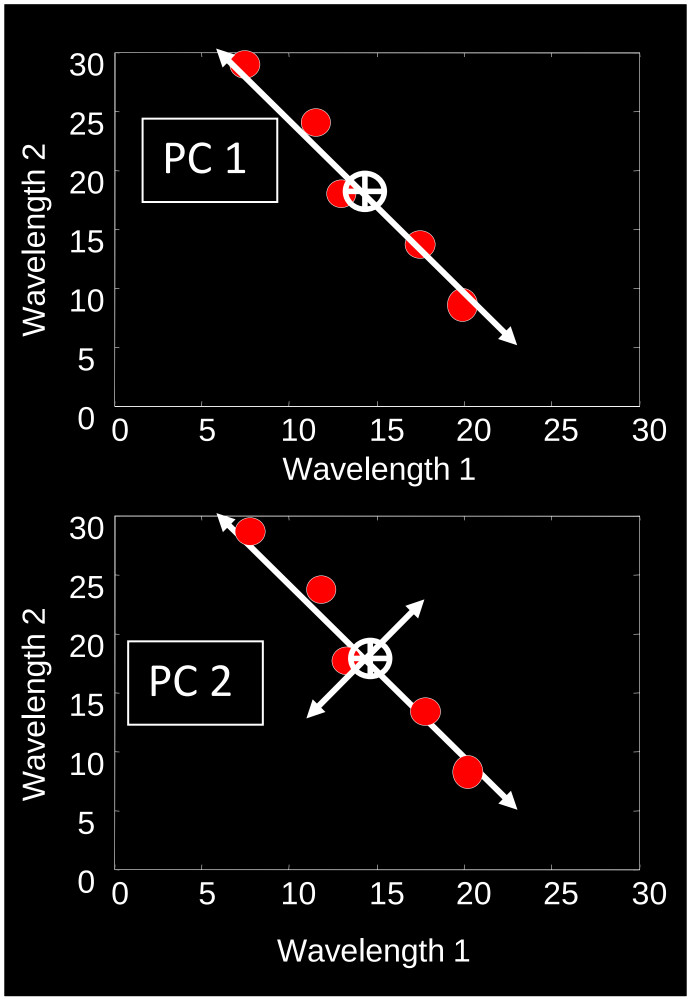
CS4495/6495
Introduction to Computer Vision

8B-L2 *Principle Component Analysis (and its use in Computer Vision)*

Figure 2. Seven of the eigenfaces calculated from the input images of Figure 1.

Principal Components

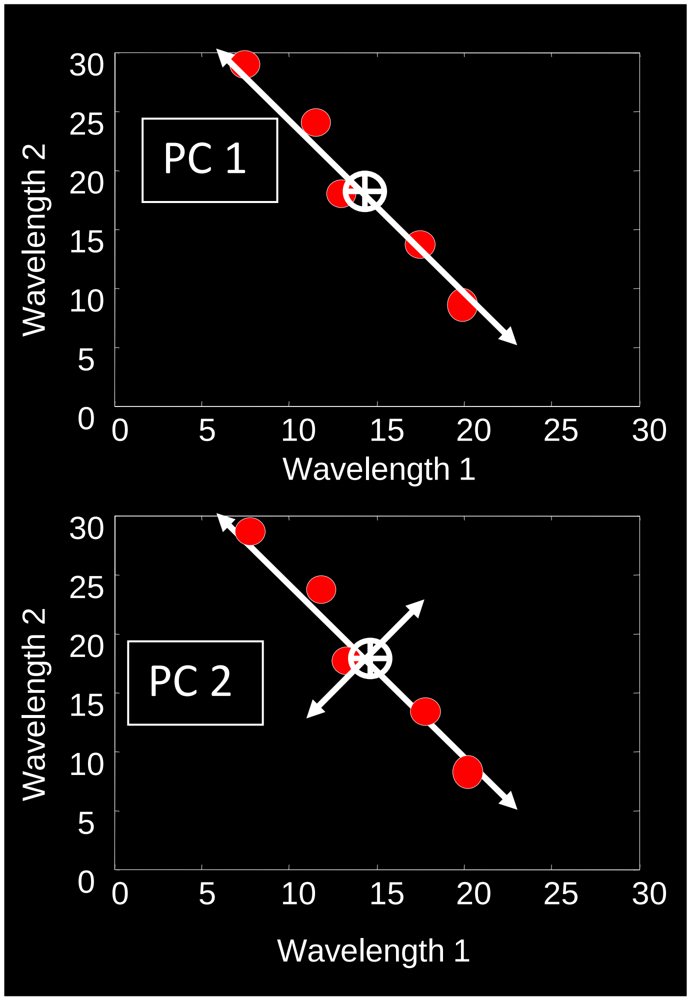
- *Principal components* are all about the directions in a feature space along which points have the greatest variance.



