Least squares

CE40282-1: Linear Algebra
Hamid R. Rabiee and Maryam Ramezani
Sharif University of Technology

Least squares problem

- given $A \in \mathbf{R}^{m \times n}$ and $b \in \mathbf{R}^m$, find vector $x \in \mathbf{R}^n$ that minimizes

$$\|Ax - b\|^2 = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} A_{ij} x_j - b_i \right)^2$$

- "least squares" because we minimize a sum of squares of affine functions:

$$\|Ax - b\|^2 = \sum_{i=1}^{m} r_i(x)^2, \qquad r_i(x) = \sum_{j=1}^{n} A_{ij} x_j - b_i$$

- the problem is also called the linear least squares problem

Least squares and linear equations

$$\text{minimize} \quad \|Ax - b\|^2$$

- solution of the least squares problem: any $\hat{x}$ that satisfies

$$\|A\hat{x} - b\| \leq \|Ax - b\| \quad \text{for all } x$$

- Note:

$\hat{r} = A\hat{x} - b$ is the *residual vector*

if $\hat{r} = 0$, then $\hat{x}$ solves the linear equation $Ax = b$

if $\hat{r} \neq 0$, then $\hat{x}$ is a *least squares approximate solution* of the equation

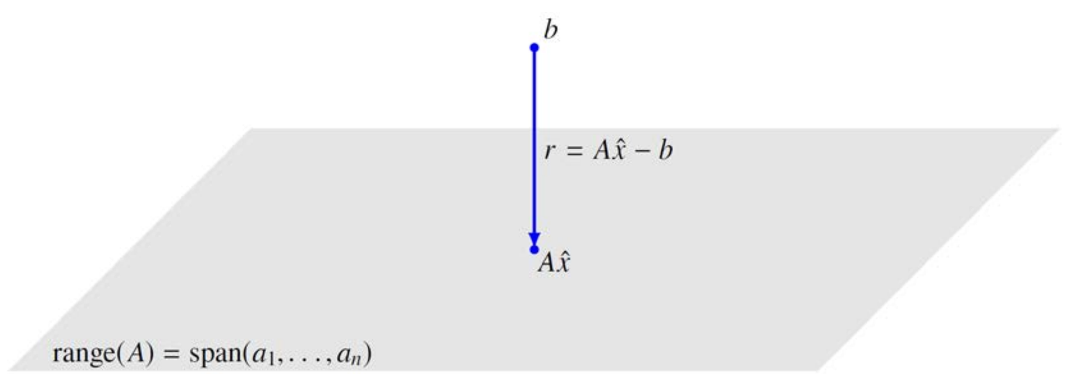
in most least squares applications, $m > n$ and $Ax = b$ has no solution

Column interpretation

- least squares problem in terms of columns $a_1, a_2, \ldots, a_n$ of A:

$$\text{minimize} \quad \|Ax - b\|^2 = \|\sum_{j=1}^{n} a_j x_j - b\|^2$$

- The solution is closest to b among all linear combinations of columns of A $A\hat{x} = \hat{x}_1 a_1 + \cdots + \hat{x}_n a_n$
- $A\hat{x}$ is the vector in $\text{range}(A) = \text{span}(a_1, a_2, \ldots, a_n)$ closest to b
- geometric intuition suggests that $\hat{r} = A\hat{x} - b$ is orthogonal to $\text{range}(A)$



