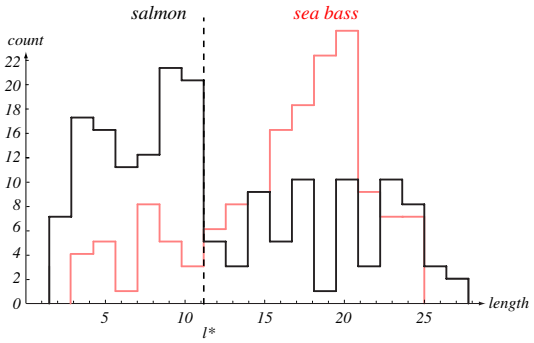
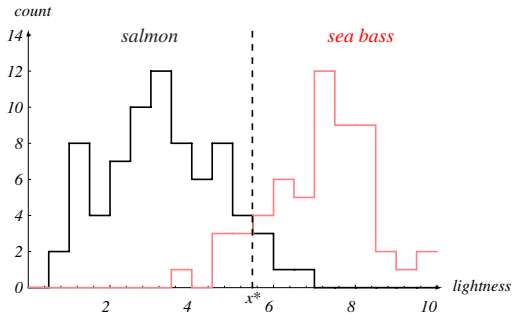
Dimension Reduction and Component Analysis

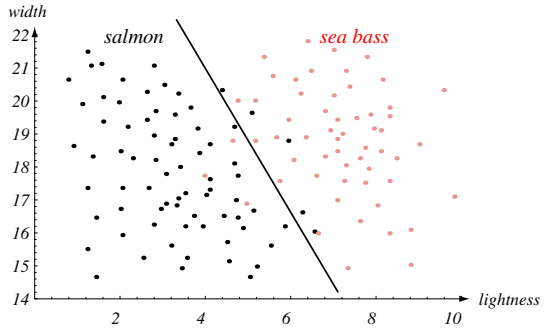
Jason Corso

SUNY at Buffalo

- We learned about estimating parametric models and how these then form classifiers and define decision boundaries
- Now we turn back to the question of dimensionality.
- Recall the fish example, where we experimented with the length feature first, then the lightness feature, and then decided upon a combination of the width and lightness.

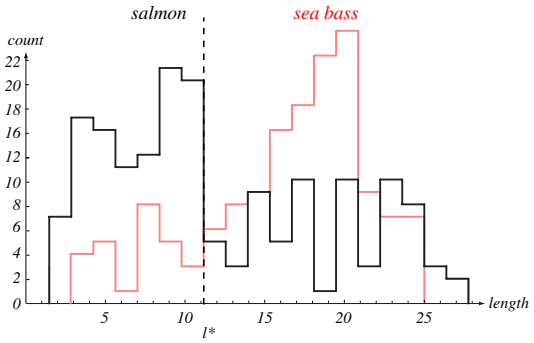


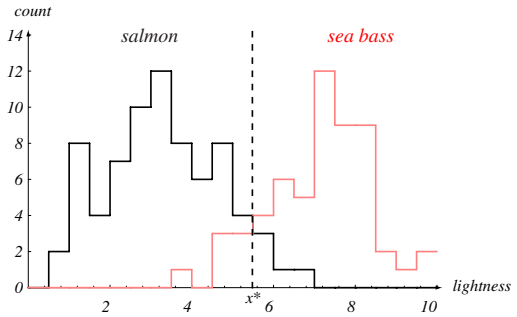


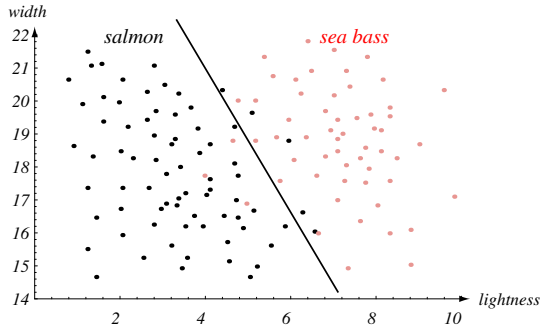


- We developed some intuition saying *the more features I add, the better my classifier will be...*

- We learned about estimating parametric models and how these then form classifiers and define decision boundaries
- Now we turn back to the question of dimensionality.
- Recall the fish example, where we experimented with the length feature first, then the lightness feature, and then decided upon a combination of the width and lightness.

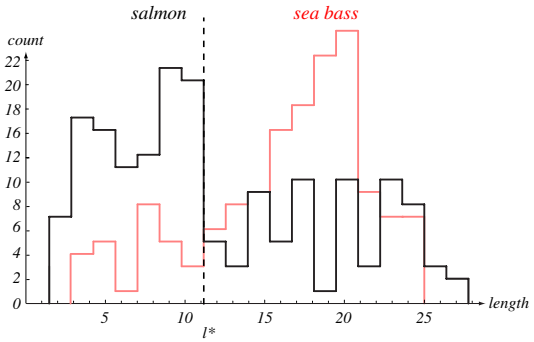


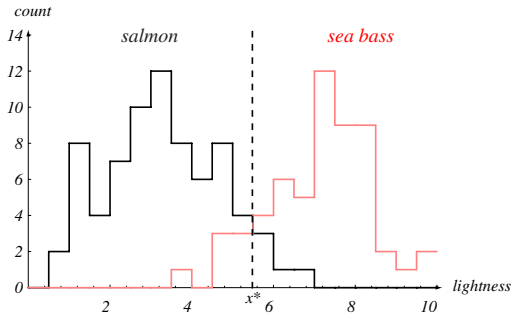


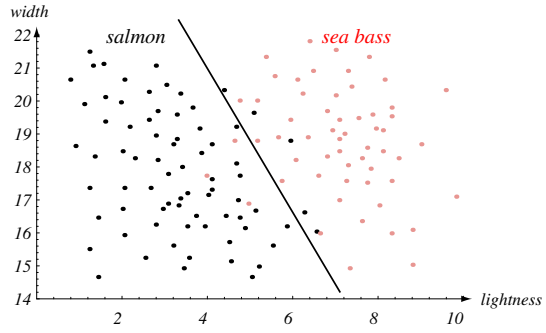


- We developed some intuition saying *the more features I add, the better my classifier will be...*
- We will see that in theory this may be, but in practice, this is not the case—the probability of error will increase after a certain number of features (dimensionality) has been reached.

- We learned about estimating parametric models and how these then form classifiers and define decision boundaries
- Now we turn back to the question of dimensionality.
- Recall the fish example, where we experimented with the length feature first, then the lightness feature, and then decided upon a combination of the width and lightness.







- We developed some intuition saying *the more features I add, the better my classifier will be...*
- We will see that in theory this may be, but in practice, this is not the case—the probability of error will increase after a certain number of features (dimensionality) has been reached.
- We will first explore this point and then discuss a set of methods for dimension reduction.

High Dimensions Often Test Our Intuitions

- Consider a simple arrangement: you have a sphere of radius $r = 1$ in a space of D dimensions.
- We want to compute what is the fraction of the volume of the sphere that lies between radius $r = 1 - \epsilon$ and $r = 1$.

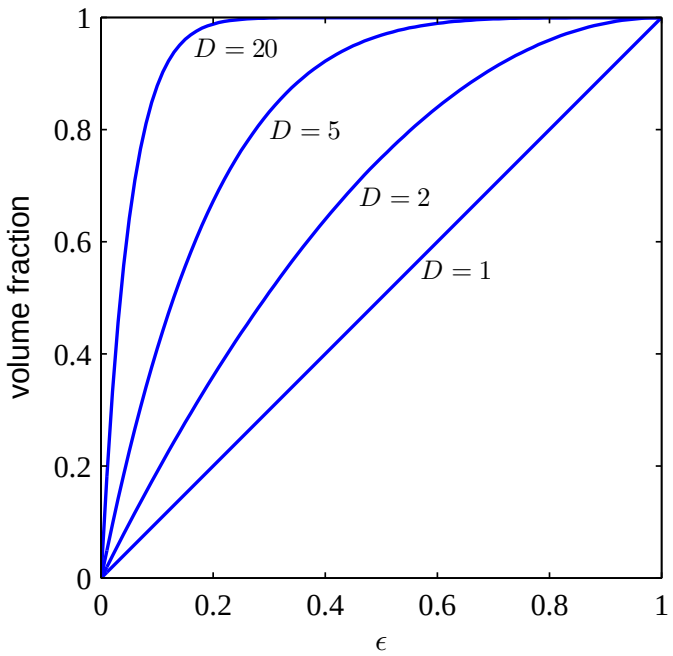
High Dimensions Often Test Our Intuitions

- Consider a simple arrangement: you have a sphere of radius $r = 1$ in a space of D dimensions.
- We want to compute what is the fraction of the volume of the sphere that lies between radius $r = 1 - \epsilon$ and $r = 1$.
- Noting that the volume of the sphere will scale with r^D, we have:

$$V_D(r) = K_D r^D \tag{1}$$

where K_D is some constant (depending only on D).

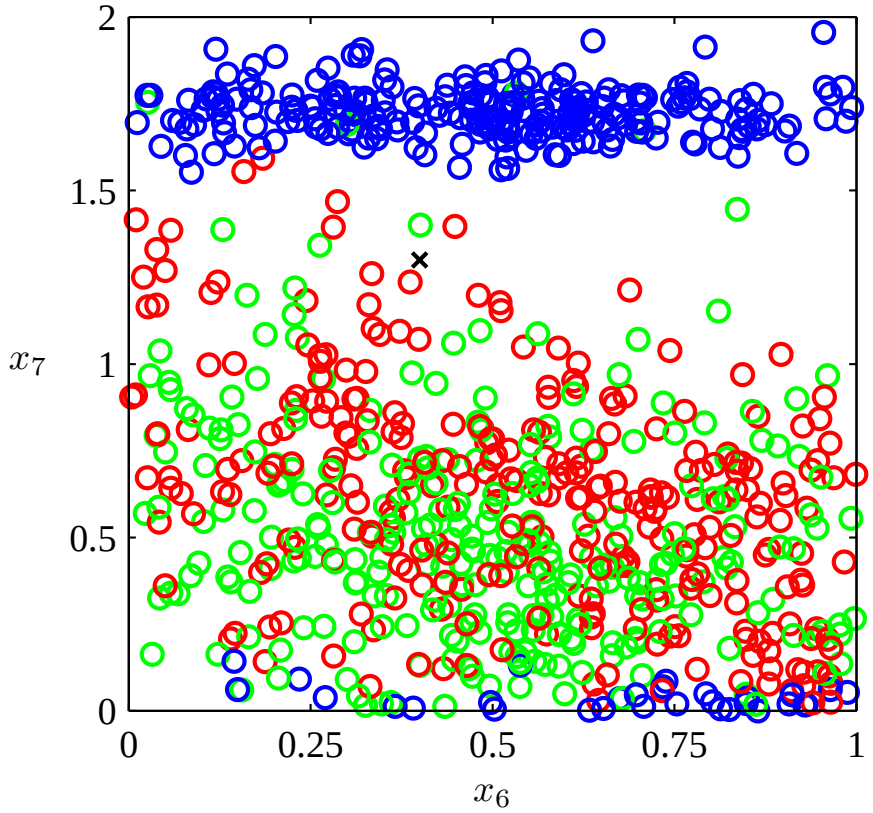
$$\frac{V_D(1) - V_D(1-\epsilon)}{V_D(1)} = 1 - (1-\epsilon)^D \tag{2}$$



Let's Build Some More Intuition

Example from Bishop PRML

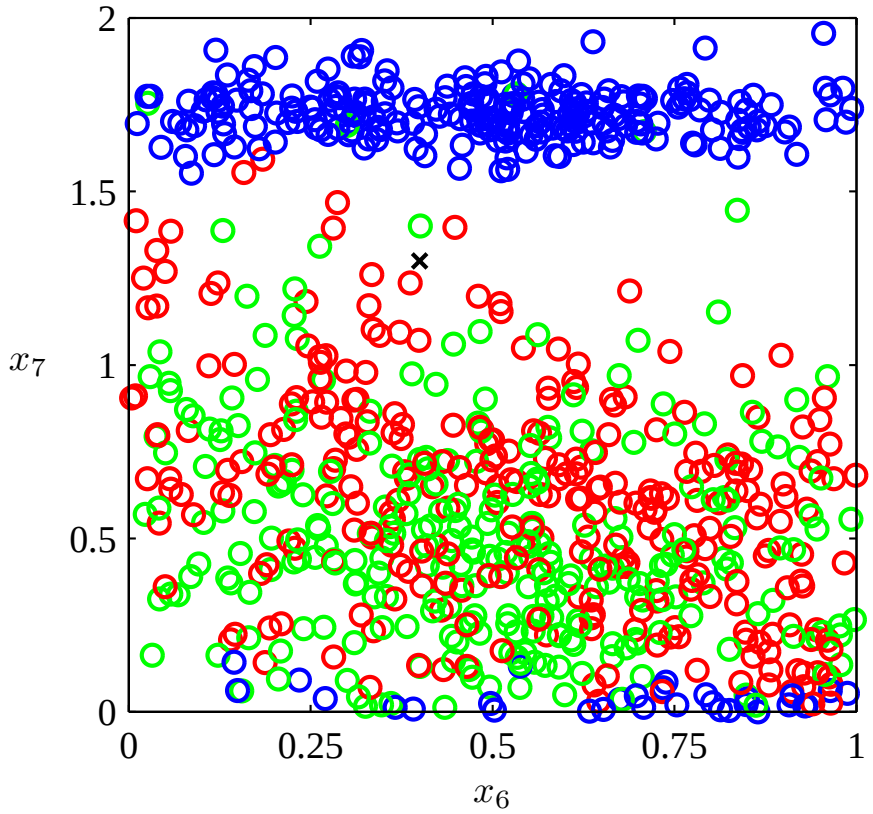
- Dataset: Measurements taken from a pipeline containing a mixture of oil.
  - Three classes present (different geometrical configuration): homogeneous, annular, and laminar.
  - Each data point is a 12 dimensional input vector consisting of measurements taken with gamma ray densitometers, which measure the attenuation of gamma rays passing along narrow beams through the pipe.



Let's Build Some More Intuition

Example from Bishop PRML

- 100 data points of features x_6 and x_7 are shown on the right.
- **Goal:** Classify the new data point at the 'x'.
- Suggestions on how we might approach this classification problem?

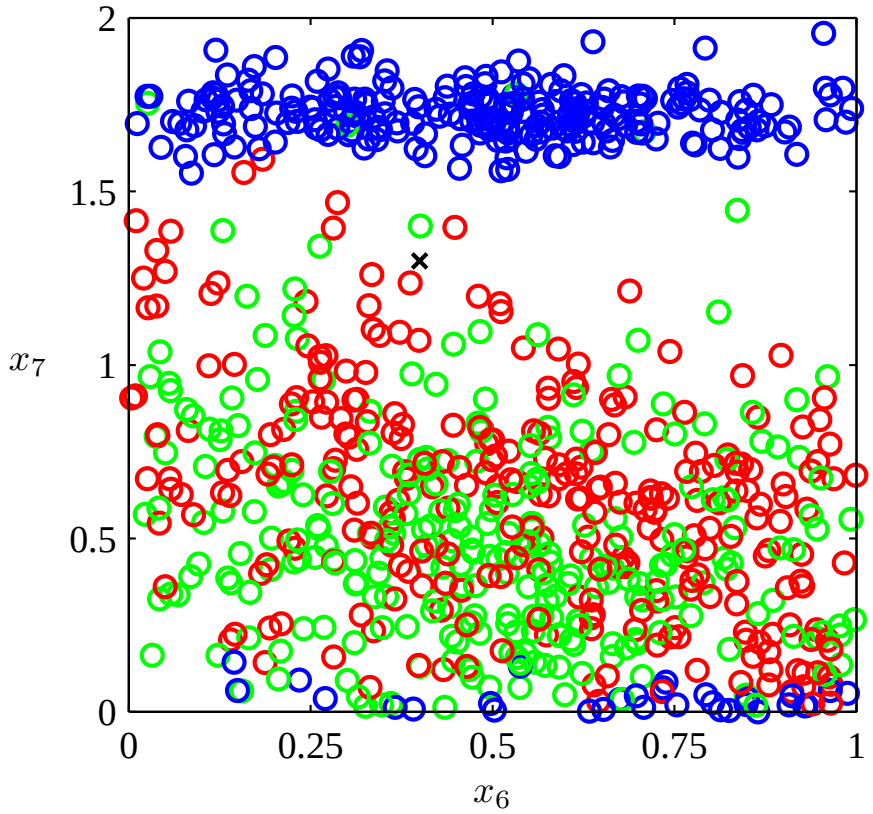


Let's Build Some More Intuition

Example from Bishop PRML

Observations we can make:

- The cross is surrounded by many red points and some green points.
- Blue points are quite far from the cross.
- **Nearest-Neighbor Intuition:** The query point should be determined more strongly by nearby points from the training set and less strongly by more distant points.

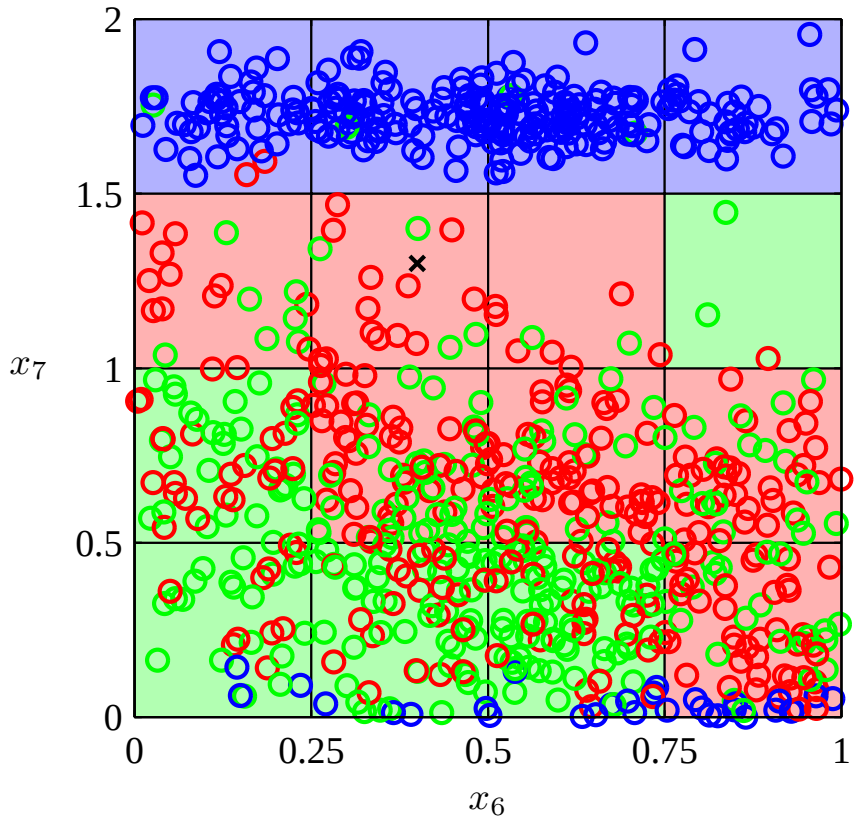


Let's Build Some More Intuition

Example from Bishop PRML

One simple way of doing it is:

- We can divide the feature space up into regular cells.
- For each, cell, we associated the class that occurs most frequently in that cell (in our training data).
- Then, for a query point, we determine which cell it falls into and then assign in the label associated with the cell.

