Cogsci 109

Virginia de Sa

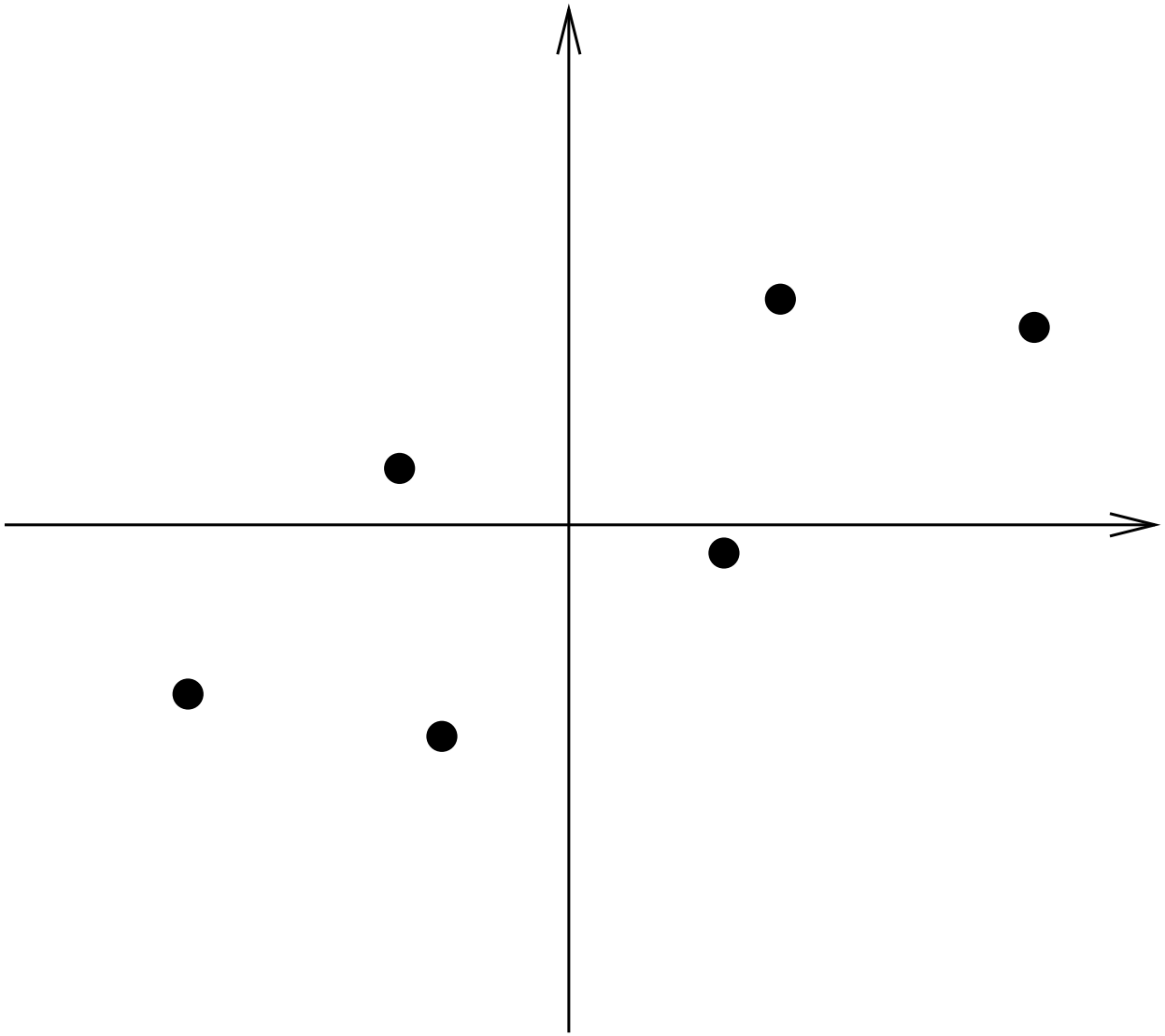
PCA

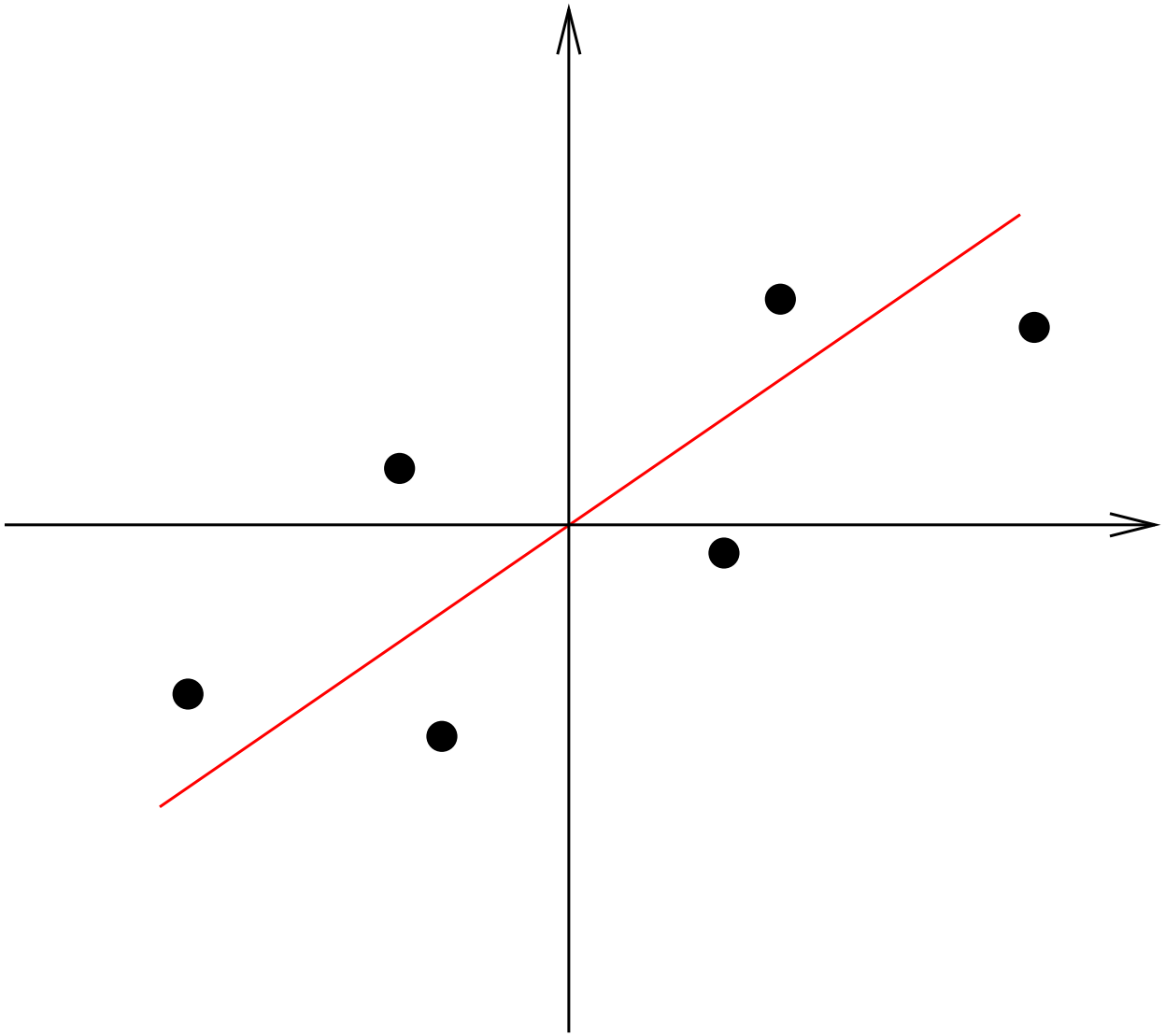
Principal Components Analysis

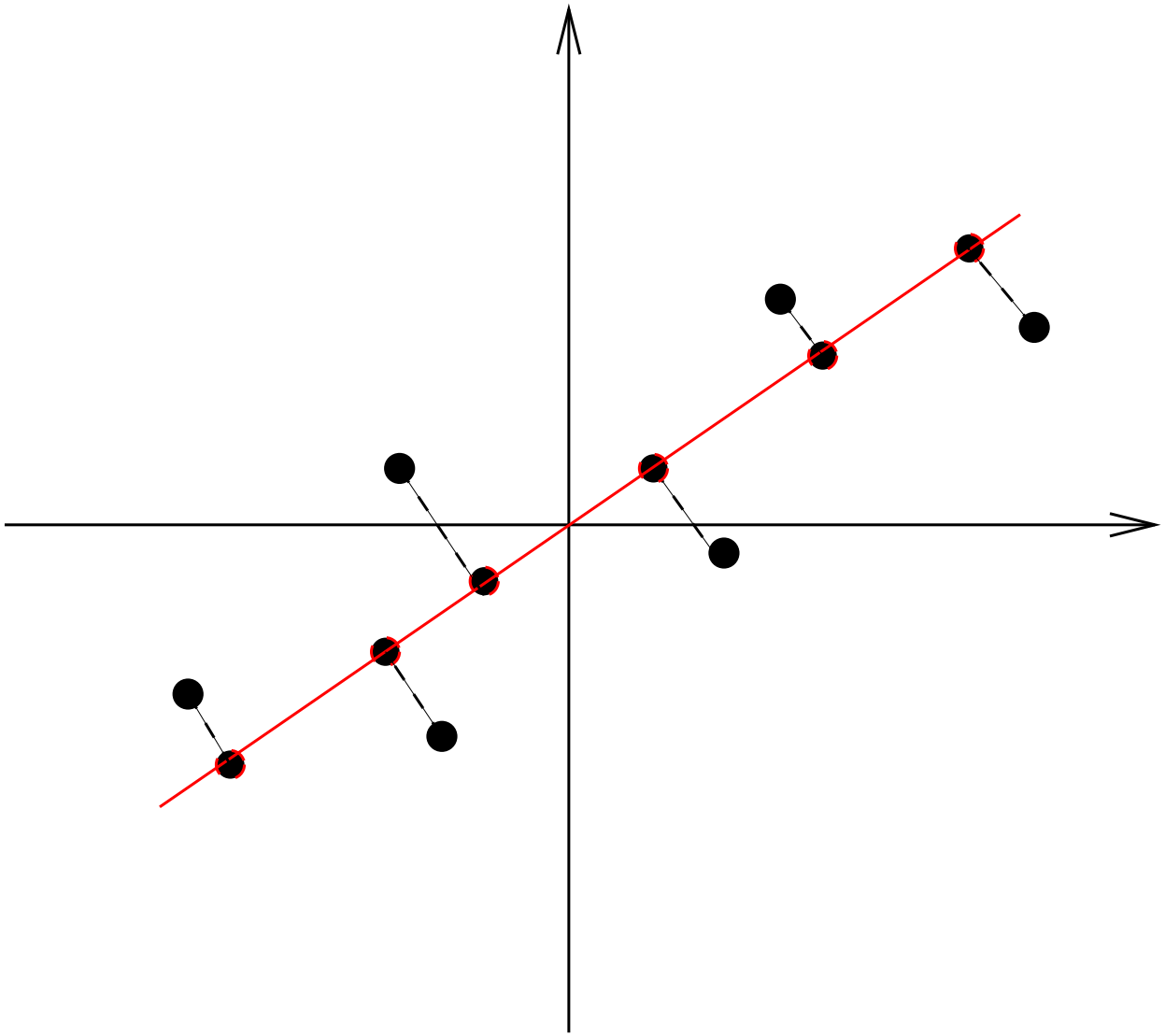
Principal Components Analysis (PCA) finds a linear subspace that contains most of the variance of the data. (PCA finds the linear subspace with the greatest projected variance.)

