

6.867 Machine Learning Fall 2017

## Lecture 20. Principle Component Analysis

<http://stellar.mit.edu/S/course/6/fa17/6.867/>

# Announcements

- Quiz 2
  - Thursday, November 30 7pm-9pm
  - Make up:
    - Tuesday, November 28, 4pm-6pm
  - Syllabus:
    - Till and including Lecture on November 16, 2017
  - Quiz Review:
    - Monday, November 27/Wednesday, November 29 TA OH
    - It will be posted online with solutions before Thanks Giving Break
    - It will cover concepts relevant for exam questions

# Where Are We?

Date	Topic
9/7	Introduction, Overview, Basics
9/12	Regression: Risk Min., Max likelihood
9/14	Linear Regression: Shrinkage, Bias-Variance
9/19	Linear Regression: Bayesian
9/21	Classification: Discrim analysis; perceptron
9/26	Classification: Logistic reg
9/28	Classification: Support vector machines
10/3	Kernels
10/5	K-Nearest neighbors, VC-dimension
10/10	<i>Holiday</i>
10/12	Neural nets: basics, back-propagation
10/17	Neural nets: regularization, CNNs, auto-encoders
10/19	<b>Exam 1 7:30 - 9:30 PM</b>
10/24	Neural nets: recursive neural networks, LSTM
10/26	Neural nets: RNN sequence models, attention
10/31	Tree-models, Bagging, Random Forest
11/2	Bayesian networks: Learning and Inference
11/7	Mixture models: EM, K-means
11/9	Topic models
11/14	Deep generative models
11/16	Sampling method: Why, MCMC, Gibbs
11/21	Matrix data: principal component analysis
11/23	<i>Holiday</i>
11/28	Matrix data: recommendation systems
11/30	<b>Exam 2 7:30 - 9:30</b>
12/5	Matrix data: non-neg factor, PMI/Word2Vec
12/7	Reinforcement Learning – Part I
12/12	Reinforcement Learning – Part II

Regression

Classification

Non-linear methods

Distribution Repr.

