EE 247

Q1.

(a) $\because$ Q Orthogonal $\therefore$ By def: $Q^T Q = QQ^T = I$ $Q \in \mathbb{R}^{n \times n}$ $n \in \mathbb{R}$

(i) $(Q^T)^T Q^T$

$= QQ^T$

$= I$

$\therefore$ Q^T is orthogonal

$(Q^{-1})^T (Q^{-1})$

$= (Q^T)^{-1} Q^{-1}$

From $Q^T Q = I$, We also get $Q^T = Q^{-1}$

$\therefore$ $(Q^T)^{-1} Q^{-1} = (Q^{-1})^{-1} Q^{-1} = QQ^{-1} = I$

$\therefore$ Q^{-1} is orthogonal

(ii)

Let $Q v_i = \lambda_i v_i$ $\langle \lambda_i, v_i \rangle$ is a pair of eigenvalue - eigenvector

$(Q v_i)^T = v_i^T Q^T = (\lambda_i v_i)^T = \lambda_i v_i^T$ $(\lambda_i \in \mathbb{R})$

$\therefore$ $(Q v_i)^T Q v_i = \lambda_i v_i^T \cdot \lambda_i v_i$

$v_i^T Q^T Q v_i = \lambda_i^2 v_i^T v_i$

$\because$ $Q^T Q = I$

$\therefore$ $v_i^T I v_i = \lambda_i^2 v_i^T v_i$

$\because$ $v_i^T I = v_i$

$\therefore$ $v_i^T v_i = \lambda_i^2 v_i^T v_i$

$\therefore$ $\|v_i\|_2^2 = \lambda_i^2 \|v_i\|_2^2$

$\therefore$ $\lambda_i^2 = 1$ $\therefore$ $\lambda_i = 1$ OR $\lambda_i = -1$

(iii) $\det(Q) = \prod_{i=1}^{n} \lambda_i$

$\because \lambda_i = 1$ OR -1

$\therefore \det(Q) = 1$ OR -1

(iv) $Qv_i = \lambda_i v_i$

$\|Qv_i\|_2^2 = v_i^T Q^T Q v_i = v_i^T v_i = \|v_i\|_2^2$

$\therefore$ Length of v_i is preserved

(b) Consider Matrix A diagonalizable

Assume A not Full Rank

$A = TDT^{-1}$

- D Diagonal

$A = U\Sigma V^T = U_1 S V_1^T$; - S Consists of non-zero Singular Values of A

- Columns of U_1, V_1^T As orthonormal basis of Col (A)

- U, V Orthogonal ; Σ diagonal

(ii) According to how SVD is Calculated

σ_i. (Singular values of A)

$= \sqrt{\lambda_i}$ (λ_i is the eigenvalue of A^TA)

$= \sqrt{\lambda_i'}$ (λ_i' is the eigenvalue of AA^T)

(i) Do SVD on A

$A = U\Sigma V^T = U_1 S V_1^T$

$AA^T = U_1 S \underbrace{V_1^T V_1}_{I} S^T U_1^T = U_1 S^2 U_1^T$

Let's do Diagonalization on AA^T

$\because (AA^T)^T = AA^T$, AA^T is Symmetric

$\therefore AA^T = TDT^T$

$\therefore U_1 S^2 U_1^T = TDT^T$

$\therefore$ Again : $S^2 = D \rightarrow \sigma_i^2 = \lambda_i$

$U_1 = T \rightarrow$ Singular vectors (u_1) of A = Eigenvectors of AA^T

Similarly $$A^T A = V_1 S^T U_1^T U_1 S V_1^T$$

$$= V_1 S^2 V_1^T$$

$$= T D T^T$$

$\therefore$ $S^2 = D$ $\rightarrow$ $\sigma_i^2 = \lambda_i$

$V_1 = T$ $\rightarrow$ singular vector v_1 = eigenvector of $A^T A$

(c)

i. $\boxed{\text{False}}$ Counter E.g. $\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$ $\lambda_1 = \lambda_2 = 1$

Should be At Most n distinct eigenvalues

ii. $\boxed{\text{False}}$

$Av_1 = \lambda_1 v_1$

$Av_2 = \lambda_2 v_2$

$A(v_1 + v_2) = \lambda_1 v_1 + \lambda_2 v_2$, Generally $\neq \lambda'(v_1 + v_2)$

iii. $\boxed{\text{True}}$

A : P.S.D Matrix

Let $Av_i = \lambda_i v_i$

$$v_i^T A v_i = v_i^T \lambda_i v_i$$
$$= \lambda_i v_i^T v_i \quad (\lambda_i \in \mathbb{R})$$
$$= \lambda_i \|v_i\|_2^2$$

$\because \|v_i\|_2^2 \geq 0$

A P.S.D. $\therefore v_i^T A v_i \geq 0$

$\therefore \lambda_i$ Must be ≥ 0

iv. $\boxed{\text{True}}$

e.g. $A = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$ Rank $(A) = 2$

$\lambda_1 = \lambda_2 = 1$

\# of distinct $\lambda_i = 1 <$ Rank (A)

v. $\boxed{\text{True}}$

$Av_1 = \lambda v_1$

$Av_2 = \lambda v_2$

$A(v_1 + v_2) = \lambda v_1 + \lambda v_2$
$= \lambda(v_1 + v_2)$

Let $v_1 + v_2 = v' \Rightarrow Av' = \lambda v'$

Q2.

