Principal Component Analysis (PCA)
CSC411/2515 Tutorial

Harris Chan
Based on previous tutorial slides by Wenjie Luo, Ladislav Rampasek

University of Toronto

hchan@cs.toronto.edu

October 19th, 2017

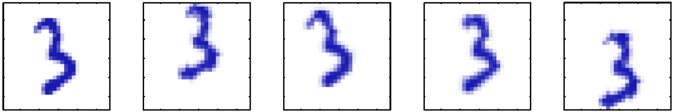
Overview

Dimensionality Reduction

- We have some data $X \in \mathbb{R}^{N \times D}$, where D can be very large.
- We want a new representation of the data $Z \in \mathbb{R}^{N \times K}$ where $K << D$.
  - For computational reasons
  - To better understand / visualize the data
  - For compression
  - etc.
- We will restrict ourselves to textbflinear transformation.

Example

- In this dataset, there are only 3 degrees of freedom: (1) horizontal translations; (2) vertical translations; (3) Rotations.



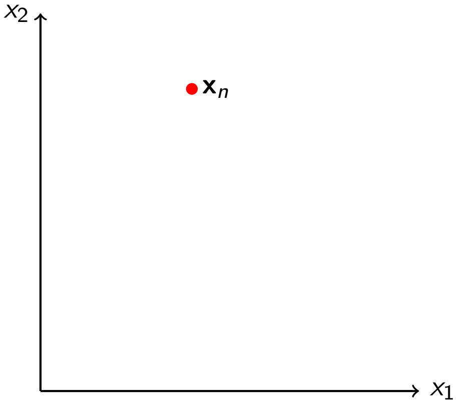
- But each image is $100 \times 100 = 10000$ pixels, so X will be 10000 elements wide!

What is a Good Transformation?

- The goal is to find good directions u that preserves "important" aspects of the data
- In linear setting: $z = x^T u$
- This will turn out to be the **top-K eigenvalues of the data covariance**.
- 2 ways to view this:
  1. Find directions of *maximum variation*
  2. Find projections that *minimizes the reconstruction error*

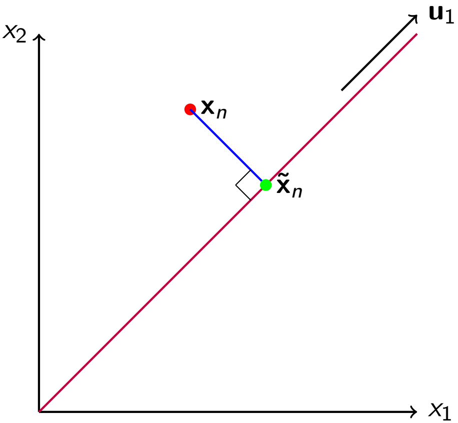
Two Derivations of PCA

Consider the n-th datapoint $\mathbf{x}_n$ that has 2 dimensions, x_1 and x_2:



Two Derivations of PCA

We can pick a direction $\mathbf{u}_1$ to project $\mathbf{x}_n$ onto, creating a projected point $\tilde{\mathbf{x}}_n$:



Two Derivations of PCA

By Pythagorean theorem:

$$\underbrace{R^2}_{\textit{Original Dist}} = \underbrace{D^2}_{\textit{Variance}} + \underbrace{\epsilon^2}_{\textit{Reconstr. Err}}$$

Since R^2 is fixed:

Problem Equivalence

Maximize D^2 *(variance)*
$\Leftrightarrow$ *Minimize* ϵ^2 *(reconstruction error)*

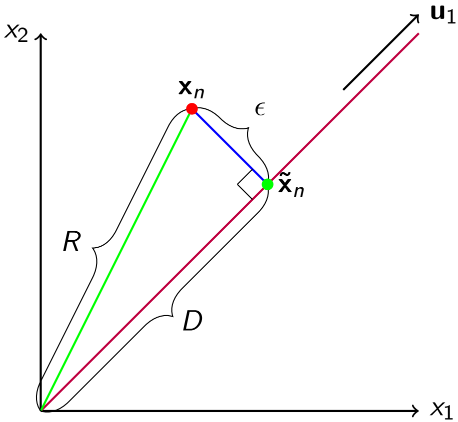
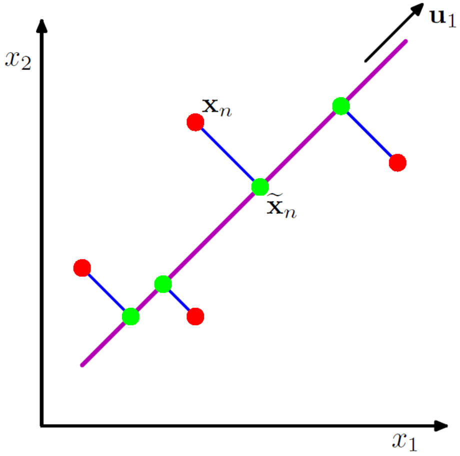