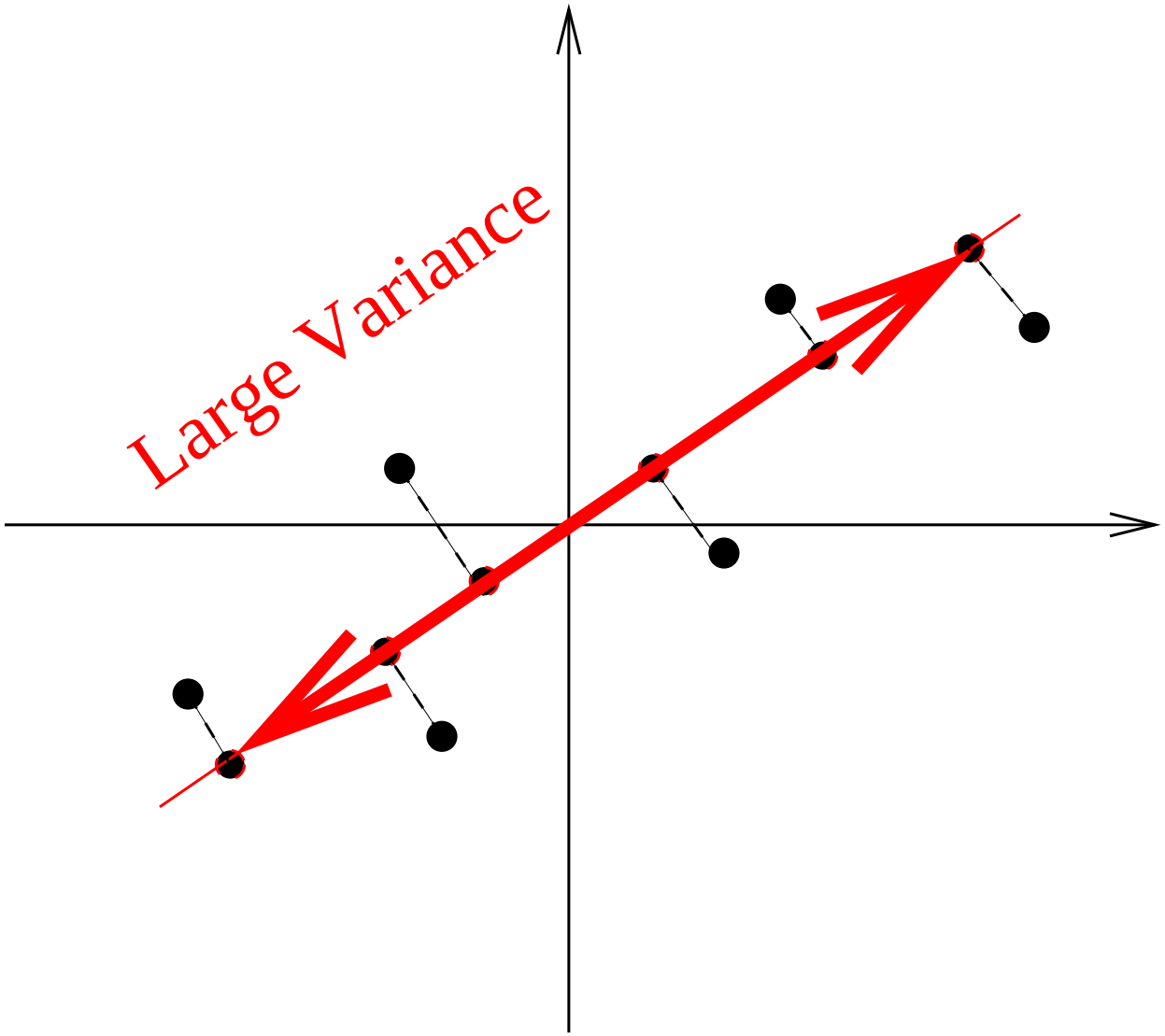
It is commonly used for dimensionality reduction – project to a reduced subspace that contains most of the variance.

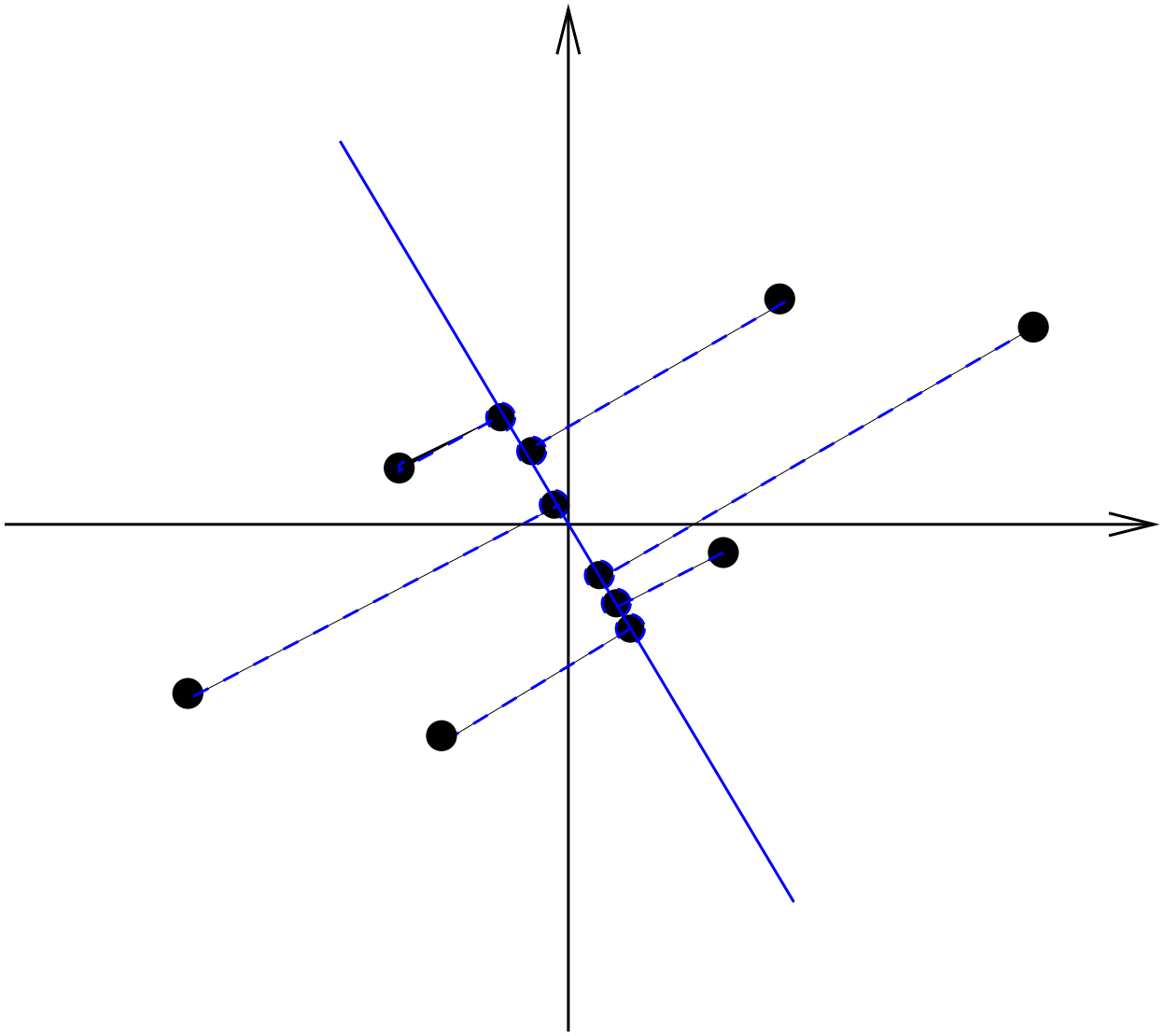
Once the desired dimensionality subspace is found, the data are represented as the projected point in that subspace.