Principal Components

- First PC is the direction of maximum variance. Technically (and mathematically) it's from the origin, but we actually mean the mean.

- Subsequent PCs are orthogonal to previous PCs and describe maximum residual variance

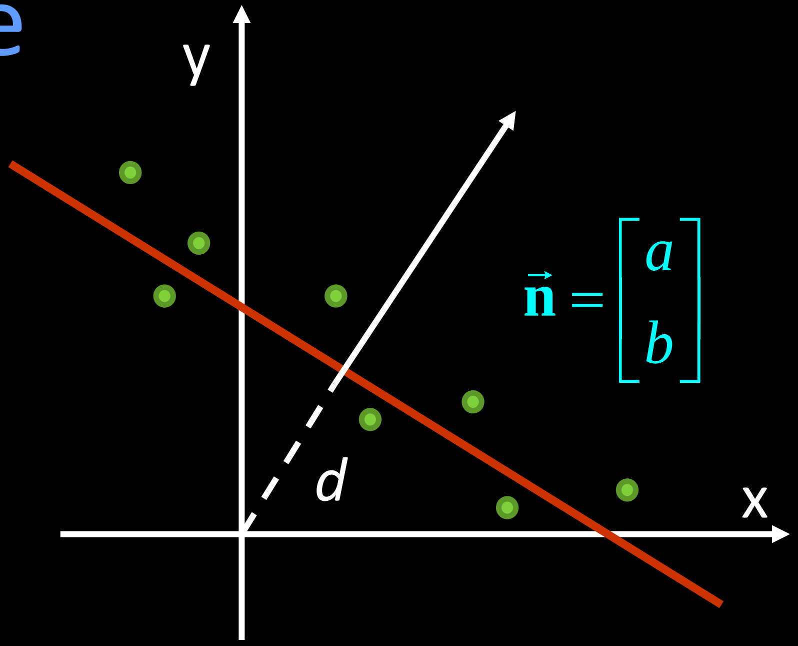


2D example: Fitting a line

$$E(a,b,d)=\sum_i (ax_i+by_i-d)^2$$

$$\frac{\partial E}{\partial d}=0 \rightarrow -2\sum_i (ax_i+by_i-d)=0$$

$$d=a\overline{x}+b\overline{y}$$



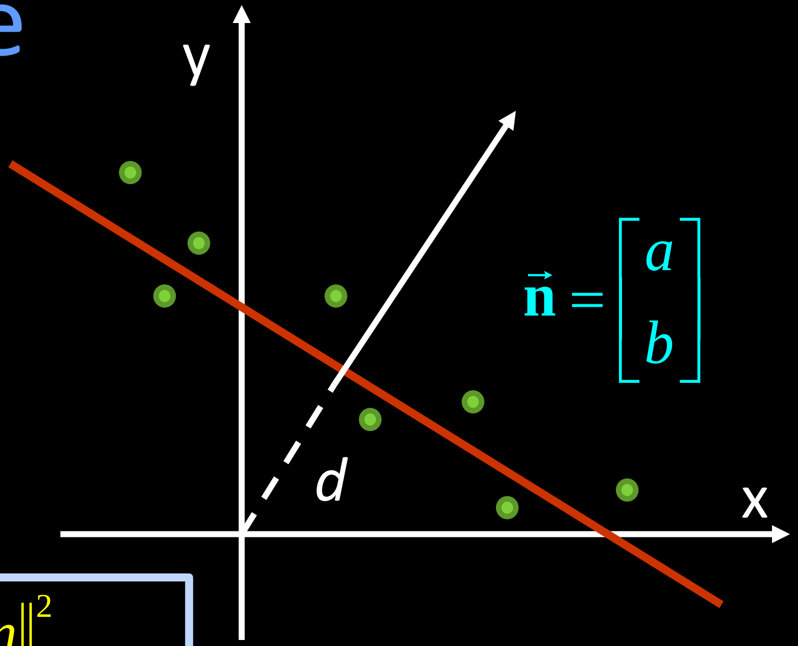
2D example: Fitting a line

Substitute $d = a\bar{x} + b\bar{y}$:

$$E = \sum_i [a(x_i - \bar{x}) + b(y_i - \bar{y})]^2 = \|\mathbf{B}n\|^2$$

where $\mathbf{B} = \begin{pmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ x_2 - \bar{x} & y_2 - \bar{y} \\ \ldots & \ldots \\ x_n - \bar{x} & y_n - \bar{y} \end{pmatrix}$

so minimize $\|\mathbf{B}n\|^2$
subject to $\|n\| = 1$ gives
axis of least inertia

