

# Unsupervised Learning: Dimension Reduction

# Why Dimension Reduction?

For Big-Data:

- Data visualization becomes very difficult! (Cannot draw 2D scatterplots between all pairs of features).
- Big-Data often has a high degrees of redundancy. (i.e. correlation among features).
- Many features may be uninformative for the particular problem under study (noise features).
- Dimension reduction ideally allows us retain information on most important features of the data, while reducing noise and simplifying visualization & analysis.

# What is Dimension Reduction?

- Map the data into a new low-dimensional space where important characteristics of the data are preserved.
- The new space often gives a (linear or non-linear) transformation of the original data.
- Visualization and analysis (clustering/prediction/...) is then performed in the new space.
- In many cases, (especially for non-linear transformations) interpretation becomes difficult.

# Principal Components Analysis (PCA)

# PCA

Set-up:

- Data matrix:  $X_{n \times p}$ ,  $n$  observations and  $p$  features.

Idea:

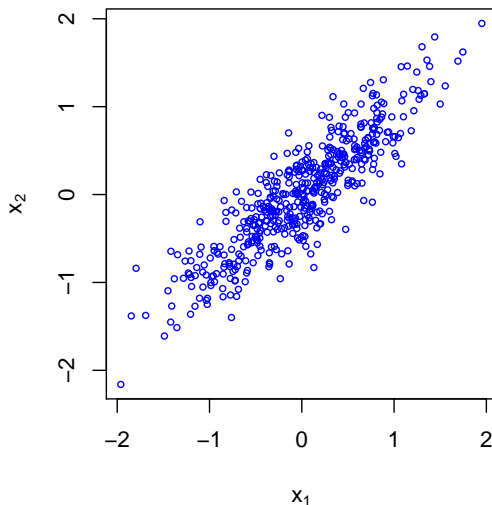
- Not all  $p$  features are needed (much redundant info).
- Find low-dimensional linear representations that capture most of the variation in the data.

Uses:

- Ubiquitously used - Dimension reduction, data visualization, pattern recognition, exploratory analysis, etc.
- “Best” linear dimension reduction possible.

# PCA - Main Idea

**Question:** What is a good 1D representation of the data?



# PCA - Main Idea

Some Possibilities:

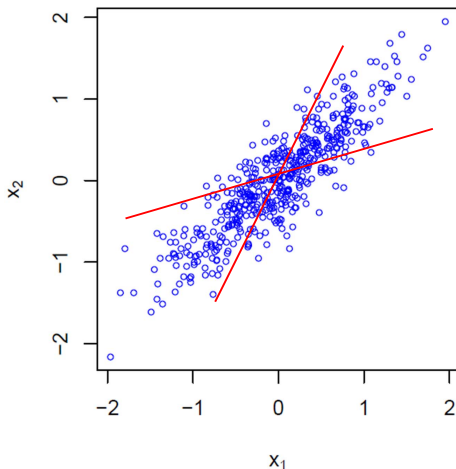
- Use one of the variables (e.g.  $x_1$ ).
- Better idea: use a linear combination of the variables (i.e. a weighted average).

$$z_1 = v_1 x_1 + v_2 x_2 = \mathbf{v}^T \mathbf{X}$$

How to choose the weights ( $v_1$  and  $v_2$ )?

# PCA - Main Idea

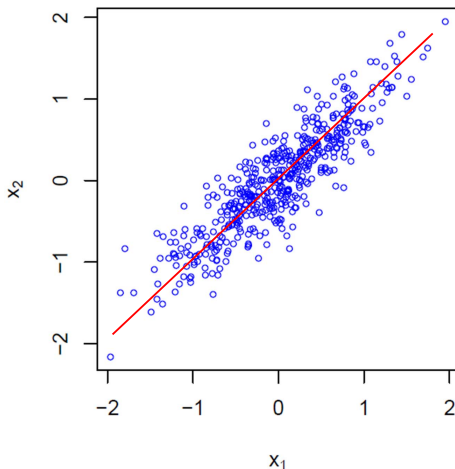
Many possibilities, but which one is a good choice?





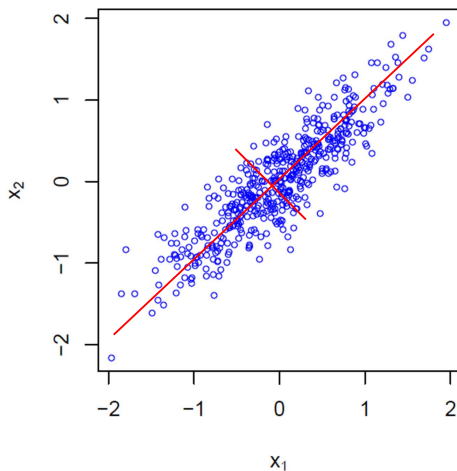
# PCA - Main Idea

Find line that maximizes the variance of the data projected onto the line:

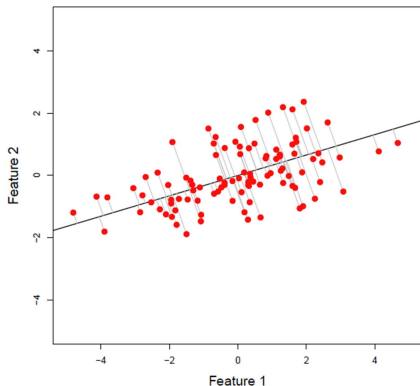


# PCA - Main Idea

Subsequent components orthogonal (perpendicular).



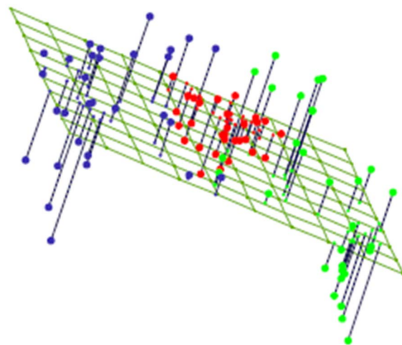
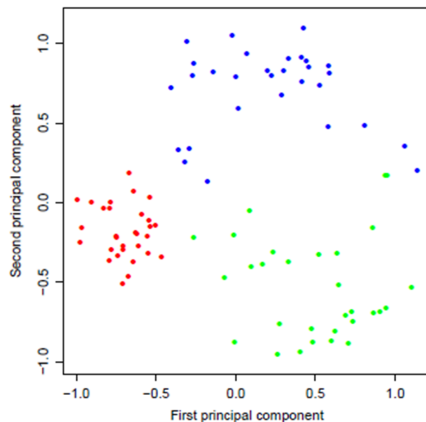
# PCA - Main Idea



- PCA minimizes orthogonal projection onto line:  $Z = v_1x_1 + v_2x_2$ .
- Slope of line =  $v_2/v_1$  (if features centered).
- Note: Not same as OLS which minimizes projection of  $y$  onto  $x$ !

