EECS 275 Matrix Computation

Ming-Hsuan Yang

Electrical Engineering and Computer Science
University of California at Merced
Merced, CA 95344
http://faculty.ucmerced.edu/mhyang

Lecture 9
Overview

- Least squares minimization
- Regression
- Regularization
Reading

- Chapter 11 of *Numerical Linear Algebra* by Llyod Trefethen and David Bau
- Chapter 5 of *Matrix Computations* by Gene Golub and Charles Van Loan
- Chapter 4 of *Matrix Analysis and Applied Linear Algebra* by Carl Meyer
- Chapter 11 and Chapter 15 of *Matrix Algebra From a Statistician’s Perspective* by David Harville
Matrix differentiation

- First order differentiation of linear form:
  \[ a^\top x = x^\top a = \sum_i a_i x_i \]

  \[
  \frac{\partial x^\top a}{\partial x} = \frac{\partial a^\top x}{\partial x} = \begin{bmatrix}
    \frac{\partial x^\top a}{\partial x_1} \\
    \vdots \\
    \frac{\partial x^\top a}{\partial x_n}
  \end{bmatrix} = a
  \]

  \[
  \frac{\partial x_i}{\partial x_j} = \begin{cases} 
    1 & \text{if } i = j \\
    0 & \text{if } i \neq j 
  \end{cases}
  \]

  \[
  \frac{\partial x^\top a}{\partial x_j} = \frac{\partial (\sum_i a_i x_i)}{\partial x_j} = \sum_i a_i \frac{\partial x_i}{\partial x_j} = a_j
  \]

- Likewise

  \[
  \frac{\partial x^\top a}{\partial x^\top} = a^\top
  \]

  \[
  \frac{\partial (Ax)}{\partial x^\top} = A \quad \frac{\partial (Ax)^\top}{\partial x} = A^\top
  \]
Matrix differentiation (cont’d)

- First order differentiation of quadratic form:

\[ \mathbf{x}^\top \mathbf{A} \mathbf{x} = \sum_{i,k} a_{ik} x_i x_k \]

\[ \frac{\partial \mathbf{x}^\top \mathbf{A} \mathbf{x}}{\partial \mathbf{x}} = (\mathbf{A} + \mathbf{A}^\top) \mathbf{x} \]

\[ \frac{\partial (x_i x_k)}{\partial x_j} = \begin{cases} 
2x_j & \text{if } i = k = j \\
x_i & \text{if } k = j, i \neq j \\
x_k & \text{if } i = j, k \neq j \\
0 & \text{otherwise}
\end{cases} \]

\[ \frac{\partial \mathbf{x}^\top \mathbf{A} \mathbf{x}}{\partial x_j} = \frac{\partial \sum_{i,k} a_{ik} x_i x_k}{\partial x_j} = \frac{\partial (a_{jj} x_j^2 + \sum_{i \neq j} a_{ij} x_i x_j + \sum_{k \neq j} a_{jk} x_j x_k + \sum_{i \neq j, k \neq j} a_{ik} x_i x_k)}{\partial x_j} \]

\[ = a_{jj} \frac{\partial x_j^2}{\partial x_j} + \sum_{i \neq j} a_{ij} \frac{\partial (x_i x_j)}{\partial x_j} + \sum_{k \neq j} a_{jk} \frac{\partial (x_j x_k)}{\partial x_j} + \sum_{i \neq j, k \neq j} a_{ik} \frac{\partial (x_i x_k)}{\partial x_j} \]

\[ = 2a_{jj} x_j + \sum_{i \neq j} a_{ij} x_i + \sum_{k \neq j} a_{jk} x_k + 0 \]

\[ = \sum_i a_{ij} x_i + \sum_k a_{jk} x_k \]
Matrix differentiation (cont’d)

- First order differentiation of quadratic form:
  \[
  \frac{\partial x^\top A x}{\partial x} = (A + A^\top)x
  \]

- Let \( W \) be a symmetric matrix, it can be easily shown that
  \[
  \frac{\partial}{\partial s} (x - As)^\top W (x - As) = -2A^\top W (x - As)
  \]
  \[
  \frac{\partial}{\partial s} (x - s)^\top W (x - s) = -2W (x - s)
  \]
  \[
  \frac{\partial}{\partial x} (x - As)^\top W (x - As) = 2W (x - As)
  \]
Matrix differentiation (cont’d)

