PCA Principal Components Analysis

data dependent dimensionality reduction

data set $x_1, \ldots, x_n \in \mathbb{R}^d$

Credit for figures:
Roughgarden & Valiant
John Benedetto
Novembre et al
Alex Williams
Sandipan Dey
Victor Lavrenko

Useful for

- visualization
- interpretation
- compression
- finding intrinsic dimensionality

Example: $n = 4$ data pts
$d = 4$ dimensions

		kale	taco bell	sashimi	pop tarts
x_A	Alice	10	1	2	7
x_B	Bob	7	2	1	10
x_C	Carolyn	2	9	7	3
x_D	Dave	3	6	10	2
$\bar{x}$		5.5	4.5	5	5.5

Claim:
each row can be expressed ≈ as $x_i = \vec{x} + a_{i1}\vec{v}_1 + a_{i2}\vec{v}_2$

$\vec{x} = (5.5, 4.5, 5, 5.5)$

$+ \; 1 \cdot \vec{v}_1 = (3, -3, -3, 3)$ ← notice. $(v_1 \cdot v_2 = 0)$

$+ \; 1 \cdot \vec{v}_2 = (1, -1, 1, -1)$

$(9.5, 0.5, 3, 7.5)$

		kale	taco bell	sashimi	pop tarts
x_A	Alice	10	1	2	7
x_B	Bob	7	2	1	10
x_C	Carolyn	2	9	7	3
x_D	Dave	3	6	10	2
	$\vec{x}$	5.5	4.5	5	5.5

Claim!

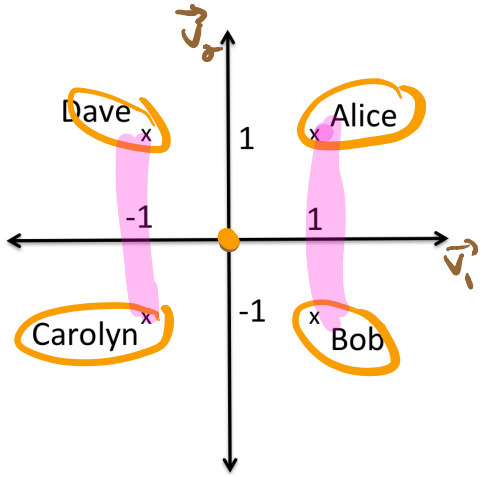
$x_A \approx \vec{x} + 1 \cdot \vec{v}_1 + 1 \cdot \vec{v}_2$

$x_B \approx \vec{x} + 1 \cdot \vec{v}_1 + (-1) \cdot \vec{v}_2$

$x_C \approx \vec{x} + (-1) \cdot \vec{v}_1 + (-1) \cdot \vec{v}_2$

$x_D \approx \vec{x} + (-1) \cdot \vec{v}_1 + 1 \cdot \vec{v}_2$

	(a_{i1}, a_{i2})
Alice	1, 1
Bob	1, -1
Carol	-1, -1
Dave	-1, 1



Why bother?

① visualization

② helps interpret data

each pt has "$\vec{v}_1$ coordinate"

"$\vec{v}_2$ coordinate"

Goal: express n d-dimensional vectors $\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$ as linear combinations of k vectors $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$

so $\vec{x}_i \approx \sum_{j=1}^{k} a_{ij} \vec{v}_j$

(assuming $\sum_{i=1}^{n} \vec{x}_i = 0$)

Preprocessing:

① Subtract mean

② Scale coordinates so that variance is 1 in each coords

[to avoid sensitivity to units]

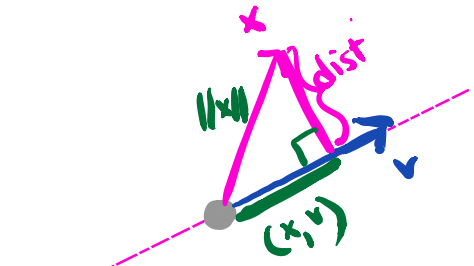
Henceforth we will assume $\sum_{i=1}^{n} x_i = 0$