Mixtures, EM

Latent Variable Models

Sampling

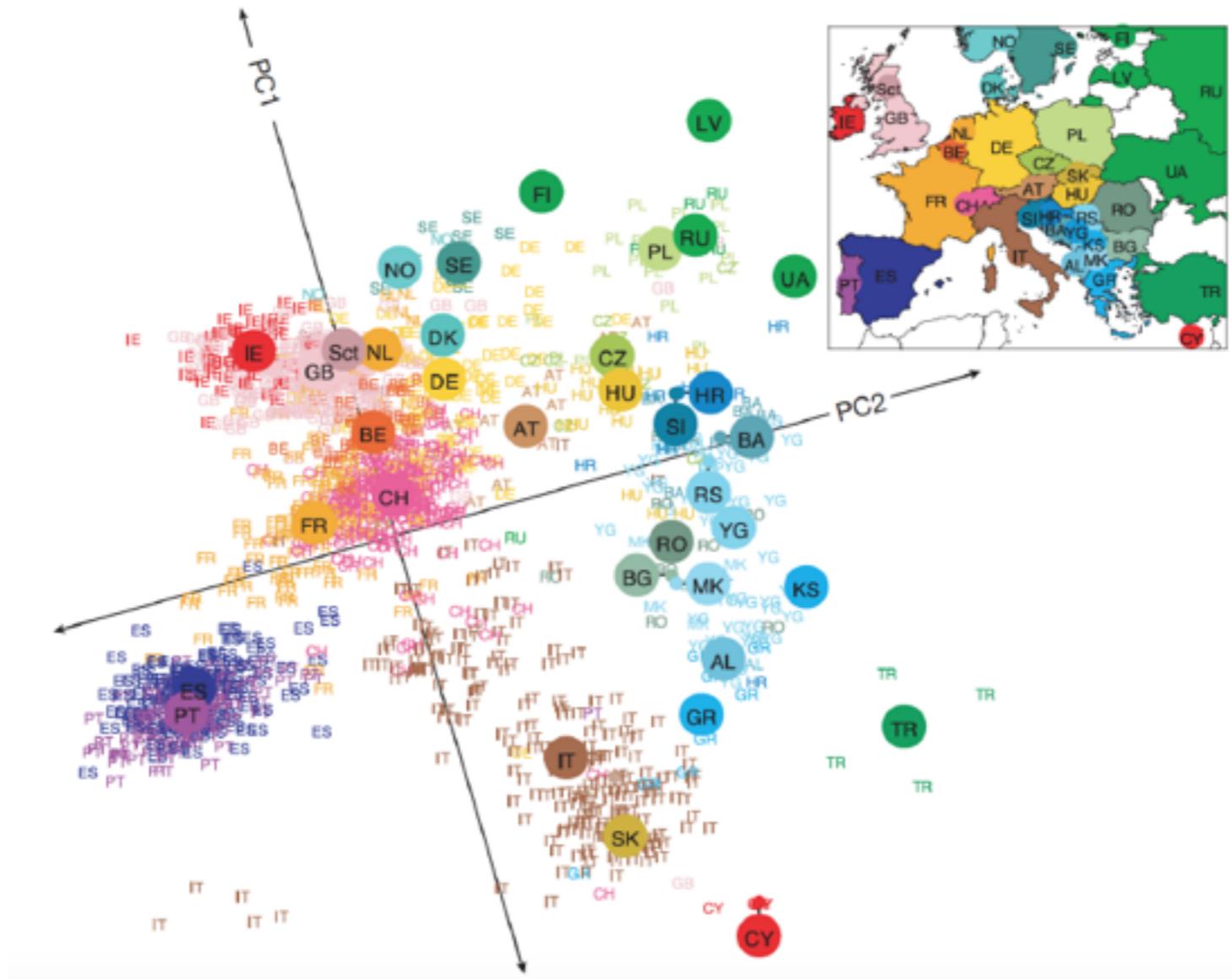
Matrix data

Re-inforcement learning

Supervised

Unsupervised

# Example of Principle Component Analysis (PCA)



Genes Mirror Geography Within Europe, Novembre et al (2008)

# Outline

- PCA (a la (*Kosambi-*)Karhunen-Loeve transformation)
  - Formulation
  - Applications
  - Implementation
- Probabilistic PCA
  - Maximum Likelihood
  - EM algorithm
- Non-linear PCA
  - Kernel PCA

# What is PCA

- Answer 1: Maximizing variance
  - Project data into lower dimensional subspace
  - So that variance of projected data is maximized
- Answer 2: Minimizing error (or movement)
  - Project data into lower dimensional subspace
  - So that the square distance “movement” of data is minimized
- Both lead to the same algorithm

# PCA via Maximizing Variance

- Setup
  - Data:  $x_1, \dots, x_N$  with  $x_n \in \mathbb{R}^d$
  - Goal: project each  $x_n$  into  $\mathbb{R}^m$ ,  $m \ll d$
- Example: linear projection with  $m = 1$ 
  - Let  $\mathbf{u}_1 \in \mathbb{R}^d$  such that  $\mathbf{u}_1^T \mathbf{u}_1 = 1$
  - Projection:  $x_n \rightarrow \mathbf{u}_1^T x_n$ ,  $n \in [N]$
  - Mean of projected data:  $\mathbf{u}_1^T \bar{x}$  where  $\bar{x} = \frac{1}{N} \left( \sum_n x_n \right)$
  - Variance of projected data:

$$\frac{1}{N} \sum_n \left( \mathbf{u}_1^T x_n - \mathbf{u}_1^T \bar{x} \right)^2$$

# PCA via Maximizing Variance

- Re-writing variance

$$\begin{aligned} & \frac{1}{N} \sum_n \left( \mathbf{u}_1^T x_n - \mathbf{u}_1^T \bar{x} \right)^2 \\ &= \frac{1}{N} \sum_n \left( \mathbf{u}_1^T (x_n - \bar{x}) (x_n - \bar{x})^T \mathbf{u}_1 \right) \\ &= \mathbf{u}_1^T \left[ \frac{1}{N} \sum_n (x_n - \bar{x}) (x_n - \bar{x})^T \right] \mathbf{u}_1 \\ &= \mathbf{u}_1^T S \mathbf{u}_1, \quad S = \left[ \frac{1}{N} \sum_n (x_n - \bar{x}) (x_n - \bar{x})^T \right] \end{aligned}$$

- Goal: find  $\mathbf{u}_1 \in \mathbb{R}^d$ ,  $\mathbf{u}_1^T \mathbf{u}_1 = 1$ 
  - so as to minimize  $\mathbf{u}_1^T S \mathbf{u}_1$

# PCA via Maximizing Variance

- Goal:

$$\begin{aligned} & \text{minimize} && \mathbf{u}_1^T S \mathbf{u}_1 \\ & \text{such that} && \mathbf{u}_1^T \mathbf{u}_1 = 1, \quad \mathbf{u}_1 \in \mathbb{R}^d \end{aligned}$$

- Lagrangian:

$$L(\mathbf{u}_1, \lambda_1) = \mathbf{u}_1^T S \mathbf{u}_1 - \lambda_1 (\mathbf{u}_1^T \mathbf{u}_1 - 1)$$

$$\nabla_{\mathbf{u}_1} L(\mathbf{u}_1, \lambda_1) = 2S\mathbf{u}_1 - 2\lambda_1 \mathbf{u}_1$$

- Zero-gradient point satisfies

$$S\mathbf{u}_1 = \lambda_1 \mathbf{u}_1$$

$$\mathbf{u}_1^T S \mathbf{u}_1 = \lambda_1 \mathbf{u}_1^T \mathbf{u}_1 = \lambda_1$$

# PCA via Maximizing Variance

- That is, find  $\mathbf{u}_1$ 
  - Eigenvector:  $S\mathbf{u}_1 = \lambda_1\mathbf{u}_1$
  - With maximal eigenvalue:  $\mathbf{u}_1^T S \mathbf{u}_1 = \lambda_1 \mathbf{u}_1^T \mathbf{u}_1 = \lambda_1$
  - Projection:  $x_n \rightarrow \mathbf{u}_1^T x_n, n \in [N]$
- PCA over  $\mathbb{R}^m$ ,  $m < d$ 
  - Compute covariance matrix  $S = \left[ \frac{1}{N} \sum_n (x_n - \bar{x})(x_n - \bar{x})^T \right]$
  - Compute top  $m$  eigenvectors:  $\mathbf{u}_1, \dots, \mathbf{u}_m$
  - Projection:  $x_n \rightarrow (\mathbf{u}_1^T x_n, \dots, \mathbf{u}_m^T x_n) \quad n \in [N]$