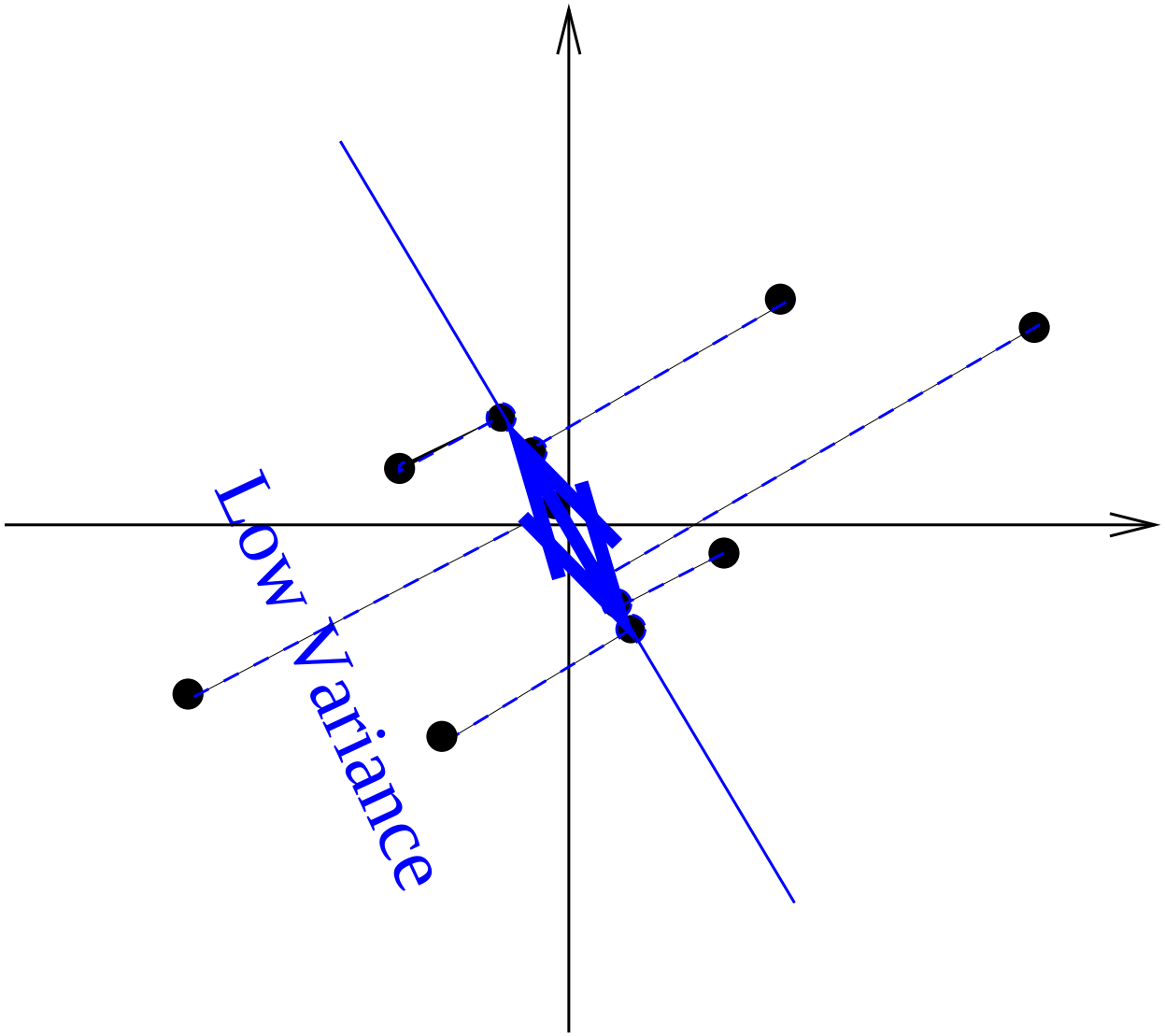







Large Variance




Low Variance

Random Vectors

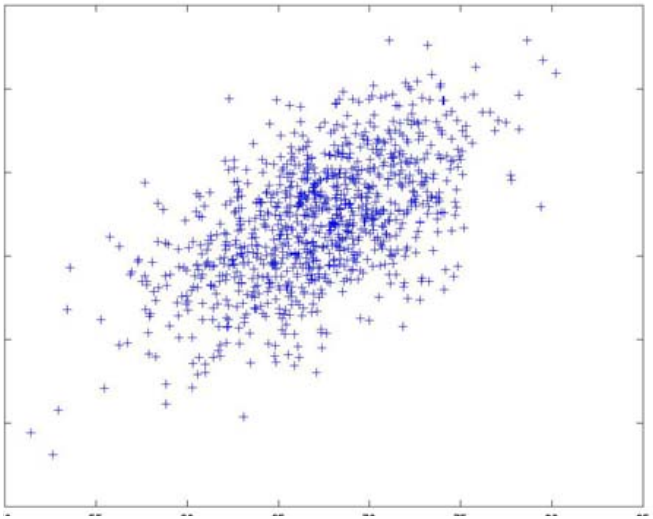
Consider asking a "random" person for her height and weight.

H and W are two random variables.

We can ask about their expected value (mean) and variance

We can also ask about how we expect them to covary.

We expect height and weight to be positively correlated.



Random Vectors

An n-dimensional **random vector** consists of n random variables associated with the same event (e.g. height and weight of individuals or height and wake-up time of individuals)

$$V = \begin{bmatrix} X \\ Y \end{bmatrix}$$

sample n times from V

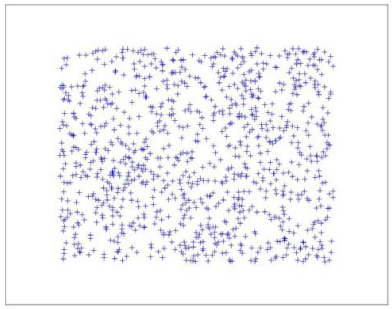
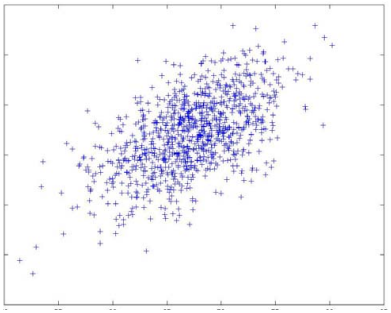
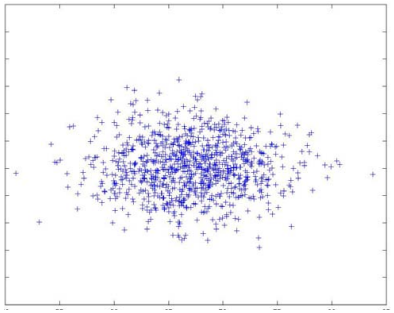
$$v_1 \quad v_2 \quad \ldots \quad v_n$$

$$\begin{bmatrix} x_1 & x_2 & \ldots & x_n \\ y_1 & y_2 & \ldots & y_n \end{bmatrix}$$

Random Vectors

What will the scatter plot (plot of samples) of V look like (plotting x dimension along x axis and y dimension along y axis)

- What will the scatter plot of V look like?

What do we expect the sample mean to be (this is a good estimate of the expected value of the random vector)?

What do we expect the sample variance in each dimension to be?

Covariance Matrices

These slides are from Jochen Triesch (my former co-teacher of a precursor course) based on slides created by Tim Marks (our TA of that course).

PCA

Principal Components Analysis (PCA) first finds the direction of greatest projected variance

We want to work in a space where the data have zero mean. This can easily be done by setting $\mathbf{z} = \mathbf{x} - \mathbf{m}$

Consider a projection vector (direction) $\mathbf{w}$ If y is the length of the projection of $\mathbf{x}$ onto the vector $\mathbf{w}$

Let's take $\mathbf{w}$ to have unit length ($||\mathbf{w}|| = 1$)

$$\begin{aligned} y &= \mathbf{w} \cdot \mathbf{z} \\ &= \sum_{i=1}^{n} w_i z_i \\ &= ||\mathbf{w}||||\mathbf{z}||cos\theta \end{aligned}$$

PCA finds the direction $\mathbf{w}$ for which the variance of the projection of the data points is maximal

$$E(y) = E(\sum_i w_i z_i) = \sum_i E(w_i z_i) = \sum_i w_i E(z_i) = 0$$

$$\begin{aligned} Var(y) &= E((y - E(y))^2) \\ &= E(y^2) \\ &= E((\mathbf{w^T z})^2) \\ &= E((\mathbf{w}^T\mathbf{z})(\mathbf{z}^T\mathbf{w})) \\ &= \mathbf{w}^T E(\mathbf{z}\mathbf{z}^T)\mathbf{w} \\ &= \mathbf{w}^T C_z \mathbf{w} \end{aligned}$$

where C_z is the covariance matrix of $\mathbf{z}$ (and the covariance matrix of $\mathbf{x}$)

$$C_z = 1/N \sum_{n=1}^{N} (\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})'$$

The unit length vector $\mathbf{w}$ that maximizes $Var(y)$ is the eigenvector of C_z corresponding to the largest eigenvalue.

PCA

The unit length vector $\mathbf{w}$ that maximizes $Var(y)$ is the eigenvector of C_z corresponding to the largest eigenvalue.

This can be proved using Lagrange multipliers (which we do in 118B) but we can also see more intuitively why this is true

First note that if **w** is a unit length eigenvector of C_z with eigenvalue λ then

$$Var(y) = \mathbf{w}^T C_z \mathbf{w} = \mathbf{w}^T \lambda \mathbf{w} = \lambda$$

Clicker Question

$$Var(y) = \mathbf{w^T C_z w} = \mathbf{w^T} \lambda \mathbf{w} = \lambda$$

What assumption was necessary to go from the red to the blue above

- A : w is an eigenvector of C_z
- B : w is unit length
- C : w has eigenvalue λ
- D : B and C
- E : A and C

Clicker Question

$$Var(y) = \mathbf{w^T C_z w} = \mathbf{w^T} \lambda \mathbf{w} = \lambda$$

What assumption was necessary to go from the blue to the magenta above

- A : w is an eigenvector of C_z
- B : w is unit length
- C : w has eigenvalue λ
- D : B and C
- E : A and C

PCA

We are trying to show that: The unit length vector $\mathbf{w}$ that maximizes $Var(y)$ is the eigenvector of C_z corresponding to the largest eigenvalue.

We have showed that if $\mathbf{w}$ is a unit length eigenvector of C_z with eigenvalue λ then

$$Var(y) = \lambda$$

To continue, we need to remember that Covariance matrices are symmetric (Why?) and that real-valued symmetric matrices have a set of perpendicular eigenvectors (which can be normalized to unit length).

PCA

If $\mathbf{w}$ is not an eigenvector then it can be written as a linear combination of the perpendicular unit length eigenvectors. That is

$$\mathbf{w} = a_1\mathbf{w_1} + a_2\mathbf{w_2} + ...a_n\mathbf{w_n}$$

where $\mathbf{w_i}$ are the eigenvectors of C_z and $\mathbf{w_1}$ has the largest eigenvalue λ_1.

Since $\mathbf{w}$ has unit length, we know that

$$\mathbf{w}^T\mathbf{w} = 1$$

PCA

Since $\mathbf{w}$ has unit length, we know that

$$\mathbf{w}^T\mathbf{w} = 1$$

which means

$$(a_1\mathbf{w_1} + a_2\mathbf{w_2} + ...a_n\mathbf{w_n})^T(a_1\mathbf{w_1} + a_2\mathbf{w_2} + ...a_n\mathbf{w_n}) = 1]$$

$$a_1^2{\mathbf{w_1}}^T\mathbf{w_1} + a_1a_2{\mathbf{w_1}}^T\mathbf{w_2} + ... = 1$$

$$a_1^2 + a_2^2 + a_3^2 + ...a_n^2 = 1$$

Note we have used the fact that $\mathbf{w_i}$ is orthogonal to $\mathbf{w_j}$ for all $i \neq j$ to get the last equation from the previous one.

PCA

Now

$$
\begin{aligned}
Var(y) &= \mathbf{w}^T C_z \mathbf{w} \\
&= (a_1\mathbf{w_1} + a_2\mathbf{w_2} + ...)^T C_z (a_1\mathbf{w_1} + a_2\mathbf{w_2} + ...) \\
&= (a_1\mathbf{w_1} + a_2\mathbf{w_2} + ...)^T (a_1\lambda_1\mathbf{w_1} + a_2\lambda_2\mathbf{w_2} + ...a_n\lambda_n\mathbf{w_n}) \\
&= a_1^2\lambda_1 \mathbf{w_1}^T\mathbf{w_1} + a_1a_2\lambda_2\mathbf{w_1}^T\mathbf{w_2} + ... + a_2^2\lambda_2\mathbf{w_2}^T\mathbf{w_2} + ... \\
&= a_1^2\lambda_1 + a_2^2\lambda_2 + ... + a_n^2\lambda_n \\
&\leq a_1^2\lambda_1 + a_2^2\lambda_1 + ...a_n^2\lambda_1 \\
&= \lambda_1
\end{aligned}
$$

So the unit length vector $\mathbf{w}$ that maximizes the projected variance ($Var(y)$ is the eigenvector with the largest eigenvalue λ_1.

