Eigenfaces for recognition

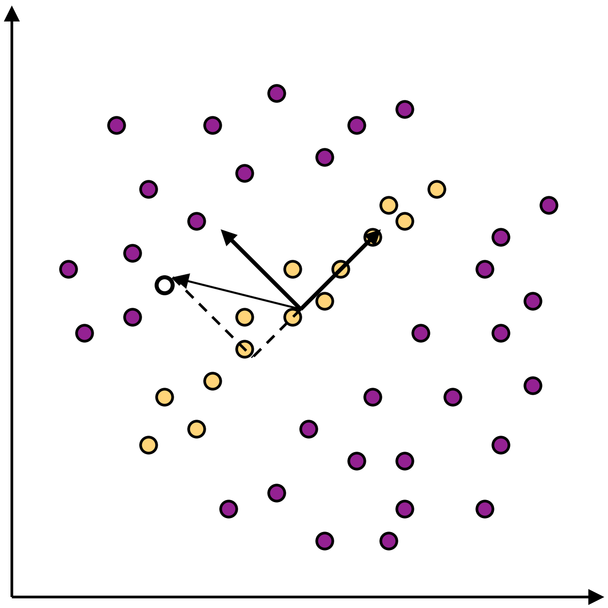
Matthew Turk and Alex Pentland

J. Cognitive Neuroscience

1991

Linear subspaces

$\overline{x}$ is the mean of the orange points



convert **x** into $\mathbf{v_1}$, $\mathbf{v_2}$ coordinates

$$\mathbf{x} \rightarrow ((\mathbf{x} - \overline{x}) \cdot \mathbf{v_1}, (\mathbf{x} - \overline{x}) \cdot \mathbf{v_2})$$

What does the $\mathbf{v_2}$ coordinate measure?

- distance to line
- use it for classification—near 0 for orange pts

What does the $\mathbf{v_1}$ coordinate measure?

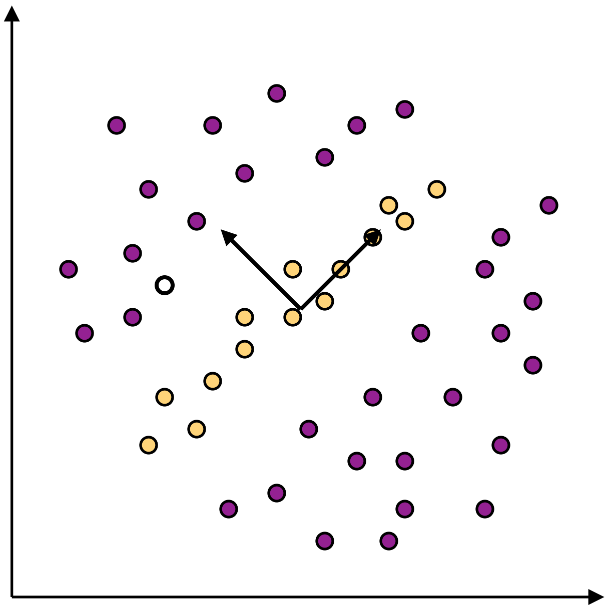
- position along line
- use it to specify which orange point it is

Suppose the data points are arranged as above

Idea: fit a line, classifier measures distance to line

Dimensionality reduction

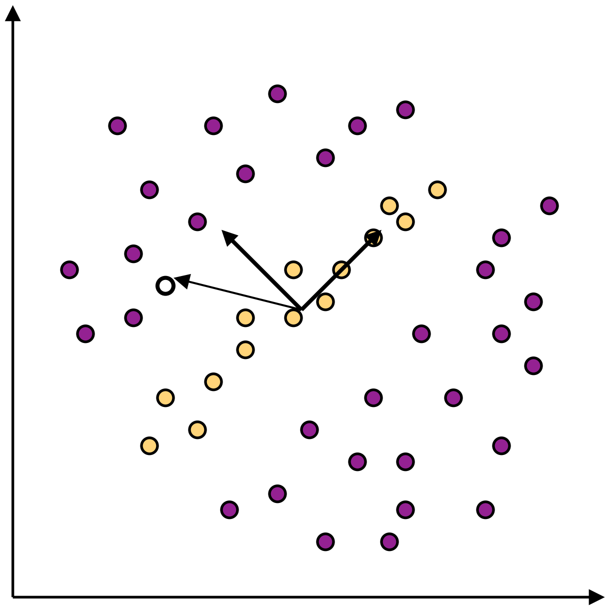
$\overline{x}$ is the mean of the orange points