Row interpretation

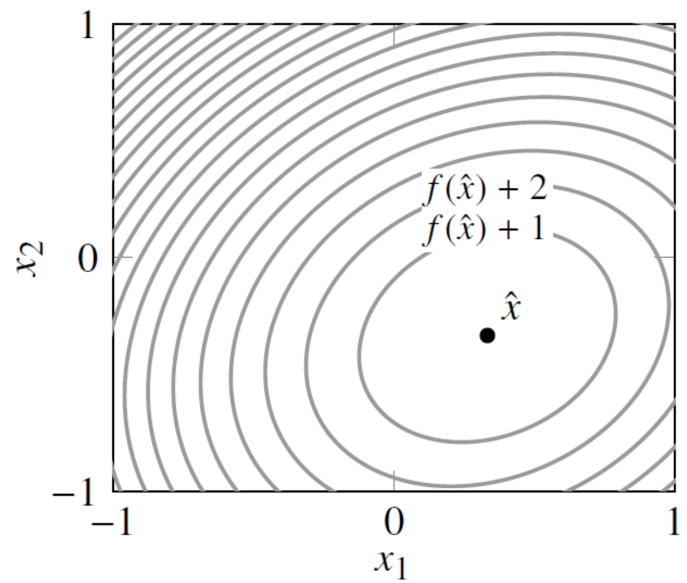
- suppose $\tilde{a}_1^T, \ldots, \tilde{a}_m^T$ are rows of A
- residual components are $r_i = \tilde{a}_i^T x - b_i$
- least squares objective is

$$\|Ax - b\|^2 = (\tilde{a}_1^T x - b_1)^2 + \cdots + (\tilde{a}_m^T x - b_m)^2$$

  the sum of squares of the residuals
- so least squares minimizes sum of squares of residuals
  - solving $Ax = b$ is making all residuals zero
  - least squares attempts to make them all small

Example

$$A = \begin{bmatrix} 2 & 0 \\ -1 & 1 \\ 0 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$



- $Ax = b$ has no solution
- least squares problem is to choose x to minimize

$$\|Ax - b\|^2 = (2x_1 - 1)^2 + (-x_1 + x_2)^2 + (2x_2 + 1)^2$$

- least squares approximate solution is $\hat{x} = (1/3, -1/3)$ (say, via calculus)
- $\|A\hat{x} - b\|^2 = 2/3$ is smallest posible value of $\|Ax - b\|^2$
- $A\hat{x} = (2/3, -2/3, -2/3)$ is linear combination of columns of A closest to b

Solution of a least squares problem

- A has linearly independent columns, then below vector is the unique solution of the least squares problem

$$\text{minimize} \quad ||Ax - b||^2$$

$$\begin{aligned} \hat{x} &= (A^T A)^{-1} A^T b \\ &= A^\dagger b \end{aligned}$$

pseudo-inverse of a left-invertible matrix

  - Proof?

Derivation from calculus

- $f(x) = \|Ax - b\|^2 = \sum_{i=1}^{m} \left(\sum_{j=1}^{n} A_{ij} x_j - b_i \right)^2$

- partial derivative of f with respect to x_k

$$\frac{\partial f}{\partial x_k}(x) = 2 \sum_{i=1}^{m} A_{ik} \left(\sum_{j=1}^{n} A_{ij} x_j - b_i \right) = 2(A^T(Ax - b))_k$$

