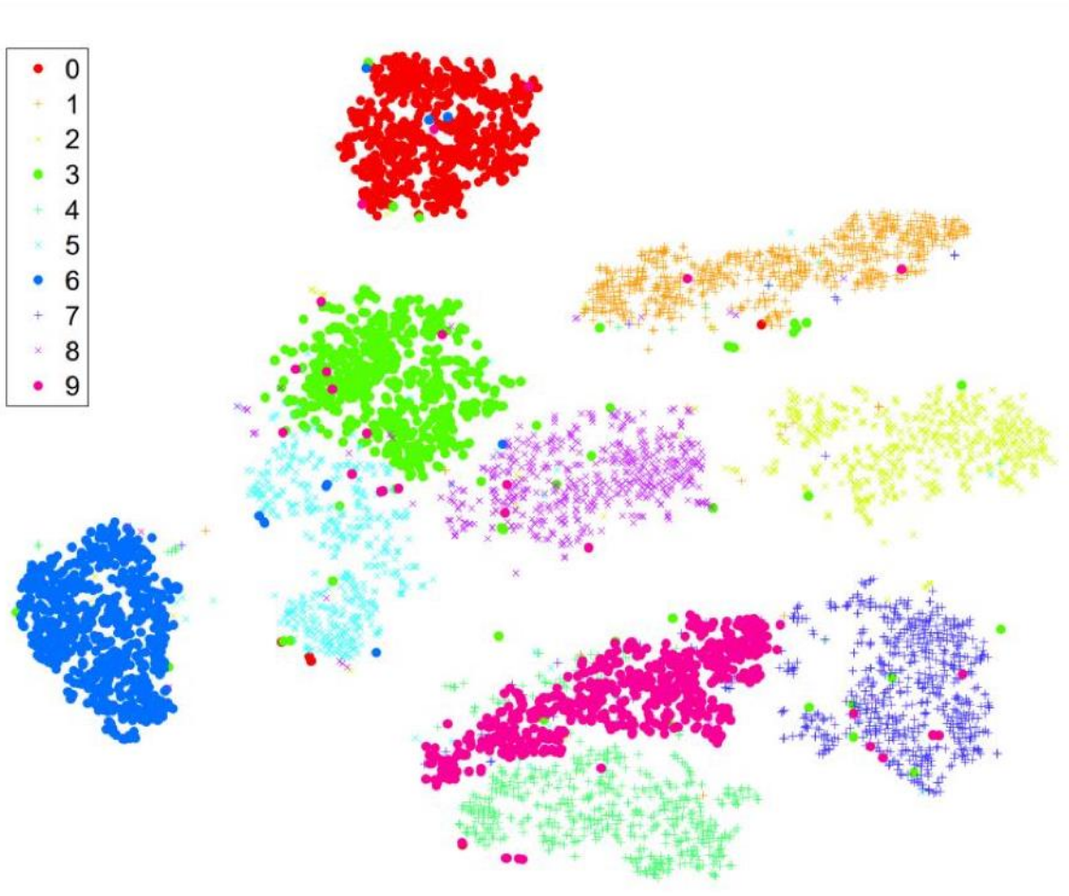


Figure 1: The result of running the SNE algorithm on 3000 256-dimensional grayscale images of handwritten digits. Pictures of the original data vectors $\mathbf{x}_i$ (scans of handwritten digit) are shown at the location corresponding to their low-dimensional images $\mathbf{y}_i$ as found by SNE. The classes are quite well separated even though SNE had no information about class labels. Furthermore, within each class, properties like orientation, skew and stroke-thickness tend to vary smoothly across the space. Not all points are shown: to produce this display, digits are chosen in random order and are only displayed if a 16 x 16 region of the display centered on the 2-D location of the digit in the embedding does not overlap any of the 16 x16 regions for digits that have already been displayed.



(a) Visualization by t-SNE.

t-SNE

t-Distributed Stochastic Neighbor Embedding

- Student-t Probability density $p(x) \propto (1 + \frac{x^2}{v})^{-(v+1)/2}$
  - for $v = 1$ we get $p(x) \propto \frac{1}{1+x^2}$
- Probability goes to zero much slower then a Gaussian.
- Can show it is equivalent to averaging Gaussians with some prior over σ^2
- We can now redefine Q_{ij} as

$$Q_{ij} = \frac{(1 + ||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}}{\sum_k \sum_{l \neq k} (1 + ||\mathbf{y}_k - \mathbf{y}_l||^2)^{-1}}$$

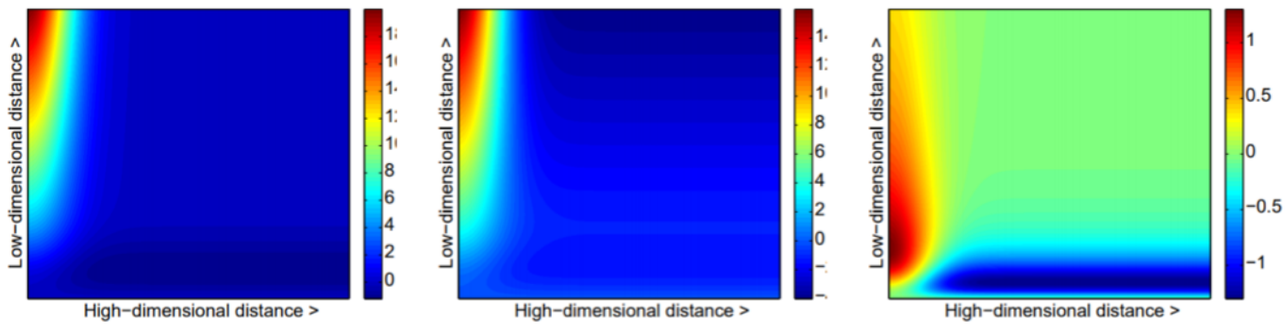
- We leave P_{ij} as is!

t-SNE gradients

- Can show that the gradients of t-SNE objective are

$$\frac{\partial L}{\partial \mathbf{y}^{(i)}} = \sum_j (P_{ij} - Q_{ij})(\mathbf{y}^{(i)} - \mathbf{y}^{(j)})(1 + ||y_i - y_j||^2)^{-1}$$

