Non-hierarchical Clustering and dimensionality reduction techniques

Mikhail Dozmorov
Fall 2017

K-means clustering

- k-means clustering is a method of cluster analysis which aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean.
- It is similar to the expectation-maximization algorithm for mixtures of Gaussians in that they both attempt to find the centers of natural clusters in the data.

How does K-Means work?

- We would like to partition that data set into K clusters $C_1, \ldots, C_K$
- Each observation belong to at least one of the K clusters
- The clusters are non-overlapping, i.e. no observation belongs to more than one cluster
- The objective is to have a minimal “within-cluster-variation”, i.e. the elements within a cluster should be as similar as possible
- One way of achieving this is to minimize the sum of all the pair-wise squared Euclidean distances between the observations in each cluster.

$$minimize_{C_1,\ldots,C_K}\{\sum_{k=1}^{K}\frac{1}{|C_k|}\sum_{i,i'\in C_k}\sum_{j=1}^{p}(x_{ij}-x_{i'j})^2\}$$

K-means clustering algorithm

- Initialize: choose k points as cluster means
- Repeat until convergence:
  - Assignment: place each point X_i in the cluster with the closest mean.
  - Update: recalculate the mean for each cluster
- K-means always converges.
  - The assignment and update steps always either reduce the objective function or leave it unchanged.

K-means clustering algorithm

```
Begin
    Assign each item a class in 1 to K (randomly)
    For 1 to max-iteration {
        For each class 1 to K {
            Calculate centroid (one of the "K means")
            Calculate distance from centroid to each item
        }
        Assign each item the class of the nearest centroid
        Exit if no items are re-assigned (convergence)
    }
End
```

J. B. MacQueen "**Some Methods for classification and Analysis of Multivariate Observations**" 1967 https://projecteuclid.org/euclid.bsmsp/1200512992

K-means clustering

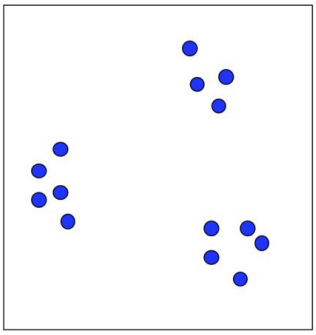
- Advantage: gives sharp partitions of the data
- Disadvantage: need to specify the number of clusters (K).
- Goal: find a set of k clusters that minimizes the distances of each point in the cluster to the cluster mean:

$$centroid_j = \hat{Y}_j = \frac{1}{N_{Y_j}} \sum_{i \in Y_j} X_i$$

$$argmin_C \sum_{i=1}^{k} \sum_{j \in C(i)} |X_j - \hat{Y}_i|^2$$

K-means steps

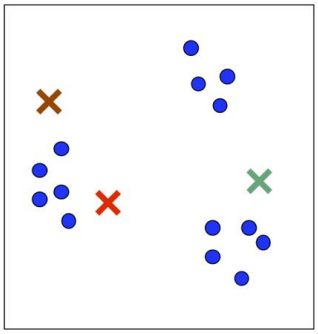
- Simplified example
- Expression for two genes for 14 samples
- Some structure can be seen



Iteration = 0

K-means steps

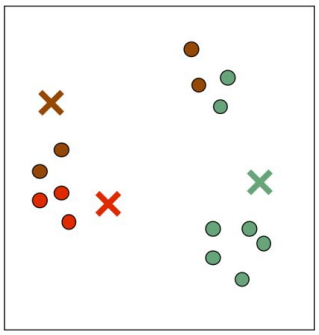
- Choose K centroids
- These are starting values that the user picks.
- There are some data driven ways to do it



Iteration = 0

K-means steps

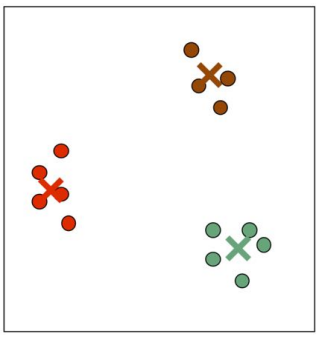
- Find the closest centroid for each point
- This is where distance is used
- This is "first partition" into K clusters