# PCA - Main Idea

3D Projection onto a Hyperplane:



# PCA - Criterion

PCA Criterion - PC 1 (Population):

$$\underset{\mathbf{v}}{\text{maximize}} \quad \text{Var}(\mathbf{X}\mathbf{v}) \quad \text{subject to } \|\mathbf{v}\|_2 = 1$$

$$\underset{\mathbf{v}}{\text{maximize}} \quad \mathbf{v}^T \text{Var}(\mathbf{X}) \mathbf{v} \quad \text{subject to } \|\mathbf{v}\|_2 = 1$$

$$\underset{\mathbf{v}}{\text{maximize}} \quad \mathbf{v}^T \Sigma \mathbf{v} \quad \text{subject to } \|\mathbf{v}\|_2 = 1$$

where  $\Sigma = \text{Cov}(\mathbf{X})$ .

- Finds linear combination of features that maximizes the variance.

# PCA - Criterion

PCA Criterion - PC  $k$  (Population):

$$\underset{\mathbf{v}_k}{\text{maximize}} \quad \mathbf{v}_k^T \Sigma \mathbf{v}_k \quad \text{subject to} \quad \|\mathbf{v}_k\|_2 = 1 \text{ \& } \mathbf{v}_k^T \mathbf{v}_j = 0 \quad \forall j < k.$$

- Subsequent linear combinations are orthogonal to previous combinations.
- **Uncorrelated.**

# PCA - Criterion

PCA Criterion - Sample Version:

$$\underset{v_1, \dots, v_K}{\text{maximize}} \quad v_k^T X^T X v_k \quad \text{subject to } \|v_k\|_2 = 1 \text{ \& } v_k^T v_j = 0 \ \forall j < k.$$

Replaces  $\Sigma$  with estimate  $X^T X / n$ .

**Solution:** Eigenvalue decomposition of  $X^T X$ . (`eigen()` in R)

# PCA - Criterion

Equivalent PCA Criterion:

$$\begin{aligned} & \underset{\mathbf{u}_1, \dots, \mathbf{u}_K, \mathbf{v}_1, \dots, \mathbf{v}_K}{\text{maximize}} \quad \mathbf{u}_k^T \mathbf{X} \mathbf{v}_k \quad \text{subject to } \|\mathbf{v}_k\|_2 = 1 \ \& \ \mathbf{v}_k^T \mathbf{v}_j = 0 \ \forall j < k. \\ & \|\mathbf{u}_k\|_2 = 1 \ \& \ \mathbf{u}_k^T \mathbf{u}_j = 0 \ \forall j < k. \end{aligned}$$

- Finds left and right projection that maximize variance.

**Solution:** Singular Value Decomposition (SVD) of  $\mathbf{X}$ . (`svd()` in R)



# PCA - Parts of the Solution

$$\text{SVD: } X_{n \times p} = U_{n \times n} D_{n \times p} V_{p \times p}^T$$

- Singular vectors: (left)  $U$  and (right)  $V$ .
  - ▶ Orthonormal -  $U^T U = I$  and  $V^T V = I$ .
- Singular values: Diagonals of  $D$ .
  - ▶  $d_1 \geq d_2 \geq \dots \geq d_r$  where  $r = \text{rank}(X)$ .

SVD Solution to PCA:

- **PCs:**  $Z = XV$  or  $Z = UD$ . ( $U$  are un-scaled PCs).
  - ▶  $z_k = Xv_k$  -  $k^{th}$  PC.
  - ▶  $z_1 \dots z_K$  gives best  $K$ -dimensional projection of the data.
- **PC Loadings:**  $V$ .
  - ▶  $v_k$  -  $k^{th}$  PC loading (feature weights).

# PCA - Properties

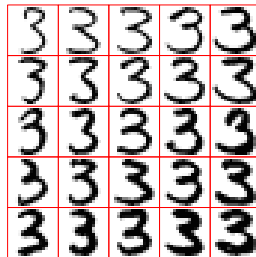
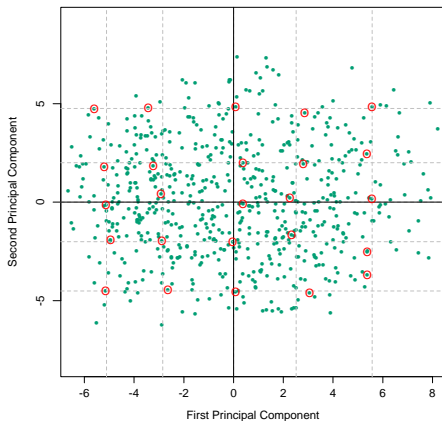
- Unique.
  - ▶  $U$  and  $V$  unique up to a sign change.
  - ▶  $D$  unique.
- Nested, Ordered components.
  - ▶  $d_1 > d_2 > \dots d_p$ .
- Orthogonal.
  - ▶  $U$  and  $V$  orthogonal.
- Global Solution.

# PCA - Pattern Recognition

- $u_1$  - first column of  $U$  encodes first major pattern in observation space.
- $v_1$  - first column of  $V$  encodes the associated first pattern in feature space.
- $d_1$  gives strength of first pattern.
- Subsequent patterns are **uncorrelated** to first pattern (i.e. orthogonal).
- $X \approx \sum_{k=1}^K d_k u_k v_k^T$  - data is comprised of a series of patterns.

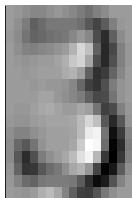
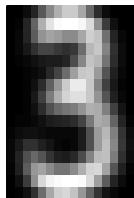
# PCA - Pattern Recognition

Patterns in observation space:



# PCA - Pattern Recognition

Patterns in feature space:



# Breakout Discussion

- What pattern does the first PC find?
- What pattern does the second PC find?
- Questions?

