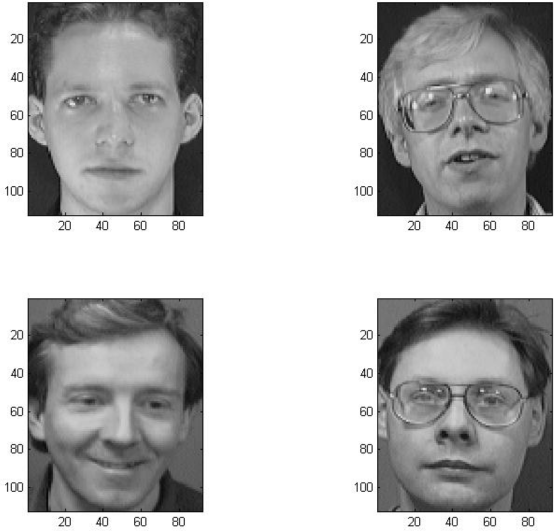
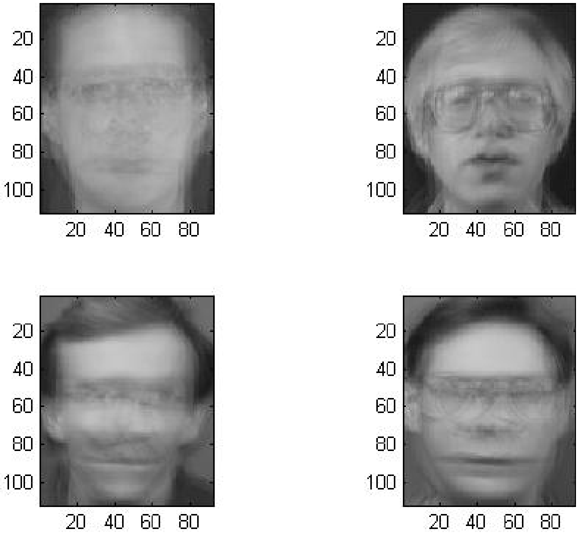
- We are going to build a simple system that does face recognition using PCA.
- Data set: AT&T Lab face image data: 400 images of size 112×92 of 40 different individuals. Each has 10 images.
- Each image is is gray scale, 10,304 pixels, represented by a matrix of size 112×92.
- Each pixel has value ranging from 1 to 255: 255 is white and 1 is black. Anything in between is some shades of grey.

- Our data matrix, in order to work with PCA, has to have dimension $n \times p$.
- Each row corresponds to an observation (image), each column corresponds to some measurement.
- Each image is already a matrix.
- An image will be unfolded to a vector by rows (or columns) of that matrix and line them up to make a long vector.
- We can construct our data matrix from the collections of vectorized images.


20
40
60
80
100
20 40 60 80
20
40
60
80
100
20 40 60 80
20
40
60
80
100
20 40 60 80
20
40
60
80
100
20 40 60 80

- In Matlab, originally images are usuallly unsigned integer. In order to work with PCA, we have to convert it to double precision.
- We want to figure out what are the principal components of these face images.
- Given a new picture, can we find out which pictures have good match with it?
- Training data has around 320 images, testing data has 80 images.


20
40
60
80
100
20 40 60 80
20
40
60
80
100
20 40 60 80
20
40
60
80
100
20 40 60 80
20
40
60
80
100
20 40 60 80

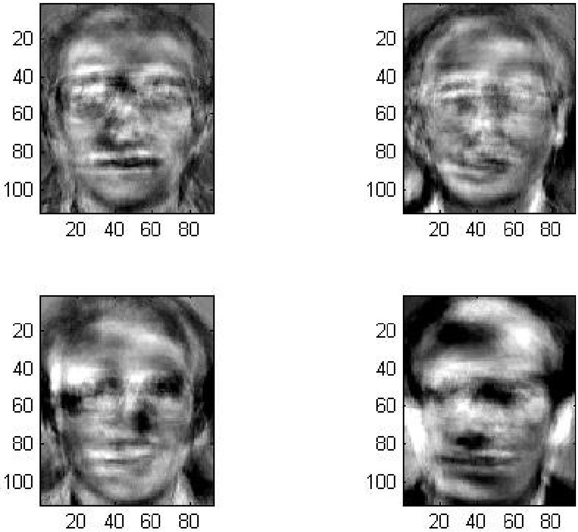
- Lets call the data matrix A and its covariance matrix $C = A^T A$.
- Images of the same person are supposed to have high correlation. Images of different persons are supposed to have low correlation.
- Use PCA to find principal components of matrix C.
- Find eigen values and eigen vectors of C.

- Matrix C usually have high dimensions: 10304×10304 for our data set.
- Calculating the eigen values and eigen vectors of such large matrix is very time consuming.
- One little trick: Eigen values and eigen vectors of C are related to those of $D = AA^T$.
- Let v and λ be eigen vector and eigen values of C, we have:

$$Cv = A^T Av = \lambda v.\text{(multiply both sides with A)}$$
$$AA^T(Av) = A\lambda v = \lambda(Av).$$

- If v and λ be eigen vector and eigen values of $C = A^T A$, then (Av) and λ is eigen vectors and eigen values of $D = AA^T$.
- Vice versa, if v and λ are eigen vectors and eigen values of $D = AA^T$ then $(A^T v)$ and λ are eigen vectors and eigen values of $C = A^T A$.
- Instead calculating eigen values and eigen vectors of large matrix C, we can find those of smaller matrix D.
- $[V, E] = eigs(D, 100, 'lm')$
- Then eigen vectors of C can be calculated by: $A^T V$.

- These eigen vectors are also called eigen faces.
- Each image in the training set can be represented as a linear combination of eigen faces (principal components).
- When a new image turns up, it will be represented as a combination of the basis vectors.
- The system output an image in the directory that is closest in distance with the new image.


20
40
60
80
100
20 40 60 80

- For a new image I, the projection of the image on to the space of the eigen faces (in other words, representing the new image as a linear combination of eigen faces)

$$P_I = I * V$$

- P_I is the new co-ordinate of the new image in the eigen face space.
- The system calculate the distance between the P_I and the images in the directory and find the closest one.