# PCA via Minimizing Projection Error

- Another view of PCA
  - Project data into lower dimensional
  - So that the square distance “movement” of data is minimized
- Orthonormal basis of  $\mathbb{R}^d$ :  $\mathbf{u}_1, \dots, \mathbf{u}_d$ 
  - (affine) projection on  $\mathbb{R}^m$ ,  $m < d$ :
$$x_n \rightarrow \tilde{x}_n = \sum_{i=1}^m z_{in} \mathbf{u}_i + \sum_{i=m+1}^d b_i \mathbf{u}_i$$
- Goal: minimize

$$J = \frac{1}{N} \sum_n (x_n - \tilde{x}_n)^T (x_n - \tilde{x}_n)$$

# PCA via Minimizing Projection Error

- Since  $\mathbf{u}_1, \dots, \mathbf{u}_d$  is orthonormal basis of  $\mathbb{R}^d$

$$x_n = \sum_{i=1}^d \alpha_{in} \mathbf{u}_i, \quad \alpha_{in} = \mathbf{u}_i^T x_n$$

- Re-writing

$$J = \frac{1}{N} \left( \sum_{i=1}^m \sum_n (\alpha_{in} - z_{in})^2 \right) + \frac{1}{N} \left( \sum_{i=m+1}^d \sum_n (\alpha_{in} - b_i)^2 \right)$$

- To minimize  $J$ , we set

$$z_{in} = \alpha_{in}, \quad i \leq m$$

$$b_i = \frac{1}{N} \sum_n \alpha_{in} = \mathbf{u}_i^T \left( \frac{1}{N} \sum_n x_n \right) = \mathbf{u}_i^T \bar{x}, \quad m < i \leq d$$

# PCA via Minimizing Projection Error

- In summary, given orthonormal basis of  $\mathbb{R}^d$ :  $\mathbf{u}_1, \dots, \mathbf{u}_d$

$$x_n = \sum_{i=1}^d \alpha_{in} \mathbf{u}_i, \quad \alpha_{in} = \mathbf{u}_i^T x_n$$

- The minimal projection error w.r.t. subspace formed by  $\mathbf{u}_1, \dots, \mathbf{u}_m$

$$J = \sum_{i=m+1}^d (\mathbf{u}_i^T \bar{x} - \mathbf{u}_i^T x_n)^2$$

$$= \sum_{i=m+1}^d \mathbf{u}_i^T S \mathbf{u}_i$$

- That is, to minimize  $J$ , we need to find  $\mathbf{u}_1, \dots, \mathbf{u}_d$

- so that the last  $(d - m)$  basis vector have minimal quadratic terms

# PCA via Minimizing Projection Error

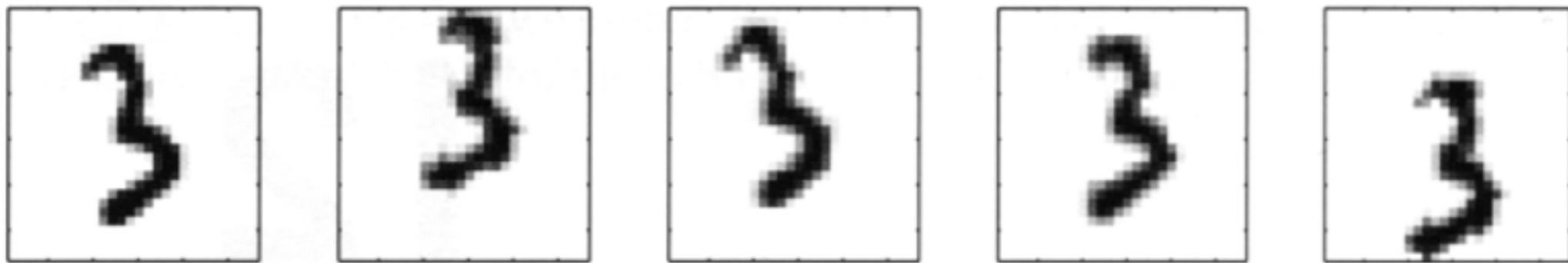
- In summary, given orthonormal basis of  $\mathbb{R}^d$ :  $\mathbf{u}_1, \dots, \mathbf{u}_d$

$$x_n = \sum_{i=1}^d \alpha_{in} \mathbf{u}_i, \quad \alpha_{in} = \mathbf{u}_i^T x_n$$

- The minimal projection error w.r.t. subspace formed by  $\mathbf{u}_1, \dots, \mathbf{u}_m$ 
  - minimizes  $\sum_{i=m+1}^d \mathbf{u}_i^T S \mathbf{u}_i$
- These turn out to selecting  $\mathbf{u}_1, \dots, \mathbf{u}_d$ 
  - as eigenvectors of  $S$
  - in decreasing order of eigenvalues!