# PCA - Data Visualization

## PC Scatterplots:

- Problem: Can't visualize all the features.
- Solution: Plot  $u_1$  vs.  $u_2$  and so forth.
- Advantages:
  - ▶ Dramatically reduces number of 2D scatterplots to visualize.
  - ▶ Focuses on patterns with most variance.

## PC Loadings Plots:

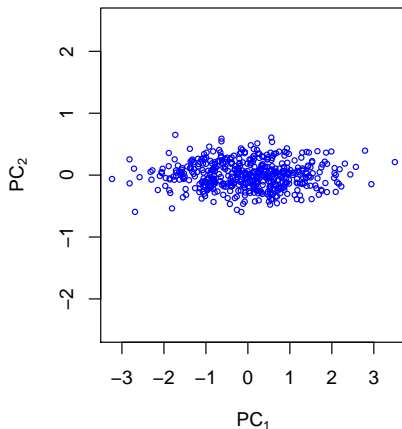
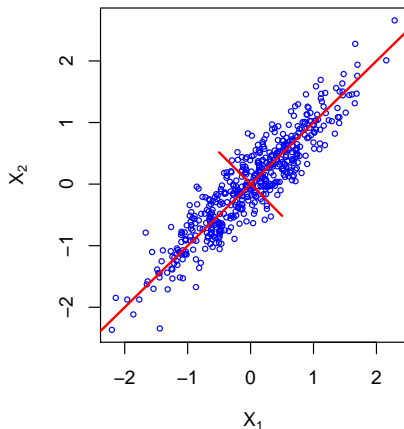
- Scatterplots of  $v_1$  vs.  $v_2$ .
- Visualizations of  $v_k$ .

## Biplot:

- Scatterplot of PC 1 vs. PC 2 with loadings of  $v_1$  vs.  $v_2$  overlaid.

# PCA - Data Visualization

## Scatterplots:



- Plotting Scatterplot PCs roughly equivalent to rotating axes of original plot.



# PCA - Dimension Reduction

Best low-rank approximation to the data:

$$\underset{\tilde{X}}{\text{minimize}} \quad ||X - \tilde{X}||_F^2 \quad \text{subject to } \text{rank}(\tilde{X}) = K$$

Solution:  $\tilde{X} = \sum_{k=1}^K d_k u_k v_k^T$  - SVD / PCA solution!

- PCA also finds best data compression to minimize reconstruction error.
- PCA yields “best” linear dimension reduction possible!

# PCA - Dimension Reduction

How much variance is explained? (i.e. extent of dimension reduction)

- Variance explained by  $k^{th}$  PC:

$$d_k^2 = \mathbf{v}_k^T \mathbf{X}^T \mathbf{X} \mathbf{v}_k.$$

- Total variance of data:

$$\sum_{k=1}^p d_k^2.$$

- Proportion of variance explained by  $k^{th}$  PC:

$$d_k^2 / \sum_{k=1}^p d_k^2.$$

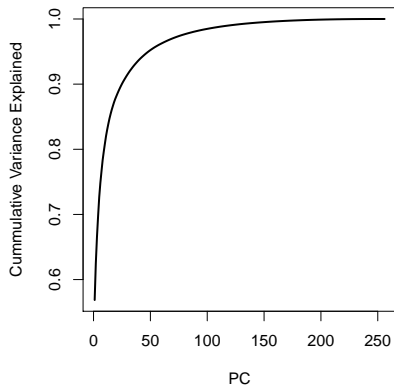
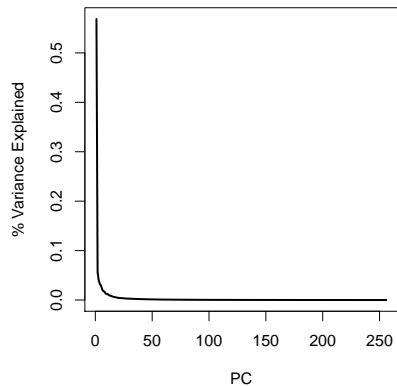
- Cumulative variance explained by first  $r$  PCs:

$$\sum_{k=1}^r d_k^2 / \sum_{k=1}^p d_k^2.$$

(Extent of dimension reduction achieved by first  $r$  PC projections.)

# PCA - Dimension Reduction

Screeplot:



# PCA - Dimension Reduction

How to choose  $K$ ?

- Elbow in screeplot.
- Take  $K$  that explains at least 90% (95%, 99%, etc.) variance.
- More sophisticated:
  - ▶ Cross-Validation done internally.
  - ▶ Validation via matrix completion.
  - ▶ Nuclear norm penalties.

# PCA - Center and Scale?

- Typically, one should center features (i.e. columns of  $X$ ).
  - ▶ Maximizing variance interpretation (assumes multivariate Gaussian model).
- Scaling changes PCA solution.
  - ▶ Features with large scale contribute more to variance, have large PC loadings.

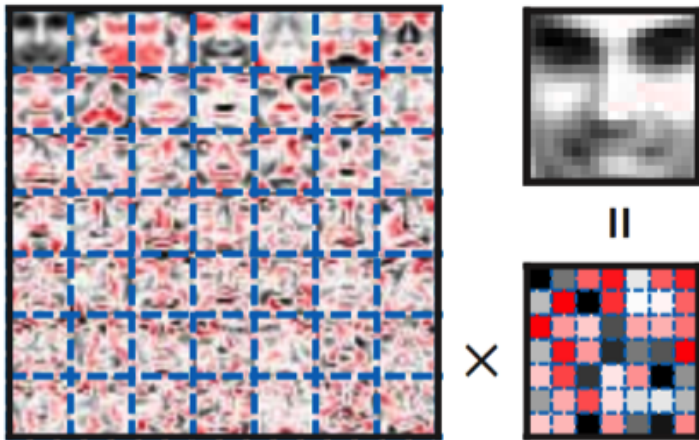
## General Suggestions:

- Scale if features measured differently. (Example - US college data).
- Don't scale if features measured in same way & scale has meaning. (Example - gene expression data).