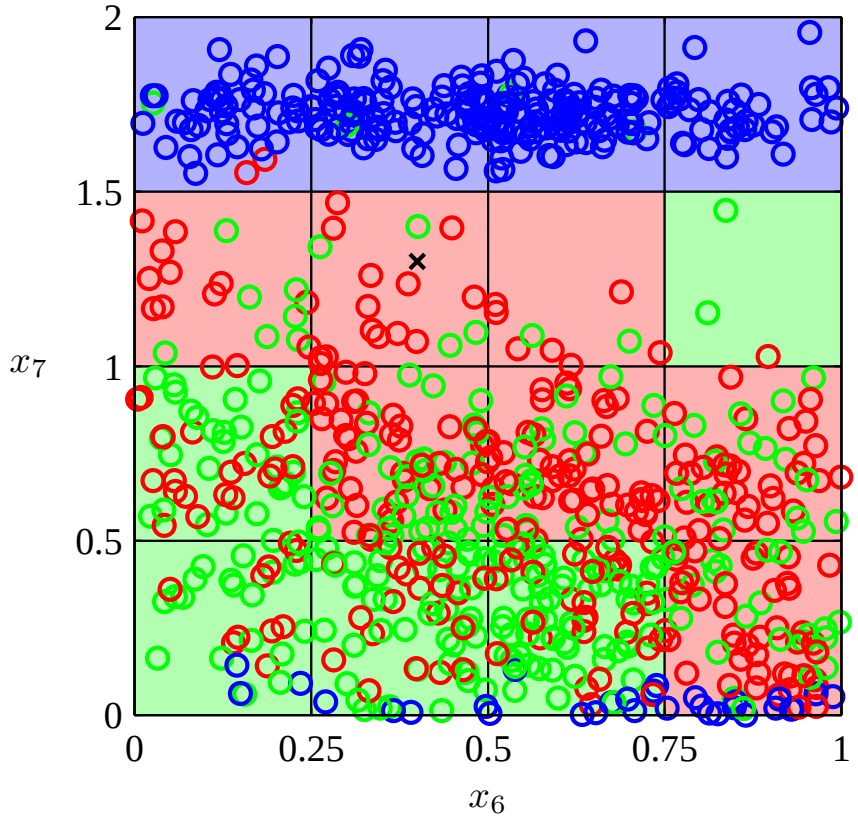


Let's Build Some More Intuition

Example from Bishop PRML

One simple way of doing it is:

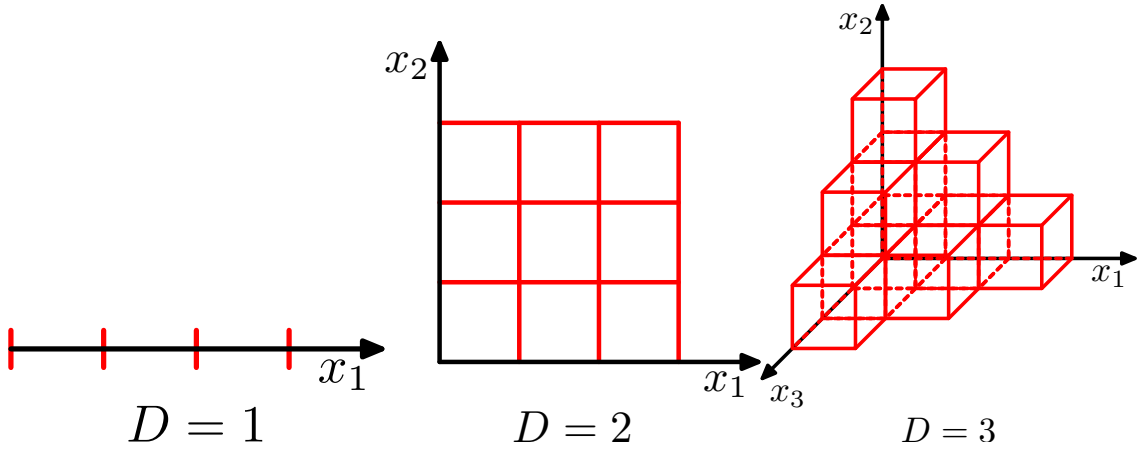
- We can divide the feature space up into regular cells.
- For each, cell, we associated the class that occurs most frequently in that cell (in our training data).
- Then, for a query point, we determine which cell it falls into and then assign in the label associated with the cell.
- What problems may exist with this approach?



Let's Build Some More Intuition

Example from Bishop PRML

- The problem we are most interested in now is the one that becomes apparent when we add more variables into the mix, corresponding to problems of higher dimensionality.
- In this case, the number of additional cells **grows exponentially** with the dimensionality of the space.
- Hence, we would need an exponentially large training data set to ensure all cells are filled.



Curse of Dimensionality

- This severe difficulty when working in high dimensions was coined **the curse of dimensionality** by Bellman in 1961.
- The idea is that the volume of a space increases exponentially with the dimensionality of the space.

Dimensionality and Classification Error?

Some parts taken from G. V. Trunk, TPAMI Vol. 1 No. 3 PP. 306-7 1979

- How does the probability of error vary as we add more features, in theory?
- Consider the following two-class problem:
  - The prior probabilities are known and equal: $P(\omega_1) = P(\omega_2) = 1/2$.
  - The class-conditional densities are Gaussian with unit covariance:

$$p(\mathbf{x}|\omega_1) \sim N(\boldsymbol{\mu}_1, \mathbf{I}) \tag{3}$$
$$p(\mathbf{x}|\omega_2) \sim N(\boldsymbol{\mu}_2, \mathbf{I}) \tag{4}$$

    where $\boldsymbol{\mu}_1 = \boldsymbol{\mu}$, $\boldsymbol{\mu}_2 = -\boldsymbol{\mu}$, and $\boldsymbol{\mu}$ is an n-vector whose ith component is $(1/i)^{1/2}$.

- The corresponding Bayesian Decision Rule is

$$\text{decide } \omega_1 \text{ if } \mathbf{x}^{\mathsf{T}}\boldsymbol{\mu} > 0 \tag{5}$$

Dimensionality and Classification Error?

Some parts taken from G. V. Trunk, TPAMI Vol. 1 No. 3 PP. 306-7 1979

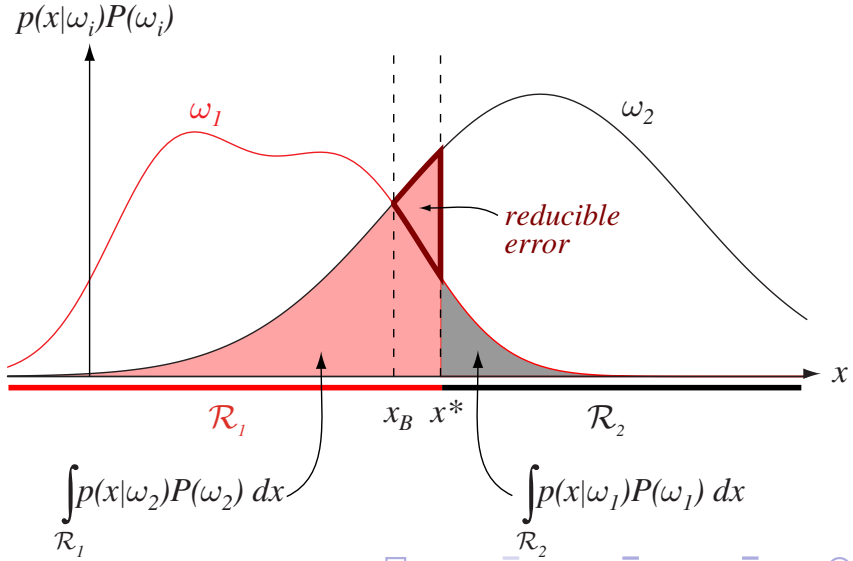
- The probability of error is

$$P(\text{error}) = \frac{1}{\sqrt{2\pi}} \int_{r/2}^{\infty} \exp\left[-z^2/2\right] dz \tag{6}$$

where

$$r^2 = \|\mu_1 - \mu_2\|^2 = 4 \sum_{i=1}^{n} (1/i). \tag{7}$$

- Let's take this integral for granted... (For more detail, you can look at DHS Problem 31 in Chapter 2 and read Section 2.7.)



Dimensionality and Classification Error?

Some parts taken from G. V. Trunk, TPAMI Vol. 1 No. 3 PP. 306-7 1979

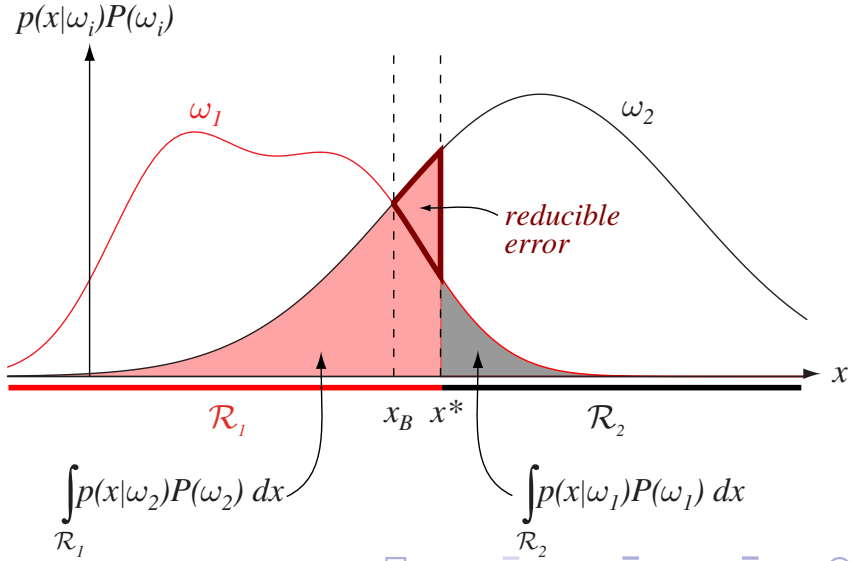
- The probability of error is

$$P(\text{error}) = \frac{1}{\sqrt{2\pi}} \int_{r/2}^{\infty} \exp\left[-z^2/2\right] dz \tag{6}$$

where

$$r^2 = \|\mu_1 - \mu_2\|^2 = 4 \sum_{i=1}^{n} (1/i). \tag{7}$$

- Let's take this integral for granted... (For more detail, you can look at DHS Problem 31 in Chapter 2 and read Section 2.7.)
- What can we say about this result as more features are added?



Dimensionality and Classification Error?

Some parts taken from G. V. Trunk, TPAMI Vol. 1 No. 3 PP. 306-7 1979

- The probability of error approaches 0 as n approach infinity because $1/i$ is a divergent series.
- More intuitively, each additional feature is going to decrease the probability of error as long as its means are different. In the general case of varying means and but same variance for a feature, we have

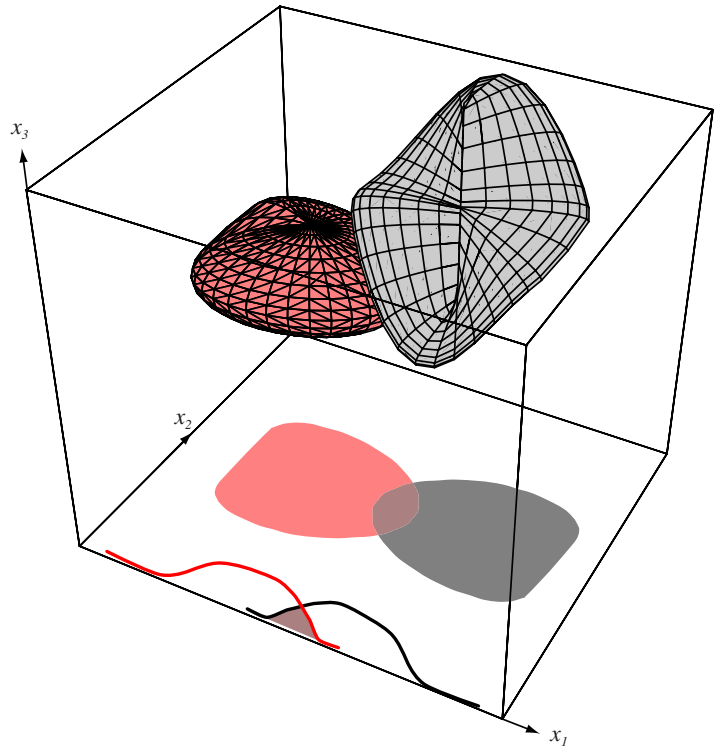
$$r^2 = \sum_{i=1}^{d} \left(\frac{\mu_{i1} - \mu_{i2}}{\sigma_i} \right)^2 \tag{8}$$

  Certainly, we prefer features that have big differences in the mean relative to their variance.
- We need to note that if the probabilistic structure of the problem is completely known then adding new features is not going to decrease the Bayes risk (or increase it).

Dimensionality and Classification Error?

Some parts taken from G. V. Trunk, TPAMI Vol. 1 No. 3 PP. 306-7 1979

- So, adding dimensions is good....



- ...in theory.
- But, in practice, performance seems to not obey this theory.

Dimensionality and Classification Error?

Some parts taken from G. V. Trunk, TPAMI Vol. 1 No. 3 PP. 306-7 1979

- Consider again the two-class problem, but this time with unknown means $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$.
- Instead, we have m labeled samples $\mathbf{x}_1, \ldots, \mathbf{x}_m$.
- Then, the best estimate of $\boldsymbol{\mu}$ for each class is the sample mean (recall the parameter estimation lecture).

$$\overline{\boldsymbol{\mu}} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i \tag{9}$$

where $\mathbf{x}_i$ comes from ω_2. The covariance matrix is $\mathbf{I}/m$.

Dimensionality and Classification Error?

Some parts taken from G. V. Trunk, TPAMI Vol. 1 No. 3 PP. 306-7 1979

- Probablity of error is given by

$$P(\text{error}) = P(\mathbf{x}^{\mathsf{T}}\boldsymbol{\mu} \geq 0|\omega_2) = \frac{1}{\sqrt{2\pi}} \int_{\gamma_n}^{\infty} \exp\left[-z^2/2\right] dz \tag{10}$$

because it has a Gaussian form as n approaches infinity where

$$\gamma_n = E(z)/\left[\text{var}(z)\right]^{1/2} \tag{11}$$

$$E(z) = \sum_{i=1}^{n} (1/i) \tag{12}$$

$$\text{var}(z) = \left(1 + \frac{1}{m}\right) \sum_{i=1}^{n} (1/i) + n/m \tag{13}$$

Dimensionality and Classification Error?

Some parts taken from G. V. Trunk, TPAMI Vol. 1 No. 3 PP. 306-7 1979

- Probablity of error is given by

$$P(\text{error}) = P(\mathbf{x}^{\mathsf{T}}\overline{\boldsymbol{\mu}} \geq 0|\omega_2) = \frac{1}{\sqrt{2\pi}} \int_{\gamma_n}^{\infty} \exp\left[-z^2/2\right] dz$$

- The key is that we can show

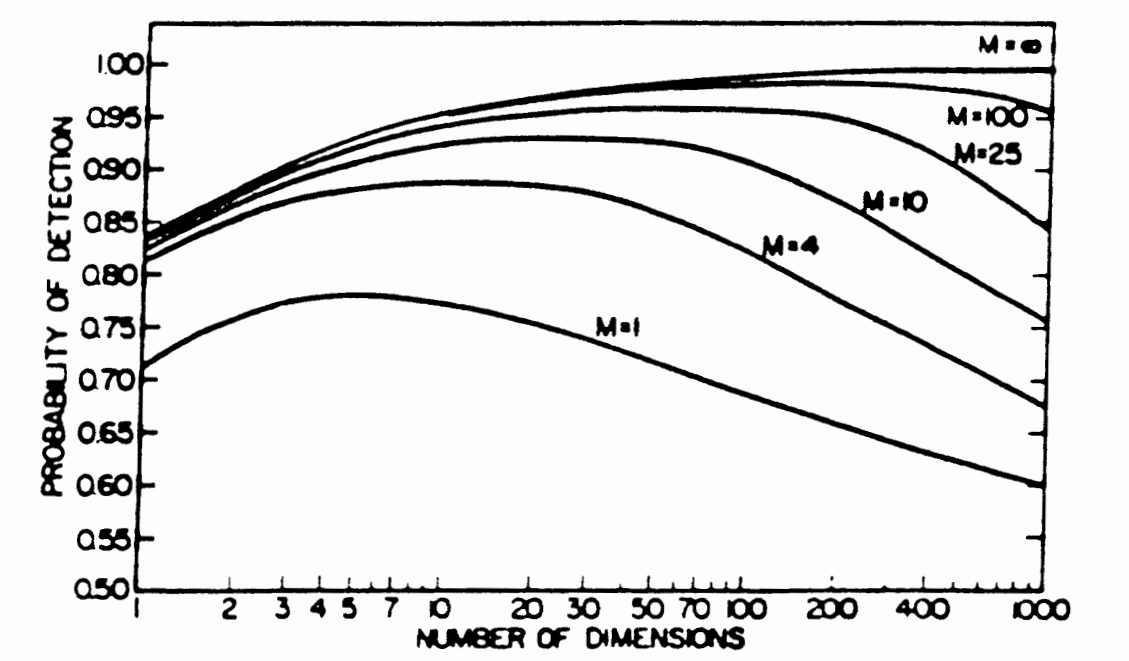
$$\lim_{n\to\infty} \gamma_n = 0 \tag{14}$$

and thus the probability of error approaches one-half as the dimensionality of the problem becomes very high.

Dimensionality and Classification Error?

Some parts taken from G. V. Trunk, TPAMI Vol. 1 No. 3 PP. 306-7 1979

- Trunk performed an experiment to investigate the convergence rate of the probability of error to one-half. He simulated the problem for dimensionality 1 to 1000 and ran 500 repetitions for each dimension.
- We see an increase in performance initially and then a decrease (as the dimensionality of the problem grows larger than the number of training samples).



Motivation for Dimension Reduction

- The discussion on the curse of dimensionality should be enough!
- Even though our problem may have a high dimension, data will often be confined to a much lower effective dimension in most real world problems.
- Computational complexity is another important point: generally, the higher the dimension, the longer the training stage will be (and potentially the measurement and classification stages).
- We seek an understanding of the underlying data to
  - Remove or reduce the influence of noisy or irrelevant features that would otherwise interfere with the classifier;
  - Identify a small set of features (perhaps, a transformation thereof) during data exploration.