- Second order derivative of quadratic form:

\[
\frac{\partial^2 (x^\top Ax)}{\partial x_s \partial x_j} = \sum_i a_{ij} \frac{\partial x_j}{\partial x_s} + \sum_k a_{jk} \frac{\partial x_k}{\partial x_s} = a_{sj} + a_{js}
\]

\[
\frac{\partial^2 (x^\top Ax)}{\partial x \partial x^\top} = A + A^\top
\]

- Recall

\[
f(x) \approx f(a) + J(a)(x - a) + \frac{1}{2}(x - a)^\top H(a)(x - a)
\]

- See *The Matrix Cookbook* by Kaare Petersen and Michael Pedersen (http://matrixcookbook.com) for details
Overdetermined linear equations

- Consider $y = Ax$ where $A \in \mathbb{R}^{m \times n}$ is skinny, i.e., $m > n$
- One can approximately solve $y \approx Ax$, and define residual or error $r = Ax - y$
- Find $x = x_{ls}$ that minimizes $\|r\|$
- $x_{ls}$ is the least squares solution
- Geometric interpretation: $Ax_{ls}$ is the point in $\text{ran}(A)$ that is closest to $y$, i.e., $Ax_{ls}$ is the projection of $y$ onto $\text{ran}(A)$
Least squares minimization

- Minimize norm of residual squared
  \[ \mathbf{r} = A\mathbf{x} - \mathbf{y} \]
  \[ \|\mathbf{r}\|^2 = \mathbf{x}^\top A^\top A\mathbf{x} - 2\mathbf{y}^\top A\mathbf{x} + \mathbf{y}^\top \mathbf{y} \]
- Set gradient with respect to \( \mathbf{x} \) to zero
  \[ \nabla_x \|\mathbf{r}\|^2 = 2A^\top A\mathbf{x} - 2A^\top \mathbf{y} = 0 \Rightarrow A^\top A\mathbf{x} = A^\top \mathbf{y} \]
  (also known as normal equations)
- Assume \( A^\top A \) is invertible, we have
  \[ \mathbf{x}_{ls} = (A^\top A)^{-1}A^\top \mathbf{y} \]
  \[ A\mathbf{x}_{ls} = A(A^\top A)^{-1}A^\top \mathbf{y} \]
Least squares minimization

\[ y = Ax \Rightarrow x_{ls} = (A^\top A)^{-1} A^\top y \]

- \( x_{ls} \) is linear function of \( y \)
- \( x_{ls} = A^{-1}y \) if \( A \) is square
- \( x_{ls} \) solves \( y = Ax_{ls} \) if \( y \in \text{ran}(A) \)
- \( A^\dagger = (A^\top A)^{-1} A^\top \) is called pseudo inverse or Moore-Penrose inverse
- \( A^\dagger \) is a left inverse of (full rank, skinny) \( A \):
  \[ A^\dagger A = (A^\top A)^{-1} A^\top A = I \]
- \( A(A^\top A)^{-1} A^\top \) is the projection matrix
Orthogonality principle

- Optimal residual

$$r = Ax_{ls} - y = (A(A^\top A)^{-1}A^\top - I)y$$

which is orthogonal to $\text{ran}(A)$:

$$\langle r, Az \rangle = y^\top (A(A^\top A)^{-1}A^\top - I)^\top Az = 0$$

for all $z \in \mathbb{R}^n$

- Since $r = Ax_{ls} - y \perp A(x - x_{ls})$ for any $x \in \text{ran}(A)$, we have

$$\|Ax - y\|^2 = \|(Ax_{ls} - y) + A(x - x_{ls})\|^2 = \|Ax_{ls} - y\|^2 + \|A(x - x_{ls})\|^2$$

which means for $x \neq x_{ls}$, $\|Ax - y\| > \|Ax_{ls} - y\|$.
Least squares minimization and orthogonal projection

- Recall if $u \in \mathbb{R}^m$, then $P = \frac{uu^T}{u^Tu}$ is an orthogonal projection.
- Given a point $x = x_\parallel + x_\perp$, its projection is

$$P_ux = uu^Tx_\parallel + uu^Tx_\perp = x_\parallel$$

- Generalize to orthogonal projections on a subspace spanned by a set of orthonormal basis $A = [u_1, \ldots, u_r]$:

$$P_A = AA^T$$

- In general, we need a normalization term for orthogonal projection if $u_1, \ldots, u_r$ is not orthonormal basis,

$$P_A = A(A^TA)^{-1}A^T$$

- Given $A = U\Sigma V^\top$, it follows that $P_A = UU^\top$ by least squares minimization.
Least squares estimation

- Numerous applications in inversion, estimation and reconstruction problems have the form

\[ y = Ax + v \]