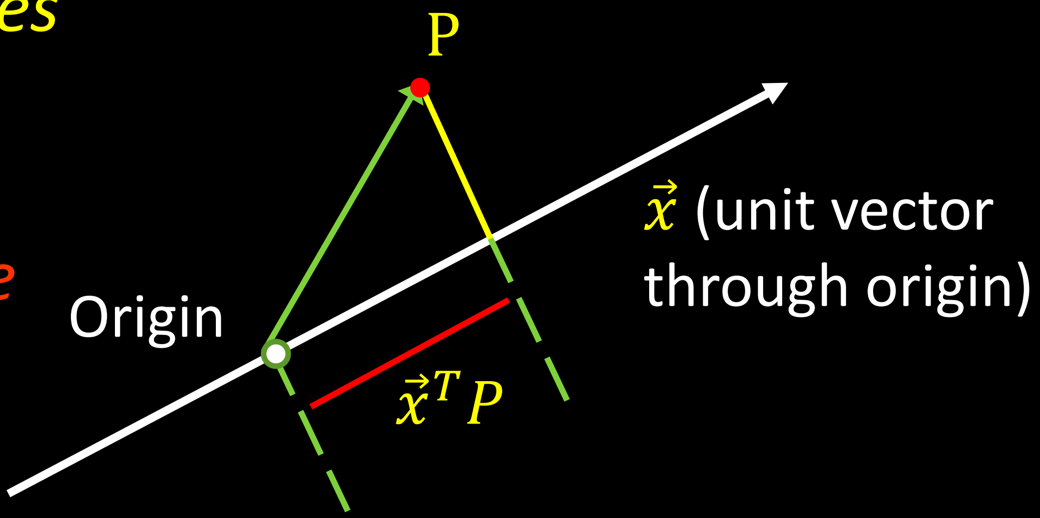


$$\vec{\mathbf{n}} = \begin{bmatrix} a \\ b \end{bmatrix}$$

Sound familiar???

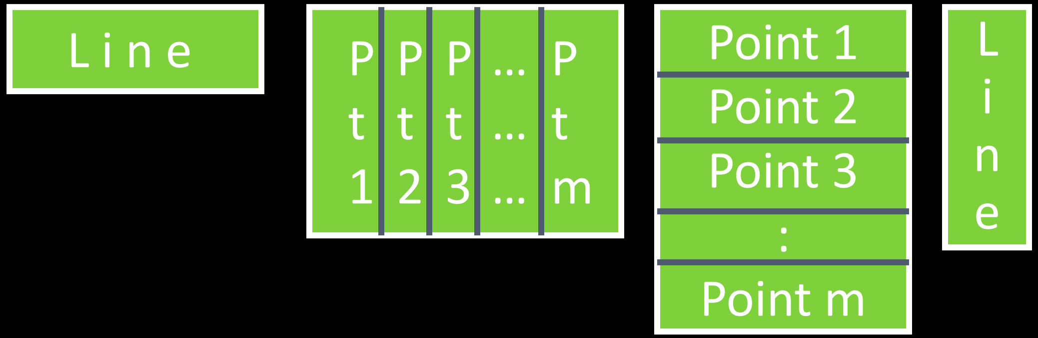
Another algebraic interpretation

Minimizing sum of squares of distances to the line is the same as *maximizing the sum of squares of the projections on that line.*



Algebraic interpretation

Trick: How is the sum of squares of projection lengths expressed in algebraic terms?