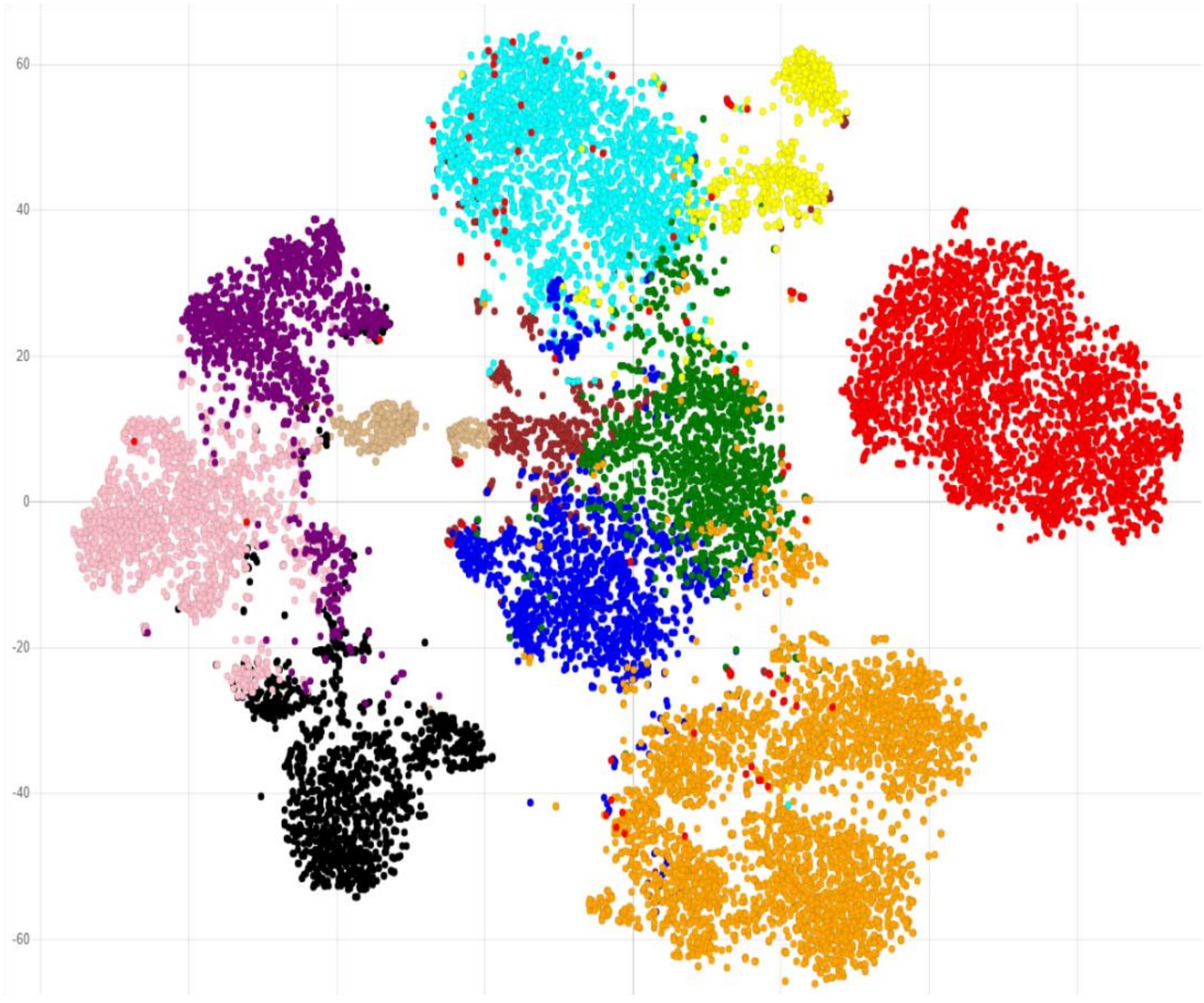
  - Compare to the SNE gradients: $\frac{\partial L}{\partial \mathbf{y}^{(i)}} = \sum_j (P_{ij} - Q_{ij})(\mathbf{y}^{(i)} - \mathbf{y}^{(j)})$



(a) Gradient of SNE. (b) Gradient of UNI-SNE. (c) Gradient of t-SNE.

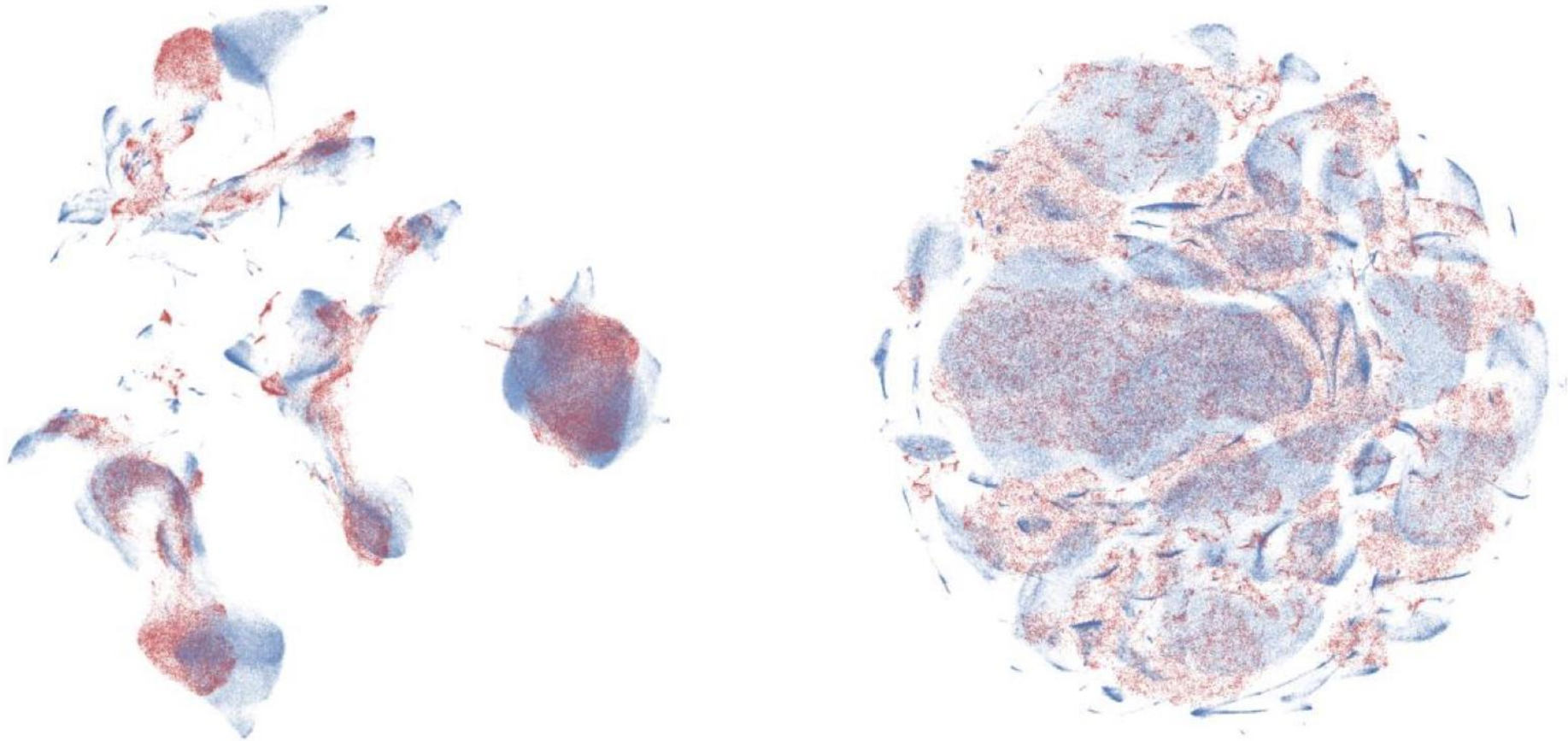
- Both repulse close dissimilar points and attract far similar points, but the $t - SNE$ has a smaller attraction term to solve crowding.