- \( x \) is what we want to estimate or reconstruct
- \( y \) is our sensor measurements
- \( v \) is unknown noise or measurement error
- \( i \)-th row of \( A \) characterizes \( i \)-th sensor

- Least squares estimation: choose \( \hat{x} \) that minimizes \( \| A\hat{x} - y \| \), i.e., deviation between
  - what we actually observe \( y \), and
  - what we would observe if \( x = \hat{x} \), and there were no noise (\( v = 0 \))

least squares estimate is

\[ \hat{x} = (A^\top A)^{-1}A^\top y \]
Best linear unbiased estimator (BLUE)

- Linear estimator with noise: \( y = Ax + v \) with \( A \) is a full rank and skinny
- A linear estimator of form \( \hat{x} = By \), is unbiased if \( \hat{x} = x \) whenever \( v = 0 \) (no estimator error when \( v = 0 \))
- Equivalent to \( BA = I \), i.e., \( B \) is the left inverse of \( A \)
- Estimator error of unbiased linear estimator is
  \[
  x - \hat{x} = x - B(Ax + v) = -Bv
  \]
- It follows that \( A^\dagger = (A^\top A)^{-1}A^\top \) is the smallest left inverse of \( A \) such that for any \( B \) with \( BA = I \), we have
  \[
  \sum_{i,j} B_{ij}^2 \geq \sum_{i,j} A_{ij}^\dagger^2
  \]
  i.e., least squares provides the best linear unbiased estimator (BLUE)
Pseudo inverse via regularization

- For $\mu > 0$, let $x_\mu$ be unique minimizer of

$$
\|Ax - y\|^2 + \mu\|x\|^2 = \left\| \begin{bmatrix} A \\ \sqrt{\mu}I \end{bmatrix} x - \begin{bmatrix} y \\ 0 \end{bmatrix} \right\|^2 = \left\| \tilde{A}x - \tilde{y} \right\|^2
$$

thus

$$
x_\mu = (\tilde{A}^\top \tilde{A})^{-1} \tilde{A}^\top \tilde{y} = (A^\top A + \mu I)^{-1} A^\top y
$$

is called regularized least squares solution for $Ax \approx y$

- Also called Tikhonov (Tychonov) regularization (ridge regression in statistics)

- As $A^\top A + \mu I > 0$ and so is invertible, then we have

$$
\lim_{\mu \to 0} x_\mu = A^\dagger y
$$

and

$$
\lim_{\mu \to 0} (A^\top A + \mu I)^{-1} A^\top = A^\dagger
$$
Minimizing weighted-sum objective

- Two (or more) objectives:
  - want $J_1 = \|Ax - y\|^2$ small
  - and also $J_2 = \|Fx - g\|^2$ small

- Consider minimize a weighted-sum objective

$$\|Ax - y\|^2 + \mu \|Fx - g\|^2 = \left\| \begin{bmatrix} A \\ \sqrt{\mu}F \end{bmatrix} x - \begin{bmatrix} y \\ \sqrt{\mu}g \end{bmatrix} \right\|^2 = \left\| \tilde{A}x - \tilde{y} \right\|^2$$

- Thus, the least squares solution is

$$x = (\tilde{A}^\top \tilde{A})^{-1} \tilde{A}^\top \tilde{y} = (A^\top A + \mu F^\top F)^{-1}(A^\top y + \mu F^\top g)$$

- Widely used function approximation, regression, optimization, image processing, computer vision, control, machine learning, graph theory, etc.
Least squares data fitting

- Linear regression: Model one scalar $y$ in terms of linear combination of $t_1, \ldots, t_n$

$$y = \alpha_0 + \alpha_1 t_1 + \cdots + \alpha_n t_n = \sum_{j=1}^{n+1} \alpha_j t_j$$

where $\alpha_j$ are unknown parameters or coefficients

- For a set of $m$ data points, $\{(t_i, y_i)\}$, $t \in \mathbb{R}^n$, want to minimize

$$\sum_{i=1}^{m} (y_i - \sum_{j=1}^{n+1} t_{ij} \alpha_j)^2$$
Least squares data fitting

- For a set of training data, \{ (t_i, y_i) \}, we form \( y \) and \( A \).