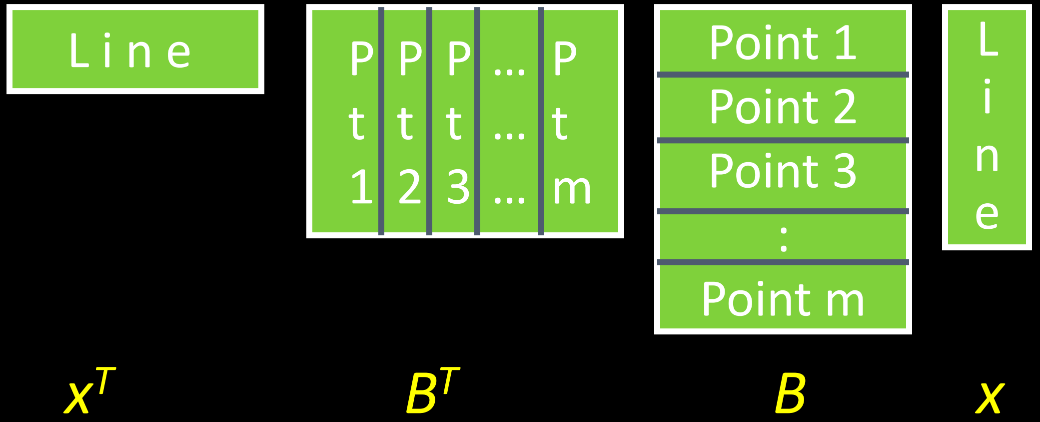


x^T B^T B x

$$\boxed{= \sum_i (\vec{x}^T P_i)^2}$$

Algebraic interpretation

Our goal: max($\mathbf{x}^T\mathbf{B}^T\mathbf{B}\mathbf{x}$), subject to $x^T x = 1$



$$\boxed{= \sum_i (\vec{x}^T P_i)^2}$$

Algebraic interpretation

max($\mathbf{x}^{\mathrm{T}}\mathbf{B}^{\mathrm{T}}\mathbf{B}\mathbf{x}$), subject to $\mathrm{x}^{\mathrm{T}}\mathrm{x} = 1$

maximize $E = \mathbf{x}^T \mathbf{M}\mathbf{x}$ subject to $\mathbf{x}^T \mathbf{x} = 1$ $(\mathbf{M} = \mathbf{B}^T \mathbf{B})$

$$E' = \mathbf{x}^T \mathbf{M}\mathbf{x} + \lambda(1 - \mathbf{x}^T \mathbf{x})$$

$$\frac{\partial E'}{\partial \mathbf{x}} = 2\mathbf{M}\mathbf{x} + 2\lambda\mathbf{x}$$

$$\frac{\partial E'}{\partial \mathbf{x}} = 0 \rightarrow \mathbf{M}\mathbf{x} = \lambda\mathbf{x} \quad (\mathbf{x} \text{ is an } \textit{eigenvector} \text{ of } \mathrm{B}^T\mathbf{B})$$

Yet another algebraic interpretation

$$\mathbf{B}^T\mathbf{B} = \begin{pmatrix} \sum_{i=1}^{n} x_i^2 & \sum_{i=1}^{n} x_i y_i \\ \sum_{i=1}^{n} x_i y_i & \sum_{i=1}^{n} y_i^2 \end{pmatrix} \text{ if about } \textit{origin}$$

So the principal components are the orthogonal directions of the covariance matrix of a set points.

Yet one more algebraic interpretation

$$\mathbf{B}^T\mathbf{B} = \sum \mathbf{x}\mathbf{x}^T \quad \text{if about } \textit{origin}$$

$$\mathbf{B}^T\mathbf{B} = \sum (\mathbf{x}-\bar{\mathbf{x}})(\mathbf{x}-\bar{\mathbf{x}})^T \textit{otherwise} - \textit{outer product}$$

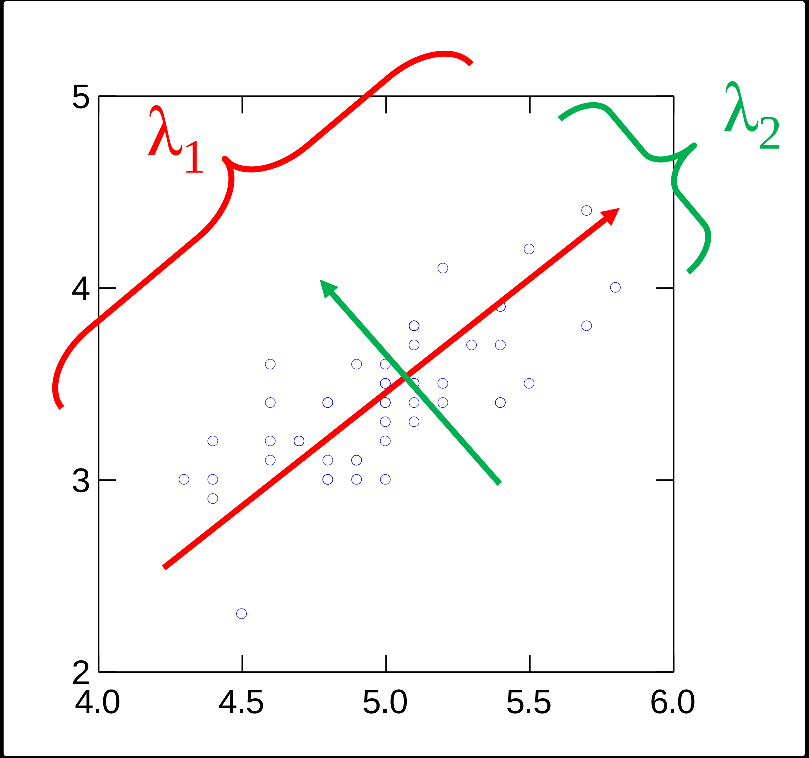
So the principal components are the orthogonal directions of the *covariance* matrix of a set points.

