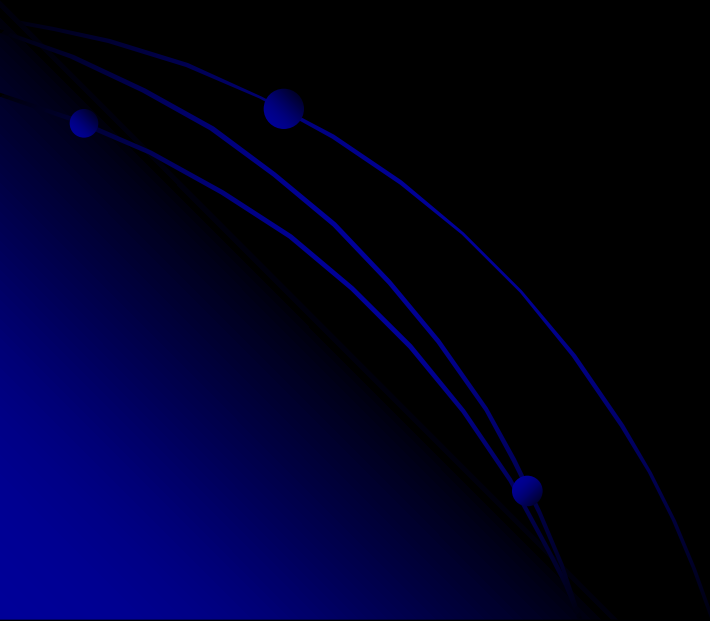
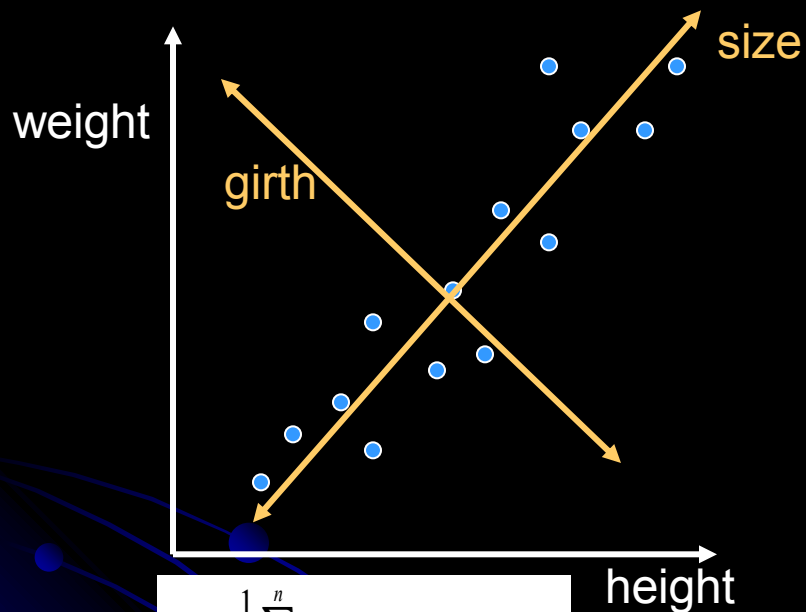


- This week
 - Today: Applications of PCA
 - Sunday night: project plans and prelim work due
- Questions?



Principal Components Analysis



$$\sigma_{xx} = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})(x_i - \bar{x})$$

$$\sigma_{xy} = \sigma_{yx} = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})$$

$$\sigma_{yy} = \frac{1}{n} \sum_{i=1}^n (y_i - \bar{y})(y_i - \bar{y})$$

$$c = \begin{pmatrix} \sigma_{xx} & \sigma_{xy} \\ \sigma_{yx} & \sigma_{yy} \end{pmatrix}$$

- Given a set of samples, find the direction(s) of greatest variance.
- We've done this!
- **Example: Spatial moments**
 - Principal axes are eigenvectors of covariance matrix
 - Eigenvalues gave relative importance of each dimension
 - Note that each point can be represented in 2D using the new coordinate system defined by the eigenvectors
 - The 1D representation obtained by projecting the point onto the principal axis is a reasonably-good approximation

Covariance Matrix (using matrix operations)

Place the points in their own column.

$$F = \begin{pmatrix} x_1 & x_2 & x_3 & \dots & x_n \\ y_1 & y_2 & y_3 & \dots & y_n \end{pmatrix}$$

Find the mean of each row.

$$\begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix}$$

Subtract it.

$$N = \begin{pmatrix} x_1 - \bar{x} & x_2 - \bar{x} & x_3 - \bar{x} & \dots & x_n - \bar{x} \\ y_1 - \bar{y} & y_2 - \bar{y} & y_3 - \bar{y} & \dots & y_n - \bar{y} \end{pmatrix}$$