iPSC single cell data



t-SNE vs. UMAP

- UMAP better represents the global structure of the dataset
- UMAP is way faster than (unoptimized) t-SNE
- UMAP is more stable to subsampling than t-SNE
- UMAP can work directly in very high ambient dimensionalities

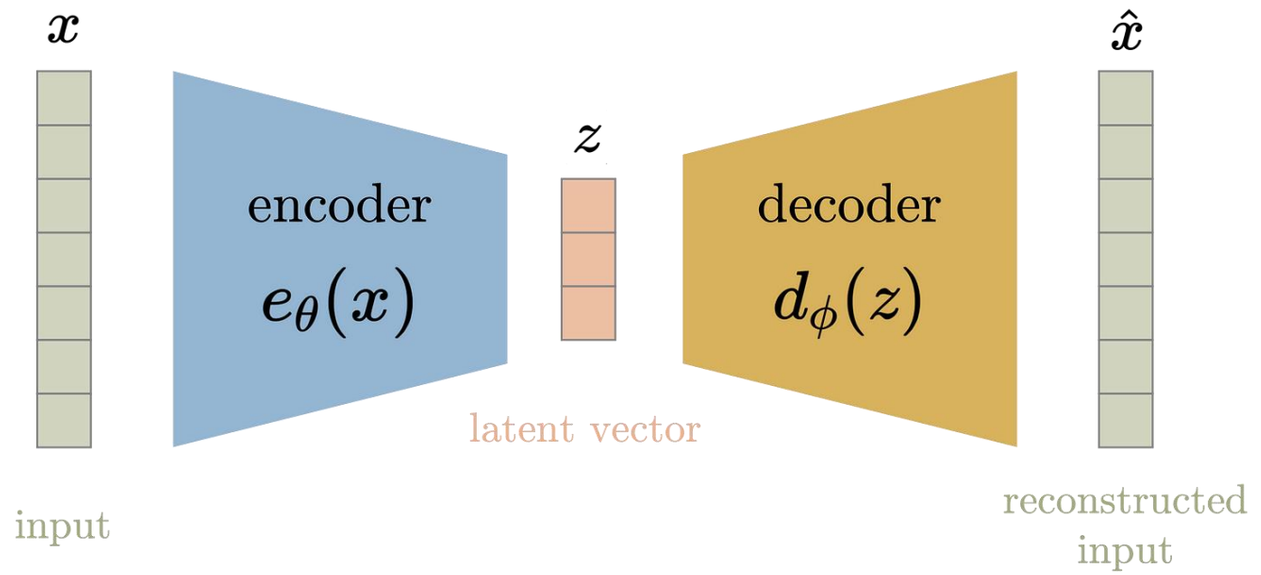


(a) UMAP (b) t-SNE

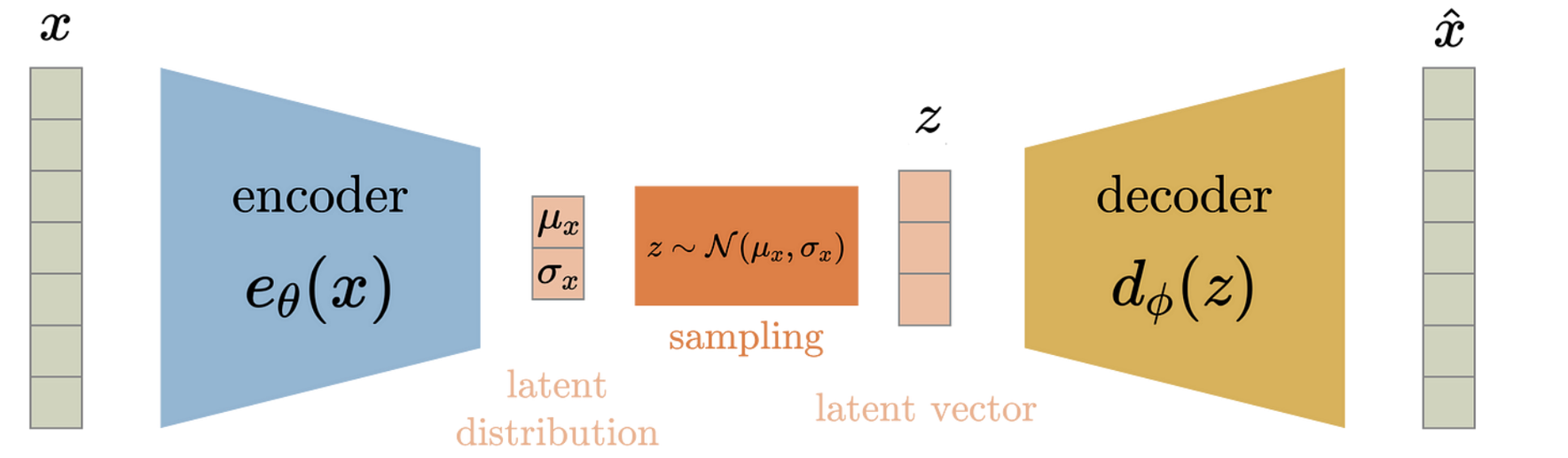
Exercise: Why? What is the mathematical relationship between UMAP and t-SNE. Hint: read Böhm, Berens, Kobak, “Attraction-Repulsion Spectrum in Neighbor Embeddings”.