Dimension Reduction Version 0

- We have n samples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.
- How can we best represent the n samples by a single vector $\mathbf{x}_0$?

Dimension Reduction Version 0

- We have n samples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.
- How can we best represent the n samples by a single vector $\mathbf{x}_0$?
- First, we need a distance function on the sample space. Let's use the Euclidean distance and the sum of squared distances criterion:

$$J_0(\mathbf{x}_0) = \sum_{k=1}^{n} \|\mathbf{x}_0 - \mathbf{x}_k\|^2 \tag{15}$$

Dimension Reduction Version 0

- We have n samples $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$.
- How can we best represent the n samples by a single vector $\mathbf{x}_0$?
- First, we need a distance function on the sample space. Let's use the Euclidean distance and the sum of squared distances criterion:

$$J_0(\mathbf{x}_0) = \sum_{k=1}^{n} \|\mathbf{x}_0 - \mathbf{x}_k\|^2 \tag{15}$$

- Then, we seek a value of $\mathbf{x}_0$ that minimizes J_0.

Dimension Reduction Version 0

- We can show that the minimizer is indeed the sample mean:

$$\mathbf{m} = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k \tag{16}$$

- We can verify it by adding $\mathbf{m} - \mathbf{m}$ into J_0:

$$J_0(\mathbf{x}_0) = \sum_{k=1}^{n}\|(\mathbf{x}_0 - \mathbf{m}) - (\mathbf{x}_k - \mathbf{m})\|^2 \tag{17}$$

$$= \sum_{k=1}^{n}\|\mathbf{x}_0 - \mathbf{m}\|^2 + \sum_{k=1}^{n}\|\mathbf{x}_k - \mathbf{m}\|^2 \tag{18}$$

- Thus, J_0 is minimized when $\mathbf{x}_0 = \mathbf{m}$. Note, the second term is independent of $\mathbf{x}_0$.

Dimension Reduction Version 0

- So, the sample mean is an initial dimension-reduced representation of the data (a zero-dimensional one).
- It is simple, but it not does reveal any of the variability in the data.
- Let's try to obtain a one-dimensional representation: i.e., let's project the data onto a line running through the sample mean.

Dimension Reduction Version 1

- Let $\mathbf{e}$ be a unit vector in the direction of the line.
- The standard equation for a line is then

$$\mathbf{x} = \mathbf{m} + a\mathbf{e} \tag{19}$$

where scalar $a \in \mathbb{R}$ governs which point along the line we are and hence corresponds to the distance of any point $\mathbf{x}$ from the mean $\mathbf{m}$.

Dimension Reduction Version 1

- Let $\mathbf{e}$ be a unit vector in the direction of the line.
- The standard equation for a line is then

$$\mathbf{x} = \mathbf{m} + a\mathbf{e} \tag{19}$$

  where scalar $a \in \mathbb{R}$ governs which point along the line we are and hence corresponds to the distance of any point $\mathbf{x}$ from the mean $\mathbf{m}$.
- Represent point $\mathbf{x}_k$ by $\mathbf{m} + a_k\mathbf{e}$.
- We can find an optimal set of coefficients by again minimizing the squared-error criterion

$$J_1(a_1, \ldots, a_n, \mathbf{e}) = \sum_{k=1}^{n} \|(\mathbf{m} + a_k\mathbf{e}) - \mathbf{x}_k\|^2 \tag{20}$$

$$= \sum_{k=1}^{n} a_k^2 \|\mathbf{e}\|^2 - 2\sum_{k=1}^{n} a_k \mathbf{e}^{\mathsf{T}}(\mathbf{x}_k - \mathbf{m}) + \sum_{k=1}^{n} \|\mathbf{x}_k - \mathbf{m}\|^2 \tag{21}$$

Dimension Reduction Version 1

- Differentiating for a_k and equating to 0 yields

$$a_k = \mathbf{e}^T(\mathbf{x}_k - \mathbf{m}) \tag{22}$$

- This indicates that the best value for a_k is the projection of the point $\mathbf{x}_k$ onto the line $\mathbf{e}$ that passes through $\mathbf{m}$.

Dimension Reduction Version 1

- Differentiating for a_k and equating to 0 yields

$$a_k = \mathbf{e}^{\mathsf{T}}(\mathbf{x}_k - \mathbf{m}) \tag{22}$$

- This indicates that the best value for a_k is the projection of the point $\mathbf{x}_k$ onto the line $\mathbf{e}$ that passes through $\mathbf{m}$.
- How do we find the best direction for that line?

Dimension Reduction Version 1

- What if we substitute the expression we computed for the best a_k directly into the J_1 criterion:

$$J_1(\mathbf{e}) = \sum_{k=1}^{n} a_k^2 - 2\sum_{k=1}^{n} a_k^2 + \sum_{k=1}^{n} \|x_k - \mathbf{m}\|^2 \tag{23}$$

$$= -\sum_{k=1}^{n} \left[\mathbf{e}^{\mathsf{T}}(\mathbf{x}_k - \mathbf{m})\right]^2 + \sum_{k=1}^{n} \|\mathbf{x}_k - \mathbf{m}\|^2 \tag{24}$$

$$= -\sum_{k=1}^{n} \mathbf{e}^{\mathsf{T}}(\mathbf{x}_k - \mathbf{m})(\mathbf{x}_k - \mathbf{m})^{\mathsf{T}}\mathbf{e} + \sum_{k=1}^{n} \|\mathbf{x}_k - \mathbf{m}\|^2 \tag{25}$$

Dimension Reduction Version 1

The Scatter Matrix

- Define the **scatter matrix** $\mathbf{S}$ as

$$\mathbf{S} = \sum_{k=1}^{n} (\mathbf{x}_k - \mathbf{m})(\mathbf{x}_k - \mathbf{m})^{\mathsf{T}} \tag{26}$$

Dimension Reduction Version 1

The Scatter Matrix

- Define the **scatter matrix** $\mathbf{S}$ as

$$\mathbf{S} = \sum_{k=1}^{n} (\mathbf{x}_k - \mathbf{m})(\mathbf{x}_k - \mathbf{m})^{\mathsf{T}} \tag{26}$$

- This should be familiar – this is a multiple of the sample covariance matrix.

Dimension Reduction Version 1

The Scatter Matrix

- Define the **scatter matrix** $\mathbf{S}$ as

$$\mathbf{S} = \sum_{k=1}^{n} (\mathbf{x}_k - \mathbf{m})(\mathbf{x}_k - \mathbf{m})^{\mathsf{T}} \tag{26}$$

- This should be familiar – this is a multiple of the sample covariance matrix.
- Putting it in:

$$J_1(\mathbf{e}) = -\mathbf{e}^{\mathsf{T}}\mathbf{S}\mathbf{e} + \sum_{k=1}^{n} \|\mathbf{x}_k - \mathbf{m}\|^2 \tag{27}$$

- The $\mathbf{e}$ that maximizes $\mathbf{e}^{\mathsf{T}}\mathbf{S}\mathbf{e}$ will minimize J_1.

Dimension Reduction Version 1

Solving for $\mathbf{e}$

- We use Lagrange multipliers to maximize $\mathbf{e}^T\mathbf{S}\mathbf{e}$ subject to the constraint that $\|\mathbf{e}\| = 1$.

$$u = \mathbf{e}^T\mathbf{S}\mathbf{e} - \lambda(\mathbf{e}^T\mathbf{e} - 1) \tag{28}$$

Dimension Reduction Version 1

Solving for $\mathbf{e}$

- We use Lagrange multipliers to maximize $\mathbf{e}^T\mathbf{S}\mathbf{e}$ subject to the constraint that $\|\mathbf{e}\| = 1$.

$$u = \mathbf{e}^T\mathbf{S}\mathbf{e} - \lambda(\mathbf{e}^T\mathbf{e} - 1) \tag{28}$$

- Differentiating w.r.t. $\mathbf{e}$ and setting equal to 0.

$$\frac{\partial u}{\partial \mathbf{e}} = 2\mathbf{S}\mathbf{e} - 2\lambda\mathbf{e} \tag{29}$$

$$\mathbf{S}\mathbf{e} = \lambda\mathbf{e} \tag{30}$$

- Does this form look familiar?

Dimension Reduction Version 1

Eigenvectors of $\mathbf{S}$

$$\mathbf{S}\mathbf{e} = \lambda\mathbf{e}$$

- This is an eigenproblem.
- Hence, it follows that the best one-dimensional estimate (in a least-squares sense) for the data is the eigenvector corresponding to the largest eigenvalue of $\mathbf{S}$.
- So, we will project the data onto the largest eigenvector of $\mathbf{S}$ and translate it to pass through the mean.

Principal Component Analysis

- We're already done...basically.

Principal Component Analysis

- We're already done...basically.
- This idea readily extends to multiple dimensions, say $d' < d$ dimensions.
- We replace the earlier equation of the line with

$$\mathbf{x} = \mathbf{m} + \sum_{i=1}^{d'} a_i \mathbf{e}_i \tag{31}$$

- And we have a new criterion function

$$J_{d'} = \sum_{k=1}^{n} \left\| \left(\mathbf{m} + \sum_{i=1}^{d'} a_{ki} e_i \right) - \mathbf{x}_k \right\|^2 \tag{32}$$

Principal Component Analysis

- $J_{d'}$ is minimized when the vectors $\mathbf{e}_1, \ldots, \mathbf{e}_{d'}$ are the d' eigenvectors fo the scatter matrix having the largest eigenvalues.
- These vectors are orthogonal.
- They form a natural set of basis vectors for representing any feature $\mathbf{x}$.
- The coefficients a_i are called the **principal components**.
- Visualize the basis vectors as the principal axes of a hyperellipsoid surrounding the data (a cloud of points).
- Principle components reduces the dimension of the data by restricting attention to those directions of maximum variation, or scatter.

Fisher Linear Discriminant

- Description vs. Discrimination
- PCA is likely going to be useful for representing data.
- But, there is no reason to assume that it would be good for discriminating between two classes of data.
  - 'Q' versus 'O'.
- **Discriminant Analysis** seeks directions that are efficient for discrimination.

Let's Formalize the Situation

- Suppose we have a set of n d-dimensional samples with n_1 in set $\mathcal{D}_1$, and similarly for set $\mathcal{D}_2$.

$$\mathcal{D}_i = \{\mathbf{x}_1, \ldots, \mathbf{x}_{n_i}\}, \quad i = \{1, 2\} \tag{33}$$

Let's Formalize the Situation

- Suppose we have a set of n d-dimensional samples with n_1 in set $\mathcal{D}_1$, and similarly for set $\mathcal{D}_2$.

$$\mathcal{D}_i = \{\mathbf{x}_1, \ldots, \mathbf{x}_{n_i}\}, \quad i = \{1, 2\} \tag{33}$$

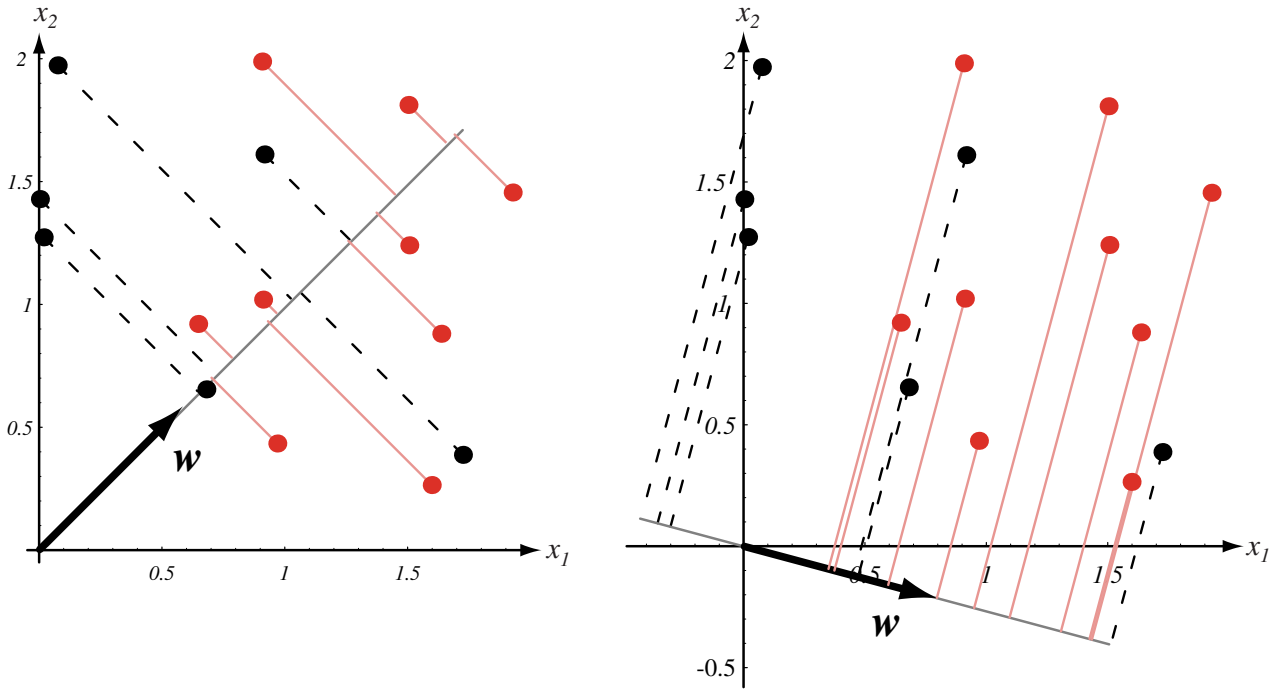
- We can form a linear combination of the components of a sample $\mathbf{x}$:

$$y = \mathbf{w}^{\mathsf{T}}\mathbf{x} \tag{34}$$

which yields a corresponding set of n samples $y_1, \ldots, y_n$ split into subsets $\mathcal{Y}_1$ and $\mathcal{Y}_2$.

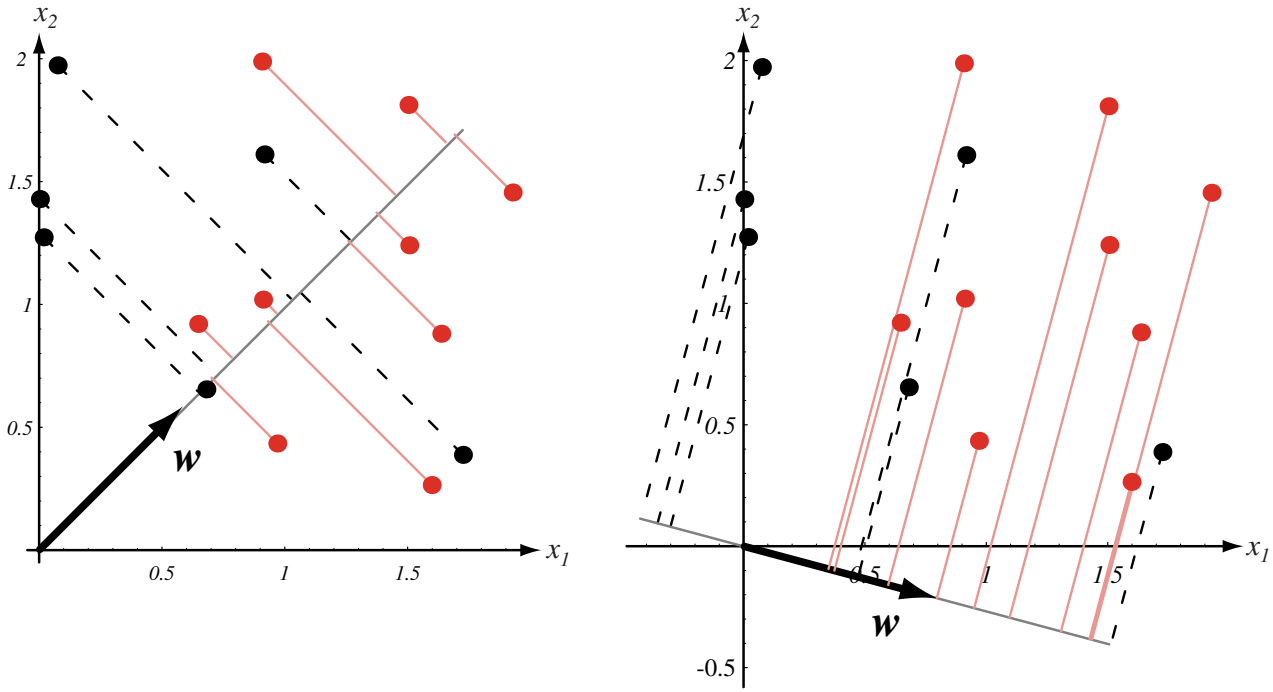
Geometrically, This is a Projection

- If we constrain the norm of $\mathbf{w}$ to be 1 (i.e., $\|\mathbf{w}\| = 1$) then we can conceptualize that each y_i is the projection of the corresponding $\mathbf{x}_i$ onto a line in the direction of $\mathbf{w}$.



Geometrically, This is a Projection

- If we constrain the norm of $\mathbf{w}$ to be 1 (i.e., $\|\mathbf{w}\| = 1$) then we can conceptualize that each y_i is the projection of the corresponding $\mathbf{x}_i$ onto a line in the direction of $\mathbf{w}$.



- Does the magnitude of $\mathbf{w}$ have any real significance?

What is a Good Projection?

- For our two-class setup, it should be clear that we want the projection that will have those samples from class ω_1 falling into one cluster (on the line) and those samples from class ω_2 falling into a separate cluster (on the line).

What is a Good Projection?

- For our two-class setup, it should be clear that we want the projection that will have those samples from class ω_1 falling into one cluster (on the line) and those samples from class ω_2 falling into a separate cluster (on the line).
- However, this may not be possible depending on our underlying classes.

What is a Good Projection?

- For our two-class setup, it should be clear that we want the projection that will have those samples from class ω_1 falling into one cluster (on the line) and those samples from class ω_2 falling into a separate cluster (on the line).
- However, this may not be possible depending on our underlying classes.
- So, how do we find the best direction $\mathbf{w}$?

Separation of the Projected Means

- Let $\mathbf{m}_i$ be the d-dimensional sample mean for class i:

$$\mathbf{m}_i = \frac{1}{n_i} \sum_{x \in \mathcal{D}_i} \mathbf{x} \ . \tag{35}$$

Separation of the Projected Means

- Let $\mathbf{m}_i$ be the d-dimensional sample mean for class i:

$$\mathbf{m}_i = \frac{1}{n_i} \sum_{x \in \mathcal{D}_i} \mathbf{x} \ . \tag{35}$$

- Then the sample mean for the projected points is

$$\tilde{m}_i = \frac{1}{n_i} \sum_{y \in \mathcal{Y}_i} y \tag{36}$$

$$= \frac{1}{n_i} \sum_{x \in \mathcal{D}_i} \mathbf{w}^\mathsf{T} \mathbf{x} \tag{37}$$

$$= \mathbf{w}^\mathsf{T} \mathbf{m}_i \ . \tag{38}$$

And, thus, is simply the projection of $\mathbf{m}_i$.