Multiply $N * N^T$

You will get a 2x2 matrix, in which each entry is a summation over all n points. You could then divide by n

$$\sigma_{xx} = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})(x_i - \bar{x})$$

$$\sigma_{xy} = \sigma_{yx} = \frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})$$

$$\sigma_{yy} = \frac{1}{n} \sum_{i=1}^n (y_i - \bar{y})(y_i - \bar{y})$$

$$C = \begin{pmatrix} \sigma_{xx} & \sigma_{xy} \\ \sigma_{yx} & \sigma_{yy} \end{pmatrix}$$

Generic process

- The covariance matrix of a set of data gives the ways in which the set varies.
- The eigenvectors corresponding to the largest eigenvalues give the directions in which it varies most.
- Two applications
 - Eigenfaces
 - Time-elapsed photography

“Eigenfaces”

- Question: what are the primary ways in which faces vary?
- What happens when we apply PCA?
 - For each face, create a column vector that contains the intensity of **all the pixels** from that face
 - This is a point in a high dimensional space (e.g., 65536 for a 256x256 pixel image)
 - Create a matrix F of all M faces in the training set.
 - Subtract off the “average face”, μ , to get N
 - Compute the $rc \times rc$ covariance matrix $C = N \cdot N^T$.

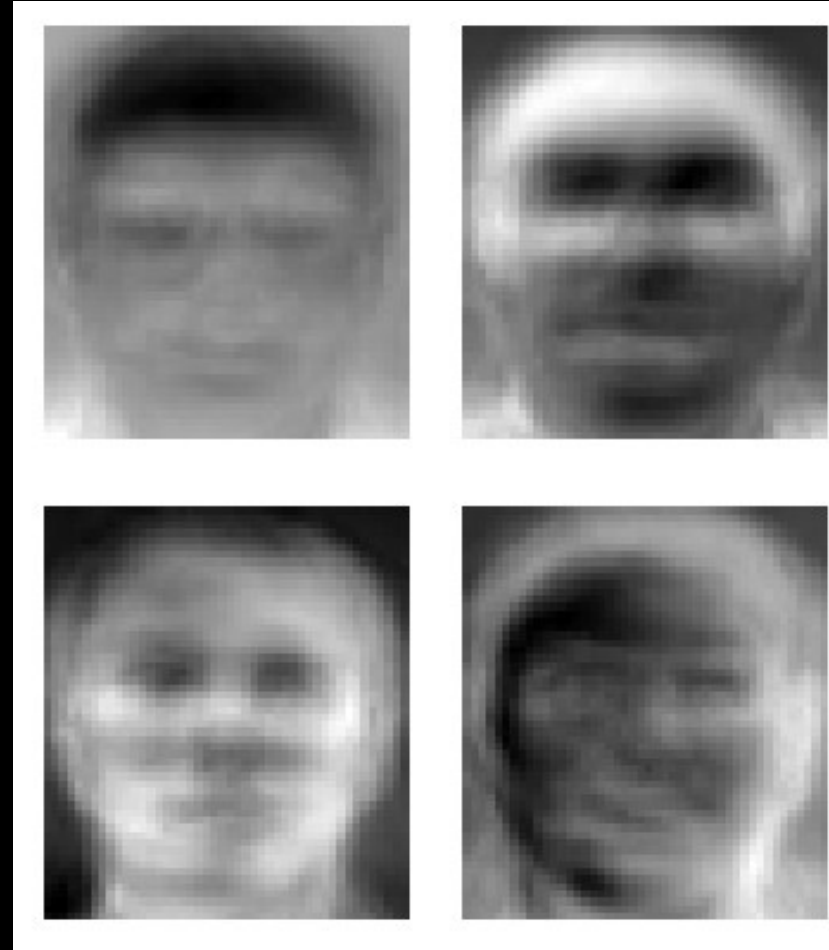
$$F = \begin{bmatrix} p_{11,1} & p_{11,2} & p_{11,3} & \cdots & p_{11,M} \\ p_{12,1} & p_{12,2} & p_{12,3} & \cdots & p_{12,M} \\ p_{13,1} & p_{13,2} & p_{13,3} & \cdots & p_{13,M} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{rc,1} & p_{rc,2} & p_{rc,3} & \cdots & p_{rc,M} \end{bmatrix}$$

“Eigenfaces”