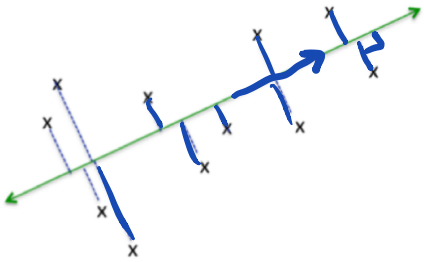
k=1

choose $\vec{v} \in \mathbb{R}^d$ with $\|v\|=1$

to minimize

$$\frac{1}{n}\sum_{i=1}^{n}\left(\text{dist}\left(x_i \leftrightarrow \text{line defined by } v\right)\right)^2$$





$$(x,v)^2 + \text{dist}^2 = \|x\|^2$$

$(x,v)^2$: max; dist^2: min

objective: choose v s.t. $\|v\|=1$

to max $\frac{1}{n}\sum_{i=1}^{n}(x_i,v)^2$

max avg squared projections

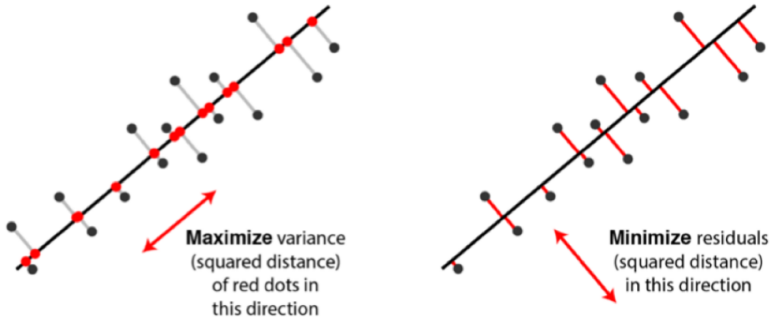
max variance of projections.

(x_1,v), (x_2,v) ... (x_n,v)

$\frac{1}{n}\sum_{i=1}^{n}(x_i,v)^2$

$\frac{1}{n}\sum_{i=1}^{n}(x_i,v) = \left(\frac{1}{n}\sum_{i=1}^{n}x_i, v\right) = 0$

$\text{Var}(X) = E\left((X-E(X))^2\right)$



Two equivalent views of principal component analysis.

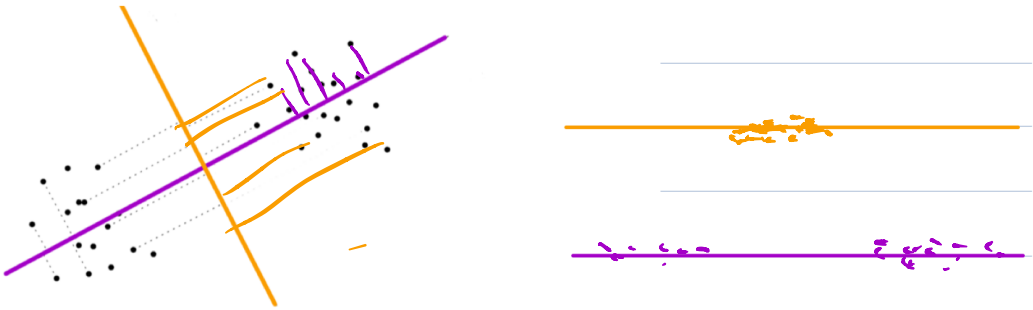
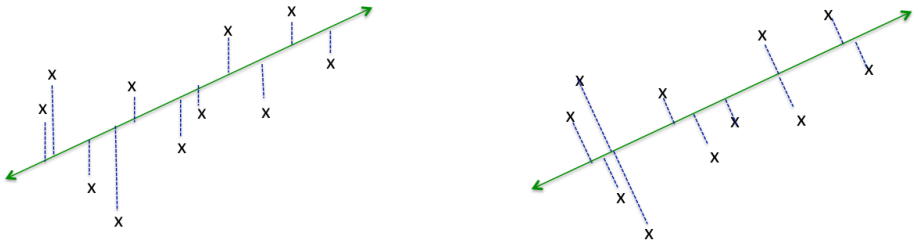


Figure 5: For the good line, the projection of the points onto the line keeps the two clusters separated, while the projection onto the bad line merges the two clusters.