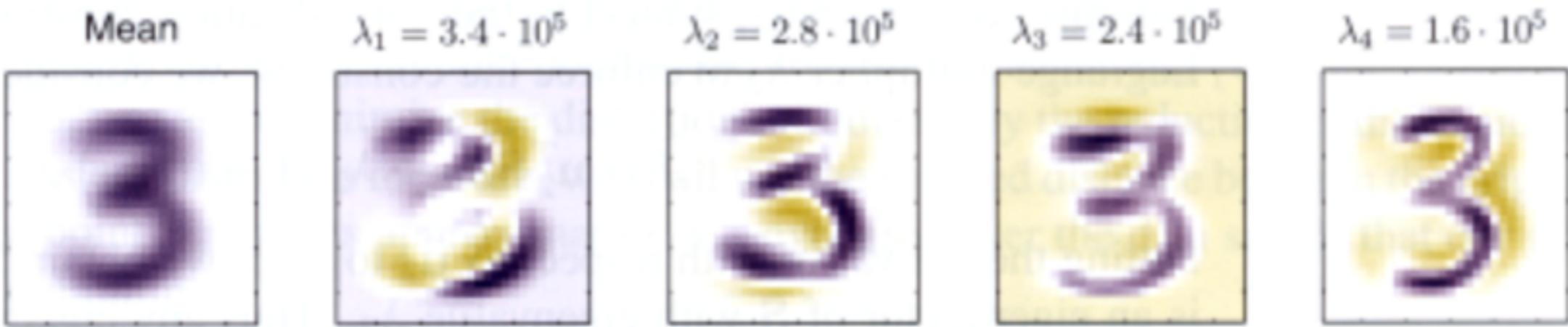
# Applications of PCA

- Visualization, data compression, data pre-processing, ...
- A Data Compression Example (Bishop, Chapter 12):



A synthetic data set obtained by taking one of the off-line digit images and creating multiple copies in each of which the digit has undergone a random displacement and rotation within some larger image field. The resulting images each have  $100 \times 100 = 10,000$  pixels.