Autoencoder dimensionality reduction

https://towardsdatascience.com/difference-between-autoencoder-ae-and-variational-autoencoder-vae-ed7be1c038f2



$$loss = \|x - \hat{x}\|_2 = \|x - d_\phi(z)\|_2 = \|x - d_\phi(e_\theta(x))\|_2$$



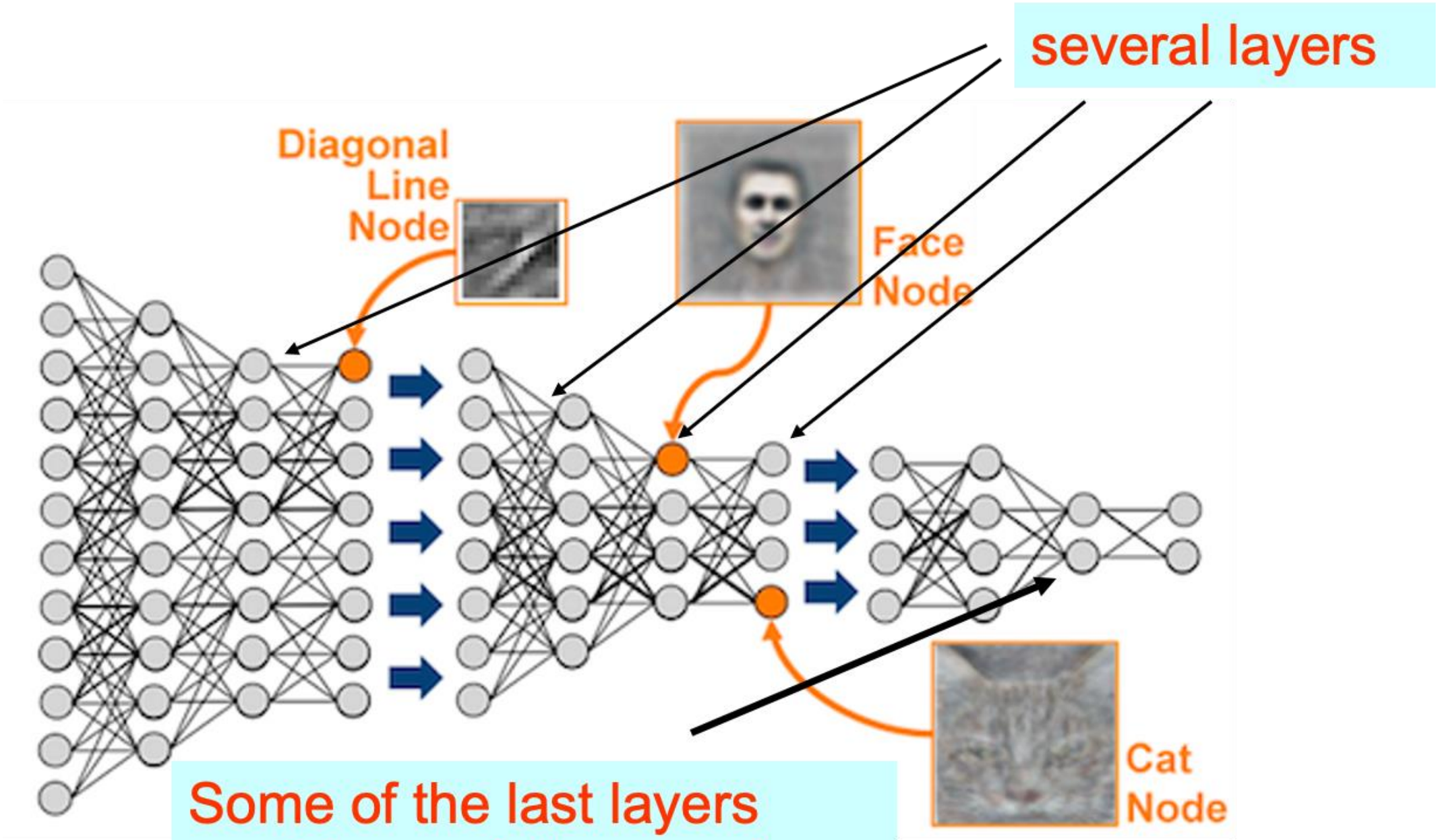
$$\text{reconstruction loss} = \|x - \hat{x}\|_2 = \|x - d_\phi(z)\|_2 = \|x - d_\phi(\mu_x + \sigma_x \epsilon)\|_2$$

$$\mu_x, \sigma_x = e_\theta(x), \quad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

$$similarity\ loss = KL\ Divergence = D_{KL}(\mathcal{N}(\mu_x, \sigma_x) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I}))$$

