CS 340 Lec. 6: Linear Dimensionality Reduction

AD

January 2011

- Introduction & Motivation
- Brief Review of Linear Algebra
- Principal Component Analysis
- Applications
- Singular Value Decomposition

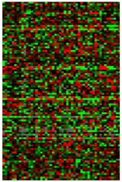
Lots of High Dimensional Data

face images

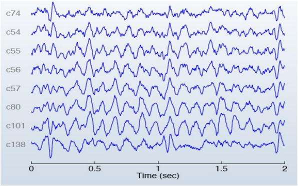
Zambian President Levy Mwanawasa has won a second term in office in an election his challenger Michael Sata accused him of rigging, official results showed on Monday.

According to media reports, a pair of hackers said on Saturday that the Firefox Web browser, commonly perceived as the safer and more customizable alternative to market leader Internet Explorer, is critically flawed. A presentation on the flaw was shown during the ToorCon hacker conference in San Diego.

documents



gene expression data



MEG readings

Motivation

Why do dimensionality reduction?

- Computational: compress data $\Rightarrow$ time/space efficiency
- Statistical: fewer dimension $\Rightarrow$ better generalization
- Visualization: understand structure of data
- Anomaly detection: describe normal data, detect outliers

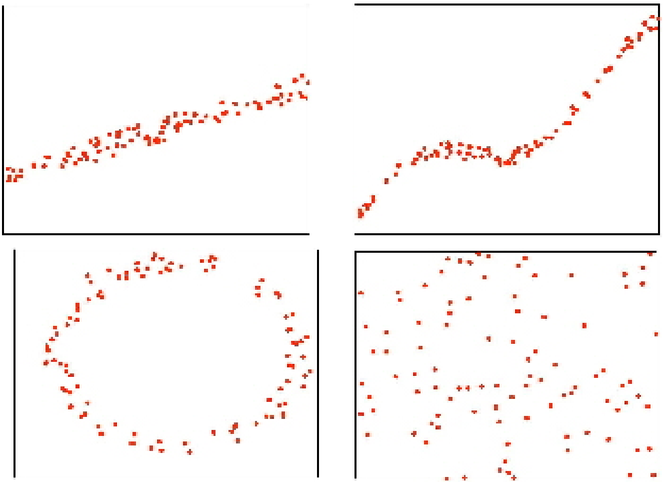
Dimensionality reduction in this course:

- Linear methods (this week)
- Clustering.
- Feature selection.

Types of Problems

- **Supervised learning** (classification, regression):
  Applications: face recognition, gene expression prediction
  Techniques: kNN, SVM, least squares ($+$ dimensionality reduction preprocessing)
- **Structure discovery**: find an alternative representation $\mathbf{z}$ of data $\mathbf{x}$
  Applications: visualization
  Techniques: clustering, linear dimensionality reduction
- **Density estimation** $p(\mathbf{x})$: model the data,
  Applications: anomaly detection, language modeling
  Techniques: clustering, linear dimensionality reduction

What is the true dimensionality of these data?



What is the true dimensionality of this data?

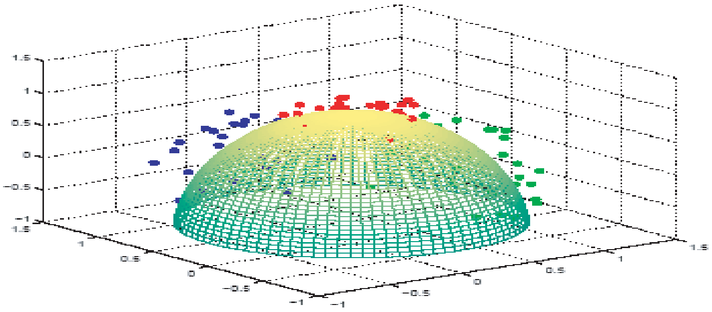
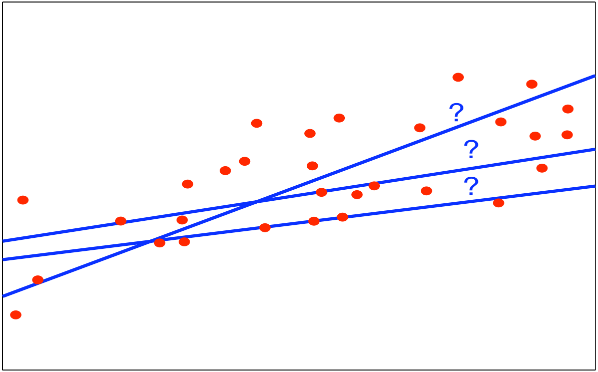


Figure: Simulated data in three classes, near the surface of a half-sphere

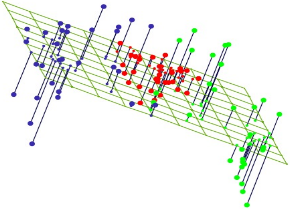
Linear Dimensionality Reduction



Which line should I pick?

Linear Dimensionality Reduction

Represent each face as a high-dimensional vector $\mathbf{x} \in \mathbb{R}^{361}$ by a lower-dimensional vector say $\mathbf{z} \in \mathbb{R}^{10}$.



