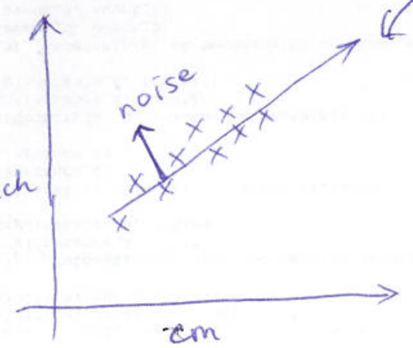
Principal Component Analysis.

Given $\{x^{(1)} \dots x^{(m)}\}$ $\quad x^{(i)} \in \mathbb{R}^n$

Goal: reduce to lower dim. data $(k \ll n)$



what you care about is this axis. → principal axis along which the data varies.

Reduce from 2D to 1D → remove noise

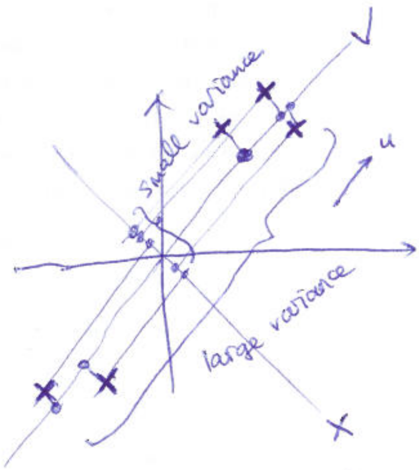
Pre-processing:

Set $\mu = \frac{1}{m} \sum_{i=1}^{m} x^{(i)}$.
Replace $x^{(i)}$ with $x^{(i)} - \mu$. } zero-out the mean.

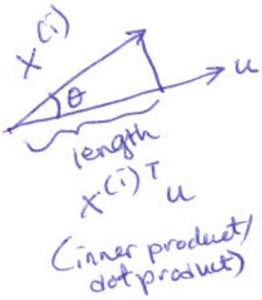
Set $\sigma_j^2 = \frac{1}{m} \sum_{i=1}^{m} \left(x_j^{(i)}\right)^2$ → variance of each feature

Replace $x_j^{(i)}$ with $\frac{x_j^{(i)}}{\sigma_j}$ } Normalize to unit variance → each of the feature has now equal variance.

} when features are in diff scales.



← find direction u, st when I project the data to it; the projection varies widely.



If $\|u\| = 1$, $x^{(i)}$ projected on u has length $x^{(i)T} u$

Choose u to maximize:

$$\max_{u:\|u\|=1} \frac{1}{m} \sum_{i=1}^{m} \left(x^{(i)T} u\right)^2$$

$$= \frac{1}{m} \sum_{i=1}^{m} \left(u^T x^{(i)}\right)\left(x^{(i)T} u\right) = u^T \left[\frac{1}{m} \sum_{i=1}^{m} x^{(i)} x^{(i)T}\right] u$$

②

$\Rightarrow$ u is the principal eigenvector of Σ = cov. matrix $= \frac{1}{m} \sum_{i=1}^{m} x^{(i)} x^{(i)T}$

largest eigenvalue

Eigenvector.

$Au = \lambda u$

λ eigenvalue, u eigenvector

erivation.

$$\max \; u^T \Sigma u \quad \text{st} \quad u^T u = 1$$

$$\mathcal{L}(u, \lambda) = u^T \Sigma u - \lambda (u^T u - 1)$$

lagrange

$$\frac{\partial \mathcal{L}(u,\lambda)}{\partial u} = \Sigma u - \lambda u \overset{\text{set}}{=} 0$$

$\Sigma = n \times n$
$u = n \times 1$
$x^{(i)} = n \times 1$

$$\Sigma u = \lambda u.$$

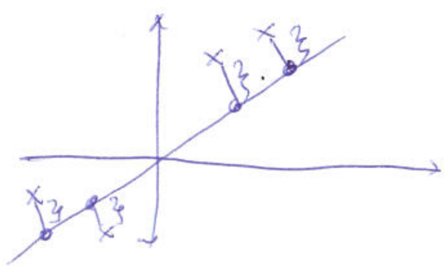
u = principal eigenvector of Σ.

