

Basic Numerical Optimization

Note: the slides are based on EE263 at Stanford. Reorganized, revised, and typed by Hao Su

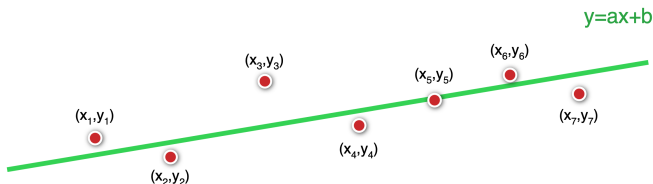
Outline

- ▶ Least-squares
 - ▶ least-squares (approximate) solution of overdetermined equations
 - ▶ minimal norm solution of underdetermined equations
 - ▶ unified solution form by SVD
- ▶ Low-rank Approximation
 - ▶ eigenface problem
 - ▶ principal component analysis

Outline

- ▶ Least-squares
 - ▶ least-squares (approximate) solution of overdetermined equations
 - ▶ least-norm solution of underdetermined equations
 - ▶ unified solution form by SVD
- ▶ Low-rank Approximation
 - ▶ eigenface problem
 - ▶ principal component analysis

Example Application: Line Fitting



- ▶ Given $\{(x_i, y_i)\}$, find line through them.
i.e., find a and b in $y = ax + b$
- ▶ Using matrix and vectors, we look for a and b such that

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \approx \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix}$$

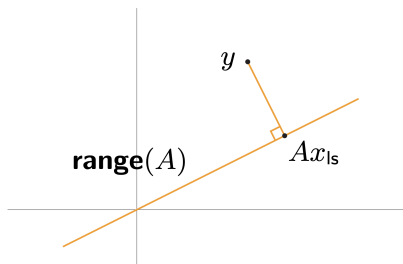
Overdetermined Linear Equations

- ▶ consider $y = Ax$ where $A \in \mathbb{R}^{m \times n}$ is (strictly) skinny, i.e., $m > n$
 - ▶ called *overdetermined* set of linear equations (more equations than unknowns)
 - ▶ for most y , cannot solve for x
- ▶ one approach to *approximately* solve $y = Ax$:
 - ▶ define *residual* or error $r = Ax - y$
 - ▶ find $x = x_{ls}$ that minimize $\|r\|$
- ▶ x_{ls} called *least-squares* (approximate) solution of $y = Ax$

Geometric Interpretation

Given $y \in \mathbb{R}^m$, find $x \in \mathbb{R}^n$ to minimize $\|Ax - y\|$

Ax_{ls} is point in **range**(A) closest to y (Ax_{ls} is *projection* of y onto **range**(A))



Least-squares (approximate) Solution

- ▶ assume A is full rank, skinny
- ▶ to find x_{ls} , we'll minimize norm of residual squared,

$$\|r\|^2 = x^T A^T A x - 2y^T A x + y^T y$$

- ▶ set gradient w.r.t. x to zero:

$$\nabla_x \|r\|^2 = 2A^T A x - 2A^T y = 0$$

- ▶ yields the *normal equation*: $A^T A x = A^T y$
- ▶ assumptions imply $A^T A$ invertible, so we have

$$x_{ls} = (A^T A)^{-1} A^T y$$

... a very famous formula

Least-squares (approximate) Solution

- ▶ x_{ls} is linear function of y
- ▶ $x_{ls} = A^{-1}y$ if A is square
- ▶ x_{ls} solves $y = Ax_{ls}$ if $y \in \mathbf{range}(A)$

Least-squares (approximate) Solution

for A skinny and full rank, the *pseudo-inverse* of A is

$$A^\dagger = (A^T A)^{-1} A^T$$

- ▶ for A skinny and full rank, A^\dagger is a *left inverse* of A

$$A^\dagger A = (A^T A)^{-1} A^T A = I$$

- ▶ if A is not skinny and full rank then A^\dagger has a different definition

Underdetermined Linear Equations

- ▶ consider $y = Ax$ where $A \in \mathbb{R}^{m \times n}$ is (strictly) fat, i.e., $m < n$
 - ▶ called *underdetermined* set of linear equations (more unknowns than equations)
 - ▶ the solution may not be unique
- ▶ we find a specific solution to $y = Ax$ and the null space of A :