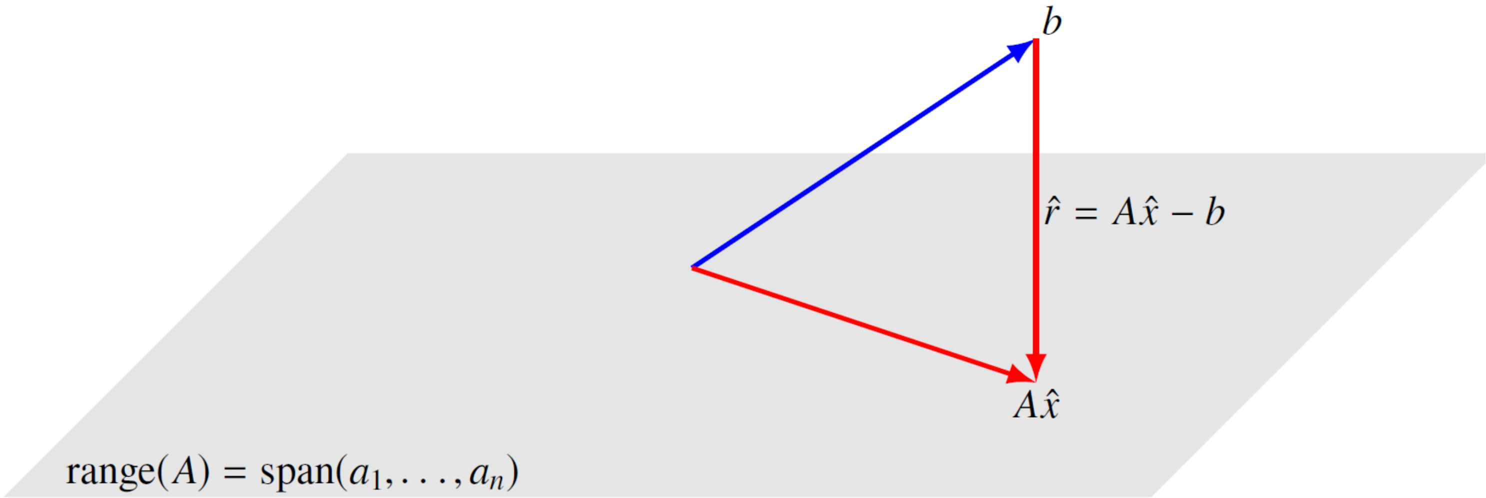
- gradient of f is

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}(x), \frac{\partial f}{\partial x_2}(x), \ldots, \frac{\partial f}{\partial x_n}(x) \right) = 2A^T(Ax - b)$$

- minimizer $\hat{x}$ of $f(x)$ satisfies $\nabla f(\hat{x}) = 2A^T(A\hat{x} - b) = 0 \rightarrow \hat{x} = (A^T A)^{-1} A^T b$

Geometric interpretation

- residual vector $\hat{r} = A\hat{x} - b$ satisfies $A^T\hat{r} = A^T(A\hat{x} - b) = 0$



residual vector $\hat{r}$ is orthogonal to every column of A; hence, to $\mathrm{range}(A)$

projection on $\mathrm{range}(A)$ is a matrix-vector multiplication with the matrix

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

Conclusion

Let A be an $m \times n$ matrix. The following statements are logically equivalent:

a. The equation $A\mathbf{x} = \mathbf{b}$ has a unique least-squares solution for each $\mathbf{b}$ in $\mathbb{R}^m$

b. The columns of A are linearly independent.

c. The matrix A^TA is invertible.

When these statements are true, the least-squares solution $\hat{\mathbf{x}}$ is given by

$$\hat{\mathbf{x}} = (A^TA)^{-1}A^T\mathbf{b}$$

When a least-squares solution $\hat{\mathbf{x}}$ is used to produce $A\hat{\mathbf{x}}$ as an approximation to $\mathbf{b}$, the distance from $\mathbf{b}$ to $A\hat{\mathbf{x}}$ is called the **least-squares error** of this approximation.

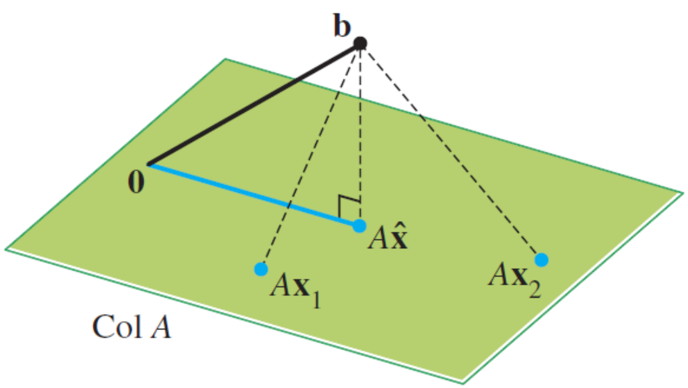
Solving least squares problems (Method 1)

- Normal equations of the least squares problem $A^T A x = A^T b$
  - Coefficient matrix $A^T A$ is the ......
  - Equivalent to $\nabla f(x) = 0$ where $f(x) =$
  - All solutions of the least squares problem satisfy the normal equations