(a)

(i) $P(A \text{ not hit})$

$$= \sum_{n=1}^{\infty} P(A \text{ not hit}, n^{th} \text{ round})$$

Assume duel ends in the nth round and A not hit

$P(A^c, n) \iff \underset{1}{\underline{\text{Both Miss}}} \quad \underset{2}{\underline{\text{Both Miss}}} \quad \underset{3}{\underline{\text{Both Miss}}} \quad \cdots \quad \underset{n-1}{\underline{\text{Both Miss}}} \quad \underset{n}{\underline{\text{B Hit}}}$

$\therefore P(A \text{ not hit}, n^{th} \text{ round})$

$$= [(1-P_A)(1-P_B)]^{n-1} \cdot (1-P_A) \cdot P_B$$

$$= (1-P_A)^n \cdot (1-P_B)^{n-1} \cdot P_B$$

$$\therefore P(A \text{ not hit}) = \sum_{n=1}^{\infty} P_B \cdot (1-P_B)^{n-1} \cdot (1-P_A)^n$$

It's a sum of ∞ Geometric Series

$$\therefore P(A \text{ not hit}) = \frac{a_1}{1-r}$$

$$= \frac{P_B (1-P_A)}{1-(1-P_B)(1-P_A)}$$

(ii) $P(\text{Both } A, B \text{ are hit})$

Similar to (i)

$$P(\text{Both } A \,\&\, B \text{ hit}) = \sum_{n=1}^{\infty} (\text{Both } A \,\&\, B \text{ hit}, \ n^{th} \text{ Round})$$

$$= \sum_{n=1}^{\infty} \left[(1-P_A)(1-P_B)\right]^{n-1} P_A P_B$$

$$= \frac{P_A P_B}{1-(1-P_A)(1-P_B)}$$

(iii) Let $S \Leftrightarrow$ Duel Ending in the n^{th} Round

A hit	B hit	Both hit	Both^{S} Miss

$$\therefore P(S) = P(A \text{ hit}, n^{th} \text{ Round}) + P(B \text{ hit}, n^{th} \text{ Round}) + P(A,B \text{ hit}, n^{th} \text{ Round}) + \underbrace{P(A, B \text{ Miss}, n^{th} \text{ Round})}_{=0 \ \because \ A, B \text{ Miss Duel won't end}}$$

$$= \left[(1-P_A)(1-P_B)\right]^{n-1} P_A (1-P_B) + \left[(1-P_A)(1-P_B)\right]^{n-1} P_B (1-P_A) + \left[(1-P_A)(1-P_B)\right]^{n-1} P_A P_B$$

$$= \left[(1-P_A)(1-P_B)\right]^{n-1} \left[P_A(1-P_B) + P_B(1-P_A) + P_A P_B\right]$$

$$= \left[(1-P_A)(1-P_B)\right]^{n-1} \left[P_A + P_B - P_A P_B\right]$$

$$= \left[(1-P_A)(1-P_B)\right]^{n-1} \left[1-(1-P_A)(1-P_B)\right]$$

(iv)

$P(A^c$, duel ends @ n^{th} round$)$

$= P(B$ hits , n^{th} round $)$

$= [(1-P_A)(1-P_B)]^{n-1} \; P_B$

$P(A^c \mid n^{th}$ round$)$

$= P(A^c, n^{th}$ round$) / P(n^{th}$ round$)$

$$= \frac{[(1-P_A)(1-P_B)]^{n-1} P_B}{[(1-P_A)(1-P_B)]^{n-1}(P_A+P_B-P_AP_B)}$$

$$= \frac{P_B}{P_A+P_B-P_AP_B}$$

(v)

$P(A, B \mid n^{th}$ round$)$

$$= \frac{P(A, B \text{ hit}, n^{th} \text{ round})}{P(n^{th} \text{ round})}$$

$$= \frac{[(1-P_A)(1-P_B)]^{n-1} P_AP_B}{[(1-P_A)(1-P_B)]^{n-1}(P_A+P_B-P_AP_B)}$$

$$= \frac{P_A P_B}{P_A+P_B-P_AP_B}$$

(b) Let R.V. $X \sim$ # of Faculties Isolated

Let $$x_i \sim \begin{cases} 1 & i^{th} \text{ Faculty Iso} \\ 0 & i^{th} \text{ faculty not Iso} \end{cases} \quad i \in [1, 18]$$

$$\therefore X = \sum_{i=1}^{18} x_i$$

$$\therefore E[X] = E\left[\sum_{i=1}^{18} x_i\right] = \sum_{i=1}^{18} E[x_i] = 18 \cdot E[x_i]$$

$\because x_i \sim$ Bernoulli R.V.

$$\therefore E[x_i] = P[x_i = 1] = P[i^{th} \text{ Faculty Iso}]$$

Assume if $x_i \in ECE$

"Iso":

Left ECE Right

$\uparrow$ $\uparrow$

$12/17$ $11/16$

$$\therefore P[x_i = 1] = P[x_i = 1 \mid x_i \in ECE]\, P[x_i \in ECE]$$
$$+$$
$$P[x_i = 1 \mid x_i \in CS]\, P[x_i \in CS]$$
$$+$$
$$P[x_i = 1 \mid x_i \in MTH]\, P[x_i \in MTH]$$

$$= \left(\tfrac{12}{17} \cdot \tfrac{11}{16} \cdot \tfrac{6}{18}\right) \times 3$$

$$= \tfrac{12}{17} \cdot \tfrac{11}{16} = \frac{33}{68}$$

$$\therefore E[x_i] = 33/68$$

$$\therefore E[X] = 18 \cdot 33/68$$

$$= \frac{297}{34}$$

(ii) Similarly $E[X] = 18\, E[x_i]$

Semi
- Happy

$\underline{\quad\quad}$ $\underline{ECE}$ $\underline{\quad\quad}$

$\frac{5}{17}$ $\qquad\qquad$ $\frac{12}{16}$

OR

$\frac{12}{17}$ $\qquad\qquad$ $\frac{5}{16}$

$$\therefore E[x_i] = \left(\frac{5}{17}\cdot\frac{12}{16} + \frac{12}{17}\cdot\frac{5}{16}\right)\cdot\frac{1}{3}\cdot 3$$

$$= \frac{15}{34}$$

$$\therefore E[X] = 18\, E[x_i]$$

$$= \frac{135}{17}$$

(iii)

$\underline{\quad\quad}$ $\underline{ECE}$ $\underline{\quad\quad}$

$\frac{5}{17}$ $\qquad\qquad$ $\frac{4}{16}$

$$\therefore E[x_i] = \frac{5}{17}\cdot\frac{4}{16} = \frac{5}{68}$$

$$\therefore E[X] = 18\, E[x_i] = \frac{45}{34}$$

(d)

$$E[Ax+b] = E[Ax] + E[b]$$

$$= A\,E[x] + b$$

$$= A \cdot \begin{bmatrix} E[x_1] \\ E[x_2] \\ \vdots \\ E[x_n] \end{bmatrix} + b$$