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \frac{1}{2} \|x\|^2 \\ \text{s.t.} & y = Ax \end{array}$$

- ▶ this is called the least-norm solution

Least-norm Solution

$$\begin{array}{ll}\underset{x}{\text{minimize}} & \frac{1}{2}\|x\|^2 \\ \text{s.t.} & y = Ax\end{array}$$

- ▶ assume A is full (row-)rank, fat
- ▶ we use Lagrangian multiplier method to solve x :

$$\begin{aligned}L(x, \lambda) &= \frac{1}{2}\|x\|^2 - \lambda^T(y - Ax) \\ \nabla_x L(x, \lambda) &= x - A^T \lambda\end{aligned}$$

Set $\nabla_x L(x, \lambda) = 0$, we have $x = A^T \lambda$, so $y = Ax = AA^T \lambda$

Note that A is fat and full rank, so AA^T invertible

So, $\lambda = (AA^T)^{-1}y$ By $x = A^T \lambda$, we have

$$x = A^T(AA^T)^{-1}y$$

Least-norm Solution

for A fat and full rank, the *pseudo-inverse* of A is

$$A^\dagger = A^T(AA^T)^{-1}$$

- ▶ for A fat and full rank, A^\dagger is a *right inverse* of A

$$AA^\dagger = AA^T(AA^T)^{-1} = I$$

Unifying least-square and least-norm solutions by SVD

Let the SVD decomposition of A be $A = U\Sigma V^T$ (the economic form of Σ that all the diagonals are non-zero).

- ▶ For skinny matrix, the least-square solution:

$$x = (A^T A)^{-1} A^T y = V \Sigma^{-1} U^T y$$

- ▶ For fat matrix, the least-norm solution:

$$x = A^T (A A^T)^{-1} y = V \Sigma^{-1} U^T y$$

Solution to linear equation system $y = Ax$

$$x = V \Sigma^{-1} U^T y$$

Note:

- ▶ For least-norm solution, $x = V \Sigma^{-1} U^T y$ is a special solution
- ▶ Ex: how to obtain all the solutions? (Hint: the null space of U^T)

Outline

- ▶ Least-squares
 - ▶ least-squares (approximate) solution of overdetermined equations
 - ▶ least-norm solution of underdetermined equations
 - ▶ unified solution form by SVD
- ▶ Low-rank Approximation
 - ▶ eigenface problem
 - ▶ principal component analysis

Example Application: Face Retrieval

Suppose you have *10 million* face images,

Question:

- ▶ How can you find the 5 faces closest to a query (maybe yours!) in just 0.1 sec?
- ▶ How can you show all of them in a single picture?

Example Application: Face Retrieval

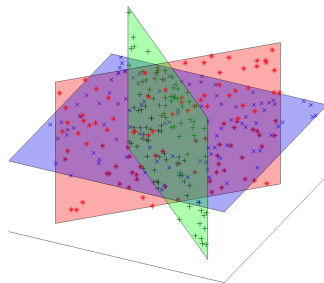
Suppose you have *10 million* face images,

Question:

- ▶ How can you find the 5 faces closest to a query (maybe yours!) in just 0.1 sec?
- ▶ How can you show all of them in a single picture?
- ▶ *SVD can help you do it!*

Data as Points in a Euclidean Space

- ▶ While data can be represented as high-dimensional vectors
- ▶ *Lower-dimensional structure* is often present in data



The Space of All Face Images

- ▶ When viewed as vectors of pixel values, face images are extremely high-dimensional
 - ▶ 100×100 image=10,000 dims
 - ▶ Slow and lots of storage is needed
- ▶ But very few 10,000-dimensional vectors are valid face images
- ▶ We want to *effectively* model the subspace of face images

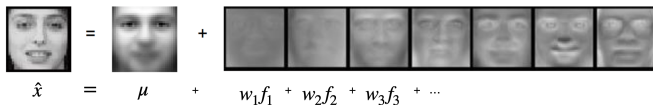
Low-Dimensional Face Space



$$\hat{x} = \mu + w_1 f_1 + w_2 f_2 + w_3 f_3 + \dots$$

The diagram shows a target face image \hat{x} on the left, followed by an equals sign. To the right of the equals sign is a mean face image μ , followed by a plus sign. To the right of the plus sign is a row of seven grayscale images representing basis functions f_1, f_2, f_3, \dots . Each basis function is preceded by a weight w_1, w_2, w_3, \dots and a plus sign. The basis functions show localized variations in facial features, such as eyes, nose, mouth, and glasses.