Distance Between Projected Means

- The distance between projected means is thus

$$|\tilde{m}_1 - \tilde{m}_2| = |\mathbf{w}^\mathsf{T}(\mathbf{m}_1 - \mathbf{m}_2)| \tag{39}$$

Distance Between Projected Means

- The distance between projected means is thus

$$|\tilde{m}_1 - \tilde{m}_2| = |\mathbf{w}^\mathsf{T}(\mathbf{m}_1 - \mathbf{m}_2)| \tag{39}$$

- The scale of $\mathbf{w}$: we can make this distance arbitrarily large by scaling $\mathbf{w}$.

Distance Between Projected Means

- The distance between projected means is thus

$$|\tilde{m}_1 - \tilde{m}_2| = |\mathbf{w}^\mathsf{T}(\mathbf{m}_1 - \mathbf{m}_2)| \tag{39}$$

- The scale of $\mathbf{w}$: we can make this distance arbitrarily large by scaling $\mathbf{w}$.
- Rather, we want to make this distance large relative to some measure of the standard deviation. ...This story we've heard before.

The Scatter

- To capture this variation, we compute the **scatter**

$$\tilde{s}_i^2 = \sum_{y \in \mathcal{Y}_i} (y - \tilde{m}_i)^2 \tag{40}$$

which is essentially a scaled sampled variance.

The Scatter

- To capture this variation, we compute the **scatter**

$$\tilde{s}_i^2 = \sum_{y \in \mathcal{Y}_i} (y - \tilde{m}_i)^2 \tag{40}$$

  which is essentially a scaled sampled variance.
- From this, we can estimate the variance of the pooled data:

$$\frac{1}{n}\left(\tilde{s}_1^2 + \tilde{s}_2^2\right) \quad . \tag{41}$$

- $\tilde{s}_1^2 + \tilde{s}_2^2$ is called the total **within-class scatter** of the projected samples.

Fisher Linear Discriminant

- The Fisher Linear Discriminant will select the $\mathbf{w}$ that maximizes

$$J(\mathbf{w}) = \frac{|\tilde{m}_1 - \tilde{m}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2} \quad . \tag{42}$$

- It does so independently of the magnitude of $\mathbf{w}$.

Fisher Linear Discriminant

- The Fisher Linear Discriminant will select the $\mathbf{w}$ that maximizes

$$J(\mathbf{w}) = \frac{|\tilde{m}_1 - \tilde{m}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2} \quad . \tag{42}$$

- It does so independently of the magnitude of $\mathbf{w}$.
- This term is the ratio of the distance between the projected means scaled by the within-class scatter (the variation of the data).
- Recall the similar term from earlier in the lecture which indicated the amount a feature will reduce the probability of error is proportional to the ratio of the difference of the means to the variance. FLD will choose the maximum...

Fisher Linear Discriminant

- The Fisher Linear Discriminant will select the $\mathbf{w}$ that maximizes

$$J(\mathbf{w}) = \frac{|\tilde{m}_1 - \tilde{m}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2} \quad . \tag{42}$$

- It does so independently of the magnitude of $\mathbf{w}$.
- This term is the ratio of the distance between the projected means scaled by the within-class scatter (the variation of the data).
- Recall the similar term from earlier in the lecture which indicated the amount a feature will reduce the probability of error is proportional to the ratio of the difference of the means to the variance. FLD will choose the maximum...
- We need to rewrite $J(\cdot)$ as a function of $\mathbf{w}$.

Within-Class Scatter Matrix

- Define scatter matrices $\mathbf{S}_i$:

$$\mathbf{S}_i = \sum_{x \in \mathcal{D}_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^\mathsf{T} \tag{43}$$

and

$$\mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2 \ . \tag{44}$$

Within-Class Scatter Matrix

- Define scatter matrices $\mathbf{S}_i$:

$$\mathbf{S}_i = \sum_{x \in \mathcal{D}_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^{\mathsf{T}} \tag{43}$$

and

$$\mathbf{S}_W = \mathbf{S}_1 + \mathbf{S}_2 \ . \tag{44}$$

- $\mathbf{S}_W$ is called the **within-class scatter matrix**.
- $\mathbf{S}_W$ is symmetric and positive semidefinite.
- In typical cases, when is $\mathbf{S}_W$ nonsingular?

Within-Class Scatter Matrix

- Deriving the sum of scatters.

$$\tilde{s}_i^2 = \sum_{x \in \mathcal{D}_i} \left(\mathbf{w}^\mathsf{T}\mathbf{x} - \mathbf{w}^\mathsf{T}\mathbf{m}_i \right)^2 \tag{45}$$

$$= \sum_{x \in \mathcal{D}_i} \mathbf{w}^\mathsf{T} \left(\mathbf{x} - \mathbf{m}_i \right) \left(\mathbf{x} - \mathbf{m}_i \right)^\mathsf{T} \mathbf{w} \tag{46}$$

$$= \mathbf{w}^\mathsf{T} \mathbf{S}_i \mathbf{w} \tag{47}$$

- We can therefore write the sum of the scatters as an explicit function of $\mathbf{w}$:

$$\tilde{s}_1^2 + \tilde{s}_2^2 = \mathbf{w}^\mathsf{T} \mathbf{S}_W \mathbf{w} \tag{48}$$

Between-Class Scatter Matrix

- The separation of the projected means obeys

$$(\tilde{m}_1 - \tilde{m}_2)^2 = \left(\mathbf{w}^\mathsf{T}\mathbf{m}_1 - \mathbf{w}^\mathsf{T}\mathbf{m}_2\right)^2 \tag{49}$$

$$= \mathbf{w}^\mathsf{T}(\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^\mathsf{T}\mathbf{w} \tag{50}$$

$$= \mathbf{w}^\mathsf{T}\mathbf{S}_B\mathbf{w} \tag{51}$$

- Here, $\mathbf{S}_B$ is called the **between-class scatter matrix**:

$$\mathbf{S}_B = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^\mathsf{T} \tag{52}$$

- $\mathbf{S}_B$ is also symmetric and positive semidefinite.
- When is $\mathbf{S}_B$ nonsingular?

Rewriting the FLD $J(\cdot)$

- We can rewrite our objective as a function of $\mathbf{w}$.

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathsf{T}}\mathbf{S}_B\mathbf{w}}{\mathbf{w}^{\mathsf{T}}\mathbf{S}_W\mathbf{w}} \tag{53}$$

- This is the generalized Rayleigh quotient.

Rewriting the FLD $J(\cdot)$

- We can rewrite our objective as a function of $\mathbf{w}$.

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathsf{T}}\mathbf{S}_B\mathbf{w}}{\mathbf{w}^{\mathsf{T}}\mathbf{S}_W\mathbf{w}} \tag{53}$$

- This is the generalized Rayleigh quotient.
- The vector $\mathbf{w}$ that maximizes $J(\cdot)$ must satisfy

$$\mathbf{S}_B\mathbf{w} = \lambda\mathbf{S}_W\mathbf{w} \tag{54}$$

which is a generalized eigenvalue problem.

Rewriting the FLD $J(\cdot)$

- We can rewrite our objective as a function of $\mathbf{w}$.

$$J(\mathbf{w}) = \frac{\mathbf{w}^{\mathsf{T}}\mathbf{S}_B\mathbf{w}}{\mathbf{w}^{\mathsf{T}}\mathbf{S}_W\mathbf{w}} \tag{53}$$

- This is the generalized Rayleigh quotient.
- The vector $\mathbf{w}$ that maximizes $J(\cdot)$ must satisfy

$$\mathbf{S}_B\mathbf{w} = \lambda\mathbf{S}_W\mathbf{w} \tag{54}$$

  which is a generalized eigenvalue problem.
- For nonsingular $\mathbf{S}_W$ (typical), we can write this as a standard eigenvalue problem:

$$\mathbf{S}_W^{-1}\mathbf{S}_B\mathbf{w} = \lambda\mathbf{w} \tag{55}$$

Simplifying

- Since $\|\mathbf{w}\|$ is not important and $\mathbf{S}_B\mathbf{w}$ is always in the direction of $(\mathbf{m}_1 - \mathbf{m}_2)$, we can simplify

$$\mathbf{w}^* = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2) \tag{56}$$

- $\mathbf{w}^*$ maximizes $J(\cdot)$ and is the Fisher Linear Discriminant.

Simplifying

- Since $\|\mathbf{w}\|$ is not important and $\mathbf{S}_B\mathbf{w}$ is always in the direction of $(\mathbf{m}_1 - \mathbf{m}_2)$, we can simplify

$$\mathbf{w}^* = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2) \tag{56}$$

- $\mathbf{w}^*$ maximizes $J(\cdot)$ and is the Fisher Linear Discriminant.
- The FLD converts a many-dimensional problem to a one-dimensional one.

Simplifying

- Since $\|\mathbf{w}\|$ is not important and $\mathbf{S}_B\mathbf{w}$ is always in the direction of $(\mathbf{m}_1 - \mathbf{m}_2)$, we can simplify

$$\mathbf{w}^* = \mathbf{S}_W^{-1}(\mathbf{m}_1 - \mathbf{m}_2) \tag{56}$$

- $\mathbf{w}^*$ maximizes $J(\cdot)$ and is the Fisher Linear Discriminant.
- The FLD converts a many-dimensional problem to a one-dimensional one.
- One still must find the threshold. This is easy for known densities, but not so easy in general.

A Classic PCA vs. FLD comparison

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- PCA maximizes the total scatter across all classes.
- PCA projections are thus optimal for reconstruction from a low dimensional basis, but not necessarily from a discrimination standpoint.
- FLD maximizes the ratio of the between-class scatter and the within-class scatter.
- FLD tries to "shape" the scatter to make it more effective for classification.

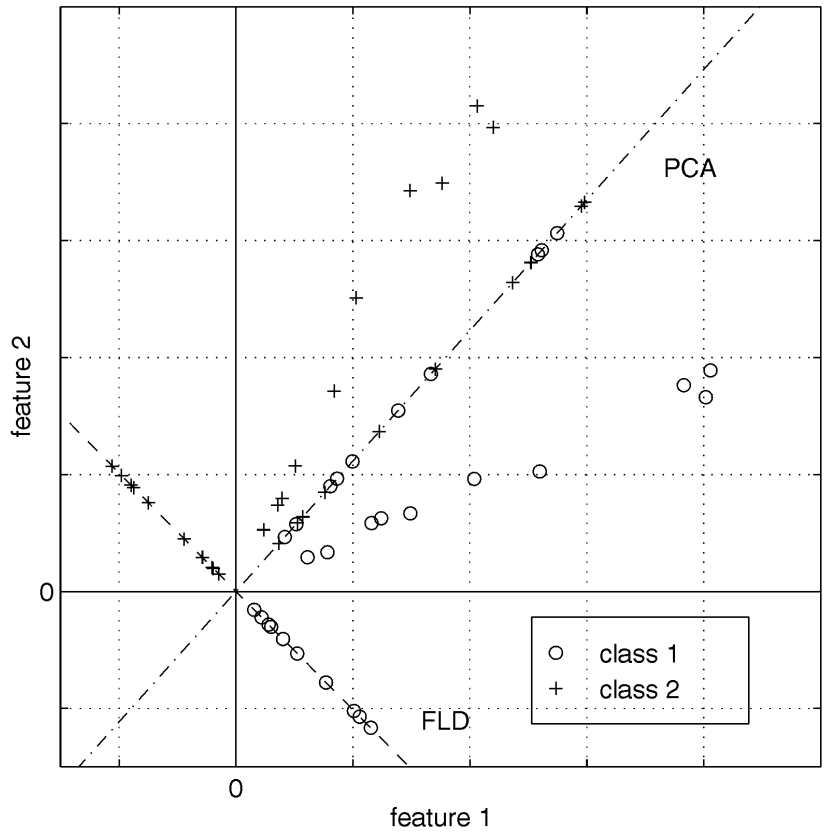


Fig. 2. A comparison of principal component analysis (PCA) and Fisher's linear discriminant (FLD) for a two class problem where data for each class lies near a linear subspace.

Case Study: EigenFaces versus FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Analysis of classic pattern recognition techniques (PCA and FLD) to do face recognition.
- Fixed pose but varying illumination.
- The variation in the resulting images caused by the varying illumination will nearly always dominate the variation caused by an identity change.

Fig. 1. The same person seen under different lighting conditions can appear dramatically different: In the left image, the dominant light source is nearly head-on; in the right image, the dominant light source is from above and to the right.

Case Study: EigenFaces versus FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

Is Linear Okay For Faces?

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Fact: all of the images of a Lambertian surface, taken from a fixed viewpoint, but under varying illumination, lie in a 3D linear subspace of the high-dimensional image space.

Is Linear Okay For Faces?

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Fact: all of the images of a Lambertian surface, taken from a fixed viewpoint, but under varying illumination, lie in a 3D linear subspace of the high-dimensional image space.
- But, in the presence of shadowing, specularities, and facial expressions, the above statement will not hold. This will ultimately result in deviations from the 3D linear subspace and worse classification accuracy.

Method 1: Correlation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Consider a set of N training images, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$.
- We know that each of the N images belongs to one of c classes and can define a $C(\cdot)$ function to map the image $\mathbf{x}$ into a class ω_c.

Method 1: Correlation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Consider a set of N training images, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$.
- We know that each of the N images belongs to one of c classes and can define a $C(\cdot)$ function to map the image $\mathbf{x}$ into a class ω_c.
- **Pre-Processing** – each image is normalized to have zero mean and unit variance.
- Why?

Method 1: Correlation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Consider a set of N training images, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$.
- We know that each of the N images belongs to one of c classes and can define a $C(\cdot)$ function to map the image $\mathbf{x}$ into a class ω_c.
- **Pre-Processing** – each image is normalized to have zero mean and unit variance.
- Why?
- Gets rid of the light source intensity and the effects of a camera's automatic gain control.

Method 1: Correlation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Consider a set of N training images, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$.
- We know that each of the N images belongs to one of c classes and can define a $C(\cdot)$ function to map the image $\mathbf{x}$ into a class ω_c.
- **Pre-Processing** – each image is normalized to have zero mean and unit variance.
- Why?
- Gets rid of the light source intensity and the effects of a camera's automatic gain control.
- For a query image $\mathbf{x}$, we select the class of the training image that is the nearest neighbor in the image space:

$$\mathbf{x}^* = \arg\min_{\{\mathbf{x}_i\}} \|\mathbf{x} - \mathbf{x}_i\| \quad \text{then decide } C(\mathbf{x}^*) \tag{57}$$

where we have **vectorized** each image.

Method 1: Correlation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- What are the advantages and disadvantages of the correlation based method in this context?

Method 1: Correlation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- What are the advantages and disadvantages of the correlation based method in this context?
- Computationally expensive.
- Require large amount of storage.
- Noise may play a role.
- Highly parallelizable.

Method 2: EigenFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997. The original paper is Turk and Pentland, "Eigenfaces for Recognition" Journal of Cognitive Neuroscience, 3(1). 1991.

- Quickly recall the main idea of PCA.
- Define a linear projection of the original n-d image space into an m-d space with $m < n$ or $m \ll n$, which yields new vectors $\mathbf{y}$:

$$\mathbf{y}_k = W^{\mathsf{T}} \mathbf{x}_k \qquad k = 1, 2, \ldots, N \tag{58}$$

where $W \in \mathbb{R}^{n \times m}$.

Method 2: EigenFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997. The original paper is Turk and Pentland, "Eigenfaces for Recognition" Journal of Cognitive Neuroscience, 3(1). 1991.

- Quickly recall the main idea of PCA.
- Define a linear projection of the original n-d image space into an m-d space with $m < n$ or $m \ll n$, which yields new vectors $\mathbf{y}$:

$$\mathbf{y}_k = W^{\mathsf{T}} \mathbf{x}_k \qquad k = 1, 2, \ldots, N \tag{58}$$

  where $W \in \mathbb{R}^{n \times m}$.
- Define the total scatter matrix S_T as

$$S_T = \sum_{k=1}^{N} (\mathbf{x}_k - \boldsymbol{\mu}) (\mathbf{x}_k - \boldsymbol{\mu})^{\mathsf{T}} \tag{59}$$

  where μ is the **sample mean image**.

Method 2: EigenFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997. The original paper is Turk and Pentland, "Eigenfaces for Recognition" Journal of Cognitive Neuroscience, 3(1). 1991.

- The scatter of the projected vectors $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$ is

$$W^\mathsf{T} S_T W \tag{60}$$

Method 2: EigenFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997. The original paper is Turk and Pentland, "Eigenfaces for Recognition" Journal of Cognitive Neuroscience, 3(1). 1991.