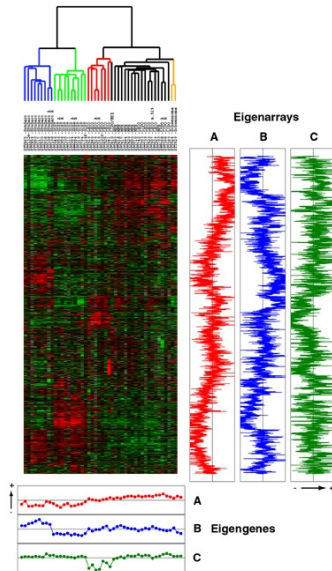
# PCA - Applications

“EigenImages” or “EigenFaces”

## PCA



# PCA - Applications



# PCA - Summary

## Strengths:

- “Best” linear dimension reduction.
- Interpretation:
  - ▶ Ordered / orthogonal components.
  - ▶ Unique, global solution.
- Data visualization.
- Pattern recognition.
- Others?

## Weaknesses:

- Non-linear patterns.
- Others?



# PCA - Best Practices

When someone gives you a data matrix ...

Apply PCA first!

# Breakout Discussion

Quick Quiz (T/F):

- You should always center and scale before applying PCA.
- PCA will perform poorly if your data doesn't lie on a line.
- You can only visualize the first two PCs.
- PCA only finds patterns amongst the observations.
- PC  $i$  always contains a more important pattern than PC  $i + 1$ .

Discussion:

- How will you use PCA for your work?
- Questions?

# PCA Extensions

# Sparse PCA

## Motivation:

- When  $p \gg n$ , many features irrelevant.
- PCA can perform poorly.

## Idea:

- Sparsity in  $V$ : zero out irrelevant features from PC loadings.
- Advantage: Find important features that contribute to major patterns in the data.

## How?

- Typically, optimize PCA criterion with sparsity-encouraging penalty of  $V$ .
- Many methods - active area of research!

In R: SPC in PMA package.

# Functional PCA

Motivation:

- Times series, ordered data, spatial data.

Idea:

- Want PC loadings to be smooth (vary continuously) over time or space.
- Advantage: Improve interpretation.

How?

- Typically, optimize PCA criterion with a penalty that encourages smoothness of  $V$  over time or space.
- Many methods for both functional data (data in the form of curves) and discretely-sampled functional data (e.g. discrete time points or specific locations).

In R: package `fpca`.

# Kernel PCA

Motivation:

- Non-linear patterns.

Idea:

- Embed inner product distances  $(x_i^T x_{i'})$  in a higher-dimensional “kernel” space,  $k(x_i, x_{i'})$ .
- Kernel examples:
  - ▶ Radial:  $k(x_i, x_{i'}) = e^{-\|x_i - x_{i'}\|_2^2 / 2\sigma^2}$ .
  - ▶ Polynomial:  $k(x_i, x_{i'}) = (cx_i^T x_{i'} + 1)^d$ .
- Kernel Matrix:  $K_{n \times n} : K_{ii'} = k(x_i, x_{i'})$ .
  - ▶ Idea:  $K$  a non-linear distance matrix.
- Find major non-linear patterns by performing PCA on  $K$ :

$$K = U D^2 U^T$$

# Supervised Dimension Reduction

## Partial Least Squares:

- Best dimension reduction of cross-covariance between  $X$  and  $Y$  such that factors are orthogonal to  $X$ .

## Canonical Correlations Analysis:

- Best dimension reduction of cross-covariance between  $X$  and  $Y$  such that bi-projection is orthogonal to  $X$  or  $Y$ .

## Linear Discriminant Analysis (classification):

- Best dimension reduction of between class covariance matrix relative to within class covariance.

# Non-Negative Matrix Factorization (NMF)

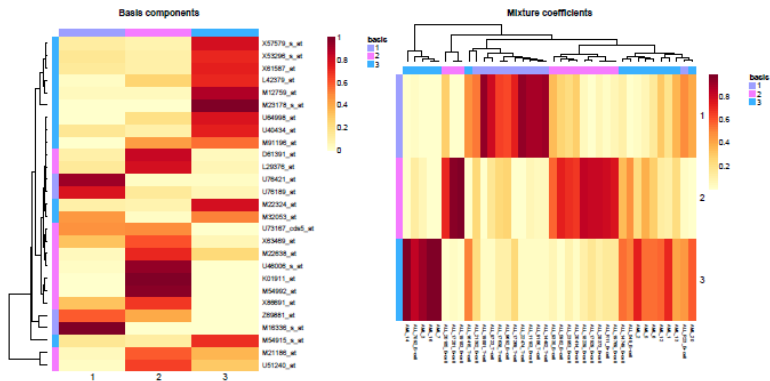


# NMF

Idea:  $X_{n \times p} \approx W_{n \times K} H_{K \times p} = \sum_{k=1}^K W_{:,k} H_{k,:}$  with  $K \ll p$ .

- $X \geq 0$  - non-negative data matrix.
- $W \geq 0$  - non-negative observation factors; often sparse (Basis Factors).
  - ▶  $W_{:,k} \geq 0$  -  $k^{th}$  observation factor.
- $H_{kj} \geq 0$  - non-negative feature factors; often sparse (Mixture Factors).
  - ▶  $H_{k,:} \geq 0$  - mixture of features that comprise the  $k^{th}$  factor.

Like PCA except finds patterns with same direction of correlation.



# NMF Interpretation

## Topic Modeling:

- $X$  a matrix of news articles (rows) by words (columns) whose entries are word counts.
  - ▶  $X \approx \sum_{k=1}^K W_{:,k} H_{k,:}$  - sum of topics.
  - ▶  $X_{ij} = W_{i,:}^T H_{:,j} = \sum_{k=1}^K W_{ik} H_{kj}$ .
- Topic  $k$ : Outer-product of  $k^{th}$  column of  $W$  ( $W_{:,k}$ ) and  $k^{th}$  row of  $H$  ( $H_{k,:}$ ).
  - ▶ E.g. Covid-19.
- $H_{k,:}$  non-zeros- words contributing to topic  $k$ .
  - ▶ E.g. virus, hospitalizations, cases, deaths, masks, testing etc.
- $W_{:,k}$  non-zeros - news articles belonging to topic  $k$ .
  - ▶ E.g. "Virus surge visible across Texas: The tsunami is here" (*Washington Post*).