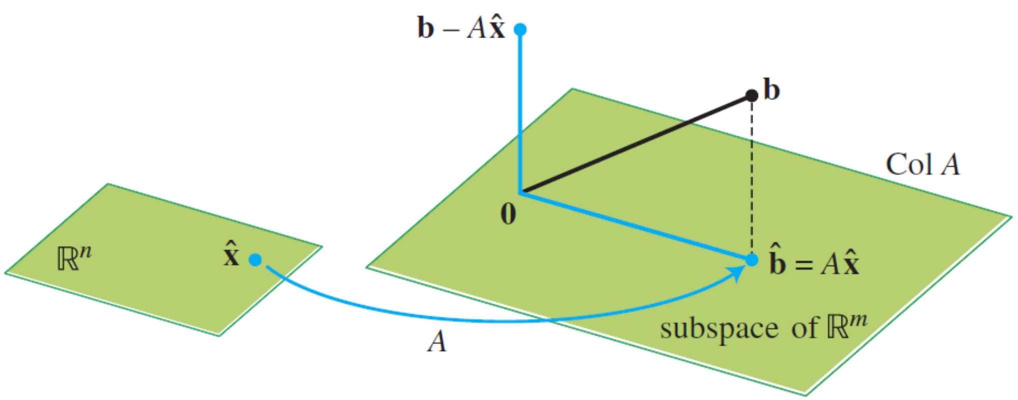
$$\hat{x} = (A^T A)^{-1} A^T b$$

Normal equation

The set of least-squares solutions of $A\mathbf{x} = \mathbf{b}$ coincides with the nonempty set of solutions of the normal equations $A^TA\mathbf{x} = A^T\mathbf{b}$.



The vector $\mathbf{b}$ is closer to $A\hat{\mathbf{x}}$ than to $A\mathbf{x}$ for other $\mathbf{x}$.



The least-squares solution $\hat{\mathbf{x}}$ is in $\mathbb{R}^n$.

Solving least squares problems (Method 2): QR factorization

- Rewrite least squares solution using QR factorization $A = QR$

- Complexity: $2mn^2$

Algorithm: Least squares via QR factorization

Input: $A : m \times n$ left-invertible
Input: $b : m \times 1$
Output: $x_{LS} : n \times 1$
Find QR factorization $A = QR$
Compute $Q^T b$
Solve $Rx_{LS} = Q^T b$ using back substitution

- Identical to algorithm for solving Ax = b for square invertible A, but when A is tall, gives least squares approximate solution

Solving least squares problems

- Example

  a 3×2 matrix with "almost linearly dependent" columns

$$A = \begin{bmatrix} 1 & -1 \\ 0 & 10^{-5} \\ 0 & 0 \end{bmatrix}, \qquad b = \begin{bmatrix} 0 \\ 10^{-5} \\ 1 \end{bmatrix},$$

  round intermediate results to 8 significant decimal digits

- Solve using both methods
  - Which one is more stable? Why?

Review: Linear-in-parameters model

- we choose the model $\hat{f}(x)$ from a family of models

$$\hat{f}(x) = \theta_1 f_1(x) + \theta_2 f_2(x) + \cdots + \theta_p f_p(x)$$

model parameters

scalar valued basis functions (chosen by us)

Least squares regression

- Remember the regression model (affine function):

$$\hat{f}(x) = x^T\beta + v$$

- the prediction error for example i is:

$$\begin{aligned} r^{(i)} &= y^{(i)} - \hat{f}(x^{(i)}) \\ &= y^{(i)} - (x^{(i)})^T\beta - v \end{aligned}$$

- the MSE is:

$$\frac{1}{N}\sum_{i=1}^{N}(r^{(i)})^2 = \frac{1}{N}\sum_{i=1}^{N}\left(y^{(i)} - (x^{(i)})^T\beta - v\right)^2$$

Least squares regression

choose the model parameters v, β that minimize the MSE

$$\frac{1}{N}\sum_{i=1}^{N}\left(v + (x^{(i)})^T\beta - y^{(i)}\right)^2$$

this is a least squares problem: minimize $\|A\theta - y^{\mathrm{d}}\|^2$ with

$$A = \begin{bmatrix} 1 & (x^{(1)})^T \\ 1 & (x^{(2)})^T \\ \vdots & \vdots \\ 1 & (x^{(N)})^T \end{bmatrix}, \qquad \theta = \begin{bmatrix} v \\ \beta \end{bmatrix}, \qquad y^{\mathrm{d}} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$$

we write the solution as $\hat{\theta} = (\hat{v}, \hat{\beta})$

Least squares regression

- Example

$$\hat{f}(x) = \theta_1 + \theta_2 x + \theta_3 x^2 + \cdots + \theta_p x^{p-1}$$