- The scatter of the projected vectors $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_N\}$ is

$$W^{\mathsf{T}} S_T W \tag{60}$$

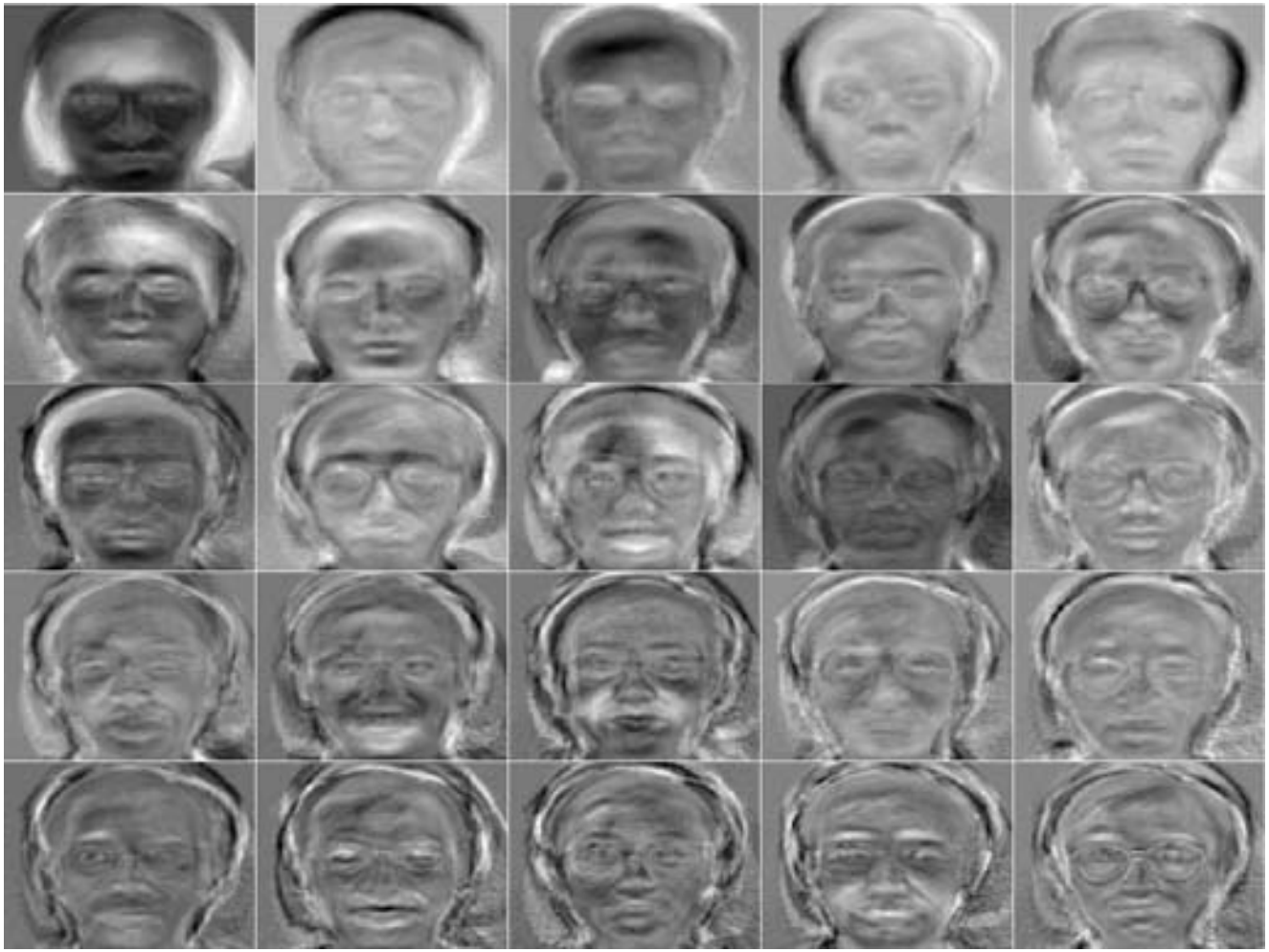
- W_{opt} is chosen to maximize the determinant of the total scatter matrix of the projected vectors:

$$W_{\text{opt}} = \arg\max_W |W^{\mathsf{T}} S_T W| \tag{61}$$

$$= \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_2 & \ldots & \mathbf{w}_m \end{bmatrix} \tag{62}$$

where $\{\mathbf{w}_i | i = 1, d, \ldots, m\}$ is the set of n-d eigenvectors of S_T corresponding to the largest m eigenvalues.

- An Example:



Source: http://www.cs.princeton.edu/ cdecoro/eigenfaces/. (not sure if this dataset include lighting variation...)

Method 2: EigenFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997. The original paper is Turk and Pentland, "Eigenfaces for Recognition" Journal of Cognitive Neuroscience, 3(1). 1991.

- What are the advantages and disadvantages of the eigenfaces method in this context?

Method 2: EigenFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997. The original paper is Turk and Pentland, "Eigenfaces for Recognition" Journal of Cognitive Neuroscience, 3(1). 1991.

- What are the advantages and disadvantages of the eigenfaces method in this context?
- The scatter being maximized is due not only to the between-class scatter that is useful for classification but also to the within-clas scatter, which is generally undesirable for classification.
- If PCA is presented faces with varying illumination, the projection matrix W_{opt} will contain principal components that retain the variation in lighting. If this variation is higher than the variation due to class identity, then PCA will suffer greatly for classification.
- Yields a more compact representation than the correlation-based method.

Method 3: Linear Subspaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Use Lambertian model directly.
- Consider a point p on a Lambertian surface illuminated by a point light source at infinity.
- Let $\mathbf{s} \in \mathbb{R}^3$ signify the product of the light source intensity with the unit vector for the light source direction.
- The image intensity of the surface at p when viewed by a camera is

$$E(p) = a(p)\mathbf{n}(p)^{\mathsf{T}}\mathbf{s} \tag{63}$$

  where $\mathbf{n}(p)$ is the unit inward normal vector to the surface at point p, and $a(p)$ is the albedo of the surface at p (a scalar).
- Hence, the image intensity of the point p is linear on $\mathbf{s}$.

Method 3: Linear Subspaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- So, if we assume no shadowing, given three images of a Lambertian surface from the same viewpoint under three known, linearly independent light source directions, the albedo and surface normal can be recovered.
- Alternatively, one can reconstruct the image of the surface under an arbitrary lighting direction by a linear combination of the three original images.
- This fact can be used for classification.

Method 3: Linear Subspaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- So, if we assume no shadowing, given three images of a Lambertian surface from the same viewpoint under three known, linearly independent light source directions, the albedo and surface normal can be recovered.
- Alternatively, one can reconstruct the image of the surface under an arbitrary lighting direction by a linear combination of the three original images.
- This fact can be used for classification.
- For each face (class) use three or more images taken under different lighting conditions to construct a 3D basis for the linear subspace.
- For recognition, compute the distance of a new image to each linear subspace and choose the face corresponding to the shortest distance.

Method 3: Linear Subspaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Pros and Cons?

Method 3: Linear Subspaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Pros and Cons?
- If there is no noise or shadowing, this method will achieve error free classification under any lighting conditions (and if the surface is indeed Lambertian).
- Faces inevitably have self-shadowing.
- Faces have expressions...
- Still pretty computationally expensive (linear in number of classes).

Method 4: FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Recall the Fisher Linear Discriminant setup.
- The between-class scatter matrix

$$S_B = \sum_{i=1}^{c} N_i (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^{\mathsf{T}} \tag{64}$$

- The within-class scatter matrix

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in \mathcal{D}_i} (x_k - \boldsymbol{\mu}_i)(x_k - \boldsymbol{\mu}_i)^{\mathsf{T}} \tag{65}$$

- The optimal projection W_{opt} is chosen as the matrix with orthonormal columns which maximizes the ratio of the determinant of the between-class scatter matrix of the projected vectors to the determinant of the within-class scatter of the projected vectors:

$$W_{\text{opt}} = \arg\max_{W} \frac{|W^{\mathsf{T}} S_B W|}{|W^{\mathsf{T}} S_W W|} \tag{66}$$

Method 4: FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- The eigenvectors $\{\mathbf{w}_i | i = 1, 2, \ldots, m\}$ corresponding to the m largest eigenvalues of the following generalized eigenvalue problem comprise W_{opt}:

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i, \quad i = 1, 2, \ldots, m \tag{67}$$

- This is a multi-class version of the FLD, which we will discuss in more detail.

Method 4: FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- The eigenvectors $\{\mathbf{w}_i | i = 1, 2, \ldots, m\}$ corresponding to the m largest eigenvalues of the following generalized eigenvalue problem comprise W_{opt}:

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i, \quad i = 1, 2, \ldots, m \tag{67}$$

- This is a multi-class version of the FLD, which we will discuss in more detail.
- In face recognition, things get a little more complicated because the within-class scatter matrix S_W is always singular.

Method 4: FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- The eigenvectors $\{\mathbf{w}_i | i = 1, 2, \ldots, m\}$ corresponding to the m largest eigenvalues of the following generalized eigenvalue problem comprise W_{opt}:

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i, \quad i = 1, 2, \ldots, m \tag{67}$$

- This is a multi-class version of the FLD, which we will discuss in more detail.
- In face recognition, things get a little more complicated because the within-class scatter matrix S_W is always singular.
- This is because the rank of S_W is at most $N - c$ and the number of images in the learning set are commonly much smaller than the number of pixels in the image.
- This means we can choose W such that the within-class scatter is exactly zero.

Method 4: FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- To overcome this, project the image set to a lower dimensional space so that the resulting within-class scatter matrix S_W is nonsingular.
- How?

Method 4: FisherFaces

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- To overcome this, project the image set to a lower dimensional space so that the resulting within-class scatter matrix S_W is nonsingular.
- How?
- PCA to first reduce the dimension to $N - c$ and the FLD to reduce it to $c - 1$.
- $W_{\text{opt}}^{\mathsf{T}}$ is given by the product $W_{\text{FLD}}^{\mathsf{T}} W_{\text{PCA}}^{\mathsf{T}}$ where

$$W_{\text{PCA}} = \arg\max_W |W^{\mathsf{T}} S_T W| \tag{68}$$

$$W_{\text{FLD}} = \arg\max_W \frac{|W^{\mathsf{T}} W_{\text{PCA}}^{\mathsf{T}} S_B W_{\text{PCA}} W|}{|W^{\mathsf{T}} W_{\text{PCA}}^{\mathsf{T}} S_W W_{\text{PCA}} W|} \tag{69}$$

Experiment 1 and 2: Variation in Lighting

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Hypothesis: face recognition algorithms will perform better if they exploit the fact that images of a Lambertian surface lie in a linear subspace.
- Used Hallinan's Harvard Database which sampled the space of light source directions in 15 degree incrememements.
- Used 330 images of five people (66 of each) and extracted five subsets.
- Classification is nearest neighbor in all cases.

Experiment 1 and 2: Variation in Lighting

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

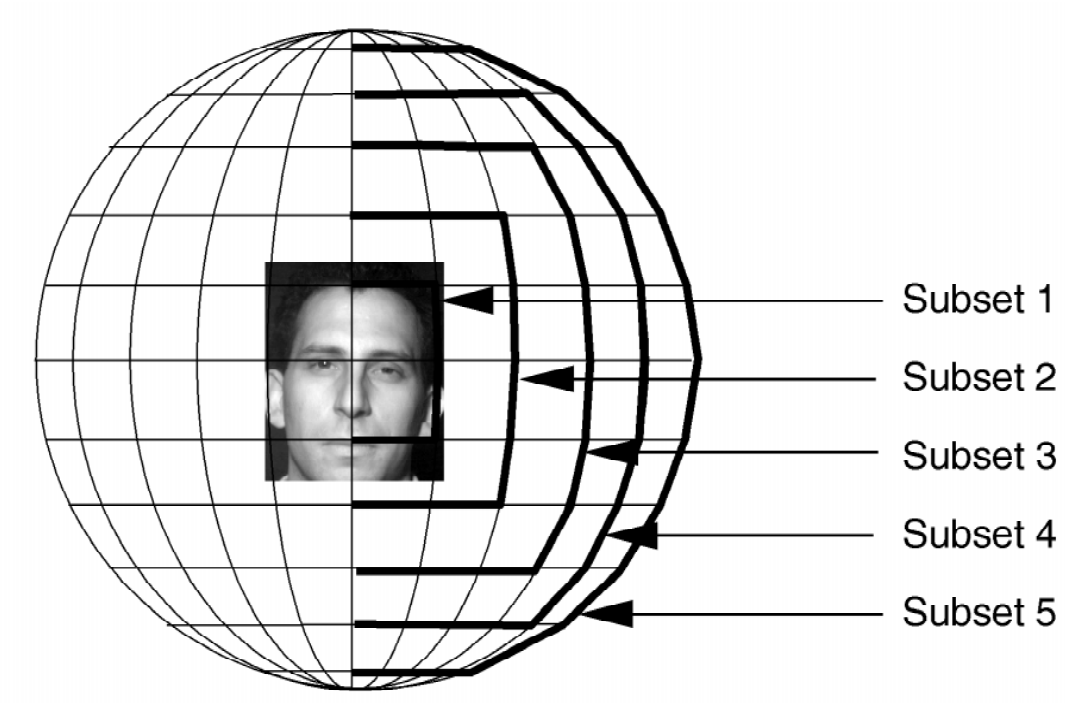
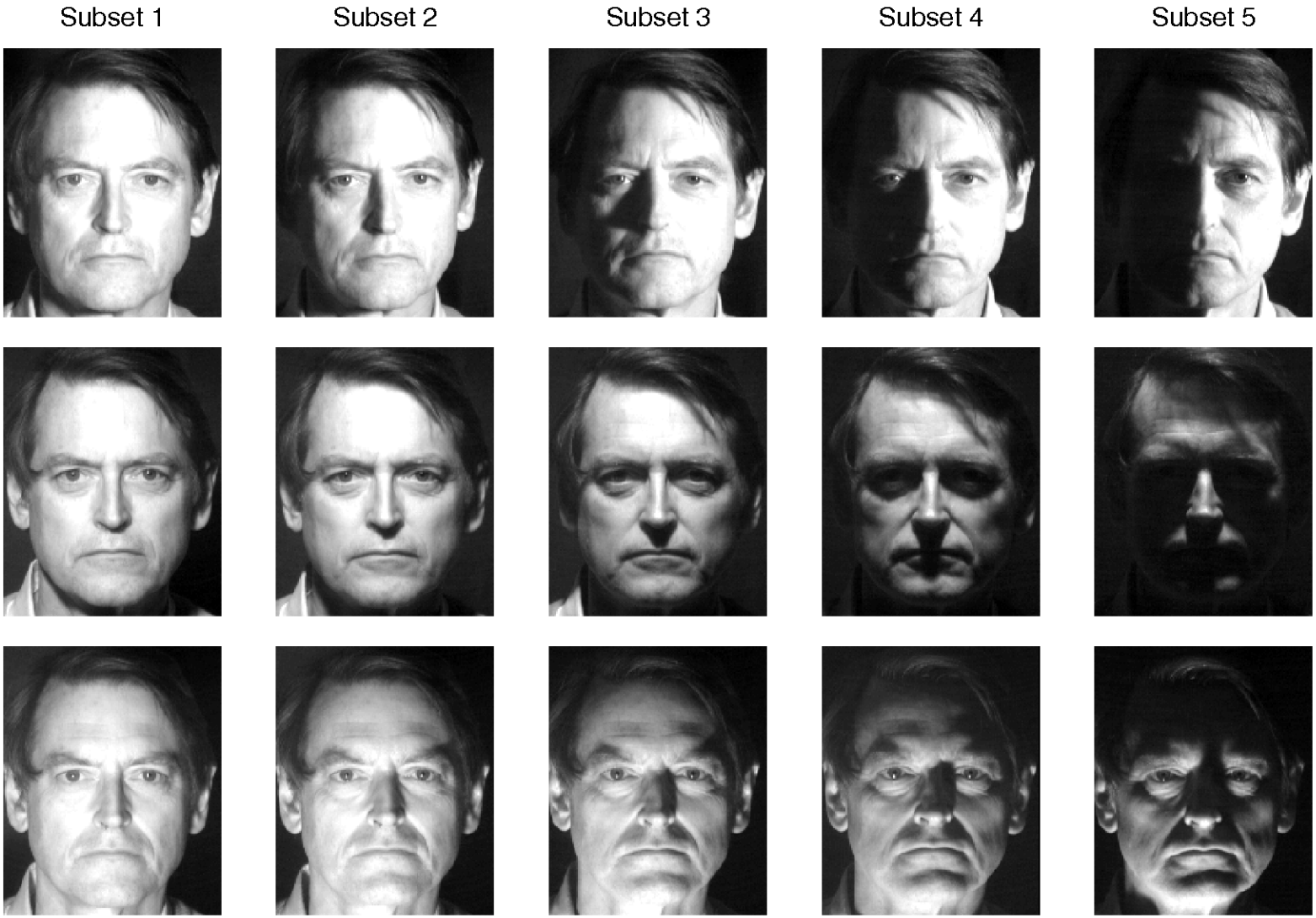


Fig. 3. The highlighted lines of longitude and latitude indicate the light source directions for Subsets 1 through 5. Each intersection of a longitudinal and latitudinal line on the right side of the illustration has a corresponding image in the database.

Experiment 1: Variation in Lighting

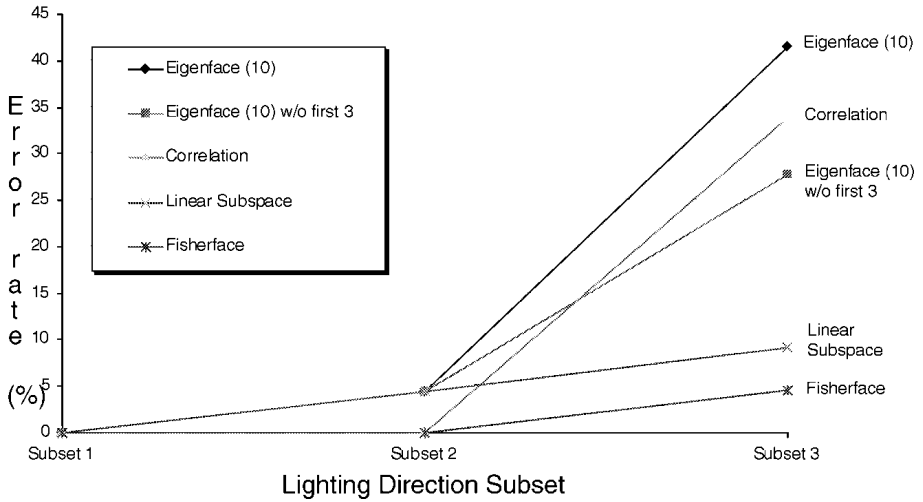
Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.



Experiment 1: Variation in Lighting – Extrapolation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Train on Subset 1.
- Test of Subsets 1,2,and 3.

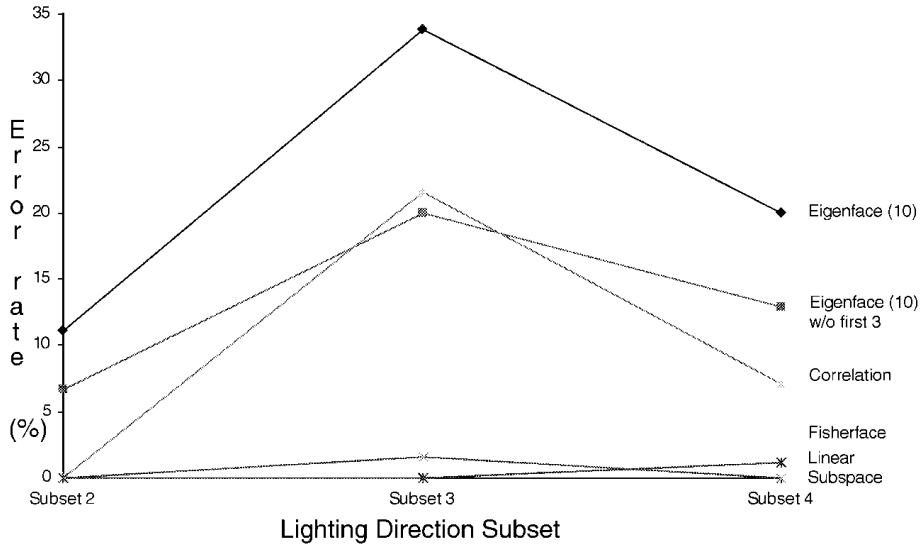


Extrapolating from Subset 1				
Method	Reduced Space	Error Rate (%)		
		Subset 1	Subset 2	Subset 3
Eigenface	4	0.0	31.1	47.7
	10	0.0	4.4	41.5
Eigenface w/o 1st 3	4	0.0	13.3	41.5
	10	0.0	4.4	27.7
Correlation	29	0.0	0.0	33.9
Linear Subspace	15	0.0	4.4	9.2
Fisherface	4	0.0	0.0	4.6

Experiment 2: Variation in Lighting – Interpolation

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- Train on Subsets 1 and 5.
- Test of Subsets 2, 3, and 4.