(e) $Cov[x] = E[(x - E[x])(x - E[x])^T]$

$$Cov[Ax+b] = E[(Ax + \cancel{b} - A \cdot E[x] - \cancel{b})(Ax + \cancel{b} - A \cdot E[x] - \cancel{b})^T]$$

$$= E[A(x - E[x]) \cdot (A(x - E[x])^T]$$

$$= E[A \cdot [(x - E[x])(x - E[x])^T] \cdot A^T]$$

$$= E[A] \cdot E[(x - E[x])(x - E[x])^T] \cdot E[A^T]$$

$$= A \cdot Cov[x] \cdot A^T$$

(c) $D \sim$ Cancer Dangerous

$+|D \sim$ True Positive Test

$+|D^c \sim$ False Positive Test

$P[D] = 0.0005 \quad P[+|D] = 0.9 \quad P[+|D^c] = 0.01$

(i) $$P[D|+] = \frac{P[D,+]}{P[+]}$$

$$= \frac{P[+|D]\,P[D]}{P[+|D]\,P[D] + P[+|D^c]\,P[D^c]}$$

$$= \frac{0.9 \times 0.0005}{0.9 \times 0.0005 + 0.01 \times (1-0.0005)}$$

$$= \frac{90}{2089} \approx 0.043$$

(ii) $$P[D|-] = \frac{P[D,-]}{P[-]} = \frac{P[-|D]\,P[D]}{P[-|D]\,P[D] + P[-|D^c]\,P[D^c]}$$

$$= \frac{(1-0.9) \times 0.0005}{(1-0.9) \times 0.0005 + (1-0.01) \times (1-0.0005)}$$

$$\approx 5.0528 \times 10^{-5}$$

Q3.

(a) $\nabla_{\vec{x}}\, x^T A y = \dfrac{\partial (x^T A y)}{\partial x} = \dfrac{\partial (x^T (Ay))}{\partial x}$

$\because A \in \mathbb{R}^{n\times m}$ $\quad \because x \in \mathbb{R}^{n}$

$y \in \mathbb{R}^{m\times 1}$ $\quad Ay \in \mathbb{R}^{n}$

$\therefore \nabla_{\vec{x}}\, x^T A y = Ay$ (property 69)

Check: ① $Ay \in \mathbb{R}^n \quad x \in \mathbb{R}^n$,

② if all scalar

$$\frac{\partial (x^T A y)}{\partial x} = \frac{\partial (x A y)}{\partial x} = Ay$$

(b) $\nabla_{\vec{y}}\, x^T A y = \dfrac{\partial ((x^T A) y)}{\partial y} = \dfrac{\partial ((A^T x)^T y)}{\partial y}$

$A^T \in \mathbb{R}^{m\times n}$ $\quad \therefore A^T x \in \mathbb{R}^{m}$ $\quad y \in \mathbb{R}^{m}$

$x \in \mathbb{R}^{n}$

$\therefore \nabla_{\vec{y}}\, x^T A y = \nabla_{\vec{y}} (A^T x)^T y = A^T x$ (property 69)

Check: ① $A^T x \in \mathbb{R}^m \quad y \in \mathbb{R}^m$

② if all scalar

$$\frac{\partial x^T A y}{\partial y} = x^T A = A^T x$$

(c) $\nabla_A\, x^T A y = \underset{\underset{n\times 1}{\uparrow}}{x}\, \underset{\underset{1\times m}{\uparrow}}{y^T}$ (property 70)

Check: ① $xy^T \in \mathbb{R}^{n\times m} \quad A \in \mathbb{R}^{n\times m}$

② if all scalar

$$\frac{\partial x^T A y}{\partial A} = x^T y = x y^T$$

(d) $f = x^T A x + b^T x$

$\nabla_x f = \dfrac{\partial (x^T A x + b^T x)}{\partial x} = \dfrac{\partial (x^T A x)}{\partial x} + \dfrac{\partial (b^T x)}{\partial x}$

$= (A + A^T) x + b$ (property 97)

check: ①

$(\underset{\underset{n\times n}{\uparrow}}{A} + \underset{\underset{n\times n}{\uparrow}}{A^T}) \underset{\underset{n\times 1}{\uparrow}}{x} + \underset{\underset{n\times 1}{\uparrow}}{b} \in \mathbb{R}^{n\times 1}$

$x \in \mathbb{R}^{n\times 1}$

② if all scalar

$$\frac{\partial f}{\partial x} = \frac{\partial (Ax^2 + bx)}{\partial x}$$

$$= 2Ax + b = (A + A^T)x + b$$

(e) $f = Tr(AB)$

$$\nabla_A f = \frac{\partial(Tr(AB))}{\partial A} = B^T \quad \text{(property 100)}$$

$A \in \mathbb{R}^{n\times n}$
$B \in \mathbb{R}^{n\times n}$

Check:

① $B^T \in \mathbb{R}^{n\times n}$

$A \in \mathbb{R}^{n\times n}$

② if all scalar

$$\frac{\partial(Tr(AB))}{\partial A} = \frac{\partial(AB)}{\partial A}$$
$$= B = B^T$$

(f) $f = Tr(BA + A^TB + A^2B)$

$\because Tr(A^T) = Tr(A)$

$Tr(AB) = Tr(BA)$

$Tr(A+B) = Tr(A) + Tr(B)$

$$\therefore f = Tr(BA) + Tr(A^TB) + Tr(A^2B)$$
$$= Tr(AB) + Tr(A^TB) + Tr(A^2B)$$

$$\therefore \nabla_A f = \frac{\partial(Tr(AB))}{\partial A} + \frac{\partial(A^TB)}{\partial A} + \frac{\partial(Tr(A^2B))}{\partial A}$$

$$= \underset{\substack{\uparrow \\ \text{property} \\ 100}}{B^T} + \underset{\substack{\uparrow \\ \text{property} \\ 103}}{B} + \underset{\substack{\uparrow \\ \text{property} \\ 107}}{(AB+BA)^T}$$