Comparison to linear regression



(a) Linear regression

(b) PCA

Figure 2: Linear regression minimizes the sum of squared vertical distances, while PCA minimizes the sum of squared perpendicular distances.

Goal: express n d-dimensional vectors $\vec{x}_1, \ldots, \vec{x}_n \in \mathbb{R}^d$ as linear combinations of k vectors $\vec{v}_1, \ldots, \vec{v}_k \in \mathbb{R}^d$

so $\vec{x}_i \approx \sum_{j=1}^{k} a_{ij} \vec{v}_j$

Larger k

find k dim subspace s.t.

squared distances minimized $=$ variance of projection maximized.

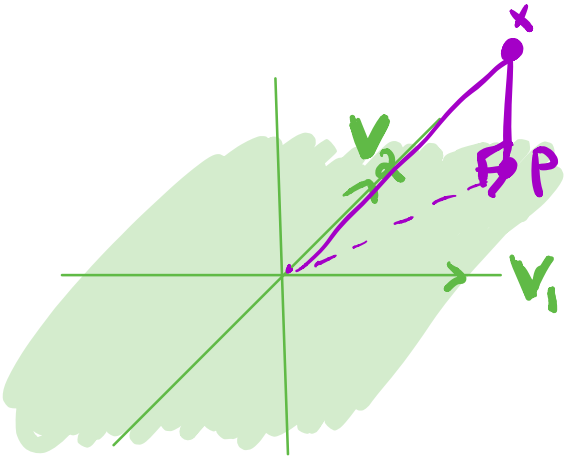
$$\max_{k \text{ dim subspaces } S} \frac{1}{n} \sum_{i=1}^{n} (\text{length of } x_i\text{'s projection})^2$$

define subspace using k orthonormal vectors $v_1, \dots, v_k \in \mathbb{R}^d$

$\|v_i\| = 1 \qquad (v_i, v_j) = 0 \text{ if } i \neq j$

subspace: span of $v_1, \dots, v_k$

find $v_1 \dots v_k$ to max $\frac{1}{n} \sum_{i=1}^{n} \boxed{\sum_{j=1}^{k} (x_i, v_j)^2}$ $\|p\|^2$



$p = a_1 v_1 + a_2 v_2$

(x, v_1)

$(x - p, v_1) = 0$

$(x, v_1) - (p, v_1) = (a_1 v_1 + a_2 v_2, v_1)$

$= a_1$

$p = (x, v_1) v_1 + (x, v_2) v_2$

Summary:

Problem: compute orthonormal vectors $v_1, \ldots, v_k$

to maximize $\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} (x_i \cdot v_j)^2$

avg sum of squared projection lengths

Application: Visualization

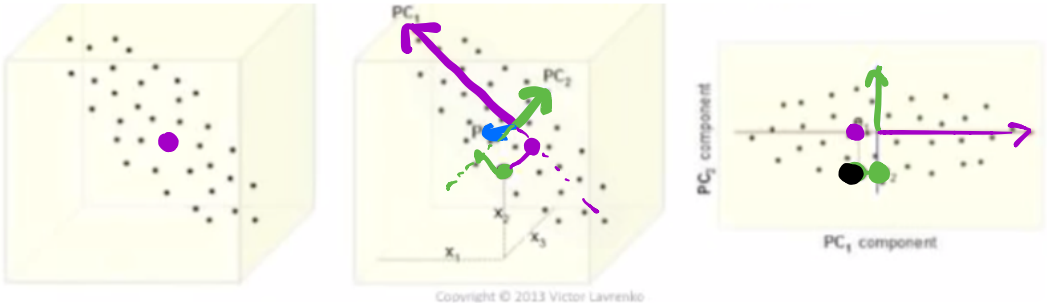
① Perform PCA $\rightarrow$ $\vec{v}_1, \ldots, \vec{v}_k$ top k "principal components"