Interpolating between Subsets 1 and 5				
Method	Reduced Space	Error Rate (%)		
		Subset 2	Subset 3	Subset 4
Eigenface	4	53.3	75.4	52.9
	10	11.11	33.9	20.0
Eigenface w/o 1st 3	4	31.11	60.0	29.4
	10	6.7	20.0	12.9
Correlation	129	0.0	21.54	7.1
Linear Subspace	15	0.0	1.5	0.0
Fisherface	4	0.0	0.0	1.2

Experiment 1 and 2: Variation in Lighting

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- All of the algorithms perform perfectly when lighting is nearly frontal. However, when lighting is moved off axis, there is significant difference between the methods (spec. the class-specific methods and the Eigenface method).

Experiment 1 and 2: Variation in Lighting

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- All of the algorithms perform perfectly when lighting is nearly frontal. However, when lighting is moved off axis, there is significant difference between the methods (spec. the class-specific methods and the Eigenface method).
- Empirically demonstrated that the Eigenface method is equivalent to correlation when the number of Eigenfaces equals the size of the training set (Exp. 1).

Experiment 1 and 2: Variation in Lighting

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- All of the algorithms perform perfectly when lighting is nearly frontal. However, when lighting is moved off axis, there is significant difference between the methods (spec. the class-specific methods and the Eigenface method).
- Empirically demonstrated that the Eigenface method is equivalent to correlation when the number of Eigenfaces equals the size of the training set (Exp. 1).
- In the Eigenface method, removing the first three principal components results in better performance under variable lighting conditions.

Experiment 1 and 2: Variation in Lighting

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- All of the algorithms perform perfectly when lighting is nearly frontal. However, when lighting is moved off axis, there is significant difference between the methods (spec. the class-specific methods and the Eigenface method).
- Empirically demonstrated that the Eigenface method is equivalent to correlation when the number of Eigenfaces equals the size of the training set (Exp. 1).
- In the Eigenface method, removing the first three principal components results in better performance under variable lighting conditions.
- Linear Subspace has comparable error rates with the FisherFace method, but it requires 3x as much storage and takes three times as long.

Experiment 1 and 2: Variation in Lighting

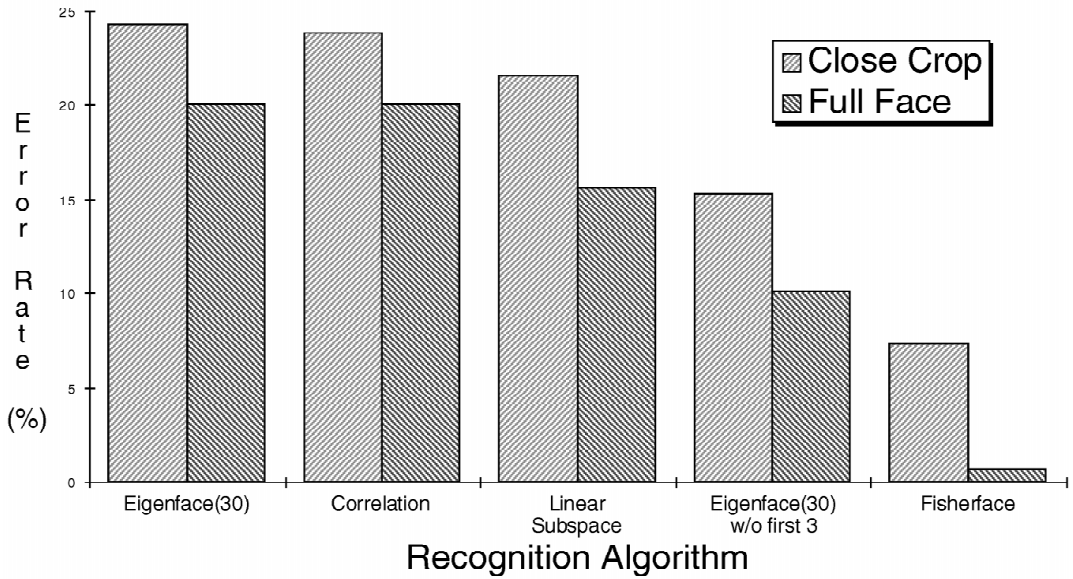
Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

- All of the algorithms perform perfectly when lighting is nearly frontal. However, when lighting is moved off axis, there is significant difference between the methods (spec. the class-specific methods and the Eigenface method).
- Empirically demonstrated that the Eigenface method is equivalent to correlation when the number of Eigenfaces equals the size of the training set (Exp. 1).
- In the Eigenface method, removing the first three principal components results in better performance under variable lighting conditions.
- Linear Subspace has comparable error rates with the FisherFace method, but it requires 3x as much storage and takes three times as long.
- The Fisherface method had error rates lower than the Eigenface method and required less computation time.

Experiment 3: Yale DB

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.

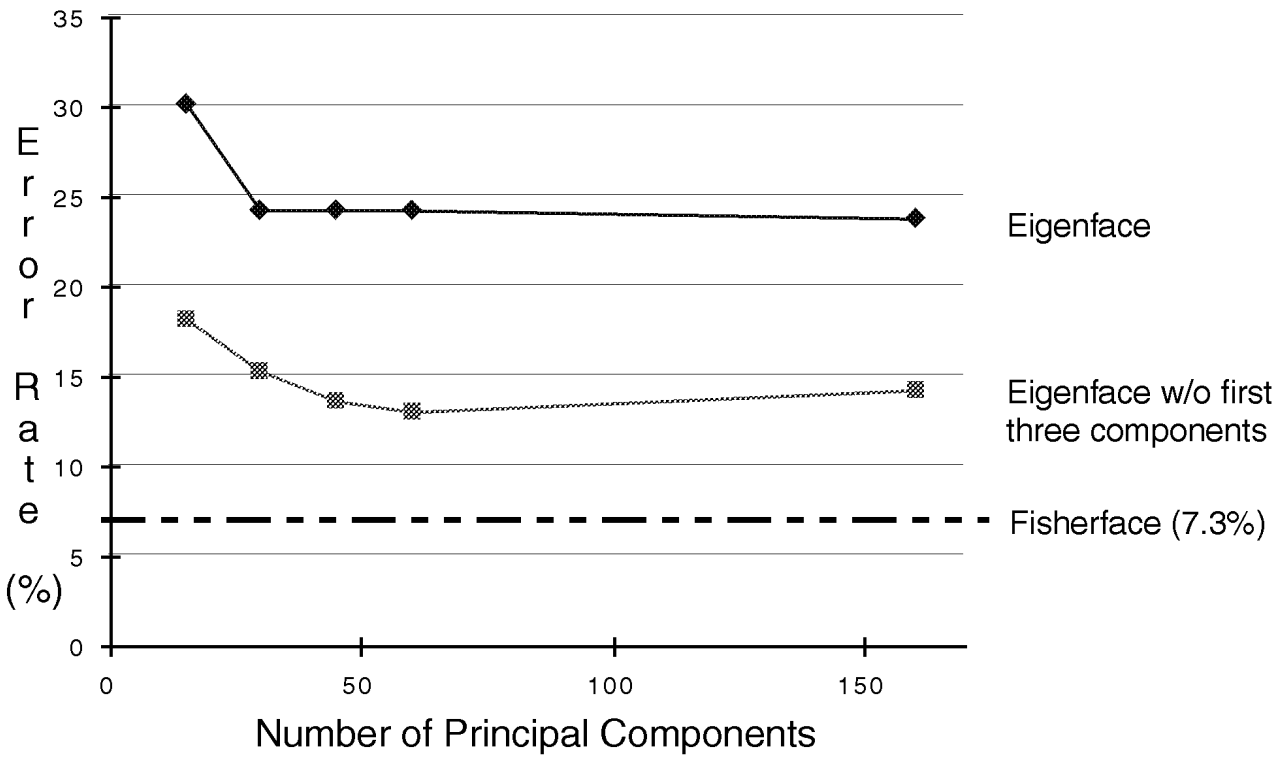
- Uses a second database of 16 subjects with ten images of images (5 varying in expression).



"Leaving-One-Out" of Yale Database			
Method	Reduced Space	Error Rate (%)	
		Close Crop	Full Face
Eigenface	30	24.4	19.4
Eigenface w/o 1st 3	30	15.3	10.8
Correlation	160	23.9	20.0
Linear Subspace	48	21.6	15.6
Fisherface	15	7.3	0.6

Experiment 3: Comparing Variation with Number of Principal Components

Source: Belhumeur et al. IEEE TPAMI 19(7) 711–720. 1997.



Can PCA outperform FLD for recognition?

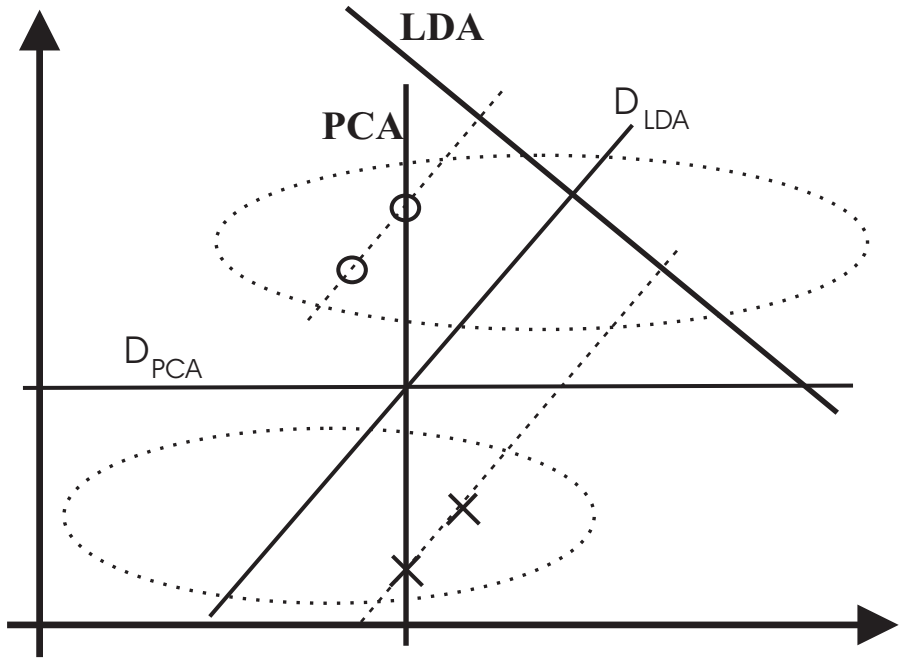
Source: Martínez and Kak. PCA versus LDA. PAMI 23(2), 2001.

- There may be situations in which PCA might outperform FLD.
- Can you think of such a situation?

Can PCA outperform FLD for recognition?

Source: Martínez and Kak. PCA versus LDA. PAMI 23(2), 2001.

- There may be situations in which PCA might outperform FLD.
- Can you think of such a situation?

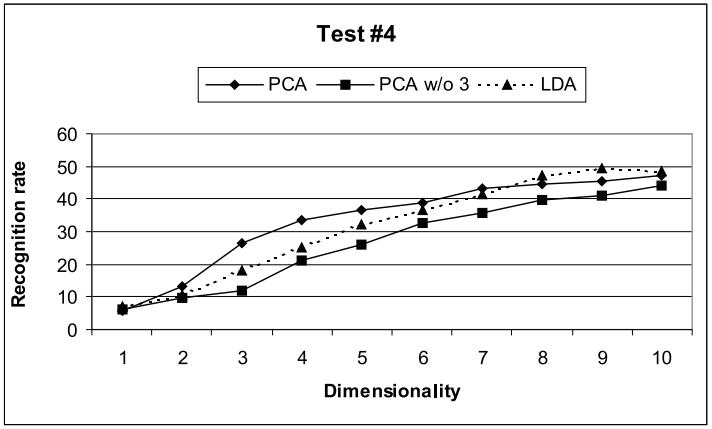


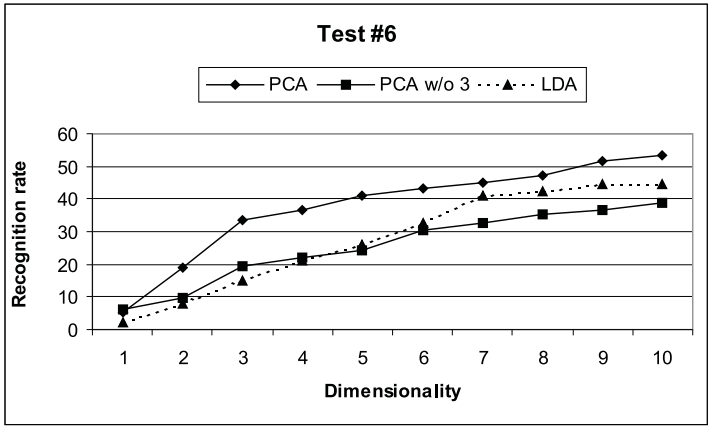
- When we have few training data, then it may be preferable to describe total scatter.

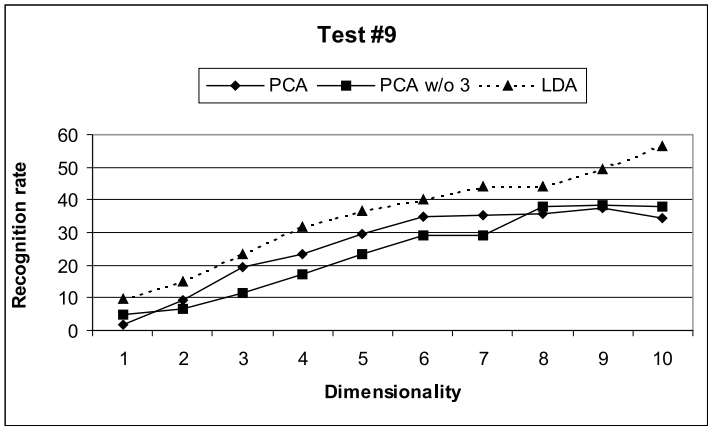
Can PCA outperform FLD for recognition?

Source: Martínez and Kak. PCA versus LDA. PAMI 23(2), 2001.

- They tested on the AR face database and found affirmative results.







Multiple Discriminant Analysis

- We can generalize the Fisher Linear Discriminant to multiple classes.
- Indeed we saw it done in the Case Study on Faces. But, let's cover it with some rigor.
- Let's make sure we're all at the same place: How many discriminant functions will be involved in a c class problem?

Multiple Discriminant Analysis

- We can generalize the Fisher Linear Discriminant to multiple classes.
- Indeed we saw it done in the Case Study on Faces. But, let's cover it with some rigor.
- Let's make sure we're all at the same place: How many discriminant functions will be involved in a c class problem?
- **Key:** There will be $c-1$ projection functions for a c class problem and hence the projection will be from a d-dimensional space to a $(c-1)$-dimensional space.
- d must be greater than or equal to c.

MDA The Projection

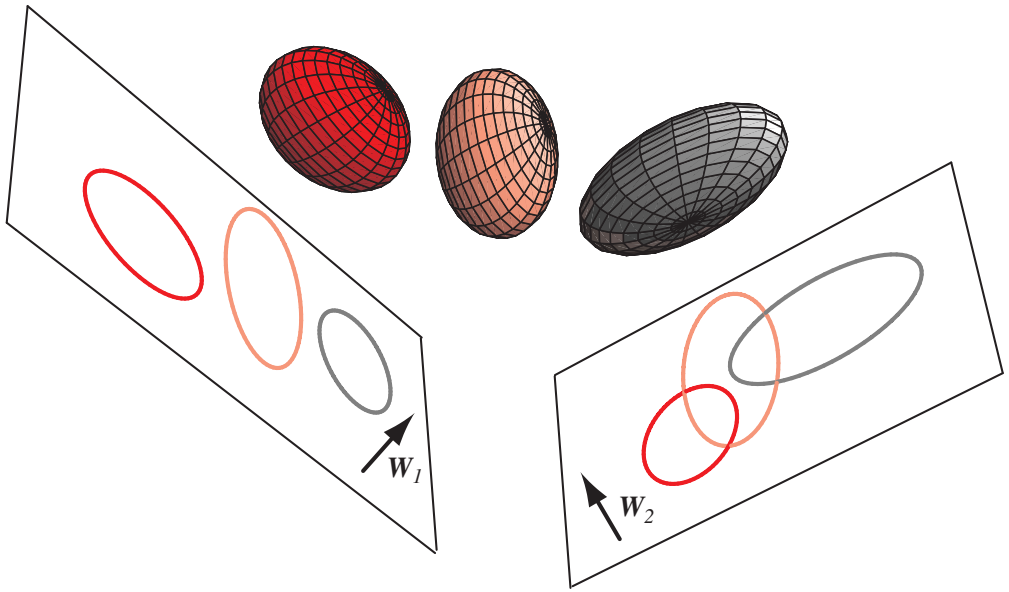
The projection is accomplished by $c-1$ discriminant functions:

$$y_i = \mathbf{w}_i^{\mathsf{T}}\mathbf{x} \qquad i = 1, \ldots, c-1 \tag{70}$$

which is summarized in matrix form as

$$\mathbf{y} = \mathbf{W}^{\mathsf{T}}\mathbf{x} \tag{71}$$

where $\mathbf{W}$ is the $d \times (c-1)$ projection function.



MDA Within-Class Scatter Matrix

- The generalization for the Within-Class Scatter Matrix is straightforward:

$$\mathbf{S}_W = \sum_{i=1}^{c} \mathbf{S}_i \tag{72}$$

where, as before,

$$\mathbf{S}_i = \sum_{x \in \mathcal{D}_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^{\mathsf{T}}$$

and

$$\mathbf{m}_i = \frac{1}{n_i} \sum_{x \in \mathcal{D}_i} \mathbf{x} \ .$$

MDA Between-Class Scatter Matrix

- The between-class scatter matrix $\mathbf{S}_B$ is not so easy to generalize.