Figure 12.2 from Bishop's Textbook:



Principal Component Analysis: Maximum Variance

- Our goal is to maximize the variance of the projected data:

$$maximize \frac{1}{2N}\sum_{n=1}^{N}(\mathbf{u}_1^T\mathbf{x}_n - \mathbf{u}_1^T\bar{\mathbf{x}}_n) = \mathbf{u}_1^T S\mathbf{u}_1 \tag{1}$$

- Where the sample mean and covariance is given by:

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n \tag{2}$$

$$S = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^T \tag{3}$$

$$\tag{4}$$

Lagrange Multiplier

- If we want to find a stationary point of a function of multiple variables $f(\mathbf{x})$ subject to one or more constraints $g(\mathbf{x}) = 0$:
  1. Introduce Lagrangian function:

$$L(\mathbf{x}, \lambda) \equiv f(\mathbf{x}) + \lambda g(\mathbf{x}) \tag{5}$$

  2. Find its stationary point w.r.t. both x and λ
- If you are not familiar with it, check out Appendix E in Bishop's book

Finding $\mathbf{u}_1$

- We want to maximize $\mathbf{u}_1^T S \mathbf{u}_1$ subject to $\|\mathbf{u}_1\| = 1$ (since we are finding direction)
- Use Lagrange multiplier α_1 to express this as:

$$\mathbf{u}_1^T S \mathbf{u}_1 + \alpha_1 (1 - \mathbf{u}_1^T \mathbf{u}_1) \tag{6}$$

- Take derivative and set to 0:

$$S\mathbf{u}_1 - \alpha_1 \mathbf{u}_1 = 0 \tag{7}$$

$$S\mathbf{u}_1 = \alpha_1 \mathbf{u}_1 \tag{8}$$

- So $\mathbf{u}_1$ is an eigenvector of S with eigenvalue α_1
- In fact, it must be the eigenvector with the maximum eigenvalue, since this maximizes the objective

Finding $\mathbf{u}_2$

- We want to maximize $\mathbf{u}_2^T S \mathbf{u}_2$ subject to $\|\mathbf{u}_2\| = 1$ and $\mathbf{u}_2^T \mathbf{u}_1 = 0$ (orthogonal to $\mathbf{u}_1$)
- Use Lagrange form:

$$\mathbf{u}_s^T S \mathbf{u}_s + \alpha_s (1 - \mathbf{u}_s^T \mathbf{u}_2) - \beta \mathbf{u}_2^T \mathbf{u}_1 \tag{9}$$

- Take derivative and set to 0 to find β:

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{u}_2} = S\mathbf{u}_2 - \alpha_2 \mathbf{u}_2 - \beta \mathbf{u}_1 = 0 && (10)\\
\implies \mathbf{u}_1^T S \mathbf{u}_2 - \alpha_2 \mathbf{u}_1^T \mathbf{u}_2 - \beta \mathbf{u}_1^T \mathbf{u}_1 = 0 && (11)\\
\implies \alpha_1 \mathbf{u}_1^T \mathbf{u}_2 - \alpha_2 \mathbf{u}_1^T \mathbf{u}_2 - \beta \mathbf{u}_1^T \mathbf{u}_1 = 0 && (12)\\
\implies \alpha_1 \cdot 0 - \alpha_2 \cdot 0 - \beta \cdot 1 = 0 && (13)\\
\implies \beta = 0 && (14)
\end{aligned}$$

Finding $\mathbf{u}_2$

- We want to maximize $\mathbf{u}_2^T S \mathbf{u}_2$ subject to $\|\mathbf{u}_2\| = 1$ and $\mathbf{u}_2^T \mathbf{u}_1 = 0$ (orthogonal to $\mathbf{u}_1$)
- Use Lagrange form:

$$\mathbf{u}_s^T S \mathbf{u}_s + \alpha_s(1 - \mathbf{u}_s^T \mathbf{u}_2) - \underbrace{\beta \mathbf{u}_2^T \mathbf{u}_1}_{0} \tag{15}$$

- Take derivative and set to 0 to find α_2:

$$\frac{\partial}{\partial \mathbf{u}_2} = S\mathbf{u}_2 - \alpha_2 \mathbf{u}_2 = 0 \tag{16}$$

$$\implies S\mathbf{u}_2 = \alpha_2 \mathbf{u}_2 \tag{17}$$

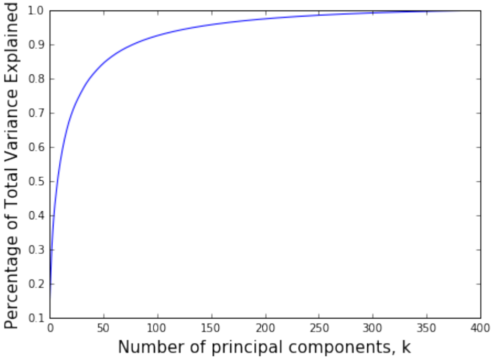
- So α_2 must be the second largest eigenvalue of S.