Check:

① $\underset{\substack{\uparrow \\ n\times n}}{B^T} + \underset{\substack{\uparrow \\ n\times n}}{B} + (\underset{\substack{\uparrow \\ n\times n}}{AB} + \underset{\substack{\uparrow \\ n\times n}}{BA})^T \in \mathbb{R}^{n\times n}$

$A \in \mathbb{R}^{n\times n}$

② if all scalar

$$f = Tr(BA + A^TB + A^2B)$$
$$= 2AB + A^2B$$

$$\frac{\partial f}{\partial A} = 2B + 2AB$$
$$= B^T + B + (AB+BA)^T$$

(g) $\|A+\lambda B\|_F^2$

$$= Tr\left((A+\lambda B)^T(A+\lambda B)\right)$$

$$= Tr\left((A^T+\lambda B^T)(A+\lambda B)\right)$$

$$= Tr\left(A^TA + A^T\lambda B + \lambda B^TA + \lambda^2 B^TB\right)$$

$$= Tr(A^TA) + \lambda\, Tr(A^TB) + \lambda Tr(B^TA) + \lambda^2\, Tr(B^TB)$$

$$= Tr(A^TA) + \lambda \cdot Tr(A^TB) + \lambda\, Tr\left((B^TA)^T\right) + \lambda^2\, Tr(B^TB)$$

$$= Tr(A^TA) + 2\lambda \cdot Tr(A^TB) + \lambda^2\, Tr(B^TB)$$

Dropping term with no A dependance, $f = Tr(A^TA) + 2\lambda \cdot Tr(A^TB)$

$$\therefore \nabla_A f = \underset{\substack{\uparrow \\ \text{property} \\ 115}}{2A} + \underset{\substack{\uparrow \\ \text{property} \\ 103}}{2\lambda \cdot B}$$

Check. ① $A \in \mathbb{R}^{n\times n}$

$$\underset{\substack{\uparrow \\ n\times n}}{2A} + \underset{\substack{\uparrow \\ n\times n}}{2\lambda \cdot B} \in \mathbb{R}^{n\times n}$$

② if all scalar

$$\frac{\partial f}{\partial A} = \frac{\partial\left((A+\lambda B)^2\right)}{\partial A}$$

$$= 2(A+\lambda B)\cdot 1$$

$$= 2A + 2\lambda B$$

Q4.

$X \in \mathbb{R}^{m\times n}$; $X = \begin{bmatrix} | & | & & | \\ x_1 & x_2 & \cdots & x_n \\ | & | & & | \end{bmatrix}$ $\quad \|x_1\|_2^2 = \sum_{i=1}^{m} (x_{i1})^2 \Leftarrow$ Sum all entries in Column 1

$\therefore$ n columns

$$\|X\|_F^2 = \sum_{i=1}^{n} \|x_i\|_2^2 \Leftarrow \text{Sum Column norm}$$

$$= \sum_{i=1}^{m} \sum_{j=1}^{n} (x_{ij})^2$$

$\because \underset{\mathbb{R}^n}{\underset{\uparrow}{\hat{y}}} = W \underset{\mathbb{R}^n}{\underset{\uparrow}{x}}$ $\quad \therefore W \in \mathbb{R}^{n\times n}$ $\quad \cdot\ \mathcal{L}(W) = \frac{1}{2}\sum_{i=1}^{n} \| y^{(i)} - Wx^{(i)} \|^2$

$\therefore$ Find: $\frac{\partial \mathcal{L}(W)}{\partial W} / \nabla_{\vec{W}}(\mathcal{L}(W))$

Let $Y = \begin{bmatrix} | & | & & | \\ y^{(1)}, & y^{(2)}, & \cdots, & y^{(n)} \\ | & | & & | \end{bmatrix}$

$X = \begin{bmatrix} | & | & & | \\ x^{(1)}, & x^{(2)}, & \cdots, & x^{(n)} \\ | & | & & | \end{bmatrix}$

$\therefore$ According to $\|X\|_F^2 = \sum_{i=1}^{n} \|x_i\|_2^2$

$$\therefore \mathcal{L}(W) = \frac{1}{2}\sum_{i=1}^{n} \| y^{(i)} - Wx^{(i)} \|^2$$

$$= \frac{1}{2} \cdot \| Y - WX \|_F^2$$

$$= \frac{1}{2} \cdot Tr\left((Y-WX)^T \cdot (Y-WX)\right)$$

$$= \frac{1}{2} \cdot Tr\left((Y^T - X^T W^T) \cdot (Y-WX)\right)$$

$$= \frac{1}{2} \cdot Tr\left(Y^T Y - Y^T W X - X^T W^T Y + X^T W^T W X\right)$$

$$= \frac{1}{2} \cdot \left[Tr(Y^T Y) - Tr(Y^T W X) - Tr\left((Y^T W X)^T\right) + Tr(X^T W^T W X) \right]$$

$$= \frac{1}{2} \cdot \left[\cancel{Tr(Y^T Y)} - 2 \cdot Tr(Y^T W X) + Tr(X^T W^T W X) \right]$$

Drop terms with no dependence to W

$$\therefore \mathcal{L}(W) = -Tr(Y^T W X) + \frac{1}{2} \cdot Tr(X^T W^T W X)$$

$$\because Tr(ABC) = Tr(CAB) = Tr(BCA)$$

$$\therefore Tr(\underline{X^T}\,\underline{W^T W X}) = Tr(W^T W X X^T)$$

$$\therefore \frac{\partial \mathcal{L}(W)}{\partial W} = -(Y^T)^T X^T + \frac{1}{2} \cdot \left(W \cdot (XX^T)^T + W \cdot (XX^T)\right)$$

$$= -YX^T + \frac{1}{2} \cdot 2 \cdot WXX^T$$

$$= -YX^T + WXX^T$$ (property 101, 113)

Set Left Hand to 0

$$\therefore YX^T = WXX^T$$

$$\therefore W = YX^T(XX^T)^{-1}$$

Q5.