② $\forall \vec{x}_i$ define

"$\vec{v}_1$-coord"	(x_i, v_1)
"$\vec{v}_2$-coord"	(x_i, v_2)
⋮	
"$\vec{v}_k$-coord"	(x_i, v_k)

③ Plot points

$x_i \rightarrow ((x_i, v_1), (x_i, v_2), \ldots, (x_i, v_k))$



* look for clusters

* look for pts particularly large along $\vec{v}_1$

Cool example
Genetic data of Europeans
[Novembre et al]

3192 people
200,000 SNPs
DNA positions that
tend to exhibit gene mutation.

$n = 3192$
$d = 200{,}000$

using only genomic data

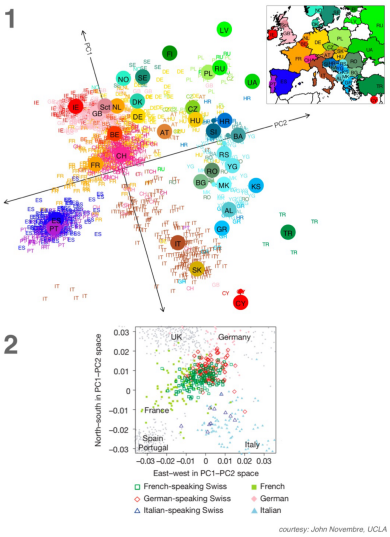


Figure 1: The genetic map of Europe using PCA, with the geographic map of Europe for reference. Figure 2: The same map, but zoomed in on Switzerland. Swiss individuals tend to cluster with countries that speak the same language. *(Courtesy: John Novembre, UCLA)*

How PCA works

$k=1$

$$\arg\max_{\|v\|=1} \frac{1}{n}\sum_{i=1}^{n}(x_i, v)^2$$

$x_i \in \mathbb{R}^d$
$v \in \mathbb{R}^d$

$$X = \begin{pmatrix} — x_1 — \\ \\ — x_n — \end{pmatrix} \quad n \times d \qquad Xv = \begin{pmatrix} x_1 \cdot v \\ \vdots \\ x_n \cdot v \end{pmatrix}$$

obj: $\frac{1}{n}(Xv)^T Xv$

$= \frac{1}{n} v^T X^T X v$

$A = \frac{1}{n} X^T X$

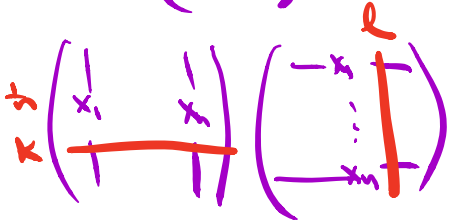
empirical covariance matrix

$$A_{ke} = \frac{1}{n}\sum_{i=1}^{n} x_{ik} x_{ie}$$

$(X^T X)^T = X^T X$

X: n docs.
d words

x_{ij} = # times word j appears in doc i



X_k random sample of k^{th} feature.

$x_{1k}, x_{2k} \cdots, x_{nk}$

X_e random sample of e^{th} feature.

$$A_{ke} = \underset{j}{\mathrm{Cov}}(X_k, X_e)$$

$$\frac{1}{n} X^T X = \begin{pmatrix} \mathrm{Var}(X_1) & & \\ & \ddots & \\ & & \mathrm{Var}(X_d) \end{pmatrix}$$

2 rvs Y & Z.

$\mathrm{Cov}(Y,Z) = E\big((Y - E(Y))(Z - E(Z))\big)$

if means are 0.