If we want k-dim subspace,
Choose $u_1 \ldots u_k$ to be k top eigenvector of Σ
$\hookrightarrow$ corresp. to k highest eigenvalues.

$u_1 \ldots u_k$: new basis of representing data
at beginning we have $x^{(1)} \ldots x^{(m)} \in \mathbb{R}^n$
New representation:

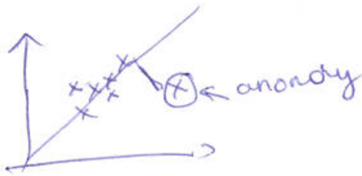
$$y^{(i)} = (u_1^T x^{(i)}, u_2^T x^{(i)}, \ldots u_k^T x^{(i)}) \cdot \quad y^{(i)} \in \mathbb{R}^k.$$

another view of PCA.
minimize sum of squared distance btwn the pt of its projection.



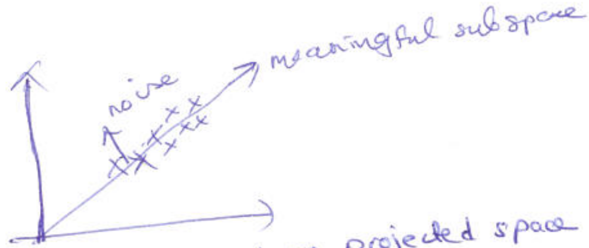
Use of PCA

- visualization of high-dim data.
- compression. → ~~feature selection~~ work with lower dim data instead of high-dim data
- learning → more features → more complex → overfitting
  use PCA to reduce dimensionality of features
- Anomaly detection → pts far away from your subspace

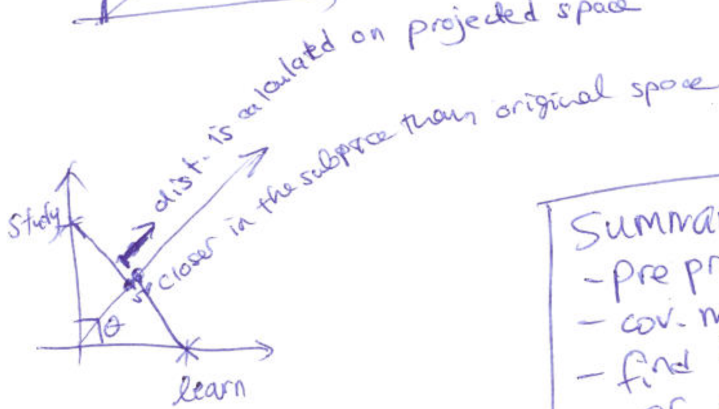


- matching/ Distance calculation.
  datapts often lie in a subspace — the rest of the space
  low-dim maybe noise

where data "lives"



In orig space similarity = $\frac{x^{(i)T} x^{(j)}}{\|x^{(i)}\| \|x^{(j)}\|} = \cos\theta = 0$.
= no words in common.



Summary PCA
- preprocess Σ
- cov. matrix computation
- find top k eigenvectors of Σ

Latent Semantic Indexing LSI.
- does not have the preprocessing step

Problem of PCA
- Covariance matrix computation
  when your data
  100 × 100 pixel image ; $x^{(i)} \in \mathbb{R}^{10,000}$
  Σ 10,000 × 10,000 = 100 mill. entries

Implement PCA using SVD — singular value decomposition

Say we have any $A \in \mathbb{R}^{m\times n}$

decompose $A = U D V^T$

$$\underset{m\times n}{A} = \underset{m\times n}{U}\ \underset{n\times n}{D}\ \underset{n\times n}{V^T}$$

$$D = \text{diagonal} = \begin{bmatrix} \sigma_1 & & & 0 \\ & \sigma_2 & & \\ & & \ddots & \\ 0 & & & \sigma_n \end{bmatrix}$$

σ_i = singular values of matrix A

$$\begin{bmatrix} A = \\ m\times n \end{bmatrix} = \begin{bmatrix} U \\ m\times n \end{bmatrix} \begin{bmatrix} D \\ n\times n \end{bmatrix} \begin{bmatrix} V^T \\ n\times n \end{bmatrix}$$

(using svd command in matlab).

svd = $O(n^3)$ not sure?