- We can represent the orange points with *only* their $\mathbf{v_1}$ coordinates (since $\mathbf{v_2}$ coordinates are all essentially 0)
- This makes it much cheaper to store and compare points
- A bigger deal for higher dimensional problems

Linear subspaces

$\overline{x}$ is the mean of the orange points



Consider the variation along direction **v** among all of the orange points:

$$var(\mathbf{v}) = \sum_{\text{orange point } \mathbf{x}} \|(\mathbf{x}-\overline{\mathbf{x}})^{\mathbf{T}} \cdot \mathbf{v}\|^{\mathbf{2}}$$

What unit vector **v** minimizes *var*?

$$\mathbf{v}_2 = min_{\mathbf{v}} \ \{var(\mathbf{v})\}$$

What unit vector **v** maximizes *var*?

$$\mathbf{v}_1 = max_{\mathbf{v}} \ \{var(\mathbf{v})\}$$

$$\begin{aligned} var(\mathbf{v}) &= \sum_{\mathbf{x}} \|(\mathbf{x}-\overline{\mathbf{x}})^{\mathbf{T}} \cdot \mathbf{v}\| \\ &= \sum_{\mathbf{x}} \mathbf{v}^{\mathbf{T}}(\mathbf{x}-\overline{\mathbf{x}})(\mathbf{x}-\overline{\mathbf{x}})^{\mathbf{T}}\mathbf{v} \\ &= \mathbf{v}^{\mathbf{T}}\left[\sum_{\mathbf{x}}(\mathbf{x}-\overline{\mathbf{x}})(\mathbf{x}-\overline{\mathbf{x}})^{\mathbf{T}}\right]\mathbf{v} \\ &= \mathbf{v}^{\mathbf{T}}\mathbf{A}\mathbf{v} \quad \text{where } \mathbf{A} = \sum_{\mathbf{x}}(\mathbf{x}-\overline{\mathbf{x}})(\mathbf{x}-\overline{\mathbf{x}})^{\mathbf{T}} \end{aligned}$$

Solution: $\mathbf{v_1}$ is eigenvector of **A** with *largest* eigenvalue
$\mathbf{v_2}$ is eigenvector of **A** with *smallest* eigenvalue

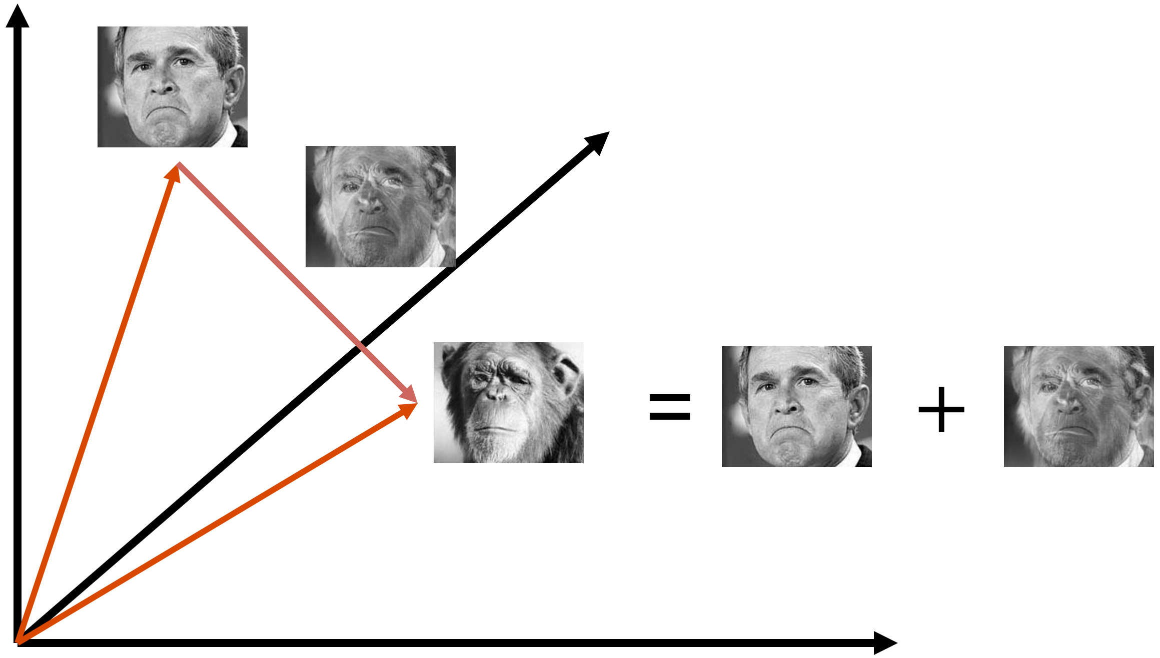
Principal component analysis

- Suppose each data point is N-dimensional
  - Same procedure applies:

$$
\begin{aligned}
var(\mathbf{v}) &= \sum_{\mathbf{x}} \|(\mathbf{x}-\overline{\mathbf{x}})^{\mathbf{T}} \cdot \mathbf{v}\| \\
&= \mathbf{v}^{\mathbf{T}}\mathbf{A}\mathbf{v} \quad \text{where } \mathbf{A} = \sum_{\mathbf{x}} (\mathbf{x}-\overline{\mathbf{x}})(\mathbf{x}-\overline{\mathbf{x}})^{\mathbf{T}}
\end{aligned}
$$

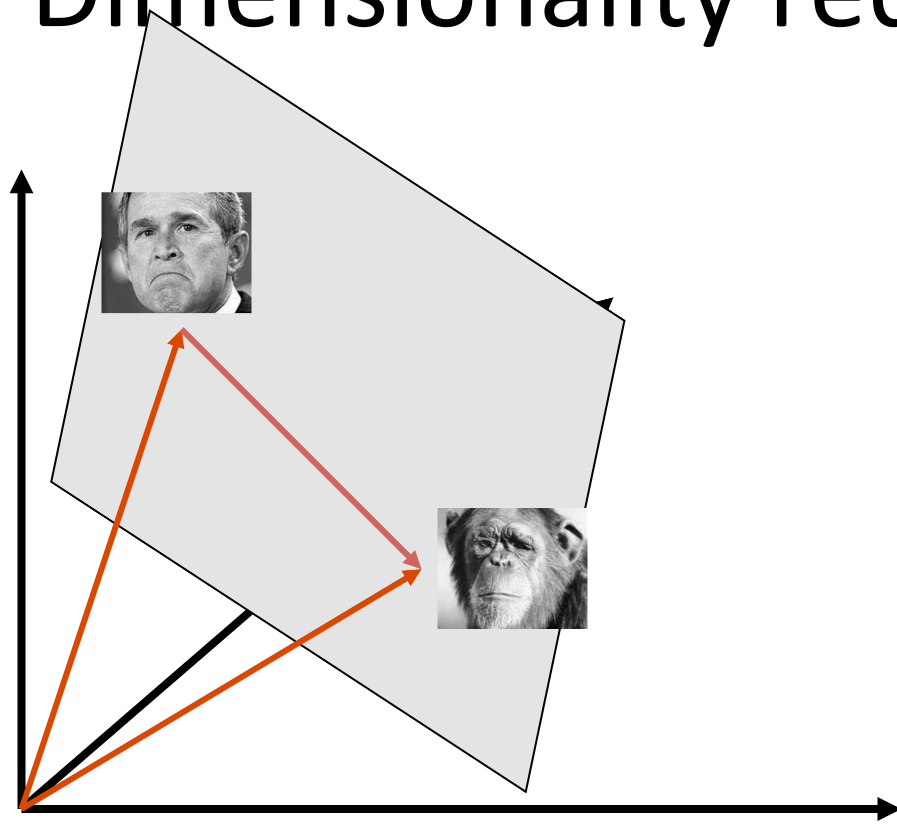
  - The eigenvectors of **A** define a new coordinate system
    - eigenvector with largest eigenvalue captures the most variation among training vectors **x**
    - eigenvector with smallest eigenvalue has least variation
  - We can compress the data using the top few eigenvectors
    - corresponds to choosing a "linear subspace"
      - represent points on a line, plane, or "hyper-plane"
    - these eigenvectors are known as the ***principal components***

The space of faces



- An image is a point in a high dimensional space
  - An N x M image is a point in R^{NM}
  - We can define vectors in this space as we did in the 2D case

Dimensionality reduction



- The set of faces is a "subspace" of the set of images
  - We can find the best subspace using PCA
  - This is like fitting a "hyper-plane" to the set of faces
    - spanned by vectors $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_K}$
    - any face $\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_k\mathbf{v_k}$