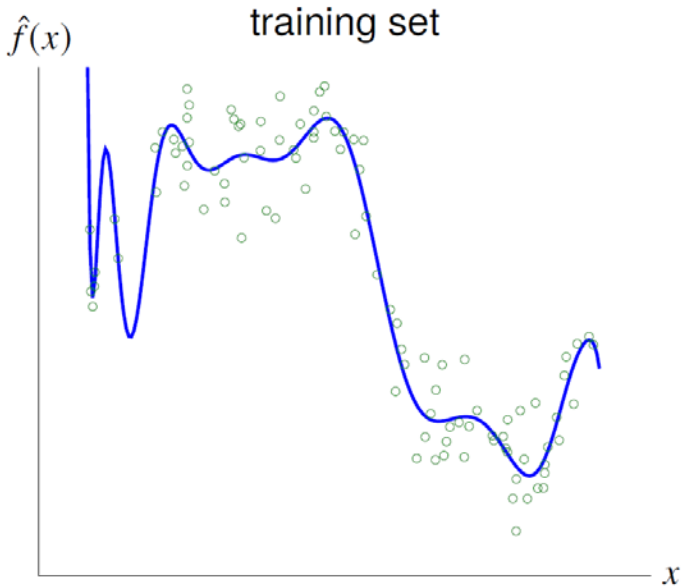
  - a linear-in-parameters model with basis functions ………
  - least squares model fitting in matrix notation?

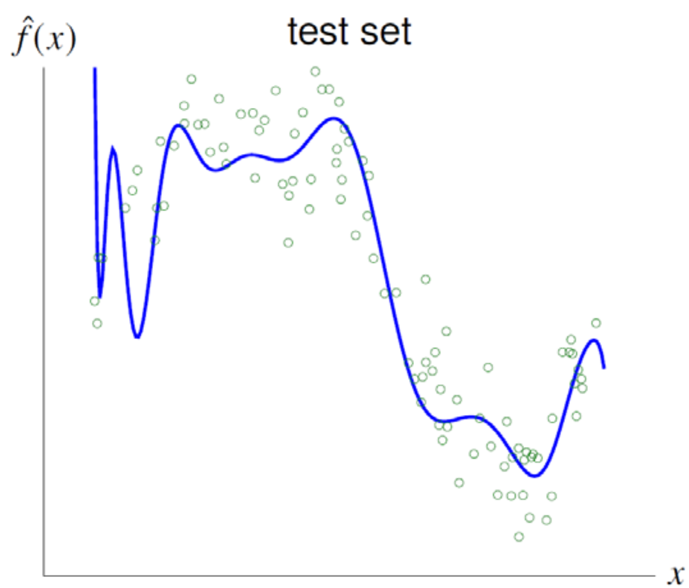
Generalization and validation

- Generalization ability: ability of model to predict outcomes for new, unseen data
- Model validation: to assess generalization ability,
  - divide data in two sets: training set and test (or validation) set
  - use training set to fit model
  - use test set to get an idea of generalization ability
  - this is also called out-of-sample validation
- Over-fit model
  - model with low prediction error on training set, bad generalization ability
  - prediction error on training set is much smaller than on test set

Over-fitting

polynomial of degree 20 on training and test set





over-fitting is evident at the left end of the interval

Cross-validation

- an extension of out-of-sample validation
  - divide data in K sets (*folds*); typical values are $K = 5$, $K = 10$
  - for $i = 1$ to K, fit model i using fold i as test set and other data as training set
  - compare parameters and train/test RMS errors for the K models
- Remember the house price problem (data set of N= 774 house sales)

House price model with 5 folds (155 or 154 examples each)