Cov $\Rightarrow E(YZ)$

Problem: find $\vec{v}$ with $\|v\|=1$

to maximize $v^T A v$

where $A = \frac{1}{n} X^T X$

Suppose magically A diagonal

optimal $v = (1,0,0)$

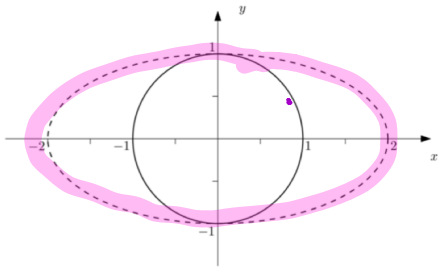


Figure 1: The point (x, y) on the unit circle is mapped to $(2x, y)$.

$$\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}$$

$$A = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{pmatrix}$$

$\max v^T A v \quad \|v\|^2 = 1$

$v^T A v = (v_1, v_2, v_3) \begin{pmatrix} 3 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.5 \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$

$\Rightarrow 3v_1^2 + 1 v_2^2 + 0.5 v_3^2$

$v_1^2 + v_2^2 + v_3^2 = 1$

$v_1 = 1 \quad v_2 v_3 = 0$

$\lambda_1 \geq \lambda_2 \geq \dots \lambda_d$

$$A = \begin{pmatrix} \lambda_1 & & 0 \\ & \lambda_2 & \\ 0 & & \ddots \\ & & & \lambda_d \end{pmatrix}$$

$\vec{v} = e_1$

Every symmetric matrix is "diagonals in disguise"

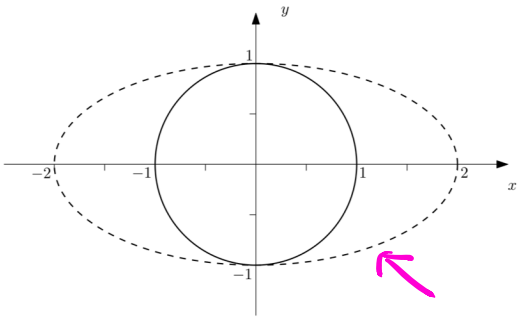


Figure 1: The point (x, y) on the unit circle is mapped to $(2x, y)$.

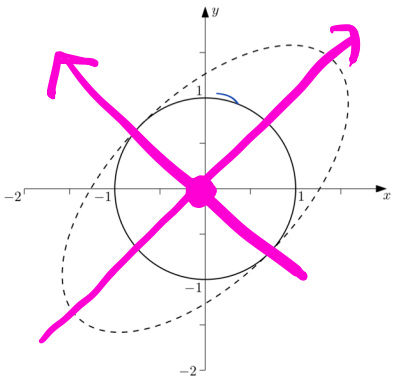


Figure 2: The same scaling as Figure 1, but now rotated 45 degrees.

$$\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} \frac{3}{2} & \frac{1}{2} \\ \frac{1}{2} & \frac{3}{2} \end{pmatrix} = \underbrace{\begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}}_{\text{rotate back } 45^\circ} \cdot \underbrace{\begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}}_{\text{stretch}} \cdot \underbrace{\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}}_{\text{rotate clockwise } 45^\circ}.$$

$A = Q D Q^T$

Q: orthogonal matrix whose columns are orthonormal

D: diagonal $\begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_2 \end{pmatrix}$

$Q^T Q = Q Q^T = I$

Orthogonal matrices preserve length

$\|Qw\|^2 = (Qw)^T Qw$

$= w^T Q^T Q w = w^T w = \|w\|^2$

I

$A = Q D Q^T$

$Q^T = \begin{pmatrix} — z_1^T — \\ \vdots \\ — z_n^T — \end{pmatrix}$

$Q = \begin{pmatrix} | & & | \\ z_1 & \cdots & z_n \\ | & & | \end{pmatrix}$

also has direction of max stretch

"rotated axis" that gets stretched the most.