MDA Between-Class Scatter Matrix

- The between-class scatter matrix $\mathbf{S}_B$ is not so easy to generalize.
- Let's define a **total mean vector**

$$\mathbf{m} = \frac{1}{n}\sum_{\mathbf{x}} \mathbf{x} = \frac{1}{n}\sum_{i=1}^{c} n_i \mathbf{m}_i \tag{73}$$

- Recall then **total scatter matrix**

$$\mathbf{S}_T = \sum_{\mathbf{x}} (\mathbf{x}-\mathbf{m})(\mathbf{x}-\mathbf{m})^{\mathsf{T}} \tag{74}$$

MDA Between-Class Scatter Matrix

- Then it follows that

$$\mathbf{S}_T = \sum_{i=1}^{c} \sum_{\mathbf{x} \in \mathcal{D}_i} (\mathbf{x} - \mathbf{m}_i + \mathbf{m}_i - \mathbf{m})(\mathbf{x} - \mathbf{m}_i + \mathbf{m}_i - \mathbf{m})^{\mathsf{T}} \tag{75}$$

$$= \sum_{i=1}^{c} \sum_{\mathbf{x} \in \mathcal{D}_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^{\mathsf{T}} + \sum_{i=1}^{c} \sum_{\mathbf{x} \in \mathcal{D}_i} (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^{\mathsf{T}} \tag{76}$$

$$= \mathbf{S}_W + \sum_{i=1}^{c} n_i (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^{\mathsf{T}} \tag{77}$$

MDA Between-Class Scatter Matrix

- Then it follows that

$$\mathbf{S}_T = \sum_{i=1}^{c} \sum_{\mathbf{x} \in \mathcal{D}_i} (\mathbf{x} - \mathbf{m}_i + \mathbf{m}_i - \mathbf{m})(\mathbf{x} - \mathbf{m}_i + \mathbf{m}_i - \mathbf{m})^{\mathsf{T}} \tag{75}$$

$$= \sum_{i=1}^{c} \sum_{\mathbf{x} \in \mathcal{D}_i} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^{\mathsf{T}} + \sum_{i=1}^{c} \sum_{\mathbf{x} \in \mathcal{D}_i} (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^{\mathsf{T}} \tag{76}$$

$$= \mathbf{S}_W + \sum_{i=1}^{c} n_i (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^{\mathsf{T}} \tag{77}$$

- So, we can define this second term as a **generalized between-class scatter matrix**.
- The total scatter is the the sum of the within-class scatter and the between-class scatter.

MDA Objective

Page 1

- We again seek a criterion that will maximize the between-class scatter of the projected vectors to the within-class scatter of the projected vectors.
- Recall that we can write down the scatter in terms of these projections:

$$\widetilde{\mathbf{m}}_i = \frac{1}{n_i} \sum_{\mathbf{y} \in \mathcal{Y}_i} \mathbf{y} \qquad \text{and} \qquad \widetilde{\mathbf{m}} = \frac{1}{n} \sum_{i=1}^{c} n_i \widetilde{\mathbf{m}}_i \tag{78}$$

$$\widetilde{\mathbf{S}}_W = \sum_{i=1}^{c} \sum_{\mathbf{y} \in \mathcal{Y}_i} (\mathbf{y} - \widetilde{\mathbf{m}}_i)(\mathbf{y} - \widetilde{\mathbf{m}}_i)^{\mathsf{T}} \tag{79}$$

$$\widetilde{\mathbf{S}}_B = \sum_{i=1}^{c} n_i (\widetilde{\mathbf{m}}_i - \widetilde{\mathbf{m}})(\widetilde{\mathbf{m}}_i - \widetilde{\mathbf{m}})^{\mathsf{T}} \tag{80}$$

MDA Objective

Page 2

- Then, we can show

$$\widetilde{\mathbf{S}}_W = \mathbf{W}^\mathsf{T}\mathbf{S}_W\mathbf{W} \tag{81}$$

$$\widetilde{\mathbf{S}}_B = \mathbf{W}^\mathsf{T}\mathbf{S}_B\mathbf{W} \tag{82}$$

MDA Objective

Page 2

- Then, we can show

$$\widetilde{\mathbf{S}}_W = \mathbf{W}^\mathsf{T}\mathbf{S}_W\mathbf{W} \tag{81}$$

$$\widetilde{\mathbf{S}}_B = \mathbf{W}^\mathsf{T}\mathbf{S}_B\mathbf{W} \tag{82}$$

- A simple scalar measure of scatter is the **determinant** of the scatter matrix. The determinant is the product of the eigenvalues and thus is the product of the variation along the principal directions.

$$J(\mathbf{W}) = \frac{|\widetilde{\mathbf{S}}_B|}{|\widetilde{\mathbf{S}}_W|} = \frac{|\mathbf{W}^\mathsf{T}\mathbf{S}_B\mathbf{W}|}{|\mathbf{W}^\mathsf{T}\mathbf{S}_W\mathbf{W}|} \tag{83}$$

MDA Solution

- The columns of an optimal $\mathbf{W}$ are the generalized eigenvectors that correspond to the largest eigenvalues in

$$\mathbf{S}_B \mathbf{w}_i = \lambda_i \mathbf{S}_W \mathbf{w}_i \tag{84}$$

MDA Solution

- The columns of an optimal $\mathbf{W}$ are the generalized eigenvectors that correspond to the largest eigenvalues in

$$\mathbf{S}_B \mathbf{w}_i = \lambda_i \mathbf{S}_W \mathbf{w}_i \tag{84}$$

- If $\mathbf{S}_W$ is nonsingular, then this can be converted to a conventional eigenvalue problem. Or, we could notice that the rank of $\mathbf{S}_B$ is at most $c-1$ and do some clever algebra...

MDA Solution

- The columns of an optimal $\mathbf{W}$ are the generalized eigenvectors that correspond to the largest eigenvalues in

$$\mathbf{S}_B \mathbf{w}_i = \lambda_i \mathbf{S}_W \mathbf{w}_i \tag{84}$$

- If $\mathbf{S}_W$ is nonsingular, then this can be converted to a conventional eigenvalue problem. Or, we could notice that the rank of $\mathbf{S}_B$ is at most $c-1$ and do some clever algebra...
- The solution for $\mathbf{W}$ is, however, not unique and would allow arbitrary scaling and rotation, but these would not change the ultimate classification.

IMPCA

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- **Observation:** To apply PCA and FLD on images, we need to first "vectorize" them, which can lead to high-dimensional vectors. Solving the associated eigen-problems is a very time-consuming process.

IMPCA

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- **Observation:** To apply PCA and FLD on images, we need to first "vectorize" them, which can lead to high-dimensional vectors. Solving the associated eigen-problems is a very time-consuming process.
- **So, can we apply PCA on the images directly?**

IMPCA

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- **Observation:** To apply PCA and FLD on images, we need to first "vectorize" them, which can lead to high-dimensional vectors. Solving the associated eigen-problems is a very time-consuming process.
- **So, can we apply PCA on the images directly?**
- Yes!
- This will be accomplished by what the authors' call the **image total covariance matrix**.

IMPCA: Problem Set-Up

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- Define $\mathbf{A} \in \mathbb{R}^{m \times n}$ as our image.
- Let $\mathbf{w}$ denote an n-dimensional column vector, which will represent the subspace onto which we will project an image.

$$\mathbf{y} = \mathbf{A}\mathbf{w} \tag{85}$$

  which yields m-dimensional vector $\mathbf{y}$.
- You might think of $\mathbf{w}$ this as a "feature selector."
- We, again, want to maximize the total scatter...

IMPCA: Image Total Scatter Matrix

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- We have M total data samples.
- The sample mean image is

$$\mathbf{M} = \frac{1}{M}\sum_{j=1}^{M}\mathbf{A}_j \tag{86}$$

- And the projected sample mean is

$$\tilde{\mathbf{m}} = \frac{1}{M}\sum_{j=1}^{M}\mathbf{y}_j \tag{87}$$

$$= \frac{1}{M}\sum_{j=1}^{M}\mathbf{A}_j\mathbf{w} \tag{88}$$

$$= \mathbf{M}\mathbf{w} \tag{89}$$

IMPCA: Image Total Scatter Matrix

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- The scatter of the projected samples is

$$\widetilde{\mathbf{S}} = \sum_{j=1}^{M} (\mathbf{y}_j - \widetilde{\mathbf{m}})(\mathbf{y}_j - \widetilde{\mathbf{m}})^{\mathsf{T}} \tag{90}$$

$$= \sum_{j=1}^{M} \left[(\mathbf{A}_j - \mathbf{M})\mathbf{w}\right]\left[(\mathbf{A}_j - \mathbf{M})\mathbf{w}\right]^{\mathsf{T}} \tag{91}$$

$$tr(\widetilde{\mathbf{S}}) = \mathbf{w}^{\mathsf{T}} \left(\sum_{j=1}^{M} (\mathbf{A}_j - \mathbf{M})^{\mathsf{T}} (\mathbf{A}_j - \mathbf{M}) \right) \mathbf{w} \tag{92}$$

IMPCA: Image Total Scatter Matrix

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- So, the **image total scatter matrix** is

$$\mathbf{S}_I = \sum_{j=1}^{M} (\mathbf{A}_j - \mathbf{M})^{\mathsf{T}} (\mathbf{A}_j - \mathbf{M}) \tag{93}$$

IMPCA: Image Total Scatter Matrix

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- So, the **image total scatter matrix** is

$$\mathbf{S}_I = \sum_{j=1}^{M} (\mathbf{A}_j - \mathbf{M})^{\mathsf{T}} (\mathbf{A}_j - \mathbf{M}) \tag{93}$$

- And, a suitable criterion, in a form we've seen before is

$$J_I(\mathbf{w}) = \mathbf{w}^{\mathsf{T}} \mathbf{S}_I \mathbf{w} \tag{94}$$

IMPCA: Image Total Scatter Matrix

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- So, the **image total scatter matrix** is

$$\mathbf{S}_I = \sum_{j=1}^{M} (\mathbf{A}_j - \mathbf{M})^{\mathsf{T}} (\mathbf{A}_j - \mathbf{M}) \tag{93}$$

- And, a suitable criterion, in a form we've seen before is

$$J_I(\mathbf{w}) = \mathbf{w}^{\mathsf{T}} \mathbf{S}_I \mathbf{w} \tag{94}$$

- We know already that the vectors $\mathbf{w}$ that maximize J_I are the orthonormal eigenvectors of $\mathbf{S}_I$ corresponding to the largest eigenvalues of $\mathbf{S}_I$.

IMPCA: Experimental Results

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- Dataset is the ORL database
  - 10 different images taken of 40 individuals.
  - Facial expressions and details are varying.
  - Even the pose can vary slightly: 20 degree rotation/tilt and 10% scale.
  - Size is normalized (92×112).

- The first five images of each person are used for training and the second five are used for testing.

IMPCA: Experimental Results

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- IMPCA varying the number of extracted eigenvectors (NN classifier):

Projection vector number	1	2	3	4	5
Minimum distance	73.0	83.0	86.5	88.5	88.5
Nearest neighbor	85.0	92.0	93.5	94.5	94.5

Projection vector number	6	7	8	9	10
Minimum distance	88.5	90.0	90.5	91.0	91.0
Nearest neighbor	95.0	95.0	95.5	93.5	94.0

IMPCA: Experimental Results

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- IMPCA vs. EigenFaces vs. FisherFaces for Recognition

Recognition rate	Eigenfaces	Fisherfaces	IMPCA
Minimum distance	89.5% (46)	88.5% (39)	91.0%
Nearest neighbor	93.5% (37)	88.5% (39)	95.5%

IMPCA: Experimental Results

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- IMPCA vs. EigenFaces vs. FisherFaces for Recognition

Recognition rate	Eigenfaces	Fisherfaces	IMPCA
Minimum distance	89.5% (46)	88.5% (39)	91.0%
Nearest neighbor	93.5% (37)	88.5% (39)	95.5%

- IMPCA vs. EigenFaces vs. FisherFaces for Speed

Time (s)	Feature extraction time	Classification time	Total time
Eigenfaces (37)	371.79	5.16	376.95
Fisherfaces (39)	378.10	5.27	383.37
IMPCA (112×8)	27.14	25.04	52.18

IMPCA: Discussion

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- There is a clear speed-up because the amount of computation has been greatly reduced.

IMPCA: Discussion

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- There is a clear speed-up because the amount of computation has been greatly reduced.
- However, this speed-up comes at some cost. What is that cost?

IMPCA: Discussion

Source: Yang and Yang: IMPCA vs. PCA, Pattern Recognition v35, 2002.

- There is a clear speed-up because the amount of computation has been greatly reduced.
- However, this speed-up comes at some cost. What is that cost?
- Why does IMPCA work in this case?
- What is IMPCA really doing?

Locally Linear Embedding

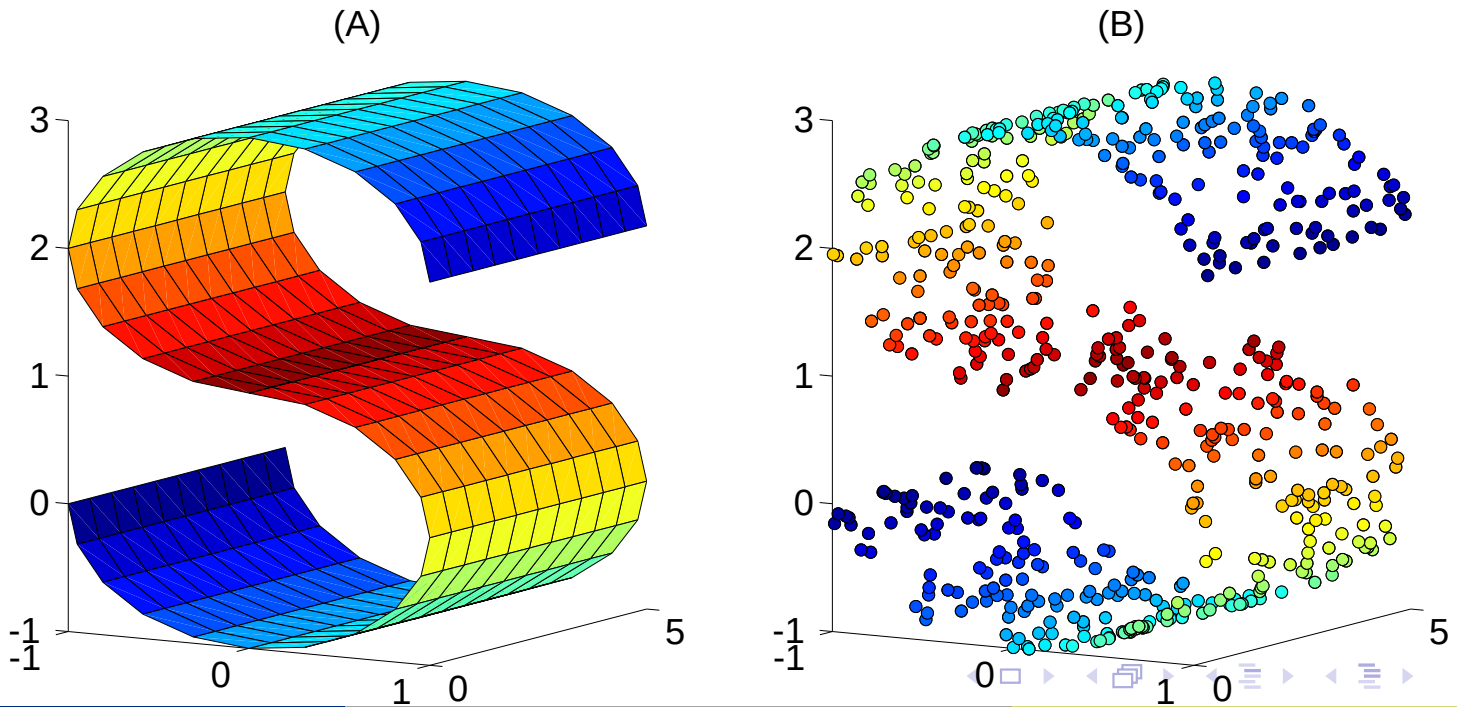
Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- So far, we have covered a few methods for dimension reduction that all make the underlying assumption the data in high dimension lives on a planar manifold in the lower dimension.
  - The methods are easy to implement.
  - The methods do not suffer from local minima.

Locally Linear Embedding

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- So far, we have covered a few methods for dimension reduction that all make the underlying assumption the data in high dimension lives on a planar manifold in the lower dimension.
  - The methods are easy to implement.
  - The methods do not suffer from local minima.
- But, what if the data is non-linear?



The Non-Linear Problem

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- In non-linear dimension reduction, one must discover the global internal coordinates of the manifold without signals that explicitly indicate how the data should be embedded in the lower dimension (or even how many dimensions should be used).

The Non-Linear Problem

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- In non-linear dimension reduction, one must discover the global internal coordinates of the manifold without signals that explicitly indicate how the data should be embedded in the lower dimension (or even how many dimensions should be used).
- The LLE way of doing this is to make the assumption that, given enough data samples, the local frame of a particular point $\mathbf{x}_i$ is linear. LLE proceeds to preserve this local structure while simultaneously reducing the global dimension (indeed preserving the local structure gives LLE the necessary constraints to discover the manifold).

LLE: Neighborhoods

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- Suppose we have n real-valued vectors $\mathbf{x}_i$ in dataset $\mathcal{D}$ each of dimension D. We assume these data are sampled from some smooth underlying manifold.

LLE: Neighborhoods

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- Suppose we have n real-valued vectors $\mathbf{x}_i$ in dataset $\mathcal{D}$ each of dimension D. We assume these data are sampled from some smooth underlying manifold.
- Furthermore, we assume that each data point and its neighbors lie on or close to a locally linear patch of the manifold.

LLE: Neighborhoods

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- Suppose we have n real-valued vectors $\mathbf{x}_i$ in dataset $\mathcal{D}$ each of dimension D. We assume these data are sampled from some smooth underlying manifold.
- Furthermore, we assume that each data point and its neighbors lie on or close to a locally linear patch of the manifold.
- LLE characterizes the **local geometry** of these patches by linear coefficients that reconstruct each data point from its neighbors.
  - If the neighbors form the D-dimensional simplex, then these coefficients form the barycentric coordinates of the data point.
  - In the simplest form of LLE, one identifies K such nearest neighbors based on the Euclidean distance.
  - But, one can use all points in a ball of fixed radius, or even more sophisticated metrics.

LLE: Neighborhoods

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- We can write down the reconstruction error with the following cost function:

$$J_{\text{LLE}_1}(\mathbf{W}) = \sum_i \left| \mathbf{x}_i - \sum_j W_{ij}\mathbf{x}_j \right|^2 \tag{95}$$

- Notice that each row of the weight matrix $\mathbf{W}$ will be nonzero for only K columns; i.e., $\mathbf{W}$ is a quite sparse matrix. I.e., if we define the set $N(\mathbf{x}_i)$ as the K neighbors of $\mathbf{x}_i$, then we enforce

$$W_{ij} = 0 \quad \text{if} \quad \mathbf{x}_j \notin N(\mathbf{x}_i) \tag{96}$$

- We will enforce that the rows sum to 1, i.e., $\sum_j W_{ij} = 1$. (This is for invariance.)

LLE: Neighborhoods

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- Note that the constrained weights that minimize these reconstruction errors obey important symmetries: for any data point, they are invariant to rotations, rescalings, and translations of that data point and its neighbors.

LLE: Neighborhoods

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- Note that the constrained weights that minimize these reconstruction errors obey important symmetries: for any data point, they are invariant to rotations, rescalings, and translations of that data point and its neighbors.
- This means that these weights characterize intrinsic geometric properties of the neighborhood as opposed to any properties that depend on a particular frame of reference.