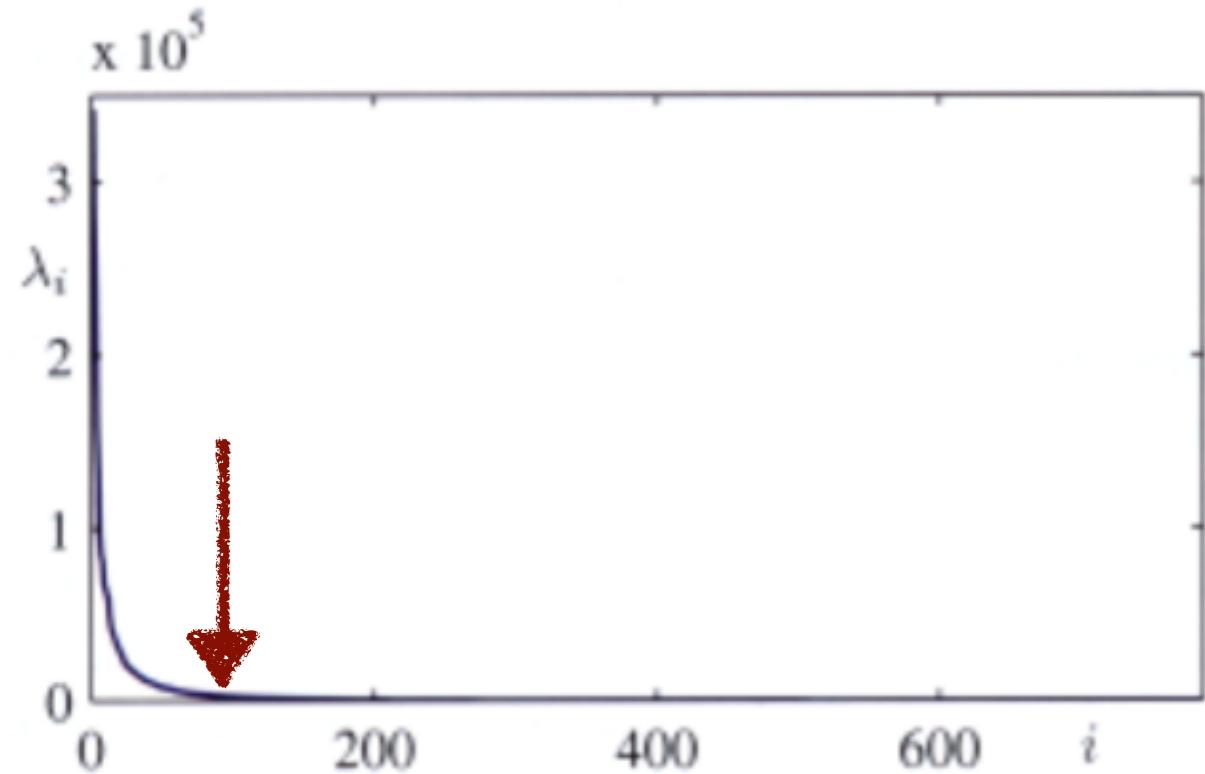
# Applications of PCA



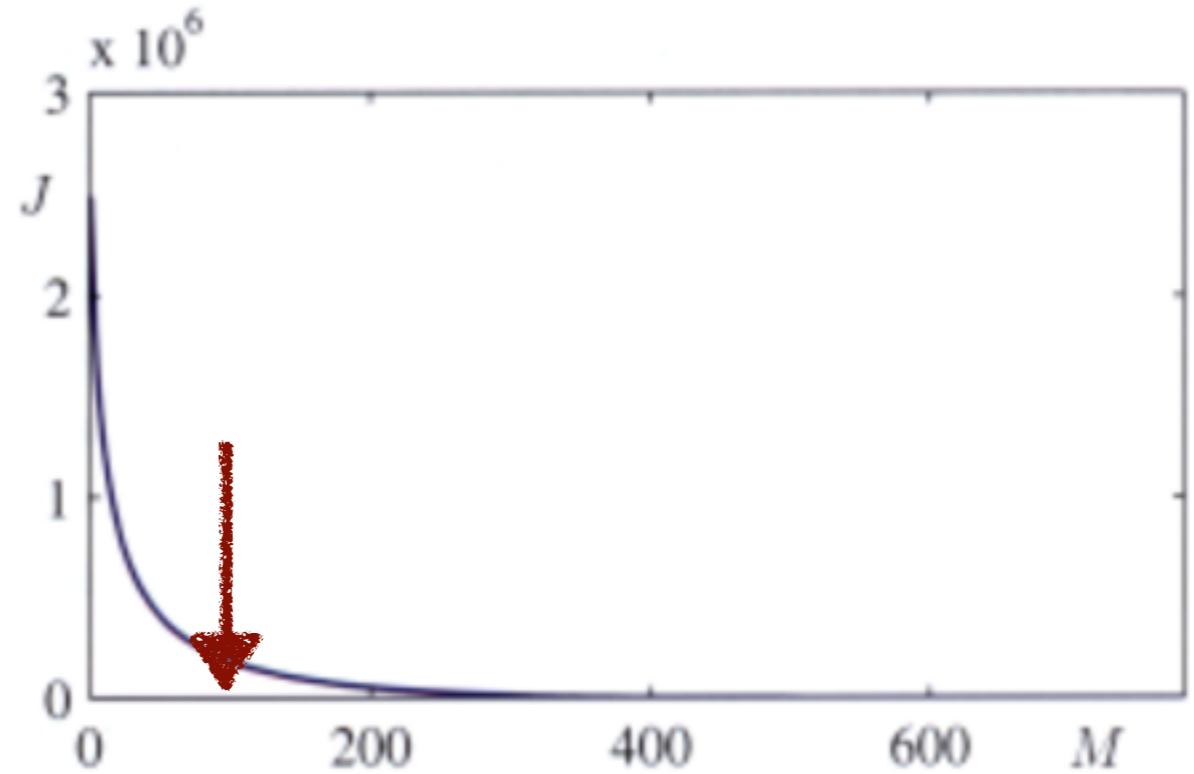
The mean vector  $\bar{x}$  along with the first four PCA eigenvectors  $u_1, \dots, u_4$  for the off-line digits data set, together with the corresponding eigenvalues.

# Applications of PCA

compressed fraction = 100/10000 = 1%



eigenvalues of correlation matrix



projection error vs projection dim

# Applications of PCA

- Principle Component Regression (PCR)
- Recall setting of Regression
  - Given observed features  $\mathbf{x} \in \mathbb{R}^d$  predict target  $\mathbf{y} \in \mathbb{R}$
  - Training data  $(x_n, y_n), 1 \leq n \leq N$
- PCR (at high level)
  - Perform PCA for data matrix of features
$$X^T X \in \mathbb{R}^{d \times d}, X = [x_n^T]$$
  - Using Ordinary Least Squares, learn model on “projected” features
  - “lift” model back to original space

# Applications of PCA

- Principle Component Regression (PCR)
  - Spectral decomposition of data matrix  $X^T X = V \Lambda V^T$
  - Projected data  $W_m = X V_m$   
 $= [X\mathbf{v}_1, \dots, X\mathbf{v}_m]$
  - PCR model estimate  
$$\hat{\beta}_m = (W_m^T W_m)^{-1} W_m^T \mathbf{y}$$
  - Final model estimate  
$$\hat{\beta} = V_m \hat{\beta}_m$$
- Benefits of PCR
  - Variance reduction, Addressing Multi-collinearity, Dimensionality Reduction, Induced Regularization, Efficiency.

# PCA: Implementation

- PCA: usually  $N < d$ 
  - Compute  $d \times d$  correlation matrix:  $N \times d \times d$
  - Computing top  $m$  eigenvectors:  $md^2$
- A useful implementation trick
  - Compute  $N \times d$  matrix:  $X = \begin{bmatrix} (x_1 - \bar{x})^T \\ \dots \\ (x_N - \bar{x})^T \end{bmatrix}$
  - Then  $S = \frac{1}{N} X^T X$
  - Therefore

$$\frac{1}{N} X^T X \mathbf{u}_i = \lambda_i \mathbf{u}_i$$

# PCA: Implementation

- Continuing:

$$\frac{1}{N} X^T X \mathbf{u}_i = \lambda_i \mathbf{u}_i \Rightarrow \frac{1}{N} X X^T (X \mathbf{u}_i) = \lambda_i (X \mathbf{u}_i)$$

- That is:

$$\frac{1}{N} X X^T \mathbf{v}_i = \lambda_i \mathbf{v}_i$$

- And again

$$\frac{1}{N} X^T X (X^T \mathbf{v}_i) = \lambda_i (X^T \mathbf{v}_i)$$

- In summary, PCA with  $N < d$ :