PCA overview

1-D PCA projects the data onto the eigenvector with the largest eigenvalue.

In general PCA projects the data onto the subspace spanned by the n eigenvectors with the largest eigenvalues.

Fun things you can do with PCA

file:/Users/desa/classes/108c/siggraph99.mpg

http://gravis.cs.unibas.ch/Sigg99.html

PCA overview

- Zero mean the data (compute mean and subtract from all the data) $\mathbf{z}^{(i)} = \mathbf{x}^{(\mathbf{i})} - \mathbf{m}$
- Compute the covariance matrix and find (and normalize) it's eigenvectors
- Sort the eigenvectors in order of decreasing eigenvalue (and put as columns in V)
- Compute principal components $\mathbf{c}^{(\mathbf{i})} = V'\mathbf{z}^{(\mathbf{i})}$ $(c_j^{(i)} = v_j \cdot z^{(i)})$
- Reconstruct zero-meaned pattern using $\hat{\mathbf{z}}^{(\mathbf{i})} = V_{reduced}\mathbf{c}^{(\mathbf{i})}{}_{reduced}$
- $\hat{\mathbf{x}}^{(\mathbf{i})} = \hat{\mathbf{z}}^{(\mathbf{i})} + \mathbf{m}$ (add back in the mean)

PCA step by step

Zero mean the data (compute mean and subtract from all the data)
$\mathbf{z}^{(i)} = \mathbf{x}^{(\mathbf{i})} - \mathbf{m}$

(X is matrix of column vectors $x^{(i)}$, Z is matrix of mean subtracted data) assume we have n data vectors

```
>>m =1/n*sum(X,2);
>>Z=X-repmat(m,1,n);
```

PCA step by step

Compute the covariance matrix and find (and normalize) it's eigenvectors

```
>>C = 1/n* Z* Z';
>>[Vp,Dp]=eig(C);
>>[V,D]=eigsort(Vp,Dp);  (we will provide you with eigsort)
```

Remember Z is the matrix consisting of columns of mean-subtracted data

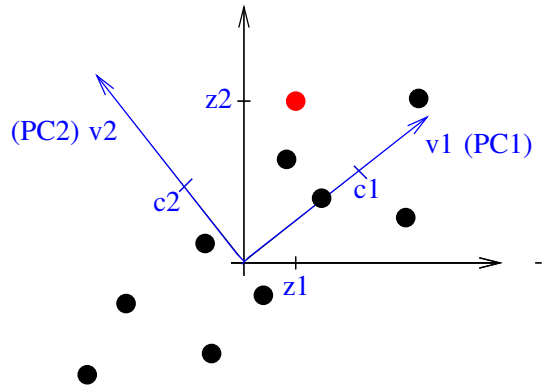
PCA step by step

Compute principal components $\mathbf{c^{(i)}} = V'\mathbf{z^{(i)}}$ $(c_j^{(i)} = v_j \cdot z^{(i)})$

– PCA transforms the point $\boldsymbol{x}$ (original coordinates) into the point $\boldsymbol{c}$ (new coordinates).

- by subtracting the mean: $\mathbf{z} = \boldsymbol{x} - \boldsymbol{m}$
- and multiplying the result by V^{T}

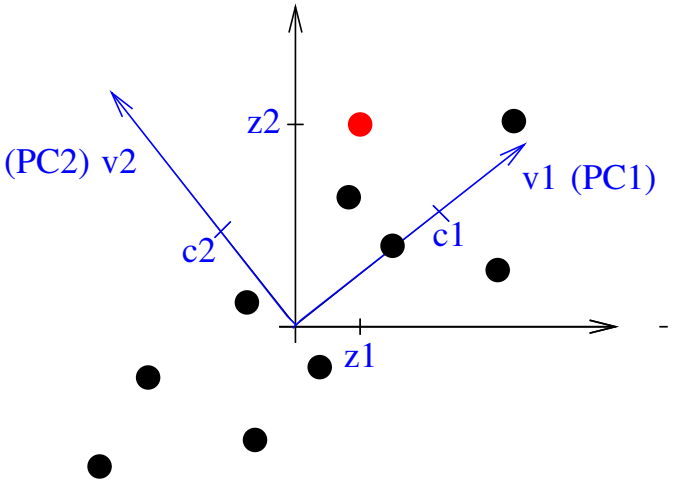
$$\underset{V^{\mathrm{T}}}{\begin{bmatrix} \leftarrow & \mathbf{v_1} & \rightarrow \\ \leftarrow & \mathbf{v}_2 & \rightarrow \\ & \vdots & \\ \leftarrow & \mathbf{v}_D & \rightarrow \end{bmatrix}} \underset{\mathbf{z}}{\begin{bmatrix} \uparrow \\ \mathbf{z} \\ \downarrow \end{bmatrix}} = \underset{\mathbf{c}}{\begin{bmatrix} \mathbf{v_1} \cdot \mathbf{z} \\ \mathbf{v}_2 \cdot \mathbf{z} \\ \vdots \\ \mathbf{v}_D \cdot \mathbf{z} \end{bmatrix}}$$



- Can think of as a rotation because V^T is an orthogonal matrix
- Can think of as projection of $\mathbf{z^{(i)}}$ onto PC axes, because $\mathbf{v_1} \cdot \mathbf{z^{(i)}}$ is projection of $\mathbf{z^{(i)}}$ onto PC1 axis, $\mathbf{v_2} \cdot \mathbf{z^{(i)}}$ is projection onto PC2 axis etc...

```
>>z=Z(:,i)
>>c=V'*z;
```

Clicker question



What does the c vector represent?

A) The covariance matrix

B) The top eigenvector of the Covariance matrix

C) The coordinates of the zero-meaned data point in the Principal components space

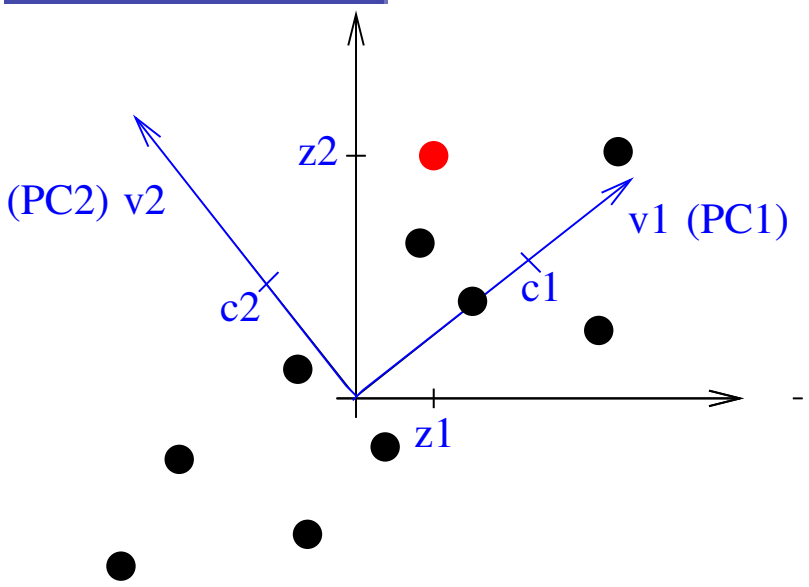
PCA step by step

Reconstruct zero-meaned pattern using $\hat{\mathbf{z}}^{(\mathbf{i})} = V_{reduced}\mathbf{c}^{(\mathbf{i})}{}_{reduced}$

- PCA expresses the mean-subtracted point, $\mathbf{z} = \boldsymbol{x} - \boldsymbol{m}$, as a weighted sum of the eigenvectors $\mathbf{v}_i$:

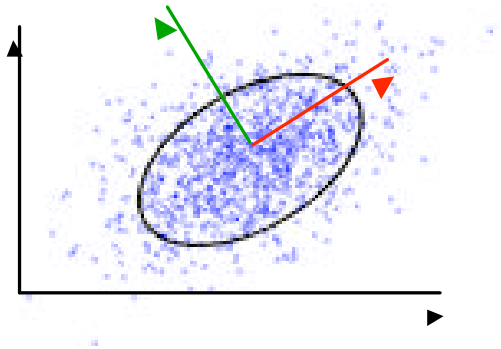
$$\mathbf{z} = V\,\mathbf{c}$$

$$\begin{bmatrix} \uparrow \\ \mathbf{z} \\ \downarrow \end{bmatrix} = \begin{bmatrix} \uparrow & \uparrow & & \uparrow \\ \mathbf{v_1} & \mathbf{v}_2 & \cdots & \mathbf{v}_D \\ \downarrow & \downarrow & & \downarrow \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_D \end{bmatrix} = c_1 \begin{bmatrix} \uparrow \\ \mathbf{v_1} \\ \downarrow \end{bmatrix} + c_2 \begin{bmatrix} \uparrow \\ \mathbf{v}_2 \\ \downarrow \end{bmatrix} + \cdots + c_D \begin{bmatrix} \uparrow \\ \mathbf{v}_D \\ \downarrow \end{bmatrix}$$



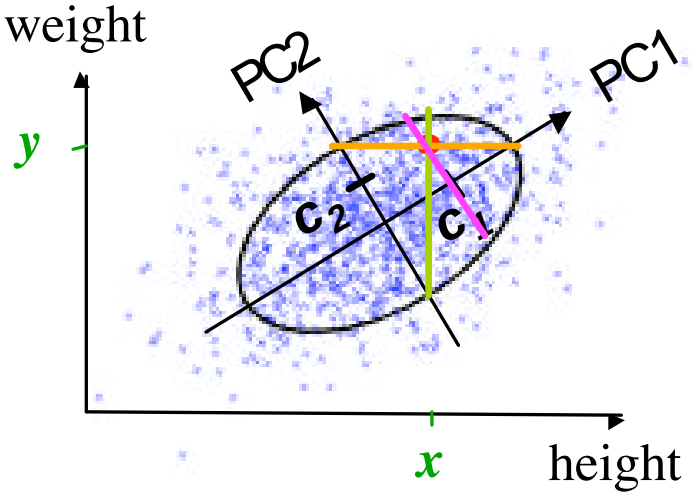
Eigenvalues and variance

- The eigenvectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n$ of the covariance matrix have corresponding eigenvalues $\lambda_1, \lambda_1, \ldots, \lambda_n$.
  - It turns out that λ_1 is the variance of the distribution in the $\boldsymbol{v}_1$ direction, λ_2 is the variance of the distribution in the $\boldsymbol{v}_2$ direction, and so on.
    - The largest eigenvalue corresponds to the principal component in the direction of greatest variance, the next largest eigenvalue corresponds to the principal component in the perpendicular direction of next greatest variance, etc.
- Which eigenvector (green or red) corresponds to the smaller eigenvalue?



