Homework 1

1 Problem 1

1.1 Problem 1a

1.1.1 Problem Statement

1. (25 points) **Linear algebra refresher.**
   (a) (12 points) Let $\mathbf{A}$ be a square matrix, and further let $\mathbf{A}\mathbf{A}^T = \mathbf{I}$.
      i. (3 points) Construct a 2×2 example of $\mathbf{A}$ and derive the eigenvalues and eigenvectors of this example. Show all work (i.e., do not use a computer's eigenvalue decomposition capabilities). You may not use a diagonal matrix as your 2×2 example. What do you notice about the eigenvalues and eigenvectors?
      ii. (3 points) Show that $\mathbf{A}$ has eigenvalues with norm 1.
      iii. (3 points) Show that the eigenvectors of $\mathbf{A}$ corresponding to distinct eigenvalues are orthogonal.
      iv. (3 points) In words, describe what may happen to a vector $\mathbf{x}$ under the transformation $\mathbf{A}\mathbf{x}$.

1.1.2 Solution

1. A is a square matrix where $AA^T = I$

a) i) if $AA^T = I$ then A is an orthonormal matrix

$$A = \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix}$$

$$AA^T = \begin{bmatrix} 1/\sqrt{2} & -1/\sqrt{2} \\ 1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = I$$

$$\det(\lambda I - A) = 0$$

$$\det \left| \begin{bmatrix} \lambda - 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & \lambda - 1/\sqrt{2} \end{bmatrix} \right| = 0$$

$$(\lambda - 1/\sqrt{2})^2 + \frac{1}{2} = 0$$

$$\lambda^2 - \frac{2}{\sqrt{2}}\lambda + 1 = 0$$

$$\lambda^2 - \sqrt{2}\lambda + 1 = 0$$

$$\lambda = \frac{\sqrt{2} \pm \sqrt{2 - (4)(1)(1)}}{2}$$

$$\lambda = \frac{\sqrt{2}}{2} \pm \frac{\sqrt{2}i}{2}$$

$$\boxed{\lambda = \frac{1}{\sqrt{2}} \pm \frac{i}{\sqrt{2}}}$$ norm of 1

$$\begin{bmatrix} \frac{i}{\sqrt{2}} & 1/\sqrt{2} \\ -1/\sqrt{2} & \frac{i}{\sqrt{2}} \end{bmatrix} v = 0$$

$$\boxed{v = \begin{bmatrix} i \\ 1 \end{bmatrix}, \begin{bmatrix} -i \\ 1 \end{bmatrix}}$$ orthogonal

ii) eigenvalues of 1

$Av = \lambda v$

$(Av)^T = (\lambda v)^T$

$(Av)^T Av = (\lambda v)^T \lambda v$

$v^T \underbrace{A^T A}_{I} v = |\lambda|^2 v^T v$

$v^T v = |\lambda|^2 v^T v$

$\boxed{|\lambda| = 1}$

norm of eigenvalue must be 1

iii) $\underset{n\times1}{A v_1} = \lambda_1 v_1$ ①

$\underset{1\times n}{v_2} A = \lambda_2 v_2$ ②

multiplying ① by v_2 on left

$v_2 A v_1 = \lambda_1 v_2 v_1$

multiplying ② by v_1 on right

$v_2 A v_1 = \lambda_2 v_2 v_1$

equating the two expressions

$\lambda_1 v_2 v_1 = \lambda_2 v_2 v_1$

$(\lambda_1 - \lambda_2) v_2 v_1 = 0$

if $\lambda_1 \neq \lambda_2$

then v_1 and v_2 must be orthogonal bc $\underset{1\times n}{v_2} \underset{n\times1}{v_1} = 0$

iv. Under transformation Ax,
the norm of x is preserved so it can
only experience rotation or reflection.