Iteration = 1

K-means steps

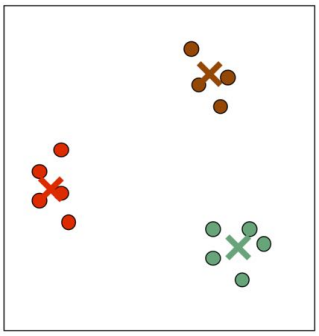
- Take the middle of each cluster
- Re-compute centroids in relation to the middle
- Use the new centroids to calculate distance



Iteration = 3

K-means steps

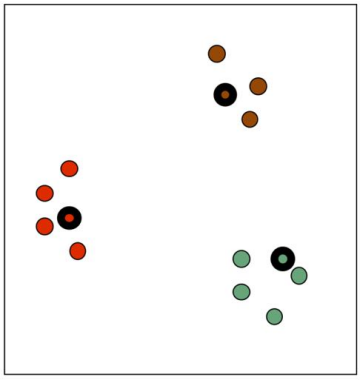
- Expression for two genes for 14 samples



Iteration = 3

PAM (K-medoids)

- **Centroid** - The average of the samples within a cluster
- **Medoid** - The “representative object” within a cluster
- Initializing requires choosing medoids at random.



K-means limitations

- Final results depend on starting values
- How do we chose K? There are methods but not much theory saying what is best.
- Where are the pretty pictures?

Alternatives

K-means

- Initialize: choose k points as cluster means
- Repeat until convergence:
  - Assignment: place each point X_i in the cluster with the closest mean.
  - Update: recalculate the mean for each cluster

Fuzzy k-means

- Initialize: choose k points as cluster means
- Repeat until convergence:
  - Assignment: calculate probability of each point belonging to each cluster.
  - Update: recalculate the mean for each cluster using these probabilities

Alternatives

K-means

$$\underset{C}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{j \in C(i)} \left| X_j - \hat{Y}_i \right|^2$$

$$\text{centroid}_j = \hat{Y}_j = \frac{1}{N_{Y_j}} \sum_{i \in Y_j} X_i$$

Fuzzy k-means

$$\underset{\mu, Y}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{j=1}^{N} \mu_{i,j}^{r} \left| X_j - \hat{Y}_i \right|^2$$

$$\text{centroid}_j = \hat{Y}_j = \frac{\sum_{i=1}^{N} \mu_{i,j}^{r} X_i}{\sum_{i=1}^{N} \mu_{i,j}^{r}}$$

- $\mu_{i,j}^{r}$ is the degree of membership of x_i in the cluster j. Larger values of r make the clusters more fuzzy.
- Relationship to EM and Gaussian mixture models

https://home.deib.polimi.it/matteucc/Clustering/tutorial_html/cmeans.html

Limits of K-means

K-means uses Euclidean distance

$$centroid_j = \hat{Y}_j = \frac{1}{N_{Y_j}} \sum_{i \in Y_j} X_i$$

$$argmin_C \sum_{i=1}^{k} \sum_{j \in C(i)} |X_j - \hat{Y}_i|^2$$

- Gives most weight to largest differences
- Can't be used if data are qualitative
- Centroid usually does not represent any datum

Self-organizing (Kohonen) maps

- Self organizing map (SOM) is a learning method which produces low dimension data (e.g. $2D$) from high dimension data (nD) through the use of self-organizing neural networks
- E.g. an apple is different from a banana in more then two ways but they can be differentiated based on their size and color only.

Projection methods

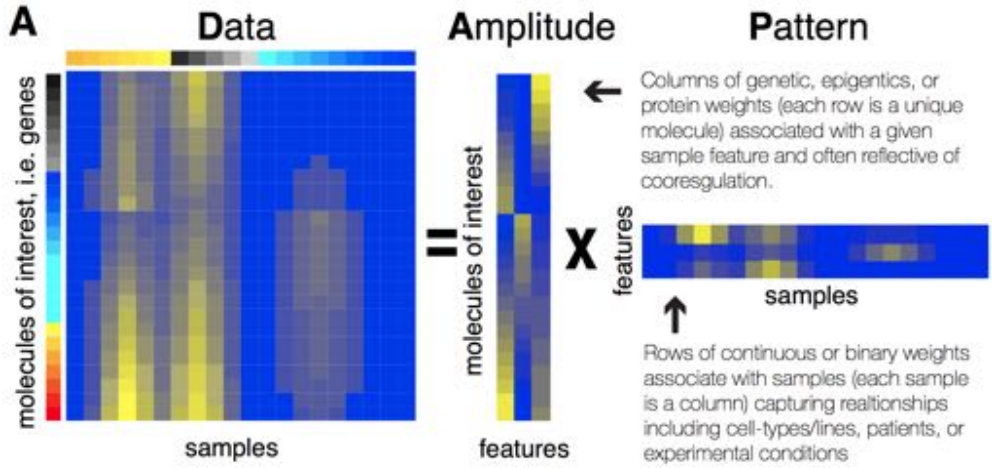
Projection (dimensionality reduction) methods