Solution to maximization problem

$$\max_{v \text{ s.t. } \|v\|=1} v^T A v$$

$v^T A v = v^T Q D Q^T v$

$y = Q^T v$ $\quad y = (y_1, \ldots, y_n)^T$

$= y^T D y$

$D = \begin{pmatrix} \lambda_1 & & 0 \\ & \lambda_2 & \\ 0 & & \ddots \\ & & & \lambda_d \end{pmatrix}$

$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$

opt y is e_1

$Q^T v = e_1$

$Q Q^T v = Q e_1$

I

$v = Q e_1$

$Q^T v = e_1$

$Q Q^T v = Q e_1$

I

$v = Q e_1$

$\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$

Every matrix of form $A = X^T X$ can be written as $Q^T D Q$ where all $\lambda_i \geq 0$

$$Q^T = \begin{pmatrix} | & & | \\ z_1 & \cdots & z_n \\ | & & | \end{pmatrix}^T$$

$$Q = \begin{pmatrix} | & & | \\ z_1 & \cdots & z_n \\ | & & | \end{pmatrix}$$

$$D = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_d \end{pmatrix}$$

Conclusion: for $k=1$, soln is first column of Q

for general k, soln is first k cols of Q

$\equiv$ top k principal components

$\equiv$ top k eigenvectors of A

$A = Q D Q^T$

$AQe_i = QDQ^TQe_i$

$= QDe_i$

$= Q\lambda_i e_i$

$= \lambda_i Qe_i$

Qe_i is an eigenvector w/ eigenvalue λ_i:

$B\omega = \lambda\omega$

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Computing principal components:

1. SVD. [cubic]
2. power iteration

Algorithm 1
POWER ITERATION

Given matrix $\mathbf{A} = \mathbf{X}^\top \mathbf{X}$:

- Select random unit vector $\mathbf{u}_0$
- For $i = 1, 2, \ldots$, set $\mathbf{u}_i = \mathbf{A}^i \mathbf{u}_0$. If $\mathbf{u}_i / ||\mathbf{u}_i|| \approx \mathbf{u}_{i-1} / ||\mathbf{u}_{i-1}||$, then return $\mathbf{u}_i / ||\mathbf{u}_i||$.

To see why this works:

Suppose $v^1, v^2, \ldots, v^n$ are eigenvectors of A

Then they form an orthonormal basis.

$\Rightarrow \forall$ vector u: $u = \sum_{i=1}^{n} c_i v_i$

$$Au = A \sum c_i v_i = \sum c_i A v_i$$

$$= \sum c_i \lambda_i v_i$$

$$A^k u = A^{k-1} Au = \sum c_i \lambda_i A^{k-1} v_i$$

$$= \sum c_i \lambda_i^k v_i \qquad c_i \lambda_i A^{k-2} \lambda_i v_i \ldots$$

$$= c_1 \lambda_1^k \left[v_1 + \left(\frac{\lambda_2}{\lambda_1}\right)^k v_2 + \left(\frac{\lambda_3}{\lambda_1}\right)^k v_3 + \cdots \right]$$

$$\approx c_1 \lambda_1^k v_1$$

⇒ when we normalize,

get v_1

this depended on $\lambda_2 < \lambda_1$

& then can show that $O\left(\frac{\log n}{\log(\lambda_1/\lambda_2)}\right)$

iterations suffice to get very close.

1. Find the top component, $\mathbf{v}_1$, using power iteration.
2. Project the data matrix orthogonally to $\mathbf{v}_1$:

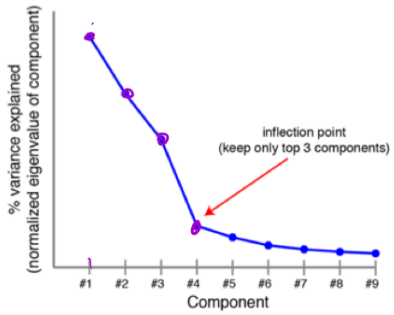
$$\begin{bmatrix} — & \mathbf{x}_1 & — \\ — & \mathbf{x}_2 & — \\ & \vdots & \\ — & \mathbf{x}_m & — \end{bmatrix} \mapsto \begin{bmatrix} — & (\mathbf{x}_1 - \langle \mathbf{x}_1, \mathbf{v}_1 \rangle \mathbf{v}_1) & — \\ — & (\mathbf{x}_2 - \langle \mathbf{x}_2, \mathbf{v}_1 \rangle \mathbf{v}_1) & — \\ & \vdots & \\ — & (\mathbf{x}_m - \langle \mathbf{x}_m, \mathbf{v}_1 \rangle \mathbf{v}_1) & — \end{bmatrix}.$$

   This corresponds to subtracting out the variance of the data that is already explained by the first principal component $\mathbf{v}_1$.
3. Recurse by finding the top $k-1$ principal components of the new data matrix.