Reconstruction Formulation


$$\hat{x} = \mu + w_1 f_1 + w_2 f_2 + w_3 f_3 + \dots$$

- ▶ Data matrix of face images: $X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_n^T \end{bmatrix} \in \mathbb{R}^{n \times m}$, each row is a face image

- ▶ Orthonormal basis of the face subspace: $F = \begin{bmatrix} f_1^T \\ f_2^T \\ \vdots \\ f_r^T \end{bmatrix} \in \mathbb{R}^{r \times m}$, $r \ll m$

- ▶ Face coordinates: $W = \begin{bmatrix} w_1^T \\ w_2^T \\ \vdots \\ w_n^T \end{bmatrix} \in \mathbb{R}^{n \times r}$

- ▶ Reconstruction: $\hat{X} = WF + \mu$, where $\mu \in \mathbb{R}^{n \times m}$ replicates the mean face vector at each row.

Optimization Formulation of Face Subspace Learning

- ▶ Frobenius norm of a matrix: $\|X\|_F = \sqrt{\sum_{ij} x_{ij}^2}$

- ▶ We use $\|\cdot\|_F$ to measure $X \approx \hat{X}$:

$$\|X - \hat{X}\|_F^2 = \|X - (WF + \mu)\|_F^2 = \|(X - \mu) - WF\|_F^2$$

- ▶ Let $D = X - \mu$, we have an optimization problem:

$$\underset{W \in \mathbb{R}^{n \times r}, F \in \mathbb{R}^{r \times m}}{\text{minimize}} \quad \|D - WF\|_F^2$$

- ▶ We do not know how to obtain the global minimum of the above problem (non-convex); however, we can solve the following equivalent problem:

$$\begin{aligned} & \underset{\hat{D}}{\text{minimize}} \quad \|D - \hat{D}\|_F^2 \\ & \text{s.t.} \quad \text{rank}(\hat{D}) \leq r \end{aligned}$$

Low-rank Approximation Theorem

$$\begin{aligned} & \underset{\hat{D}}{\text{minimize}} && \|D - \hat{D}\|_F^2 \\ & \text{s.t.} && \text{rank}(\hat{D}) \leq r \end{aligned}$$

- ▶ Let $D = U\Sigma V^T \in \mathbb{R}^{n \times m}$, $n \geq m$ be the singular value decomposition of D and partition U , Σ , and V as follows:

$$U = [U_1 \ U_2], \Sigma = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix}, V = [V_1, V_2],$$

where $U_1 \in \mathbb{R}^{m \times r}$, $\Sigma_1 \in \mathbb{R}^{r \times r}$, and $V_1 \in \mathbb{R}^{n \times r}$.

- ▶ Then the solution is

$$\hat{D} = U_1 \Sigma_1 V_1^T$$

Principal Component Analysis

SVD for the eigenface problem

$$\text{Let } W = U_1 \Sigma_1 \text{ and } F = V_1^T$$

This is a general dimension reduction technique!

Principal Component Analysis

Goal: Find r -dim projection that best preserves data

1. Compute mean vector μ
2. Subtract μ from data matrix
3. SVD and select top r right-singular vectors
4. Project points onto the subspace spanned by them

Reconstruction Results for Faces

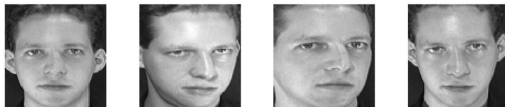
$P = 4$



$P = 200$



$P = 400$



- after computing eigenfaces using 400 face images from ORL face database

Homework: PCA for 2D plane detection in 3D point cloud

Congrats!

You've learned all the (difficult) math tools that this course requires!