- Linearly decompose the dataset into components that have a desired property.
- There are largely two kinds of projection methods: principal component analysis (PCA) and independent component analysis (ICA).
- PCA produces a low-dimensisonal representation of a dataset.
- Each successive principal component is selected to be orthonormal to the previous ones, and to capture the maximum information that is not already present in the previous components.
- Components are linear combinations of the original data
- PCA also serves as a tool for data visualization

Why dimensionality reduction

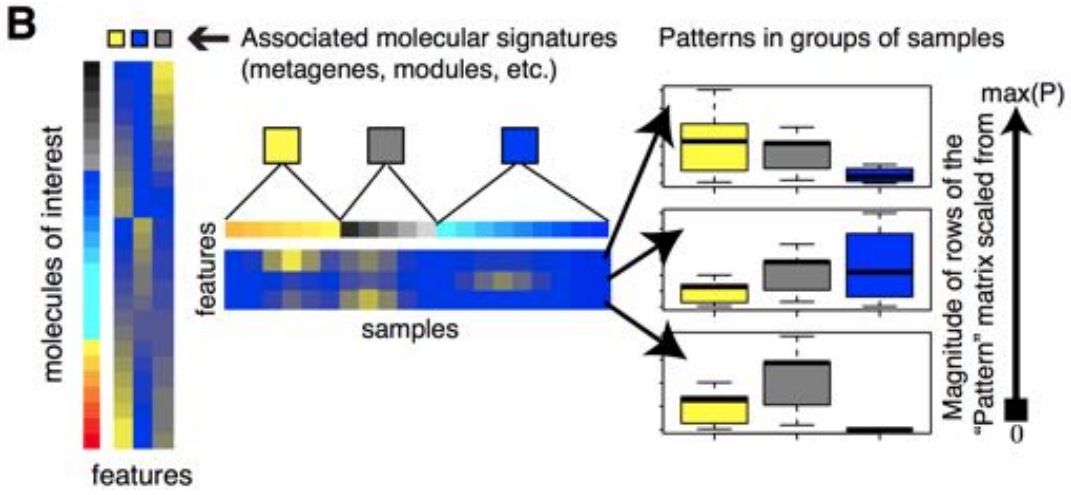
- Start with many measurements (high dimensional).
- Want to reduce to few features (lower-dimensional space).
- One way is to extract features based on capturing groups of variance.
- Another could be to preferentially select some of the current features most representative of the data.

Intuition behind dimensionality reduction



https://www.biorxiv.org/content/early/2017/10/02/196915.1

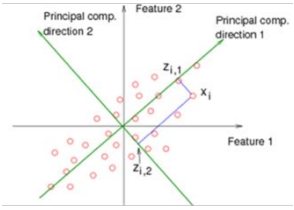
Intuition behind dimensionality reduction



https://www.biorxiv.org/content/early/2017/10/02/196915.1

PCA: quick theory

- Eigenvectors of covariance matrix.
- Find orthogonal groups of variance.
- Given from most to least variance.
- Components of variation.
- Linear combinations explaining the variance.



http://setosa.io/ev/principal-component-analysis/

Principal Components Analysis

- Performs a rotation of the data that maximizes the variance in the new axes
- Projects high dimensional data into a low dimensional sub-space (visualized in 2-3 dims)
- Often captures much of the total data variation in a few dimensions (< 5)
- Exact solutions require a fully determined system (matrix with full rank), i.e. a “square” matrix with independent rows

Principal Components Analysis: details

- The first principal component of a set of features $X_1, X_2, \ldots, X_p$ is the normalized linear combination of the features

$$Z_1 = \phi_{11} X_1 + \phi_{21} X_2 + \ldots + \phi_{p1} X_p$$

that has the largest variance. Note "normalized" - $\sum_{j=1}^{p} \phi_{j1}^2 = 1$

- The elements $\phi_{11}, \phi_{21}, \ldots \phi_{p1}$ are the **loadings** of the first principal component. Together, them make up the principal component loading vector $\phi_1 = (\phi_{11}, \phi_{21}, \ldots \phi_{p1})^T$
- The loadings are constrained so that their sum of squares is equal to one, since othewise setting these elements to be arbitrary large in absolute value couldd result in an arbitrary large variance.