Eigenvectors

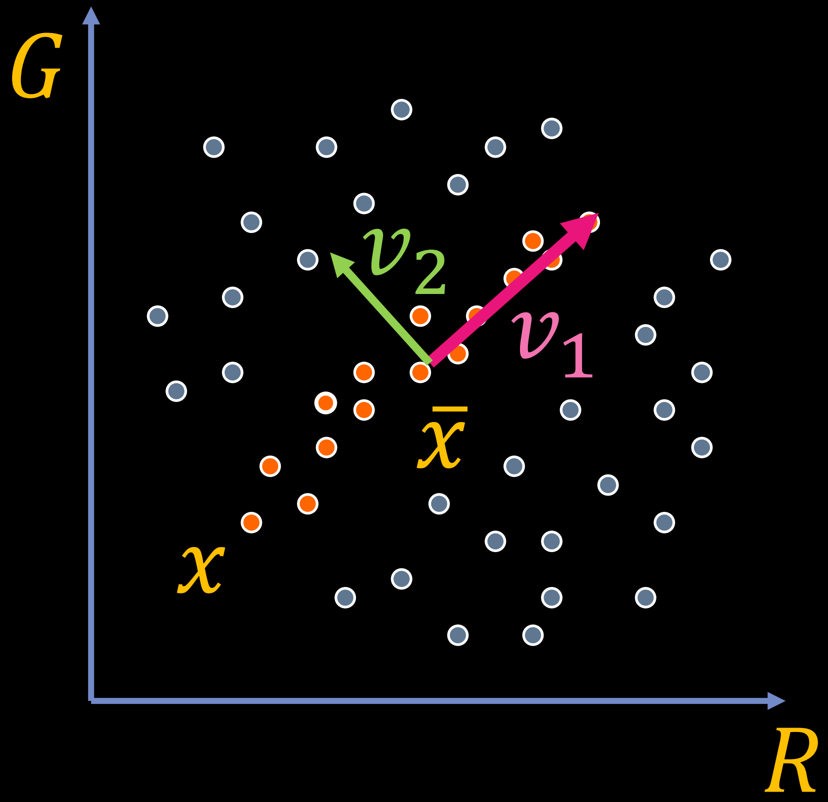
How many eigenvectors are there?

- For Real Symmetric Matrices of size NxN
  - Except in degenerate cases when eigenvalues repeat, there are N distinct eigenvectors

PCA: Eigenvalues



Dimensionality Reduction



We can represent the **orange** points with *only* their v_1 coordinates

Higher Dimensions

- Suppose each data point is N-dimensional

$$\mathrm{var}(\mathbf{v}) = \sum_{\mathbf{x}} \left\| (\mathbf{x} - \bar{\mathbf{x}})^T \cdot \mathbf{v} \right\|$$

Outer product

$$= \mathbf{v}^T \mathbf{A}\, \mathbf{v} \text{ where } \mathbf{A} = \sum (\mathbf{x} - \bar{\mathbf{x}})(\mathbf{x} - \bar{\mathbf{x}})^T$$

- The *ev* with largest eigenvalue λ captures the most variation among training vectors x
  - eigenvector with smallest eigenvalue has least variation

The space of all face images

When viewed as *vectors* of pixel values, face images are extremely high-dimensional