$$loss = reconstruction\ loss + similarity\ loss$$

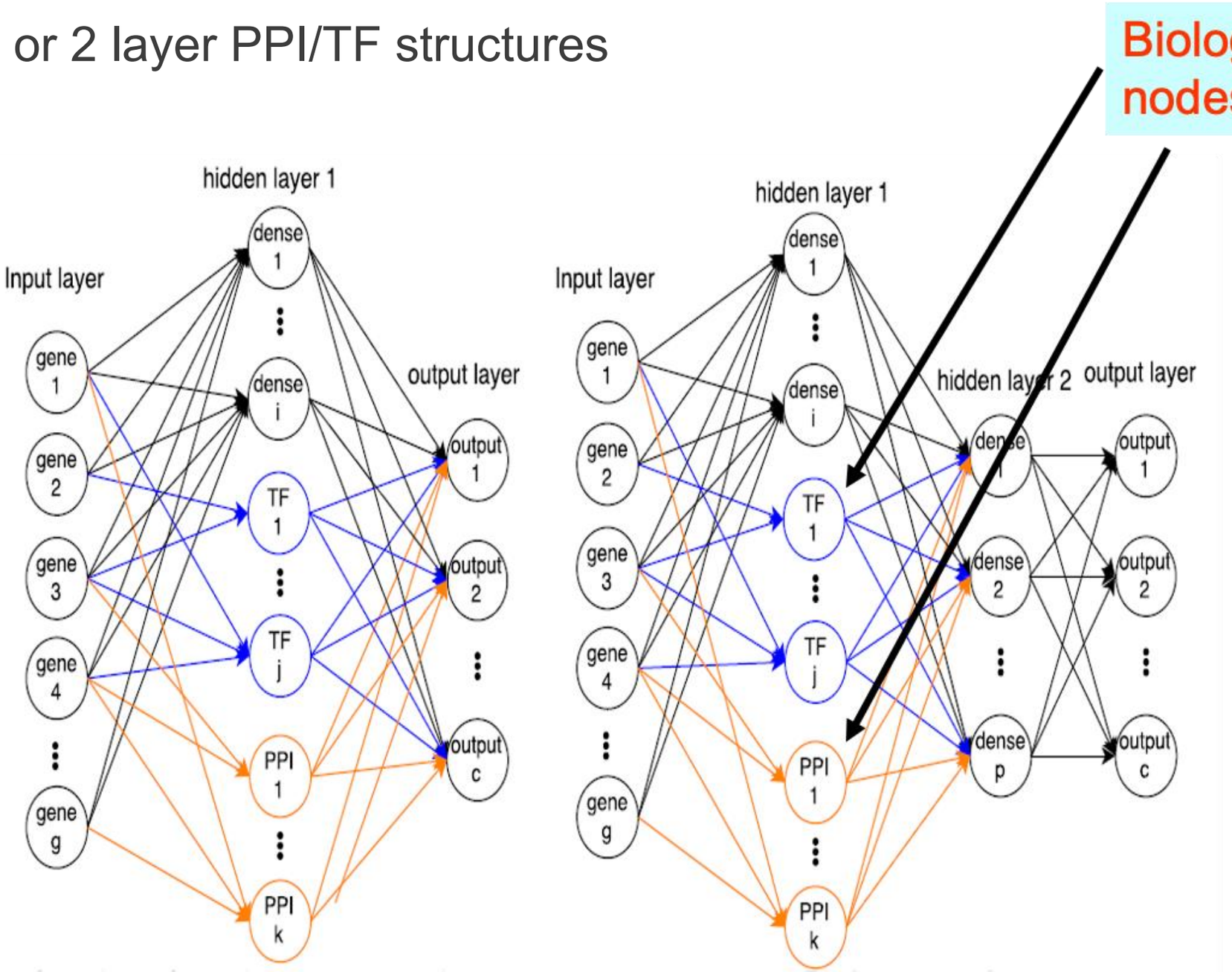
Supervised dimensionality reduction: Neural Networks



Some of the last layers contain much fewer values than the input!