Review of Linear Algebra

- $x = 3.4$, $\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_d \end{pmatrix}$, $A = \begin{pmatrix} a_{11} & \cdots & a_{1p} \\ \vdots & & \\ a_{d1} & \cdots & a_{dp} \end{pmatrix}$,
  $B = \begin{pmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & & \\ b_{p1} & \cdots & b_{pn} \end{pmatrix}$.
- Here x is scalar (1×1), $\mathbf{x}$ is $d \times 1$, A is $d \times p$ and B is $p \times n$.
- Transposition: $x^{\mathsf{T}} = x$, $\mathbf{x}^{\mathsf{T}} = \begin{pmatrix} x_1 & \cdots & x_d \end{pmatrix}$, $\left(A^{\mathsf{T}}\right)_{i,j} = a_{ji}$.
- Quantities whose inner dimensions match may be "multiplied" by summing over this index. The outer dimensions give the dimensions of the answer.

$$(A\mathbf{x})_i = \sum_{j=1} a_{i,j} x_j, \quad (AB)_{i,j} = \sum_{k=1}^{p} a_{i,k} b_{k,j}$$

- $\mathbf{x}^{\mathsf{T}}\mathbf{x}$ scalar, $\mathbf{x}\mathbf{x}^{\mathsf{T}}$ $d \times d$, $A\mathbf{x}$ $d \times 1$, AB $d \times n$, $\mathbf{x}^{\mathsf{T}} A \mathbf{x}$ scalar.

Review of Linear Algebra

- Simple and valid manipulations

$$
\begin{aligned}
(AB)\,C &= A\,(BC)\,,\ A\,(B+C) = AB + AC, \\
(A+B)^{\mathsf{T}} &= A^{\mathsf{T}} + B^{\mathsf{T}},\ (AB)^{\mathsf{T}} = B^{\mathsf{T}} A^{\mathsf{T}}
\end{aligned}
$$

- Consider a square matrix A then $\mathbf{u}$ is an eigenvector of A and λ is its associated eigenvalue iff

$$A\mathbf{u} = \lambda \mathbf{u}.$$

- If the matrix is diagonalizable

$$A\mathbf{U} = \mathbf{U}D \Leftrightarrow A = \mathbf{U}D\mathbf{U}^{-1}$$

- Q: Prove that $A^k = \mathbf{U}D^k\mathbf{U}^{-1}$. Why is this expression useful?

Review of Linear Algebra: PD Matrices

- A real-valued square matrix A is called (semi-)positive definite if

$$\mathbf{x}^{\mathsf{T}} A \mathbf{x} \geq \mathbf{0}$$

- Q: Prove that for any matrix $\mathbf{M}$, the matrix $\mathbf{M}^{\mathsf{T}}\mathbf{M}$ is (semi-)positive definite.
- Q: Prove that a positive definite matrix only admits positive eigenvalues.

Review of Linear Algebra: Inner Product

- Let $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$, then the inner product of $\mathbf{u}$ and $\mathbf{v}$ is a scalar

$$\mathbf{u}^\mathsf{T}\mathbf{v} = \mathbf{v}^\mathsf{T}\mathbf{u} = \sum_{i=1}^{d} u_i v_i$$

- The (Euclidean) length/norm of a vector $\mathbf{u}$ is written $\|\mathbf{u}\|$ and is defined as the square root of the inner product of the vector with itself

$$\|\mathbf{u}\| = \sqrt{\mathbf{u}^\mathsf{T}\mathbf{u}} = \sqrt{\sum_{i=1}^{d} u_i^2}.$$

- If the angle between vectors $\mathbf{u}$ and $\mathbf{v}$ is θ then

$$\cos(\theta) = \frac{\mathbf{u}^\mathsf{T}\mathbf{v}}{\|\mathbf{u}\| \, \|\mathbf{v}\|}.$$

Approximating High-dimensional Vectors

- We are given N data $\{\mathbf{x}_i\}_{i=1}^N$ where $\mathbf{x}_i \in \mathbb{R}^d$ and we want to approximate them by $\{\widehat{\mathbf{x}}_i\}_{i=1}^N$ using

$$\widehat{\mathbf{x}}_i = \sum_{j=1}^{k} z_{j,i} \mathbf{w}_j$$

where $z_{i,j} \in \mathbb{R}$ and $\{\mathbf{w}_i\}_{i=1}^k$ are $\mathbb{R}^d$-valued **basis** vector.

- This can be rewritten as

$$\widehat{\mathbf{x}}_i = \mathbf{W}\mathbf{z}_i$$

for

$$\begin{aligned} \mathbf{W} &= \begin{pmatrix} \mathbf{w}_1 & \cdots & \mathbf{w}_k \end{pmatrix}, \; d \times k \text{ matrix} \\ \mathbf{z}_i &= \begin{pmatrix} z_{1,i} & \cdots & z_{k,i} \end{pmatrix}^{\mathrm{T}}, \; k \times 1 \text{ vector} \end{aligned}$$

Approximating High-dimensional Vectors

- An even more compact notation is

$$\underbrace{\widehat{\mathbf{X}}}_{d \times N} = \underbrace{\mathbf{W}}_{d \times k} \underbrace{\mathbf{Z}}_{k \times N}$$

  where $\widehat{\mathbf{X}} = \begin{pmatrix} \widehat{\mathbf{x}}_1 & \cdots & \widehat{\mathbf{x}}_N \end{pmatrix}$ and $\mathbf{Z} = \begin{pmatrix} \mathbf{z}_1 & \cdots & \mathbf{z}_N \end{pmatrix}$.
- We can gain very significantly in terms of storage if $k << d$ as we only need to store $\mathbf{W}$ (size $d \times k$) and $\mathbf{Z}$ (size $k \times N$) to compute $\widehat{\mathbf{X}}$ instead of $\mathbf{X}$ (size $d \times N$).
- Example: For $d = 1000$, $k = 10$ and $N = 10^6$, we have $d \times N / \left(d \times k + k \times N\right) \approx 100$.