Eigenfaces

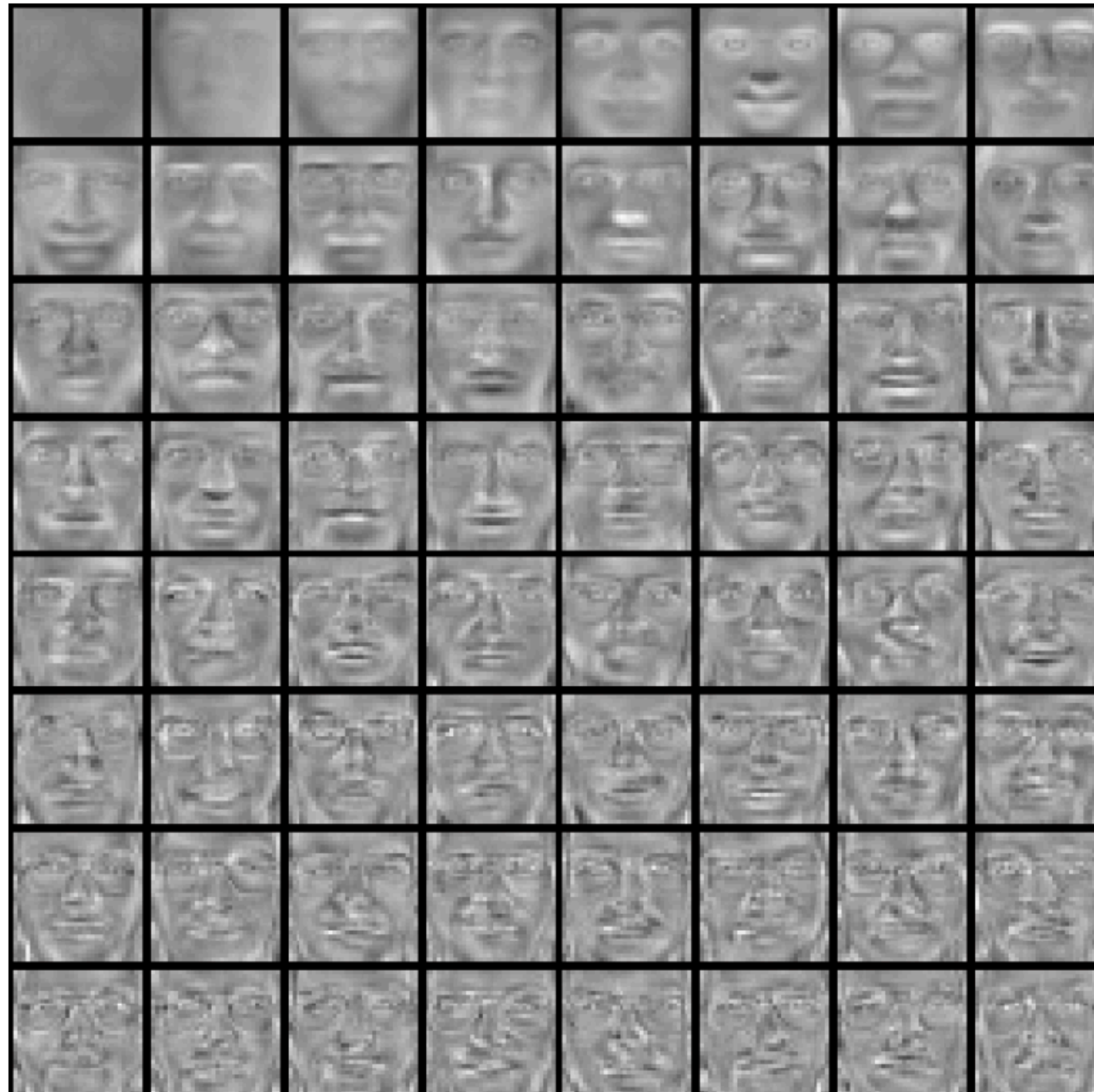
- PCA extracts the eigenvectors of **A**
  - Gives a set of vectors $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, ...
  - Each vector is a direction in face space
    - what do these look like?

Training images

$\mathbf{x}_1, \ldots, \mathbf{x}_N$

Mean: $\boldsymbol{\mu}$

Top eigenvectors: $\mathbf{u}_1, \ldots \mathbf{u}_k$



Projecting onto the eigenfaces

- The eigenfaces $\mathbf{v_1}, \ldots, \mathbf{v_K}$ span the space of faces
  - A face is converted to eigenface coordinates by

$$\mathbf{x} \rightarrow (\underbrace{(\mathbf{x} - \overline{\mathbf{x}}) \cdot \mathbf{v_1}}_{a_1}, \underbrace{(\mathbf{x} - \overline{\mathbf{x}}) \cdot \mathbf{v_2}}_{a_2}, \ldots, \underbrace{(\mathbf{x} - \overline{\mathbf{x}}) \cdot \mathbf{v_K}}_{a_K})$$

$$\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_K\mathbf{v_K}$$

Recognition with eigenfaces

1. Process the image database (set of images with labels)
   - Run PCA—compute eigenfaces
   - Calculate the K coefficients for each image