- Compute eigenvectors  $\mathbf{v}_i$ ,  $i \leq N$  of  $Q = \frac{1}{N} X X^T$  in  $O(N^3)$
- Obtain eigenvectors  $\mathbf{u}_i$ ,  $i \leq N$  of  $S$  as

$$\mathbf{u}_i \propto X^T \mathbf{v}_i$$

# PCA: Implementation

- In summary:
  - Let  $N \times d$  data matrix: 
$$X = \begin{bmatrix} (x_1 - \bar{x})^T \\ \dots \\ (x_N - \bar{x})^T \end{bmatrix}$$
  - Compute eigenvectors  $\mathbf{v}_i$ ,  $i \leq N$  of  $Q = \frac{1}{N}XX^T$
  - The  $m$  dimensional projection of  $N$  points is given by  $N \times m$  matrix:

$$[c_1 \mathbf{v}_1 \cdots c_m \mathbf{v}_m]$$

- Because

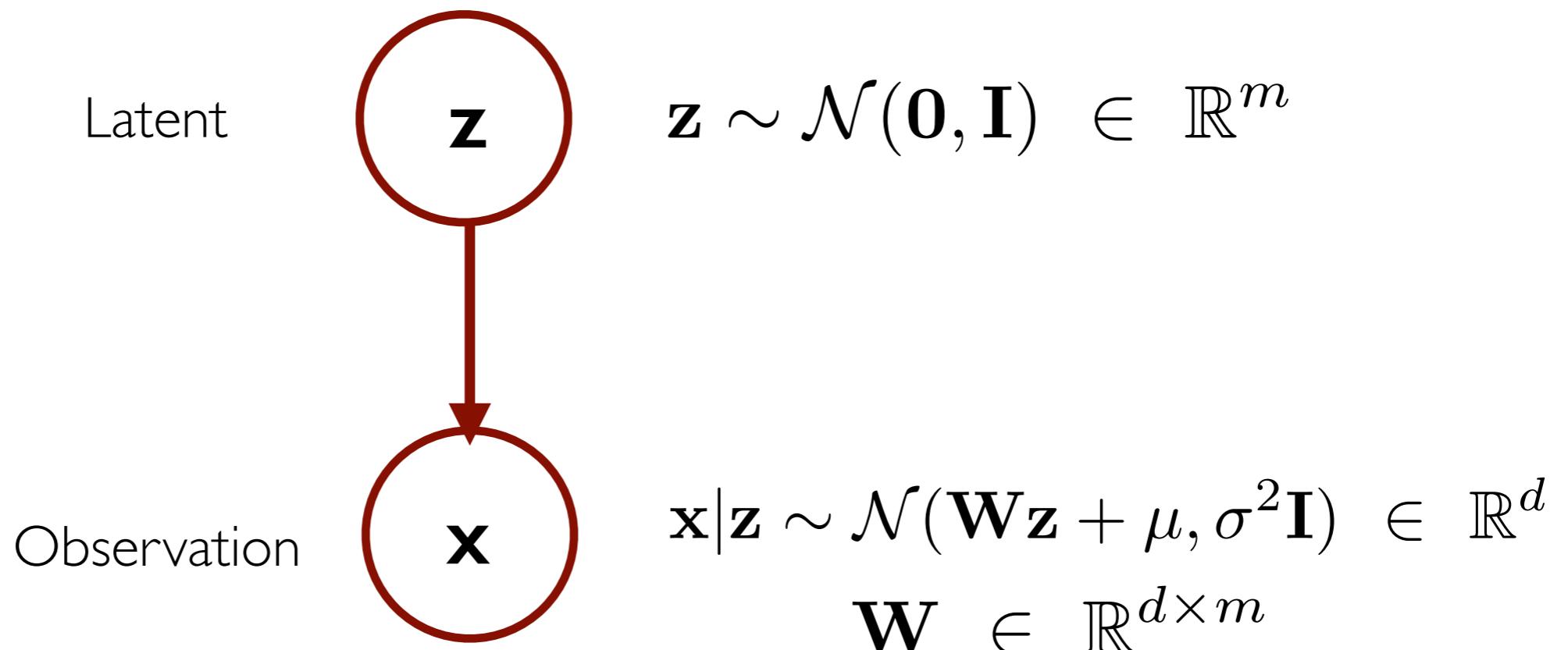
$$X\mathbf{u}_i \propto XX^T\mathbf{v}_i$$

$$\propto Q\mathbf{v}_i$$

$$\propto \lambda_i \mathbf{v}_i$$

# Probabilistic PCA

- Probabilistic latent variable model:



- Equivalently:

$$\mathbf{x} = \mathbf{Wz} + \mu + \varepsilon, \quad \varepsilon \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}) \in \mathbb{R}^d$$

# Probabilistic PCA

- Probabilistic latent variable model:

$$\mathbf{x} = \mathbf{W}\mathbf{z} + \mu + \varepsilon, \quad \varepsilon \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}) \in \mathbb{R}^d$$