Approximating High-dimensional Vectors

- How should we select $\mathbf{W}$ and $\mathbf{Z}$ to ensure $\widehat{\mathbf{X}} \approx \mathbf{X}$?
- We introduce the reconstruction error $\mathbf{X} - \widehat{\mathbf{X}}$ and propose to minimize the square of its Frobenius norm

$$
\begin{aligned}
J(\mathbf{W}, \mathbf{Z}) &= \frac{1}{N} \left\| \mathbf{X} - \widehat{\mathbf{X}} \right\|_F^2 = \frac{1}{N} \sum_{i=1}^{N} \left\| \mathbf{x}_i - \widehat{\mathbf{x}}_i \right\|^2 \\
&= \frac{1}{N} \sum_{j=1}^{d} \sum_{i=1}^{N} \left(x_{j,i} - \widehat{x}_{j,i} \right)^2
\end{aligned}
$$

subject to $\mathbf{W}$ be an *orthonormal matrix*; i.e.

$$
\mathbf{w}_i^{\mathsf{T}} \mathbf{w}_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \Leftrightarrow \mathbf{W}^{\mathsf{T}} \mathbf{W} = \mathbf{I}_k .
$$

- Q: What is the minimum total squared reconstruction error for $k = d$? What about $k > d$?

Preliminaries: Normalization and Centering of the Data

- It is standard to normalize and center the data beforehand.
- This ensures that PCA finds the "interesting" directions of variation, not the ones which just happen to be large because of the units of measurement that are used.
- Hence in practice if the "original data" were $\{\mathbf{x}_i\}$, we compute

$$m_j = \frac{1}{N}\sum_{i=1}^{N} x_{j,i},\ \sigma_j^2 = \frac{1}{N}\sum_{i=1}^{N}\left(x_{j,i} - m_j\right)^2$$

and we set $\overline{\mathbf{x}}_i = \begin{pmatrix} \overline{x}_{1,i} & \cdots & \overline{x}_{d,i} \end{pmatrix}^{\mathsf{T}}$ where

$$\overline{x}_{j,i} = \frac{x_{j,i} - m_j}{\sigma_j}.$$

Finding the first principal component

- Consider first the case $k = 1$ then we want to minimize

$$
\begin{aligned}
J(\mathbf{W}, \mathbf{Z}) &= J\left(\mathbf{w}_1, \mathbf{z}^1\right) = \frac{1}{N}\sum_{i=1}^{N} \left\|\mathbf{x}_i - z_{1,i}\mathbf{w}_1\right\|^2 \\
&= \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i^{\mathsf{T}}\mathbf{x}_i - 2\mathbf{x}_i^{\mathsf{T}} z_{1,i}\mathbf{w}_1 + z_{1,i}\mathbf{w}_1^{\mathsf{T}}\mathbf{w}_1 z_{1,i} \\
&= \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i^{\mathsf{T}}\mathbf{x}_i - 2z_{1,i}\mathbf{x}_i^{\mathsf{T}}\mathbf{w}_1 + z_{1,i}^2
\end{aligned}
$$

  subject to $\mathbf{w}_1^{\mathsf{T}}\mathbf{w}_1 = 1$ with $\mathbf{z}^1 = \begin{pmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,N} \end{pmatrix}$.
- Taking derivative w.r.t. $z_{1,i}$ and setting it equal to zero

$$
\frac{\partial J\left(\mathbf{w}_1, \mathbf{z}^1\right)}{\partial z_{1,i}} = -2\mathbf{x}_i^{\mathsf{T}}\mathbf{w}_1 + 2z_{1,i} = 0 \Leftrightarrow z_{1,i} = \mathbf{x}_i^{\mathsf{T}}\mathbf{w}_1
$$

- Optimal reconstruction weights are obtained by orthogonally projecting the data onto the first principal direction $\mathbf{w}_1$.