PCA for data compression

– What if you wanted to transmit someone's height and weight, but you could only give a single number?
  - Could give only height, x
    — = (uncertainty when height is known)
  - Could give only weight, y
    — = (uncertainty when weight is known)
  - Could give only c_1, the value of first PC
    — = (uncertainty when first PC is known)
    – Giving the first PC minimizes the squared error of the result.



– To compress D-dimensional data into k dimensions, order the principal components in order of largest-to-smallest eigenvalue, and only save the first k components.

PCA step by step

Reconstruct zero-meaned pattern using $\hat{\mathbf{z}}^{(\mathbf{i})} = V_{reduced}\mathbf{c}^{(\mathbf{i})}{}_{reduced}$

- PCA *approximates* the mean-subtracted point, $\mathbf{z} = \boldsymbol{x} - \boldsymbol{m}$, as a weighted sum of the first k eigenvectors:

$$\overline{\mathbf{z}} = \overline{V}\,\overline{\mathbf{c}}$$

$$\begin{bmatrix} \uparrow \\ \overline{\mathbf{z}} \\ \downarrow \end{bmatrix} = \begin{bmatrix} \uparrow & \uparrow & & \uparrow \\ \mathbf{v_1} & \mathbf{v}_2 & \cdots & \mathbf{v}_k \\ \downarrow & \downarrow & & \downarrow \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \end{bmatrix} = c_1 \begin{bmatrix} \uparrow \\ \mathbf{v_1} \\ \downarrow \end{bmatrix} + c_2 \begin{bmatrix} \uparrow \\ \mathbf{v}_2 \\ \downarrow \end{bmatrix} + \cdots + c_k \begin{bmatrix} \uparrow \\ \mathbf{v}_k \\ \downarrow \end{bmatrix}$$

```
>>  zhat= V(:,1:k)*c(1:k)
```

Clicker question

Reconstruct zero-meaned pattern using $\hat{\mathbf{z}}^{(\mathbf{i})} = V_{reduced}\mathbf{c}^{(\mathbf{i})}{}_{reduced}$

- PCA *approximates* the mean-subtracted point, $\mathbf{z} = \boldsymbol{x} - \boldsymbol{m}$, as a weighted sum of the first k eigenvectors:

$$\overline{\mathbf{z}} = \overline{V}\,\overline{\mathbf{c}}$$

$$\begin{bmatrix} \uparrow \\ \overline{\mathbf{z}} \\ \downarrow \end{bmatrix} = \begin{bmatrix} \uparrow & \uparrow & & \uparrow \\ \mathbf{v_1} & \mathbf{v}_2 & \cdots & \mathbf{v}_k \\ \downarrow & \downarrow & & \downarrow \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \end{bmatrix} = c_1 \begin{bmatrix} \uparrow \\ \mathbf{v_1} \\ \downarrow \end{bmatrix} + c_2 \begin{bmatrix} \uparrow \\ \mathbf{v}_2 \\ \downarrow \end{bmatrix} + \cdots + c_k \begin{bmatrix} \uparrow \\ \mathbf{v}_k \\ \downarrow \end{bmatrix}$$

What would happen if you used all D eigenvectors ?

A) You would get an approximation to the zero-meaned pattern

B) You would perfectly reconstruct the zero-meaned pattern

PCA step by step

$\hat{\mathbf{x}}^{(\mathbf{i})} = \hat{\mathbf{z}}^{(\mathbf{i})} + \mathbf{m}$ (add back in the mean)

```
>>xhat=zhat + m;
```

PCA overview

- Zero mean the data (compute mean and subtract from all the data) $\mathbf{z}^{(i)} = \mathbf{x}^{(\mathbf{i})} - \mathbf{m}$
- Compute the covariance matrix and find (and normalize) it's eigenvectors
- Sort the eigenvectors in order of decreasing eigenvalue (and put as columns in V)
- Compute principal components $\mathbf{c}^{(\mathbf{i})} = V'\mathbf{z}^{(\mathbf{i})}$ $(c_j^{(i)} = v_j \cdot z^{(i)})$
- Reconstruct zero-meaned pattern using $\hat{\mathbf{z}}^{(\mathbf{i})} = V_{reduced}\mathbf{c}^{(\mathbf{i})}{}_{reduced}$
- $\hat{\mathbf{x}}^{(\mathbf{i})} = \hat{\mathbf{z}}^{(\mathbf{i})} + \mathbf{m}$ (add back in the mean)

Notes on PCA

In the previous slides we went through the steps of PCA for reducing the dimensionality for one of our datapoints ($x * (i)$) We can actually do the same thing for all data points at once using matrix operations.

- Zero mean the data (compute mean and subtract from all the data) $Z = X - repmat(m, 1, n)$
- Compute the covariance matrix and find (and normalize) it's eigenvectors
- Sort the eigenvectors in order of decreasing eigenvalue (and put as columns in V)
- Compute principal components $C = V' * Z$
- Reconstruct zero-meaned pattern using $\hat{Z} = V_{reduced}\mathbf{C_{reduced}}$
  $\hat{Z} = V(:, 1:k) * C(1:k, :)$
- $\hat{\mathbf{X}} = \hat{\mathbf{Z}} + repmat(\mathbf{m}, 1, n)$ (add back in the mean)

Notes on PCA

Note that the V matrix is orthogonal (as the eigenvectors are orthogonal and unit length)

$$\begin{bmatrix} \uparrow & \uparrow & & \uparrow \\ \mathbf{v}^{(1)} & \mathbf{v}^{(2)} & \cdots & \mathbf{v}^{(D)} \\ \downarrow & \downarrow & & \downarrow \end{bmatrix}$$

PCA minimizes square reconstruction cost

We have derived PCA as finding the subspace of maximal projected variance.

It can also be derived as finding the subspace for which there is minimal squared error (between data before and after projection)

$$\sum_i ||\mathbf{x}^{(i)} - (\mathbf{x}^{(i)^T}\mathbf{w})\mathbf{w}||^2$$

Properties of Covariance Matrices

- Covariance matrices are positive semidefinite

  $\mathbf{v}^T A \mathbf{v} \geq 0 \forall \mathbf{v}$

- all the eigenvalues are real and non-negative

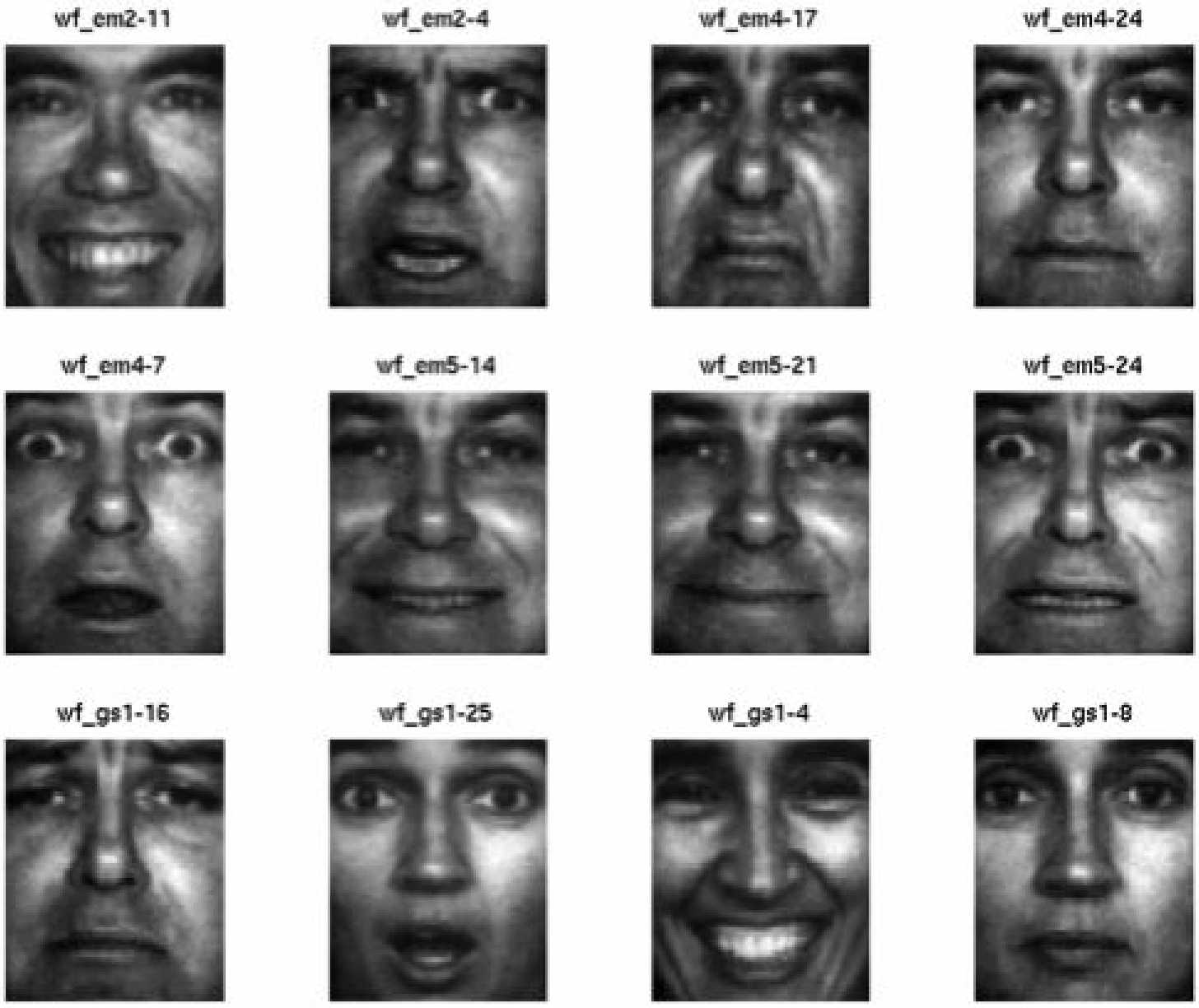
- the eigenvectors can be chosen to be mutually orthonormal