1.2 Problem 1b

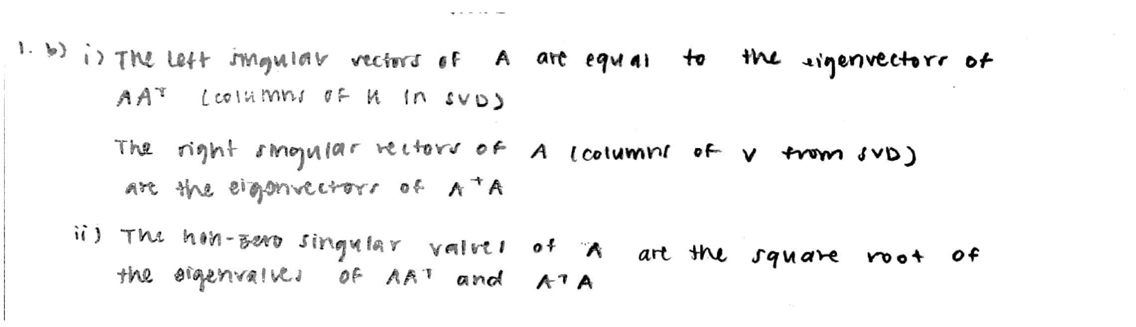
1.2.1 Problem Statement

(b) (8 points) Let **A** be a matrix.

i. (4 points) What is the relationship between the singular vectors of **A** and the eigenvectors of $\mathbf{A}\mathbf{A}^T$? What about $\mathbf{A}^T\mathbf{A}$?

ii. (4 points) What is the relationship between the singular values of **A** and the eigenvalues of $\mathbf{A}\mathbf{A}^T$? What about $\mathbf{A}^T\mathbf{A}$?

1.2.2 Solution



1. b) i) The left singular vectors of A are equal to the eigenvectors of AA^T (columns of U in SVD)

The right singular vectors of A (columns of V from SVD) are the eigenvectors of A^TA

ii) The non-zero singular values of A are the square root of the eigenvalues of AA^T and A^TA

1.3 Problem 1c

1.3.1 Problem Statement

(c) (5 points) True or False. Partial credit on an incorrect solution may be awarded if you justify your answer.

i. Every linear operator in an n-dimensional vector space has n distinct eigenvalues.

ii. A non-zero sum of two eigenvectors of a matrix **A** is an eigenvector.

iii. If a matrix **A** has the positive semidefinite property, i.e., $\mathbf{x}^T\mathbf{A}\mathbf{x} \geq 0$ for all **x**, then its eigenvalues must be non-negative.

iv. The rank of a matrix can exceed the number of non-zero eigenvalues.

v. A non-zero sum of two eigenvectors of a matrix **A** corresponding to the same eigenvalue λ is always an eigenvector.

1.3.2 Solution

c) i. Every linear operator in n-dim vector space has n-distinct eigenvalues

$\boxed{\text{FALSE}}$ can have repeated eigenvalues

ii. A non-zero sum of two eigenvectors of a matrix A is also an eigenvector

$\boxed{\text{FALSE}}$ If v is an eigenvector & −v is. $v - v = 0$ but can't have eigenvector of zeros (trivial solution)

iii. If matrix A has positive semidefinite property $x^T A x \geq 0$ for all x, then its eigenvalues must be non negative

$\boxed{\text{TRUE}}$ $x^T A x = \lambda \underbrace{x^T x}_{>0} \geq 0 \Rightarrow \lambda \geq 0$

iv. The rank of a matrix can exceed the number of non-zero eigenvalues

$\boxed{\text{TRUE}}$ ex. $A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$ $\lambda_1, \lambda_2 = 0$, $\text{rank}(A) = 1$

v. A non-zero sum of two eigenvectors of a matrix A corresponding to the same eigenvalue λ is always an eigenvector

$\boxed{\text{TRUE}}$ $Av_1 = \lambda v_1 \quad Av_2 = \lambda v_2$

$$(A - \lambda I)v_1 = 0$$
$$+ \quad (A - \lambda I)v_2 = 0$$

$$(A - \lambda I)\underbrace{(v_1 + v_2)}_{\bar{v}} = 0$$

2 Problem 2

2.1 Problem 2a

2.1.1 Problem Statement

2. (22 points) **Probability refresher.**

   (a) (9 points) A jar of coins is equally populated with two types of coins. One is type "H50" and comes up heads with probability 0.5. Another is type "H60" and comes up heads with probability 0.6.

      i. (3 points) You take one coin from the jar and flip it. It lands tails. What is the posterior probability that this is an H50 coin?

      ii. (3 points) You put the coin back, take another, and flip it 4 times. It lands T, H, H, H. How likely is the coin to be type H50?

      iii. (3 points) A new jar is now equally populated with coins of type H50, H55, and H60 (with probabilities of coming up heads 0.5, 0.55, and 0.6 respectively. You take one coin and flip it 10 times. It lands heads 9 times. How likely is the coin to be of each possible type?