Finding the first principal component

- Minimizing $J(\mathbf{w}_1)$ is thus equivalent to maximizing

$$\begin{aligned}\frac{1}{N}\sum_{i=1}^{N} z_{1,i}^2 &= \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{w}_1^{\mathsf{T}}\mathbf{x}_i\right)^2 \\ &= \mathbf{w}_1^{\mathsf{T}}\underbrace{\left(\frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i\mathbf{x}_i^{\mathsf{T}}\right)}_{\widehat{\Sigma}}\mathbf{w}_1 \text{ s.t. } \|\mathbf{w}_1\| = 1.\end{aligned}$$

- Assume the data have been centered, so that

$$\sum_{i=1}^{N}\mathbf{x}_i = 0$$

then

$$\frac{1}{N}\sum_{i=1}^{n}\left(\mathbf{w}_1^{\mathsf{T}}\mathbf{x}_i\right)^2 \approx \mathbb{E}\left(\left(\mathbf{w}_1^{\mathsf{T}}\mathbf{x}\right)^2\right) \approx \mathbb{V}ar\left(\mathbf{w}_1^{\mathsf{T}}\mathbf{x}\right);$$

i.e. we seek $\mathbf{w}_1$ maximizing the variance of the projected data.

- Note additionally that we have

$$\widehat{\Sigma} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i\mathbf{x}_i^{\mathsf{T}} \approx \mathbb{E}\left(\mathbf{x}\mathbf{x}^{\mathsf{T}}\right) = Cov\left(\mathbf{x}\right).$$

- That is $\widehat{\Sigma}$ is an estimate of the covariance/correlation matrix of the data.

Finding the first principal component

- Minimizing $J(\mathbf{w}_1)$ is equivalent to maximizing

$$\mathbf{w}_1^{\mathrm{T}} \widehat{\Sigma} \mathbf{w}_1 \text{ s.t. } \|\mathbf{w}_1\| = 1.$$

- **Proposition**: The vector $\mathbf{w}_1^{\mathrm{opt}}$ minimizing $J(\mathbf{w}_1)$ is the eigenvector (selected such that $\|\mathbf{w}_1\| = 1$) associated to the largest eigenvalue of $\widehat{\Sigma}$.
- **Proof**: $\widehat{\Sigma}$ is a symmetric matrix so it is diagonalizable by an orthornormal matrix U; i.e.

$$\widehat{\Sigma}\mathbf{U} = \mathbf{U}\mathbf{D} \Leftrightarrow \widehat{\Sigma} = \mathbf{U}\mathbf{D}\mathbf{U}^{\mathrm{T}}$$

  with $\mathbf{D}$ diagonal. Without loss of generality, we pick $\mathbf{D} = \mathrm{diag}\left(\sigma_1^2, ..., \sigma_d^2\right)$ where $\sigma_1^2 \geq \sigma_2^2 \geq \cdots \geq \sigma_d^2$.

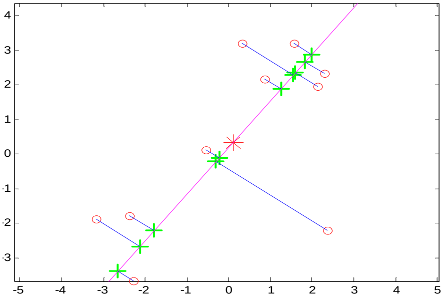
Finding the first principal component

- It follows that

$$\begin{aligned}
\underset{\mathbf{w}_1:\|\mathbf{w}_1\|=1}{\arg\max}\ \mathbf{w}_1^{\mathrm{T}}\widehat{\Sigma}\mathbf{w}_1 &= \underset{\mathbf{w}_1:\|\mathbf{w}_1\|=1}{\arg\max}\ \left(\mathbf{U}^{\mathrm{T}}\mathbf{w}_1\right)^{\mathrm{T}}\mathbf{D}\left(\mathbf{U}^{\mathrm{T}}\mathbf{w}_1\right) \\
&= \underset{\mathbf{y}=\mathbf{U}^{\mathrm{T}}\mathbf{w}_1:\|\mathbf{y}\|=1}{\arg\max}\ \mathbf{y}^{\mathrm{T}}D\mathbf{y} \\
&= \underset{\mathbf{y}=\mathbf{U}^{\mathrm{T}}\mathbf{w}_1:\|\mathbf{y}\|=1}{\arg\max}\ \sum_{i=1}^{d}\sigma_i^2 y_i^2
\end{aligned}$$

so $\mathbf{y}^{\text{opt}} = \begin{pmatrix} 1 & 0 & \ldots & 0 \end{pmatrix}^{\mathrm{T}} \Rightarrow \mathbf{w}_1 = \mathbf{U}\mathbf{y}^{\text{opt}} = \mathbf{u}_1.$

PCA Example



We have $d = 2$ and $k = 1$. Circles are the original data points, crosses are the reconstructions. The red star is the data mean.

The second principal component

- We want to minimize

$$J\left(\mathbf{w}_1, \mathbf{z}^1, \mathbf{w}_2, \mathbf{z}^2\right) = \frac{1}{N}\sum_{i=1}^{N} \left\| \mathbf{x}_i - z_{1,i}\mathbf{w}_1 - z_{2,i}\mathbf{w}_2 \right\|^2$$

  s.t. $\|\mathbf{w}_1\| = \|\mathbf{w}_2\| = 1$ and $\mathbf{w}_1^{\mathrm{T}}\mathbf{w}_2 = 0$.
- Optimizing w.r.t $\mathbf{w}_1, \mathbf{z}^1$ gives the same results as before. If we optimize w.r.t $z_{2,i}$, we find

$$\frac{\partial J\left(\mathbf{w}_1^{\mathrm{opt}}, \mathbf{z}^{\mathrm{opt},1}, \mathbf{w}_2, \mathbf{z}^2\right)}{\partial z_{2,i}} = -2\mathbf{x}_i^{\mathrm{T}}\mathbf{w}_2 + 2z_{2,i} = 0 \Leftrightarrow z_{2,i} = \mathbf{x}_i^{\mathrm{T}}\mathbf{w}_2$$

- Similarly it can be proved that $\mathbf{w}_2^{\mathrm{opt}}$ is the eigenvector (selected such that $\|\mathbf{w}_2\| = 1$) associated to the second largest eigenvalue of $\widehat{\Sigma}$.