PCA on faces

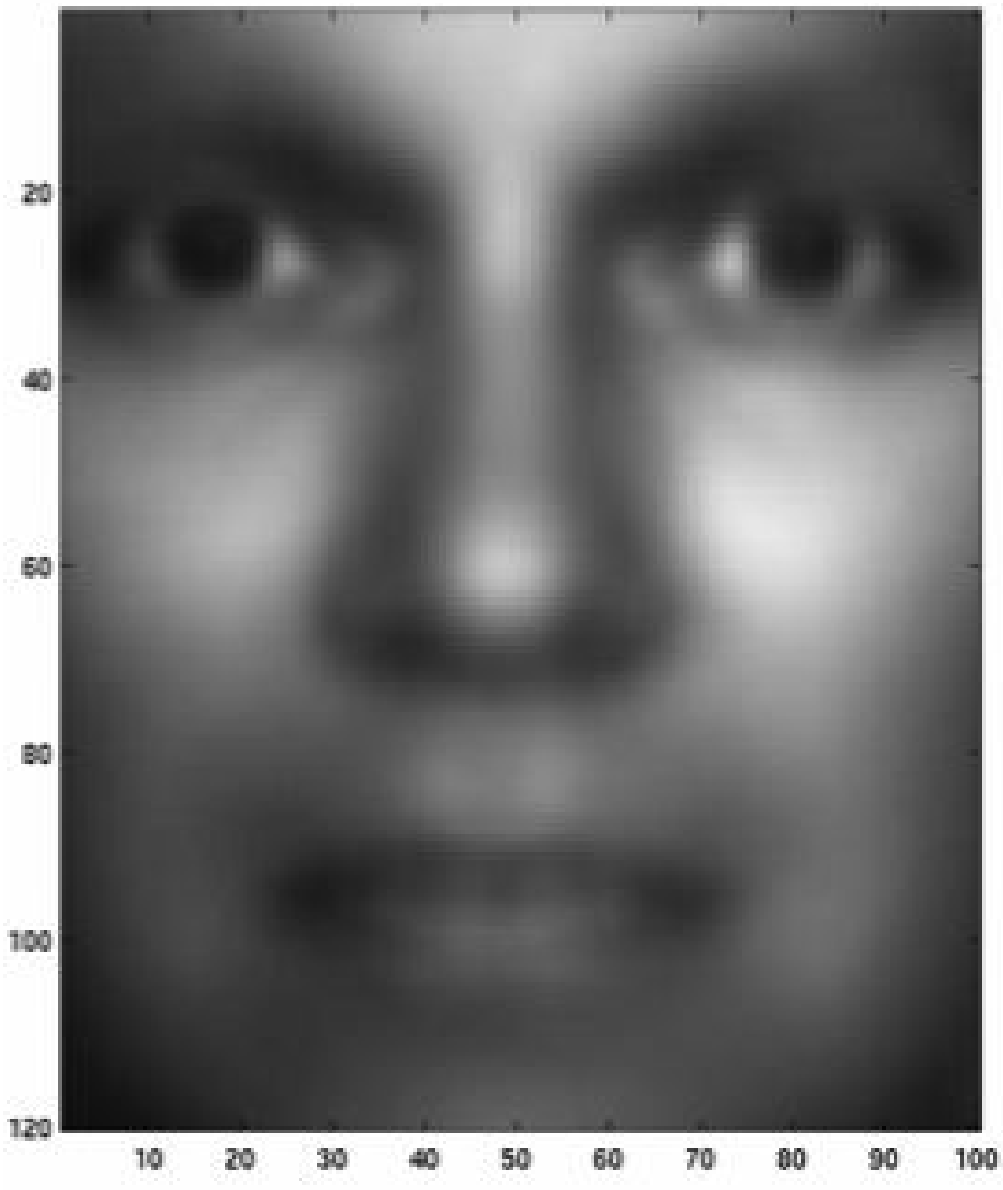
We have 120 by 100 pixel grayscale images of 97 faces.

Each image is a 12000 dimensional vector

Example Faces



Mean Face



PCA on face images

- subtract mean face
- construct Data matrix Z where columns are mean-subtracted faces
- construct Covariance matrix $\frac{1}{n}ZZ^T$
- find eigenvectors of $\frac{1}{n}ZZ^T$ (or equivalently the eigenvectors of ZZ^T since dropping the $\frac{1}{n}$ does not change the eigenvectors but just changes the eigenvalues (which are not used) by a factor of n)

ZZ^T is 12000 by 12000 dimensions (very large) so we use a trick to compute the eigenvectors

The transpose trick

on board

The transpose trick

The eigenvectors of the covariance matrix of face space are called eigenfaces (eigenvectors of ZZ^T)

When the dimensionality of the input space (12000 in our case) is much larger than the number of patterns (97 in our case) then ZZ^T is much larger (12000x12000 for us) than Z^TZ (97x97 for us) and we can get the eigenvectors of ZZ^T by computing them from Z^TZ and then multiplying the resultant eigenvectors by Z.

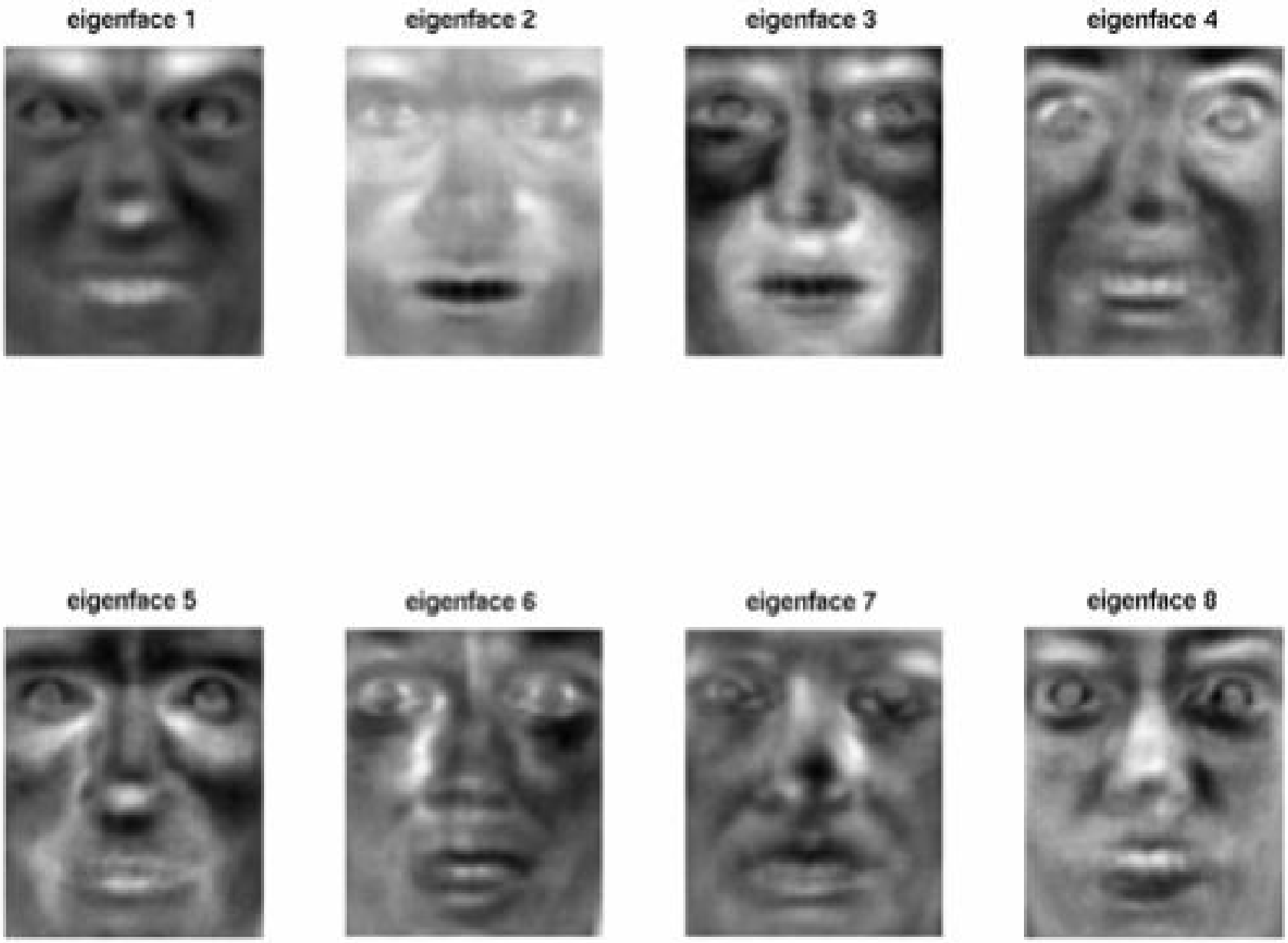
Since if v is an eigenvector of Z^TZ with eigenvalue λ then

$$\begin{aligned} Z^TZv &= \lambda v \\ ZZ^TZv &= \lambda Zv \\ ZZ^T(Zv) &= \lambda(Zv) \end{aligned}$$

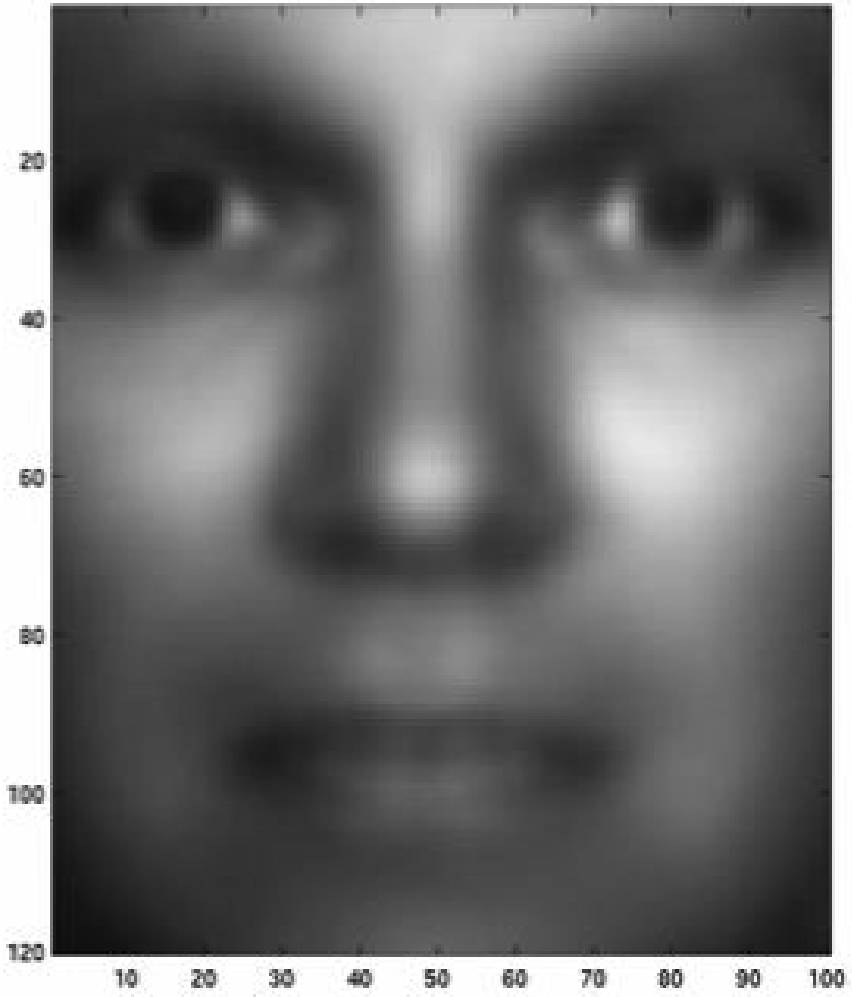
and thus Zv is an eigenvector of ZZ^T with eigenvalue λ (the same eigenvalue).