# NMF Criterion - Continuous Data

$$\begin{aligned} & \underset{W, H}{\text{minimize}} \quad ||X - WH||_F^2 \\ & \text{subject to} \quad W_{ik} \geq 0 \ \& \ H_{kj} \geq 0 \end{aligned}$$

(PCA criterion except with non-negativity constraints.)

Algorithm Updates: (Alternating Non-negative Least Squares)

$$\begin{aligned} \hat{W} &= \left( XH^T (H^T H)^{-1} \right)_+ \\ \hat{H} &= \left( (W^T W)^{-1} W^T X \right)_+ \end{aligned}$$

Local Solution.

# NMF Criterion - Count Data

$$\begin{aligned} & \underset{W, H}{\text{minimize}} && \sum_{i=1}^n \sum_{j=1}^p [X_{ij} \log(W_i H_j) - W_i H_j] \\ & \text{subject to} && W_{ik} \geq 0 \ \& \ H_{kj} \geq 0 \end{aligned}$$

Algorithm Updates:

$$\begin{aligned} \hat{W}_{ik} &= \hat{W}_{ik} \left( \frac{\sum_{j=1}^p \hat{H}_{kj} X_{ij} / \hat{W}_i^T \hat{H}_j}{\sum_{j=1}^p \hat{H}_{kj}} \right) \\ \hat{H}_{kj} &= \hat{H}_{kj} \left( \frac{\sum_{i=1}^n \hat{W}_{ik} X_{ij} / \hat{W}_i^T \hat{H}_j}{\sum_{i=1}^n \hat{W}_{ik}} \right) \end{aligned}$$

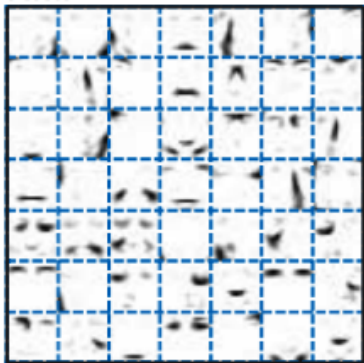
Local solution.

# NMF - Uses

- ① Dimension Reduction / Pattern Recognition.
  - ▶ Similar to PCA (e.g. component scatterplots) except that patterns of correlation found in the same direction.
- ② Archetypal Analysis.
  - ▶ Caricatures (segments; contrastive categorization) vs. Prototypes (averages).
- ③ Soft-clustering.
  - ▶ Discussed Next Lecture!

# NMF - Archetypal Analysis

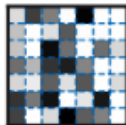
NMF



Original



$\times$

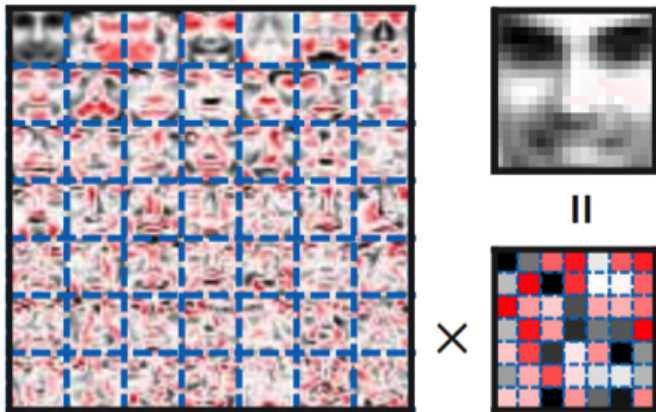


$=$



# NMF - Archetypal Analysis

PCA





# PCA vs. NMF

## Similarities:

- Linear Dimension Reduction.
- Interpretation.

## Differences:

- Factors are unordered.
- Factors NOT orthogonal.
- Changing  $K$  can fundamentally change factors.
- Non-unique, non-global solution.
- Depends on initialization. (Run several times and take the best).

# Choosing $K$

Choice depends on goal:

- Dimension Reduction:
  - ▶ Residual sums of squares (or dispersion) - Screeplot.
- Clustering:
  - ▶ Consensus, silhouette, etc. (Discussed next lecture!).
- Archetypal Analysis:
  - ▶ Sparsity, factor purity, etc.

# NMF - Summary

## Strengths:

- Interpretation (often more appealing than PCA!).
- Applications - Clustering & Archetypal Analysis.
- Pattern Recognition.
- Others?

## Weaknesses:

- Local solutions that depend strongly on  $K$ .
- Others?

In R: NMF package.

# Independent Components Analysis (ICA)

# ICA

Pre-processing Step: Reduce  $X_{n \times p}$  to  $\tilde{X}_{K \times p}$  with  $K < n$  # independent sources. (Typically via PCA!)

Idea:  $\tilde{X}_{K \times p} \approx A_{K \times K} S_{K \times p}$ .

- Assumption:  $\tilde{X}$  a matrix of  $K$  scrambled independent signals.
- $A_{K \times K}$  *Mixing Matrix* - denotes how signals are scrambled to form sources in data.
- $S_{K \times p}$  *Signal Matrix* - each row of  $S$  is an independent signal.

PCA finds uncorrelated, but not independent signals.

# ICA Uses

## ① Blind Source Separation.

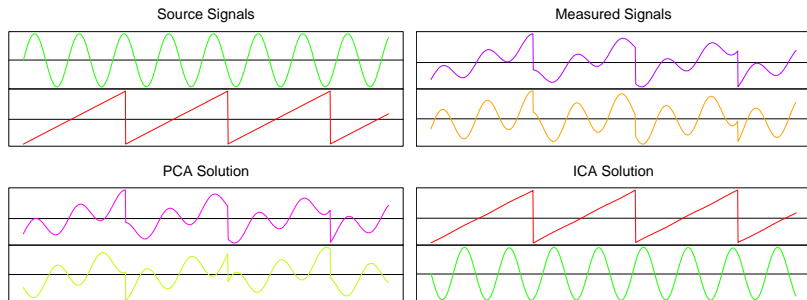
- ▶ Assume  $K$  independent signals got scrambled, but record  $K$  scrambled versions of the signal.
- ▶ Cocktail Party Problem.

## ② Denoising.

- ▶ Noise - independent from true signals.

# ICA vs. PCA

## Blind Source Separation:



# ICA Algorithms

## Fast ICA:

- Finds projections of  $X$  that are “non-Gaussian”.
- Uses non-Gaussian contrast functions:
  - ▶  $g(x) = x^4$ .
  - ▶  $g(x) = \tanh(x)$ .
- Generalization of projection pursuit.