General Case

- Compute the eigendecomposition of

$$\widehat{\Sigma} = \frac{1}{N}\mathbf{X}\mathbf{X}^{\mathrm{T}} = \mathbf{U}\mathbf{D}\mathbf{U}^{\mathrm{T}}$$

with $\sigma_1^2 \geq \sigma_2^2 \geq \cdots \geq \sigma_d^2$ and keep only the associated k eigenvectors

$$\mathbf{U}_k = \begin{pmatrix} \mathbf{u}_1 & \cdots & \mathbf{u}_k \end{pmatrix}$$

- The estimate is given by

$$\widehat{\mathbf{X}} = \mathbf{U}_k \underbrace{\left(\mathbf{U}_k^{\mathrm{T}}\mathbf{X}\right)}_{\mathbf{Z}^{\mathrm{opt}}} = \sum_{j=1}^{k} \mathbf{u}_j \underbrace{\left(\mathbf{u}_j^{\mathrm{T}}\mathbf{X}\right)}_{\text{“loadings”}}$$

- It can be additionally shown that (for $k < d$)

$$\frac{1}{N}\left\|\mathbf{X} - \widehat{\mathbf{X}}\right\|_F^2 = \sum_{j=k+1}^{d} \sigma_j^2$$

Important Practical Remark

- If you have centered and normalize the data beforehand, don't forget to correct later on!!
- Suppose you have considered

$$\overline{\mathbf{X}}=\Phi^{-1}\left(\mathbf{X}-\mu\right)\Leftrightarrow\mathbf{X}=\mu+\Phi\overline{\mathbf{X}}$$

then the reconstruction will be

$$\widehat{\mathbf{X}}=\mu+\Phi\widehat{\overline{\mathbf{X}}}$$

where $\widehat{\overline{\mathbf{X}}}$ is the PCA approximation of $\overline{\mathbf{X}}$.

Reconstruction Error

- We have

$$\mathbf{x}_i - \widehat{\mathbf{x}}_i = \sum_{j=k+1}^{d} \mathbf{u}_j \left(\mathbf{u}_j^{\mathsf{T}} \mathbf{x}_i \right)$$

- Hence it follows that

$$\begin{aligned} \left\| \mathbf{x}_i - \widehat{\mathbf{x}}_i \right\|^2 &= \left(\sum_{j=k+1}^{d} \mathbf{u}_j \left(\mathbf{u}_j^{\mathsf{T}} \mathbf{x}_i \right) \right)^{\mathsf{T}} \left(\sum_{j=k+1}^{d} \mathbf{u}_j \left(\mathbf{u}_j^{\mathsf{T}} \mathbf{x}_i \right) \right) \\ &= \left(\sum_{j=k+1}^{d} \left(\mathbf{u}_j^{\mathsf{T}} \mathbf{x}_i \right) \mathbf{u}_j^{\mathsf{T}} \right) \left(\sum_{j=k+1}^{d} \mathbf{u}_j \left(\mathbf{u}_j^{\mathsf{T}} \mathbf{x}_i \right) \right) \\ &= \sum_{j=k+1}^{d} \left(\mathbf{u}_j^{\mathsf{T}} \mathbf{x}_i \right)^2 \text{ as } \mathbf{u}_j^{\mathsf{T}} \mathbf{u}_l = 1 \text{ if } j = l \text{ and } 0 \text{ if } j \neq l \\ &= \sum_{j=k+1}^{d} \mathbf{u}_j^{\mathsf{T}} \mathbf{x}_i \mathbf{x}_i^{\mathsf{T}} \mathbf{u}_j \end{aligned}$$