LLE: Neighborhoods

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- Note that the constrained weights that minimize these reconstruction errors obey important symmetries: for any data point, they are invariant to rotations, rescalings, and translations of that data point and its neighbors.
- This means that these weights characterize intrinsic geometric properties of the neighborhood as opposed to any properties that depend on a particular frame of reference.
- If we suppose the data lie on or near a smooth nonlinear manifold of dimension $d \ll D$. Then, to a good approximation, there exists a linear mapping (a translation, rotation, and scaling) that maps the high dimensional coordinates of each neighborhood to global internal coordinates on the manifold.

LLE: Neighborhoods

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- So, we expect that the characterization of the local neighborhoods ($\mathbf{W}$) in the original space to be equally valid for the local patches on the manifold.
- In other words, the same weights W_{ij} that reconstruct point $\mathbf{x}_i$ in the original space should also reconstruct it in the embedded manifold coordinate system.
- First, let's solve for the weights. And, then we'll see how to use this point to ultimately compute the global dimension reduction.

Solving for the Weights

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- Solving for the weights $\mathbf{W}$ is a constrained least-squares problem.
- Consider a particular $\mathbf{x}$ with K nearest neighbors $\boldsymbol{\eta}_j$ and weights w_j which sum to one. We can write the reconstruction error as

$$\epsilon = \left| \mathbf{x} - \sum_j w_j \boldsymbol{\eta}_j \right|^2 \tag{97}$$

$$= \left| \sum_j w_j (\mathbf{x} - \boldsymbol{\eta}_j) \right|^2 \tag{98}$$

$$= \sum_{jk} w_j w_k C_{jk} \tag{99}$$

where C_{jk} is the covariance $(\mathbf{x} - \boldsymbol{\eta}_j)(\mathbf{x} - \boldsymbol{\eta}_k)^\mathsf{T}$.

Solving for the Weights

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- We need to add a Lagrange multiplier to enforce the constraint $\sum_j w_j = 1$ and then the weights can be solved in closed form.
- The optimal weights, in terms of the local covariance matrix, are

$$w_j = \frac{\sum_k C_{jk}^{-1}}{\sum_{lm} C_{lm}^{-1}} \quad . \tag{100}$$

Solving for the Weights

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- We need to add a Lagrange multiplier to enforce the constraint $\sum_j w_j = 1$ and then the weights can be solved in closed form.
- The optimal weights, in terms of the local covariance matrix, are

$$w_j = \frac{\sum_k C_{jk}^{-1}}{\sum_{lm} C_{lm}^{-1}} \quad . \tag{100}$$

- But, rather than explicitly inverting the covariance matrix, one can solve the linear system

$$\sum_k C_{jk} w_k = 1 \tag{101}$$

and rescale the weights so that they sum to one.

Stage 2: Neighborhood Preserving Mapping

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- Each high-dimensional input vector $\mathbf{x}_i$ is mapped to a low-dimension vector $\mathbf{y}_i$ representing the global internal coordinates on the manifold.
- LLE does this by choosing the d-dimension coordinates $\mathbf{y}_i$ to minimize the embedding cost function:

$$J_{\text{LLE}_2}(\mathbf{y}) = \sum_i \left| \mathbf{y}_i - \sum_j W_{ij} \mathbf{y}_j \right|^2 \tag{102}$$

- The basis for the cost function is the same—**locally linear reconstruction errors**—but here, the weights $\mathbf{W}$ are fixed and the coordinates $\mathbf{y}$ are optimized.

Stage 2: Neighborhood Preserving Mapping

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- This defines a quadratic:

$$J_{\text{LLE}_2}(\mathbf{y}) = \sum_i \left| \mathbf{y}_i - \sum_j W_{ij}\mathbf{y}_j \right|^2 \tag{103}$$

$$= \sum_{ij} M_{ij}(\mathbf{y}_i^{\mathsf{T}}\mathbf{y}_j) \tag{104}$$

where

$$M_{ij} = \delta_{ij} - W_{ij} - W_{ji} + \sum_k W_{ki}W_{kj} \tag{105}$$

with δ_{ij} is 1 if $j \neq i$ and 0 otherwise.

Stage 2: Neighborhood Preserving Mapping

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- To remove the degree of freedom for translation, we enforce the coordinates to be centered at the origin:

$$\sum_i \mathbf{y}_i = \mathbf{0} \tag{106}$$

- To avoid degenerate solutions, we constrain the embedding vectors to have unit covariance, with their outer-products satisfying

$$\frac{1}{n} \sum_i \mathbf{y}_i \mathbf{y}_i^\top = \mathbf{I} \tag{107}$$

- The optimal embedding is found by the bottom $d+1$ non-zero eigenvectors, i.e., those $d+1$ eigenvectors corresponding to the smallest but non-zero $d+1$ eigenvalues. The bottom eigenvector is the unit vector the corresponds to the free translation, it is discarded.

Putting It Together

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

LLE ALGORITHM

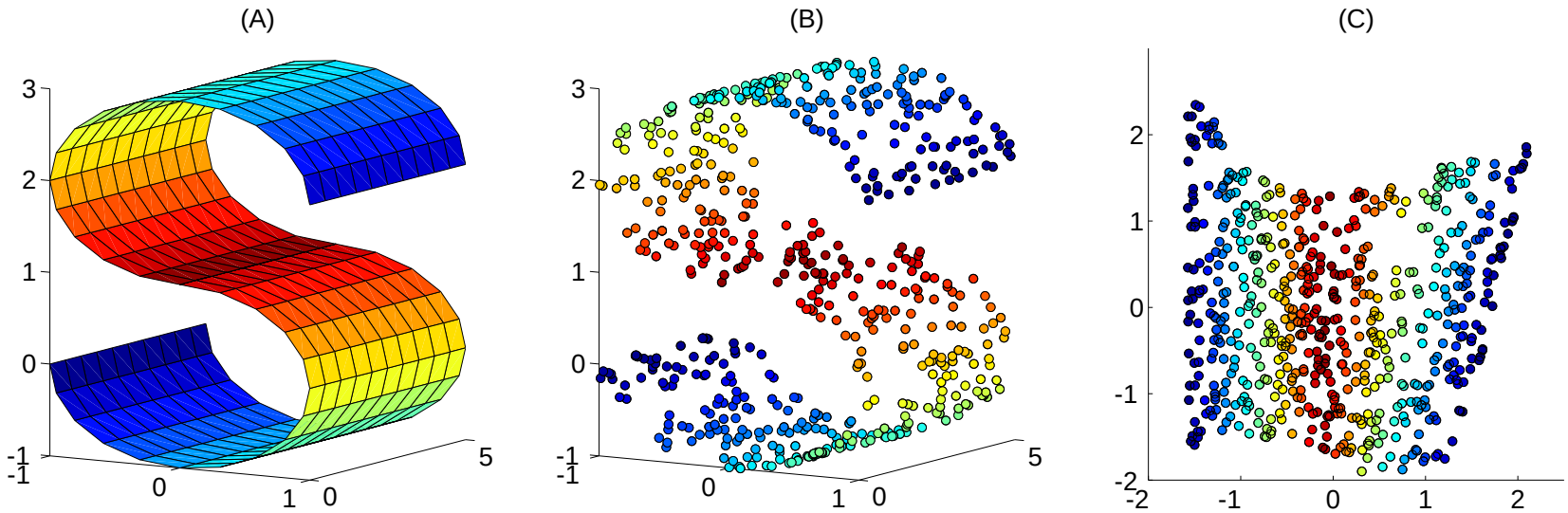
1. Compute the neighbors of each data point, $\vec{X}_i$.
2. Compute the weights W_{ij} that best reconstruct each data point $\vec{X}_i$ from its neighbors, minimizing the cost in eq. (1) by constrained linear fits.
3. Compute the vectors $\vec{Y}_i$ best reconstructed by the weights W_{ij}, minimizing the quadratic form in eq. (2) by its bottom nonzero eigenvectors.

LLE illustrates a general principle of manifold learning, elucidated by Tenenbaum et al[11], that overlapping local neighborhoods—collectively analyzed—can provide information about global geometry. (From Saul and Roweis 2001)

Example 1

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

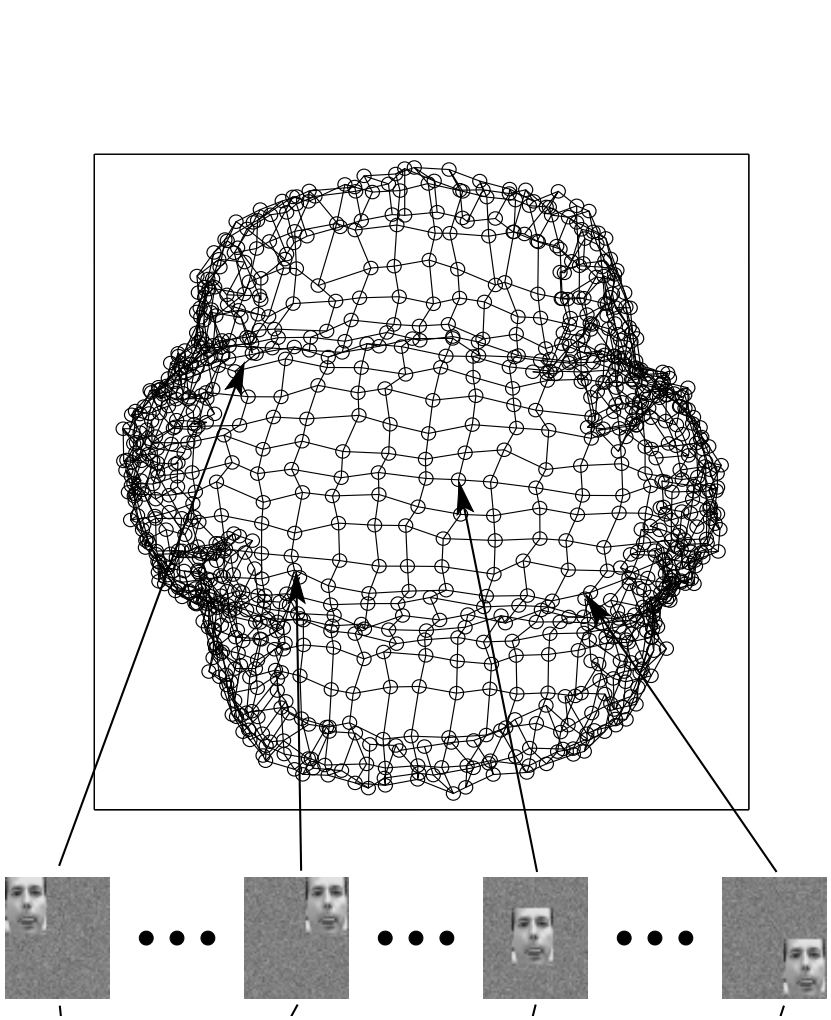
K=12.

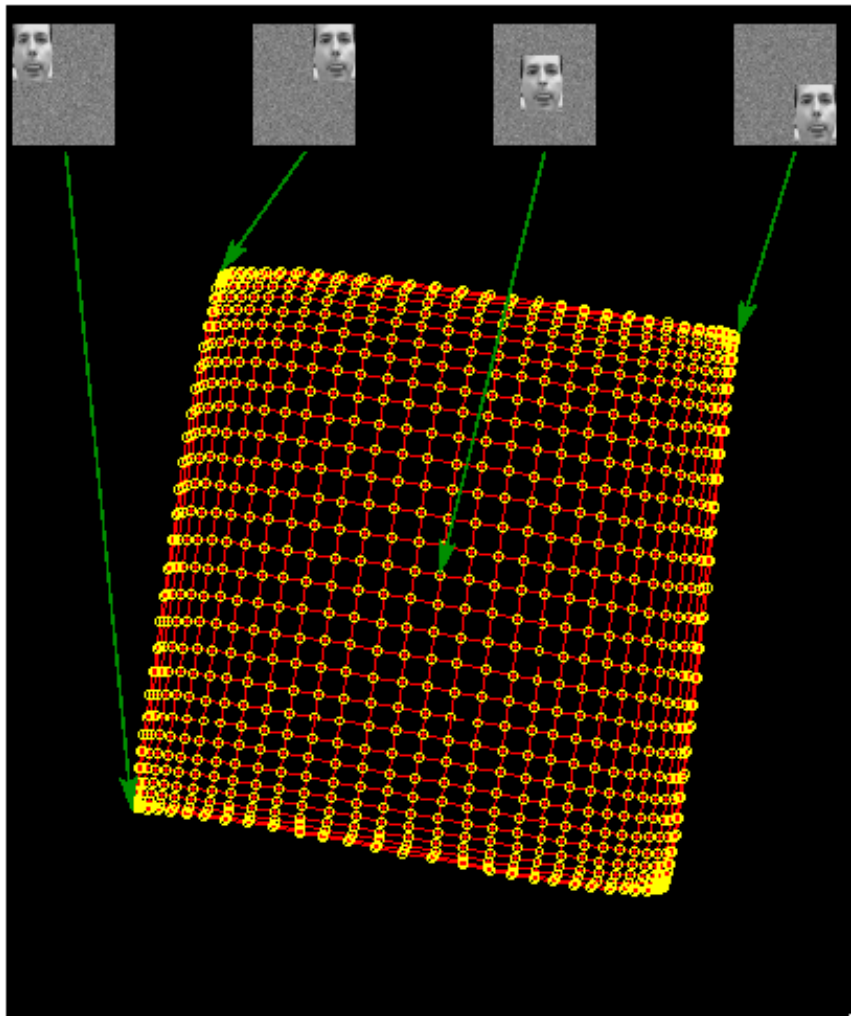


Example 2

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

K=4.

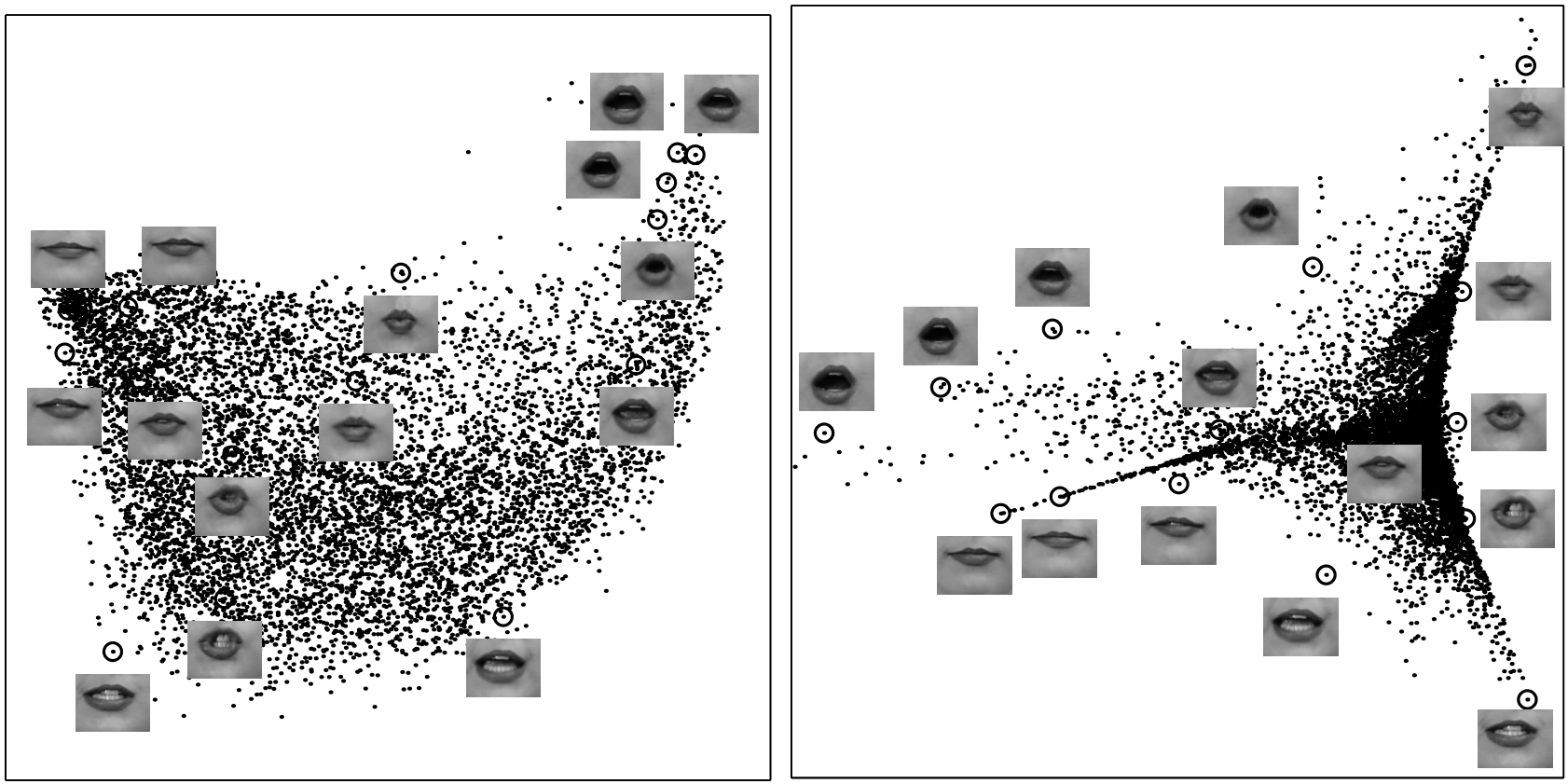




Example 3

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

K=16.



If the lip images described a nearly linear manifold, these two methods would yield similar results; thus the significant differences in these embeddings reveal the presence of nonlinear structure.

LLE Complexity

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- Computing nearest neighbors scales $O(Dn^2)$ in the worst case. But, in many situations, space partitioning methods can be used to find the K nearest neighbors in $O(n \log n)$ time.

LLE Complexity

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- Computing nearest neighbors scales $O(Dn^2)$ in the worst case. But, in many situations, space partitioning methods can be used to find the K nearest neighbors in $O(n \log n)$ time.
- Computing the weights is $O(DnK^3)$.

LLE Complexity

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- Computing nearest neighbors scales $O(Dn^2)$ in the worst case. But, in many situations, space partitioning methods can be used to find the K nearest neighbors in $O(n \log n)$ time.
- Computing the weights is $O(DnK^3)$.
- Computing the projection is $O(dn^2)$.

LLE Complexity

Source: Saul and Roweis, An Introduction to Locally Linear Embedding, 2001

- Computing nearest neighbors scales $O(Dn^2)$ in the worst case. But, in many situations, space partitioning methods can be used to find the K nearest neighbors in $O(n \log n)$ time.
- Computing the weights is $O(DnK^3)$.
- Computing the projection is $O(dn^2)$.
- All matrix computations are on very sparse matrices and can thus be implemented quite efficiently.