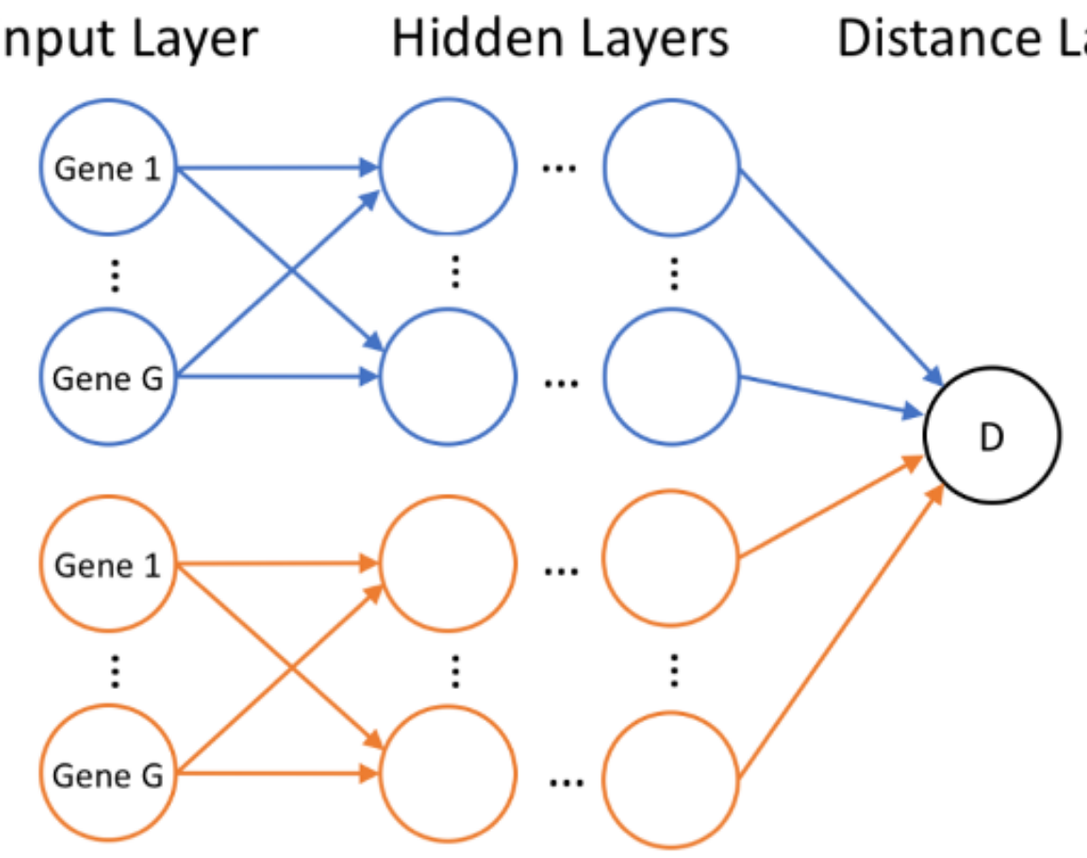
Adding prior biological knowledge

- Protein interactions
- Transcription factors
- 1 or 2 layer PPI/TF structures



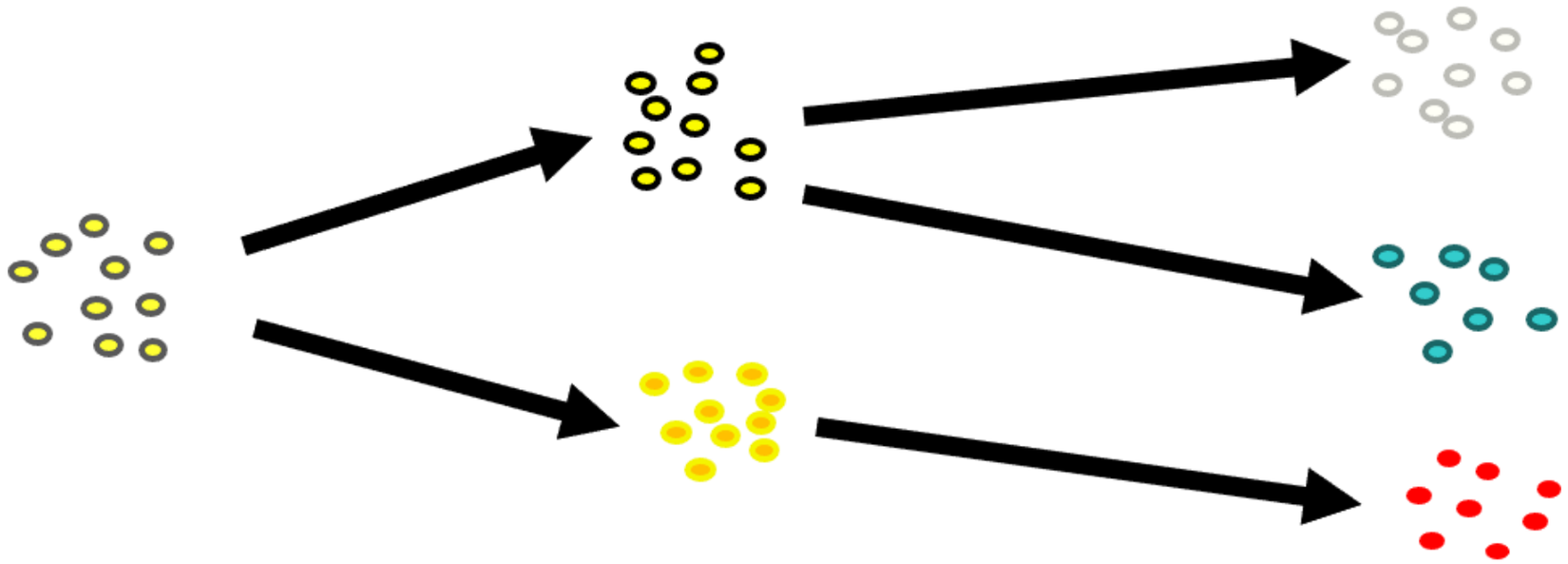
Additional NN architectures

- Siamese
  - Still supervised, but not trying to maximize accurate classification.
  - Instead, input is composed of pairs that are either similar or not
  - Output is binary label (similar or not)
  - Thus, these networks directly optimize a reduced dimension layer for KNN.



Determining temporal trajectories driving cellular differentiation

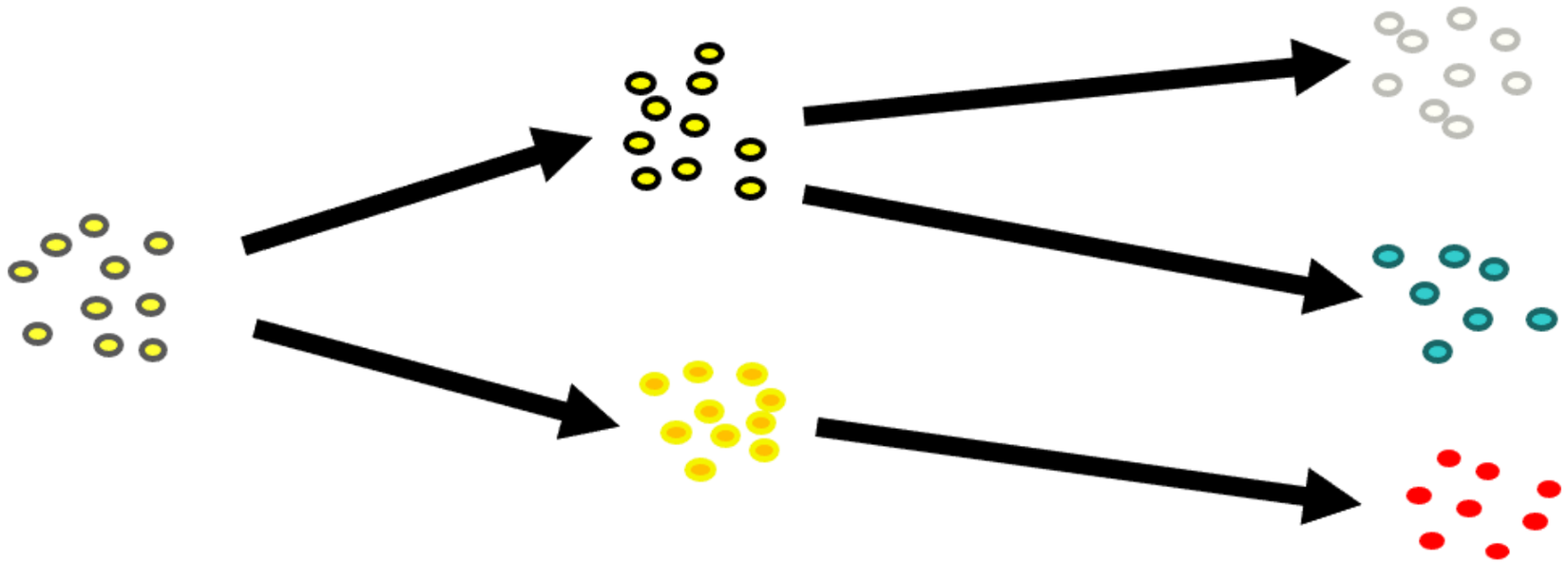
- Development, stem cell differentiation, cell fate decisions etc. are now studied using high throughput single cell data
- They often share the following attributes
  - The process ends in several distinct, though not necessarily known, states (cell fates, cell states)
  - It starts with a set of progenitor, or undifferentiated, cells
  - A key goal is to understand the branching process and regulatory networks that govern the differentiation process