PCA In General

- We can compute the entire PCA solution by just computing the eigenvectors with the top-K eigenvalues.
- These can be found using the singular value decomposition (SVD) of S.

Choosing the number of K

- How do we choose the number of components?
- Idea: Look at the spectrum of covariance, pick K to capture most of the variation



- More principled: Bayesian treatment (beyond this course)

Principal Component Analysis: Minimum Reconstruction Error

- We can also think of PCA as minimizing the *reconstruction error* of compressed data:

$$\textit{minimize}\ \frac{1}{2N}\sum_{n=1}^{N}\|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2 \tag{18}$$

- We will omit some details for now, but the key is that we define some K-dimensional basis such that:

$$\tilde{\mathbf{x}} = W\mathbf{x} + \textit{const} \tag{19}$$

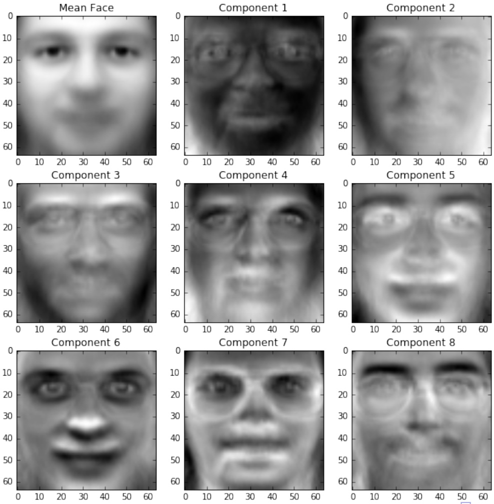
- The solution will turn out to be the same as the maximum variance formulation

PCA Demo

We'll apply PCA using scikit-learn in Python on various datasets for visualization / compression:

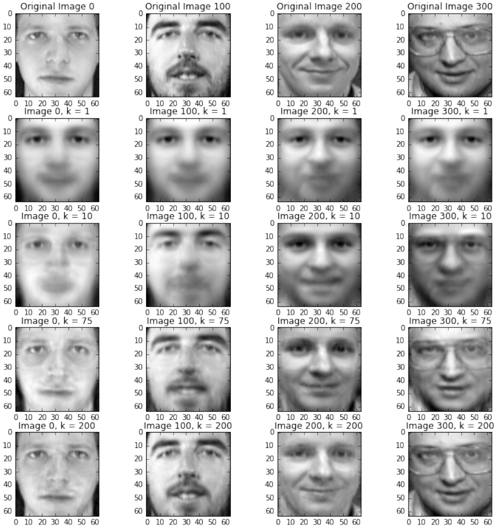
- Synthetic 2D data: Show the principal components learned and what the transformed data looks like
- MNIST digits: Compression and Reconstruction
- Olivetti faces dataset: Compression and Reconstruction
- Iris dataset: Visualization

For example: Olivetti Faces dataset. Apply PCA on the face images to find the principle components, and project the data down to k-dimensions



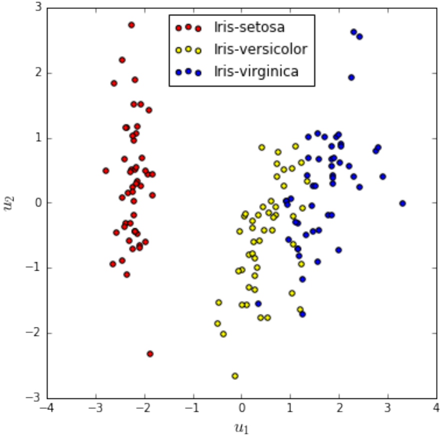
PCA Application: Compression & Reconstruction

Reconstruction when using various values of k:



PCA Application: Visualization

- PCA can be used to find the 'best' viewing angle to project onto a 2-D plane (or 3D) to better understand the data
- Example on the Iris dataset:



Summary

- PCA is a linear projection of D-dimensional $\{\mathbf{x}_n\}$ to $K \leq D$ vector space given by $\{\mathbf{u}_k\}$ basis vectors such that it:
  - Maximizes variance in the projected data points
  - Minimizes projection error (square loss)
  - $\{\mathbf{u}_k\}$ are orthonormal
  - $\{\mathbf{u}_k\}$ turns out to be the first K eigenvectors of the data covariance matrix with K largest eigenvalues
  - Can be computed in $O(KD^2)$

Summary

- PCA is good for:
  - Dimensionality reduction
  - Visualization
  - Compression (with loss)
  - Denoising (by removing small variances in the data)
  - Can be used for data **whitening** = decorrelation, so that features have unit covariance
- Caution! In classification task, if the class labels' signal in the data has small variance, PCA may remove it completely

Thanks!