## Others:

- Infomax (entropy).

Not Statistically Independent!



# PCA vs. ICA

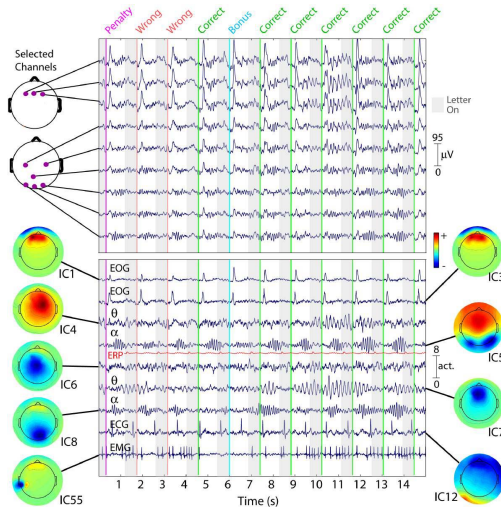
## Similarities:

- Linear Dimension Reduction.
- Interpretation.

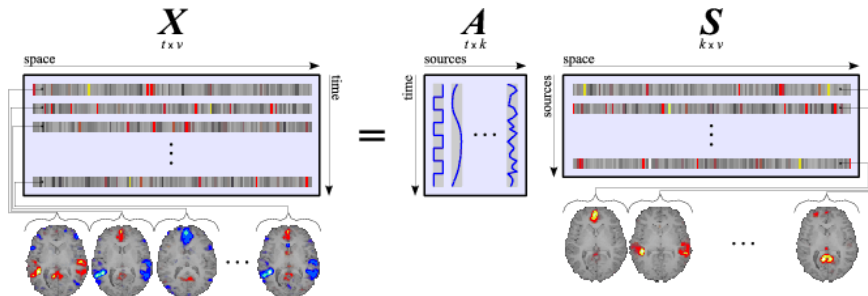
## Differences:

- Factors are unordered.
- Factors NOT invariant - same solution by applying a permutation.
- Factors NOT orthogonal.
- Changing  $K$  can fundamentally change factors.
- Non-unique.
- No optimization criterion to evaluate solution.

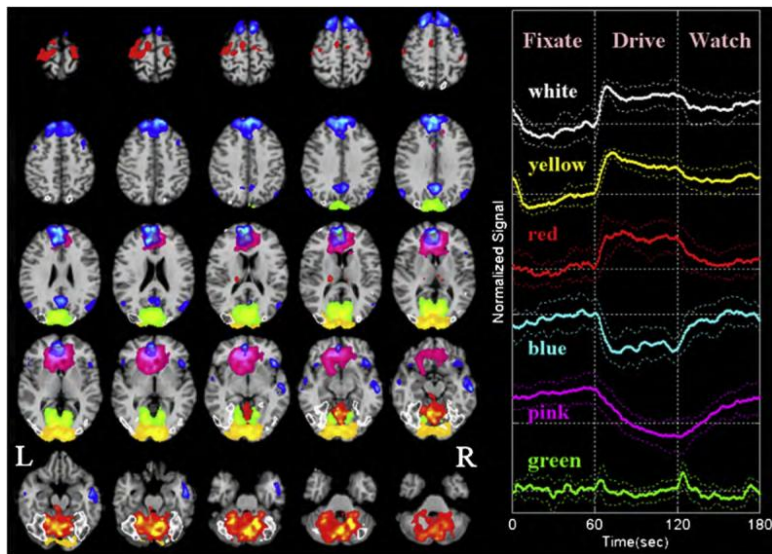
# ICA Applications - EEG



# ICA Applications - fMRI



# ICA Applications - fMRI



# ICA Summary

## Strengths:

- Interpretation.
- Applications - Blind Source Separation & Denoising.
- Others?

## Weaknesses:

- Solutions that depend strongly on  $K$ .
- Solutions can be rotated.
- Others?

In R: `fastICA` package.

# Multidimensional Scaling (MDS)

# Multidimensional Scaling (MDS)

Idea:

- Visually represent proximities (similarities or distances) between objects in a lower dimensional space.
- Input: Matrix of similarities or dissimilarities,  $D_{n \times n}$  (don't need the data itself!).
- Goal: Find projections ( $z_1, \dots, z_K$  where  $z \in \mathbb{R}^n$ ) that preserve original distances in  $D$  in a lower dimensional space ( $K \ll n$ ).
- 2 Types: Classical (Metric) MDS and Non-metric MDS.
- Non-linear dimension reduction.

# MDS - Example

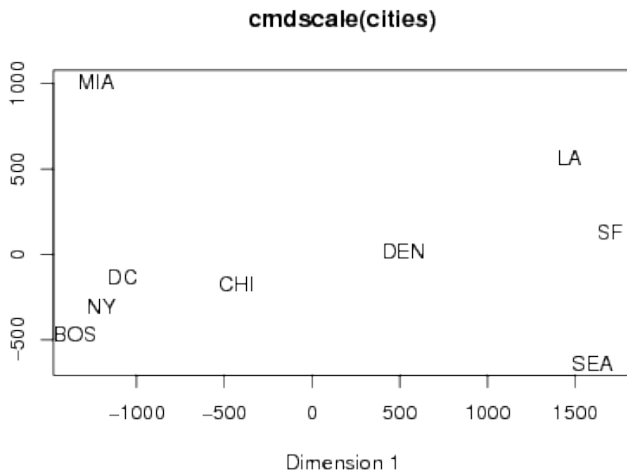
Consider the distances between nine American cities:

	BOS	CHI	DC	DEN	LA	MIA	NY	SEA	SF
BOS	0	963	429	1949	2979	1504	206	2976	3095
CHI	963	0	671	996	2054	1329	802	2013	2142
DC	429	671	0	1616	2631	1075	233	2684	2799
DEN	1949	996	1616	0	1059	2037	1771	1307	1235
LA	2979	2054	2631	1059	0	2687	2786	1131	379
MIA	1504	1329	1075	2037	2687	0	1308	3273	3053
NY	206	802	233	1771	2786	1308	0	2815	2934
SEA	2976	2013	2684	1307	1131	3273	2815	0	808
SF	3095	2142	2799	1235	379	3053	2934	808	0