New challenges when modeling time series single cell data

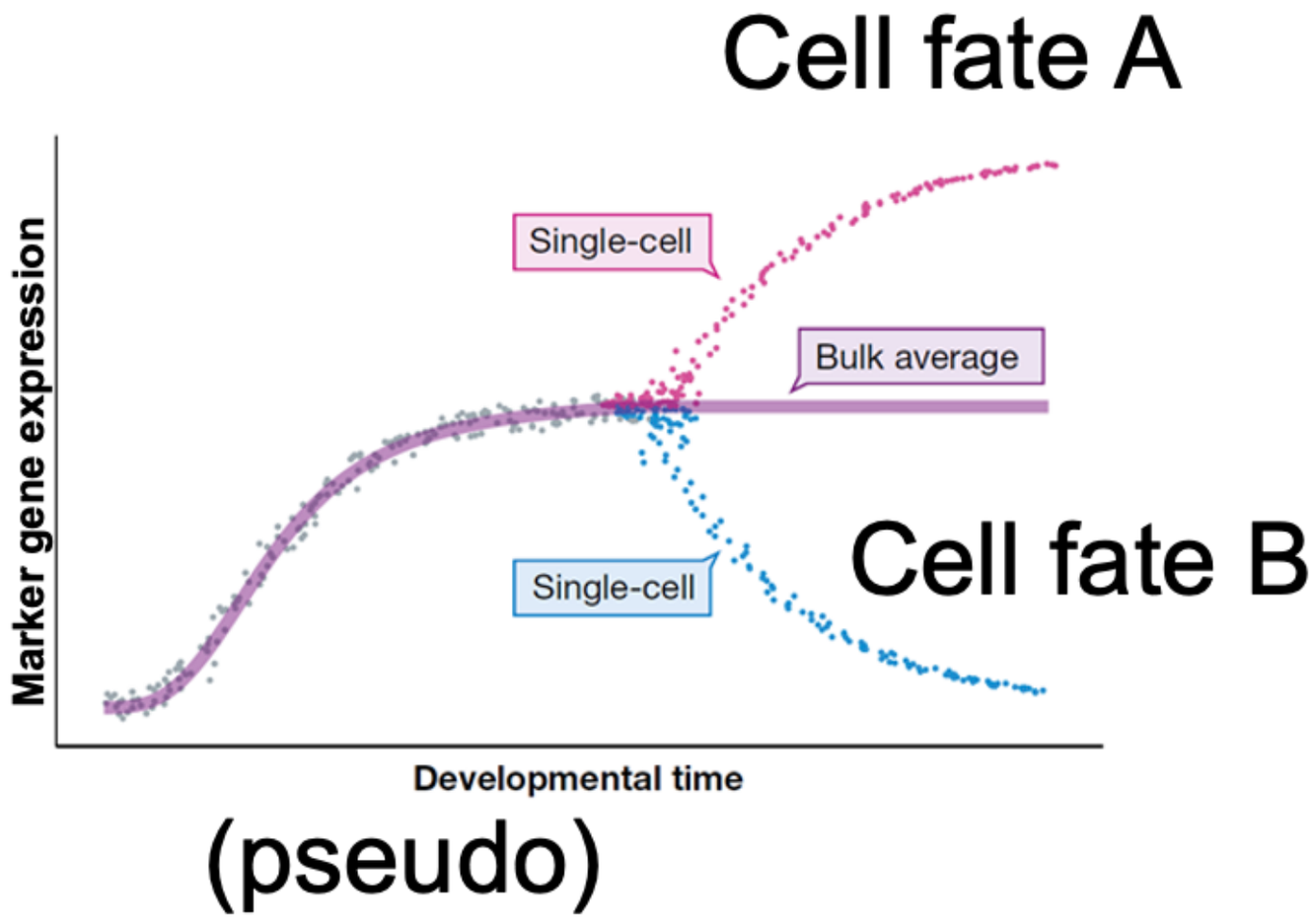
- While the temporal trajectories are clear when there are no branches (or when using whole tissue / organism data), it is not clear how to link single cells across different time points

- Temporal information is not always correlated with cell fate / condition



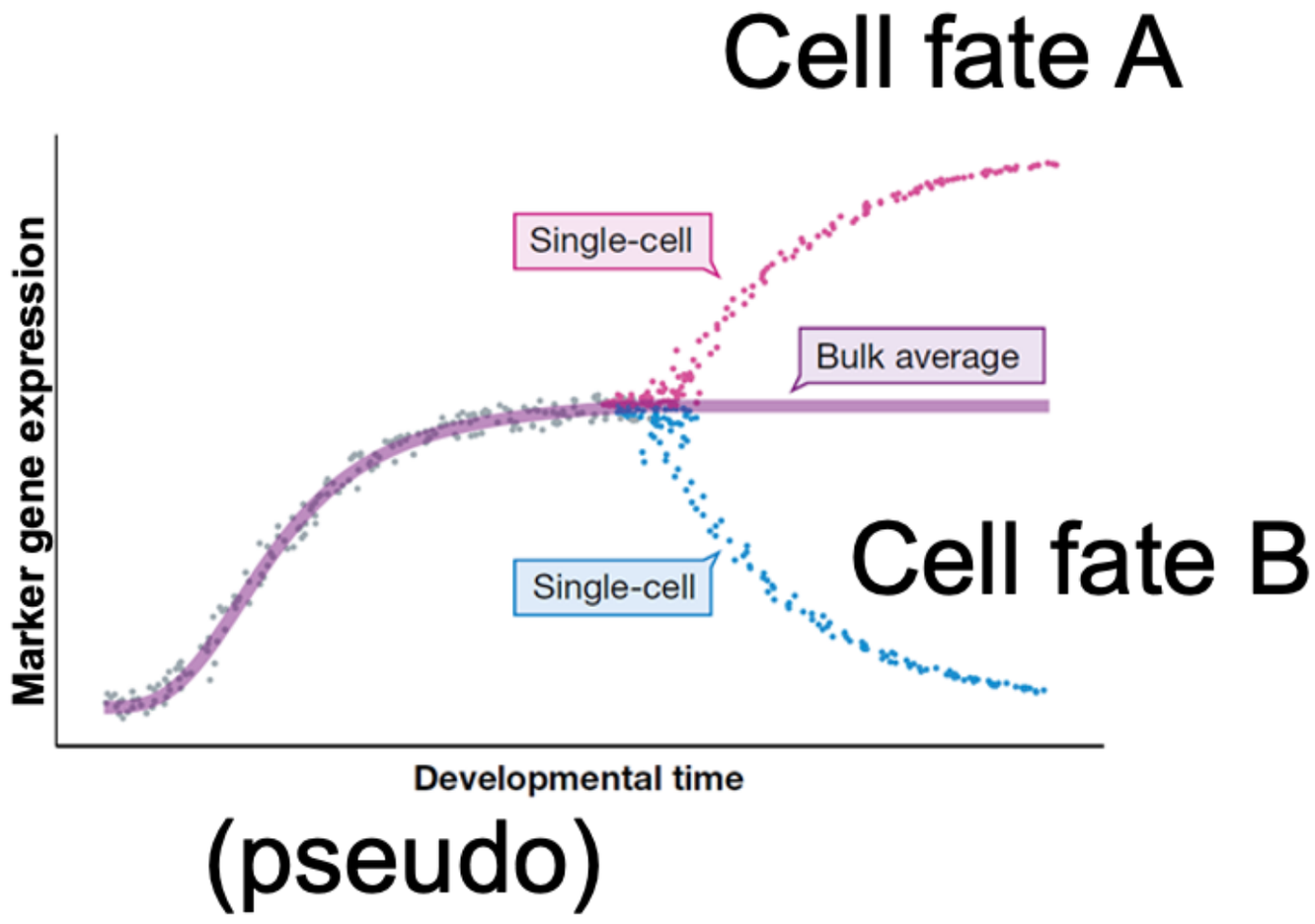
Reconstructing cell fate trajectories using time series scRNA-Seq data