We've known from the Lecture that:

$$\mathcal{L}(\theta) = \frac{1}{2} \sum_{i=1}^{N} (y^{(i)} - \theta^T \hat{x}^{(i)})^2$$

$$= \frac{1}{2} (Y^T Y - 2Y^T Y \theta + \theta^T X^T X \theta)$$

Now: $\mathcal{L}(\theta) = \frac{1}{2} \sum_{i=1}^{N} (y^{(i)} - \theta^T \hat{x}^{(i)})^2 + \frac{\lambda}{2} \|\theta\|_2^2$

$$= \frac{1}{2} (\cancel{Y^T}Y - 2Y^T X \theta + \theta^T X^T X \theta) + \frac{\lambda}{2} \theta^T \theta$$

Find $\frac{\partial(\mathcal{L}(\theta))}{\partial \theta}$: Get rid of $Y^T Y$ since it's not θ related

$$\mathcal{L}(\theta) = -Y^T X \theta + \frac{1}{2} \theta^T X^T X \theta + \frac{\lambda}{2} \theta^T \theta$$

$$\nabla_{\vec{\theta}} (-Y^T X \theta) = -(Y^T X)^T \quad \text{(property 69)}$$

$$= -X^T Y$$

$$\frac{1}{2} \theta^T X^T X \theta + \frac{\lambda}{2} \theta^T \theta$$

$$= \frac{1}{2} \theta^T (X^T X \theta + \lambda I \cdot \theta)$$

$$= \frac{1}{2} \theta^T \cdot (X^T X + \lambda I) \cdot \theta$$

$$\therefore \nabla_{\vec{\theta}} \left(\frac{1}{2} \theta^T X^T X \theta + \frac{\lambda}{2} \theta^T \theta\right) = \frac{1}{2} \cdot \left(X^T X + \lambda I + (X^T X + \lambda I)^T\right) \theta$$

$$= \frac{1}{2} \cdot (X^T X + \lambda I + X^T X + \lambda I) \theta$$

$$= (X^T X + \lambda I) \theta$$

$$\therefore \nabla_{\vec{\theta}} (\mathcal{L}(\theta)) = -X^T Y + (X^T X + \lambda I) \theta$$

Let $\nabla_{\vec{\theta}} (\mathcal{L}(\theta)) = 0$

$$\therefore X^T Y = (X^T X + \lambda I) \theta^* \quad \therefore \theta^* = (X^T X + \lambda I)^{-1} X^T Y$$

linear_regression

January 22, 2024

0.1 Linear regression workbook

This workbook will walk you through a linear regression example. It will provide familiarity with Jupyter Notebook and Python. Please print (to pdf) a completed version of this workbook for submission with HW #1.

ECE C147/C247, Winter Quarter 2024, Prof. J.C. Kao, TAs: T.Monsoor, Y. Liu, S. Rajesh, L. Julakanti, K. Pang

[19]:

```
import numpy as np
import matplotlib.pyplot as plt

#allows matlab plots to be generated in line
%matplotlib inline
```

0.1.1 Data generation

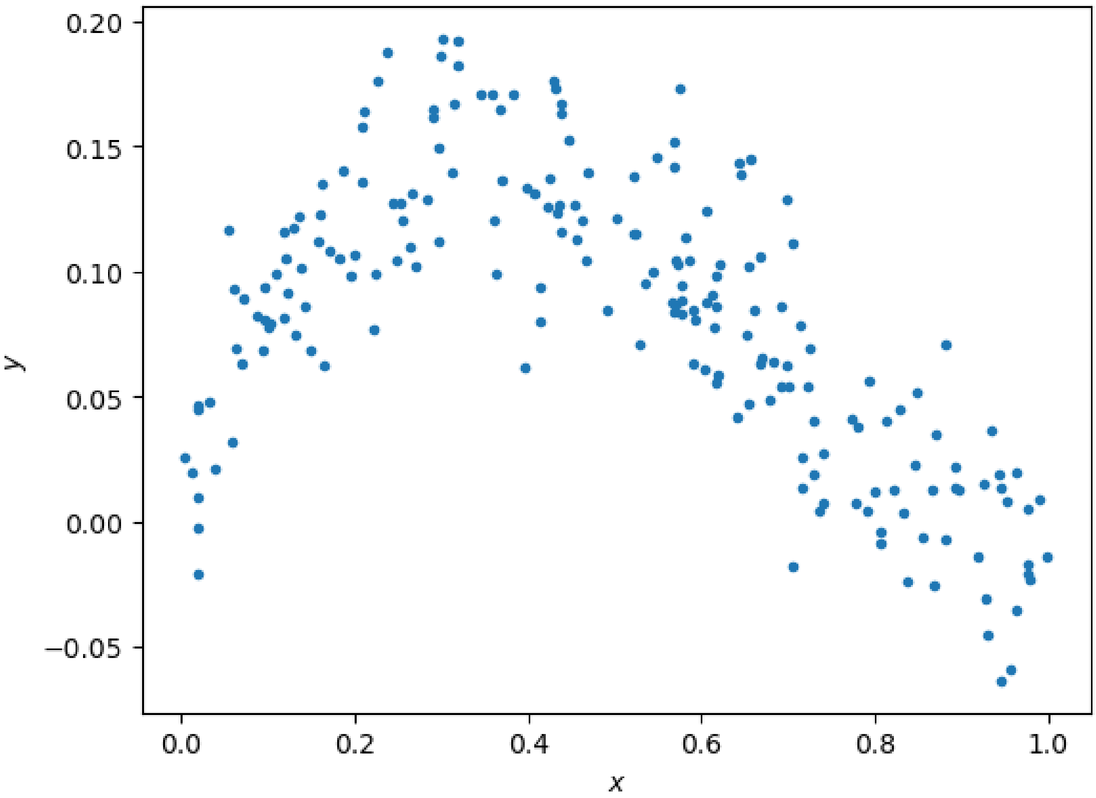
For any example, we first have to generate some appropriate data to use. The following cell generates data according to the model: $y = x - 2x^2 + x^3 + \epsilon$

[20]:

```
np.random.seed(0)  # Sets the random seed.
num_train = 200     # Number of training data points

# Generate the training data
x = np.random.uniform(low=0, high=1, size=(num_train,))
y = x - 2*x**2 + x**3 + np.random.normal(loc=0, scale=0.03, size=(num_train,))
f = plt.figure()
ax = f.gca()
ax.plot(x, y, '.')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
```

[20]:

```
Text(0, 0.5, '$y$')
```



0.1.2 QUESTIONS:

Write your answers in the markdown cell below this one:

(1) What is the generating distribution of x?