2.1.2 Solution

2. a)

H50 coin

$$f_k[k] = \begin{cases} 0.5 & \text{Heads} \\ 0.5 & \text{Tails} \\ 0 & \text{ow} \end{cases}$$

H60 coin

$$f_k[k] = \begin{cases} 0.6 & \text{Heads} \\ 0.4 & \text{tails} \\ 0 & \text{ow} \end{cases}$$

i) known: result is tails

$$P(H50 \mid T) = \frac{P(T \mid H50)\,P(H50)}{P(T)}$$ — using Baye's Law

$$= \frac{(0.5)(0.5)}{P(T)}$$

$$P(T) = P(T \mid H50)\,P(H50) + P(T \mid H60)\,P(H60)$$

$$= (0.5)(0.5) + (0.4)(0.5)$$

$$= 0.45$$

$$P(H50 \mid T) = \frac{0.25}{0.45}$$

$$\boxed{P(H50 \mid T) = \frac{5}{9}}$$

ii. $$P(H50 \mid THHH) = \frac{P(THHH \mid H50)\,P(H50)}{P(THHH)}$$

$$= \frac{(0.5)^4(0.5)}{P(THHH)}$$

$$P(THHH) = P(THHH \mid H50)\,P(H50) + P(THHH \mid H60)\,P(H60)$$

$$= (0.5)^4(0.5) + (0.4)(0.6)^3(0.5)$$

$$P(H50 \mid THHH) = \frac{(0.5)^5}{(0.5)^5 + (0.4)(0.6)^3(0.5)}$$

$$\boxed{P(H50 \mid THHH) = 0.4197}$$

Problem 2 C247 Neural Net Homework 1 Helene Levy

2. a) iii.

$$P(H50 \mid H^{(9)}T) = \frac{P(H^{(9)}T \mid H50)\, P(H50)}{P(H^{(9)}T)}$$

$$P(H^{(9)}T) = P(H^{(9)}T \mid H50)P(H50) + P(H^{(9)}T \mid H55)P(H55) + P(H^{(9)}T \mid H60)P(H60)$$

$$= (0.5)^9(0.5)(1/3) + (0.55)^9(0.45)(1/3) + (0.6)^9(0.4)(1/3)$$

$$= 0.0024$$

$$P(H50 \mid H^{(9)}T) = \frac{(0.5)^9(0.5)(1/3)}{0.0024}$$

$$\boxed{P(H50 \mid H^{(9)}T) = 0.1379}$$

$$P(H55 \mid H^{(9)}T) = \frac{(0.55)^9(0.45)(1/3)}{0.0024}$$

$$\boxed{P(H55 \mid H^{(9)}T) = 0.2927}$$

$$P(H60 \mid H^{(9)}T) = \frac{(0.6)^9(0.4)(1/3)}{0.0024}$$

$$\boxed{P(H60 \mid H^{(9)}T) = 0.5694}$$

2.2 Problem 2b

2.2.1 Problem Statement

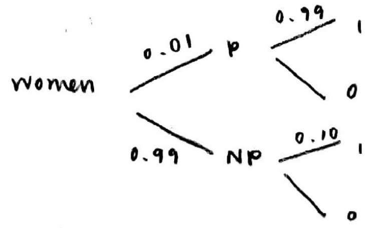
(b) (3 points) Consider a pregnancy test with the following statistics.

- If the woman is pregnant, the test returns "positive" (or 1, indicating the woman is pregnant) 99% of the time.
- If the woman is not pregnant, the test returns "positive" 10% of the time.
- At any given point in time, 99% of the female population is not pregnant.

What is the probability that a woman is pregnant given she received a positive test? The answer should make intuitive sense; given an explanation of the result that you find.

2.2.2 Solution

2. b) pregnancy test   P = pregnant   NP = not pregnant



want $P(P \mid 1) = \dfrac{P(1 \mid P)\,P(P)}{P(1)}$

$P(1) = P(1 \mid P)\,P(P) + P(1 \mid NP)\,P(NP)$

$= (0.99)(0.01) + (0.10)(0.99) = 0.1089$

2. b) $P(P \mid 1) = \dfrac{(0.99)(0.01)}{0.1089}$

$\boxed{P(P \mid 1) = 0.0909}$

A woman that receives a positive test is only actually pregnant 9% of the time $\longrightarrow$ pregnancy tests are not very reliable

2.3 Problem 2c

2.3.1 Problem Statement

(c) (5 points) Let $x_1, x_2, \ldots, x_n$ be identically distributed random variables. A random vector, $\mathbf{x}$, is defined as

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

What is $\mathbb{E}(\mathbf{A}\mathbf{x} + \mathbf{b})$ in terms of $\mathbb{E}(\mathbf{x})$, given that $\mathbf{A}$ and $\mathbf{b}$ are deterministic?