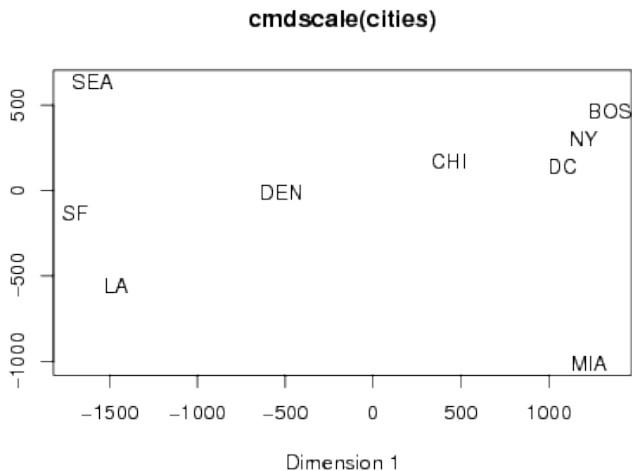
Can we represent these cities in a 2D space like a map?



# MDS - Example



# MDS - Example



# MDS - Example



# Classical (Metric) MDS

- Idea: Perform PCA (eigenvalue decomposition) on the doubly centered distance matrix,  $D$ .
  - ▶  $H = I - \frac{1}{n}11^T$  is the centering matrix.
  - ▶  $z_1, \dots, z_K$  are the top  $K$  PCs of  $H D H$ .
- Fact: When  $D$  is the matrix of Euclidean distances, classical MDS is equivalent to PCA.
  - ▶ Differences for other distance metrics.

In R: `cmdscale`.

# Non-Metric MDS

- Idea: Optimize stress function that keeps distances in  $Z$  close to that of  $D$ .

Stress Functions:

- Least squares or Kruskal-Shephard Scaling:

$$S_D(z_1, z_2, \dots, z_K) = \sqrt{\sum_{i \neq i'} (d_{ii'} - \|z_i - z_{i'}\|)^2}.$$

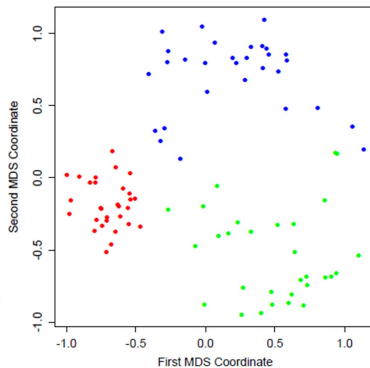
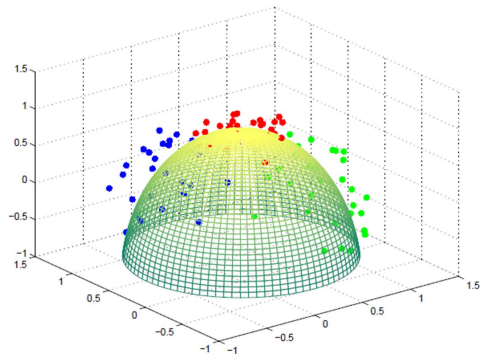
- Sammon mapping: preserve smaller pairwise distances

$$\sum_{i \neq i'} \frac{(d_{ii'} - \|z_i - z_{i'}\|)^2}{d_{ii'}}.$$

- Shepard-Kruskal nonmetric scaling ( $\theta(\cdot)$ : an increasing function):

$$\frac{\sum_{ii'} [\theta(\|z_i - z_{i'}\|) - d_{ii'}]^2}{\sum_{ii'} d_{ii'}^2}.$$

# MDS - Example



# MDS Properties

- Data not needed - only dissimilarities.
- Choosing  $K$ :
  - ▶ Scree plot (like PCA).
  - ▶ Shepard Diagram - plot proximities against distances in  $Z$ .
- Interpreting MDS maps:
  - ▶ Axes and orientation arbitrary.
  - ▶ Can be rotated.
  - ▶ Only relative locations important.
  - ▶ Typically looks for objects close in the MDS map.
  - ▶ Can be distortions.

# MDS vs. PCA

## Similarities:

- Dimension reduction for visualization.

## Differences:

- Non-linear vs. Linear.
- Local solution & arbitrary map.
- Non-unique & local solution.
- Only yields visualization / patterns among  $n$  objects.



# MDS - Summary

## Strengths:

- Visualizing proximities.
- Only need dissimilarities.
- Non-linear dimension reduction.
- Others?

## Weaknesses:

- Arbitrary maps.
- High-dimensional settings? ( $p \gg n$  - more features than objects)
- Others?

In R: `dist`; `cmdscale` - classical MDS; `isoMDS` - Kruskals's MDS and `sammon` in MASS package.

# Neighbor Embeddings: tSNE & UMAP

# Neighbor Embeddings

Idea:

- Find lower dimensional embedding that preserves relationships between “close neighbors”.
- Non-linear dimension reduction (manifold learning).
- Useful for visualizations and clustering.
- Works well for high-dimensional data ( $p \gg n$ ).

# t-Stochastic Neighbor Embedding (tSNE)

- Set up: Represent data,  $x_1, \dots, x_n \in \mathbb{R}^p$ , in a lower dimensional space,  $z_1, \dots, z_n \in \mathbb{R}^k$  where  $k \ll p$  (usually  $k = 2$ ).
- Data Similarity (normalized Gaussian kernel;  $\sigma$  related to perplexity):

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2}, \quad p_{j|i} = \frac{\exp(-(x_i - x_j)^2 / 2\sigma^2)}{\sum_{k \neq i} \exp(-(x_i - x_k)^2 / 2\sigma^2)}$$

- Lower-Dimensional Similarity (normalized t/Cauchy kernel):

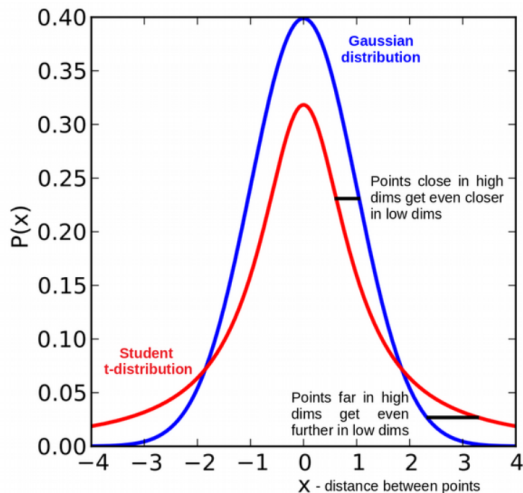
$$q(z_i, z_j) = \frac{g(|z_i - z_j|)}{\sum_{k \neq i} g(|z_i - z_k|)}, \quad g(z) = \frac{1}{1 + z^2}.$$

- Minimize Kullback-Leibler divergence:

$$\underset{z_1, z_2, \dots, z_n}{\text{minimize}} \quad \sum_{i,j} p_{ij} \log \left( \frac{p_{ij}}{q(z_i, z_j)} \right)$$

- Gradient descent (with random initialization).