(2) What is the distribution of the additive noise ϵ?

0.1.3 ANSWERS:

(1) x is a uniform distribution

(2) ϵ is a normal distribution with mean 0 and STD 0.03

0.1.4 Fitting data to the model (5 points)

Here, we'll do linear regression to fit the parameters of a model $y = ax + b$.

[21]:
```
# xhat = (x, 1)
xhat = np.vstack((x, np.ones_like(x)))

# ==================== #
# START YOUR CODE HERE #
# ==================== #
```

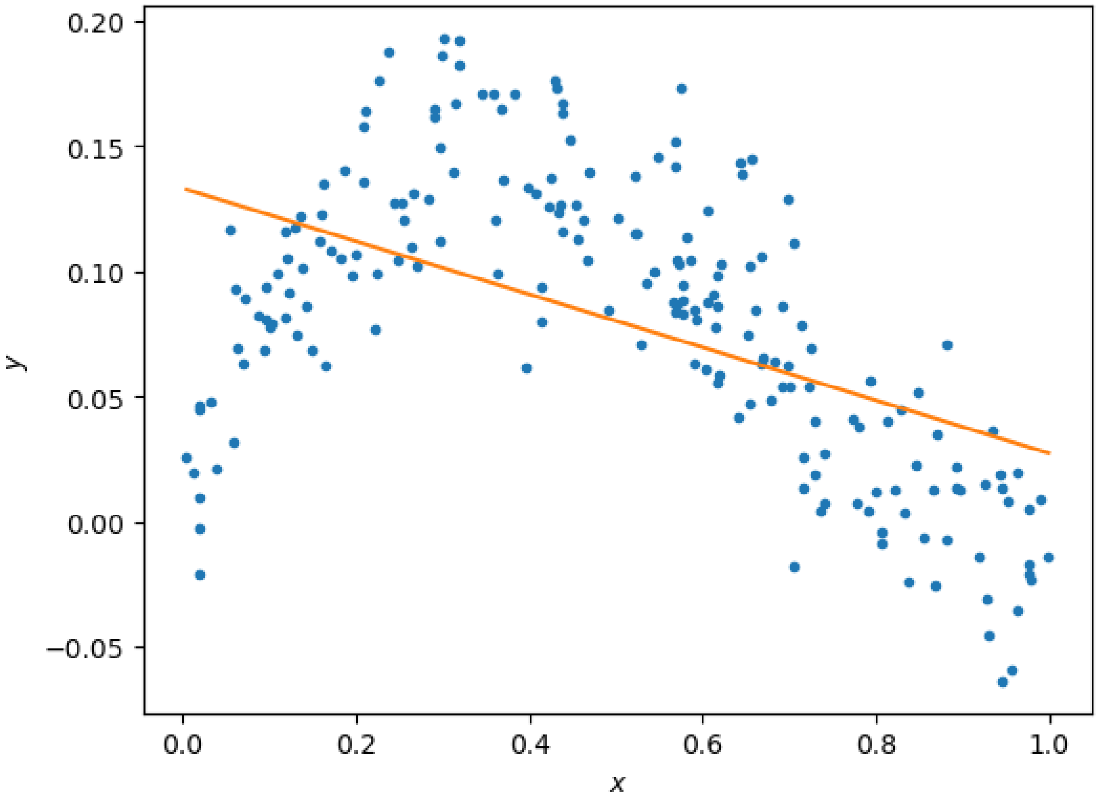
```
# GOAL: create a variable theta; theta is a numpy array whose elements are [a,␣
 ↪b]

theta = (np.linalg.inv((xhat)@(xhat.T))) @ (xhat@y) # please modify this line

# ================== #
# END YOUR CODE HERE #
# ================== #
```

[22]:

```
# Plot the data and your model fit.
f = plt.figure()
ax = f.gca()
ax.plot(x, y, '.')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')

# Plot the regression line
xs = np.linspace(min(x), max(x),50)
xs = np.vstack((xs, np.ones_like(xs)))
plt.plot(xs[0,:], theta.dot(xs))
```

[22]:

```
[<matplotlib.lines.Line2D at 0x1e7a296c790>]
```



0.1.5 QUESTIONS

(1) Does the linear model under- or overfit the data?

(2) How to change the model to improve the fitting?

0.1.6 ANSWERS

(1) It underfits the data.

(2) We can instead use a higher order polynomial.