- 100x100 image = 10,000 dimensions

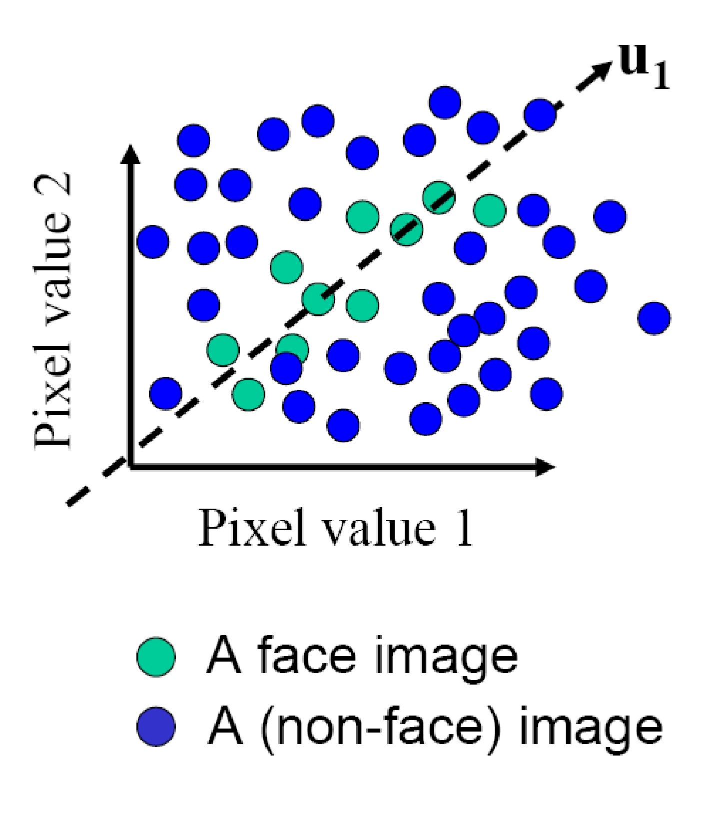
The space of all face images

However, relatively few 10,000-dimensional vectors correspond to valid face images

- We want to effectively model the subspace of face images

The space of all face images

We want to construct a *low-dimensional* *linear* *subspace* that best explains the variation in the set of face images



Principal Component Analysis

- Given: M data points $x_1, \ldots, x_M$ in R^d where d is big

- We want some directions in R^d that capture most of the variation of the x_i. The coefficients would be:

$$u(x_i) = u^T(x_i - \mu)$$

(μ: mean of data points)

- What unit vector u in R^d captures the most variance of the data?

Principal Component Analysis

- Direction that maximizes the variance of the projected data:

$$\mathrm{var}(\mathbf{u}) = \frac{1}{M}\sum_{i=1}^{M}\underbrace{\mathbf{u}^T(\mathbf{x}_i - \mu)}_{\text{Projection of data point}}(\mathbf{u}^T(\mathbf{x}_i - \mu))^T$$

$$= \mathbf{u}^T\left[\underbrace{\frac{1}{M}\sum_{i=1}^{N}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T}_{\text{Covariance matrix of data}}\right]\mathbf{u}$$

$$= \mathbf{u}^T\boldsymbol{\Sigma}\mathbf{u}$$

Principal Component Analysis

- Direction that maximizes the variance of the projected data:

$$\mathrm{var}(\mathbf{u}) = \mathbf{u}^T \boldsymbol{\Sigma} \mathbf{u}$$

Since u is a unit vector that can be expressed in terms of some linear sum of the eigenvectors of Σ, then direction of u that maximizes the variance is the eigenvector associated with the largest eigenvalue of Σ

Principal component analysis

- The *direction that captures the maximum covariance* of the data is the *eigenvector* corresponding to the *largest eigenvalue* of the *data covariance* matrix

- Furthermore, the top *k* orthogonal directions that capture the most variance of the data are the *k* eigenvectors corresponding to the *k* largest eigenvalues

- But first, we'll need the PCA d>>>n trick…

The dimensionality trick

Let Φ_i be the (very big vector length d) that is face image I with the mean image subtracted.

Define $C = \frac{1}{M} \sum \Phi_i \Phi_i^T = AA^T$

where $A = [\Phi_1 \Phi_2 \ldots \Phi_M]$ is the matrix of faces, and is $d \times M$.

Note: C is a huge $d \times d$ matrix (remember d is the length of the vector of the image).

The dimensionality trick

So $C = AA^T$ is a huge matrix.

But consider $A^T A$. It is only $M \times M$. Finding those eigenvalues is easy.

Suppose $\mathbf{v}_i$ is an eigenvector $A^T A$:

$$A^T A\mathbf{v}_i = \lambda \mathbf{v}_i$$

Premultiply by A:

$$AA^T A\mathbf{v}_i = \lambda A\mathbf{v}_i$$

So: $\boxed{A\mathbf{v}_i \text{ are the eigenvectors of } C = AA^T}$

How many eigenvectors are there?