2.3.2 Solution

c)

$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ iid random variables

want $E[Ax+b]$ in terms of $E[x]$

properties of expectations

$E[cX] = cE[x]$

$E[x+c] = E[x]+c$

$Ax = \sum_i \sum_j a_{ij} x_j$

$$E[Ax+b] = E[Ax]+b$$
$$= E\left[\sum_i \sum_j a_{ij} x_j\right] + b$$
$$= \left[\sum_i \sum_j a_{ij} E[x_j]\right] + b$$

$$\boxed{E[Ax+b] = AE[x]+b}$$

2.4 Problem 2d

2.4.1 Problem Statement

(d) (5 points) Let

$$\mathbf{cov}(\mathbf{x}) = \mathbb{E}\left((\mathbf{x} - \mathbb{E}\mathbf{x})(\mathbf{x} - \mathbb{E}\mathbf{x})^T\right)$$

What is $\mathbf{cov}(\mathbf{Ax}+\mathbf{b})$ in terms of $\mathbf{cov}(\mathbf{x})$, given that $\mathbf{A}$ and $\mathbf{b}$ are deterministic?

2.4.2 Solution

2.d) $\text{cov}(x) = E((x - E[x])(x - E[x])^T)$

$$\text{cov}(Ax+b) = E((Ax+b - E[Ax+b])(Ax+b - E[Ax+b])^T)$$
$$= E((Ax+\cancel{b} - AE[x] - \cancel{b})(Ax+\cancel{b} - AE[x] - \cancel{b})^T)$$
$$= E(A(x - E[x])(x - E[x])^T A^T)$$
$$\boxed{\text{cov}(Ax+b) = A\,\text{cov}(x)\,A^T}$$

3 Problem 3

3.1 Problem Statement

3. (13 points) **Multivariate derivatives.**

(a) (2 points) Let $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{y} \in \mathbb{R}^m$, and $\mathbf{A} \in \mathbb{R}^{n\times m}$. What is $\nabla_{\mathbf{x}} \mathbf{x}^T \mathbf{A} \mathbf{y}$?

(b) (2 points) What is $\nabla_{\mathbf{y}} \mathbf{x}^T \mathbf{A} \mathbf{y}$?

(c) (3 points) What is $\nabla_{\mathbf{A}} \mathbf{x}^T \mathbf{A} \mathbf{y}$?

(d) (3 points) Let $f = \mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b}^T \mathbf{x}$. What is $\nabla_{\mathbf{x}} f$?

(e) (3 points) Let $f = \text{tr}(\mathbf{AB})$. What is $\nabla_{\mathbf{A}} f$?

3.2 Solution

3. a) $x \in \mathbb{R}^n,\ y \in \mathbb{R}^m,\ A \in \mathbb{R}^{n\times m}$

$\nabla_x x^T A y$

$$x^TAy = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & & a_{1m} \\ a_{21} & \ddots & & \\ a_{n1} & & \ddots & a_{nm} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

$$= \begin{bmatrix} (x_1a_{11} + \cdots x_n a_{n1}) & (x_1 a_{12} \cdots + x_n a_{n2}) & (x_1 a_{1m} + \cdots + x_n a_{nm}) \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}$$

$$= (x_1a_{11} + \cdots + x_n a_{n1}) y_1 + (x_1 a_{12} + \cdots + x_n a_{n2}) y_2 + (x_1 a_{1m} + \cdots + x_n a_{nm}) y_m$$

first row of $A \times Y$

$$\nabla_x f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} = \begin{bmatrix} a_{11}y_1 + a_{12}y_2 \cdots a_{1m} y_m \\ a_{21}y_1 + a_{22}y_2 \cdots a_{2m} y_m \\ \cdots \end{bmatrix}$$

$$\boxed{\nabla_x x^TAy = Ay}$$

b) $\nabla_y x^T A y$

x multiplied by first column of A

$$\nabla_y f = \begin{bmatrix} \frac{\partial f}{\partial y_1} \\ \vdots \\ \frac{\partial f}{\partial y_m} \end{bmatrix} = \begin{bmatrix} x_1 a_{11} + \cdots + x_n a_{n1} \\ x_1 a_{12} + \cdots + x_n a_{n2} \\ \\ \end{bmatrix}$$