- Hence

$$\mathbf{x} \sim \mathcal{N}(\mu_{\mathbf{x}}, \Sigma_{\mathbf{x}}) \in \mathbb{R}^d$$

$$\begin{aligned}\mu_{\mathbf{x}} &= \mathbb{E}[\mathbf{x}] \\ &= \mathbf{W}\mathbb{E}[\mathbf{z}] + \mu \\ &= \mu\end{aligned}$$

$$\begin{aligned}\Sigma_{\mathbf{x}} &= \mathbb{E}[\mathbf{x}\mathbf{x}^T] - \mu\mu^T \\ &= \mathbf{W}\mathbb{E}[\mathbf{z}\mathbf{z}^T]\mathbf{W} + \mathbb{E}[\varepsilon\varepsilon^T] \\ &= \mathbf{W}\mathbf{W}^T + \sigma^2 \mathbf{I} \\ &\equiv \mathbf{C} \in \mathbb{R}^{d \times d}\end{aligned}$$

# Probabilistic PCA

- Probabilistic latent variable model:

$$\mathbf{x} = \mathbf{W}\mathbf{z} + \mu + \varepsilon, \quad \varepsilon \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I}) \in \mathbb{R}^d$$

- Conversely

$$\mathbf{z}|\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{z}|\mathbf{x}}, \boldsymbol{\Sigma}_{\mathbf{z}|\mathbf{x}})$$

$$\boldsymbol{\mu}_{\mathbf{z}|\mathbf{x}} = \mathbf{M}^{-1} \mathbf{W}^T (\mathbf{x} - \boldsymbol{\mu}) \qquad \qquad \boldsymbol{\Sigma}_{\mathbf{z}|\mathbf{x}} = \sigma^2 \mathbf{M}^{-1}$$

where  $\mathbf{M} = \mathbf{W}^T \mathbf{W} + \sigma^2 \mathbf{I} \in \mathbb{R}^{m \times m}$

# Probabilistic PCA

- Maximum Likelihood Estimator
  - data:  $x_1, \dots, x_N$
  - (model) parameters:  $\mu, \mathbf{W}, \sigma^2$
  - log-likelihood: recall  $\mathbf{C} = \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I} \in \mathbb{R}^{d \times d}$ 
$$-\log \mathbb{P}(x_1, \dots, x_N | \mu, \mathbf{W}, \sigma^2) \propto \frac{1}{2} \sum_{n=1}^N (x_n - \mu)^T \mathbf{C}^{-1} (x_n - \mu)$$
$$+ \frac{N}{2} \log |\mathbf{C}|$$
$$\equiv F(\mu, \mathbf{C})$$
- Goal: find  $\mu, \mathbf{C}$  that minimizes  $F(\mu, \mathbf{C})$

# Probabilistic PCA

- Goal: find  $\mu, \mathbf{C}$  that minimizes  $F(\mu, \mathbf{C})$
- That is, we need Maximum Likelihood Estimation for parameter of Gaussian
- As we have learnt, its simply empirical mean and covariance

$$\mu = \bar{x} = \frac{1}{N} \sum_n x_n$$

$$\mathbf{C} = S = \frac{1}{N} X^T X$$

with 
$$X = \begin{bmatrix} (x_1 - \bar{x})^T \\ \dots \\ (x_N - \bar{x})^T \end{bmatrix}$$

# Probabilistic PCA

- That is, find  $\mathbf{W}$ ,  $\sigma^2$  such that

$$S \approx \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I}$$

- If  $\sigma = 0$  and since  $\mathbf{W} \in \mathbb{R}^{d \times m}$ 
    - We are looking for best rank  $m$  approximation of  $S$
    - Then  $\mathbf{W} = \sum_{i=1}^m \lambda_i u_i v_i^T$
    - where
- $$S = \sum_{i=1}^d \lambda_i u_i v_i^T \quad \lambda_1 \geq \dots \geq \lambda_d \geq 0$$

# Probabilistic PCA

- The corresponding mean latent variable for a given data  $\mathbf{x}$

$$\begin{aligned}\mathbb{E}[\mathbf{z}|\mathbf{x}] &= \mathbf{M}^{-1} \mathbf{W}^T (\mathbf{x} - \boldsymbol{\mu}) \\ &= (\mathbf{W}^T \mathbf{W})^{-1} \mathbf{W}^T (\mathbf{x} - \bar{\mathbf{x}})\end{aligned}$$

- That is, projection of  $\mathbf{x} - \bar{\mathbf{x}}$  on space spanned by columns of  $\mathbf{W}$
- That is, we recover the PCA algorithm:
  - Identify top  $m$  eigenvectors of
  - Project data points on space spanned by these  $m$  eigenvectors  $\mathbf{S}$

# Probabilistic PCA: via EM

- Starting point of EM is maximum likelihood:

$$\begin{aligned} & \text{maximize } \mathbb{E}_{\mathbf{z}} \left[ \log \mathbb{P}(x_1, \dots, x_n; z_1, \dots, z_n | \mathbf{W}, \mu, \sigma^2) \right] \\ & \text{over } \mu, \mathbf{W}, \sigma^2 \end{aligned}$$