Thus Z times the top (ones associated with the largest eigenvalues) eigenvectors of $Z^T Z$ give the top eigenvectors of ZZ^T.

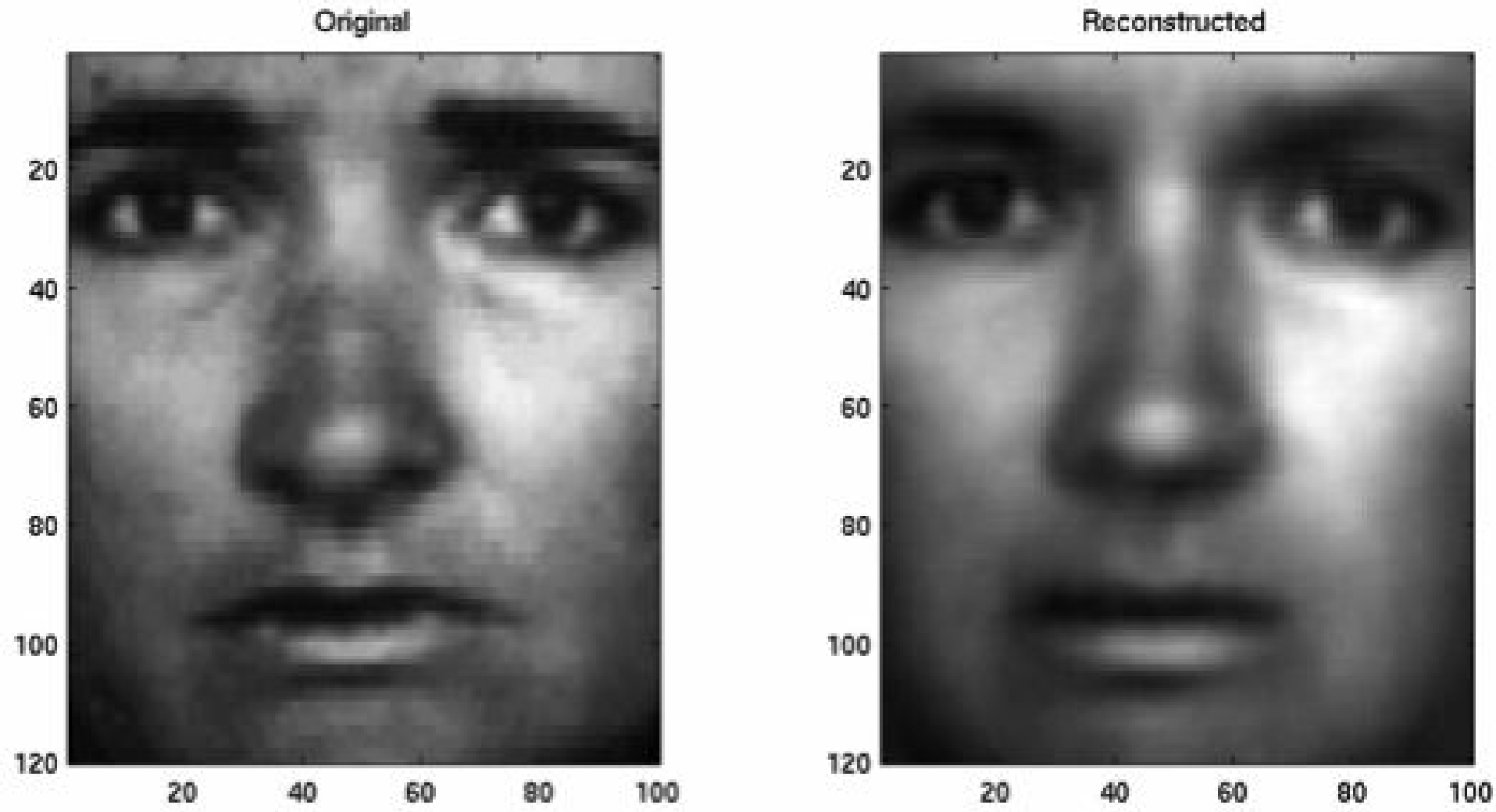
The first eight eigenfaces



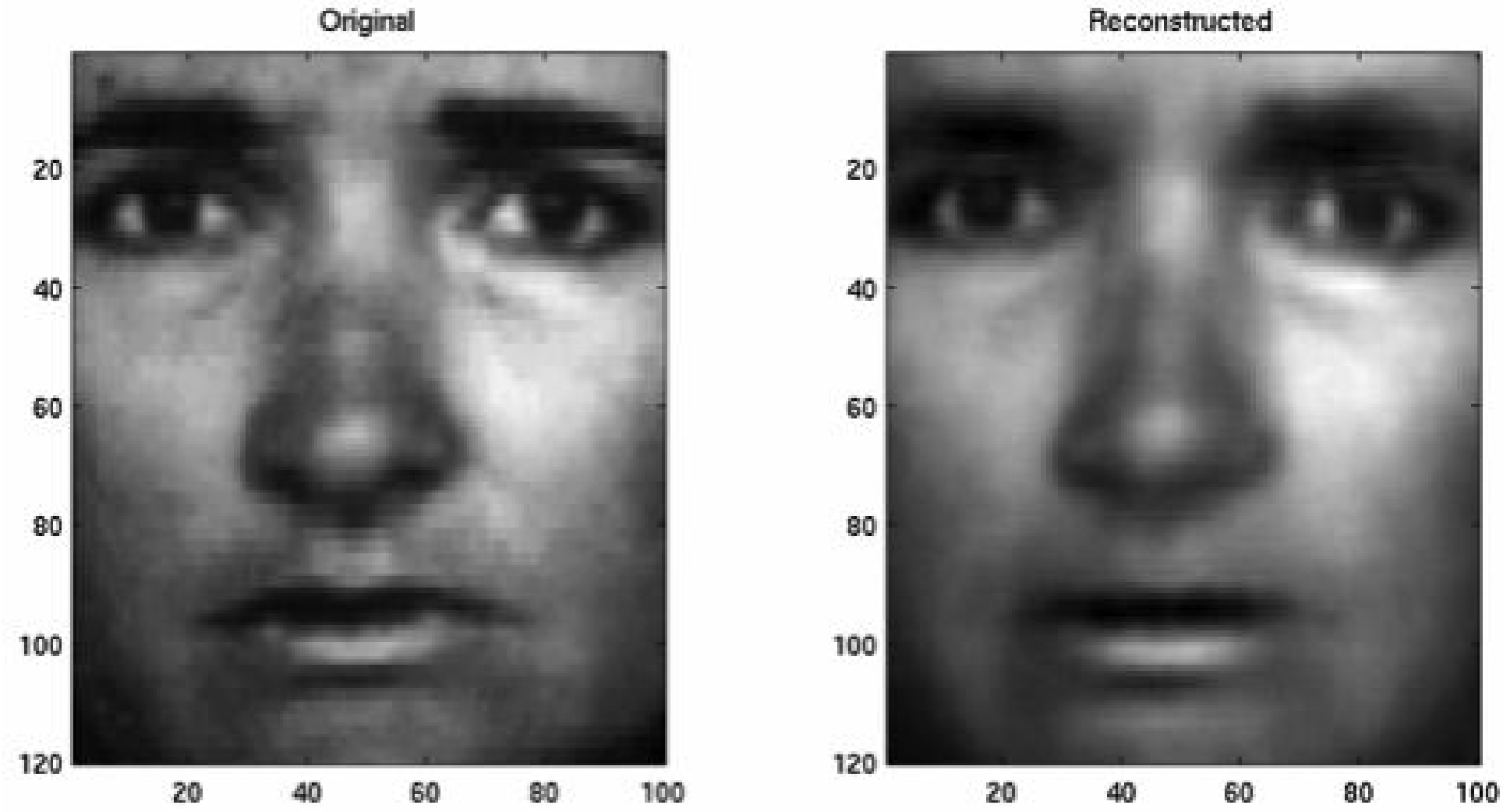
Reconstructing a face (with the mean face)



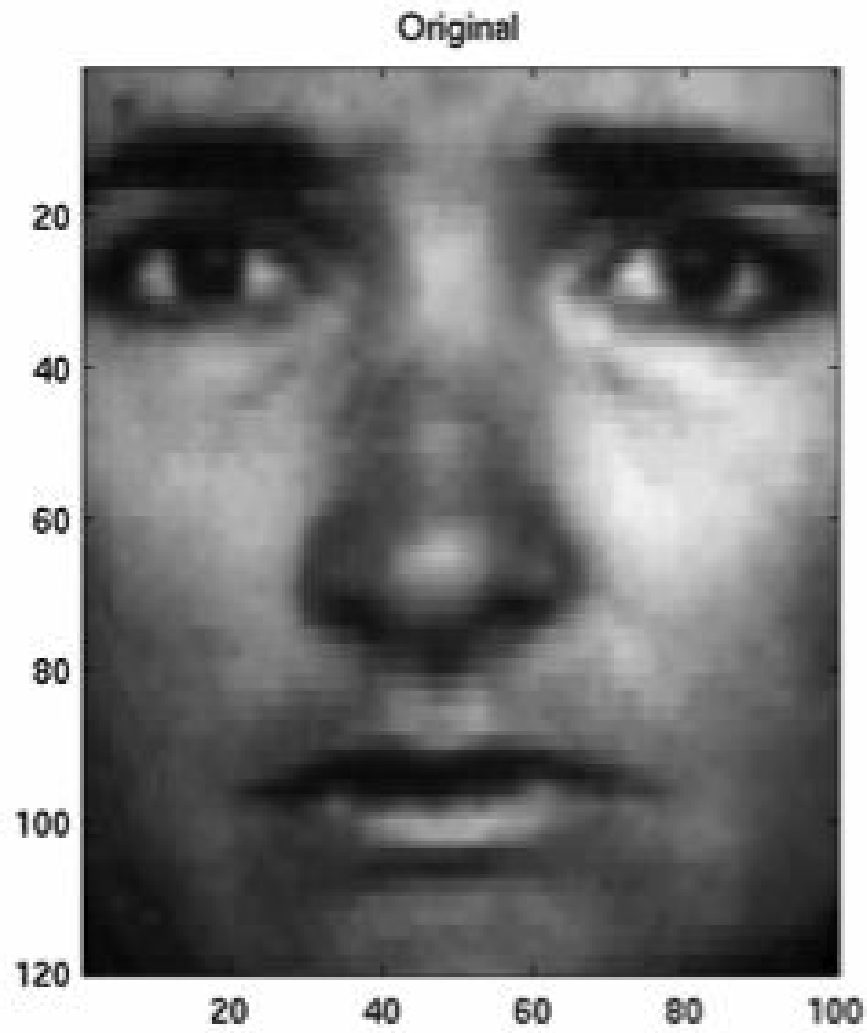
Reconstructing a face (with first 10 eigenfaces)