$$\nabla_y f = \overset{m\times n}{A^T} \overset{n\times 1}{x}$$

$$\boxed{\nabla_y x^TAy = A^Tx}$$

c) $\nabla_A x^T A y$

$$\nabla_A f = \begin{bmatrix} \frac{\partial f}{\partial a_{11}} & \frac{\partial f}{\partial a_{12}} & \cdots & \\ \vdots & \ddots & & \\ & & & \frac{\partial F}{\partial nm} \end{bmatrix} = \begin{bmatrix} x_1y_1 & x_1y_2 \cdots & x_1y_m \\ x_2y_1 & \ddots & \vdots \\ \vdots & & \\ x_ny_1 & \cdots & x_ny_m \end{bmatrix} = xy^T$$

$$\boxed{\nabla_A x^TAy = xy^T}$$

d)

$f = x^T A x + b^T x$

$$\frac{\partial f}{\partial x_1} = \sum_{j=1}^{n} a_{1j} x_j + \sum_{i=1}^{n} a_{i1} x_i + b_1$$

$$= (Ax)_1 + (A^T x)_1 + b_1$$

$$\frac{\partial f}{\partial x_i} = (Ax)_i + (A^T x)_i + b_i$$

$$\nabla_x f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix} = \begin{bmatrix} (Ax)_1 + (A^T x)_1 + b_1 \\ (Ax)_2 + (A^T x)_2 + b_2 \\ \vdots \\ (Ax)_n + (A^T x)_n + b_n \end{bmatrix} = Ax + A^T x + b$$

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

e)

$f = tr(AB)$

want $\nabla_A f$

$$A = \overset{n \times m}{\begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nm} \end{bmatrix}} \qquad B = \overset{m \times n}{\begin{bmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & \ddots & \vdots \\ b_{m1} & \cdots & b_{mn} \end{bmatrix}}$$

$$\begin{bmatrix} \vec{a}_1 \\ \vdots \\ \vec{a}_n \end{bmatrix} \qquad B = \begin{bmatrix} \vec{b}_1 & \cdots & \vec{b}_n \end{bmatrix}$$

$$tr(AB) = \vec{a}_1 \vec{b}_1 + \vec{a}_2 \vec{b}_2 + \cdots \vec{a}_n \vec{b}_n$$

rows of a multiplied by columns of B

$$\frac{\partial f}{\partial A} = \begin{bmatrix} \frac{\partial f}{\partial a_{11}} & \frac{\partial f}{\partial a_{12}} & \cdots \\ \frac{\partial f}{\partial a_{21}} & \ddots & \\ \vdots & & \frac{\partial f}{\partial a_{nm}} \end{bmatrix} = \begin{bmatrix} b_{11} & b_{21} & \\ b_{12} & \ddots & \\ & & b_{mn} \end{bmatrix}$$

$$\boxed{\nabla_A f = B^T}$$

4 Problem 4

4.1 Problem Statement

4. (10 points) **Deriving least-squares with matrix derivatives.**
In least-squares, we seek to estimate some multivariate output $\mathbf{y}$ via the model

$$\hat{\mathbf{y}} = \mathbf{W}\mathbf{x}$$

In the training set we're given paired data examples $(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})$ from $i = 1, \ldots, n$. Least-squares is the following quadratic optimization problem:

$$\min_{\mathbf{W}} \quad \frac{1}{2}\sum_{i=1}^{n} \left\| \mathbf{y}^{(i)} - \mathbf{W}\mathbf{x}^{(i)} \right\|^2$$

Derive the optimal $\mathbf{W}$.

Hint: you may find the following derivatives useful:

$$\begin{aligned} \frac{\partial \mathrm{tr}(\mathbf{W}\mathbf{A})}{\partial \mathbf{W}} &= \mathbf{A}^T \\ \frac{\partial \mathrm{tr}(\mathbf{W}\mathbf{A}\mathbf{W}^T)}{\partial \mathbf{W}} &= \mathbf{W}\mathbf{A}^T + \mathbf{W}\mathbf{A} \end{aligned}$$