Computation of Principal Components

- Input: a $n \times p$ data set X. Since we are only interested in variance, we assume that each of the variables in X has been centered to have mean zero (that is, the column means of X are zero).
- We then look for the linear combination of the sample feature falues of the form

$$z_{i1} = \phi_{11} x_{i1} + \phi_{21} x_{i2} + \ldots + \phi_{p1} x_{ip}$$

for $i = 1, \ldots, n$ that has largest sample variance under the constraint that $\sum_{j=1}^{p} \phi_{j1}^2 = 1$

- Since each of the x_{ij} has mean zero, so does z_{i1}.
- Hence the sample variance of the z_{i1} can be written as $\frac{1}{n} \sum_{i=1}^{n} z_{i1}^2$

Computation of Principal Components

- Plugging in the sample variance equation the first principal component loading vector solves the optimization problem

$$maximize_{\phi_{11},\ldots\phi_{p1}} \frac{1}{n} \sum_{i=1}^{n} (\sum_{j=1}^{p} \phi_{j1} x_{ij})^2$$

subject to $\sum_{j=1}^{p} \phi_{j1}^2 = 1$

- The problem can be solved via a singular value decomposition of the matrix X
- Z_1 is the first principal component with values $z_{11}, \ldots z_{n1}$

Geometry of PCA

- The loading vector ϕ_1 with elements $\phi_{11}, \phi_{21}, \ldots \phi_{p1}$ defines a direction in feature space along which the data vary the most
- If we project the n data points $x_1, \ldots, x_n$ onto this direction, the projected values (the new coordinates) are the principal component scores $z_{11}, \ldots z_{n1}$ themselves

Further principal components

- The second principal component is the linear combination of $X_1, \ldots X_p$ that has maximal variance among all linear combinations that are *uncorrelated* with Z_1
- The second ptincipal component scores $z_{12}, \ldots z_{n2}$ take the form

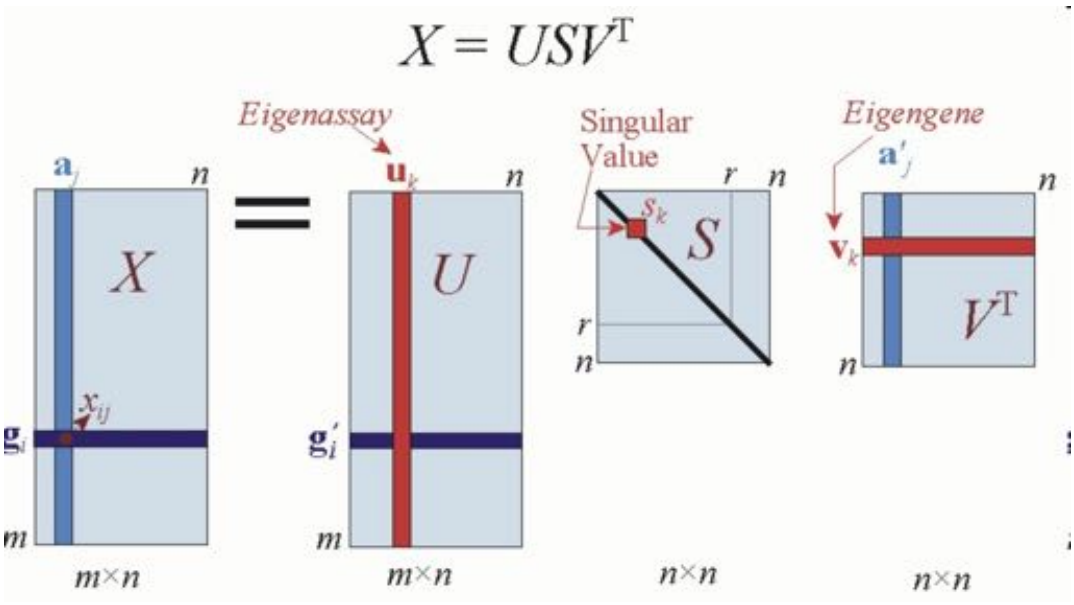
$$z_{i1} = \phi_{12} x_{i1} + \phi_{22} x_{i2} + \ldots + \phi_{p2} x_{ip}$$

where ϕ_2 is the second principal component loading vector, with elements $\phi_{12}, \phi_{22}, \ldots \phi_{p2}$