0.1.7 Fitting data to the model (5 points)

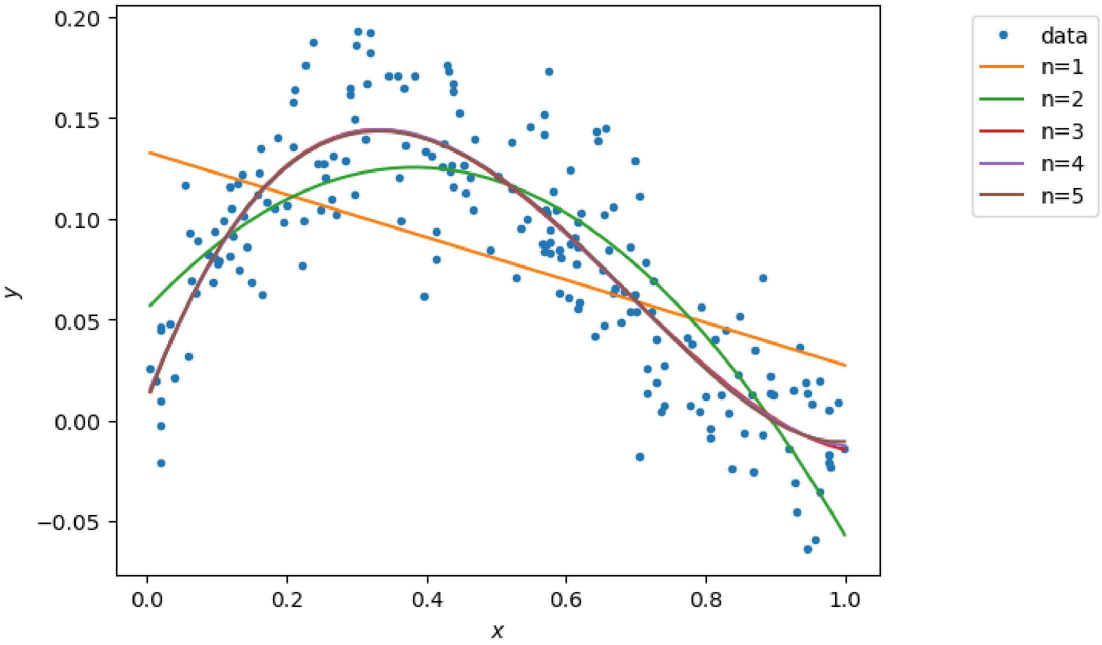
Here, we'll now do regression to polynomial models of orders 1 to 5. Note, the order 1 model is the linear model you prior fit.

[23]:

```
N = 5
xhats = []
thetas = []

# ==================== #
# START YOUR CODE HERE #
# ==================== #

# GOAL: create a variable thetas.
# thetas is a list, where theta[i] are the model parameters for the polynomial fit of order i+1.
#   i.e., thetas[0] is equivalent to theta above.
#   i.e., thetas[1] should be a length 3 np.array with the coefficients of the x^2, x, and 1 respectively.
#   ... etc.

xhats.append(xhat)
thetas.append(theta)
xhat_i = xhat
for i in range(2,6):
    x_power = np.array(x**i)
    xhat_i = np.vstack((x_power, xhat_i))
    theta_i = (np.linalg.inv((xhat_i)@(xhat_i.T))) @ (xhat_i@y)
    xhats.append(xhat_i)
    thetas.append(theta_i)

# ================== #
# END YOUR CODE HERE #
# ================== #
```

[24]:

```
# Plot the data
f = plt.figure()
ax = f.gca()
ax.plot(x, y, '.')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')

# Plot the regression lines
plot_xs = []
for i in np.arange(N):
    if i == 0:
        plot_x = np.vstack((np.linspace(min(x), max(x),50), np.ones(50)))
    else:
        plot_x = np.vstack((plot_x[-2]**(i+1), plot_x))
    plot_xs.append(plot_x)

for i in np.arange(N):
    ax.plot(plot_xs[i][-2,:], thetas[i].dot(plot_xs[i]))

labels = ['data']
[labels.append('n={}'.format(i+1)) for i in np.arange(N)]
bbox_to_anchor=(1.3, 1)
lgd = ax.legend(labels, bbox_to_anchor=bbox_to_anchor)
```



0.1.8 Calculating the training error (5 points)

Here, we'll now calculate the training error of polynomial models of orders 1 to 5.

[25]:

```
training_errors = []

# ==================== #
# START YOUR CODE HERE #
# ==================== #

# GOAL: create a variable training_errors, a list of 5 elements,
# where training_errors[i] are the training loss for the polynomial fit of␣
 ↪order i+1.
for i in range(5):
    y_predict = thetas[i]@xhats[i]

    my_errors = 0

    for j in range(len(y)):
        error_ = ((y[j] - y_predict[j])**2)
        my_errors = my_errors + error_

    training_errors.append(my_errors)

# ================== #
# END YOUR CODE HERE #
# ================== #

print ('Training errors are: \n', training_errors)
```

```
Training errors are:
 [0.4759922176725402, 0.21849844418537054, 0.16339207602210745,
0.16330707470593958, 0.16322958391050585]
```

0.1.9 QUESTIONS

(1) What polynomial has the best training error?

(2) Why is this expected?

0.1.10 ANSWERS

(1) 5th order polynomial has the best training error.

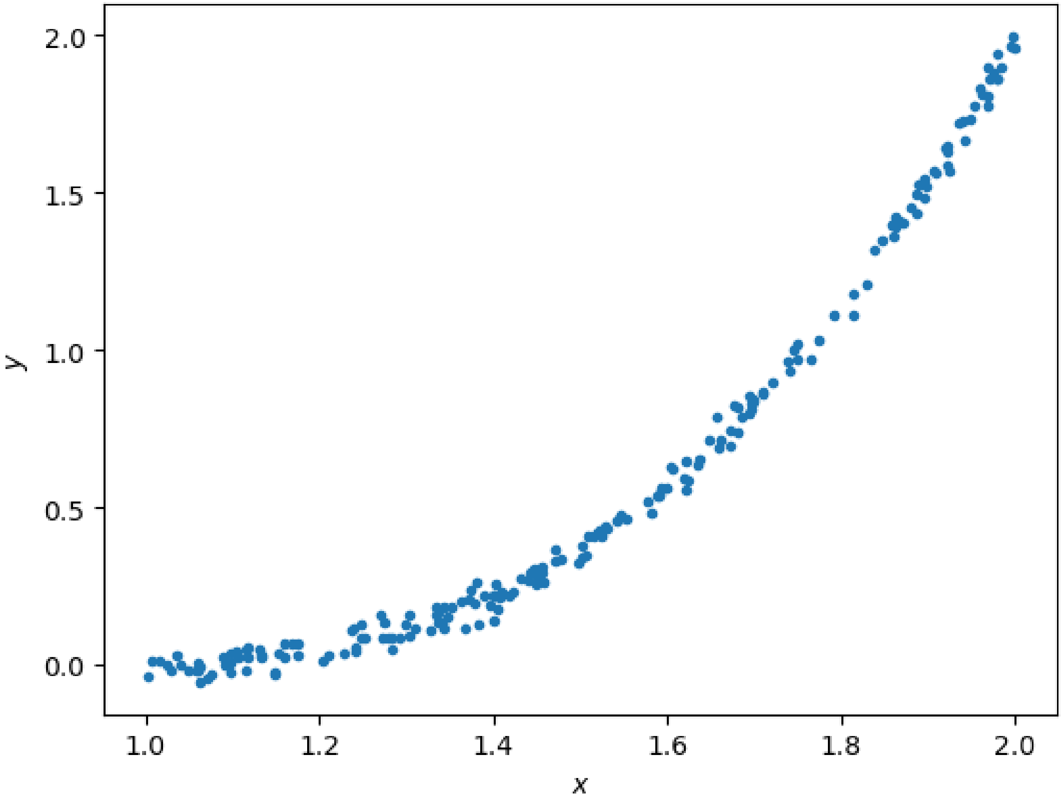
(2) This is expected because we can fit more data training data points with higer order polynomial models.