4.2 Solution

1. Want to estimate y with model:

$$\hat{y} = Wx$$

given data $x^{(i)}, y^{(i)}$

least-squared optimization:

$$\min_W \frac{1}{2} \sum_{i=1}^{n} \| y^{(i)} - Wx^{(i)} \|^2$$

write in matrix/vector form

$$\mathcal{L} = \frac{1}{2} \sum_{i=1}^{N} (y^{(i)} - Wx^{(i)})^2$$

$$= \frac{1}{2} \sum_{i=1}^{N} (y^{(i)} - Wx^{(i)})^T (y^{(i)} - Wx^{(i)})$$

$$= \frac{1}{2} (Y - WX)^T (Y - WX)$$

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

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

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

$$\frac{\partial \mathcal{L}}{\partial W} = \frac{1}{2} \left[0 - 2YX^T + 2WXX^T \right]$$

$$= -YX^T + WXX^T = 0$$

$$WXX^T = YX^T$$

$$\boxed{\begin{array}{l} W = YX^T \underbrace{(XX^T)^{-1}}_{\text{right pseudo-inverse}} \\ W = YX^+ \end{array}}$$

5 Problem 5

5.1 Problem Statement

5. (30 points) **Hello World in Jupyter.**
Complete the Jupyter notebook `linear_regression.ipynb`. Print out the Jupyter notebook and submit it to Gradescope.

5.2 Solution

linear_regression

January 18, 2021

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 2021, Prof. J.C. Kao, TAs: N. Evirgen, A. Ghosh, S. Mathur, T. Monsoor, G. Zhao

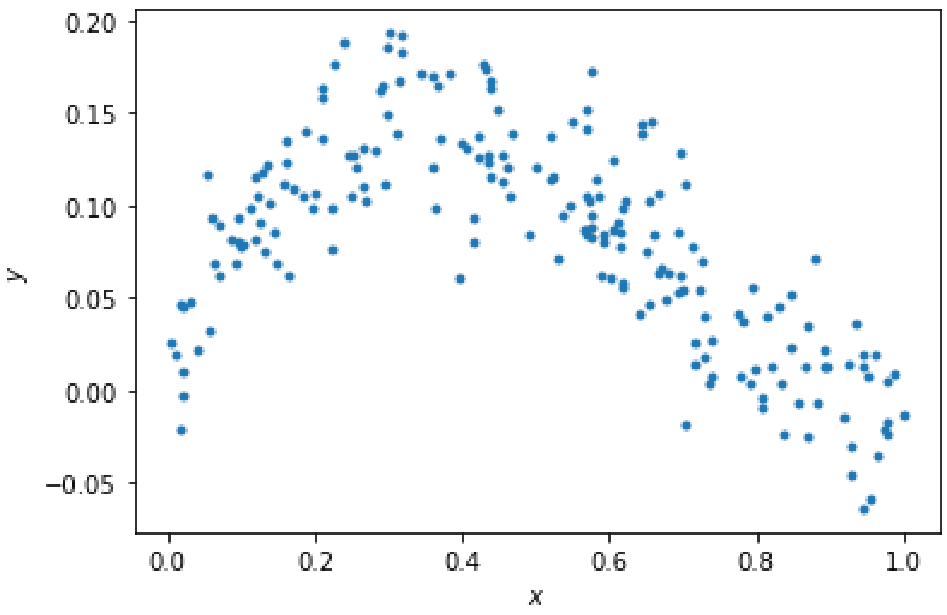
[163]:
```
import numpy as np
import matplotlib.pyplot as plt

#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: $y = x - 2x^2 + x^3 + \epsilon$

[164]:
```
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('$y$')
```

[164]:
```
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) It is a uniform distribution with max height 1

(2) It is a normal distribution with std = 0.03, mean = 0.