- The log likelihood can be written as

$$\begin{aligned} -\mathbb{E}_{\mathbf{z}} \left[ \log \mathbb{P}(x_1, \dots, x_n; z_1, \dots, z_n) \right] &= - \sum_{n=1}^N \mathbb{E}_{\mathbf{z}} \left[ \log \mathbb{P}(x_n | z_n) + \log \mathbb{P}(z_n) \right] \\ &\propto \sum_{n=1}^N \frac{d}{2} \log(2\pi\sigma^2) + \frac{1}{2} \text{Tr}(\mathbb{E}_{\mathbf{z}}[z_n z_n^T]) + \frac{1}{2\sigma^2} \|x_n - \mu\|^2 \\ &\quad - \frac{1}{\sigma^2} \mathbb{E}_{\mathbf{z}}[z_n^T] \mathbf{W}^T (x_n - \mu) + \frac{1}{2\sigma^2} \text{Tr}(\mathbb{E}_{\mathbf{z}}[z_n z_n^T] \mathbf{W}^T \mathbf{W}) \end{aligned}$$

# Probabilistic PCA

- EM suggests iterative way to solve this:
  - E-step: assume parameters fixed
    - evaluate conditional distribution of  $\mathbf{z}$  given the parameters
  - M-step: maximize overall log-likelihood over choice of parameters
    - using the conditional distribution of  $\mathbf{z}$  evaluated in E-step
  - Iterate the above E and M steps
- Let's see these in our context next

# Probabilistic PCA

- E-step

$$\mathbb{E}[z_n] = \mathbf{M}^{-1} \mathbf{W}^T (x_n - \bar{x})$$

$$\begin{aligned}\mathbb{E}[z_n z_n^T] &= \text{Cov}[z_n] + \mathbb{E}[z_n] \mathbb{E}[z_n]^T \\ &= \sigma^2 \mathbf{M}^{-1} + \mathbf{M}^{-1} \mathbf{W}^T (x_n - \bar{x}) (x_n - \bar{x})^T \mathbf{W} \mathbf{M}^{-1}\end{aligned}$$

- M-step

$$\mathbf{W}_{\text{new}} = \left[ \sum_{n=1}^N (x_n - \bar{x}) \mathbb{E}[z_n]^T \right] \left[ \sum_{n=1}^N \mathbb{E}[z_n z_n^T] \right]^{-1}$$

$$\begin{aligned}\sigma_{\text{new}}^2 &= \frac{1}{Nd} \sum_{n=1}^N \left\{ \|x_n - \bar{x}\|^2 - 2\mathbb{E}[z_n]^T \mathbf{W}_{\text{new}}^T (x_n - \bar{x}) \right. \\ &\quad \left. + \text{Tr}(\mathbb{E}[z_n z_n^T] \mathbf{W}_{\text{new}}^T \mathbf{W}_{\text{new}}) \right\}\end{aligned}$$

# Probabilistic PCA

- Assuming normalized data ( $\bar{x} = \mathbf{0}$ ), PCA is utilizes matrix

$$S = \frac{1}{N} \sum_n x_n x_n^T$$

- Consider non-linear transformation:  $\mathbf{x} \rightarrow \phi(\mathbf{x})$

$$S' = \frac{1}{N} \sum_n \phi(x_n) \phi(x_n)^T$$

- And let top  $m$  eigenvectors of  $S'$  be

$$S' \mathbf{v}_i = \lambda_i \mathbf{v}_i, \quad i \in [m]$$

# Probabilistic PCA

- That is

$$\frac{1}{N} \sum_n \phi(x_n) (\phi(x_n)^T \mathbf{v}_i) = \lambda_i \mathbf{v}_i$$

- Equivalently

$$\mathbf{v}_i = \sum_n a_{in} \phi(x_n)$$

- Using this back into definition of eigenvector, we obtain

$$\frac{1}{N} \sum_n \phi(x_n) \phi(x_n)^T \sum_{p=1}^N a_{ip} \phi(x_p) = \lambda_i \sum_{p=1}^N a_{ip} \phi(x_p)$$

# Probabilistic PCA

- Define kernel

$$k(x, x') = \phi(x)^T \phi(x')$$

- Multiplying both sides with  $\phi(x_\ell)$

$$\frac{1}{N} \sum_n \phi(x_\ell)^T \phi(x_n) \sum_{p=1}^N a_{ip} \phi(x_n)^T \phi(x_p) = \lambda_i \sum_{p=1}^N a_{ip} \phi(x_\ell)^T \phi(x_p)$$

- That is

$$\frac{1}{N} \sum_n K(x_\ell, x_n) \sum_{p=1}^N a_{ip} K(x_n, x_p) = \lambda_i \sum_{p=1}^N a_{ip} K(x_\ell, x_p)$$

- Equivalently

$$\mathbf{K}^2 \mathbf{a}_i = \lambda_i N \mathbf{K} \mathbf{a}_i \quad \Rightarrow \quad \mathbf{K} \mathbf{a}_i = \lambda_i N \mathbf{a}_i$$

# Probabilistic PCA

- Projection:

$$\begin{aligned}y_i(x) &= \phi(x)^T \mathbf{v}_i \\&= \sum_n a_{in} \phi(x)^T \phi(x_n) \\&= \sum_n a_{in} K(x, x_n)\end{aligned}$$

- This suggests a way to find projection of a new data point *without* re-computing PCA!

# Beyond PCA

- Independent component analysis
  - When distribution over latent variables factorizes
- Auto-encoder
  - Using neural networks for modeling latent variable model
  - We've seen already seen this in class
- Matrix latent variable model
  - Next