U's column: eigenvector of AA^T

V's columns: eigenvector of A^TA

Can be used to compute eigenvector of PCA efficiently.

$$\Sigma = \frac{1}{m}\sum_{i=1}^{m} x^{(i)} x^{(i)T}$$

given Design matrix

$$X = \begin{bmatrix} - x^{(1)} - \\ - x^{(2)} - \\ \vdots \\ - x^{(m)} - \end{bmatrix}$$

$$\Sigma = X^T X = \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ | & | & & | \end{bmatrix} \begin{bmatrix} - x^{(1)} - \\ - x^{(2)} - \\ \vdots \\ - x^{(m)} - \end{bmatrix}$$

To get top k eigenvector of Σ,

$X = U D V^T$

top k columns of V are the top k eigenvectors of $X^T X = \Sigma$.

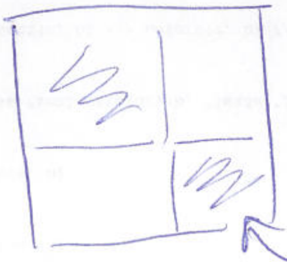
Spectral clustering.

data

affinity matrix

$$W_{nm} = e^{-\frac{\|s_n - s_m\|}{\sigma^2}}$$

$\Rightarrow$



disconnected graph

$\Rightarrow A = \text{func}(W) \Rightarrow$

Smallest ~~top~~ K eigenvectors of A

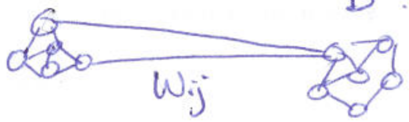
reduce dim of data to $m \times k$

$\Downarrow$

Clustering w K-means

$\Downarrow$

map back to orig. data



$d_i = \sum_j w_{ij}$ = degree of a vertex.

$D = \text{diag}(d_1, d_2, \dots d_n)$ degree matrix

$|A|$ = # of vertices in A

$\text{vol}(A) = \sum_{i \in A} d_i$

$$\text{cut}(A,B) := \sum_{i \in A, j \in B} w_{ij}$$

Balanced min-cut $\quad \min \text{cut}(A,B)$ st $|A| = |B|$

Ratio cut $\quad \text{cut}(A,B)\left(\frac{1}{|A|} + \frac{1}{|B|}\right)$.

Normalized cut $\quad \text{cut}(A,B)\left(\frac{1}{\text{vol}(A)} + \frac{1}{\text{vol}(B)}\right)$.

} NP-hard use spectral clustering to approx

$$\min \text{Cut}(A,B) = \min \frac{1}{2} f^T(D-W)f \qquad f_i = \begin{cases} 1 & \text{if } x_i \in A \\ -1 & \text{if } x_i \in B \end{cases}$$

$L = D - W$ un-normalized Graph Laplacian

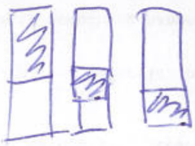
$$\min_{f \in \mathbb{R}^n} f^T L f \text{ st. } f^T 1 = 0 \quad f^T f = n$$

$$= \left.\min_{f \in \mathbb{R}^n} \frac{f^T L f}{f^T f}\right\} \text{ st } f^T 1 = 0. \Rightarrow \lambda = \text{smallest eigenvalue of } L$$

for connected graph; first eigenvector is constant (all 1s)

dis connected graph ., Laplacian is block diagonal

& first k Laplacian eigen vectors are



Another paper has used $P = D^{-1} W$ ("normalized" affinity matrix) instead of L; and take the eigenvector corresponding to the <u>largest</u> eigenvalue instead of the <u>smallest</u>. The two algorithms, the one using L matrix and the one using P matrix can be shown to be similar:

$$\underbrace{(D - W)}_{L} r = \mu D r$$

$$D^{-1}(D-W) r = \mu r$$

$$\underbrace{D^{-1}(W)}_{P} r = (1-\mu) r$$

$$\lambda = 1-\mu$$

$$v = r.$$

equality between using L and P matrix: i.e.

$L v = \lambda v \rightarrow$ take smallest eigenval

$P v = (1-\mu) v \rightarrow$ take largest eigenval