	Model parameters								RMS error	
Fold	v	β_1	β_2	β_3	β_4	β_5	β_6	β_7	Train	Test
1	122.5	166.9	−39.3	−16.3	−24.0	−100.4	−106.7	−26.0	67.3	72.8
2	101.0	186.7	−55.8	−18.7	−14.8	−99.1	−109.6	−17.9	67.8	70.8
3	133.6	167.2	−23.6	−18.7	−14.7	−109.3	−114.4	−28.5	69.7	63.8
4	108.4	171.2	−41.3	−15.4	−17.7	−94.2	−103.6	−29.8	65.6	78.9
5	114.5	185.7	−52.7	−20.9	−23.3	−102.8	−110.5	−23.4	70.7	58.3

Boolean (two-way) classification

- Problem:
  - a data fitting problem where the outcome y can take two values $+1$, -1

    values of y represent two categories (true/false, spam/not spam, . . .)

    model $\hat{y} = \hat{f}(x)$ is called a *Boolean classifier*

- Least squares classifier
  - use least squares to fit model $\tilde{f}(x)$ to training set $(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})$
  - $\tilde{f}(x)$ can be a regression model $\tilde{f}(x) = x^T\beta + v$ or linear in parameters

$$\tilde{f}(x) = \theta_1 f_1(x) + \cdots + \theta_p f_p(x)$$

  - take sign of $\tilde{f}(x)$ to get a Boolean classifier

$$\hat{f}(x) = \mathrm{sign}(\tilde{f}(x)) = \begin{cases} +1 & \text{if } \tilde{f}(x) \geq 0 \\ -1 & \text{if } \tilde{f}(x) < 0 \end{cases}$$

Multi-class classification

- Problem:
  - a data fitting problem where the outcome y can takes values $1, \ldots, K$
  - values of y represent K labels or categories
  - multi-class classifier $\hat{y} = \hat{f}(x)$ maps x to an element of $\{1, 2, \ldots, K\}$
- Least squares multi-class classifier
  - for $k = 1, \ldots, K$, compute Boolean classifier to distinguish class k from not k

$$\hat{f}_k(x) = \operatorname{sign}(\tilde{f}_k(x))$$

  - define multi-class classifier as

$$\hat{f}(x) = \underset{k=1,\ldots,K}{\operatorname{argmax}} \tilde{f}_k(x)$$

Multi-objective least squares

- we have several objectives

$$J_1 = \|A_1x - b_1\|^2, \qquad \ldots, \qquad J_k = \|A_kx - b_k\|^2$$

  - A_i is an $m_i \times n$ matrix, b_i is an m_i-vector
  - we seek *one* x that makes all k objectives small
  - usually there is a trade-off: no single x minimizes all objectives simultaneously

- **Weighted least squares formulation**: find x that minimizes

$$\lambda_1\|A_1x - b_1\|^2 + \cdots + \lambda_k\|A_kx - b_k\|^2$$

  - coefficients $\lambda_1, \ldots, \lambda_k$ are positive weights
  - weights λ_i express relative importance of different objectives
  - without loss of generality, we can choose $\lambda_1 = 1$

Solution of weighted least squares

- weighted least squares is equivalent to a standard least squares problem

$$\text{minimize} \quad \left\| \begin{bmatrix} \sqrt{\lambda_1}A_1 \\ \sqrt{\lambda_2}A_2 \\ \vdots \\ \sqrt{\lambda_k}A_k \end{bmatrix} x - \begin{bmatrix} \sqrt{\lambda_1}b_1 \\ \sqrt{\lambda_2}b_2 \\ \vdots \\ \sqrt{\lambda_k}b_k \end{bmatrix} \right\|^2$$

- Solution is unique if the *stacked matrix* has linearly independent columns
- Each matrix A_i may have linearly dependent columns (or be a wide matrix)
- if the stacked matrix has linearly independent columns, the solution is

$$\hat{x} = \left(\lambda_1 A_1^T A_1 + \cdots + \lambda_k A_k^T A_k\right)^{-1} \left(\lambda_1 A_1^T b_1 + \cdots + \lambda_k A_k^T b_k\right)$$