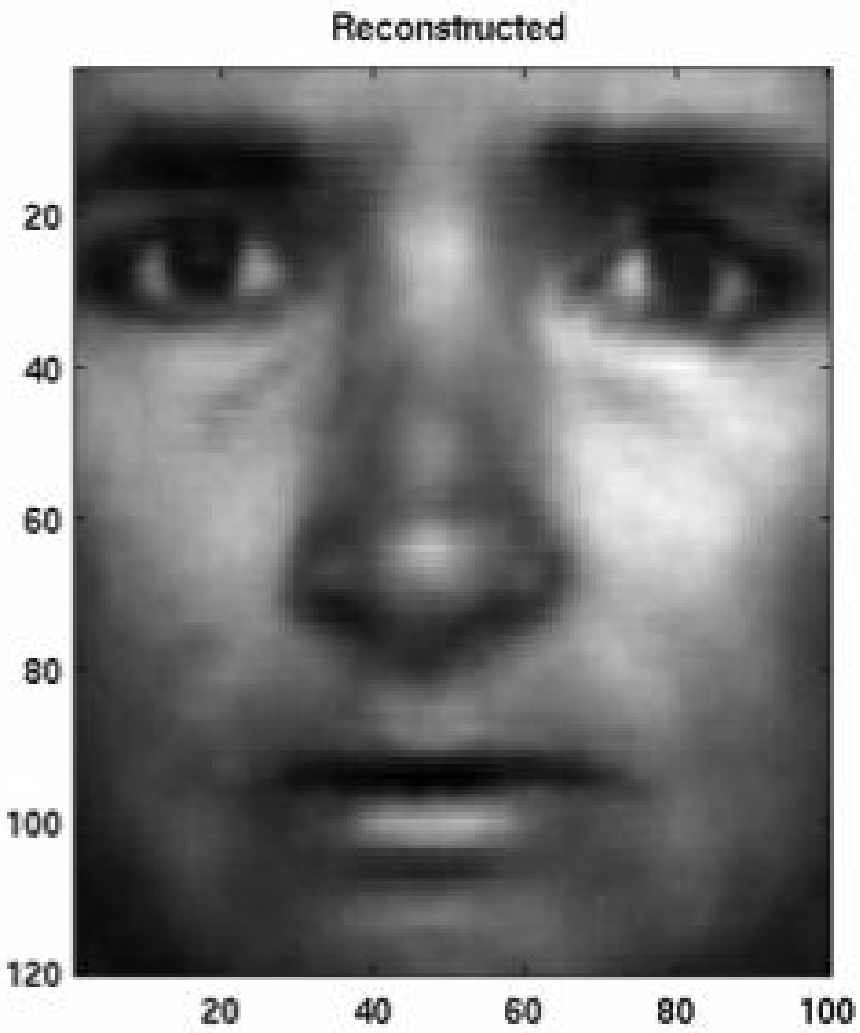


Reconstructing a face (with first 20 eigenfaces)

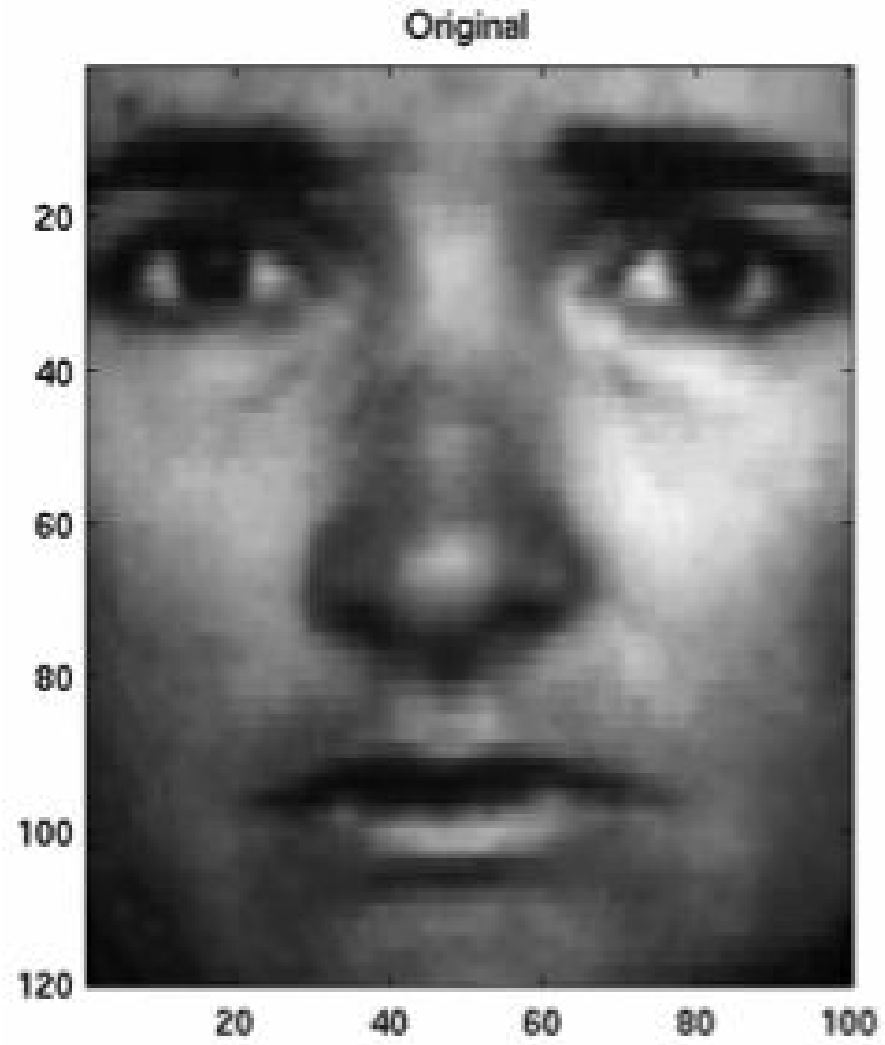


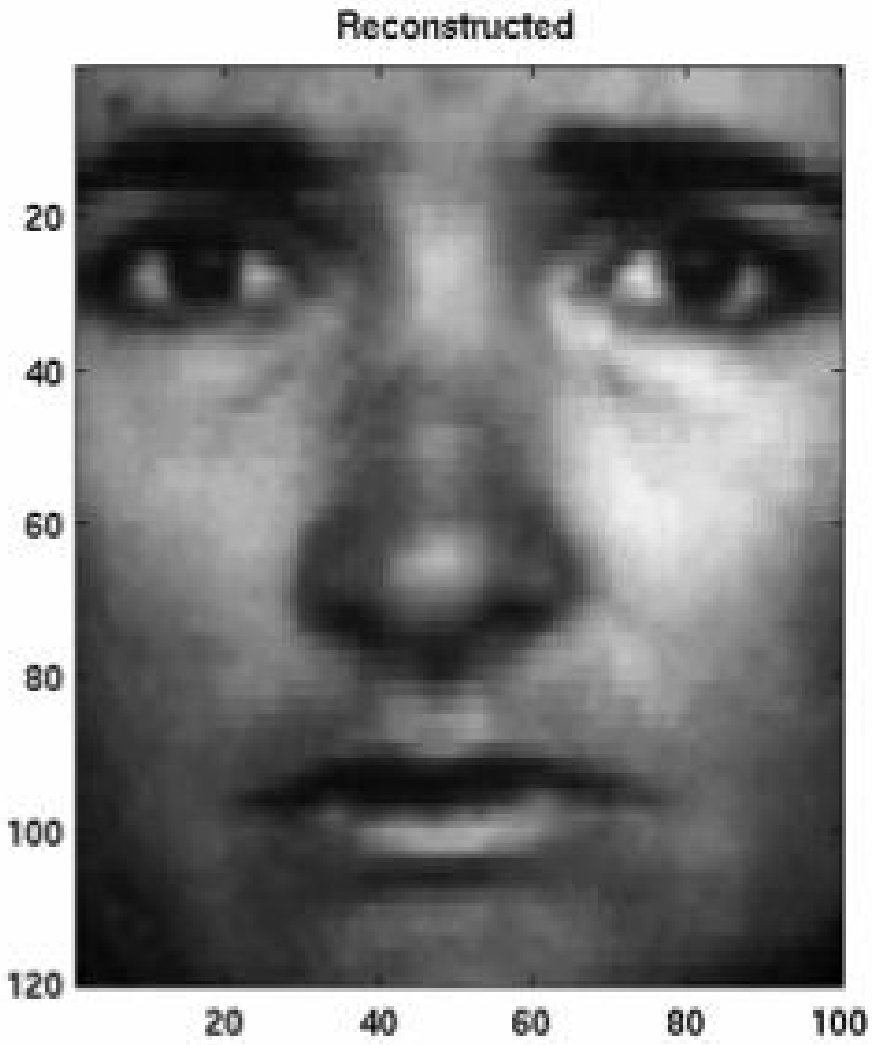
Reconstructing a face (with first 40 eigenfaces)





Reconstructing a face (with all 97 eigenfaces)





Remember Matrix Transformation - A rotation matrix is orthogonal

Multiplying a vector x by a matrix $y = Ax$ transforms x to a new vector y (which may have a different number of dimensions if A is not square). We can talk about different properties that the transformation given by A represents.

rotation matrix is given by

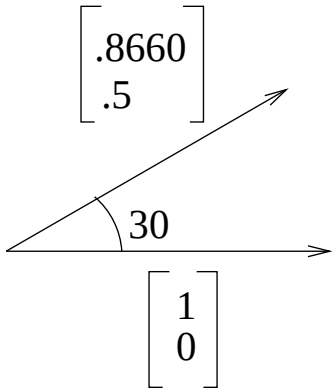
$$A = \begin{bmatrix} cos(\theta) & -sin(\theta) \\ sin(\theta) & cos(\theta) \end{bmatrix}$$

So to rotate vector

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

by $30\,\mathrm{deg}$ we multiply

$$\begin{bmatrix} .8660 & -.5 \\ .5 & .8660 \end{bmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} .8660 \\ .5 \end{pmatrix}$$



Rotation matrices are orthogonal

$$\begin{bmatrix} .8660 & -.5 \\ .5 & .8660 \end{bmatrix}^T \begin{bmatrix} .8660 & -.5 \\ .5 & .8660 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

This makes sense because to invert a rotation of angle α we want to rotate by $-\alpha$ and since $\cos(-\theta) = \cos(\theta)$ and $\sin(-\theta) = -\sin(\theta)$, the inverse of a rotation matrix is the transpose (try it!).

Remember Eigenvalues and Eigenvectors

When a Matrix multiplies a vector in general the direction and magnitude of the vector will change.

BUT there are special vectors where only the magnitude changes (on multiplication by the Matrix). These are called **eigenvectors** The value by which the length changes is the associated **eigenvalue**

We say that x is an eigenvector of A iff

$$Ax = \lambda x$$

In other words, x is an eigenvector if when you multiply it by A it returns a multiple of itself. λ is called the associated eigenvalue.

on-line demo