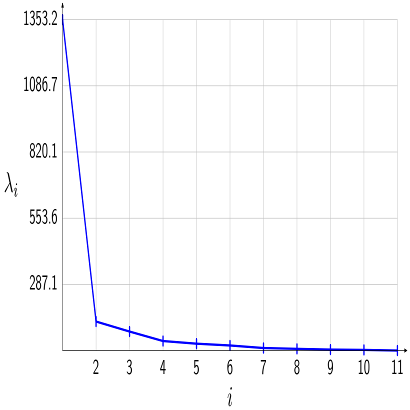
Reconstruction Error

- Thus we have

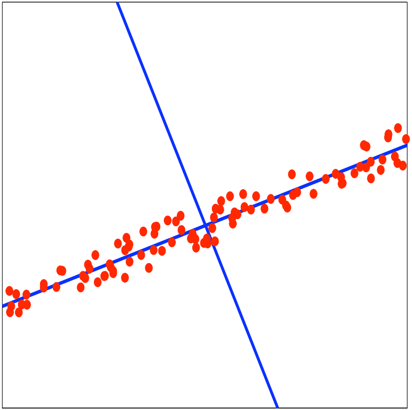
$$
\begin{aligned}
\frac{1}{N}\left\|\mathbf{X}-\widehat{\mathbf{X}}\right\|_F^2 &= \frac{1}{N}\sum_{i=1}^{N}\|\mathbf{x}_i-\widehat{\mathbf{x}}_i\|^2 \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{j=k+1}^{d}\mathbf{u}_j^{\mathsf{T}}\mathbf{x}_i\mathbf{x}_i^{\mathsf{T}}\mathbf{u}_j\right) \\
&= \frac{1}{N}\sum_{j=k+1}^{d}\mathbf{u}_j^{\mathsf{T}}\left(\sum_{i=1}^{N}\mathbf{x}_i\mathbf{x}_i^{\mathsf{T}}\right)\mathbf{u}_j \\
&= \sum_{j=k+1}^{d}\mathbf{u}_j^{\mathsf{T}}\widehat{\Sigma}\mathbf{u}_j \\
&= \sum_{j=k+1}^{d}\sigma_j^2
\end{aligned}
$$

How Many Principal Components?

- Magnitude of eigenvalues indicate fraction of variance captured.
- Typically eigenvalues drop off sharply so you don't need too many.

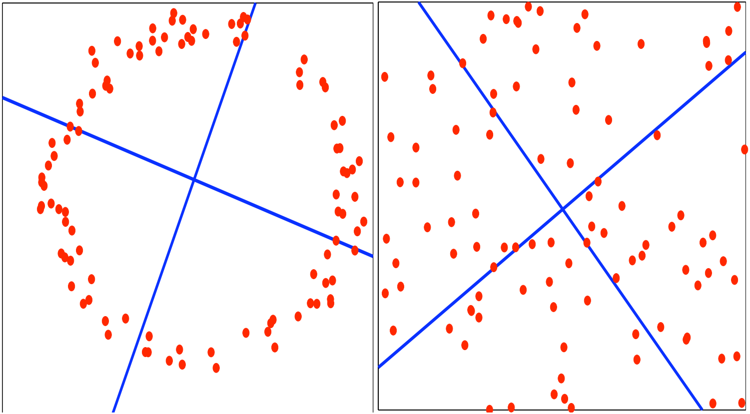


PCA Example



Example where PCA is of interest.

PCA Example



Examples where PCA is of no interest.

Image Compression using PCA

- Given one single image, how can you use the PCA to perform image compression?
- Many different approaches are possible
  - Example 1: Interpret the columns of the image as different data points $\mathbf{x}_i$.
  - Example 2: Interpret the rows of the image as different data points $\mathbf{x}_i$.
  - Example 3: Partition the image in non-overlapping small blocks, blocks are now $\mathbf{x}_i$.
- Note: There are better ways to compress images!

Computing the Principal Components

- Computing $\widehat{\Sigma}$ takes $O\left(Nd^2\right)$ operations and computing the eigenvectors of the $d \times d$ matrix $\widehat{\Sigma}$ takes $O\left(d^3\right)$ operations. This can be very prohibitive!
- If $d >> N$, then we can compute the eigenvectors based on the eigenvectors of the so-called $N \times N$ Gram matrix $\mathbf{X}^{\mathsf{T}}\mathbf{X}$ in $O\left(N^3\right)$ instead.
- Assume $\mathbf{v}_i$ is an eigenvector of $\mathbf{X}^{\mathsf{T}}\mathbf{X}$ such that $\|\mathbf{v}_i\| = 1$ associated to the eigenvalue λ_i then by definition

$$\mathbf{X}^{\mathsf{T}}\mathbf{X}\mathbf{v}_i = \lambda_i \mathbf{v}_i$$

so by multiplying both sides by $\mathbf{X}$ then

$$\underbrace{\mathbf{X}\mathbf{X}^{\mathsf{T}}}_{N\widehat{\Sigma}}\left(\mathbf{X}\mathbf{v}_i\right) = \lambda_i\left(\mathbf{X}\mathbf{v}_i\right)$$

- That is $\mathbf{X}\mathbf{v}_i = \widetilde{\mathbf{u}}_i$ is an eigenvector of $\widehat{\Sigma}$ associated to the eigenvalue $\frac{\lambda_i}{N}$ and we can have a unit norm eigenvector by selecting $\mathbf{u}_i = \lambda_i^{-1/2}\widetilde{\mathbf{u}}_i$.