Further principal components

- Constraining Z_2 to be uncorrelated with Z_1 is equivalent to constraining the direction ϕ_2 to be orthogonal to the direction ϕ_1. And so on for the other components
- The principal component directions $\phi_1, \phi_2, \phi_3, \ldots$ are the ordered sequence of right singular vectors of the matrix X
- The variances of the components are the $\frac{1}{n}$ times the squares of the singular values
- There are at most $min(n-1, p)$ principal components

Singular Value Decomposition



https://research.fb.com/fast-randomized-svd/

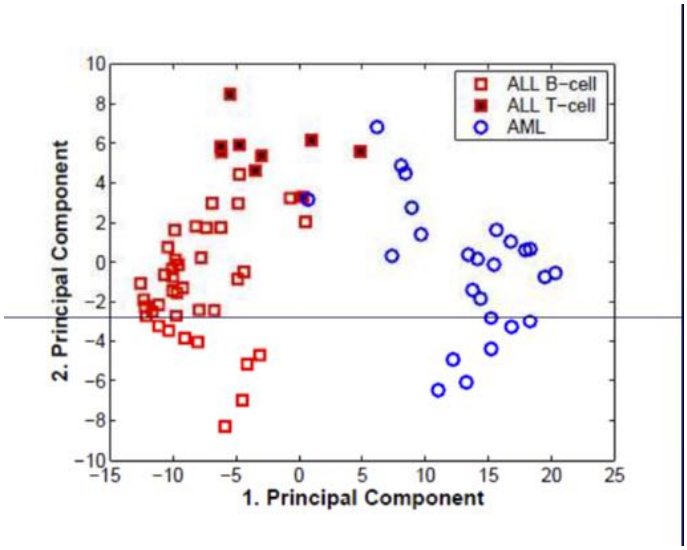
PCA for gene expression

- Given a gene-by-sample matrix X we decompose (centered and scaled) X as USV^T
- We don't usually care about total expression level and the dynamic range which may be dependent on technical factors
- U, V are orthonormal
- S diagonal-elements are eigenvalues = variance explained

PCA for gene expression

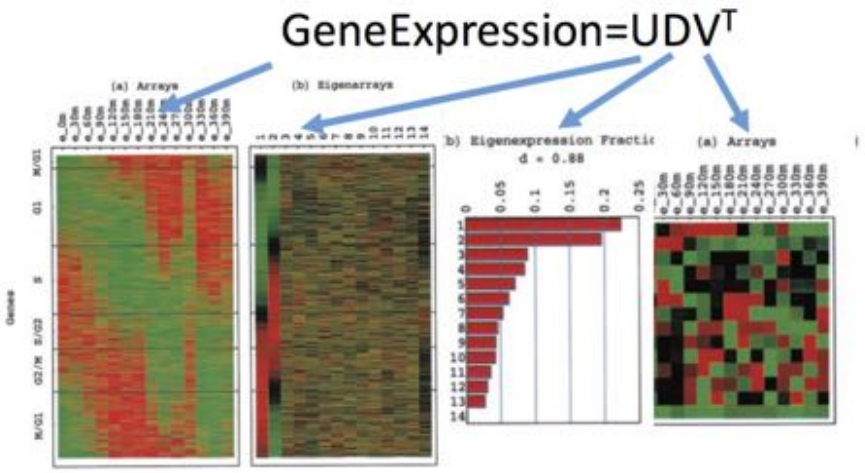
- Columns of V are
  - Principle components
  - Eigengenes/metagenes that span the space of the gene transcriptional responses
- Columns of U are
  - The “loadings”, or the correlation between the column and the component
  - Eigenarrays/metaarrays - span the space of the gene transcriptional responses
- Truncating U, V, D to the first k dimensions gives the best k-rank approximation of X