0.1.4 Fitting data to the model (5 points)

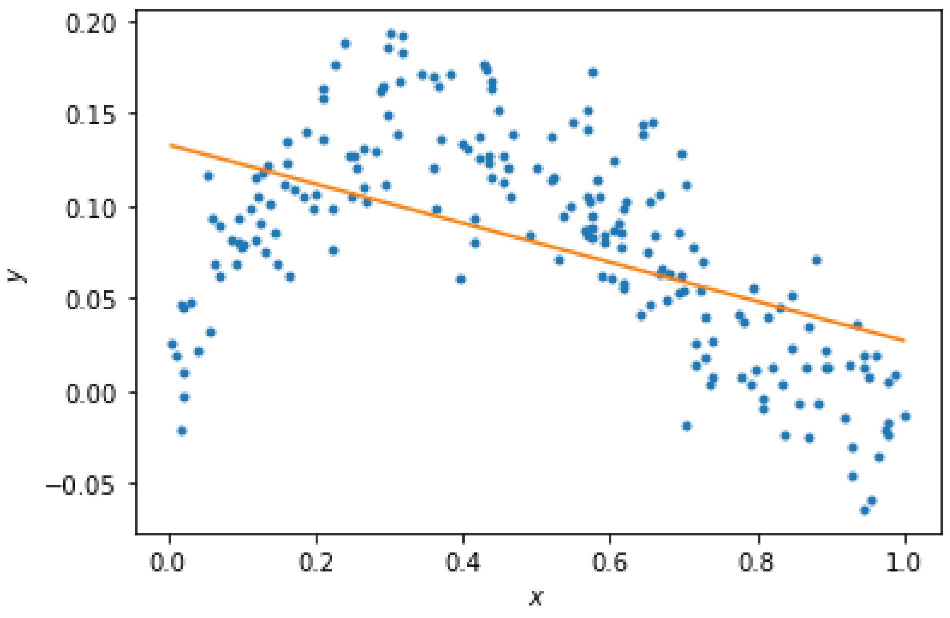
Here, we'll do linear regression to fit the parameters of a model $y = ax + b$.

[165]:
```
# 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.zeros(2) # please modify this line
theta = np.linalg.inv(xhat.dot(xhat.T)).dot(xhat.dot(y))

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

[166]:
```
# 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))
```

[166]:
```
[<matplotlib.lines.Line2D at 0x145bd1020b8>]
```



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) The linear model under fits the data.

(2) You can increase the model complexity to improve the fitting. ex. ax^2+bx+c

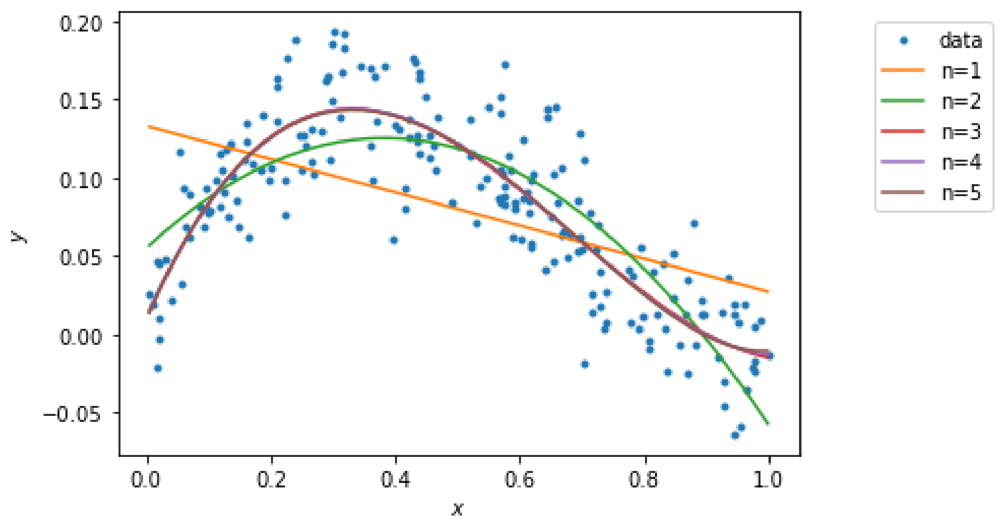
0.1.7 Fitting data to the model (10 points)

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.

[167]:
```
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.
for i in np.arange(N):
    if i == 0:
        xhats.append(xhat)
        thetas.append(theta)
    else:
        xhat = np.vstack((x**(i+1),xhat))
        xhats.append(xhat)
        thetas.append(np.linalg.inv(xhats[i].dot(xhats[i].T)).dot(xhats[i].
↪dot(y)))

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

[168]:
```
# 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 (10 points)

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

[169]:
```
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 np.arange(N):
    training_errors.append(np.sqrt(np.sum((y - thetas[i].dot(xhats[i]))**2)/␣
 ↪len(y)))

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

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

```
Training errors are:
 [0.04878484486357111, 0.03305287008607351, 0.028582518785273934,
0.028575083088762807, 0.028568302706890546]
```

0.1.9 QUESTIONS

(1) What polynomial has the best training error?

(2) Why is this expected?