# t-Stochastic Neighbor Embedding (tSNE)



# UMAP

Similar to tSNE, but with a few differences:

- Data Similarity (Gaussian kernel, general distances, K-Nearest Neighbors, no normalization):

$$p_{ij} = p_{i|j} + p_{j|i} - p_{i|j}p_{j|i}, \quad p_{i|j} = \exp(-d(x_i, x_j)/\sigma_i).$$

- Lower-Dimensional Similarity (generalized Cauchy kernel):

$$q(z_i, z_j) = \frac{1}{1 + a(z_i - z_j)^{2b}}, \quad a > 1, b < 1 \text{ (thicker tails)}.$$

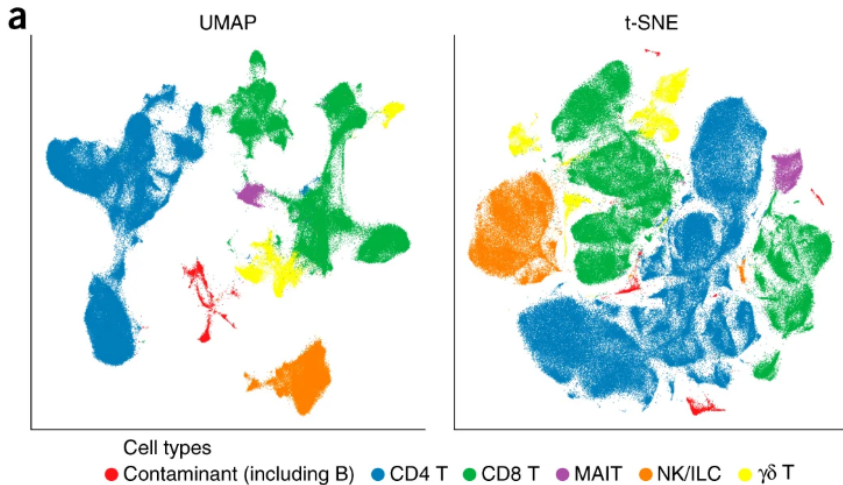
- Minimize cross-entropy:

$$\underset{z_1, z_2, \dots, z_n}{\text{minimize}} \quad \sum_{i,j} p_{ij} \log \left( \frac{p_{ij}}{q(z_i, z_j)} \right) + (1 - p_{ij}) \log \left( \frac{1 - p_{ij}}{1 - q(z_i, z_j)} \right)$$

- Stochastic gradient descent with spectral clustering initialization.

Overall: UMAP better preserves global structures & more tightly packs close neighbors.

# tSNE & UMAP - Applications

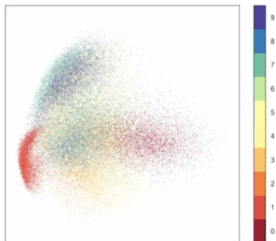


Clustering Cell Types in Single Cell RNA-seq.

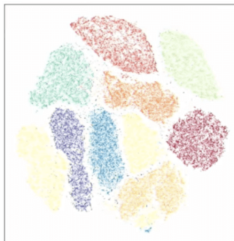
# Global vs. Local Interpretation

MNIST Digits

PCA



t-SNE



UMAP





# tSNE & UMAP - Summary

## Strengths:

- Data visualization.
- Clustering.
- Non-linear dimension reduction.
- Preserves relationships between close neighbors.

## Weaknesses:

- Local, non-unique solutions.
- Patterns only amongst observations.
- Sensitive to hyperparameters.
- Can have distortions.
- Others?

# Dimension Reduction & Visualization Wrap-Up

Techniques Covered:

- PCA.
- NMF.
- ICA.
- MDS.
- SNE - tSNE & UMAP.

# Breakout Discussion

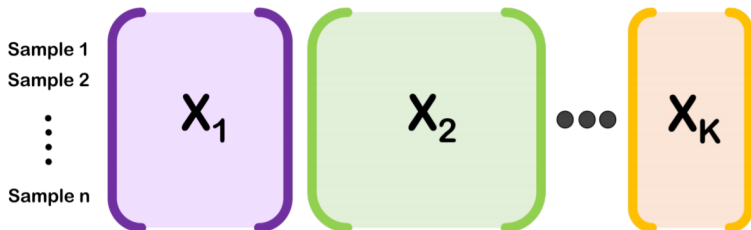
## Questions:

- Which techniques are useful for finding patterns amongst observations AND features? Just observations?
- List some advantages of PCA over MDS / tSNE / UMAP. List some disadvantages.
- When would you want use each of the techniques?
- Why do we need so many dimension reduction and visualization techniques?
- If you had new (test) data, how would you use this to validate the patterns you discovered?

# Research Highlight

## Integrated PCA

### Multi-View Data:



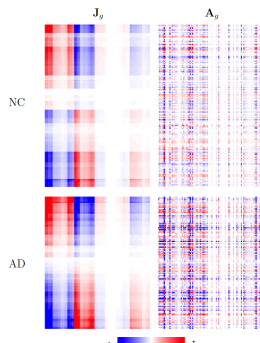
**Goal:** How do you find joint patterns across the shared subjects?

**Solution:** iPCA! (Tang and Allen, 2021)

- Generalization of PCA using matrix-variate normal.

# Research Highlight

## Multi-Group PCA and beyond



**Solution:** Joint and Individual Component Regression (JICO) (Wang et al., JCGS, to appear)

- Generalization of PCA for multi-group data using continuum regression framework; covers JIVE (Lock et al., 13) as a special case.