"Greedy alg" correctness: k-dim subspace that maximizes norms of projections contains k-1-dimensional space that maximizes norms of projections.

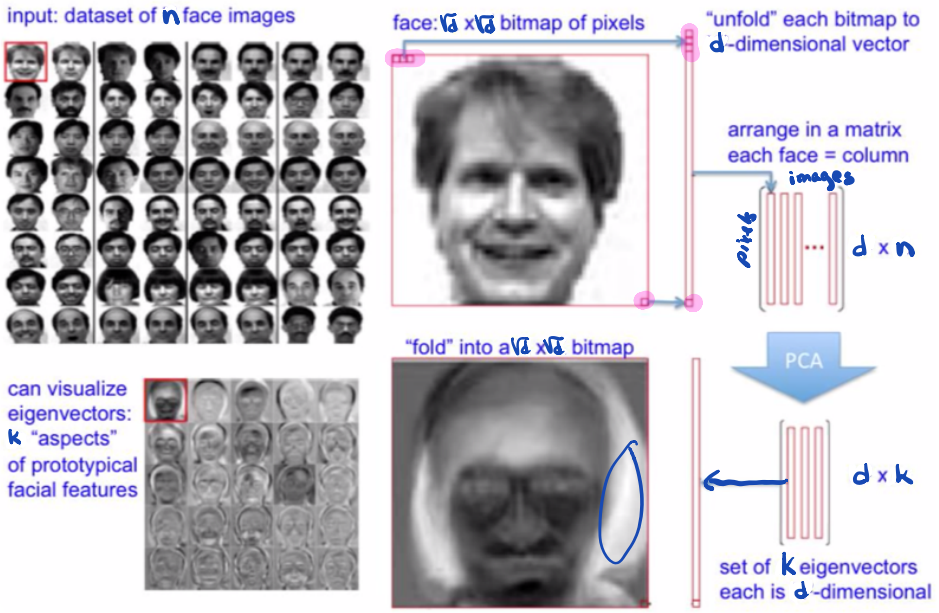
How to choose k?

- For data visualization: a few
- compression: Look at eigenvalues. As soon as small enough, happy.



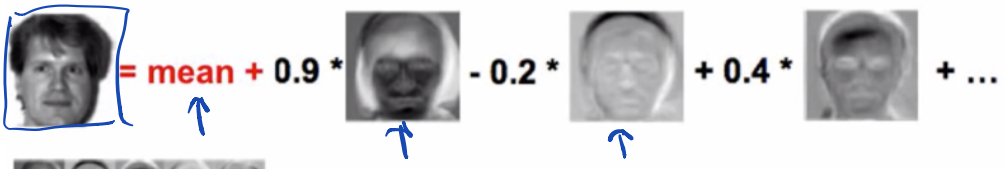
Scree plot. Principal components are ranked by the amount of variance they capture in the original dataset, a scree plot can provide some sense of how many components are needed.

PCA example: Eigen Faces

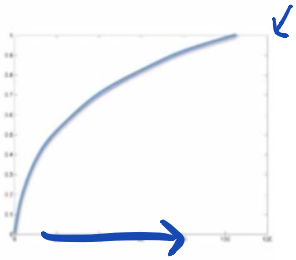


Eigenfaces Slides by Victor Lavrenko

Eigen Faces: Projection



- Project new face to space of eigen-faces
- Represent vector as a linear combination of principal components
- How many do we need?



(Eigen) Face Recognition

- Face similarity
  - in the reduced space
  - insensitive to lighting expression, orientation
- Projecting new "faces"
  - everything is a face

new face

projected to eigenfaces