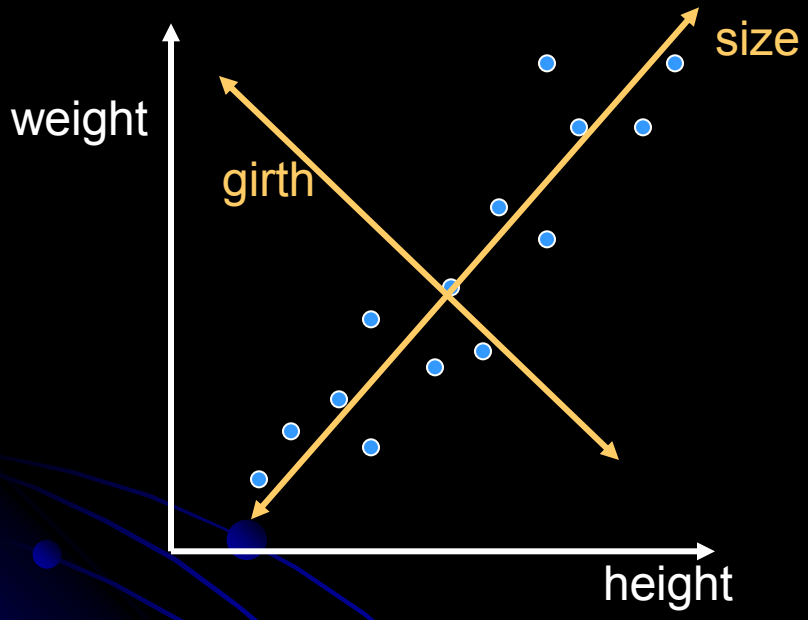
- Question: what are the primary ways in which faces vary?
- What happens when we apply PCA?
 - The eigenvectors are the directions of greatest variability
 - Note that these are in 65536-D; thus form a **face**.
 - This is an “**eigenface**”
 - **Here are the first 4 from the ORL face dataset.**

“Eigenfaces”

- Question: what are the primary ways in which faces vary?
- What happens when we apply PCA?
 - The eigenvectors are the directions of greatest variability
 - Note that these are in 65536-D; thus form a *face*.
 - This is an “*eigenface*”
 - ***Here are the first 4 from the ORL face dataset.***



Interlude: Projecting points onto lines



- We can project each point onto the principal axis. How?

Interlude: Projecting a point onto a line

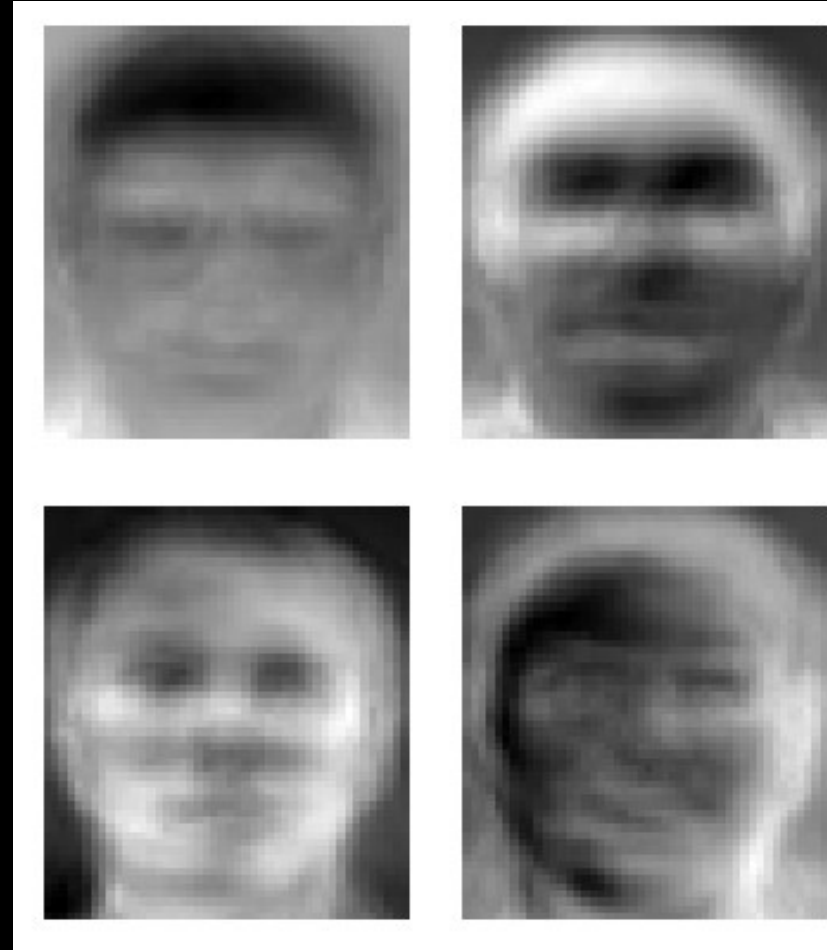
- Assuming the axis is represented by a unit vector \mathbf{u} , we can just take the dot-product of the point \mathbf{p} and the vector.
- $\mathbf{u}^* \mathbf{p} = \mathbf{u}^T \mathbf{p}$ (which is 1D)
- Example: Project (5,2) onto line $y=x$.
- If we want to project onto two vectors, \mathbf{u} and \mathbf{v} simultaneously:
- Create $\mathbf{w} = [\mathbf{u} \ \mathbf{v}]$, then compute $\mathbf{w}^T \mathbf{p}$, which is 2D.
 - Result: \mathbf{p} is now in terms of \mathbf{u} and \mathbf{v} .
 - This generalizes to arbitrary dimensions.