Principal Components Analysis



Example: Leukemia data sets by Golub et al.: Classification of ALL and AML

PCA applied to cell cycle data



Singular value decomposition for genome-wide expression data processing and modeling. *PNAS*

ICA - Independent Component Analysis

- PCA assumes multivariate normally distributed data - gene expression data are super-Gaussian
- ICA models observations as a linear combination of latent feature variables, or components, which are chosen to be as *statistically independent* as possible.
- For microarray data, observations consist of microarray gene expression measurements, and independent components are interpreted to be transcriptional modules that often correspond to specific biological processes

http://www.sciencedirect.com/science/article/pii/S1532046410001000

ICA - Independent Component Analysis

- X - an $m \times n$ matrix of n genes and m experiments
- ICA models this expression matrix as a linear combination of intependent biological processes by decomposing X as:

$$X = AS$$

- S is a $k \times n$ source matrix
- A is a $m \times k$ mixing matrix
- k is a user supplied parameter $\leq min(m, n)$

Same preprocessing as for PCA - filter, center, scale

ICA - Independent Component Analysis

- S is a $k \times n$ source matrix
- The components, or rows of S, are independent in the sense that the gene weights in each component reflect samplings of independent random variables.
- In the context of gene expression, this suggests that the sets of genes comprising the groups strongly contributing to each component have independent compositions.
- Columns of A are the distribution of the component's expression in arrays (rows of S)

`fastICA` R package, https://cran.r-project.org/web/packages/fastICA/index.html

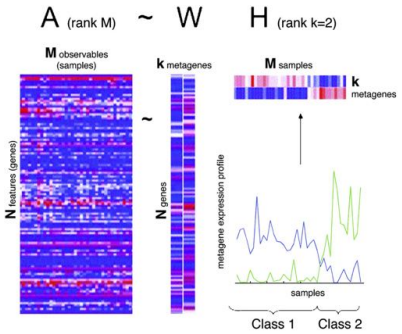
http://www.sciencedirect.com/science/article/pii/S1532046410001000

Independent component analysis

- The source matrix S is used to biologically interpret the components by studying their contributing genes
- The matrix A is used to associate the component with sample features by studying the distribution of the samples on the components according to their characteristics (e.g clinical or molecular variables).
- `MineICA` - Analysis of an ICA decomposition obtained on genomics data
  https://bioconductor.org/packages/release/bioc/html/MineICA.html

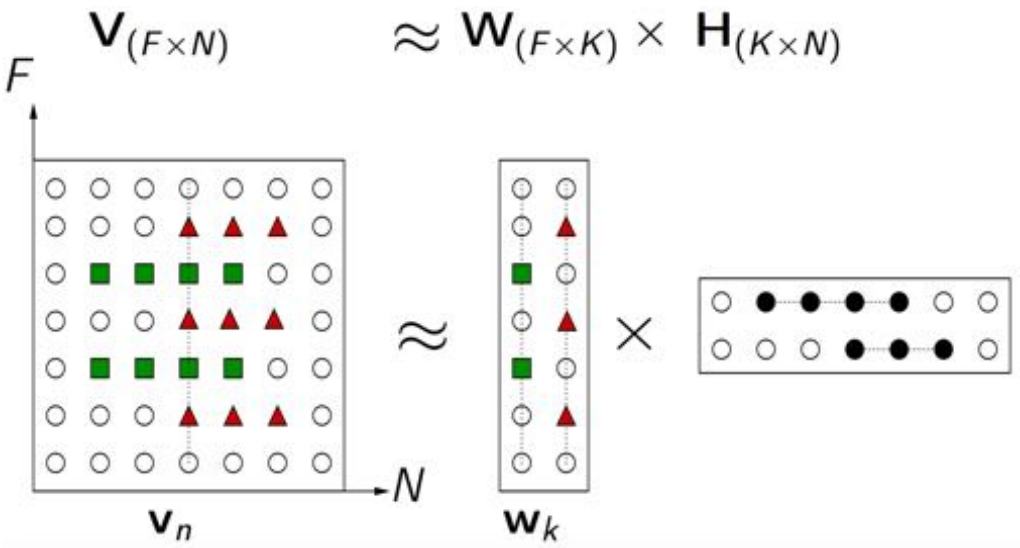
Other decomposition techniques

- Non-negative matrix factorization
- $A = WH$ (A, W, H are non-negative)
- H defined a meta-gene space: similar to eigengenes
- Classification can be done in the meta-gene space