0.1.11 Generating new samples and testing error (5 points)

Here, we'll now generate new samples and calculate testing error of polynomial models of orders 1 to 5.

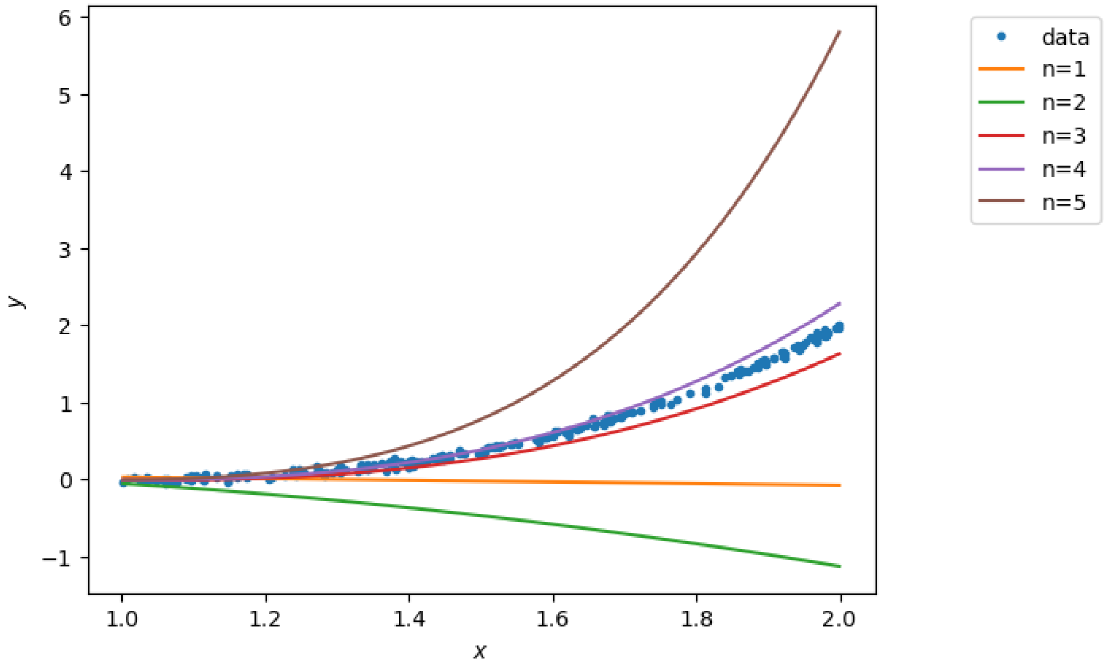
[26]:
```
x = np.random.uniform(low=1, high=2, size=(num_train,))
y = x - 2*x**2 + x**3 + np.random.normal(loc=0, scale=0.03, size=(num_train,))
f = plt.figure()
ax = f.gca()
ax.plot(x, y, '.')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')
```

[26]:
```
Text(0, 0.5, '$y$')
```



[27]:
```
xhats = []
for i in np.arange(N):
    if i == 0:
        xhat = np.vstack((x, np.ones_like(x)))
        plot_x = np.vstack((np.linspace(min(x), max(x),50), np.ones(50)))
    else:
        xhat = np.vstack((x**(i+1), xhat))
        plot_x = np.vstack((plot_x[-2]**(i+1), plot_x))

    xhats.append(xhat)
```

[28]:
```
# Plot the data
f = plt.figure()
ax = f.gca()
ax.plot(x, y, '.')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')

# Plot the regression lines
plot_xs = []
for i in np.arange(N):
    if i == 0:
        plot_x = np.vstack((np.linspace(min(x), max(x),50), np.ones(50)))
    else:
        plot_x = np.vstack((plot_x[-2]**(i+1), plot_x))
    plot_xs.append(plot_x)

for i in np.arange(N):
    ax.plot(plot_xs[i][-2,:], thetas[i].dot(plot_xs[i]))

labels = ['data']
[labels.append('n={}'.format(i+1)) for i in np.arange(N)]
bbox_to_anchor=(1.3, 1)
lgd = ax.legend(labels, bbox_to_anchor=bbox_to_anchor)
```



[29]:

```
testing_errors = []

# ==================== #
# START YOUR CODE HERE #
# ==================== #

# GOAL: create a variable testing_errors, a list of 5 elements,
# where testing_errors[i] are the testing loss for the polynomial fit of order␣
 ↪i+1.
for i in range(5):
    y_predict = thetas[i]@xhats[i]
    my_errors = 0
    for j in range(len(y)):
        error_ = ((y[j] - y_predict[j])**2)
        my_errors = my_errors + error_
    testing_errors.append(my_errors)

# ================== #
# END YOUR CODE HERE #
# ================== #

print ('Testing errors are: \n', testing_errors)
```

```
Testing errors are:
 [161.72330369101164, 426.38384890115776, 6.251394216552787, 2.3741530378949403,
429.82043635305246]
```

0.1.12 QUESTIONS

(1) What polynomial has the best testing error?

(2) Why polynomial models of orders 5 does not generalize well?

0.1.13 ANSWERS

(1) 4th order polynomial has the best testing error.

(2) Because it has the issue of overfitting. It fits the training data well but instead loses generality since it can’t predict unseen data well.

[ ]:

```
```