Mathematics of Image and Data Analysis
Math 5467

Lecture 5: Principal Component Analysis

Instructor: Jeff Calder
Email: jcalder@umn.edu

http://www-users.math.umn.edu/~jwcalder/5467S21

Last time

- Principal Component Analysis (PCA) Theory

Today

Applications of PCA:

- PCA-based image compression
- PCA-based handwritten digit recognition

Principal Component Analysis (PCA)

Given points $x_1, x_2, \ldots, x_m$ in $\mathbb{R}^n$, find the k-dimensional linear or affine subspace that "best fits" the data in the mean-squared sense. That is, we seek an affine subspace $A = x_0 + L$ that minimizes the energy

$$E(x_0, L) = \sum_{i=1}^{m} \|x_i - \mathrm{Proj}_A x_i\|^2.$$

PCA: Set $X = \begin{bmatrix} x_1 & x_2 & \cdots & x_m \end{bmatrix}^T$.

1. (Optional) Center the data $X = X - x_0$, where $x_0 = \frac{1}{m}\sum_{i=1}^{m} x_i$.
2. Find the top k eigenvectors $p_1, \ldots, p_k$ of the covariance matrix $M = X^T X$.
3. The best linear subspace that fits X is

$$L = \mathrm{span}\{p_1, \ldots, p_k\}.$$

If the data was centered, then the affine space of best fit is $x_0 + L$.

Projection vs dimension reduction

Let us write

$$P_k = \begin{bmatrix} p_1 & p_2 & \cdots & p_k \end{bmatrix},$$

and $L = \operatorname{span}(p_1, \ldots, p_k)$.

Then for $x \in \mathbb{R}^n$:

- Projection: $\operatorname{Proj}_L x = P_k P_k^T x \in \mathbb{R}^n$.
- Coordinates of x in L: $P_k^T x \in \mathbb{R}^k$.

For $X = \begin{bmatrix} x_1 & x_2 & \cdots & x_m \end{bmatrix}^T$ the equivalent formulas are

- Projection: $X P_k P_k^T \in \mathbb{R}^{m \times n}$.
- Coordinates of X in L: $X P_k \in \mathbb{R}^{m \times k}$.

PCA for image compression

An image is just a matrix X, where $X(i,j)$ is the brightness at pixel (i,j).

Naive PCA-based compression is

$$X \approx X P_k P_k^T.$$

Why is this compression?

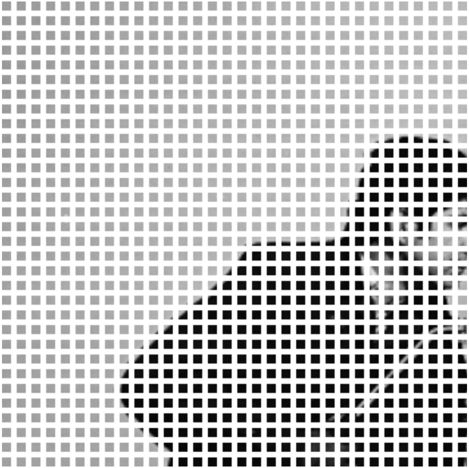
- We store $XP_k \in \mathbb{R}^{m\times k}$ and $P_k \in \mathbb{R}^{n\times k}$ instead of $X \in \mathbb{R}^{m\times n}$.
  - If $m = n$, then compression ratio is n to $2k$.

This would work well if the row-space of X has some low-dimensional structure.

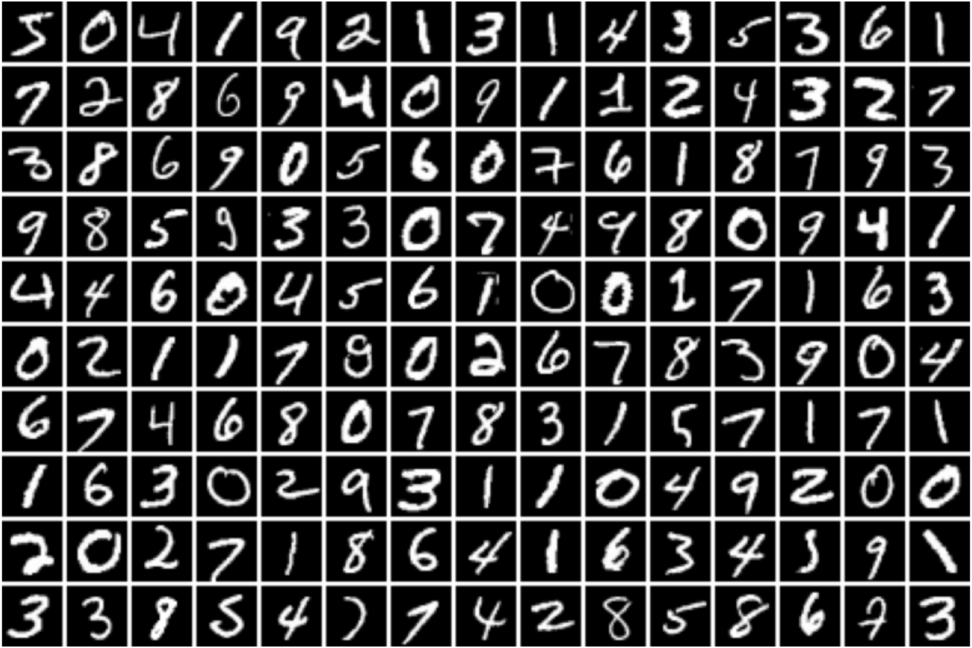
- Not usually true for images.

Patch-based compression (.ipynb)

A better idea is to split the image into blocks or patches.



Handwritten digit recognition



Handwritten digit recognition (.ipynb)

IDEA: Use PCA to learn low dimensional affine spaces $A_0, A_1, \ldots, A_9$ approximating each MNIST digit. To classify a new image x, we find the closest affine space by minimizing the residual

$$d_i(x) := \|x - \mathrm{Proj}_{A_i} x\| = \|(I - P_i P_i^T)(x - \overline{x}_i)\|$$

over $i = 0, 1, \ldots, 9$.

In Python, it is easier to do the computation above with a data matrix

$$X = \begin{bmatrix} x_1 & x_2 & \cdots & x_m \end{bmatrix}^T$$

in which case the formula is

$$X - \mathrm{Proj}_{A_i} X = (X - \overline{x}_i)(I - P_i P_i^T).$$