Jean-Philippe Brunet et al. PNAS 2004;101:4164-4169

NMF, general formulation



Why nonnegativity

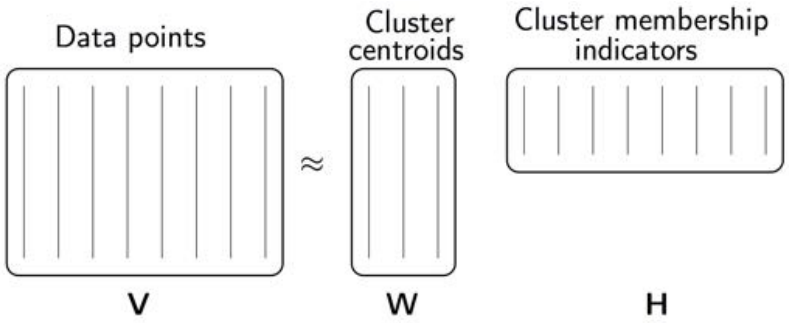
NMF is more than 30-year old!

- previous variants referred to as:
  - nonnegative rank fatorization (Jeter and Pye, 1981; Chen, 1984);
  - positive matrix factorization (Paatero and Tapper, 1994);
- popularized by Lee and Seung (1999) for "learning the parts of objects".

Since then, widely used in various research areas for diverse applications

NMF for clustering

NMF can handle overlapping clusters and provides soft cluster membership indications.



NMF

- Many computational methods
  - Cost function $|A - WH|$
  - Squared error - aka Frobenius norm
  - Kullback–Leibler divergence
- Optimization procedure
  - Most use stochastic initialization, and the results don't always converge to the same answer

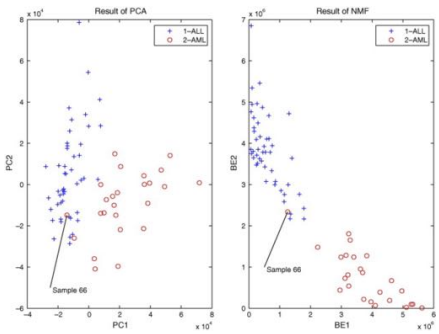
NMF

- $A = WH$: Toy Biological interpretation
- Assume $k = 2$
- We have 2 transcription factors that activate gene signatures $W1$ and $W2$
- H represents the activity of each factor in each sample
- TF effects are additive

NMF

- NMF operates in the original non-negative measurement space
- Highly expressed genes matter more
- Positivity constraint is advantageous: positive correlation among genes is more likely to be biologically meaningful
- NMF may more accurately capture the data generating process

NMF vs. PCA



- Results of PCA vs NMF for reducing the leukemia data with 72 samples in visualization. Sample 66 is mislabeled. However in 2-D display, the reduced data by NMF can clearly show this mistake while that by PCA cannot demonstrate the wrong. ‘PC’ stands for principal component and ‘BE’ means basis experiment.

Weixiang Liu, Kehong Yuan, Datian Ye “**Reducing microarray data via nonnegative matrix factorization for visualization and clustering analysis**” Journal of Biomedical Informatic 2008,

Multidimensional scaling

MDS attempts to

- Identify abstract variables which have generated the inter-object similarity measures
- Reduce the dimension of the data in a non-linear fashion
- Reproduce non-linear higher-dimensional structures on a lower-dimensional display

Kruskal's stress

$$stress = \sqrt{\frac{\sum (d_{ij} - \hat{d}_{ij})^2}{\sum d_{ij}^2}}$$

- Goodness-of-fit - Measures degree of correspondence between distances among points on the MDS map and the matrix input.
- Start with distances d_{ij}
- Fit decreasing numbers $\hat{d}_{ij}$
- Subtract, square, sum
- Take a square root
- Divide by a scaling factor

MDS Basic Algorithm

- Obtain and order the M pairs of similarities
- Try a configuration in q dimensions
  - Determine inter-item distances and reference numbers
  - Minimize Kruskal's stress
- Move the points around to obtain an improved configuration
- Repeat until minimum stress is obtained