Singular Value Decomposition

- Given a $d \times N$ matrix $\mathbf{X}$, the SVD of $\mathbf{X}$ is a factorization of the form

$$\underbrace{\mathbf{X}}_{d \times N} = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathsf{T}} = \sum_{i=1}^{r} \lambda_i \underbrace{\mathbf{u}_i}_{d \times 1} \underbrace{\mathbf{v}_i^{\mathsf{T}}}_{1 \times N}$$

  where $r = \min(d, N)$, $\mathbf{U}$ are the left singular vectors with $\mathbf{U}^{\mathsf{T}}\mathbf{U} = \mathbf{I}_r$, $\mathbf{V}$ are the right singular vectors with $\mathbf{V}^{\mathsf{T}}\mathbf{V} = \mathbf{I}_r$.
- Right singular vectors are eigenvectors of $\mathbf{X}^{\mathsf{T}}\mathbf{X}$

$$\begin{aligned} \mathbf{X}^{\mathsf{T}}\mathbf{X} &= \left(\mathbf{U}\mathbf{D}\mathbf{V}^{\mathsf{T}}\right)^{\mathsf{T}} \left(\mathbf{U}\mathbf{D}\mathbf{V}^{\mathsf{T}}\right) = \mathbf{V}\mathbf{D}\mathbf{U}^{\mathsf{T}}\mathbf{U}\mathbf{D}\mathbf{V}^{\mathsf{T}} \\ &= \mathbf{V}\mathbf{D}^2\mathbf{V}^{\mathsf{T}} \Rightarrow \mathbf{X}^{\mathsf{T}}\mathbf{X}\mathbf{V} = \mathbf{V}\mathbf{D}^2 \end{aligned}$$

- Left singular vectors are eigenvectors of $\mathbf{X}\mathbf{X}^{\mathsf{T}}$

$$\begin{aligned} \mathbf{X}\mathbf{X}^{\mathsf{T}} &= \left(\mathbf{U}\mathbf{D}\mathbf{V}^{\mathsf{T}}\right) \left(\mathbf{U}\mathbf{D}\mathbf{V}^{\mathsf{T}}\right)^{\mathsf{T}} = \mathbf{U}\mathbf{D}\mathbf{V}^{\mathsf{T}}\mathbf{V}\mathbf{D}\mathbf{U}^{\mathsf{T}} \\ &= \mathbf{U}\mathbf{D}^2\mathbf{U}^{\mathsf{T}} \Rightarrow \mathbf{X}^{\mathsf{T}}\mathbf{X}\mathbf{U} = \mathbf{U}\mathbf{D}^2 \end{aligned}$$

  and clearly $\sigma_i^2 = \lambda_i^2 / N$.

Singular Value Decomposition and PCA

- In the PCA, we have

$$\widehat{\mathbf{X}} = \mathbf{U}_k \left(\mathbf{U}_k^{\mathsf{T}} \mathbf{X} \right).$$

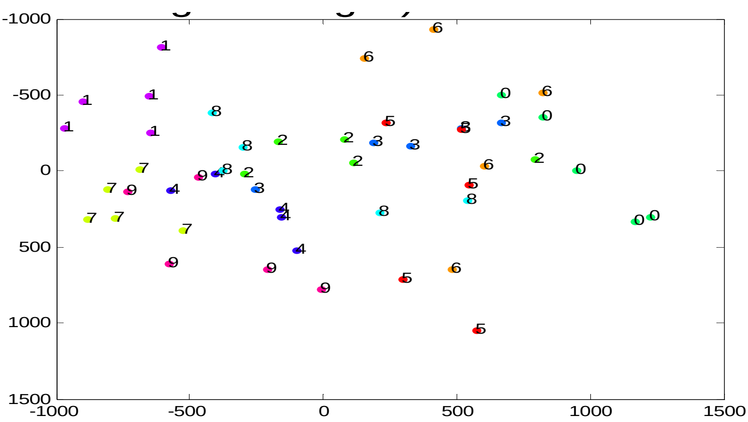
- If we plug the SVD decomposition

$$\begin{aligned} \widehat{\mathbf{X}} &= \mathbf{U}_k \left(\mathbf{U}_k^{\mathsf{T}} \mathbf{U} \right) \mathbf{D} \mathbf{V}^{\mathsf{T}} \\ &= \sum_{i=1}^{k} \lambda_i \mathbf{u}_i \mathbf{v}_i^{\mathsf{T}} \end{aligned}$$

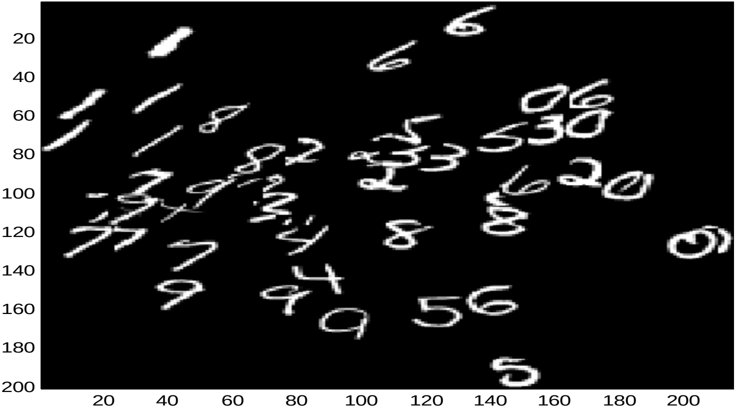
i.e. the truncated SVD yields the PCA approximation.

- This can be computationally beneficial.