2. Given a new image (to be recognized) **x**, calculate K coefficients

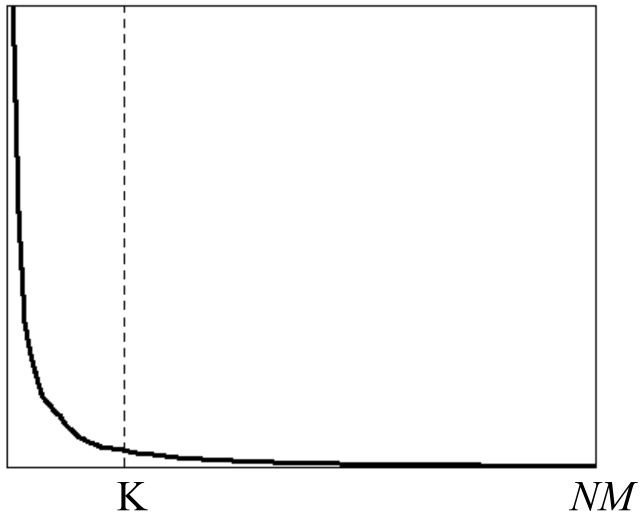
$$\mathbf{x} \rightarrow (a_1, a_2, \ldots, a_K)$$

3. Detect if x is a face

$$\|\mathbf{x} - (\overline{\mathbf{x}} + a_1\mathbf{v_1} + a_2\mathbf{v_2} + \ldots + a_K\mathbf{v_K})\| < \text{threshold}$$

4. If it is a face, who is it?
   - Find closest labeled face in database
     - nearest-neighbor in K-dimensional space

Choosing the dimension K



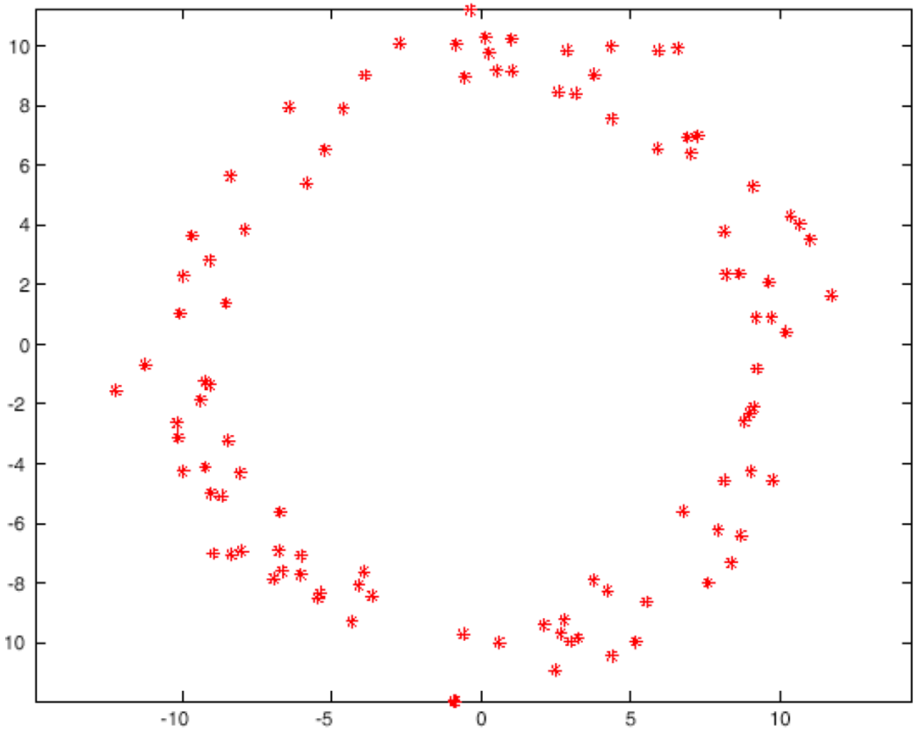
$$\frac{\sum_{i=1}^{K} \lambda_i}{\sum_{i=1}^{N} \lambda_i} > Threshold \quad \text{(e.g., 0.9 or 0.95)}$$

In this case, we say that we "preserve" 90% or 95% of the information in our data.

- How many eigenfaces to use?
- Look at the decay of the eigenvalues
  - the eigenvalue tells you the amount of variance "in the direction" of that eigenface
  - ignore eigenfaces with low variance

Limitations

- PCA assumes that the data has a Gaussian distribution (mean μ, covariance matrix Σ)



The shape of this dataset is not well described by its principal components

Limitations

- The direction of maximum variance is not always good for classification