- If had $M > d$ then there would be *d*. But $M \ll d$.
- So intuition would say there are *M* of them.
- But wait: if 2 points in 3D, how many eigenvectors? 3 points?
- Subtracting out the mean yields *M-1.*

Eigenfaces: Key idea *(Turk and Pentland, 1991)*

- Assume that most face images lie on a low-dimensional subspace determined by the first k ($k <<< d$) directions of maximum variance
- Use PCA to determine the vectors or "eigenfaces" $u_1, u_2, \ldots u_k$ that span that subspace
- Represent all face images in the dataset as linear combinations of eigenfaces. Find the coefficients by dot product.

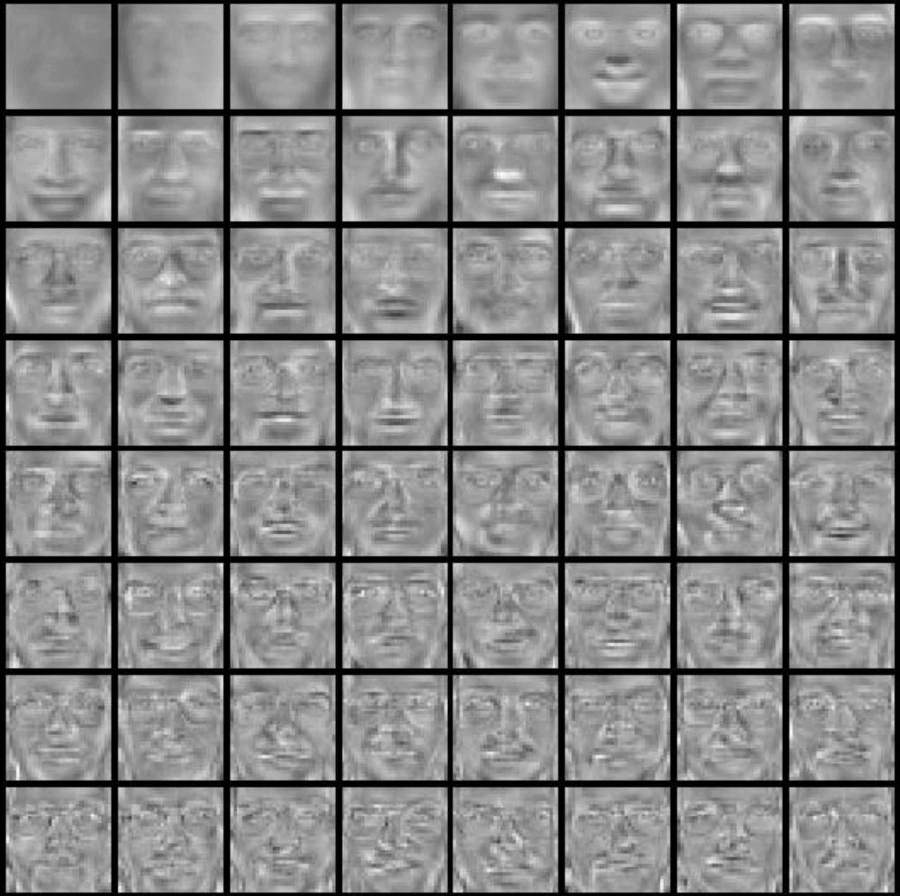
Eigenfaces example

Training images

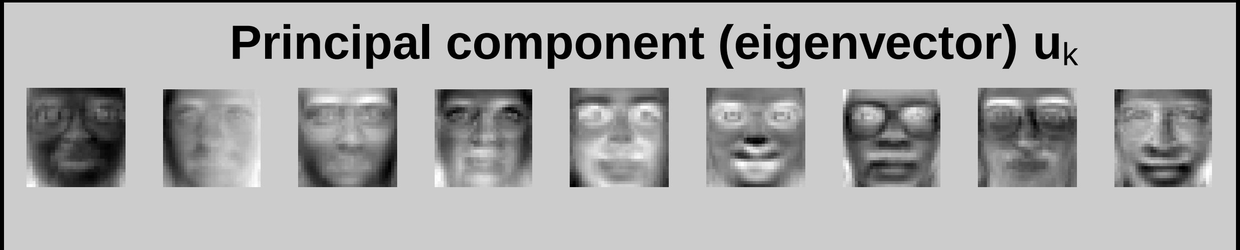
$\boldsymbol{x}_1, \ldots, x_M$

Mean: μ

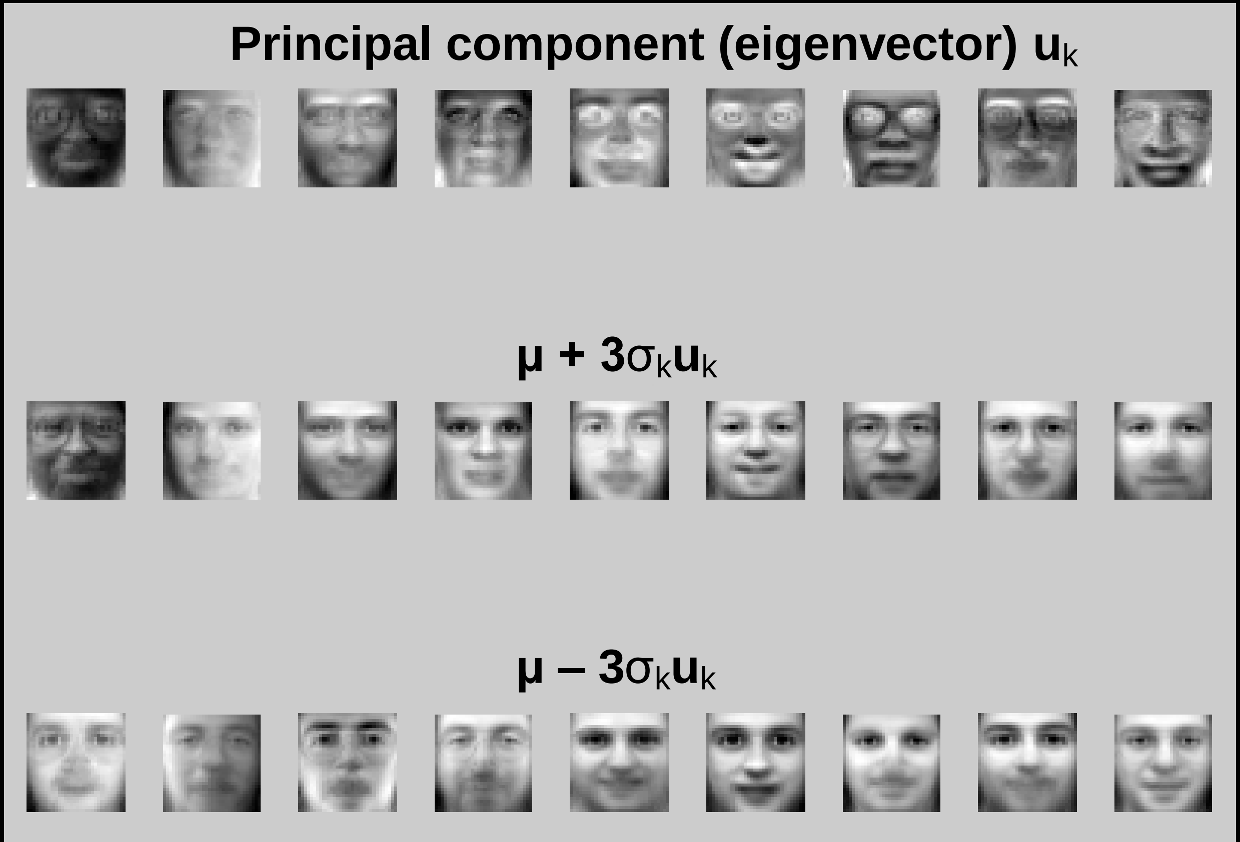
Top eigenvectors: $u_1,\ldots u_k$



Eigenfaces example



Eigenfaces example



Eigenfaces example

- Face $\mathbf{x}$ in "face space" coordinates (dot products) :

$$\mathbf{x} \rightarrow [\mathbf{u}_1^T(\mathbf{x}-\mu), \ldots, \mathbf{u}_k^T(\mathbf{x}-\mu)]$$
$$= [w_1, \ldots w_k]$$

This vector is the representation of the face.

Eigenfaces example

- Reconstruction:

 =  + 

$$\hat{\mathrm{x}} = \mu + w_1\mathbf{u}_1 + w_2\mathbf{u}_2 + w_3\mathbf{u}_3 + w_4\mathbf{u}_4 + \ldots$$

Eigenfaces example

- But what about recognition?

Recognition with eigenfaces

Given novel image $\mathbf{x}$:

- Project onto subspace:
  $$[w_1, \ldots, w_k] = [\mathrm{u}_1^T(\mathbf{x} - \mu), \ldots, \mathrm{u}_k^T(\mathbf{x} - \mu)]$$
- Optional: check reconstruction error $\mathbf{x} - \hat{\mathbf{x}}$ to determine whether image is really a face
- Classify as closest training face in k-dimensional subspace
- This is why it's a *generative* model.

And old cast of characters…

Limitations – PCA of global structure

- Global appearance method: not robust to misalignment, background variation

Limitations – PCA in general

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