Visualization



Visualization



Eigen-Faces

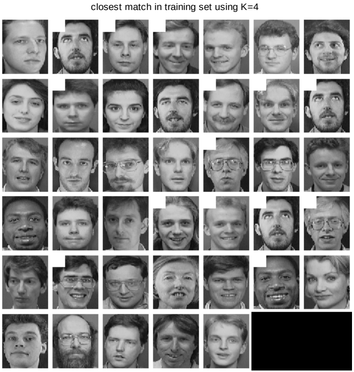
- d is the number of pixels.
- Each $\mathbf{x}_i \in \mathbb{R}^d$ is a face image.

$$X_{d \times n} \cong U_{d \times k} \; Z_{k \times n}$$

$$\left(\mathbf{z}_1 \ldots \mathbf{z}_n \right)$$

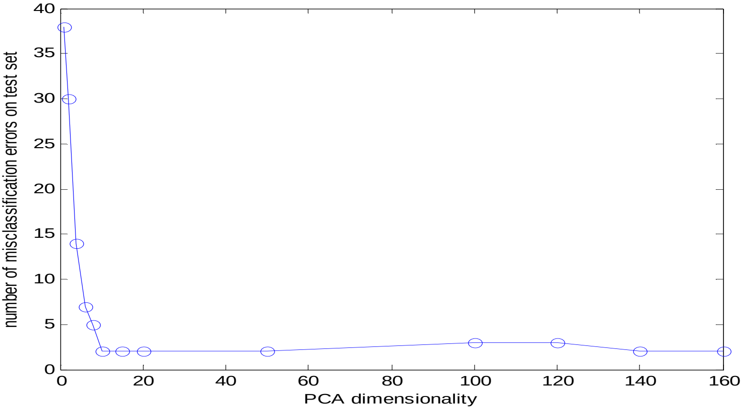
- Idea: $\mathbf{z}_i$ more "meaningful" representation of i-th face than $\mathbf{x}_i$.
- Can use $\mathbf{z}_i$ for nearest-neighbor classication
- Much faster: $O(dk + Nk)$ time instead of $O(dN)$ when $N, d >> k$.

Eigen-Faces with K-NN



Eigen-Faces with K-NN

Eigen-Faces with K-NN



Latent Semantic Analysis

- PCA can be used to cluster documents and carry out information retrieval by using concepts as opposed to exact word-matching.
- This enables us to surmount the problems of synonymy (car, auto) and polysemy (money bank, river bank).
- The data is available in a term-frequency (TF) matrix
  - N is the number of documents.
  - d is the number of words in the vocabulary.
- Each $\mathbf{x}_i \in \mathbb{R}^d$ is a vector of word counts; $x_{j,i}$ = numbers of occurences of word j in document i.

Example

- Document 1: {I, eat, chips}
- Document 2: {computer,chips,chips}
- Document 3: {intel,computer,chips}
- We have

$$\mathbf{X} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

PCA for Latent Semantic Analysis

- Using the PCA, we obtain

$$\mathbf{X} \approx \mathbf{U}_k \; \mathbf{Z}_k$$

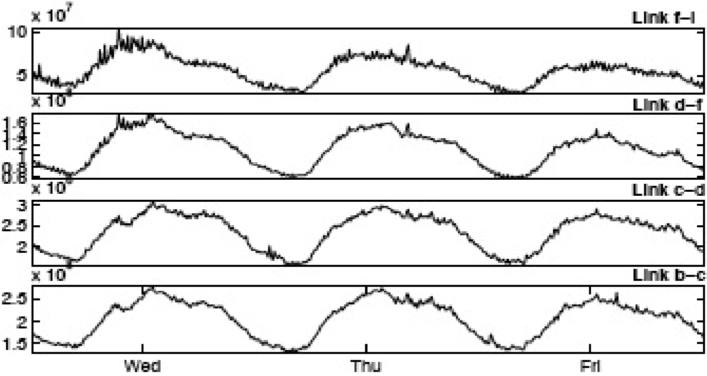
- That is we approximate the documents by a linear combination of k "basis" documents.
- How to measure similarity between two documents $\widehat{\mathbf{x}}_i$ and $\mathbf{x}_j$?

$$\widehat{\mathbf{x}}_i^{\mathrm{T}} \widehat{\mathbf{x}}_j \text{ is probably better than } \mathbf{x}_i^{\mathrm{T}} \mathbf{x}_j$$

- Applications: information retrieval.
- Note: usually no computational savings; original $\mathbf{x}$ is already sparse.

Network Anomaly Detection

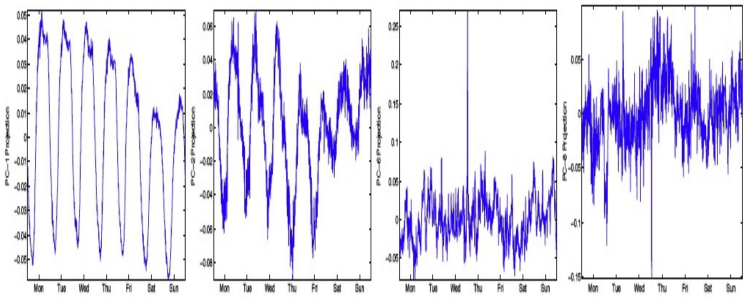
- x_{ji} = amount of traffic on link j in the network during each time interval i



- Model assumption: total traffic is sum of flows along a few "paths".
- Apply PCA: each principal component intuitively represents a "path".

Network Anomaly Detection

- Anomaly when traffic deviates from first few principal components.



u_1 u_2 u_6 u_8

(a) Normal Behavior (b) Anomalous Behavior