- Two primary directions:
  - Dimensionality reduction-based
  - Probabilistic graphical model-based



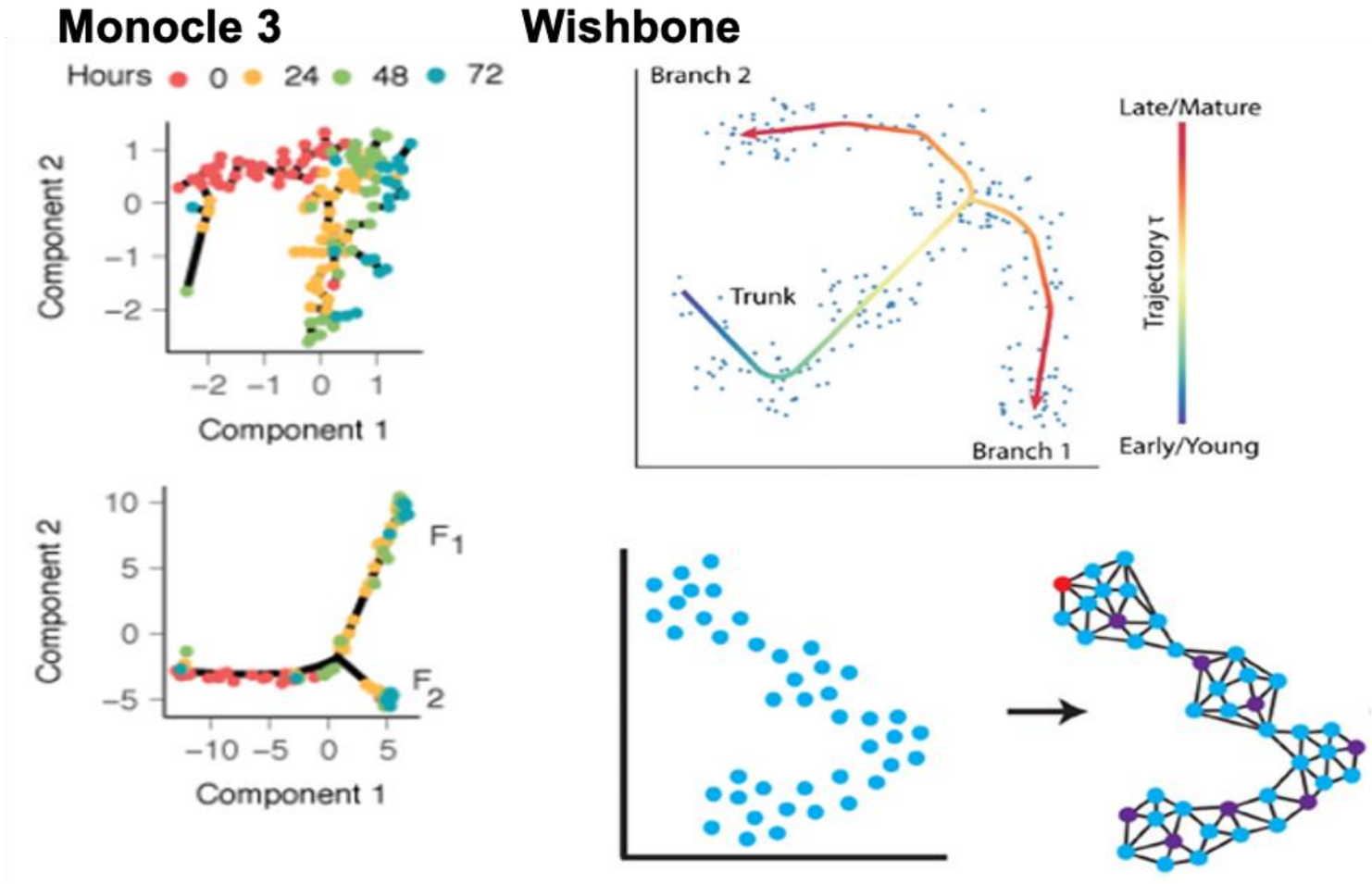
Reconstructing cell fate trajectories using time series scRNA-Seq data

- Dimensionality reduction-based
  - Perform dimensionality reduction
  - Identify cell trajectories and assign pseudo time ordering
  - Usually using MST or other tree based approaches



Dimensionality reduction-based methods

- Advantages:
  - Easy visualization (2D or 3D)
  - Continuous cell trajectories
- Disadvantages:
  - Highly dependent on dimensionality reduction methods
  - Lose a lot of information



Reconstructing cell fate trajectories using time series scRNA-Seq data

- Probabilistic graphical models
  - Identify a tree structure based on clustering
  - Define emission probability on each node and assign cells based on likelihood

- Advantages:
  - Can apply Expectation-Maximization (EM) algorithms to iteratively improve the model
  - Can use all the genes, does not lose information

- Disadvantages:
  - Small number of discrete states cannot account for larger number of biological states
  - Similar cells could be distant on the model