0.1.10 ANSWERS

(1) The polynomial degree n = 5 has the best training error

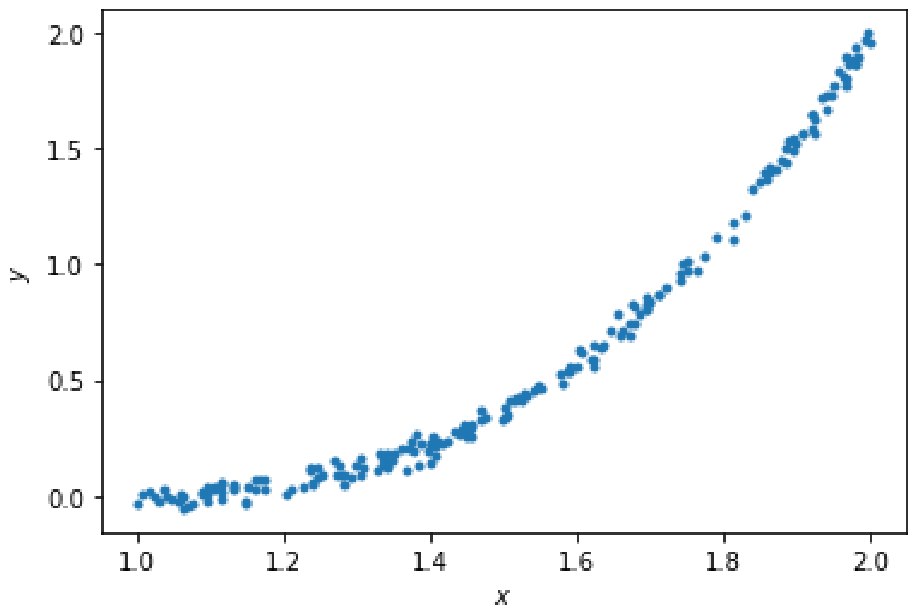
(2) This is expected because the model has more parameters to tune in order to match the data.

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.

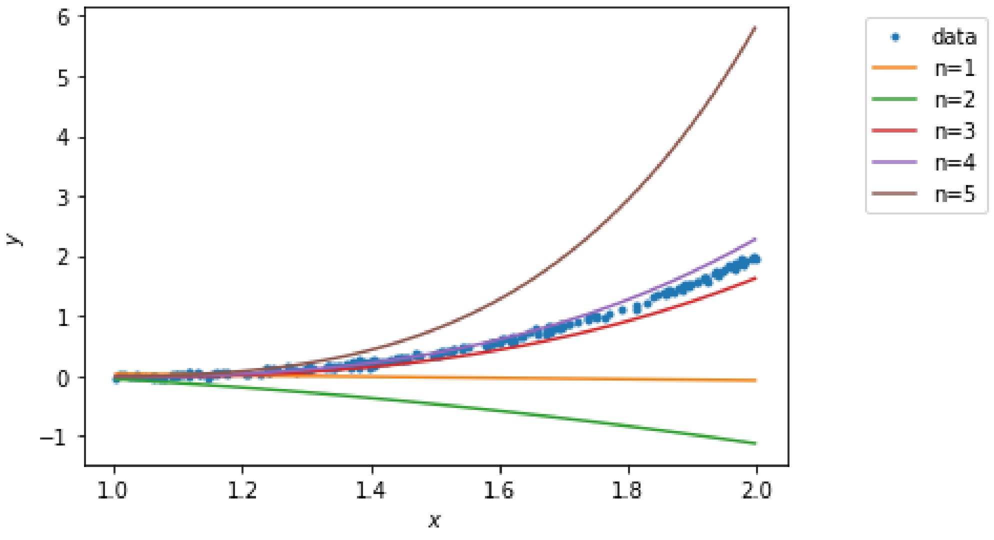
```
[170]: x = np.random.uniform(low=1, high=2, size=(num_train,))
       y = x - 2*x**2 + x**3 + np.random.normal(loc=0, scale=0.03, size=(num_train,))
       f = plt.figure()
       ax = f.gca()
       ax.plot(x, y, '.')
       ax.set_xlabel('$x$')
       ax.set_ylabel('$y$')
```

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



[171]:
```
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)
```

[172]:
```
# 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)
```



[173]:
```
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 np.arange(N):
    testing_errors.append(np.sqrt(np.sum((y - thetas[i].dot(xhats[i]))**2)/␣
 ↪len(y)))

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

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

```
Testing errors are:
 [0.8992310706681896, 1.460109326216975, 0.176796411396736, 0.10895304121260087,
1.4659816444162126]
```

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) The polynomial with model order n = 4 has the beset testing error.

(2) The polynomial model of order 5 does not generalize well because the data was overfit. This means the model does a good job tracking the training data but is not general enough to do well with

```
[ ]:
```