- In matrix form, denote \( A \) by \( m \times (n+1) \) matrix with each row an input vector, and \( x \in \mathbb{R}^{n+1} \),

\[
\begin{align*}
\mathbf{y} &= \mathbf{A} \mathbf{x} \\
\mathbf{y} &= \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \\
\mathbf{A} &= \begin{bmatrix} 1 & t_{11} & t_{12} & \cdots & t_{1n} \\ 1 & t_{21} & t_{22} & \cdots & t_{2n} \\ 1 & \cdots & \cdots & \cdots & \cdots \\ 1 & t_{m1} & t_{m2} & \cdots & t_{mn} \end{bmatrix} \\
\mathbf{x} &= \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}
\end{align*}
\]

and thus we obtain the coefficients \( \alpha_i \) from \( \mathbf{x} \), where

\[
\mathbf{x} = \mathbf{A}^\dagger \mathbf{y} = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{y}
\]

and

\[
y = \alpha_0 + \alpha_1 t_1 + \cdots + \alpha_n t_n = \sum_{j=1}^{n+1} \alpha_j t_j
\]
Least squares data fitting (cont’d)

- Estimate the relationship of weight loss ($y$) and storage time ($t_1$) and storage temperature ($t_2$) with $y = \alpha_0 + \alpha_1 t_1 + \alpha_2 t_2$

<table>
<thead>
<tr>
<th>Time</th>
<th>1</th>
<th>1</th>
<th>1</th>
<th>2</th>
<th>2</th>
<th>2</th>
<th>3</th>
<th>3</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Temp</td>
<td>-10</td>
<td>-5</td>
<td>0</td>
<td>-10</td>
<td>-5</td>
<td>0</td>
<td>-10</td>
<td>-5</td>
<td>0</td>
</tr>
<tr>
<td>Loss</td>
<td>.15</td>
<td>.18</td>
<td>.20</td>
<td>.17</td>
<td>.19</td>
<td>.22</td>
<td>.20</td>
<td>.23</td>
<td>.25</td>
</tr>
</tbody>
</table>

- Least squares solution is found by

\[
A = \begin{bmatrix}
1 & 1 & -10 \\
1 & 1 & -5 \\
\vdots \\
1 & 3 & 0
\end{bmatrix}, \quad x = \begin{bmatrix}
\alpha_0 \\
\alpha_1 \\
\alpha_2
\end{bmatrix}, \quad y = \begin{bmatrix}
.15 \\
.18 \\
\vdots \\
.25
\end{bmatrix}
\]

- Using MATLAB: $x = A\backslash y = [0.174 \ 0.025 \ 0.005]^T$

\[
y = 0.174 + 0.025 t_1 + 0.005 t_2
\]
Least squares polynomial fitting

- Fit polynomial of degree $n-1$, $n \leq m$
  
  $$y = p(t) = \alpha_0 + \alpha_1 t + \alpha_2 t^2 + \cdots + \alpha_{n-1} t^{n-1}$$

  with data $(y_i, t_i)$

- Basis functions are $f_j(t) = t^{j-1}$, $j = 1, \ldots, n$ (using geometric progression)
  
  - Straight line: $p(t) = \alpha_0 + \alpha_1 t_1$
  - Quadratic: $p(t) = \alpha_0 + \alpha_1 t_1 + \alpha_2 t_2^2$

- Cubic, quartic, and higher polynomials
Least squares polynomial fitting

- Matrix $A$ has form $A_{ij} = t_i^{j-1}$

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} \quad A = \begin{bmatrix} 1 & t_1 & t_1^2 & \cdots & t_1^{n-1} \\ 1 & t_2 & t_2^2 & \cdots & t_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & t_m & t_m^2 & \cdots & t_m^{n-1} \end{bmatrix} \quad x = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_{n-1} \end{bmatrix}$$

(called a Vandermonde matrix)

- See also kernel regression and splines
Least squares polynomial fitting (cont’d)

- Estimate the relationship between range of height of a missile

<table>
<thead>
<tr>
<th>Position</th>
<th>0</th>
<th>250</th>
<th>500</th>
<th>750</th>
<th>1000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Height</td>
<td>0</td>
<td>8</td>
<td>15</td>
<td>19</td>
<td>20</td>
</tr>
</tbody>
</table>

\[
A = \begin{bmatrix}
1 & 0.25 & 0.25^2 \\
1 & 0.5^2 & 0.5^2 \\
1 & 0.75 & 0.75^2 \\
1 & 1 & 1^2
\end{bmatrix} \quad \mathbf{y} = \begin{bmatrix}
0 \\
0.008 \\
0.015 \\
0.019 \\
0.02
\end{bmatrix}
\]

\[
f(t) = -0.0002286 + 0.03983t - 0.01943t^2
\]
Applications

- Thin plate spline: model/morph non-rigid motion