Lagrange multiplier

- Example

$$f(x) = \min(x_1 x_2)$$
$$g(x) = 1 - \mathrm{x}_1 - \mathrm{x}_2$$
$$g(x) = 0$$

$$L(x, \lambda) = f(x) + \lambda g(x)$$

$$\nabla f(x)$$

Constrained Least Square

- $$\begin{cases} \min\limits_{x} & \|Ax-b\|^2 \qquad A: m \times n \\ \text{s.t.} & Cx = d \qquad C: p \times n \end{cases}$$

$$L(x,\lambda) = \|Ax-b\|^2 + \lambda^T(Cx-d)$$

- $$\begin{cases} \nabla_x L = 2A^TAx - 2A^Tb + C^T\lambda = 0 \\ \nabla_\lambda L = Cx - d = 0 \end{cases} \Rightarrow \begin{bmatrix} 2A^TA & C^T \\ C & 0 \end{bmatrix} \begin{bmatrix} x^* \\ \lambda^* \end{bmatrix} = \begin{bmatrix} 2A^Tb \\ d \end{bmatrix}$$

- #equations: n+p #Unkowns: n+p
- KKT equations
- Least Square problem is a KKT problem with $A = I, b = 0$

Regularized data fitting

- consider linear-in-parameters model

$$\hat{f}(x) = \theta_1 f_1(x) + \cdots + \theta_p f_p(x)$$

  we assume $f_1(x)$ is the constant function 1
- keeping $\theta_2, \ldots, \theta_p$ small helps avoid over-fitting

$$J_1(\theta) = \sum_{k=1}^{N} (\hat{f}(x^{(k)}) - y^{(k)})^2, \qquad J_2(\theta) = \sum_{j=2}^{p} \theta_j^2$$

$$\text{minimize} \quad J_1(\theta) + \lambda J_2(\theta) = \sum_{k=1}^{N} (\hat{f}(x^{(k)}) - y^{(k)})^2 + \lambda \sum_{j=2}^{p} \theta_j^2$$

Solution for Weighted least squares

$$\text{minimize} \quad J_1(\theta) + \lambda J_2(\theta) = \sum_{k=1}^{N} (\hat{f}(x^{(k)}) - y^{(k)})^2 + \lambda \sum_{j=2}^{p} \theta_j^2$$

- λ is positive *regularization parameter*
- equivalent to least squares problem: minimize

$$\left\| \begin{bmatrix} A_1 \\ \sqrt{\lambda} A_2 \end{bmatrix} \theta - \begin{bmatrix} y^{\mathrm{d}} \\ 0 \end{bmatrix} \right\|^2$$

with $y^{\mathrm{d}} = (y^{(1)}, \ldots, y^{(N)})$,

$$A_1 = \begin{bmatrix} 1 & f_2(x^{(1)}) & \cdots & f_p(x^{(1)}) \\ 1 & f_2(x^{(2)}) & \cdots & f_p(x^{(2)}) \\ \vdots & \vdots & & \vdots \\ 1 & f_2(x^{(N)}) & \cdots & f_p(x^{(N)}) \end{bmatrix}, \qquad A_2 = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix}$$

- stacked matrix has linearly independent columns (for positive λ)
- value of λ can be chosen by out-of-sample validation or cross-validation

Nonlinear least squares

- find $\hat{x}$ that minimizes

$$\|f(x)\|^2 = f_1(x)^2 + \cdots + f_m(x)^2$$

- optimality condition: $\nabla \|f(\hat{x})\|^2 = 0$

  any optimal point satisfies this

  points can satisfy this and not be optimal

  can be expressed as $2Df(\hat{x})^T f(\hat{x}) = 0$

  $Df(\hat{x})$ is the $m \times n$ derivative or Jacobian matrix,

$$Df(\hat{x})_{ij} = \frac{\partial f_i}{\partial x_j}(\hat{x}), \quad i = 1, \ldots, m, \quad j = 1, \ldots, n$$

  optimality condition reduces to normal equations when f is affine

References

- Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares, Stephen Boyd Lieven Vandenberghe
- Linear Algebra and Its Applications, David C. Lay