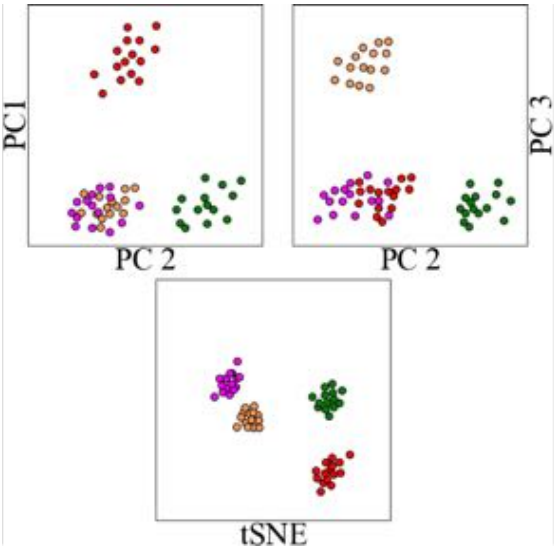
Comparison Between PCA, MDS

- **PCA** tries to preserve the covariance of the original data
- **MDS** tries to preserve the metric (ordering relations) of the original space

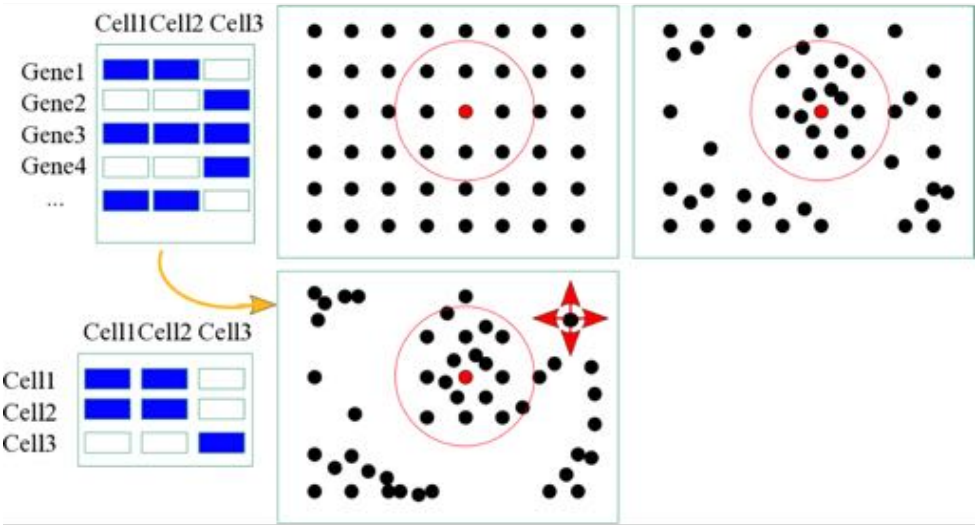
t-SNE: Nonlinear Dimensional Reduction

- Maaten, Laurens van der, and Geoffrey Hinton. “Visualizing Data Using T-SNE.” The Journal of Machine Learning Research 9, no. 2579–2605 (2008): 85.
- t-SNE, https://www.youtube.com/watch?v=EMD106bB2vY
- t-SNE tutorial https://mark-borg.github.io//blog/2016/tsne/

t-SNE: Collapsing the Visualization to 2D



t-SNE: How it works.



PCA and t-SNE Together

- Often t-SNE is performed on PCA components
- Liberal number of components.
- Removes mild signal (assumption of noise).
- Faster, on less data but, hopefully the same signal.

Learn More About t-SNE

- Awesome Blog on t-SNE parameterization: http://distill.pub/2016/misread-tsne
- Publication: https://lvdmaaten.github.io/publications/papers/JMLR_2008.pdf
- Another YouTube Video: https://www.youtube.com/watch?v=RJVL80Gg3lA
- Code: https://lvdmaaten.github.io/tsne/
- Interactive Tensor flow: http://projector.tensorflow.org/

Other approaches

- **Bi-clustering** - cluster both the genes and the experiments simultaneously to find appropriate context for clustering
  - R packages: `iBBiG`, `FABIA`, `biclust`
  - Stand-alone: `BicAT` (Biclustering Analysis Toolbox))