Application: Face detection

- If we want to project a point onto two vectors, u and v simultaneously:
- Create $w = [u \ v]$, then compute $w^T p$, **which is 2D.**
 - Result: p is now in terms of u and v .
 - In arbitrary dimensions, still **take the dot product with eigenvectors!**
- You can represent a face in terms of its eigenfaces; it's just a different basis.
- The M most important eigenvectors capture most of the variability:
 - Ignore the rest!
 - Instead of 65k dimensions, we only have M (~ 50 in practice)
 - Call these 50 dimensions "face-space"

“Eigenfaces”

- Question: what are the primary ways in which faces vary?
- What happens when we apply PCA?
 - Keep only the top M eigenfaces for “face space”.
 - We can project any face onto these eigenvectors. Thus, any face is a linear combination of the eigenfaces.
 - Can classify faces in this lower-D space.
 - There are computational tricks to make the computation feasible



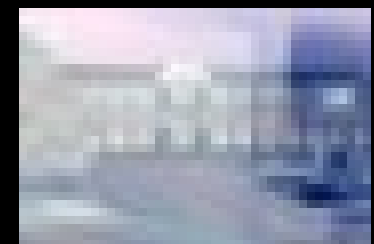
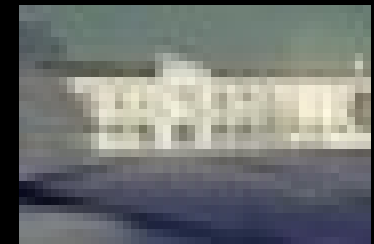
Time-elapsed photography

- Question: what are the ways that outdoor images vary over time?
- Form a matrix in which each column is an image
- Find eigs of covariance matrix

See example images on Dr. B's laptop or at the link below.

Time-elapseded photography

- Question: what are the ways that outdoor images vary over time?
- The mean and top 3 eigenvectors (scaled):
- Interpretation?



N Jacobs, N Roman, R Pless, Consistent Temporal Variations in Many Outdoor Scenes. *IEEE Computer Vision and Pattern Recognition*, Minneapolis, MN, June 2007.

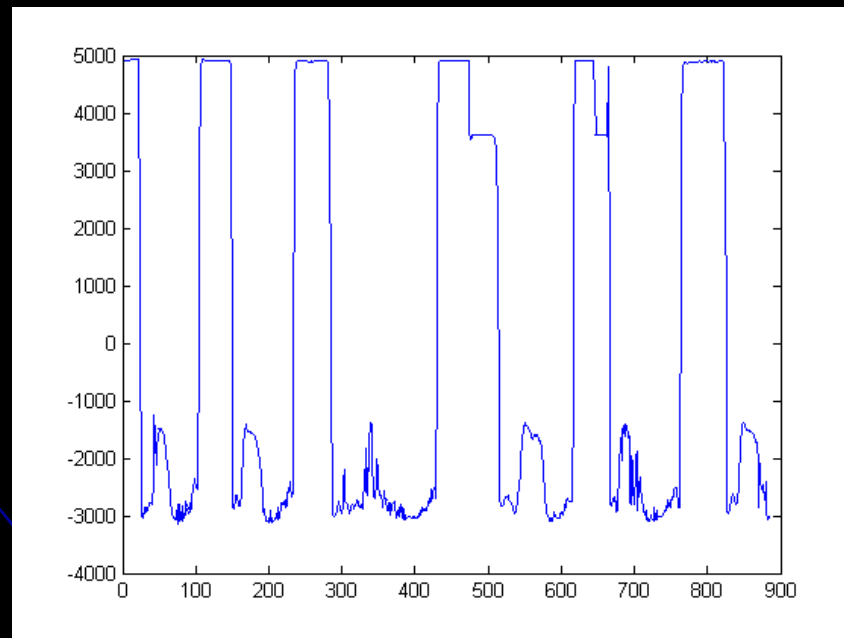
Time-elapseded photography

- Recall that each image in the dataset is a linear combination of the eigenimages.



Time-elapsed photography

- Every image's projection onto the first eigenvector



N Jacobs, N Roman, R Pless, Consistent Temporal Variations in Many Outdoor Scenes. *IEEE Computer Vision and Pattern Recognition*, Minneapolis, MN, June 2007.

Research idea

- Done:
 - Finding the PCs
 - Using to detect latitude and longitude given images from camera
 - Yet to do:
 - Classifying images based on their projection into this space, as was done for eigenfaces
- 